МИНИСТЕРСТВА ВЫСШЕГО И СРЕДНЕЕ СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ УЗБЕКИСТАН
ТАШКЕНТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

КАФЕДРА “АВТОМАТИЗАЦИЯ ПРОИЗВОДСТВЕННЫХ ПРОЦЕССОВ”

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ПО ПРЕДМЕТУ

«Интеллектуальные системы управление и принятые решение»

Область знаний	**300 000**	– Производственно-техническая сфера
Область образования	**310 000**	– Инженерное дело
Направление образования	**5311000**	– Автоматизация и управление технологических процессов и производств

ТАШКЕНТ - 2017 г

Разработана на основе утвержденной приказом Министерства высшего и среднее специального образования № ___ от _____ 2017 года типовой учебной программы по предмету “Методика преподавание специальных дисциплин”

Составил:

Н.Р.Юсупбеков – академик кафедры “Автоматизация производственных процессов” .

Учебно-методический комплекс обсуждена и одобрена на заседание кафедры «Автоматизация производственных процессов» (Протокл № ____ от _____ 2017 г.).

Зав. кафедрой: **_____** **А.Н.Юсупбеков**

Учебно-методический комплекс обсуждена и одобрена на заседание Электроника и автоматика факультета (Протокол № ____ от _____ 2017 г).

Декан Электроника и автоматика факультета: Зикриллаев Х.Ф.

Учебно-методический комплекс обсуждена и одобрена на заседание учебно-методического совета ТГТУ (№ ___ Протокол (№ ____, _____ 2017 г.)

Секретарь учебно-методического совета: Мамбетов Н.

СОДЕРЖАНИЕ

ВОПРОСНИК ДЛЯ ТК, ПК И и ИК	
РЕЙТИНГ ПО ДИСЦИПЛИНЕ	
ТЕСТОВЫЕ ЗАДАНИЯ ПО ДИСЦИПЛИНЕ	
НОРМАТИВНЫЕ ДОКУМЕНТЫ	
ГЛОССАРИЙ ПО ДИСЦИПЛИНЕ	
ПРИЛОЖЕНИЕ	

ЎЗБЕКИСТОН РЕСПУБЛИКАСИ ОЛИЙ ВА ЎРТА МАХСУС ТАЪЛИМ ВАЗИРЛИГИ

Рўйхатга олинди

№ HD - 5A 5238 - 2.2.02

2011 йил «22» 03

Ўзбекистон Республикаси Олий ва ўрта махсус таълим вазирлигининг 2011 йил «4» 05 даги 192-сонли буйруғи билан тасдиқланган

БОШҚАРИШНИНГ ИНТЕЛЛЕКТУАЛ ТИЗИМЛАРИ ВА ҚАРОР ҚАБУЛ ҚИЛИШ

фанининг

ЎҚУВ ДАСТУРИ

Билим соҳаси:	300 000 –	Ишлаб чиқариш техник соҳа
Таълим соҳаси:	310 000 –	Муҳандислик иши
Таълим йўналиши:	5311000 –	Технологик жараёнлар ва ишлаб чиқаришни автоматлаштириш ва бошқариш (тармоқлар бўйича)
Таълим мутахассислиги:	5А311001–	Технологик жараёнлар ва ишлаб чиқаришни автоматлаштириш (тармоқлар бўйича)

Тошкент – 2011

Фаннинг ўқув дастури Олий ва ўрта махсус, касб – ҳунар таълими ўқув методик бирлашмалари фаолиятини Мувофиқлаштирувчи Кенгашнинг 201*1* йил «__» _____ даги «___» - сон мажлис баёни билан маъқулланган.

Фаннинг ўқув дастури Тошкент давлат техника университетида ишлаб чиқилди.

Тузувчилар: Тошкент Давлат техника университети «Ишлаб чиқариш жараёнларини автоматлаштириш» кафедрасининг мудири, ЎзР ФА академиги, т.ф.д., проф. Юсупбеков Н.Р.

Россия Федерацияси Интеллектуал мулк ва бизнесни баҳолаш ва сертификатлаш институтининг бош мутахасиси, т.ф.д., проф. Мамаджанов Х.А.

Тақризчилар: Тошкент кимё технология институти «Информатика. Автоматлаштириш ва бошқарув» кафедраси профессори, т.ф.д. Артиков А.А.

«Ўзкимёсаноат» ДАК нинг бош мутахассиси т.ф.д., проф. Юсипов М.М.

Фаннинг ўқув дастури Тошкент давлат техника университети Илмий – услубий Кенгашида тавсия қилинган (201*1* йил «__» ___ даги «__» сонли баённома).

КИРИШ

«Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш» ўқув фани 5А311001 - «Технологик жараёнларни ва ишлаб чикаришни автоматлаштириш» мутахассислиги бўйича магистрлар тайёрлашнинг мутахассислик ўкув фанлари қаторига киритилган.

Системаларнинг янги авлоди – интеллектуал системалар (ИС) системалар компонентларини ташкил қилишнинг янги принципларини дунёга келтирди, илгариги ишланмаларда, шунингдек илмий адабиётларда учрамайдиган бошқа бир тушунчалар, атамалар, блоклар пайдо бўлди. Бошқаришда интеллектуал системлар мақсадни синтезлаш, ҳаракатларга нисбатан қарор қабул қилиш, мақсадга эришиш учун бўладиган ҳаракатларни таъминлаш, ҳаракат натижалари параметрларининг қийматларини башоратлаш ва уларни тескари алоқа ҳосил қилган ҳолда реаллари билан солиштириш, мақсадлар ёки бошқарувни тўғрилаш имконига эга.

Ўқув фанининг мақсади ва вазифалари

Ўқув фанининг мақсади – талабаларда бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш хақида кўникмалар хосил қилиш, уларга интеллектуал системалар хусусиятлари ва уларни қуриш усулларини ўргатишдан иборат.

Ўқув фанининг вазифаси – талабаларда бошқаришда интеллектуал системаларнинг хусусиятлари ва уларни қуриш усулларини танлаш кўникмаларини ҳосил қилишдан иборат.

Фан бўйича билим, малака ва кўникмага қўйиладиган талаблар

«Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш» фанини ўзлаштириш давомида магистрант:

- динамик эксперт тизимлари (ДЭТ) ҳақида;
- мақсадларни синтезлаш усуллар ҳақида;
- интеллектуал системалар таъсир натижаларини қандай башоратлаши ва бошқарувни ишлаб чиқиши ҳақида;
- реал вақтларда реал дунёда фаолият кўрсатувчи ДЭС ни қуриш асослари *ҳақида тасаввурга эга бўлиши;*
- маълумотлар ва билимлар асосида мақсад қандай синтезланиши ва датчиклар тизимидан олинган маълумотларни фаол баҳолаш асосида ҳаракатларни амалга ошириш учун қарор қандай қабул қилинишини;
- атроф муҳит ва интеллектуал системанинг хусусий ҳолатлари ҳақидаги жорий маълумотларга асосланиб, мақсад ва билимлар мавжуд бўлганда экспертли баҳолашнинг қандай амалга оширилишини *билиши;*

– интеллектуал системалар фаолиятини баҳолай олиш ва уларни бошқалари билан солиштира олиш *кўникмаларига эга бўлиши*;

– интеллектуал системалар тўғрисидаги назарий билимларини кенгайтириши, тизимлаштириши ва чуқурлаштириши;

– интеллектуал системаларни ташкил этиш тўғрисидаги таклифларни *ишлаб чиқа олиши, таҳлил қила олиши ва қарор қабул қила олиш;*

– интеллектуал системалар билан фаолият юрита олишнинг *амалий кўникмаси*

ва тажрибаларига эга бўлиши керак.

Қўйилган вазифалар ўқиш жараёнида талабаларни маъруза ва амалий машғулотларда фаол иштирок этиши, адабиётлар билан мустақил ишлаши ва ўқитувчи кузатувида мустақил таълим олиши билан амалга оширилади.

Фаннинг ўқув режадаги бошқа фанлар билан ўзаро боғлиқлиги ва услубий жиҳатдан узвий кетма-кетлиги

«Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш» фани мутахассислик фани ҳисобланиб, 3-семестрда ўқитилади. Дастурни амалга ошириш ўқув режасида режалаштирилган «Оптимал ва адаптив бошқариш системалари» ҳамда бакалавриат ихтисослик фанларидан етарли билим ва кўникмаларга эга бўлиш талаб этилади.

Фаннинг ишлаб чиқаришдаги ўрни

Кимё саноати корхоналаридаги мураккаб технологик жараёнларни бошқаришда интеллектуал системаларни қўллаш бўйича олиб борилаётган ишлар умумий ҳажмнинг анчагина қисмини ташкил қилади.

Шунинг учун ҳам Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилишни алоҳида талаблар қўйилади. Айниқса, бугунги кунда кимё саноат корхоналари ва илмий тадқиқот ишларида интеллектуал системалардан фойдаланиш куннинг долзарб масалаларидан бири ҳисобланади. Шунинг учун ушбу фан асосий мутахассислик фани ҳисобланиб, корхона ва ишлаб чиқаришларни ташкил этишнинг ажралмас бўғинидир.

Фанни ўқитишда замонавий ахборот ва педогогик технологиялар

Талабаларнинг «Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш» фанини ўзлаштиришлари учун ўқитишнинг илғор ва замонавий усулларидан фойдаланиш, янги информацион-педагогик технологияларни тадбиқ қилиш муҳим аҳамиятга эгадир. Фанни ўзлаштиришда дарслик, ўқув ва услубий қўлланмалар, маъруза матнлари, тарқатма материаллар, электрон материаллар, виртуал стендлар ҳамда намуналар ва макетлардан фойдаланилади. Маъруза ва амалий дарсларда мос равишдаги илғор педагогик технологиялардан фойдаланилади.

Асосий қисм

Фаннинг назарий машғулотлари мазмуни

Датчиклар тизимидан олинган ахборотларни баҳолаш. Мақсадни синтез қилиш ва ҳаракатни (ишлашни) бошлаш учун қарор қабул қилиш. Ахборотларни фаол баҳолаш. Ҳаракат натижаларини башоратлаш ва бошқарувни ишлаб чиқиш. Бошқарувни физик сигналларга ўзгартириш.

Тескари боғланиш занжири. Эришилган натижаларни эмоционал баҳолаш. Бошқарувни коррекциялаш (тўғрилаш). Янги мақсадларни синтез қилиш ва уларга эриишини ташкил этиш.

Динамик эксперт тизимлар. Динамик тизимларни тавсифлашнинг усуллари. Тушуниш жараёнларида тушунча, моҳият ва ақлий хулосаларни акс эттириш,

шакллантиришларни амалга оширувчи интеллектуал операторлар.. ДЭС масалаларини ечиш усуллари.

Эксперт тизимларига берилдиган билимлар: концептуал билимлар (тушунча даражасидаги); фактик, предметли билимлар; алгоритмик, процедурали билимлар.

Интеллектуал системаларда билимлар базасини яратиш ва қўллаш.

Тенгламаларни ечиш учун амалий дастурлар ва алгоритмлар. Маълумотлар базаси. Масалаларни ечиш босқичлари: ечишнинг абстракт дастурларини тузиш (масалалнинг юзага келиши, унинг қўйилиши ва спецификацияси ҳам киради); масалани мос тушувчи машина тилига таржима қилиш; трансляция ва дастурларни бажариш.

Ягона дастурий муҳит яратиш ва масаланинг қўйилиши бўйича бевосита алгоритмларни синтезлаш.

Амалий машғулотларни ташкил этиш бўйича кўрсатмалар

Амалий машғулотларда талабалар маърузаларда ўрганилган назарий билимларини бойитадилар ва мустаҳкамлайдилар. Амалий машғулотларни қуйидаги мавзуларда олиб бориш тавсия этилади:

- ДЭС нинг турлари;
- биринчи тур ДЭС нинг структураси, иккинчи тур ДЭС нинг структураси, учинчи турдаги ҳисоб-мантиқий ДЭС;
- ДЭС ларни ишлаб чиқишда юзага келадиган муаммолар: маълумотлар базаси таркибини аниқлаш ва уни шакллантириш;
- ИС даги информацион жараёнларни тавсифлаш учун маълум ишлатилаётган назария ва усулларнинг янгиларини ишлаб чиқиш;
- билимлардан фойдаланишни ташкил этиш ва акс эттириш усулларини ишлаб чиқиш;
- “эгилувчан мантиқ” дан фойдаланиш ва параллел фойдаланиш орқали алгоритмлар ва дастурий таъминотни ишлаб чиқиш;
- ДЭС ни шакллантиришда параллел алгоритмларни амалга ошириш учун мос келувчи ҳисоблаш воситаларини қидириш.

Амалий машғулотларни ташкил этиш бўйича кафедра профессор-ўқитувчилари томонидан кўрсатма ва тавсиялар ишлаб чиқилади. Унда талабалар асосий маъруза мавзулари бўйича олган билим ва кўникмаларини амалий масалалар ечиш орқали янада бойитадилар. Шунингдек, дарслик ва ўқув қўлланмалар асосида талабалар билимларини мустаҳкамлашга эришиш, тарқатма материаллардан фойдаланиш, илмий мақолалар ва тезисларни чоп этиш орқали билимини ошириш, масалалар ечиш, мавзулар бўйича кўргазмали қуроллар тайёрлаш ва бошқалар тавсия этилади.

Мустақил ишни ташкил этишнинг шакли ва мазмуни

Талаба мустақил ишни тайёрлашда муайян фаннинг хусусиятларини ҳисобга олган ҳолда қуйидаги шакллардан фойдаланиши тавсия этилади:

- дарслик ва ўқув қўлланмалар бўйича фанларнинг боблари ва мавзуларини ўрганиш;
- тарқатма материаллар бўйича маърузалар қисмини ўзлаштириш;
- автоматлаштирилган ўргатувчи ва назорат қилувчи тизимлар билан ишлаш;
- махсус адабиётлар бўйича фанлар бўлимлари ёки мавзулари устида ишлаш;

- янги техникаларни, аппаратураларни, жараён ва технологияларни ўрганиш;
- талабаларнинг ўқув – илмий - тадқиқот ишларини бажариш билан боғлиқ бўлган фанлар бўлимлари ва мавзуларни чуқур ўрганиш;
- фаол ва муаммоли ўқитиш услубидан фойдаланиладиган ўқув машғулотлари;
- масофавий (дистанцион) таълим.

Тавсия этилаётган мустақил ишларнинг мавзулари:

- интеллектуал системалар ёрдамида робастли ва адаптив бошқарувларни умумлаштириш;
- ноаниқлик шароитларида фаолият кўрсатувчи реал мураккаб объектларни бошқаришни лойиҳалаш;
- адаптив ва робаст бошқаришдан фойдаланувчи ва сунъий интеллект усуллари асосида бошқарув турини танлашни амалга оширувчи системаларни лойиҳалаш;
- адаптив бошқариш системалари;
- интеллектуал блокли системаларни қуриш;
- комбинирлашган бошқариш системаларини қуриш.

Дастурнинг информацион-услубий таъминоти

Мазкур фанни ўқитиш жараёнида таълимнинг замонавий методлари, педогогик ва ахборот-коммуникация технологиялари қўлланилиши назарда тутилган:

- Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилишнинг назарий бўлимига тегишли маъруза дарсларида замонавий компютер технологиялари ёрдамида презентацион ва электрон-дидактик технологиялари;

- Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш бўйича ўтказиладиган амалий машғулотларда ақлий ҳужум, гуруҳли фикрлаш педагогик технологияларини қўллаш назарда тутилади.

Фойдаланилаётган асосий дарсликлар ва ўқув қўлланмалар рўйхати

Асосий

1. Алиев Р.А., Абдикеев Н.М. и др. Производственные системы с искусственным интеллектом.- М.: Радио и связь, 1990.
2. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем.- СПб: Питер, 2000.
3. Искусственный интеллект: В 3-х кн.: Справочник. - М.: Радио и связь, 1990.
4. Искусственный интеллект: применение в интегрированных производственных системах / Под ред. Э.Кьюсиака.- М.: Машиностр., 1991

Қўшимча

1. Чикул В.М. Основы искусственного интеллекта.- М.: Диалог МГУ, 2000.
2. Negnevtvitsky M. Artificial Intelligence: a Guide to Intelligence Systems. - Addisson Wesley, 2002.
3. Интеллектуализация ЭВМ / Е.С.Кузин и др.- М.: Высш. шк, 1989.
4. Интернет манбалари.

ИШЧИ ДАСТУР

КИРИШ

«Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш» ўқув фани 5А311001 «Технологик жараёнларни ва ишлаб чикаришни автоматлаштириш» мутахассислиги бўйича магистрлар тайёрлашнинг мутахассислик ўкув фанлари қаторига киритилган.

Системаларнинг янги авлоди – интеллектуал системалар (ИС) системалар компонентларини ташкил қилишнинг янги принципларини дунёга келтирди, илгариги ишланмаларда, шунингдек илмий адабиётларда учрамайдиган бошқа бир тушунчалар, атамалар, блоклар пайдо бўлди. Интеллектуал системлар мақсадни синтезлаш, ҳаракатларга нисбатан қарор қабул қилиш, мақсадга эришиш учун бўладиган ҳаракатларни таъминлаш, ҳаракат натижалари параметрларининг қийматларини башоратлаш ва уларни тескари алоқа ҳосил қилган ҳолда реаллари билан солиштириш, мақсадлар ёки бошқарувни тўғрилаш имконига эга.

Ўқув фанининг мақсади ва вазифалари

Ўқув фанининг мақсади – магистрларда Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш хақида кўникмалар хосил қилиш, уларга интеллектуал системалар хусусиятлари ва уларни қуриш усулларини ўргатишдан иборат.

Ўқув фанининг вазифаси – магистрларда интеллектуал системаларнинг хусусиятлари ва уларни қуриш усулларини танлаш кўникмаларини ҳосил қилишдан иборат.

Фан бўйича билим, малака ва кўникмага қўйиладиган талаблар

«Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш» фанини ўзлаштириш давомида магистрант:

- динамик эксперт тизимлари (ДЭТ) ҳақида;
- мақсадларни синтезлаш усуллар ҳақида;
- интеллектуал системалар таъсир натижаларини қандай башоратлаши ва бошқарувни ишлаб чиқиши ҳақида;
- реал вақтларда реал дунёда фаолият кўрсатувчи ДЭС ни қуриш асослари *ҳақида тасаввурга эга бўлиши;*
- маълумотлар ва билимлар асосида мақсад қандай синтезланиши ва датчиклар тизимидан олинган маълумотларни фаол баҳолаш асосида ҳаракатларни амалга ошириш учун қарор қандай қабул қилинишини;
- атроф муҳит ва интеллектуал системанинг хусусий ҳолатлари ҳақидаги жорий маълумотларга асосланиб, мақсад ва билимлар мавжуд бўлганда экспертли баҳолашнинг қандай амалга оширилишини *билиши;*

– интеллектуал системалар фаолиятини баҳолай олиш ва уларни бошқалари билан солиштира олиш *кўникмаларига эга бўлиши*;

– интеллектуал системалар тўғрисидаги назарий билимларини кенгайтириши, тизимлаштириши ва чуқурлаштириши;

– интеллектуал системаларни ташкил этиш тўғрисидаги таклифларни *ишлаб чиқа олиши, таҳлил ва қарор қабул қила олиши;*

– интеллектуал системалар билан фаолият юрита олишнинг *амалий кўникмаси ва тажрибаларига эга бўлиши керак*.

Қўйилган вазифалар ўқиш жараёнида талабаларни маъруза ва амалий машғулотларда фаол иштирок этиши, адабиётлар билан мустақил ишлаши ва ўқитувчи кузатувида мустақил таълим олиши билан амалга оширилади.

Фаннинг ўқув режадаги бошқа фанлар билан ўзаро боғлиқлиги ва услубий жиҳатдан узвий кетма-кетлиги

«Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш» фани мутахассислик фани ҳисобланиб, 3-семестрда ўқитилади. Дастурни амалга ошириш ўқув режасида режалаштирилган «Оптимал ва адаптив бошқариш системалари» ҳамда бакалавриат ихтисослик фанларидан етарли билим ва кўникмаларга эга бўлиш талаб этилади.

Фаннинг ишлаб чиқаришдаги ўрни

Кимё саноати корхоналаридаги мураккаб технологик жараёнларни бошқаришда интеллектуал системаларни қўллаш бўйича олиб борилаётган ишлар умумий ҳажмнинг анчагина қисмини ташкил қилади.

Шунинг учун ҳам Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилишни алоҳида талаблар қўйилади. Айниқса, бугунги кунда кимё саноат корхоналари ва илмий тадқиқот ишларида интеллектуал системалардан фойдаланиш куннинг долзарб масалаларидан бири ҳисобланади. Шунинг учун ушбу фан асосий мутахассислик фани ҳисобланиб, корхона ва ишлаб чиқаришларни ташкил этишнинг ажралмас бўғинидир.

Фанни ўқитишда замонавий ахборот ва педогогик технологиялар

Магистрларнинг «Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш» фанини ўзлаштиришлари учун ўқитишнинг илғор ва замонавий усулларидан фойдаланиш, янги информацион-педагогик технологияларни тадбиқ қилиш муҳим аҳамиятга эгадир. Фанни ўзлаштиришда дарслик, ўқув ва услубий қўлланмалар, маъруза матнлари, тарқатма материаллар, электрон материаллар, виртуал стендлар ҳамда намуналар ва макетлардан фойдаланилади. Маъруза ва амалий дарсларда мос равишдаги илғор педагогик технологиялардан фойдаланилади.

Асосий қисм
Фаннинг назарий машғулотлари мазмуни – 30 с

№	Мавзу	соат
1.	Датчиклар тизимидан олинган ахборотларни баҳолаш	2
2.	Мақсадни синтез қилиш ва ҳаракатни (ишлашни) бошлаш учун қарор қабул қилиш	2
3.	Ахборотларни фаол баҳолаш	2
4.	Ҳаракат натижаларини башоратлаш ва бошқарувни ишлаб чиқиш	2
5.	Бошқарувни физик сигналларга ўзгартириш	2

6.	Тескари боғланиш занжири	2
7.	Эришилган натижаларни эмоционал баҳолаш	2
8.	Бошқарувни коррекциялаш (тўғрилаш)	2
9.	Янги мақсадларни синтез қилиш ва уларга эриишини ташкил этиш	2
10.	Динамик эксперт тизимлар	2
11.	Динамик тизимларни тавсифлашнинг усуллари	2
12.	Тушуниш жараёнларида тушунча, моҳият ва ақлий хулосаларни акс эттириш, шакллантиришларни амалга оширувчи интеллектуал операторлар	2
13.	ДЭС масалаларини ечиш усуллари	2
14.	Эксперт тизимларига берилдиган билимлар концептуал билимлар (тушунча даражасидаги); фактик, предметли билимлар; алгоритмик, процедурали билимлар.	2
15.	Интеллектуал системаларда билимлар базасини яратиш ва қўллаш	2
Жами		**30**

Амалий машғулотлар – 30 с

Амалий машғулотларда талабалар маърузаларда ўрганилган назарий билимларини бойитадилар ва мустаҳкамлайдилар.

Амалий машғулотларни мавзулар:

№	Мавзу	соат
1	ДЭС нинг турлари	4
2	Биринчи тур ДЭС нинг структураси, иккинчи тур ДЭС нинг структураси, учинчи турдаги ҳисоб-мантиқий ДЭС	4
3	ДЭС ларни ишлаб чиқишда юзага келадиган муаммолар: маълумотлар базаси таркибини аниқлаш ва уни шакллантириш	4
4	ИС даги информацион жараёнларни тавсифлаш учун маълум ишлатилаётган назария ва усулларнинг янгиларини ишлаб чиқиш	4
5	Билимлардан фойдаланишни ташкил этиш ва акс эттириш усулларини ишлаб чиқиш	4
6	“Эгилувчан мантиқ” дан фойдаланиш ва параллел фойдаланиш орқали алгоритмлар ва дастурий таъминотни ишлаб чиқиш	4
7	ДЭС ни шакллантиришда параллел алгоритмларни амалга ошириш учун мос келувчи ҳисоблаш воситаларини қидириш	6
Жами		**30**

Мустақил иш – 32 с

Магистр мустақил ишни тайёрлашда муайян фаннинг хусусиятларини ҳисобга олган ҳолда қуйидаги шакллардан фойдаланиши тавсия этилади:

- дарслик ва ўқув қўлланмалар бўйича фанларнинг боблари ва мавзуларини ўрганиш;
- тарқатма материаллар бўйича маърузалар қисмини ўзлаштириш;
- автоматлаштирилган ўргатувчи ва назорат қилувчи тизимлар билан ишлаш;

- махсус адабиётлар бўйича фанлар бўлимлари ёки мавзулари устида ишлаш;
- янги техникаларни, аппаратураларни, жараён ва технологияларни ўрганиш;
- талабаларнинг ўқув – илмий - тадқиқот ишларини бажариш билан боғлиқ бўлган фанлар бўлимлари ва мавзуларни чуқур ўрганиш;
- фаол ва муаммоли ўқитиш услубидан фойдаланиладиган ўқув машғулотлари;
- масофавий (дистанцион) таълим.

Тавсия этилаётган мустақил ишларнинг мавзулари:

- интеллектуал системалар ёрдамида робастли ва адаптив бошқарувларни умумлаштириш;
- ноаниқлик шароитларида фаолият кўрсатувчи реал мураккаб объектларни бошқаришни лойиҳалаш;
- адаптив ва робаст бошқаришдан фойдаланувчи ва сунъий интеллект усуллари асосида бошқарув турини танлашни амалга оширувчи системаларни лойиҳалаш;
- адаптив бошқариш системалари;
- интеллектуал блокли системаларни қуриш;
- комбинирлашган бошқариш системаларини қуриш.

Дастурнинг информацион-услубий таъминоти

Мазкур фанни ўқитиш жараёнида таълимнинг замонавий методлари, педогогик ва ахборот-коммуникация технологиялари қўлланилиши назарда тутилган:

- Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилишнинг назарий бўлимига тегишли маъруза дарсларида замонавий компютер технологиялари ёрдамида презентацион ва электрон-дидактик технологиялари;

- Бошқаришнинг интеллектуал тизимлари ва қарор қабул қилиш бўйича ўтказиладиган амалий машғулотларда ақлий ҳужум, гуруҳли фикрлаш педагогик технологияларини қўллаш назарда тутилади.

Фойдаланилаётган асосий дарсликлар ва ўқув қўлланмалар рўйхати

Асосий

1. Алиев Р.А., Абдикеев Н.М. и др. Производственные системы с искусственным интеллектом.- М.: Радио и связь, 1990.
2. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем.- СПб: Питер, 2000.
3. Искусственный интеллект: В 3-х кн.: Справочник. - М.: Радио и связь, 1990.
4. Искусственный интеллект: применение в интегрированных производственных системах / Под ред. Э.Кьюсиака.- М.: Машиностр., 1991
5. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001. – 576 С.

6. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002.- 744 с.,ил.
7. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998.
8. - 427 с.
9. Поспелов Д. А. Логико-лингвистические модели в системах управления -М.: Энергоиздат, 1981. -232 с.
10. Поспелов Д. А. Ситуационное управление: теория и практика. -М.: Наука, 1986, -288 с.

Қўшимча

1. Чикул В.М. Основы искусственного интеллекта.- М.: Диалог МГУ, 2000.
2. Negnevtvitsky M. Artificial Intelligence: a Guide to Intelligence Systems. - Addisson Wesley, 2002.
3. Интеллектуализация ЭВМ / Е.С.Кузин и др.- М.: Высш. шк, 1989.
4. Интернет манбалари.
5. Ярушкина Н.Г. Основы теории нечетких и гибридных систем Учебное пособие, - М.: Финансы и статистика, 2004, - 320 с.
6. Робототехника и ГАП. В 9-ти кн. Кн.6. Техническая имитация интеллекта: Учебное пособие для втузов /Под ред. И. М. Макарова, - М.: Высш. шк., 1986. -144 С.
7. Аверкин А. Н., Батыршин И. З., Блишун А. Ф., Силов В. Б., Тарасов В. Б. Нечеткие множества в моделях управления и искусственного интеллекта. - М.: Наука, 1986.
8. Лорьер Ж.-Л. Системы искусственного интеллекта. -М.: Мир, 1991,-356 с.
9. Заде Л. Лингвистическая переменная, -М.: Физматгиз, 1972.
10. Справочник по промышленной робототехнике: В 2-х кн. Кн.1/ Под ред Ш.Нофа; Пер. с англ. Д.Ф.Миронова и др. – М.: Машиностроения, 1989,- 480с.

ТАШКЕНТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

КАФЕДРА “АВТОМАТИЗАЦИЯ ПРОИЗВОДСТВЕННЫХ ПРОЦЕССОВ”

ЛЕКЦИОННЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЫ “МЕТОДИКА ПРЕПОДОВАНИЕ СПЕЦИАЛЬНЫХ ДИСЦИПЛИН”

ТАШКЕНТ – 2017

Лекция 1
Тема:Оценка информации полученной от систем датчиков
План:

1. Интерес к интеллектуальным системам управления (ИСУ)
2. Базовые интеллектуальные технологии
3. Машинная форма представления информации
4. Практическая реализация концепции ситуационного управления на основе современных интеллектуальных технологий

1. Интерес к интеллектуальным системам управления (ИСУ)

Во всем мире разработкам технологий интеллектуального управления уделяется большое внимание. Анализ публикаций показывает, что выполняемые в России и за рубежом научно-исследовательские программы, комплексные проекты и отдельные работы по интеллектуальному управлению можно сгруппировать следующим образом:

—интеллектуальное управление промышленными объектами и производственными системами,

—создание систем интеллектуального управления подвижными объектами различного назначения и транспортными средствами наземного, подводного и воздушного базирования,

—разработка средств и методов управления интеллектуальными роботами специального, промышленного, медицинского, бытового и других применений,

—разработка и создание специализированных аппаратных средств для систем интеллектуального управления

Интерес к интеллектуальным системам управления (ИСУ) объясняется рядом причин. Первая из них состоит в том, что традиционные технологии уже не могут обеспечить повышение качества управления, поскольку не учитывают всех неопределенностей, воздействующих на систему. Совершенствование известных алгоритмов адаптивного управления не всегда дает желаемый результат. Это объясняется как сложностью самих алгоритмов, так и трудностями их реализации на цифровой технике с учетом условий обеспечения устойчивости дискретной системы управления

Второй причиной, способствующей интенсификации исследований в области интеллектуальных технологий управления, является наличие фундаментальной теоретической базы, коей являются работы Д.А.Поспелова, Л.Заде и других ученых. Используя результаты этих работ, в сочетании с пониманием теории управления, можно и нужно ожидать позитивных результатов в обоснованной интеллектуализации систем автоматического управления (САУ) на основе применения современных методов и технологий обработки знаний методов и технологий обработки знаний.

Третья причина связана с тем, что в настоящее время уже назрела целесообразность использования преимуществ интеллектуальных технологий управления. При этом можно и нужно говорить о реальности применения существующей элементной базы для создания определенных классов ИСУ, относительная простота которых связана с обработкой ограниченного набора знаний в конкретной предметной области. При этом естественно возникает и требует специального исследования целый комплекс вопросов: о составе и оптимальных объемах знаний, о выборе формы их представления и способах формирования и т.д. Проблема создания новой элементной базы, например, нейросетевых структур, нечетких контроллеров и т. д., специально ориентированных на поддержку интеллектуальных технологий обработки информации и управления, остается крайне актуальным и самостоятельным направлением исследований.

И, вероятно, последняя, четвертая причина связана с тем, что дальнейшее развитие интеллектуальных технологий управления как па исполнительном уровне (интеллектуальный привод, интеллектуальный мехатронный модуль и т. д.), так и на уровне организации целесообразных действий и поведения позволяет обеспечить создание принципиально нового поколения машин, обладающих высокими техническими характеристиками и функциональными возможностями.

2. Базовые интеллектуальные технологии

В качестве базовых выделяются четыре интеллектуальных технологии:

— технология экспертных систем, ориентированная на обработку знаний с явной формой представления в виде продукционных правил, семантических сетей, предикатов и фреймообразных структур;

—технология нечеткой логики, ориентированная па обработку логико-лингвистических моделей представления знаний с помощью продукционных правил и размытых множеств;

—технология нейросетевых структур с неявной формой представления знаний, скрытых в архитектуре сети, параметрах нейронов и связей;

— технология ассоциативной памяти, ориентированная на обработку знаний с неявной формой представления в виде гиперповерхности в многомерном пространстве признаков.

Научные исследования и разработка на их основе принципов построения интеллектуальных систем управления сложными динамическими объектами прежде всего предполагает необходимость ясного понимания специфики поставленной проблемы на уровне исходных содержательных понятий. В этой связи очевидный интерес представляет толкование термина "интеллектуальная система". Его устоявшаяся трактовка, приводимая в обширной научной литературе, состоит в том, что основной отличительной чертой интеллектуальных систем является возможность системной обработки. Отсутствие или ограниченность дальнейших пояснений в подавляющем большинстве библиографических источников обусловлены трудностями предельно строгого определения знаний, формулировка которого ввиду интуитивной понятности предмета обсуждения выносится за пределы рассмотрения.

3. Машинная форма представления информации

Одна из немногих попыток фундаментального определения знаний путем перечисления и анализа их свойств представлена в работах Д А. Поспелова и его коллег. В ряду особенностей, присущих этой машинной форме представления информации, выделяется пять важнейших элементов:

— *внутренняя интерпретируемость,* понимаемая как наличие уникальных имен, идентифицирующих каждую информационную единицу;

— *структурированность,* которая обусловливает возможность рекурсивной вложимости отдельных информационных единиц друг в друга,

— *внешняя связность,* задающая возможность установления функциональных, казуальных и других типов отношений между информационными единицами,

— *шкалируемость,* характеризующая возможность введения различных метрик для фиксации количественных, порядковых и иных соотношений информационных единиц;

— *активность,* отражающая способность инициировать выполнение некоторых целесообразных действий при появлении новой информации.

Данное определение при всей его логичности не дает ответа на вопрос о применимости тех или иных информационных технологий для обработки знаний и организа-

ции интеллектуального управления. В то же время существующий пробел может быть в определенной мере восполнен анализом соответствующих тематических материалов справочно-энциклопедических изданий, содержащих необходимый набор следующих взаимосвязанных определений.

Знания — проверенный практикой результат познания деятельности, верное ее отражение *в* мышлении человека.

Мышление — процесс отражения объективной действительности в представлениях, суждениях, понятиях.

Понятие — форма мышления, отражающая свойства, связи и отношения предметов и явлений. Основная логическая функция понятия — выделение общего, которая достигается посредством отвлечения от всех особенностей отдельных предметов данного класса. В логике — мысль, в которой обобщаются и выделяются предметы некоторого класса по определенным общим и в совокупности специфическим для них признакам.

Понятие — логически оформленная мысль о классе предметов.

Сопоставление приведенных определений позволяет сформулировать некоторую точку зрения о важнейшей отличительной особенности знаний как о свойстве отражения классификационной системы соподчиненных понятий, которая обобщает закономерности, действующие в какой-либо предметной области. Тогда определение знаний обеспечивает возможность вынесения однозначной оценки о принадлежности к разряду интеллектуальных по крайней мере четырех различных информационных технологий:

— технологии экспертных систем;

— технологии нечеткой логики;

— технологии нейросетевых структур;

— технологии ассоциативной памяти.

Так, главной отличительной особенностью *технологии экспертных систем* является возможность работы с формами явного представления знаний, включая продукционные правила, предикаты, семантические сети и фреймообразные структуры. Ярко выраженная структурированность этих форм обусловливает применимость формализованных логических методов для анализа и уточнения знаний, а также вывода заключений по совокупности исходных данных. При этом собственно процесс вывода на основе знаний сводится к последовательному сопоставлению заданного описания начальной посылки с категориями той многоуровневой классификации, которая заложена в имеющейся иерархии системы продукционных правил, семантических сетей или других представлений. По существу эта технология объединяет несколько близких направлений, поскольку принятые методы логической обработки для каждой из форм явного представления знаний существенно различаются между собой. С другой стороны, с точки зрения отражательных способностей по отношению к смысловой стороне знаний эти формы их представления по некоторым оценкам считаются взаимозаменяемыми. Название этой обобщенной технологии, получившей становление в период бурного развития работ по созданию экспертных систем, начнется достаточно условным и подчеркивает ее исторические истоки.

Смежное направление в развитии интеллектуальных систем основано на применении *технологии нечеткой логики,* ориентированной на обработку логико-лингвистических моделей представления знаний. Модели такого типа предназначены для формализации неточных, размытых в смысловом отношении суждений и строятся с использованием обобщенных категорий, задающих классификацию исходных понятий на уровне нечетких множеств. Следует отметить, что соответствующие методы нечеткого логического вывода позволяют обеспечить параллельную интерпретацию имеющихся знаний с помощью специализированных средств аппаратной поддержки, обладающих высоким быстродействием.

Один из перспективных подходов к организации обработки неявных форм представления знаний связан с применением *технологии нейросетевых структур,* аккумулирующей и воспроизводящей основные функциональные особенности биологических прототипов. Эта технология построения интеллектуальных систем предполагает формирование однородных структур, состоящих из множества взаимосвязанных элементов с заданной характеристикой преобразования сигналов. Совокупность знаний, закладываемых в процессе обучения такой структуры, определяется настройкой коэффициентов межэлементных связей и позволяет обеспечить надежную классификацию предъявляемых примеров. При этом важнейшей особенностью нейросетевых структур является их высокое быстродействие, достигаемое за счет параллельности обработки информации при их аппаратной реализации.

Поиски альтернативных путей построения быстродействующих систем обработки знаний привели к развитию *технологии ассоциативной памяти.* Эта технология пред полагает использование механизмов восстановления целостных образов по их отдельным элементам и сводится к работе с многомерными массивами данных. Хранящиеся в памяти знания имеют неявную форму представления и задают классификацию при работе с многомерными массивами данных. Хранящиеся в памяти знания имеют неявную форму представления и задают классификацию понятий некоторой предметной области в виде сочетания признаков, присущих каждой качественной категории. Главные преимущества такого подхода связаны с простотой как программного, так и аппаратного воплощения ассоциативной памяти, которая обеспечивает высокое быстродействие, определяемое временем обращения к отдельной ячейке

Сравнительный анализ различных интеллектуальных технологий позволяет выделить ряд общих для них черт (табл. 1), главная из которых связана с использованием классификации тех или иных понятий в качестве средства для установления связей между отдельными явлениями рассматриваемой предметной области. Эта особенность имеет ключевое значение для разработки принципов организации интеллектуального управления на основе применения современных технологий обработки знаний.

Очевидно, что прикладное развитие интеллектуальных технологий должно соответствовать достижению качественно новых уровней в решении насущных проблем, в данном случае связанных с управлением. Специфика этой сферы достаточно полным и конструктивным образом представлена в энциклопедическом определении понятия управления: *управление* — функция организованных систем различной природы (биологических, социальных, технических), обеспечивающая сохранение их определенной структуры, поддержание режима деятельности, реализацию их программ и целей**.**

Приведенная формулировка не только точно отражает суть предмета, но и подчеркивает многогранность функции управления: от поиска путей выполнения поставленных целей до обеспечения практических действий по их реализации.

Сложность, а в ряде случаев и невозможность формализации задач управления обусловливают целесообразность и необходимость их решения с привлечением методов и технологий искусственного интеллекта.

4. Практическая реализация концепции ситуационного управления на основе современных интеллектуальных технологий

В рамках теории ситуационного управления — нового фундаментального направления, развиваемого силами отечественной научной школы под руководством Д.А. Поспелова, — были разработаны базовые основы такого подхода. Исходя из его ключевых положений каждому классу ситуаций (рис. 1), возникновение которых считается допустимым в процессе функционирования системы, ставится в соответствие неко-

торое решение по управлению (управляющее воздействие, программно-алгоритмическая управляющая процедура и т.д.).

Тогда сложившаяся ситуация, определяемая текущим состоянием как самою объекта, так и его внешней среды и идентифицируемая с помощью измерительно-информационных средств, может быть отнесена к некоторому классу, для которого требуемое управление уже считается известным.

Таким образом, практическая реализация концепции ситуационного управления на основе современных интеллектуальных технологий предполагает наличие развернутой базы знаний о принципах построения и целях функциони**ро**вания системы, специфике использования различных алгоритмов, особенностях исполнительных механизмов и управляемого объекта. В этом случае классификационный анализ имеющихся знаний с учетом текущих показаний измерительно-информационных средств должен обеспечивать параметрическую и структурную настройку управляющих алгоритмов, модификацию программы достижения целей управления, а при необходимости и их коррекцию.

Таблица 1

Свойства интеллектуальных технологий

Технология	Представление знаний	Формирование начальных знаний	Организация логического вывода	Возможность пополнения знаний	Объяснение принимаемых решений	Способ реализации и обеспечиваемое относительное быстродействие
Экспертных систем	В явном виде с помощью продукционных правил, семантических сетей, предикатов и фреймообразующих структур	С помощью эксперта в интерактивном режиме	Обеспечивается сопоставлением начальной посылки с многоуровневой классификацией, заданной иерархией продукционных правил или других представлений	Обеспечивается путем изменения продукционных правил, семантических связей и других представлений	Может быть обеспечено за счет анализа активизированной цепи логического вывода	Программный, низкое
Нечеткой логики	В полускрытом виде с помощью продукционных правил и функций принадлежности, отражающих взаимосвязь входных и выходных параметров и их физическую значимость	С помощью эксперта в интерактивном режиме или в автоматическом режиме на основе анализа статистических данных о функционировании системы	Обеспечивается выполнением продукционных правил и выбранным методом обработки функций принадлежности	Обеспечивается за счет изменения системы правил, формы и относительного размещения функций принадлежности на базовых осях	Может быть обеспечено за счет анализа срабатывающих правил	Программный и аппаратный, высокое и низкое соответственно
Нейросетевых структур	В неявном виде в архитектуре сети, параметрах нейронов и связей	На примере обучающей выборки с помощью алгоритмических процедур настройки в автоматическом режиме	Обеспечивается логикой работы сети	Обеспечивается путем изменения топологии, структуры и параметров сети	Может быть обеспечено за счет введения дополнительной объясняющей нейросети	Аппаратный, высокое
Ассоциативной памяти	В неявном виде в форме гиперповерхности в многомерном пространстве признаков в архитектуре ассоциативной памяти	Путем автоматического формирования ассоциативных связей по заданному алгоритму	Обеспечивается проецированием рабочей точки гиперповерхности на оси выбранной системы координат	Обеспечивается путем изменения пространства параметров и формы гиперповерхности	Может быть обеспечено введением дополнительной координаты с пояснениями	Программный и аппаратный, высокое

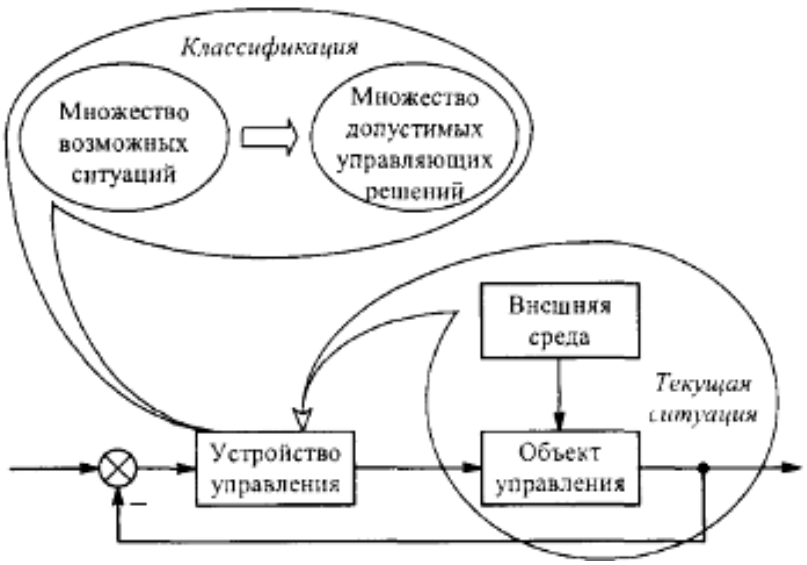


Рис.1 – Реализация принципов ситуационного управления в автоматических системах

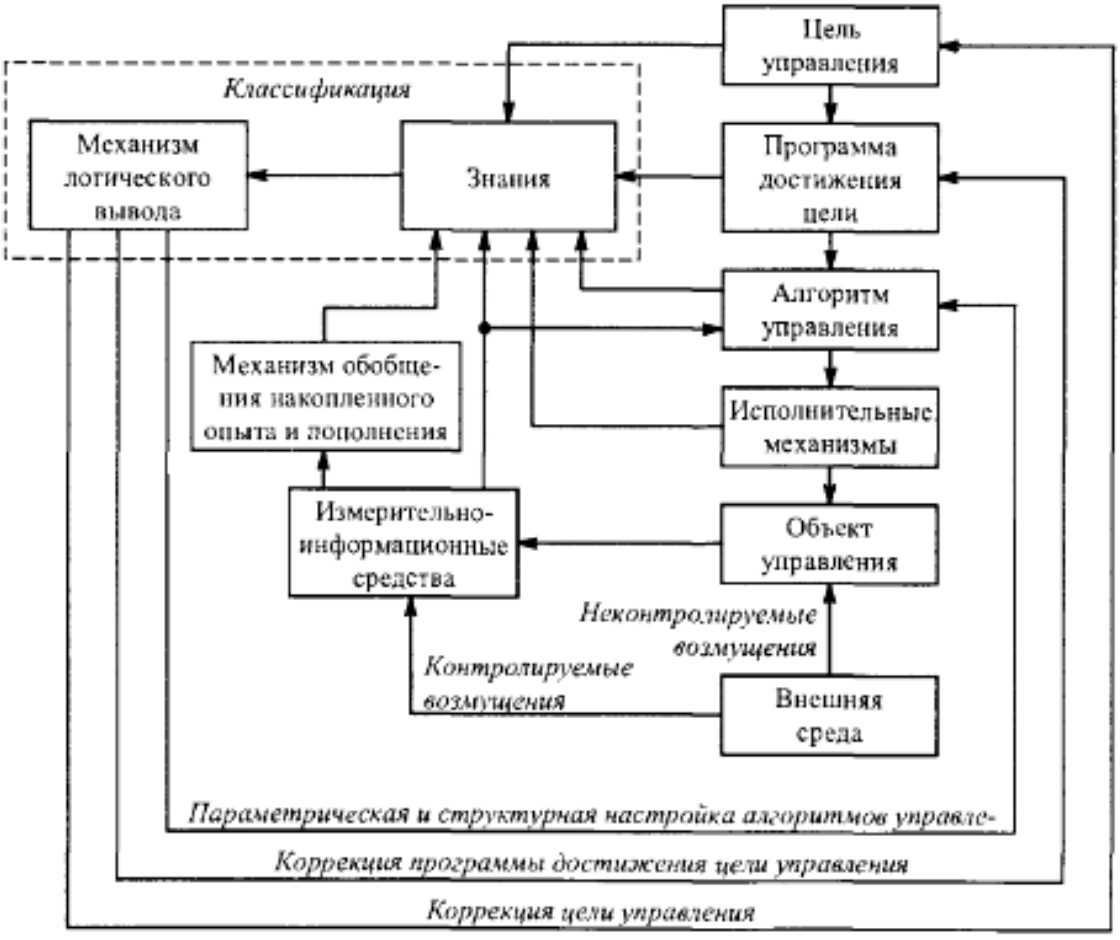


Рис. 2 – Обощенная структура системы интеллектуального управления

Важно отметить, что главная архитектурная особенность, которая отличает интеллектуальную систему управления (рис. 2) от построенной по "традиционной" схеме, связана с подключением механизмов хранения и обработки знаний для реализации способностей по выполнению требуемых функций в исполин заданных (или неопределенных) условиях при случайном характере внешних возмущений. К возмущениям подобною рода могут относиться непредусмотренное изменение целей, эксплуатационных ха-

рактеристик системы и объекта управления, параметров внешней среды и т. д. Кроме тою, состав системы при необходимости дополняется средствами самообучения, обеспечивающими обобщение накапливаемою оныга и на этой основе пополнение знаний.

В общем случае объект управления может быть достаточно сложным и включать ряд функционально-подчиненных подсистем. Иерархия их подчинения обусловливает декомпозицию исходных целей и задач управления на рекурсивную последовательность вложенных составляющих. В конечном итоге такое разделение предполагает многоуровневую организацию системы управления, обладающей развитыми интеллектуальными возможностями по анализу и распознаванию обстановки, формированию стратегии целесообразною поведения, планированию последовательности действий, а также синтезу исполнительных законов, удовлетворяющих заданным показателям качества. При этом структура системы интеллектуального управления сложным динамическим объектом (рис. 3) должна соответствовать иерархическому принципу построения и включать стратегический, тактический и исполнительный (приводной) уровни, а также комплекс необходимых измерительно-информационных средств. Корректность замыкания отдельных контуров иерархии управления определяется тем составом функциональных элементов, которые обеспечиваю требуемую адекватность информационной поддержки в процессе сбора и обобщения сенсорных данных о текущем состоянии и воздействиях внешней среды. Таким обратом, организация каждою уровня интеллектуального управления предполагает использование уникальной совокупности собственных моделей представления знаний, информационной поддержки, описания контролируемого объекта и т. д.

Главным отличием новой концепции иерархического построения систем управления сложными динамическими объектами является использование методов и технологий искусственного интеллекта как средства борьбы с неопределенностью внешней среды. Необходимость интеллектуализации каждого из уровней управления обусловлена подверженностью выполняемых ими функций влиянию различных факторов неопределенности. Практическое воплощение этой концепции предполагает избирательное использование тех или иных технологий обработки знаний в зависимости от специфики решаемых задач, особенностей управляемого объекта, его функциональною назначения, условий эксплуатации и т. д.

Как показывает обзор многочисленных работ по развитию методов обработки знаний, одна из передовых тенденций в этой области связана с попытками интеграции различных интеллектуальных технологий для сочетания их преимуществ Так, например, одновременное обеспечение высокой функциональной гибкости и быстродействия может достигаться за счет комплексного применения технологий экспертных систем и нейросетевых структур. В то же время для увеличения быстродействия ассоциативной памяти предлагаются нейроесетевые способы се реализации. Совмещение технологий экспертных систем и нечеткой логики позволяет не только повысить быстродействие интеллектуальной системы, но и сократить объем базы знаний (по верхней оценке — от одного до двух порядков). Другой подход к проблемам оптимизации интеллектуальных систем и их обучения связан с разработкой комбинированных технологий нечетких нейросетевых структур.

Результаты поисковых исследований по развитию интегрированных технологий обработки знаний имеют большую актуальность для решения задач проектирования систем интеллектуального управления с учетом противоречивости предъявляемых к ним требований. Современные специализированные программно-инструментальные средства позволяют не только подробно промоделировать создаваемую систему управления, но и оценить эффективность принятых проектных решений при различных вариантах их реализации на основе той или иной интеллектуальной технологии. В частности, пакет прикладных программ Win FACT (Windows Fuzzy and Control Tools) обеспечивает воз-

можность перехода к нейросетевому варианту реализации синтезируемой с его помощью модели нечеткого управления.

Несмотря на значительные успехи по многим теоретическим и практическим аспектам применения методов и технологий обработки знаний в задачах управления количество нерешенных в этой области вопросов не только не уменьшилось, но возросло до такой степени, что стало очевидным появление новою научного направления. Зародившись на стыке искусственного интеллекта и теории управления как двух различных сфер человеческих знаний, это направление в настоящее время приобрело большое самостоятельное значение и имеет сложившуюся программу приоритетных исследований следующей проблематики:

— анализ особенностей различных интеллектуальных технологий и методов обработки знаний применительно к задачам управления;

— разработка теоретических основ интеллектуального управления;

— разработка принципов формирования знаний для конкретных предметных областей прикладного применения технологий интеллектуального управления;

— синтез интеллектуальных регуляторов и систем управления для быстродействующих объектов, устройств и процессов;

— исследование динамики интеллектуальных систем управления;

— разработка принципов построения аппаратных и программных средств интеллектуальных систем управления.

Проведение широкомасштабных исследований в области интеллектуального управления обеспечит возможность создания принципиально нового поколения техники, предназначенной для автономного функционирования в условиях неполноты и неопределенности поступающей информации при наличии случайных возмущений внешней среды.

Контрольные вопросы:

1. Интерес к интеллектуальным системам управления (ИСУ)?
2. Перечислите базовые интеллектуальные технологии?
3. Какова машинная форма представления информации?
4. Практическая реализация концепции ситуационного управления на основе современных интеллектуальных технологий?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 2
Тема: Синтез цели и принятие решения для начала движение

План:

1. Фундаментальные и прикладные работы по созданию интеллектуальных систем управления
2. Модели УО

1. Фундаментальные и прикладные работы по созданию интеллектуальных систем управления

В последние годы за рубежом существенно повысился интерес к исследованию прикладных интеллектуальных управляющих систем, их разработке и внедрению в промышленную и непромышленную сферы. Стремительно растет число публикаций на эту тему. Сейчас уже можно говорить о становлении новою научного направления — теории интеллектуального управления. Развитые страны вкладывают в разработку интеллектуальных систем управления миллиарды долларов. Такая интенсификация разработок и соответствующих материальных затрат базируется на тщательном планировании и подробном маркетинге разработок в этой области. Так, большой интерес проявляется к созданию интеллектуальных систем управления с нечеткой логикой. По мнению экспертов, первенство в прикладном и коммерческом использовании результатов теории нечетких систем управления принадлежит японским фирмам. К настоящему времени известно более 400 примеров практического применения интеллектуальных систем управления различного назначения.

В России фундаментальные и прикладные работы по созданию интеллектуальных систем управления активно проводятся во многих организациях: и Вычислительном центре РАН. В Институте прикладной механики РАН. МЭИ. МАИ, ни кафедре "Проблемы управления" МИРЭЛ и в целом ряде других организаций. Этому предшествовал длительный период теоретических исследований в области теории искусственного интеллекта, ситуационною управления и семиотическою моделирования, главным идеологом которых был и остается профессор Д.А. Поспелов (Вычислительный центра РАН) [3, 7]. Именно под его руководством в течение многих лет с начала шестидесятых годов функционировал Всесоюзный семинар но ситуационному управлению, более чем на четверть века опередивший современные исследования в этой области. В наши дни управление на основе анализа внешних ситуаций (событий) остается одной из ключевых идей интеллектуального управления техническими и организационными системами. Другой базовой идеей послужило использование средств современной информационной технологии обработки знаний при поиске управленческих решений и формировании соответствующих управляющих воздействий.

В соответствии с классической теорией автоматического управления под *управлением* всегда подразумевается специальным образом организованное взаимодействие объекта и устройства управления.

Под *объектом управления (ОУ)* подразумевается устройство, осуществляющее некий технологический процесс, на которое подаются специально орга-

низованные воздействия. Под *устройством управления* (УУ) подразумевается устройство, формирующие эти воздействия. В тех случаях, когда как для ОУ, так и для УУ могут быть найдены свои модели, соответствующие устройства проектируются отдельно. Однако далеко не всегда работа УУ и ОУ может быть представлена отдельными моделями. Эти блоки могут оказаться неотделимыми друг от друга. В гаком случае говорят об *управляемом объекте* (УО) или о *системе автоматического управления* (САУ). для которой ищут единую формальную модель. Далее под УО понимается ОУ в совокупности с УУ, функционирование которого описывается единой формальной моделью: УО представляется единым блоком, связанным по входам и выходам с внешней средой. При этом знания о характеристиках внешней среды, типах взаимосвязей и особенностей взаимодействия УО с внешней средой составляют совокупность знаний разработчика модели системы управления.

2. Модели УО

В классической теории автоматического управления УО характеризуются рядом свойств: целевым назначением, множеством состояний, управляемостью, наблюдаемостью, устойчивостью и г. д. Перечисленные свойства позволяют уточнить взаимосвязи УО с внешней средой.

Модели УО создавались по мере возникновения требований к автоматическим системам Можно выделить следующие три класса моделей с учетом характера взаимодействия УО с внешним миром:

первый класс— информационно изолированные от внешнего мира системы, "живущие" в реальном внешнем мире и не использующие ни информации, ни воздействий из этого мира (кроме, разве что. возмущающих);

второй класс — связанные с техническим внешним миром (информационно замкнутые через внешний мир) системы, "живущие" в техническом (формализованном) внешнем мире и перерабатывающие информацию, поступающую из него;

третий класс — информационно связанные с реальным внешним миром системы, "живущие" в естественном внешнем мире и перерабатывающие информацию, поступающую из этого мира.

Системы первого класса это те, для проектирования которых и создавалась теория автоматического управления (первоначально — теория регулирования). Объектами изучения теории автоматического управления стали регуляторы. С помощью систем такого типа решается задача поддержания (без вмешательства человека-оператора) на определенном уровне или в заданных пределах требуемых значений физических ветчин, характеризующих определенный режим работы объекта. УО, состоящий из регулятора и объекта регулирования, охваченных обратной связью, составляют систему автоматического регулирования (САР). Главными задачами при построении САР являются обеспечение устойчивости, необходимых показателей качества переходных процессов и требуемой ошибки в установившемся режиме.

Потребность в разработке всевозможных автоматических устройств, работающих на принципах обработки дискретной информации, возникла, в частности, в связи с развитием сетей транспорта и связи. Эти и подобные им системы составили основу управляющих систем второго класса. Их главная особенность связана с необходимостью функционирования в дискретном времени и обработкой

дискретных сигналов, поступающих на внешние контролируемые входы Возмущения из технического внешнего мира на УО не поступают, поскольку в техническом мире их нет (если он правильно организован). Исследования в облает анализа поведения и синтеза соответствующих моделей управляющих систем второго класса привели к становлению и развитию теории дискретных устройств и конечных автоматов, на базе которой создавались различные системы, начиная от простейших автоматов и кончая вычислительными машинами дискретного действия (не описываемыми уже в классе автоматных моделей).

Появление вычислительных машин сказалось и на развитии теории управления. Довольно скоро выяснилось, что ЭВМ представляет собой универсальный преобразователь информации, способный на нечто большее, чем служить сверхбыстродействующим арифмометром. Переориентация применений вычислительных машин на выполнение функций обработки нечисловой информации послужила главной предпосылкой появления совершенно нового класса управляющих систем.

В системах третьего класса именно ЭВМ отводилась важнейшая роль. В этих системах вычислительная машина перерабатывала лишь формализованную информацию, которая подготавливалась человеком-оператором, переводившим в общем случае невычислительные задачи внешнего мира в вычислительные, а результаты вычислений — в воздействия на окружающий мир. Так появились человеко-машинные системы (первые примеры систем третьего класса). Включение "человеческого звена" в контур управления оказывало на систему двоякое влияние. С одной стороны, это давало возможность существенным образом повысить гибкость и универсальность: система становилась способной к решению широкого круга - задач в условиях неполной и неопределенной информации из окружающего мира. С другой стороны, это приводило к увеличению неопределенности поведения всей системы в целом из-за внесения в процесс управления непредсказуемых действий оператора, соответствующих сто собственным целям и задачам. Попытка устранения субъективных признаков, связанных с присутствием человека-оператора в контуре управления системы, привела к постановке задачи формализации работы этого специфического звена, вносящего существенную неопределенность в работу системы. Все усилия, предпринимавшиеся с этой целью (использовались дифференциальные и логические уравнения, модели состояний, игровые подходы и пр.), оказались напрасными.

Контрольные воросы:

1. Фундаментальные и прикладные работы по созданию интеллектуальных систем управления?
2. Какие модели УО вы знаете?
3. Каковы особенности систем третьего класса?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 3: Активные оценки информации

План:

1. Основные блоки концептуальной архитектуры интеллектуальной системы
2. Аппаратно-реализованные блоки ИС

Впервые понятие "интеллектуальная машина" или "интеллектуальная система" возникло по меньшей мере два десятка лет тому назад. В начале семидесятых годов под этим термином подразумевалась система, реализующая так называемые "антропоморфные функции". (Однако, поскольку до сего времени эти функции никто не выделил и не уточнил, это определение оказалось неконструктивным, так кок из него никоим образом не следуют хоть какие-нибудь намеки на организацию или особенности функционирования интеллектуальной системы.) С течением времени развитие таких разделов искусственного интеллекта, как инженерия знаний, компьютерные логика и лингвистика, когнитивная психология, методы и модели обучения, методы поиска и принятия решений и др. заложило теоретическую основу для создания высокоэффективных программных систем по обработке и использованию знаний для решения целого ряда прикладных задач, включая разработку систем, моделирующих творческие возможности человека. Такие системы и стали называть "интеллектуальными" И сегодня понятия "интеллектуальная система" и "система, ориентированная на обработку и использование знаний" являются синонимами. Именно "интеллектуальные системы", а не "системы искусственного интеллекта", как очень часто говорят и пишут, поскольку понятие "искусственный интеллект" есть ничто иное, как языковая метафора, заменяющая название обширного научною направления, не поддающегося до сего времени точному определению и связанного с исследованием теоретических проблем обработки и использования знаний в слабо формализуемых областях. Поэтому но аналогии с "физическими системами", "кибернетическими системами" и т.п. логичнее употреблять термин "интеллектуальные системы", а применительно к области управления — термин *"интеллектуальные системы управления"*.

1. Основные блоки концептуальной архитектуры интеллектуальной системы

Интеллектуальные системы в последнее время стали весьма распространенным коммерческим продуктом, находящим широкий спрос пользователей-специалистов в самых разнообразных областях инженерно-технической и научно-технической сфер деятельности. Концептуальная архитектура любой интеллектуальной (в частности, экспертной) системы общеизвестна и содержит следующие основные блоки:

- база знаний с развитыми механизмами вывода на знаниях;
- интеллектуальный решатель (формулирующий постановку и общий план решения задачи);
- интеллектуальный планировщик (формирующий конкретный план решения задачи);
- система объяснения;
- интерфейс с пользователем.

Интеллектуальные системы могут существенным образом различаться по архитектуре и выполняемым функциями, но в них всегда в той или иной мере присутствуют указанные блоки.

По мере совершенствования систем, ориентированных на хранение, пополнение, обработку и использование знаний, начали создаваться системы, в которых результаты принятия решении приближаются по качеству к решениям, принятым челевеком-оператором, а по скорости получения решений существенно превышают время реакции человека (особенно в непредсказуемых и непредвиденных ситуациях). Возникла идея активизировать деятельность систем путем включения в их состав специальных дополнительных блоков формирования управляющих воздействий на основе принятых решений. Такие интеллектуальные системы, непосредственно подключенные к объекту, получили название *"активные системы",* в частности "активные экспертные системы".

Строго говоря, активная экспертная система (с точки зрения новых выполняемых ею функций) уже не является экспертной, т. е. не является только системой-советчиком. В активных экспертных системах блок интерфейса с пользователем естественным образом дополняется блоком интерфейса с объектом управления.

2. Аппаратно-реализованные блоки ИС

Следующим немаловажным фактором, заставившим исследователей обратить внимание на возможность создания специальных моделей интеллектуальных систем управления, послужило развитие аппаратных средств поддержки процессов, протекающих в интеллектуальных системах. Первоначально с целью ускорения процессов обработки знаний, а позже и с целью снижения сроков создания интеллектуальных систем, в их состав стали включать аппаратно-реализованные блоки, осуществляющие некоторые функции системы. Можно выделить три основные группы таких средств:

—спецпроцессоры поддержки языков программирования высокого уровня (типа Лисп, Пролог, Рефал и др.);

—спецпроцессоры для интеллектуальных баз данных и баз знаний (в том числе для логического вывода);

—спецпроцессоры для интеллектуального интерфейса (обработки изображений, текстов и речи).

Отдельную группу составляют всевозможные аппаратно реализованные средства обработки лингвистической, в том числе нечеткой информации (нечеткие процессоры).

Интеллектуальные системы с такими блоками в своем составе получили название "систем с развитыми средствами аппаратной поддержки".

Из всех типов интеллектуальных систем наибольшее распространение в сфере управления получили экспертные системы, выступающие в роли советчиков оператора, выполняющего функции взаимодействия с внешним миром. С появлением экспертных систем с развитыми средствами аппаратной поддержки, способных оказать интеллектуальную помощь управленцу не хуже, чем высококвалифицированный специалист, закончился начальный период исследований возможностей построения систем третьего класса (в составе которых человек ис-

пользов00ался как некое весьма специфическое звено управления, формализовать работу которого так и не удалось разработчикам). Особенно удобными для целей технического управления оказались так называемые *открытые системы,* т. е. системы, способные с течением времени совершенствовать свое поведение благодаря заложенным в них алгоритмам обучения Общесистемный подход к решению задачи проектирования таких нитрированных систем привел к формированию нового научного направления — теории многоагентных интеллектуальных систем, — возникшего на стыке многих упоминавшихся выше научных направлений.

Основным предметом исследований в теории интеллектуальных систем явилась разработка структур, претендующих на обеспечение интеллектуального поведения при решении различных задач. Принцип, сформулированный Саридисом а 1989 г. и обозначенный IPDI (Increasing Precision with Decreasing Intelligence), означает, что по мере продвижения к высшим уровням иерархической структуры повышается интеллектуальность системы, но снижается ее точность, и наоборот.

Важно отметить, что под *интеллектуальностью* системы здесь подразумевается се способность работать с базой внешних событий или ситуаций для привлечения знаний, позволяющих уточнить предложенную задачу и наметить пути ее решения; под *неточностью* понимается неопределенность, или, другими словами, свобода выбора в выполнении операции по решению задачи. Каждому из уровней (которые в свою очередь также могут иметь иерархическую структуру построения) соответствует специальная подсистема, реализующая перечисленные ниже функции, характерные для этого уровня. Так, например, на верхнем уровне структуры имеется основанный на знаниях организатор, промежуточному уровню соответствует основанный на знаниях координатор, а самому нижнему уровню — система управления аппаратными средствами, решающими поставленную задачу, сведенную к конкретным алгоритмам.

Попытаемся теперь расширить смысл понятия "интеллектуальность" по сравнению с толкованием Саридиса с целью более четкого определения класса интеллектуальных систем управления и рассмотрим более подробно вопрос структурной организации систем.

Контрольные вопросы:

1. Основные блоки концептуальной архитектуры интеллектуальной системы?
2. Аппаратно-реализованные блоки ИС?
3. Какие группы входят в состав аппаратно-реализованных блоков ИС

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 4: Предсказание результатов движение и разработка управление

План:

1. Принципы оценки поведения интеллектуальных систем управления
2. Степени интеллектуальности
3. Модели и методы вывода в условиях неопределенности используемой информации
4. Уровни иерархии интеллектуальной системы управления и степень интеллектуальности

1. Принципы оценки поведения интеллектуальных систем управления

Рассмотрим класс систем управления, которые соответствуют следующим пяти принципам организации интеллектуальной управляющей структуры:

—наличие тесного информационного взаимодействия управляющих систем с реальным внешним миром и использование специально организованных информационных каналов связи;

—принципиальная открытость систем для повышения интеллектуальности и совершенствования собственною поведения.

—наличие механизмов прогноза изменений внешнего мира и собственного поведения системы в динамически меняющемся внешнем мире,

—построение управляющей системы в виде многоуровневой иерархической структуры в соответствии с правилом; повышение интеллектуальности и снижение требований к точности по мере повышения ранга иерархии в системе (и наоборот);

—сохраняемость функционирования (возможно, с некоторой потерей качества или эффективности) при разрыве связей или потере управляющих воздействий от высших уровней иерархии управляющей структуры

Прокомментируем смысл выделенных пяти принципов, отметив их исключительную важность с точки зрения оценки поведения интеллектуальных систем управления.

Первый принцип подчеркивает непосредственную связь интеллектуальных управляющих систем с внешним миром. Находясь в непрерывном взаимодействии с внешним миром, интеллектуальные системы получают из него всю необходимую информацию для принятия решений и пополнения знаний. Сама управляющая система в свою очередь может оказывать на внешний мир активное воздействие в результате реализации собственного поведения. Модель знаний о внешнем мире интеллектуальной системы должна предполагать в этом смысле возможность изменений внешнего мира и знаний о нем в результате воздействий на нею системы. Выполнение принципа информационного взаимодействия системы с внешним миром означает, что любые упрощенные модели типа моделей состояний, вероятностных описаний, игр автоматов со средой и т. П. для представления событий реальною внешнего мира непригодны. Именно в этом и состоит специфика систем управления рассматриваемого класса.

Принципиальная открытость систем в соответствии *со вторым принципом* обеспечивается наличием таких подсистем высшего ранга в иерархической струк-

туре, как самонастройка, самоорганизация и самообучение. Знания интеллектуальной системы управления состоят из двух частей: постоянных (проверенных) знаний, которыми система обладает и постоянно пользуется, и временных (проверяемых) знаний, в которых система не уверена, с которыми она экспериментирует в процессе обучения. Знания второю типа либо отбрасываются системой, либо переходят в знания первого типа в зависимости от результатов анализа своею поведения во внешнем мире. Выполнение второго принципа требует организации в управляющем системе процесса приобретения и пополнения знаний.

В соответствии с *третьим принципом* управляющую систему нельзя считать в достаточной мере интеллектуальной, если она не обладает возможностью прогноза изменений самого внешнею мира и собственного в нем поведения. Система без прогноза, функционирующая в динамически меняющемся внешнем мире, может попасть в критическую ситуацию, из которой не сможет найти выхода из-за временных ограничений на работу механизмов формирования управляющих воздействий, определяющих се поведение, адекватное сложившейся ситуации.

Четвертый принцип позволяет наметить пути построения моделей сложных управляющих систем в тех случаях, когда неточность знаний о модели объекта или о его поведении можно скомпенсировать увеличением числа уровней интеллектуальности, а также использованием совершенных механизмов принятия решений в условиях неопределенности в соответствующих алгоритмах управления.

И наконец, *пятый принцип* устанавливает лишь частичную потерю интеллектуальности (но не прекращение функционирования) при отказах в работе высших уровней иерархии системы. Сохранение автономного функционирования в рамках более простого (автоматного) поведения системы, характерного для нижних уровней структуры управления.

Также чрезвычайно важно для автономно функционирующих систем в реальном внешнем мире.

Приведенные пять принципов организации структуры интеллектуальной системы управления определяют класс исследуемых систем. Можно уточнить теперь само понятие «интеллектуальность системы управления», а также ввести понятие «уровень» и «степень интеллектуальности».

Для реализации указанных выше пяти принципов в интеллектуальной системе управления необходимо предусмотреть следующие слои обработки неопределенной информации (слои интеллектуальности):

—слой прогноза событий;

—слой самообучения и адаптации;

—слон работы с базами событий, знаний и формирования решений;

исполнительный слой.

Каждый из перечисленных слоев имеет свою функциональную специфику и в реальной системе может состоять из нескольких уровней При этом в самом нижнем исполнительном слое могут использоваться классические модели САУ. Все остальные слои более высокого ранга можно рассматривать как надстройку над традиционными классическими моделями, отвечающую требованиям современной информационной технологии работы со знаниями и существенно расширяющую возможности этих моделей. Минимальная надстройка может содержать всего лишь базу знаний, состоящую из нескольких продукционных правил. В этом простейшем случае могут отсутствовать слои самообучения и прогноза со-

бытий (или функции этих уровней могут быть совмещены с функциями обработки нескольких правил).

—

Рисунок 1 – Принципы организации интеллектуальных систем управления

2.Степени интеллектуальности

В зависимости от того, сколько слоев интеллектуальности имеет та или иная система, можно говорить о разных степенях ее интеллектуальности. Введем ряд определений:

—система управления, функционирование которой ограничено двумя нижними слоями интеллектуальности, имеет степень интеллектуальности в *малом,*

—система управления, функционирование которой ограничено тремя нижними слоями интеллектуальности, имеет степень интеллектуальности *в большом;*

—система управления, функционирование которой поддерживается всеми четырьмя слоями интеллектуальности, называется интеллектуальной *в целом.*

Учитывая важность понятия "степень интеллектуальности" для рассматриваемого класса систем, определения степени интеллектуальности

в малом, в большом и в целом мы ввели по аналогии с устойчивостью в малом, в большом и в целом для классических САУ (САР), в которых устойчивость является наиболее важным понятием.

Введение понятия степени интеллектуальности в малом, в большом и в целом непосредственно связано с традиционными вопросами разработки шкал оценок функциональных возможностей систем управления. Рассматриваемые слои интеллектуальности образуют неравномерную, частично упорядоченную последовательность на шкале оценок функциональных возможностей САУ, зависящих от целей управления, сложности и неопределенности объекта управления. Это позволяет ставить задачи разработки методологии проектирования САУ различных степеней интеллектуальности, где под степенью интеллектуальности можно подразумевать различные средства борьбы с недоопределенностью либо самого

объекта управления, либо его поведения в непредсказуемом динамическом внешнем мире.

Итак, интеллектуальные системы управления — ото системы вовсе не обладающие какой бы то ни было "интеллектуальностью" в общепринятом смысле Это прежде всего класс систем, строящихся с применением новой информационной технологии обработки и использования знаний Такой подход к построению систем управления позволяет в ряде случаев повысить динамические характеристики создаваемой системы путем лингвистической аппроксимации поведенческих характеристик управляемого объекта. Более того, мы можем в ряде случаев отказаться от организации традиционной обратной связи в САУ, cam нам удастся адекватным образом представить ее работу с помощью знаний на основе определенных правил. В качестве примера можно указать на известные исследования по поддержанию в равновесии перевернутого маятника с помощью нечетких правил. Насколько такой регулятор будет работать лучше (или хуже) традиционного, зависит только от того, насколько хорошо (или плохо) нам удалось описать с помощью правил работу обратных связей. Если аппроксимация нам не удалась, интеллектуальный регулятор будет иметь худшие динамические характеристики по сравнению с обычным (например, с ПИД-регулятором).

В системах управления, обладающих интеллектуальностью в целом, свойство интеллектуальности проявляется в таких аспектах, как управление в условиях неопределенности, самообучение и адаптация. Это сложные системы с многоуровневой иерархической структурой, способные к формированию решений, адекватных сложившейся ситуации. Как указывалось в [1]. вся история развития искусственного интеллекта связана в основном с попытками разработки наиболее совершенных методов и средств управления в условиях неопределенности. Так. на заре развития искусственного интеллекта исследовались лишь методы представления и манипулирования знаниями Следующий этап развития характеризовался исследованиями в области достоверного логического вывода с использованием знаний (резолюцией но го. в аксиоматических системах, с использованием аналитических таблиц и т. п.). От доказательного логического вывода исследователи перешли к анализу применимости методов рассуждений, включающих механизмы управления выводом. Попытки решения задач в плохо формализуемых предметных областях привели к развитию методов правдоподобных и приближенных рассуждений (абдуктивных. индуктивных, нечетких, на примерах, на основе здравою смысла, по аналогии и т. п.).

3. Модели и методы вывода в условиях неопределенности используемой информации

Разработанные к настоящему моменту соответствующие модели и методы вывода в условиях неопределенности используемой информации могут найти применение на самых верхних уровнях формирования решений в иерархии интеллектуальных систем управления. Логично предположить, что в интеллектуальных системах управления ближайшего будущего смогут найти применение и такие механизмы поддержки принимаемых решений, какими являются аргументация и обоснование. Соответствующие модели пока не получили широкого распространения в управлении, хотя попытки их использования в интеллектуальных

советующих системах предпринимались, и, судя по первым результатам, им принадлежит большое будущее.

Что же касается самообучения и адаптации интеллектуальных систем управления, то к настоящему моменту достаточно широкое распространение получили методы эволюционного моделирования на базе нейронных сетей, настраиваемых с помощью генетических алгоритмов. Концептуальная модель, организованная с использованием в контуре управления базы нечетких правил, блока нечеткого вывода, фазификатора (переводящего четкие значения входов в лингвистические значения) и дефазификатора (выполняющего обратное преобразование) совместно со средствами обучения на базе нейронной сети, составляет основу схемы так называемых "мягких вычислений" (soft computing — термин Л. Заде). Эта схема уже нашла применение в ряде промышленных разработок интеллектуальных систем управления.

Одним из наиболее распространенных способов описания поведенческих характеристик объекта управления в последнее время стало применение нечетких продукционных правил с соответствующими механизмами нечеткого вывода. В технических системах точные значения сигналов с датчиков подаются на фазифнкатор, а точные значения управляющих воздействий на объект поступают с дефазификатора. Системы управления, построенные но этой схеме, получили название "нечетких систем управления". В соответствии с данным ранее определением нечеткие системы управления (без самообучения, предсказания и т.д.) относятся к числу систем, интеллектуальных в малом. Это - чрезвычайно важный и наиболее распространенный в настоящее время подкласс интеллектуальных систем управления.

В тех случаях, когда в силу наличия большой степени неопределенности поведенческих характеристик объекта управления или внешней среды лингвистическая аппроксимация с использованием нечетких продукционных правил становится недостаточной, в структуру системы управления вводят дополнительные слои, отмеченные выше. Таким образом, можно построить систему с более высокой степенью интеллектуальности или с более развитыми средствами борьбы с неопределенностью используемой информации. В системах управления, интеллектуальных в целом, на верхних уровнях управляющей структуры используются, как правило, экспертные системы, в состав которых могут быть включены такие механизмы правдоподобного вывода на знаниях, как вывод но аналогии, на основе здравого смысла и т. п. В системах такой степени интеллектуальности могут быть автоматически сформулированы решения, вполне приемлемые в сложившейся ситуации, но неожиданные даже для экспертов. Вопрос о том. следует ли такой уровень принятия решения включать непосредственно в контур управления, остается открытым. Однако, в качестве советующих подсистем эти уровни принятия решений, несомненно, полезны.

4. Уровни иерархии интеллектуальной системы управления и степень интеллектуальности

Развивая принцип иерархического построения интеллектуальных систем, в работе предложена концептуальная модель в виде четырех уровней иерархии: исполнительного, тактического, стратегического и информационно-измерительного.

Тезис о том, что "необходимость интеллектуализации каждого из уровней управления обусловлена подверженностью выполняемых ими функции влиянию различных факторов неопределенности", хорошо интерпретируется с помощью введенного выше понятия степени интеллектуальности. Обратимся к рис. 1. Как видно, фундаментом построения интеллектуальной системы управления являются пять принципов, сформулированных в настоящей статье. На них основана иерархия как уровней управления, так и степеней интеллектуальности. Из рис. 1 следует, что каждый уровень иерархии управления может иметь различную степень интеллектуальности: от интеллектуальности в малом до интеллектуальности в целом.

Достаточно очевидно, что на стратегическом уровне необходимо иметь высокую степень интеллектуальности, что определяется набором функциональных задач по планированию целесообразного повеления сложного объекта управления и согласуется с принципом Саридиса. Более проблематичной, на первый взгляд, представляется интеллектуализация тактического и тем более приводного уровня. Тем не менее совершенно реальный смысл приобретает термин "интеллектуальный привод", под которым понимается привод с системой управления, имеющей степень интеллектуальности в малом.

Цикл работ, выполненных на кафедре "Проблемы управления" МИРЭА, убедительно показал, что применение интеллектуальных технологий позволяет синтезировать на базе достаточно простых аппаратных и программных средств новое поколение регуляторов (интеллектуальные рС1уляторы) для создания широкого спектра высококачественных адаптивных электроприводов. На первом этапе этих работ наибольшее развитие получили технологии экспертных систем и нейросетевых структур. Экспертная система выполняла функции интеллектуальной надстройки над ПИД-регулятором и периодически подстраивала его коэффициенты в зависимости от изменения параметров следящего электропривода. Занимая объем памяти около 350 Кб, экспертный регулятор обеспечивал адаптивное управление в широком диапазоне возмущений, но не обладал быстродействием, необходимым для управление в реальном масштабе времени.

Регулятор, построенный на базе 80 статических нейронов (названный нейросетевым регулятором) и обученный на оптимальный по быстродействию принцип функционирования, включался в контур системы управления последовательно с объектом. Он обеспечивал очень высокое быстродействие при слежении за различными входными воздействиями и, что особенно интересно, инвариантность к определенного рода внешним возмущениям.

Работы по созданию экспертного и нейросетевого регуляторов подтверждают, что интеллектуализация приводного уровня это не самоцель, а новый способ решения комплексной задачи адаптивного управления. В ходе дальнейших исследований рассматривались все четыре интеллектуальных технологии. В каждой из них обнаружены свои достоинства и недостатки. Перспективным представляется применение технологии ассоциативной памяти, поскольку реализованный на ней интеллектуальный регулятор привода требует менее 20 Кб памяти.

Весьма убедительные результаты получены по созданию на базе нейросетевых структур самообучающихся систем управления со степенью интеллектуальности в большом Под самообучением понимается использование комплекса методов и алгоритмов для настройки и функционирования системы управ-

ления с неизвестным динамическим объектом. В процессе синтеза системы осуществляются следующие этапы функциональная идентификация объекта, первоначальная настройка регулятора, адаптация параметров регулятора в процессе управления

На базе различных интеллектуальных технологий отрабатывались также задачи тактического уровня управления робототехнических систем Решались, в частности, задачи вывода манипулятора в заданную точку с учетом обхода препятствий. Интересные результаты по созданию системы управления тактического уровня со степенью интеллектуальности в большом были получены на основе аппарата нечеткой логики.

Общие методологические принципы, предложенные в данной статье, позволяют выработать конкретные рекомендации по определению степени интеллектуализации тех или иных уровней иерархии интеллектуального управления в зависимости от специфики объекта управления и тактико-технических требований, предъявляемых к системе управления Опыт практических разработок по применению интеллектуальных технологий экспертных систем, нейросетевых структур, ассоциативной памяти и нечеткого вывода для отдельных уровней иерархии управления с различной степенью интеллектуальности послужил объективной основой для тех принципиальных обобщений, которые представлены в данной работе.

Контрольные вопросы:

1. Принципы оценки поведения интеллектуальных систем управления?
2. Перечислите степени интеллектуальности?
3. Модели и методы вывода в условиях неопределенности используемой информации?
4. Уровни иерархии интеллектуальной системы управления и степень интеллектуальности?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 5: Преобразование управление в физические сигналы

План:

1. Физические и математические основы оптических систем контроля
2. Контроль качества поверхности (ее шероховатости)
3. Контроль изделий сложной формы в составе технологических систем.

1. Физические и математические основы оптических систем контроля

Рассмотрим случай, когда производится контроль геометрических размеров на плоскости, располагаемой перпендикулярно к оптической оси телекамеры (рис. 12.3).

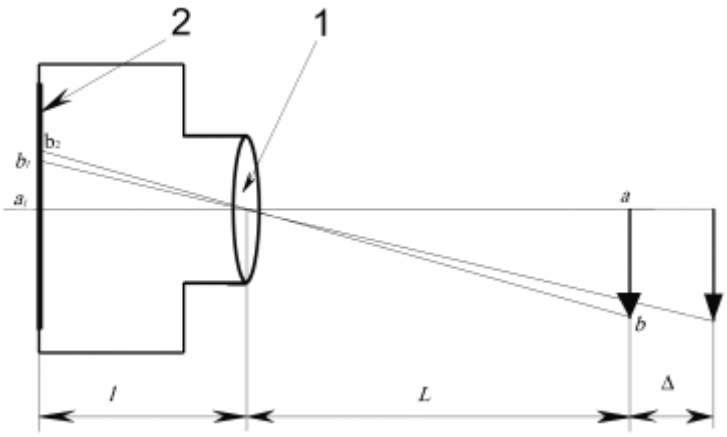


Рис. 12.3.

Изображение, зафиксированное телекамерой, записывается в *память компьютера* как *двумерный массив* координат точек анализируемой поверхности в *пикселях)*.

При определении расстояния между точками на плоскости (рис. 12.3) первоначально определяется масштаб m_0 — количество единиц длины, приходящихся на один *пиксель*. Рассмотрим эталонный отрезок ab длиной A (мм), расположенный на расстоянии L от оптического центра объектива. На фотоматрице телекамеры этому отрезку соответствует отрезок a_1b_1 длиной B_1, измеренный в *пикселях*. Масштаб определяется отношением

$$m_0 = \frac{A}{B_1}. \qquad (12.1)$$

Если отрезок ab сместить на расстояние Δ вдоль оптической оси a_1a, то согласно правилу подобия получим

$$\frac{L}{A} = \frac{1}{B_1}, \qquad \frac{L+\Delta}{A} = \frac{1}{B_2}.$$

Поделив левые и правые части полученных равенств друг на друга, получим

$$\frac{L}{L+\Delta} = \frac{B_2}{B_1}. \qquad (12.2)$$

Из (12.2) следует, что взаимосвязь между масштабами при параллельном переносе объекта контроля на расстояние определяется зависимостью

$$m_1 = m_0 \frac{L+\Delta}{L} = m_0 + m_0 \frac{\Delta}{L}. \qquad (12.3)$$

Для определения геометрических размеров анализируемой поверхности используется дифракционная решетка (4, рис. 12.2), которая представляет прозрачную пластину с нанесенными на ней темными полосами (рис. 12.4). Например, решетка, имеющая размер в плане 50x50 мм, расстояние между линиями и их ширину — 1 мм, при точности нанесения линий 1 мкм позволяет на расстоянии L=1,5—2 м выполнять измерение с точностью 6 мкм.

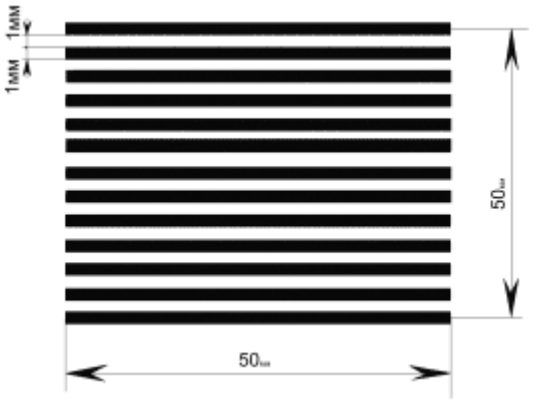


Рис. 12.4.

С помощью дифракционной решетки можно получить полную (непрерывную) информацию о контролируемой поверхности, попадающей в *поле зрения* фотоматрицы. Точность измерения лимитируется частотой полос эталлонной решетки, которая не должна превышать пяти *пикселей* фотоматрицы, при этом промежуточные значения, описывающие контролируемую поверхность, получают аналитически.

Теперь, когда понятна общая схема, связанная с фиксацией изображения в телекамере и масштабированием размеров контролируемого изделия, рассмотрим задачу определения границ изделия по перепадам интенсивности света, попадающего на каждый *пиксель* фотоматрицы. Основным признаком границы изображения является перепад интенсивности освещения в направлении, перпендикулярном линии, определяющей границу. В силу наличия микронеровностей на поверхности, а также *дифракции* света, граница всегда будет размыта. Распределение интенсивности света I(n) в направлении, перпендикулярном границе в точке n_0 имеет вид, представленный на рисунке (рис.12.5, а). Пунктиром обозначен перепад интенсивности в идеальном случае.

Производная по перемещению n от интенсивности света I(n) на границе (в точке n_0) аппроксимируется кривой Гаусса (рис. 12.5, б).

$$\frac{dI(n)}{dn} = A \cdot e^{-\frac{(n-n_0)^2}{2\sigma^2}}, \qquad (12.4)$$

где A — константа, характеризующая максимальное значение перепада интенсивности отраженного света; σ — среднеквадратичное распределение перепада интенсивности отраженного света на границе; n_0 — координаты границы.

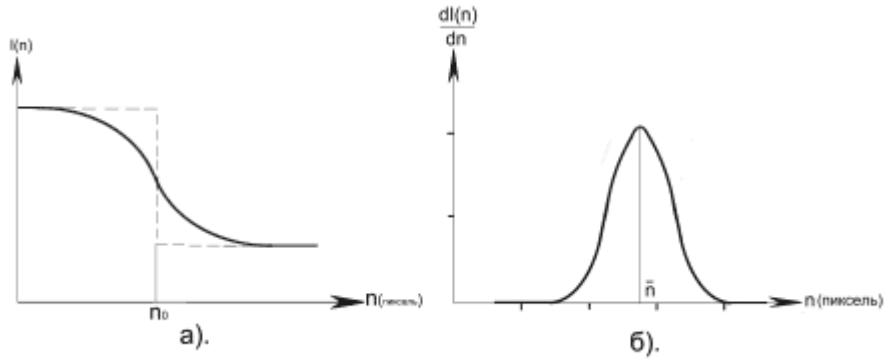


Рис. 12.5.

Для получения распределения интенсивности `I(n)`, зависимой от координат контролируемого изделия, необходимо количество *пикселей* `n` умножить на масштаб m_0. В этом случае будем иметь распределение интенсивности света, попадающего от контролируемого изделия, в координатах, представленных в системе координат фотоматрицы телекамеры. Следует отметить, что данные координаты будут изменяться с дискретностью, равной масштабу m_0, Поэтому измерение расстояний в этом случае будет осуществляться с погрешностью, равной масштабу, что недопустимо для контроля геометрических размеров сложных поверхностей.

Точность определения границ может быть значительно повышена, если интенсивность отраженного сигнала `I(r)` аппроксимировать функцией, зависимой от непрерывных координат `r`. При этом координаты, соответствующие максимальному значению перепада интенсивности, определяются *дифференцированием* непрерывной функции `I(r)` с более высокой точностью, чем в дискретном случае.

Рассмотрим решение данной задачи для двумерного случая. Перепад интенсивности на плоскости характеризуется модулем *градиента функции* `I(x,,y)` от двух переменных `x`, `y`

$$|\operatorname{grad}(I(x,y))| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2}, \qquad (12.5)$$

где переменные `x` и `y` представляют линейные величины в двух взаимно перпендикулярных направлениях и измеряемые в единицах длины.

Определение координат границы осуществляется в следующей последовательности:

1. Сканируется изображение контролируемого изделия[1)]. На основе этого строится матрица координат опорных точек поверхности и значений модуля *градиента функции* интенсивности в этих точках.
2. По полученной матрице строятся непрерывные функции *распределения вероятности* для модуля градиентов интенсивности светового сигнала. Функции *распределения вероятности* строятся вдоль границы изображения изделия таким образом, чтобы осуществлялось пересечение с данной границей.
3. Определяются значения координат, соответствующих максимальному

значению функции распределения. Данные координаты и принимаются за координаты опорных точек границы.

4. Через опорные точки границы проводится непрерывная линия, которая может быть аппроксимирована аналитическим выражением.

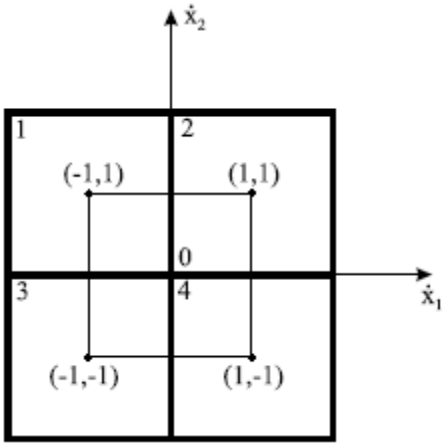


Рис. 12.6.

Сканирование изображения осуществляется *последовательным анализом* интенсивности светового излучения, попадающего на квадрат размерами 2 x 2 *пикселя* (рис. 12.6). Определяется модуль градиента интенсивности светового излучения для квадрата из четырех *пикселей* и координаты центра данного квадрата. Для этого аппроксимируем функцию интенсивности I(x,y) в области размерами 2 x 2 *пикселя* (рис. 12.6) *многочленом* второй степени от безразмерных величин x и y

$$I(x, y) = b_0 + b_1 x + b_2 y + b_3 xy, \quad (12.6)$$

где

$$\tilde{x}_i = \frac{x_{0i} - x_i}{\Delta}, \qquad \tilde{y}_i = \frac{y_{0i} - y_i}{\Delta}$$

— безразмерные величины, определяемые текущими координатами центра (точка 0) — $r_{0i}[x_{0i}, y_{0i}]$ и вычисляемые в системе координат фотоматрицы $(X,Y)_м$; $\Delta_{xi} = \Delta_{yi} = \Delta$ — половина размера *пикселя* в мм.

Представление интенсивности I(x,y) как функции от безразмерных переменных позволяет не учитывать масштаб при ее преобразовании. Центры четырех выбранных *пикселей* (рис. 12.6) имеют относительные безразмерные координаты: для первого *пикселя* (-1,1), для второго (1,1), для третьего (-1,-1) и для четвертого (1,-1).

$$\left.\frac{\partial I(\tilde{x}, \tilde{y})}{\partial \tilde{x}}\right|_{r_{0i}} = b_1, \qquad \left.\frac{\partial I(\tilde{x}, \tilde{y})}{\partial \tilde{y}}\right|_{r_{0i}} = b_2.$$

Значения *частных производных* первого порядка от I(x,y) по переменным (x,y) в центре области 2x2 *пикселя* (точка 0) равны коэффициентам b_1 и b_2 аппроксимирующего полинома (12.6).

Для построения полинома (12.6) требуется определить значения коэффициентов b_0, b_1, b_2, b_3 в пределах рассматриваемой области 2 x 2 *пикселя*. Пред-

ставим (12.6) системой из четырех уравнений в *матричной форме*

MB=I, (12.7)

где

$$M = \begin{Vmatrix} 1 & -1 & 1 & -1 \\ 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \end{Vmatrix}$$

— матрица значений *базисных функций* в точках измерения функции интенсивности света; *базисные функции* в полиноме (12.6) — это вектор R=|1, x, y, xy|^T ; B=||b_0,b_1,b_2,b_3|^T — определяемый вектор коэффициентов полинома; I=||I_1,I_2,I_3,I4||^T — вектор значений интенсивности света в каждом *пикселе*.

При умножении (12.7) на матрицу M^{-1} получим

$B=M^{-1}I$, (12.8)

Неизвестные элементы вектора B=|b_0,b_1,b_2,b_3|T вычисляются из (12.8) через известные значения интенсивности Ii для каждого i-го *пикселя*

$$b_0 = \frac{\sum_{i=1}^{4} I_i}{4}, \quad b_1 = \frac{\sum_{i=1}^{4} \tilde{x}_i I_i}{4}, \quad b_2 = \frac{\sum_{i=1}^{4} \tilde{y}_i I_i}{4}, \quad b_3 = \frac{\sum_{i=1}^{4} \tilde{x}_i \tilde{y}_i I_i}{4}. \qquad (12.9)$$

Сканируя изображение, воспринимаемое всей поверхностью фотоматрицы, квадратами 2 x 2 *пикселя* с шагом, равным одному *пикселю*, можно определить значения модуля *градиента функции* I(x,y) для каждого квадрата:

$$|\mathrm{grad}(I(x, y))| = \sqrt{b_1^2 + b_2^2} \qquad (12.10)$$

Последовательное сканирование изображения изделия позволяет получить матрицу координат и значения модуля градиента интенсивности отраженного света для опорных точек контролируемой поверхности фотоматрицы. Абсолютные значения координат центра квадратов 2 x 2 *пикселя* вычисляются по их известным безразмерным значениям

$$x_{0i} = \Delta\tilde{x}_i + x_i, y_{0i} = \Delta\tilde{y}_i + y_i. \qquad (12.11)$$

Вторым этапом анализа изображения является обработка полученной матрицы распределения модуля градиентов функции интенсивности света и построение функции *распределения вероятности* для модулей градиентов в областях, близких к границе объекта. Функции *распределения вероятности* строятся при изменении непрерывных координат таким образом, чтобы осуществлялось их пересечение с определяемой границей изображения объекта. Чаще всего распределение вероятности модуля *градиента функции* интенсивности I(x,y) строится в направлении одной из координат при постоянном значении другой.

При анализе функции *распределения вероятности* для модуля градиента интенсивности света требуется определить математическое ожидание координат, соответствующих опорным точкам границы. Математическое ожидание одной из координат границы вычисляется при постоянном значении другой как сумма про-

изведений данной координаты в i-й точке на плотность распределения интенсивности в этой точке

$$m(x) = \sum_{i=1}^{n} x_i p_x, \quad \text{где} \quad p_x = \frac{G_i}{\sum_{i=1}^{n} G_i} \quad \text{при} \quad y = \text{const}$$
$$m(y) = \sum_{i=1}^{m} y_i p_y, \quad \text{где} \quad p_y = \frac{G_i}{\sum_{i=1}^{m} G_i} \quad \text{при} \quad x = \text{const}, \qquad (12.12)$$

p_x, p_y — плотности *распределения вероятности* модуля градиента интенсивности света соответственно в направлении осей X и Y фотоматрицы; n и m — количество точек соответственно в направлении оси X и Y ; G_i — значение модуля градиента интенсивности света, попадающего на фотоматрицу телекамеры в пределах 2 x 2 *пикселя*.

После определения координат опорных точек границы через них проводится непрерывная кривая, которая может быть аппроксимирована непрерывными аналитическими функциями либо полиномами.

2. Контроль качества поверхности (ее шероховатости)

Оптическая система, рассмотренная выше, позволяет также контролировать шероховатость поверхности. Это обеспечивается зависимостью интенсивности рассеивания света при отражении его от микронеровностей поверхности (рис. 12.7). Чем больше микронеровностей, тем больше рассеивание света от поверхности и тем меньше его попадает на фотоматрицу. Зависимость интенсивности отраженного света $I(\alpha)$ от угла наблюдения α с достаточной точностью может быть представлена кривой Гаусса (рис. 12.8).

$$I(\alpha) = \frac{A}{\sigma\sqrt{2\pi}} e^{-\frac{\left(\alpha - \frac{\pi}{2}\right)^2}{2\sigma^2}}, \qquad (12.13)$$

где A — константа, характеризующая максимальное значение интенсивности отраженного света $I(\alpha)$; α — среднеквадратичное распределение интенсивности отраженного сигнала; α — угол наблюдения.

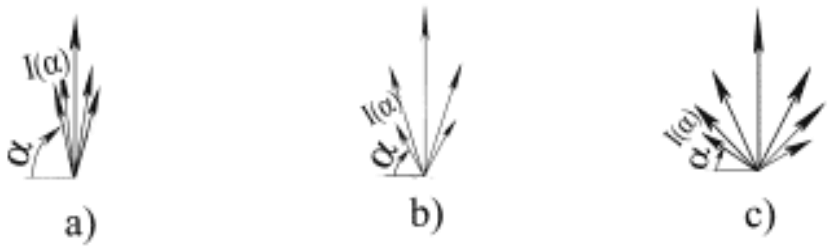


Рис. 12.7.

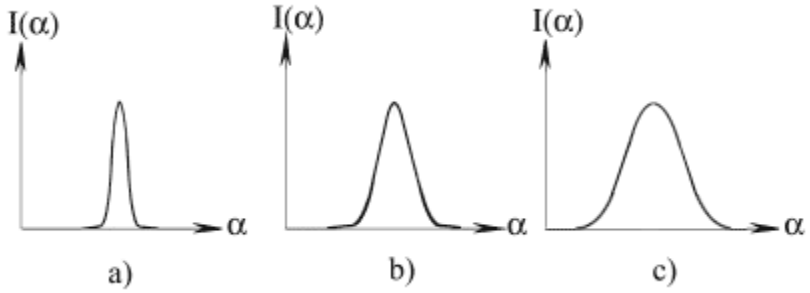


Рис. 12.8.

Для определения шероховатости поверхности требуется экспериментально построить интенсивность светового потока $I(\alpha)$ как функцию угла α. По результатам эксперимента вычисляется также среднеквадратичное отклонение интенсивности отраженного сигнала

$$\sigma = \sqrt{\sum_{i=1}^{n} \alpha_i^2 p_i - (\sum_{i=1}^{n} \alpha_i p_i)^2},$$

где α_i — угол поворота контролируемой поверхности относительно оси телекамеры; p_i — плотность распределения интенсивности света, определяемая как отношение значения интенсивности $I(\alpha_i)$ при α_i к суммарной интенсивности,

$$p_i = \frac{I(\alpha_i)}{\sum_{i=1}^{n} I(\alpha_i)},$$

n — число поворотов поверхности относительно оси телекамеры; $I(\alpha_i)$ — среднее значение интенсивности света, попадающего на фотоматрицу с анализируемой области.

Связь среднеквадратичного распределения интенсивности отраженного сигнала σ с шероховатостью осуществляется определением σ для эталонных пластин заданной шероховатости.

3. Контроль изделий сложной формы в составе технологических систем.

В составе технологического комплекса оптическая система может быть применена для контроля геометрических размеров поверхности, определения границ контура детали. На рисунке 12.9 приведена конструкция оптической системы в составе робота—станка. Прежде всего для определения геометрических параметров анализируемой поверхности и их погрешностей введем основные координатные системы, относительно которых осуществляется преобразование оптического изображения.

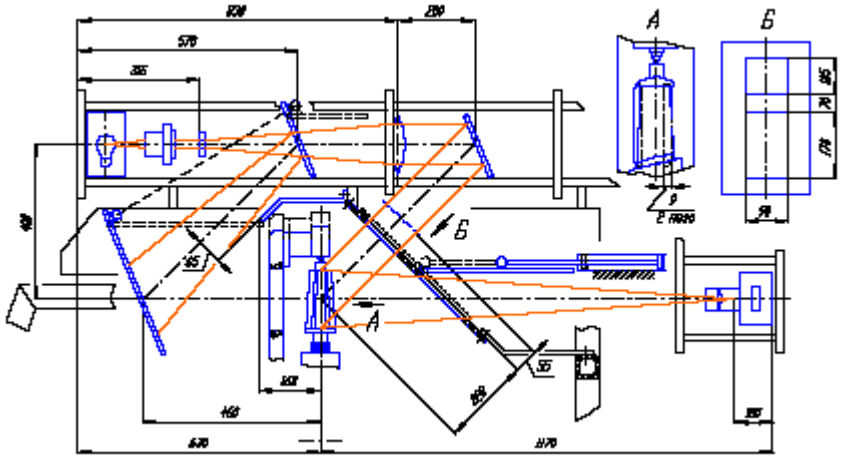


Рис. 12.9.

Основной координатной системой, относительно которой определяются геометрические параметры поверхности и ее погрешности, является система координат детали $(XYZ)_д$. Для пера лопаток турбинных двигателей ось $Z_д$ направлена вдоль оси лопатки, ось $X_д$ — по "ширине" поверхности пера лопатки и ось $Y_д$ образует правую систему координат и направлена по "толщине" пера. При этом выпуклую часть поверхности пера обычно называют "спинкой", а вогнутую "корытом".

На плоскость $Z_д0X_д$ под углом к данной плоскости падает параллельный пучок света, проходящий через дифракционную решетку (рис. 12.4) в направлении стрелки В (рис. 12.9). Благодаря этому на плоскости $Z_д0X_д$ образуется сетка из параллельных линий "зебра". Данные линии параллельны между собой, а также исходя из конструктивного расположения дифракционной решетки (рис. 12.10), остаются параллельными оси $0X_д$. Уравнения плоскостей параллельных пучков света, проходящих через дифракционную решетку и образующих параллельные линии ("зебру") на плоскости $Z_д0X_д$ можно представить в виде

$$ycos(\alpha) - zsin(\alpha) + c_i sin(\alpha) = 0, \qquad (12.14)$$

где c_i — координата z для каждого луча, расположенного в координатной плоскости $Z_д0Y_д$.

В плоскости, параллельной $Z_д0X_д$, на расстоянии OD (для рассматриваемой на рис. 12.9 системы это расстояние равно $OD=1120$ мм) располагается фотоматрица, на которую проецируется изображение "зебры".

Лучи, отраженные от поверхности, через оптическую систему с фокусным расстоянием f попадают на фотоматрицу. В этом случае принимается, что каждый луч, отраженный от поверхности, проходит через фокус (точку F) с координатами $(0, y_f, z_f)$, задаваемыми в системе координат $(XYZ)_д$. Для обеспечения симметричности преобразований принимается, что фокус располагается в плоскости $Z_д0Y_д$.

Параллельность плоскости расположения фотоматрицы и плоскости $Z_д0X_д$ должна создаваться специальной настройкой, обеспечивающей симметричность

изображения на фотоматрице точек, симметрично расположенных в плоскости $Z_д 0 X_д$, относительно перпендикуляра, опущенного из точки F на плоскость $Z_д 0 X_д$.

Рассмотрим сложную поверхность, располагаемую таким образом, чтобы отраженные от нее лучи полностью попадали в телекамеру. Параллельные лучи, проходящие через светлые линии дифракционной решетки, образуют плоские сечения, рассекающие поверхность и образующие линии, проецируемые на фотоматрицу (рис. 12.10).

Оптическая система позволяет определять координаты точек поверхности в системе координат $(XYZ)_д$. Отраженные от анализируемой поверхности линии "зебры" на плоскости фотоматрицы образуют искаженные линии "зебры". В данном случае рассматриваются непрерывные координаты линий в плоскости фотоматрицы, получаемые уже после обработки границ зон светлых и темных полос и их аппроксимации в соответствии с методикой, изложенной выше. Координаты точек на фотоматрице задаются в системе координат $(XYZ)_д$.

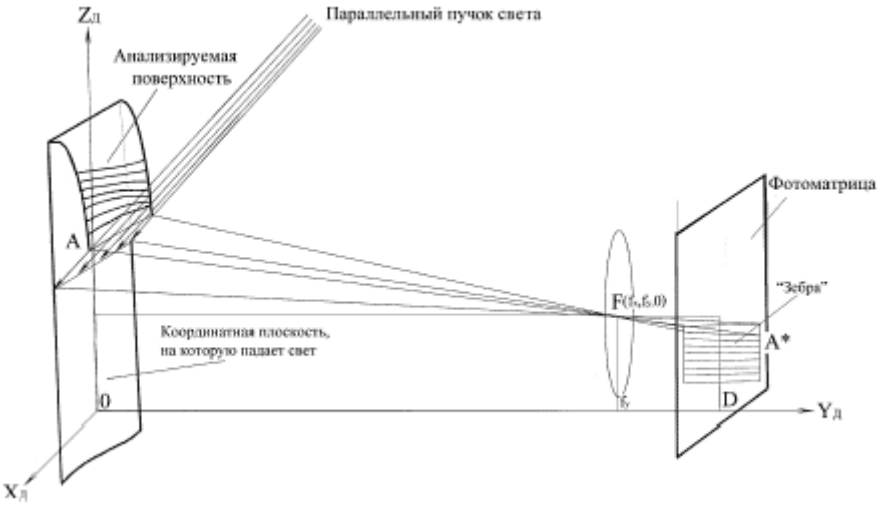


Рис. 12.10.

По координатам точек линий "зебры", определяемых в плоскости $Z_ф 0 X_ф$, определяются координаты точек данных линий на анализируемой поверхности (рис. 12.10) в системе $(XYZ)_д$. Рассмотрим методику определения координат для одной из точек линии поверхности, например, точки A (рис. 12.10), образованной пересечением плоскости параллельных лучей с поверхностью, по координатам данной точки (т. A^*) на фотоматрице — $A^*(x_A^*, y_A^*, z_A^*)$. Если данные координаты известны, то уравнение прямой, проходящей через две точки F и A^* в системе координат $(XYZ)_д$, принимает вид

$$\frac{(x-x_f)}{(x_A^*-x_f)} = \frac{(y-y_f)}{(y_A^*-y_f)} = \frac{(z-z_f)}{(z_A^*-z_f)}. \qquad (12.15)$$

Координаты точки A поверхности, соответствующие ее координатам на плоскости фотоматрицы, получаются пересечением прямой (12.15) с соответствующей плоскостью параллельных лучей (12.14). Совместное решение системы уравнений, полученной из (12.14) и (12.15),

$$\begin{cases} y\cos\alpha - z\sin\alpha + c_i\sin\alpha = 0 \\ (z_A^* - z_f)(y - y_f) = (y_A^* - y_j)(z - z_f) \\ (y_A^* - y_f)(x - x_f) = (x_A^* - x_j)(y - y_f) \end{cases} \qquad (12.16)$$

относительно x, y и z позволяет определить координаты т. A поверхности. С учетом того, что y_A^*=OD и x_f=0, получим

$$\begin{aligned} x_A &= \frac{x_A^*(D + (z_f + c_i)\operatorname{tg}\alpha)}{(z_A^* - z_f)\operatorname{tg}\alpha - (y_A^* - D)}, \\ y_A &= \frac{(z_A^* - z_f)D - (y_A^* - D)(z_f + c_i)}{(z_A^* - z_f)\operatorname{tg}\alpha - (y_A^* - D)}\operatorname{tg}\alpha, \\ z_A &= \frac{(z_A^* - z_f)(D + c_i\operatorname{tg}\alpha) - z_f(y_A^* - D)}{(z_A^* - z_f)\operatorname{tg}\alpha - (y_A^* - D)}. \end{aligned} \qquad (12.17)$$

Аналогично могут быть вычислены координаты всех точек линий "зебры" на анализируемой поверхности по их координатам в плоскости расположения фотоматрицы.

Контрольные вопросы:

Физические и математические основы оптических систем контроля?
Контроль качества поверхности (ее шероховатости)?
Контроль изделий сложной формы в составе технологических систем?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 6: Цепь обратной связи

План:

1. Прямая и обратная цепочки рассуждений
2. Основные понятия теории вероятностей
3. Нечеткая логика в экспертных системах

1. Прямая и обратная цепочки рассуждений

В экспертных системах механизм вывода позволяет на основе имеющихся фактов и правил выводить новые факты. В работе механизма вывода в зависимости от направления хода рассуждений и предметной области различают:

1) прямую цепочку рассуждений;
2) обратную цепочку рассуждений;
3) нечеткую логику, построенную на основе вероятности[17].

Сформулируем задачу в общем виде:

Имеет место ситуация и требуется предсказать ее последствия, например, во время движения у автомобиля перегрелся двигатель.

Можно разработать следующие правила:

ЕСЛИ двигатель перегрелся, ТО мотор заглохнет; ЕСЛИ мотор заглохнет, ТО это приведет к определенным денежным затратам и позднему возвращению домой. В данном случае используется прямая цепочка рассуждений, последовательность которых называется **прямой цепочкой** потому, что отправной точкой здесь служит возникшая ситуация (перегрев двигателя), часть же правила выполняется лишь в том случае, если удовлетворяется условная часть правила ЕСЛИ, т.е. сначала имеет место ситуация, а потом делаются выводы. В **обратной цепочке**, наоборот, выводы известны, а причины необходимо найти, например, автомобиль не трогается с места. В данном случае используются следующие правила:

1) ЕСЛИ автомобиль не заводится и сел аккумулятор, ТО не подается ток в стартер;
2) ЕСЛИ в стартер не подается ток, ТО автомобиль не тронется с места.

Известный результат (автомобиль не трогается с места) влечет за собой цепочку рассуждений, которая приведет нас к вызвавшим его причинам. Эта цепочка и называется обратной. Причины возникают раньше следствий, поэтому в процессе обратной цепочки просматриваются логические выводы, устанавливаются условия, которые к ним привели, и определяется, связаны ли эти условия с предыдущими логическими выводами.

Таким образом, обратная цепочка рассуждений всегда начинается со следствия, т.е. с части ТО. Она начинается с уже происшедшего события и идет к его истокам[26].

Система, реализующая прямую цепочку рассуждений на основании имеющихся условий, делает возможные логические выводы, а система, реализующая обратную цепочку рассуждений, по имеющимся выводам ищет необходимые для них условия.

2. Основные понятия теории вероятностей

Во многих эвристических правилах, т.е. правилах, сформулированных экспертом, лежит вероятность появления определенного события, вычислить кото-

рую может только эксперт. Другими словами, эксперт делает обоснованные предположения в своей проблемной области. В действительности это означает, что существуют статистические данные, позволяющие делать какие-либо предположения.

Это могут быть, например, медицинские диагнозы, которые врач ставит на основании своих наблюдений над пациентом. Опыт врача во многих случаях с большой точностью позволяет определить заболевание пациента. Конечно, есть вероятность, что врач ошибся, поэтому часто рассматриваются и другие диагнозы.

Компания, занимающаяся добычей нефти, может на основе данных, полученных в результате проведенных исследований, сделать вывод, что найдено место рождение нефти и вынести решение - бурить скважину. Однако может оказаться, что в скважине нефти нет.

Байес разработал вероятностную методику, основанную на утверждении, что какое-то событие произойдет, потому что раньше уже произошло какое-то другое событие. Теория Байеса входит в теорию вероятностей в раздел, названный «Условная вероятность».

В экспертных системах широко применяются статистические решения, опирающиеся на теорию Байеса, использование которой описано ниже[24].

Теория вероятностей изучает случайные события. Весьма часто человек, сам того не замечая, высказывает предположение или делает вывод пользуясь терминологией теории вероятностей. Например, «Я на 80% уверен, что все в порядке».

Вероятность можно определить следующим образом.

$$P = \frac{\text{Число экспериментов, исходом которых является какое-то событие}}{\text{Общее число экспериментов}}$$

Байес занимался разработкой теории условной вероятности.

Условная вероятность – это вероятность наступления какого-то события *S* при условии, что уже наступило какое-то другое событие *e*. Условная вероятность обозначается *P* (*S/e*). Вероятность наступления двух событий установливается следующим образом:

*P (e и S)= P(S/e)*P(e).*

Уравнение читается так: есть вероятность того, что произойдут два события: *e* и *S,* причем e произойдет первым, что равно вероятности наступления события *S*, если известно, что произошло событие *e*, умноженное на вероятность появления события *e*. Рассмотрим пример использования данного уравнения. Предположим, что из набора букв *IIOO* случайным образом выбирается либо буква *I,* либо *O*. Применяя уравнение условной вероятности, можно вычислить вероятность того, что в двух попытках сначала попадется буква *O*, а затем *I*. Поставив в уравнение *O* и *I*, получим:

*P(O и I) = P (I/O)*P(O).*

Вероятность выбора буквы *О*, т.е. *P(O)* равна *2/4*, так как всегда в наборе две буквы *I* и две-буквы *О*. Вероятность того, что после буквы *О* будет выбрана *I, P(I,O) = 2/3*. Чтобы понять, почему это так, рассмотрим случай, когда буква *О* уже выбрана и остались три буквы *IIO*. Вероятность выбора буквы *I* составляет 2/3, поскольку из трех оставшихся букв две - *I*. Вероятность *P(I и O)* равна :

*P(I/O)*P(O) = 2/3*2/4 = 1/3.*

В экспертных системах используется еще одно уравнение условной вероятности: *P(S) = P(S/e)*P(e)+P(S/not e)*P(not e)*.

Уравнение читается так: вероятность появления события *S, P(S),* равна вероятности возникновения события *S* при условии появления события *e, P(S/e),* умноженной на вероятность появления события *e, P(e)*, плюс вероятность появления события *S* при условии, что событие *e* не произошло, *P(S/not e),* умноженная на вероятность, что событие *e* не произошло, *P(not e).*

3. Нечеткая логика в экспертных системах

Рассмотрим другой аспект теории вероятностей. Не всегда можно описать событие с помощью точно определенных правил. Например, можно сказать, что у человека легкое недомогание, если температура больше 37 градусов, но меньше 38. При большей же температуре заболевание может оказаться серьезным. Люди не всегда могут точно ответить на вопросы. Можно ли узнать какая у человека температура, если он говорит, что слегка заболел? Скорее всего нет.

Такие слова как «высокий», «горячий» и «легкий», представляют собой лингвистические переменные, которые нельзя определить одним значением. Лингвистическая переменная может принимать различные значения из некоего интервала, границы которого могут меняться в зависимости от обстоятельств. Например, границы интервала для лингвистической переменной «холодный» могут меняться в зависимости от того, идет ли речь о зиме или весне. Использование этих понятий при формулировании правил называется **нечеткой логикой.**

Применяя лингвистические переменные, можно вычислить значения некоторых вероятностей, не обременяя пользователя лишними вопросами. Для этого необходимо несколько конкретизировать лингвистические переменные. Пользователю экспертной системы нужно позволить добавлять к этим переменным определения, например «маленький» или «средний», однако экспертная система должна точно знать, что именно под этим подразумевается[24,25].

В нечеткой логике используются коэффициенты уверенности (КУ). Поскольку эвристические правила ЕСЛИ - ТО исключительно основываются на человеческом опыте, никогда нельзя с полной определенностью сказать, что они верны. Пользователь экспертной системы также не может быть полностью уверен, что значения, которые он присваивает переменным, абсолютно корректны. Например, правило:

ЕСЛИ процентные ставки = падают и налоги уменьшаются,

ТО уровень цен на бирже = растет

Верно не всегда и потому ему можно приписать значение некоторой уверенности. КУ может иметь значение от -1 до 1. Отрицательное значение КУ указывает на степень уверенности в том, что правило не верно, а положительное же значение - что правило верно. Таким образом, КУ, равный +1, указывает на полную уверенность в том, что правило верно, а -1 - на полную уверенность в некор-

ректности данного правила. Безусловно, правила, для которых КУ равно -1, рассматривать нет смысла.

Пусть приведенное правило имеет КУ, равный 0,9, и нельзя утверждать, что процентные ставки падают, т.е. первому условию правила назначен КУ, равный 0,6. Кроме того, допустим, что налоги колеблются (то увеличиваются, то уменьшаются) и потому предположить уменьшение налогов можно при условии, если КУ равен 0,8. Тогда это правило можно записать так:

ЕСЛИ процентные ставки = падают (КУ =0,6) И
налоги уменьшаются (КУ =0,8),
ТО уровень цен на бирже = растет (КУ правила = 0,9)

Коэффициент уверенности, что уровень цен на бирже будет расти может быть подсчитан следующим образом: выбирается минимальный КУ для условий части ЕСЛИ правила, и умножается на КУ всего правила приведенного примера:

(min(0.6, 0.8)) * 0.9 = 0.6 * 0.9 = 0.54.

Следовательно, при КУ = 0.54 можно сказать, что уровень цен на бирже будет падать.

Во многих случаях изначально заданы граничные значения коэффициента уверенности. Логический вывод считается верным только в том случае, если его КУ превышает заранее заданные граничные значения. Работа с базой знаний продолжается до тех пор, пока значение коэффициента уверенности логического вывода больше граничного значения. В процессе работы выполняются определенные вычисления. Предположим, что для частного логического вывода КУ равно 0,4. Это значение запоминается. Затем оно сравнивается с граничным значением КУ (допустим, что оно равно 0,8). Запомненное значение оказалось меньше граничного, и, следовательно, работа с базой знаний продолжается. Если при работе с базой знаний встретился тот же самый логический вывод, то КУ для нового правила умножается на 1 минус значение запомненного ранее КУ и результат прибавляется к запомненному ранее КУ. Значение КУ, равное 1, свидетельствует об абсолютной уверенности в правильности вывода. Затем вновь запомненное значение КУ сравнивается с граничным и если оно больше, то выполняется логический вывод, в противном случае, работа с базой знаний продолжается. Вышесказанное можно записать следующим образом:

Запомненный КУ = Ранее запомненный КУ +
(1 - Ранее запомненный КУ) * КУ нового правила

Например:

Граничное значение КУ = 0,8

Правило: ЕСЛИ А, ТО В(КУ = 0,6)

Запомненный КУ: 0,6

Новое правило: ЕСЛИ С, ТО В (КУ = 0,7)

Запомненный КУ = 0,6 + (1 - 0,6) * 0.7 = 0,88 (граничные значения превышены, и выполняется вывод).

Контрольные вопросы:

1. Прямая и обратная цепочки рассуждений?
2. Основные понятия теории вероятностей?
3. Нечеткая логика в экспертных системах?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 7: Эмоциональные оценки полученных результатов

План:

1. Основные направления интеллектуализации прикладных систем и систем принятия решений
2. Методы искусственного интеллекта в прикладных системах и системах принятии решений
3. Интеллектуальные информационные технологии в прикладных системах и системах принятия решений

1. Основные направления интеллектуализации прикладных систем и систем принятия решений

В настоящее время возникает потребность в создании прикладных систем различного назначения, поддержки принятия решений (ПР) и планирования действий в динамически меняющейся ситуации в условиях неполных или нечетких данных с использованием экспертных знаний. Например, в процессе принятия управленческих решений лицу, принимающему решение (ЛПР), приходится учитывать большое количество показателей, критериев, факторов, влияющих на поставленную в задаче цель.

Для эффективного решения подобных задач целесообразно использовать системы принятия решений (СПР) в совокупности с интеллектуальными компонентами. Это требует на стадии разработки системы совместного применения методов принятия решений и методов искусственного интеллекта (ИИ). Обобщенная структура такой системы представлена на рис.1.1.

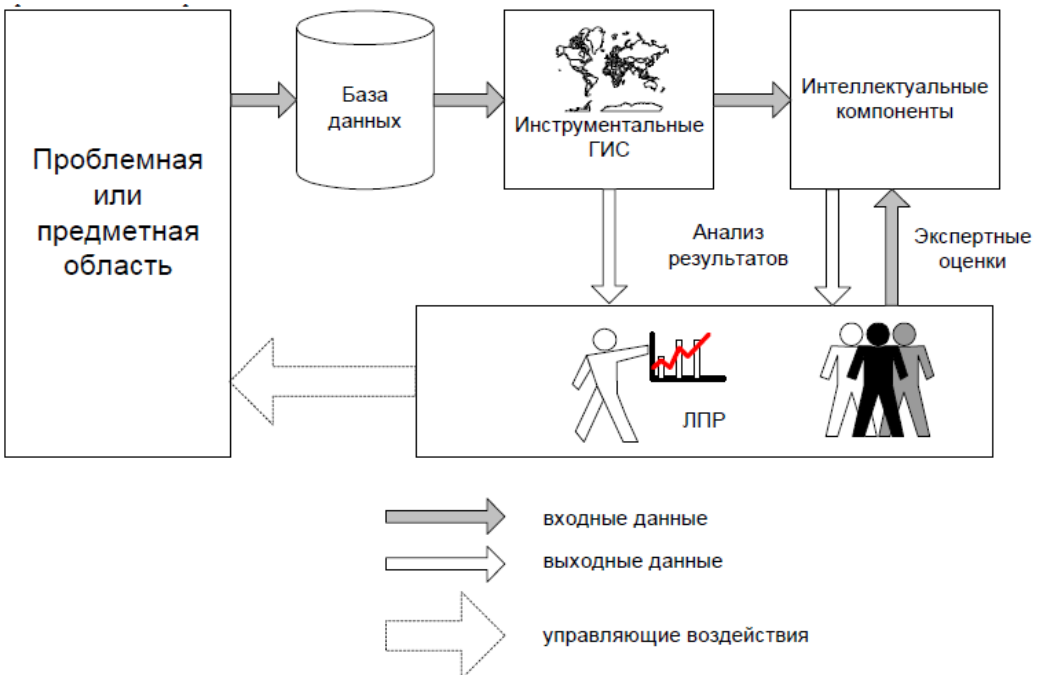


Рис. 1.1. Обобщенная структура системы, использующей интеллектуальные компоненты

Разработка интеллектуальных компонент СПР ведется с использованием интеллектуальных информационных технологий [2]. Изначально под интеллектуальной информационной технологией подразумевались средства обработки ин-

формации, не связанные с алгоритмическими вычислениями. Сегодня благодаря многочисленным публикациям в области искусственного интеллекта термин «Интеллектуальная информационная технология» трактуется одинаково.

2. Методы искусственного интеллекта в прикладных системах и системах принятии решений

До недавнего времени постановка и решение задач принятия решений опиралась на традиционные математические модели и методы. Но в ряде задач ПР зависимости настолько сложны, что не допускают своего обычного аналитического представления, или из-за неполноты информации некоторые элементы модели остаются неизвестными.

Это объясняется следующими причинами [13]:

• Не все цели управления объектом могут быть выражены в виде количественных соотношений.

• Между рядом параметров, оказывающих влияние на процесс принятия решений, не удается установить точных количественных зависимостей.

• Процесс принятия решений является многошаговым, и содержание каждого шага не может быть заранее однозначно определено.

• Существующие способы описания объектов и протекающих в них процессов приводят к столь громоздким конструкциям, что их практическое использование невозможно.

В этих случаях средства ИИ позволяют более успешно справиться с неполнотой информации и отсутствием определенности.

Такие объекты называют по-разному: плохо определенные *или* слабо структурированные, организационные. Примерами таких объектов управления являются, например, экономические и социальные объекты. Но независимо от названия эти новые объекты обладают рядом присущих только им свойств. Перечислим основные их свойства.

1. **Уникальность**. Каждый объект обладает такой структурой и функционирует так, что система управления им должна строиться с учетом всех его качеств и к нему нельзя применить какую-либо стандартную процедуру управления. Это обстоятельство резко удорожает построение системы управления, ибо фактически нужно создавать столько систем управления, сколько объектов мы хотим автоматизировать.

2. **Отсутствие формализуемой цели существования**. Не для всех объектов (даже созданных человеком) можно четко сформулировать цель их существования. При управлении городами, отраслями народного хозяйства, регионами, экосистемами возникают затруднения при попытке четко сформулировать цель существования этих объектов. Это приводит к большим сложностям в формировании критерия управления, ибо критерий управления в традиционных системах управления был тесно связан с целью существования объекта. Критерий управления самолетом был основан на достижении им своей цели существования – перевозки грузов и людей по воздуху, управления производством телевизоров – на повышении качества выпускаемой продукции. Поэтому в различных системах управления, создаваемых для объектов нового класса, очень часто можно наблюдать реализацию различных критериев управления.

3. **Отсутствие оптимальности.** Следствием того, о чем говорилось в предшествующих двух пунктах, является неправомочность постановки для объектов новой природы классической задачи оптимизации. Из-за отсутствия цели существования (в рамках теории управления) для рассматриваемых объектов нельзя построить объективный критерий управления. Критерий управления становится субъективным, зависящим от лица, принимающего решения.

При решении о выборе тех или характеристик будущего изделия, при принятии решении о структуре и способах функционирования проектируемой системы невозможно оценить правильность выбора. Отсюда следует, что в этих случаях невозможно говорить об оптимальности получаемых решений. Качество создаваемой системы для управления объектами новой природы может оцениваться только субъективно самим ЛПР или их коллективом. Поэтому здесь уместнее говорить о целесообразности результата управления, а не его оптимальности.

4. **Динамичность**. Существует обширный класс объектов управления, которые с течением времени изменяют свою структуру и функционирование, т.е. объекты эволюционируют во времени. Эта динамичность должна быть учтена в системах управления подобными объектами в виде эволюции процесса управления.

5. **Неполнота описания. Первая причина неполноты описания** объекта состоит в том, что эксперты не могут оценить тот уровень полноты описания, который нужен специалисту по управлению. Как правило, эксперты, знающие объект управления, не в состоянии сразу же обеспечить информацию, которой бы хватило для создания системы управления объекта.

Вторая причина неполноты описания объекта – незнание некоторых сторон функционирования самими специалистами. Могут возникнуть ситуации, никогда не встречавшиеся ранее. К таким ситуациям относятся всевозможные аварийные ситуации.

Третья причина неполноты описания объекта – отсутствие четкого понимания функционирования объекта у специалиста–технолога. Выдавая управленцу большое количество информации, технолог тем не менее не сообщает ему самой главной, по которой сам принимает решение.

Четвертая причина неполноты описания объекта – многие особенности функционирования объекта, а иногда и его структуры не могут быть описаны количественно.

Они допускают лишь качественное, словесное описание. Переход от качественных описаний к некоторым формальным представлениям должен производиться управленцем, который не всегда в состоянии решить такую сложную проблему.

Активная природа элементов управляемого объекта. Во многих объектах управления люди являются элементами их структуры. Это так называемые организационные системы. В отличие от всех других элементов, образующих объекты, люди функционируют в нем с учетом своих личных интересов и целей. Их цели и интересы могут противоречить целям управления; они могут оказывать обратное воздействие на саму систему управления.

Индивидуальное поведение таких элементов структуры практически невозможно учесть при создании системы управления, и требуются специальные приемы для учета их воздействия на функционирование объекта управления.

Использование интеллектуальных компонент, описанных в подразд. 1.2, приводит к новым технологиям в принятии решений.

3. Интеллектуальные информационные технологии в прикладных системах и системах принятия решений

Достижения в области информационных технологий и, в частности, в искусственном интеллекте вызвали появление новых средств принятия решений, управления и оптимизации сложных систем на основании принимаемых решений. Современная интеллектуальная информационная технология включает следующие основные направления [8]:

1) **нейросетевые алгоритмы** обработки информации и нейрокомпьютеры;

2) **эволюционные (генетические) алгоритмы;**

3) **системы,** реализующие методы, **основанные на знаниях;**

4) обработка **нечеткой информации и нечеткий вывод;**

5) **многоагентные системы** как средство распараллеливания поиска и обработки информации в интеллектуальных системах.

Кратко охарактеризуем каждое из этих направлений.

1. Искусственные **нейронные сети** применяются для решения целого класса задач, где используются не уравнения, описывающие функционирование объекта, и не правила, как в традиционных экспертных системах, а опыт. Нейронные сети обладают способностью к обобщению информации, к обучению и самообучению на основе собственного опыта функционирования и т.п. В интеллектуальных компонентах сложных систем, в том числе иГИС, применяются несколько типов искусственных нейронных сетей: многослойный перцептрон, сеть Кохонена, сеть Хопфилда и др. Эти модели используются для автоматизации принятия решений в следующих ситуациях:

• когда невозможно построить алгоритм или логическое исчисление из–за сложности учета всех сочетаний факторов,

• когда сложно формализовать закономерности, связывающие условия задачи с результатом.

В мощных интеллектуальных системах могут совмещаться нейросетевые и логические механизмы принятия решений.

В качестве области практического приложения нейросетей в интеллектуальных компонентах ГИС можно указать системы искусственного зрения и обработки изображений, системы аэрокосмической съемки, подсистемы автоматизации планирования действий в мониторинговых системах и системах предупреждения чрезвычайных ситуаций.

2. Интеллектуальные компоненты ГИС могут быть реализованы и на базе **эволюционных (генетических) алгоритмов,** которые используются для поиска рациональных решений.

Эволюционные алгоритмы основаны на дарвиновских принципах генерации, тестирования и отбора перспективных популяций. Эволюционные алгоритмы оперируют с популяцией индивидов $P(t) = \{ xt\ 1\ , \ldots, tnx \}$, где t – номер итерации. Каждый индивид представляет собой некоторое возможное решение из множества допустимых решений S.

Каждое решение $t\ xi$ оценивается некоторой мерой его пригодности. На итерации $t + 1$ формируется новая популяция путем отбора более пригодных инди-

видов (шаг селекции). Некоторые члены этой новой популяции подвергаются преобразованиям (шаг изменений) с помощью генетических операторов. Это делается для того, чтобы образовать новые решения. Преобразования бывают двух типов. Первый тип – это унарные (одноместные) преобразования $mk : s \rightarrow S$ (типа мутаций), которые приводят к появлению новых индивидов путем малых изменений одного индивида. Второй тип преобразований – это преобразования перекрестного типа $cj : Sn \rightarrow S$, которые порождают новые индивиды путем комбинирования составных частей – генов нескольких индивидов. Через несколько поколений могут возникнуть решения, близкие к оптимальным. Генетические алгоритмы используют параллельную обработку множества альтернативных решений и организуют поиск наиболее перспективных из них.

В качестве примера применения эволюционного алгоритма можно привести задачу поиска графа, удовлетворяющего некоторым требованиям (оптимальный маршрут оптимальная топология коммуникационной сети). Такая задача возникает при реализации механизма вывода в семантических сетях. Граф рассматривается как индивид эволюционного алгоритма. Начальными данными эволюционной программы служит начальная популяция графов, порождаемая случайным образом или эвристически. Задается оценочная функция, которая выражает пригодность каждого графа и формализует отношение предпочтения (лучше, хуже) на множестве индивидов. Мутационные операторы осуществляют преобразования графа. Перекрестные операторы комбинируют структуры двух или более графов. Если искомый граф должен быть связным и ациклическим (т. е. деревом), − некоторый мутационный оператор может удалять ребра и для связывания двух возникших новых подграфов всякий раз добавлять некоторое новое ребро. С помощью оценочной функции устанавливается, не нарушено ли в результате мутаций свойство графа быть деревом. Если порожденный граф этому свойству не удовлетворяет, на шаге селекции необходимо отбросить графы-поддеревья.

Контрольные вопросы:

1. Основные направления интеллектуализации прикладных систем и систем принятия решений?
2. Методы искусственного интеллекта в прикладных системах и системах принятии решений?
3. Интеллектуальные информационные технологии в прикладных системах и системах принятия решений?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 8: Коррекция управления

План:

1. Процесс создания прототипа
2. Система управления
3. Система оптического контроля

1. Процесс создания прототипа

Обычная ошибка разработчиков при создании прототипа состоит в том, что процесс приобретения знаний откладывают до полного понимания структуры БЗ и всех тестовых примеров. Тем самым эта наиболее трудоемкая часть работы отодвигается на поздние этапы. Процесс накопления знаний позволяет уточнить используемые понятия и отношения, поэтому необходимо начинать приобретение знаний, как только составлены (или выбраны) ИС, позволяющие работать с простейшим представлением знаний и простейшими управляющими структурами. Такой подход позволяет как можно раньше начать выполнение отдельных подзадач и обнаружить, что в ряде случаев для их решения необходимы дополнительные знания. Иными словами, первый прототип экспертной системы (ЭС-1) должен появиться через 1 - 3 месяца после начала работы. Разработка прототипа является чрезвычайно важным шагом в создании ЭС. Некоторые фрагменты прототипа могут войти в окончательную версию ЭС, но не это является наиболее важной целью создания прототипа. Главное, чтобы прототип обеспечил проверку адекватности идей, методов и способов представления, выбранных при построении данной ЭС, решаемым задачам.

Создание первого прототипа должно подтвердить, что выбранные методы решений и способы представления пригодны для успешного решения по крайней мере ряда задач из области экспертизы. При разработке первого прототипа обычно оставляют в стороне вопросы, требующие значительных трудозатрат: построение сложных моделей; учет сложных временных, причинных и модальных отношений; понимание намерений пользователей (экспертов); моделирование рассуждений, содержащих неточные понятия.

Итак, можно сделать вывод, что в первом прототипе реализуется (используется) простейшая процедура вывода. При его разработке основная цель состоит в том, чтобы получить решение задачи, не заботясь пока об эффективности. После разработки первого прототипа необходимо расширить круг задач, решаемых системой, для того, чтобы собрать пожелания и замечания, которые будут учтены в очередной версии системы (ЭС-2). Для этого осуществляется развитие ЭС-1 путем добавления:

• средств для исследования базы знаний и последовательностей выводов, генерируемых системой (что обеспечивает прозрачность и понимаемость системы разработчиком);

• средств для сбора замечаний пользователей;

• средств хранения библиотеки задач, решенных системой. Библиотека необходима для того, чтобы при каждой модификации системы можно было проверить, решаются ли все старые задачи и в новой версии.

В ходе приобретения знаний инженер по знаниям должен получить знания от эксперта, структурировать их и представить в виде, понятном экспертной системе. Процесс извлечения знаний сложен и длителен, так как эксперт часто или не осознает, какими знаниями он пользуется, или не может их вербализовать (содержательно выразить). Для достижения эффективного функционирования экспертной системы необходимо осуществить структурирование знаний. Наиболее важным средством для структурирования знаний является иерархия классов, описывающих понятия промежуточного уровня. Во многих случаях эти понятия могут явно не упоминаться (а возможно, и не осознаваться) экспертом. Задача инженера по знаниям - выделить такие понятия, обнаружив сходные действия эксперта при обработке различных ситуаций.

При представлении правил в виде, понятном ЭС, особое внимание следует уделять трем ситуациям: некоторое правило слишком громоздко; имеется много похожих правил; используются частные, а не общие правила. Громоздкость правила может объясняться тем, что в нем отражено несколько фактов из данной проблемной области. Если это так, то правило надо разбить на несколько более мелких. Вторая ситуация имеет место тогда, когда в проблемной области существует понятие, явно не указанное экспертом, а возможно, и не имеющее имени. В этом случае новое понятие необходимо ввести в явном виде, присвоить ему специальное имя и, используя это понятие, сформулировать одно правило взамен группы подобных. Третья ситуация имеет место тогда, когда эксперт (разработчик) не использует возможности, предоставляемые объектно-ориентированным программированием, позволяющим скрыть специфику объектов в иерархии классов и ссылаться в правилах на классы, а не на конкретные объекты.

Выполнение экспериментов с версией ЭС-2 и анализ результатов их прогонов позволяют выявить недостатки системы и разработать средства для их устранения. Этот итеративный процесс может продолжаться еще несколько месяцев и зависит от сложности проблемной области, от гибкости выбранного представления и степени соответствия управляющего механизма решаемым задачам (возможно, потребуется разработка ЭС-3 и т.д.).

В целом *итеративная разработка* заключается в подходе к реализации системы как серии удачных приближений прототипов к конечной цели, а не как к единой, монолитной, интегрированной системе. Итеративная разработка особенно эффективна при создании систем с недостаточно четко определенными спецификациями, к которым прежде всего относятся экспертные системы. Поскольку подобные проекты обычно недостаточно проработаны с точки зрения системного анализа, разработчики обычно обнаруживают новые требования к системе после начала проекта. Если принят итеративный подход к разработке, то на адаптацию системы и коррекцию дальнейшего плана работ требуются относительно небольшие затраты. С другой стороны, при попытке сразу разработать целостную систему обнаружение новых требований к системе в процессе разработки может поставить под сомнение возможность ее реализации.

2. Система управления

Система управления *робота-станка* построена на базе встроенного персонального компьютера, выполняющего в реальном масштабе времени следующие операции:

- планирование траекторий перемещения *манипуляторов* на основе данных с чертежа поверхности;
- управление *исполнительными приводами*;
- обработка информации с датчиков перемещения *манипуляторов*;
- обработка информации с системы контроля геометрических размеров обрабатываемой поверхности и коррекция траектории перемещения *манипуляторов*.

Для формирования траектории перемещения инструмента необходимо иметь описание обрабатываемой поверхности. Так, для описания поверхности пера лопаток турбин, задаваемых координатами опорных точек, используются в зависимости от требуемой точности либо *сплайн-функции*, либо многомерные полиномы. В случае использования полинома второго порядка двух переменных независимо от типа полинома описание приводится к *степенному ряду* двух переменных

$$y_д=A_1z_д^2x_д^2+A_2z_д^2x_д+A_3z_дx_д^2+A_4z_д^2+A_5x_д^2+A_6z_дx_д+A_7z_д+A_8x_д+A_9, \quad (9.2)$$

где коэффициенты A_k вычисляются через постоянные коэффициенты описываемого полинома. Полное представление сложной поверхности координатамиопорных точек, а также методика описания поверхности рассмотрены в лекции 11. Для автоматизированного программирования траектории перемещения инструмента по данным с чертежа обрабатываемой поверхности разработана применительно для IBM PC Система Автоматизированного Программирования (САП).

Управление *исполнительными приводами* в первую очередь состоит в *решении обратной задачи* о положениях, которая для механизмов относительного манипулирования представляет определение обобщенных координат *манипуляторов* детали $q_д$ и инструмента $q_и$ по информации о перемещении инструмента относительно детали (рис. 9.1).

При обработке поверхности программная траектория перемещения инструмента относительно обрабатываемого изделия для каждой i -й точки поверхности задается элементами матрицы $[4 \times 4]^д A_i$, которая определяет положение режущей кромки инструмента в системе координат $(X,Y,Z)_д$ (рис. 9.5).

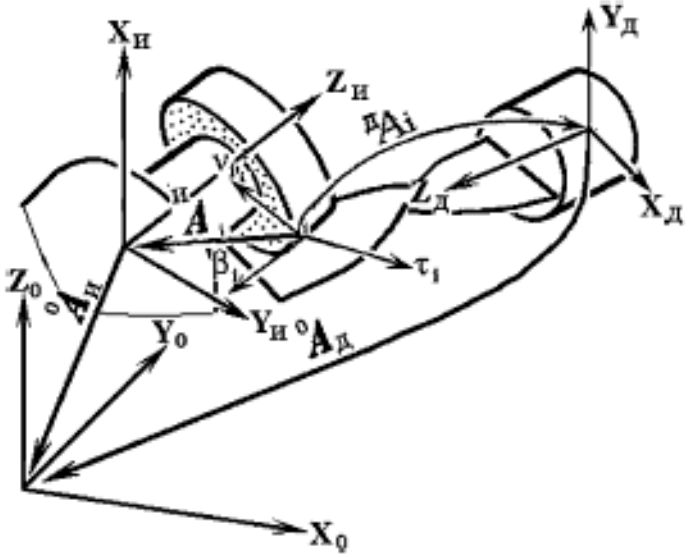


Рис. 9.5.

Для вывода уравнений кинематики и динамики механизмов используется аппарат однородных матриц и матричных преобразований, предложенных Ж.Денавитом и Р.Хартенбергом [105]. Основой данного математического аппарата являются матрицы [4 x 4], описывающие преобразование координат из одной системы (k+1) в другую (k)

$$^{k}A_{k+1} = \begin{bmatrix} ^{k}C_{k+1} & ^{k}R_{k+1} \\ 0^{T} & 1 \end{bmatrix}$$

где

$$^{k}C_{k+1} = \begin{bmatrix} \cos(\widehat{X_k X_{k+1}}) & \cos(\widehat{X_k Y_{k+1}}) & \cos(\widehat{X_k Z_{k+1}}) \\ \cos(\widehat{Y_k X_{k+1}}) & \cos(\widehat{Y_k Y_{k+1}}) & \cos(\widehat{Y_k Z_{k+1}}) \\ \cos(\widehat{Z_k X_{k+1}}) & \cos(\widehat{Z_k Y_{k+1}}) & \cos(\widehat{Z_k Z_{k+1}}) \end{bmatrix}$$

- подматрица *направляющих косинусов*, или матрица поворота осей системы координат $(XYZ)_{k+1}$ относительно осей $(XYZ)_k$;

$^{k}R_{k+1}=[x_{k+1}y_{k+1}z_{k+1}]^{T}$ - вектор, определяющий положение точки 0_{k+1} в системе координат XYZ_k;

0^{T} - подматрица преобразования перспективы;

1 - глобальный масштабирующий множитель.

Перемещение выходного звена *манипулятора* определяется положением системы координат $(XYZ)_д$ в неподвижной системе координат $(XYZ)_0$, представляемой матрицей [4 x 4] $^{0}A_д$ (рис. 9.3). Математически условие обработки поверхности состоит в обеспечении фундаментального матричного уравнения для каждой точки (i -й) траектории движения инструмента относительно детали

$^{0}A_i(q_и)=^{0}A_д(q_д)^{д}A_i$,

которое преобразуется к виду

$(^{0}A_д(q_д))^{-1}*^{0}A_i(q_и)=^{д}A_i$. (9.4)

Приравнивая три элемента 4-го столбца и три элемента, не принадлежащие одной строке и одному столбцу из 1-го, 2-го и 3-го столбца, в левой и правой частях уравнения (9.4), получим систему из шести уравнений

$F(q_и, q_д)=U$, (9.5)

которая является исходной системой и ее решение относительно $q_и$ и $q_д$ должно быть заложено в системе управления. Решение данного *трансцендентного уравнения* в явном виде определяется видом уравнения. В лекции 10 решение данного уравнения рассмотрено в линейном виде, когда в каждый момент времени рассчитываются приращения обобщенных координат, которые последовательно прибавляются к начальному значению.

3. Система оптического контроля

Система оптического контроля геометрии поверхностей сложной формы, разработанная в НПО"Луч", предназначена для определения отклонений параметров обрабатываемой поверхности от заданных и отслеживания ее качества. *Блок-схема* системы управления *робота-станка*, включающая систему контроля поверхности, приведена на рисунке 9.6, где система планирования траектории на основе данных чертежа и реальных геометрических размеров обработанной поверхности формирует промежуточные координаты $R_{прог}$.. Система управления манипуляторами на основе $R_{прог}$. формирует управляемые координаты на исполнительные приводы $q_и$ и $q_д$ и осуществляет управление приводами.

Проектирование технологических машин нового поколения для обработки сложных поверхностей включает следующие основные этапы:

Рис. 9.6. Система управления манипуляторами

- Выбор кинематической схемы технологической машины на основе функционального назначения *манипуляторов*, обеспечивающих требуемые перемещения инструмента относительно детали и независимые транспортные перемещения детали и инструмента.
- Проектирование системы управления на базе универсальной ЭВМ, имеющей необходимый объем памяти для планирования траекторий перемещения *манипуляторов* и обеспечения управления приводами по информации с датчиков положения *манипуляторов*.
- Синтез управления *исполнительными приводами* с учетом *технологического процесса* как элемента системы.
- Разработка программного обеспечения и планирование траекторий перемещения *манипуляторов*, в целях получения требуемых *показателей качества*.

Технологические машины, построенные на механизмах относительного манипулирования и оснащенные современными управляющими вычислительными системами, позволяют выполнять механическую обработку деталей со сложными поверхностями, имеют существенно меньшую металлоемкость в сравнении с оборудованием, применяемым в настоящее время для этих целей, позволяют одними и теми же механизмами выполнять транспортные и обрабатывающие операции.

Контрольные вопросы:

1. Особенности процесса создания прототипа?

2. Коррекция параметров системы управления?

3. Система оптического контроля?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 9: Синтез новых целей и их организация

План:

1. Неформализованные задачи
2. Сущность механизма афферентного синтеза
3. Структура процесса получения решения

1. Неформализованные задачи

ЭС предназначены для так называемых неформализованных задач, т.е. ЭС не отвергают и не заменяют традиционного подхода к разработке программ, ориентированного на решение формализованных задач. Следуя А.Ньюэллу и М.Саймону [13], к неформализованным (ill-structured) будем относить такие задачи, которые обладают одной или несколькими из следующих характеристик:

• задачи не могут быть заданы в числовой форме;

• цели не могут быть выражены в терминах точно определенной целевой функции;

• не существует алгоритмического решения задач;

• алгоритмическое решение существует, но его нельзя использовать из-за ограниченности ресурсов (время, память).

Неформализованные задачи обычно обладают следующими особенностями:

• ошибочностью, неоднозначностью, неполнотой и противоречивостью исходных данных;

• ошибочностью, неоднозначностью, неполнотой и противоречивостью знаний о проблемной области и решаемой задаче;

• большой размерностью пространства решения, т.е. перебор при поиске решения весьма велик;

• динамически изменяющимися данными и знаниями.

Следует подчеркнуть, что неформализованные задачи представляют большой и очень важный класс задач. Многие специалисты считают, что эти задачи являются наиболее массовым классом задач, решаемых ЭВМ.

Экспертные системы и системы искусственного интеллекта отличаются от систем обработки данных тем, что в них в основном используются символьный (а не числовой) способ представления, символьный вывод и эвристический поиск решения (а не исполнение известного алгоритма).

Специфика приложений экспертных систем по сравнению с другими системами искусственного интеллекта состоит в следующем. Экспертные системы применяются для решения только трудных практических (не игрушечных) задач. По качеству и эффективности решения экспертные системы не уступают решениям эксперта-человека. Решения экспертных систем обладают *"прозрачностью"*, т.е. могут, быть объяснены пользователю на качественном уровне (в отличие от решений, полученных с помощью числовых алгоритмов, и в особенности от решений полученных статистическими методами). Это качество экспертных систем обеспечивается их способностью рассуждать о своих знаниях и умозаключениях. Экспертные системы способны пополнять свои знания в ходе взаимодействия с экспертом. Необходимо отметить, что в настоящее время технология экспертных систем используется для решения различных типов задач (интерпретация, предсказание, диагностика, планирование, конструирование, контроль, отладка, ин-

структаж, управление) в самых разнообразных проблемных областях, таких, как финансы, нефтяная и газовая промышленность, энергетика, транспорт, фармацевтическое производство, космос, металлургия, горное дело, химия, образование, целлюлозно-бумажная промышленность, телекоммуникации и связь и др.

2. Сущность механизма афферентного синтеза

В настоящее время сущность механизма афферентного синтеза многими учеными представляется в виде структурной схемы функциональной системы живого организма (рис. 12.1). Из этой схемы видно, что на основе исходной доминирующей мотивации, возникающей в результате той или иной внутренней потребности организма и памяти, организм, стимулируемый пусковыми раздражителями – сигналами, активно оценивает состояние раздражителей внешней среды, вырабатывает цель и принимает решение о начале действий. Согласно цели все компоненты системы взаимодействуют так, чтобы выполнялась афферентная программа действия.

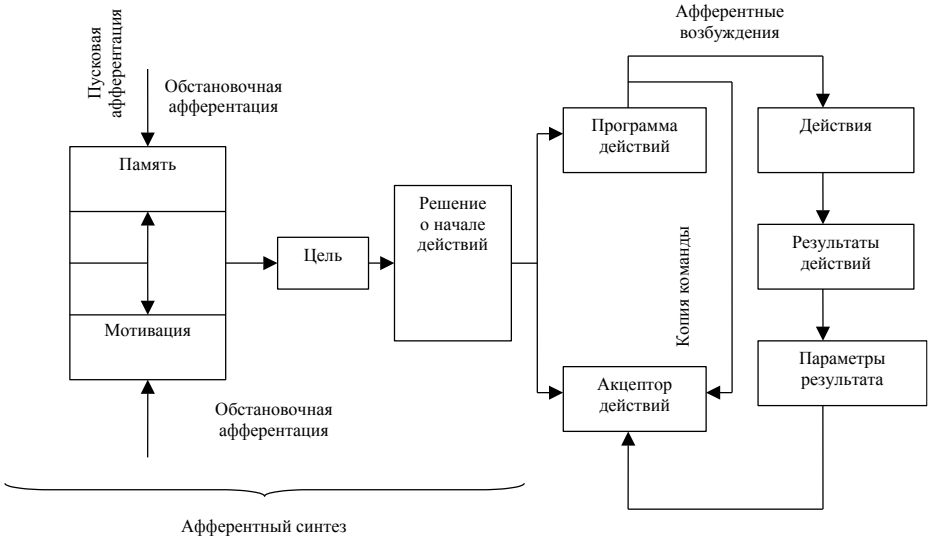


Рис. 12.1. Структурная схема функциональной системы живого организма

Аппарат акцепторов результатов действия, формирующийся на основе определенной потребности, памяти, обстановки и специальных сигналов, заключает в себе все свойства будущего результата и поэтому служит для сопоставления предсказанного и достигнутого результатов действий. Изложенное описание работы любой функциональной системы (см. рис. 12.1) позволяет составить структурную схему реальной ИС, представленной на рис. 12.2.

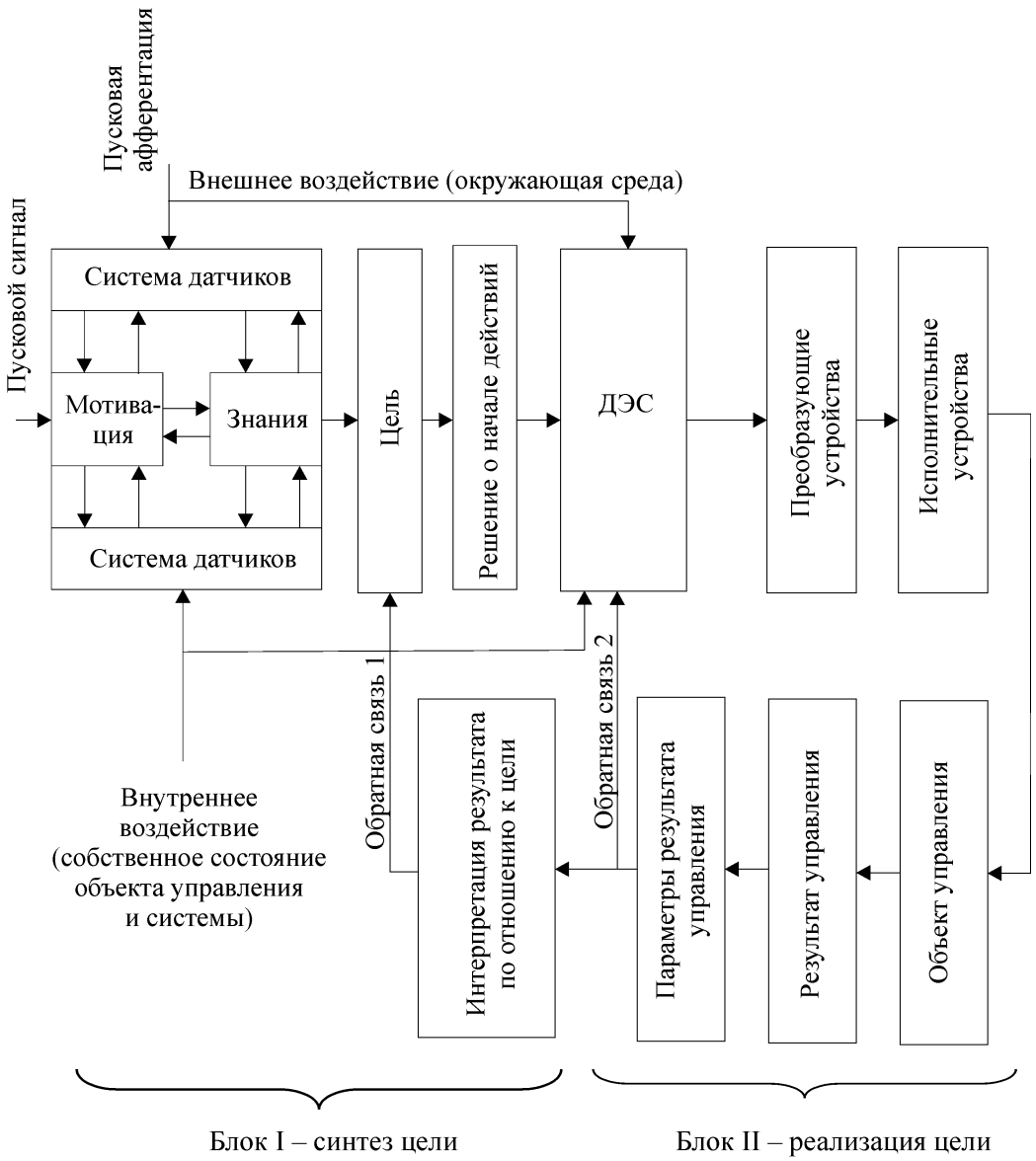


Рис. 12.2. Структурная схема реальной ИС

Приведем краткое описание работы ИС на основании сведений об окружающей среде и собственном состоянии системы: при наличии памяти и мотивации синтезируется цель, которая наряду с другими данными воспринимается динамической экспертной системой (ДЭС). Последняя с использованием базы знаний производит экспертную оценку, на основании которой принимается решение о действии и прогнозируются его результаты (акцептор действия). В соответствии с принятым решением вырабатывается управление, т.е. синтезируется тот или иной алгоритм или закон управления, который реализуется с помощью различных исполнительных органов и воздействует непосредственно на объект управления. Результаты этого воздействия сравниваются с прогнозируемыми (механизм обратной связи, акцептор действия). При несоответствии результатов на базе новой экспертной оценки принимается решение и вырабатывается и реализуется управление, устраняющее это несоответствие. При соответствии результатов предшествующее управление подтверждается. Если соответствие недостижимо, то уточняется цель. Данная структура инвариантна к объекту управления и имеет универсальный характер.

При формировании цели возникает проблема достаточности базы знаний, а следовательно, и памяти, возможности их реализации как на содержательном, так и на конструктивном уровне. Динамическая экспертная система выполняет расчет, оптимизацию, прогнозирует и моделирует результаты, поэтому должна обладать высоким быстродействием. Потенциал, накопленный при разработке алгоритмов принятия решений и выработки управления, может успешно использоваться в интеллектуальных системах, однако потребуется распараллеливание алгоритмов и их мультитранспьютерная реализация и, конечно же, не исключается синтез новых, эффективных параллельных алгоритмов. Поэтому одной из фундаментальных проблем теории интеллектуальных систем является разработка адекватных живой природе математических моделей.

Формально ИС описываются следующими шестью выражениями:

$$T \cdot X \cdot S \xrightarrow{\alpha_1} M \cdot T ;$$

$$T \cdot M \cdot S \xrightarrow{\alpha_2} C \cdot T ;$$

$$C \cdot T \cdot X \cdot S \xrightarrow{\alpha_3} R \cdot T ;$$

$$T \cdot X = \{\boldsymbol{A} \cdot T\} X \cdot T + \{\boldsymbol{B} \cdot T\} U \cdot T ;$$

$$T \cdot Y = \{\boldsymbol{D} \cdot T\} X \cdot T ;$$

$$T \cdot R \cdot Y \xrightarrow{\alpha_4} C \cdot T ,$$

где T – множество моментов времени; U – управляющее воздействие; X, S, M, C, R и Y – множества состояний системы, окружающей среды, мотивации, цели, прогнозируемого и реального результатов; $\boldsymbol{A}$, $\boldsymbol{B}$ и $\boldsymbol{D}$ – матрицы параметров; $\alpha_1 \ldots \alpha_4$ – интеллектуальные операторы.

В этом описании сочетаются представления объектов системы в виде множества значений, множества высказываний либо множества каких-то других форм. Динамические свойства ИС могут быть описаны в пространстве состояний. Интеллектуальные операторы, реализующие восприятие, представление, формирование понятия, суждения и умозаключения в процессе познания, являются формальным средством обработки сведений и знаний, а также принятия решения. Все эти аспекты положены в основу построения динамических экспертных систем (ДЭС), функционирующих в реальном времени и реальном мире.

3. Структура процесса получения решения

1. Структура процесса получения решения:

• компиляция в режиме приобретения знаний дерева вывода из обучающей выборки (индуктивные методы приобретения знаний); выбор решения из дерева вывода в режиме решения задачи;

• компиляция в режиме приобретения знаний сети вывода из специфических правил; поиск решения в сети вывода в режиме решения задачи;

• генерация сети вывода и поиск решения в режиме решения задачи; генерация сети вывода осуществляется в ходе выполнения операции сопоставления,

определяющей пары: правило n_i – совокупность данных, на которых условия этого правила удовлетворяются (см. гл. 6);

• в режиме решения задач ЭС осуществляет выработку правдоподобных предположений (при отсутствии достаточной информации для решения); выполнение рассуждений по обоснованию (опровержению) предположений; генерацию альтернативных сетей вывода; поиск решения в сетях вывода [1].

2. Поиск (выбор) решения (см. гл. 6):

• направленность поиска: от данных к цели, от целей к данным, двунаправленный поиск;

• порядок перебора вершин в сети Вывода: "поиск в ширину" -сначала раскрываются (обрабатываются) все вершины, непосредственно связанные с текущей обрабатываемой вершиной Т, "поиск в глубину" - сначала раскрывается одна наиболее значимая вершина -$Г_1$, связанная с текущей Т, затем вершина $Г_1$ делается текущей, и для нее раскрывается одна наиболее значимая вершина $Г_2$ и т. д.

3. Процесс генерации предположений и сети вывода (см. гл. 6):

• режим генерации сети вывода: генерация в режиме приобретения знаний, генерация в режиме решения задачи;

• полнота генерируемой сети вывода: операция сопоставления применяется ко всем правилам и всем типам указанных в правилах сущностей на каждом цикле работы механизма вывода (обеспечивается полнота генерируемой сети); используются различные средства для сокращения количества правил и (или) сущностей, участвующих в операции сопоставления; например, применяется алгоритм сопоставления Rete (см. гл. 6) или используются метазнания типа действий focus и invoke (см. гл. 9).

Механизм вывода для динамических проблемных сред дополнительно содержит:

• планировщик, обеспечивающий в соответствии с приоритетами всю деятельность ЭС;

• средства, гарантирующие получение лучшего решения в условиях ограниченности ресурсов;

• систему поддержания истинности значений переменных, изменяющихся во времени.

Динамические ИС по отношению к системе моделирования могут быть охарактеризованы следующим образом:

1) система моделирования отсутствует;

2) существует система моделирования общего назначения, являющаяся частью ИС;

3) существует специализированная система моделирования, являющаяся внешней по отношению к ИС, на котором реализуется ЭС.

Системы моделирования общего назначения используют для описания модели алгебраические, разностные и дифференциальные уравнения (обычно первого порядка). Как правило, в систему моделирования включается несколько альтернативных методов. Например, по выбору пользователя уравнение решается либо методом Рунге-Кутта, обеспечивающим лучшую точность, либо методом Эйлера, обеспечивающим меньшую точность, но более быстрое получение решения. Специализированные системы моделирования учитывают специфику приложения (например, модель химического предприятия).

Контрольные вопросы:

1. Неформализованные задачи?
2. Сущность механизма афферентного синтеза?
3. Структура процесса получения решения?

Литература

4. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
5. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
6. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 10: Динамические экспертные системы

План:

1. Назначение экспертных систем
2. Формальные основы экспертных систем
3. Архитектура статических и динамических экспертных систем
4. Этапы разработки экспертных систем

1. Назначение экспертных систем

В начале восьмидесятых годов в исследованиях по искусственному интеллекту сформировалось самостоятельное направление, получившее название "экспертные системы" ОС). Цель исследований по ЭС состоит в разработке программ, которые при решении задач, трудных для эксперта-человека, получают результаты, не уступающие по качеству и эффективности решениям, получаемым экспертом. Исследователи в области ЭС для названия своей дисциплины часто используют также термин "инженерия знаний", введенный Е. Фейгенбаумом [7] как "привнесение принципов и инструментария исследований из области искусственного интеллекта в решение трудных прикладных проблем, требующих знаний экспертов".

Программные средства (ПС), базирующиеся на технологии экспертных систем, или инженерии знаний (в дальнейшем будем использовать их как синонимы), получили значительное распространение в мире. Важность экспертных систем состоит в следующем:

• технология экспертных систем существенно расширяет круг практически значимых задач, решаемых на компьютерах, решение которых приносит значительный экономический эффект;

• технология ЭС является важнейшим средством в решении глобальных проблем традиционного программирования: длительность и, следовательно, высокая стоимость разработки сложных приложений; высокая стоимость сопровождения сложных систем, которая часто в несколько раз превосходит стоимость их разработки; низкий уровень повторной используемости программ и т.п.;

• объединение технологии ЭС с технологией традиционного программирования добавляет новые качества к программным продуктам за счет: обеспечения динамичной модификации приложений пользователем, а не программистом; большей "прозрачности" приложения (например, знания хранятся на ограниченном ЕЯ, что не требует комментариев к знаниям, упрощает обучение и сопровождение); лучшей графики; интерфейса и взаимодействия.

По мнению ведущих специалистов [10], в недалекой перспективе ЭС найдут следующее применение:

• ЭС будут играть ведущую роль во всех фазах проектирования, разработки, производства, распределения, продажи, поддержки и оказания услуг;

• технология ЭС, получившая коммерческое распространение, обеспечит революционный прорыв в интеграции приложений из готовых интеллектуально-взаимодействующих модулей.

ЭС предназначены для так называемых неформализованных задач, т.е. ЭС не отвергают и не заменяют традиционного подхода к разработке программ, ориентированного на решение формализованных задач. Следуя А.Ньюэллу и М.Саймону [13], к неформализованным (ill-structured) будем относить такие задачи, которые обладают одной или несколькими из следующих характеристик:

• задачи не могут быть заданы в числовой форме;

• цели не могут быть выражены в терминах точно определенной целевой функции;

• не существует алгоритмического решения задач;

• алгоритмическое решение существует, но его нельзя использовать из-за ограниченности ресурсов (время, память).

Неформализованные задачи обычно обладают следующими особенностями:

• ошибочностью, неоднозначностью, неполнотой и противоречивостью исходных данных;

• ошибочностью, неоднозначностью, неполнотой и противоречивостью знаний о проблемной области и решаемой задаче;

• большой размерностью пространства решения, т.е. перебор при поиске решения весьма велик;

• динамически изменяющимися данными и знаниями.

Следует подчеркнуть, что неформализованные задачи представляют большой и очень важный класс задач. Многие специалисты считают, что эти задачи являются наиболее массовым классом задач, решаемых ЭВМ.

Экспертные системы и системы искусственного интеллекта отличаются от систем обработки данных тем, что в них в основном используются символьный (а не числовой) способ представления, символьный вывод и эвристический поиск решения (а не исполнение известного алгоритма).

Специфика приложений экспертных систем по сравнению с другими системами искусственного интеллекта состоит в следующем. Экспертные системы применяются для решения только трудных практических (не игрушечных) задач. По качеству и эффективности решения экспертные системы не уступают решениям эксперта-человека. Решения экспертных систем обладают *"прозрачностью"*, т.е. могут, быть объяснены пользователю на качественном уровне (в отличие от решений, полученных с помощью числовых алгоритмов, и в особенности от решений полученных статистическими методами). Это качество экспертных систем обеспечивается их способностью рассуждать о своих знаниях и умозаключениях. Экспертные системы способны пополнять свои знания в ходе взаимодействия с экспертом. Необходимо отметить, что в настоящее время технология экспертных систем используется для решения различных типов задач (интерпретация, предсказание, диагностика, планирование, конструирование, контроль, отладка, инструктаж, управление) в самых разнообразных проблемных областях, таких, как финансы, нефтяная и газовая промышленность, энергетика, транспорт, фармацевтическое производство, космос, металлургия, горное дело, химия, образование, целлюлозно-бумажная промышленность, телекоммуникации и связь и др.

Приведем некоторые примеры успешного применения технологии ЭС:

• фирма DEC (США) ежегодно экономит [10] 70 млн. дол. в год благодаря ЭС XCON/XSEL, которая по заказу покупателя составляет конфигурацию вычислительной системы VAX. Использование ЭС сократило количество ошибок от 30% (допускал человек) до 1% (допускает ЭС);

• фирма Sira (США) сократила затраты на строительство трубопровода в Австралии на 40 млн. дол. [6] за счет ЭС, управляющей трубопроводом. ЭС реализована на базе описываемого ниже ИС G2 (фирма Gensym);

• фирма Monsanto (США) ежегодно экономит от 250 до 500 тыс.дол. благодаря ЭС выявления и блокирования неисправностей в нефтехимической промышленности. ЭС реализована на базе ИС G2 (фирма Gensym);

• фирма Aetna Insurance (США) уже сэкономила более 5 млн. дол., а общий планируемый эффект составит около 15-20 млн. дол. благодаря ЭС, используемой для моделирования страховых исков, обрабатываемых компанией. ЭС, реализованная на базе ИС G2, позволяет находить в деятельности компании неэффективные процессы и рабочие потоки и производить оперативные изменения для увеличения продуктивности работы.

Коммерческие успехи к фирмам-разработчикам систем искусственного интеллекта (СИИ) пришли не сразу. На протяжении 1960 - 1985 гг. успехи ИИ касались в основном исследовательских разработок, которые демонстрировали пригодность СИИ для практического использования. Начиная примерно с 1985 г. (в массовом масштабе с 1988 - 1990 гг.), в первую очередь ЭС, а в последние годы системы, воспринимающие естественный язык (ЕЯ-системы), и нейронные сети (НС) стали активно использоваться в коммерческих приложениях.

Следует обратить внимание на то, что некоторые специалисты (как правило, специалисты в программировании, а не в ИИ) продолжают утверждать, что ЭС и СИИ не оправдали возлагавшихся на них ожиданий и умерли [5]. Причины таких заблуждений состоят в том, что эти авторы рассматривали ЭС как альтернативу традиционному программированию, т.е. они исхо-

дили из того, что ЭС в одиночестве (в изоляции от других программных средств) полностью решают задачи, стоящие перед заказчиком. Надо отметить, что на заре появления ЭС специфика используемых в них языков, технологии разработки приложений и используемого оборудования (например, Lisp-машины) давала основания предполагать, что интеграция ЭС с традиционными, программными системами является сложной и, возможно, невыполнимой задачей при ограничениях, накладываемых реальными приложениями. Однако в настоящее время коммерческие инструментальные средства (ИС) для создания ЭС разрабатываются я полном соответствии с современными технологическими тенденциями традиционного программирования, что снимает проблемы, возникающие при создании интегрированных приложений. Говоря другими словами, технология ЭС нашла свое применение при создании интегрированных, а не изолированных приложений во многих областях. В настоящее время высказанные соображения становятся понятными и тем специалистам, которые считали, что ЭС умерли [11].

Причины, приведшие СИИ к коммерческому успеху, следующие.

Интегрированность. Разработаны инструментальные средства искусственного интеллекта (ИС ИИ), легко интегрирующиеся с другими информационными технологиями и средствами (с CASE, СУБД, контроллерами, концентраторами данных и т.п.).

Открытость и переносимость. ИС ИИ разрабатываются с соблюдением стандартов, обеспечивающих открытость и переносимость [14].

Использование языков традиционного программирования и рабочих станций. Переход от ИС ИИ, реализованных на языках ИИ (Lisp, Prolog и т.п.), к ИС ИИ, реализованным на языках традиционного программирования (С, С++ и т.п.), упростил обеспечение интегрированности, снизил требования приложений ИИ к быстродействию ЭВМ и объемам оперативной памяти. Использование рабочих станций (вместо ПК) резко увеличило круг приложений, которые могут быть выполнены на ЭВМ с использованием ИС ИИ.

Архитектура клиент-сервер. Разработаны ИС ИИ, поддерживающие распределенные вычисления по архитектуре клиент-сервер, что позволило: снизить стоимость оборудования, используемого в приложениях, децентрализовать приложения, повысить надежность и общую производительность (так как сокращается количество информации, пересылаемой между ЭВМ, и каждый модуль приложения выполняется на адекватном ему оборудовании).

Проблемно/предметно-ориентированные ИС ИИ. Переход от разработок ИС ИИ общего назначения (хотя они не утратили свое значение как средство для создания ориентированных ИС) к проблемно/предметно-ориентированным ИС ИИ [9] обеспечивает: сокращение сроков разработки приложений; увеличение эффективности использования ИС; упрощение и ускорение работы эксперта; повторную используемость информационного и программного обеспечения (объекты, классы, правила, процедуры).

Отметим, что перечисленные выше причины успеха могут рассматриваться как общие требования к коммерческим ИС для создания СИИ. При этом первые четыре требования вытекают из необходимости создания интегрированных приложений, т.е. приложений, объединяющих в рамках единого комплекса традиционные программные системы с системами ИИ. Для того чтобы эта интеграция была эффективной, инструментальные средства ИИ должны разрабатываться в полном соответствии с основными тенденциями традиционного программирования. Пятое и третье требования являются следствием стремления обеспечить эффективное выполнение задач ИИ на ЭВМ с традиционной архитектурой.

2. Формальные основы экспертных систем

Большинство экспертных систем базируется на понятии "формальная продукционная система". Продукционные системы берут свое начало с работ Е.Поста, который в 1943 г. ввел термины *продукция* и *каноническая (продукционная) система* [2], [3]. Е.Пост показал, что продукционная система является логической системой, эквивалентной машине Тьюринга [2]. Другими словами, продукционные системы универсальны, т.е. любая формальная система, оперирующая символами, может быть реализована в виде одной из продукционных систем Е.Поста.

Система продукций Поста задается своим алфавитом C= {c ,...,c}

и системой базисных продукций

$x_i W \rightarrow W y_i$ (i =1,...,1),

где x_i, y_i - слова в алфавите С.

Пусть некоторое слово Y начинается словом x_i . Применить к Y продукцию $x_iW \rightarrow Wy_i$ - это значит вычеркнуть из Y начальный отрезок x_i и затем к оставшемуся слову приписать слово y_i. Например, применив к слову *aba* продукцию *ab*W →W*c*, получим слово *ac*.

Каждая система продукций понимается как формальная система с правилами вывода p_i (i = 1, ... , 1), где p_i (F,Y) считается истинным (применимым), если слово Y получается из F при помощи продукции $x_iW \rightarrow Wy_i$.

Наложив на набор упорядоченных продукций неявную управляющую структуру, перейдем к понятию нормального алгоритма Маркова [2]. В алгоритме Маркова упорядоченные продукции (формулы подстановок) применяются к некоторому заданному слову.

Первая же из упорядоченных продукций, которая может быть применена к слову, применяется, изменяя слово. Затем процесс проверки применимости продукций продолжается, начиная с продукции, имеющей наивысший приоритет. Этот цикл "проверка (выполнение)" продолжается до тех пор, пока не найдется ни одной применимой продукции либо не будет применена некая продукция, помеченная как заключительная.

Психологические исследования процессов принятия решений человеком [13] показали, что, рассуждая, человек использует правила, аналогичные продукциям, т.е. правила вида "условие → действие". А.Ньюэлл [12] предложил использовать продукционные системы для моделирования на ЭВМ процесса принятия решений. Формализуя предложения Ньюэлла, определим продукционную систему (PS) следующим образом:

PS = <R, B, I>,

где R - рабочая память системы (называемая также базой данных), содержащая текущие данные (элементы рабочей памяти);

B - база знаний, содержащая множество продукций (правил вида: "условие → действие");

I - интерпретатор (решатель), реализующий процесс вывода, который в цикле выполняет следующие действия: определяет множество означиваний, т.е. множество пар: {правило (p_i), набор текущих данных (a_j), на котором это правило удовлетворяется}; выполняет определенные означивания, производя изменения в рабочей памяти.

Интерпретатор формально может быть представлен четверкой:

I = (V, S, K, W),

где V - процесс выбора из B и из R подмножества активных продукций B_v и подмножества активных данных R_v соответственно, которые будут использованы в очередном цикле работы интерпретатора. Механизм выбора может быть тривиальным (на каждом цикле выбираются все правила и все данные) или более сложным [4] для того, чтобы устранить из рассмотрения те правила, условия которых заведомо не удовлетворяются данными рабочей памяти или малополезны. В усложненных системах механизм выбора может использовать иерархию правил, метаправила или сложные схемы управления, подобные сетям Петри [1];

S - процесс сопоставления, определяющий множество означиваний, т.е. множество пар: правило (p_i) - данные (d_i), где $p_i \in P_v$, $\{d_i\} \subset R_v$, причем каждое p_i применимо к элементам множества $\{d_i\}$ (будем также говорить, что "p_i удовлетворяется на элементах множества $\{d_i\}$"). Операция сопоставления может требовать много времени, так как в общем случае влечет за собой означивание многих переменных;

K - процесс разрешения конфликтов (или процесс планирования), определяющий, какое из означиваний будет выполняться. Механизм разрешения конфликтов [4] может быть неявным или явным (например, в виде некоторого множества метаправил или процедур, описывающих выбор выполняемого правила). Метаправила позволяют обеспечить прямым и понятным способом применение динамических эвристик для разрешения конфликтов;

W - процесс, осуществляющий выполнение выбранного означенного правила (т. е. выполнение действий, указанных в правой части правила). Результатом выполнения является модификация данных в R или операция ввода-вывода.

Можно показать, что продукционные системы по Ньюэллу являются некоторым неформальным обобщением алгоритмов Маркова. Причины успешного практического использования экспертных систем состоят в том, что при их построении были учтены уроки предшествующих исследований в области искусственного интеллекта. Сформулируем эти уроки в виде трех принципов (два из них впервые высказаны Е. Фейгенбаумом [8]).

1. Мощность экспертной системы обусловлена в первую очередь мощностью базы знаний и возможностью ее пополнения и только во вторую очередь - используемыми ею методами (процедурами). В исследованиях по искусственному интеллекту господствовала обратная точка зрения. Источником интеллектуальности считали небольшое количество общих мощных процедур вывода. Однако опыт показал, что важнее иметь разнообразные специальные знания, а не общие процедуры вывода.

2. Знания, позволяющие эксперту (или экспертной системе) получить качественные и эффективные решения задач, являются в основном эвристическими, экспериментальными, неопределенными, правдоподобными. Причина этого заключается в том, что решаемые задачи являются неформализованными или слабоформализованными. Необходимо также подчеркнуть, что знания экспертов имеют индивидуальный характер, т.е. свойственны конкретному человеку.

3. Учитывая неформализованность решаемых задач и эвристический, личностный характер используемых знаний, пользователь (эксперт) должен иметь возможность непосредственного взаимодействия с экспертной системой в виде диалога.

Архитектура экспертной системы вытекает из принципов, сформулированных выше. В соответствии с первыми двумя принципами ЭС включает два компонента: решатель (процедуры вывода) и динамически изменяемую базу знаний. Выбор в качестве основы для реализации решателя систем продукций предопределяет наличие в ЭС также и рабочей памяти.

Третий принцип предъявляет к системе следующие требования:

• способность вести диалог о решаемой задаче на языке, удобном пользователю (эксперту), и, в частности, приобретать в ходе диалога новые знания;

• способность при решении задачи следовать линии рассуждения, понятной пользователю (эксперту);

• способность объяснять ход своего рассуждения на языке, удобном для пользователя (эксперта), что необходимо как при использовании, так и при совершенствовании системы (т. е. при отладке и модификации базы знаний).

Первое требование реализуется диалоговым компонентом ЭС и компонентом приобретения знаний, а для выполнения второго и третьего требований в ЭС вводится объяснительный компонент. Кроме того, второе требование накладывает ограничения на способ решения задачи: ход рассуждения в процессе решения должен быть понятен пользователю (эксперту).

3. Архитектура статических и динамических экспертных систем

Типичная статическая ЭС состоит из следующих основных компонентов (рис. 1.1):

• решателя (интерпретатора);

• рабочей памяти (РП), называемой также базой данных (БД);

• базы знаний (БЗ);

• компонентов приобретения знаний;

• объяснительного компонента;

• диалогового компонента.

База данных (рабочая память) предназначена для хранения исходных и промежуточных данных решаемой в текущий момент задачи. Этот термин совпадает по названию, но не по смыслу с термином, используемым в информационно-поисковых системах (ИПС) и системах управления базами данных (СУБД) для обозначения всех данных (в первую очередь долгосрочных), хранимых в системе.

База знаний (БЗ) в ЭС предназначена для хранения долгосрочных данных, описывающих рассматриваемую область (а не текущих данных), и правил, описывающих целесообразные преобразования данных этой области.

Решатель, используя исходные данные из рабочей, памяти и знания из БЗ, формирует такую последовательность правил, которые, будучи примененными к исходным данным, приводят к решению задачи.

Компонент приобретения знаний автоматизирует процесс наполнения ЭС знаниями, осуществляемый пользователем-экспертом.

Объяснительный компонент объясняет, как система получила решение задачи (или почему она не получила решение) и какие знания она при этом использовала, что облегчает эксперту тестирование системы и повышает доверие пользователя к полученному результату.

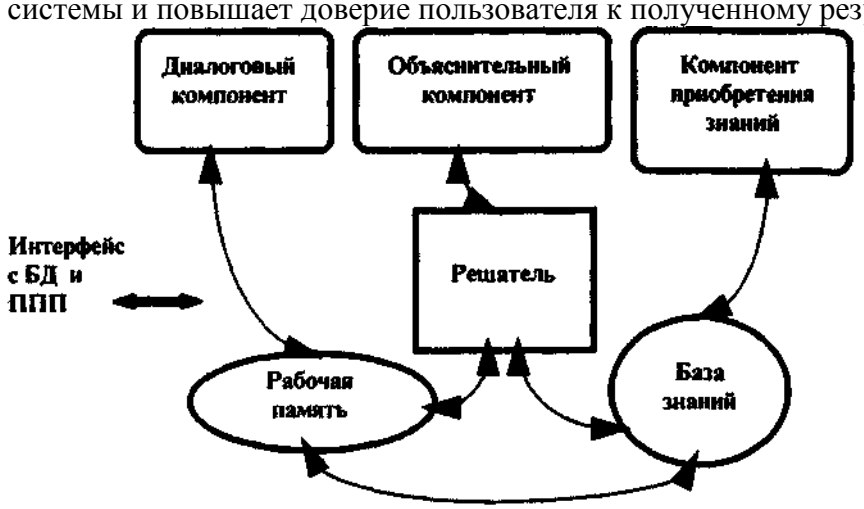


Рис. 1.1. Структура статической ЭС

Диалоговый компонент ориентирован на организацию дружественного общения с пользователем как в ходе решения задач, так и в процессе приобретения знаний и объяснения результатов работы.

В разработке ЭС участвуют представители следующих специальностей:

- эксперт в проблемной области, задачи которой будет решать ЭС;
- инженер по знаниям - специалист по разработке ЭС (используемые им технологию, методы называют технологией (методами) инженерии знаний);
- программист по разработке инструментальных средств (ИС), предназначенных для ускорения разработки ЭС.

Необходимо отметить, что отсутствие среди участников разработки инженеров по знаниям (т. е. их замена программистами) либо приводит к неудаче процесс создания ЭС, либо значительно удлиняет его.

Эксперт определяет знания (данные и правила), характеризующие проблемную область, обеспечивает полноту и правильность введенных в ЭС знаний.

Инженер по знаниям помогает эксперту выявить и структурировать знания, необходимые для работы ЭС; осуществляет выбор того ИС, которое наиболее подходит для данной проблемной области, и определяет способ представления знаний в этом ИС; выделяет и программирует (традиционными средствами) стандартные функции (типичные для данной проблемной области), которые будут использоваться в правилах, вводимых экспертом.

Программист разрабатывает ИС (если ИС разрабатывается заново), содержащее в пределе все основные компоненты ЭС, и осуществляет его сопряжение с той средой, в которой оно будет использовано.

Экспертная система работает в двух режимах: режиме приобретения знаний и в режиме решения задачи (называемом также режимом консультации или режимом использования ЭС).

В режиме приобретения знаний общение с ЭС осуществляет (через посредничество инженера по знаниям) эксперт. В этом режиме эксперт, используя компонент приобретения знаний, наполняет систему знаниями, которые позволяют ЭС в режиме решения самостоятельно (без эксперта) решать задачи из проблемной области. Эксперт описывает проблемную область в виде совокупности данных и правил. Данные определяют объекты, их характеристики и значения, существующие в области экспертизы. Правила определяют способы манипулирования с данными, характерные для рассматриваемой области.

Отметим, что режиму приобретения знаний в традиционном подходе к разработке программ соответствуют этапы алгоритмизации, программирования и отладки, выполняемые программистом. Таким образом, в отличие от традиционного подхода в случае ЭС разработку программ осуществляет не программист, а эксперт (с помощью ЭС), не владеющий программированием.

В режиме консультации общение с ЭС осуществляет конечный пользователь, которого интересует результат и (или) способ его получения. Необходимо отметить, что в зависимости

от назначения ЭС пользователь может не быть специалистом в данной проблемной области (в этом случае он обращается к ЭС за результатом, не умея получить его сам), или быть специалистом (в этом случае пользователь может сам получить результат, но он обращается к ЭС с целью либо ускорить процесс получения результата, либо возложить на ЭС рутинную работу). Следует подчеркнуть, что термин "пользователь" является многозначным, так как использовать ЭС кроме конечного пользователя может и эксперт, и инженер по знаниям, и программист. Поэтому когда хотят подчеркнуть, что речь идет о том, для кого делалась ЭС, используют термин "конечный пользователь".

В режиме консультации данные о задаче пользователя после обработки их диалоговым компонентом поступают в рабочую память. Решатель на основе входных данных из рабочей памяти, общих данных о проблемной области и правил из БЗ формирует решение задачи. Подчеркнем, что в отличие от традиционных программ ЭС при решении задачи не только исполняет предписанную последовательность операции, но и предварительно формирует ее. Если реакция системы не понятна пользователю, то он может потребовать объяснения: "Почему система задает тот или иной вопрос?", "как ответ, собираемый системой, получен?".

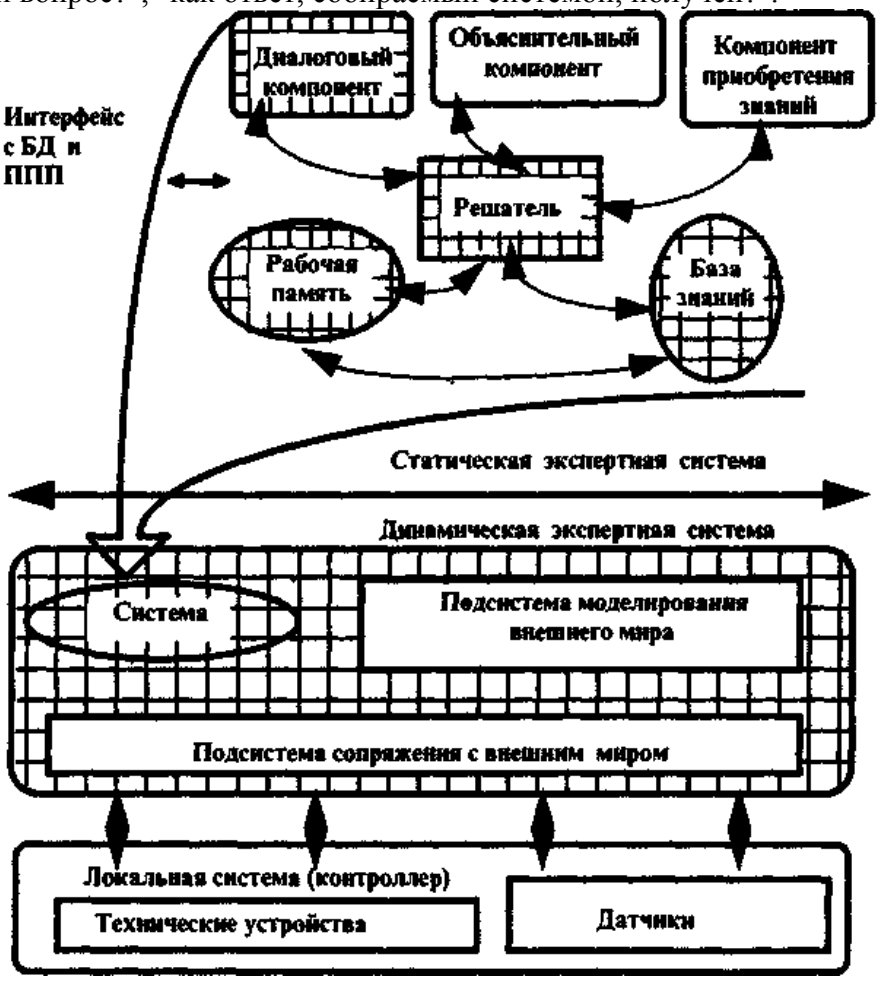


Рис. 1.2. Архитектура статических и динамических ЭС (компоненты, подвергающиеся изменениям, заштрихованы)

Структуру, приведенную на рис. 1.1, *называют структурой статической ЭС.* ЭС данного типа используются в тех приложениях, где можно не учитывать изменения окружающего мира, происходящие за время решения задачи. Первые ЭС, получившие практическое использование, были статическими. Они нашли применение в широком классе приложений (см. п.4.1).

Из общих соображений понятно, что существует огромный класс приложений, в которых требуется учитывать динамику, т. е. изменения, происходящие в окружающем мире за время исполнения приложения. На рис. 1.2 показано, что в архитектуру динамической ЭС по сравнению со статической ЭС вводятся два компонента: подсистема моделирования внешнего мира и подсистема связи с внешним окружением. Последняя осуществляет связи с внешним миром через систему датчиков и контроллеров. Кроме того, традиционные компоненты статической ЭС

(база знаний и машина вывода) претерпевают существенные изменения, чтобы отразить временную логику происходящих в реальном мире событий (подробнее см. гл.9).

Подчеркнем, что структура ЭС, представленная на рис. 1.1 и 1.2, отражает только компоненты (функции), и многое остается "за кадром". На рис. 1.3 приведена обобщенная структура современного ИС для создания динамических ЭС, содержащая кроме основных компонентов те возможности, которые позволяют создавать интегрированные приложения в соответствии с современной технологией программирования (более подробно см. гл.9).

Рис. 1.3. Структура современных ИС для ЭС

В основе ИС лежат объектно-ориентированная база знаний (ОО-технология БЗ) и механизм вывода, способный оперировать с правилами, в которых явным образом отражено время (РВ - механизм вывода). Во внутреннем кольце расположены компоненты, обеспечивающие моделирование, анимацию, активную графику, механизм общих правил и т.д. Во внешнем кольце отражены технологии и требования, обязательные в современных ИС для создания ЭС (см. п. 1.1).

4. Этапы разработки экспертных систем

Разработка ЭС имеет существенные отличия от разработки обычного программного продукта. Опыт создания ЭС показал, что использование при их разработке методологии, принятой в традиционном программировании, либо чрезмерно затягивает процесс создания ЭС, либо вообще приводит к отрицательному результату. Дело в том, что неформализованность задач, решаемых ЭС, отсутствие завершенной теории ЭС и методологии их разработки приводят к необходимости модифицировать принципы и способы построения ЭС в ходе процесса разработки по мере того, как увеличивается знание разработчиков о проблемной области.

Перед тем как приступить к разработке ЭС, инженер по знаниям должен рассмотреть вопрос, следует ли разрабатывать ЭС для данного приложения. В обобщенном виде ответ может быть таким; использовать ЭС следует только тогда, когда разработка ЭС *возможна, оправдана* и методы инженерии знаний *соответствуют* решаемой задаче. Ниже будут уточнены использованные понятия "возможно", "оправдано", "соответствует". Чтобы разработка ЭС была *возможной* для данного приложения, необходимо одновременное выполнение по крайней мере следующих требований:

1) существуют эксперты в данной области, которые решают задачу значительно лучше, чем начинающие специалисты;

2) эксперты сходятся в оценке предлагаемого решения, иначе нельзя будет оценить качество разработанной ЭС;

3) эксперты способны вербализовать (выразить на естественном языке) и объяснить используемые ими методы, в противном случае трудно рассчитывать на то, что знания экспертов будут "извлечены" и вложены в ЭС;

4) решение задачи требует только рассуждений, а не действий;

5) задача не должна быть слишком трудной (т.е. ее решение должно занимать у эксперта несколько часов или дней, а не недель);

6) задача хотя и не должна быть выражена в формальном виде, но все же должна относиться к достаточно "понятной" и структурированной области, т.е. должны быть выделены основные понятия, отношения и известные (хотя бы эксперту) способы получения решения задачи;

7) решение задачи не должно в значительной степени использовать "здравый смысл" (т.е. широкий спектр общих сведений о мире и о способе его функционирования, которые знает и умеет использовать любой нормальный человек), так как подобные знания пока не удается (в достаточном количестве) вложить в системы искусственного интеллекта.

Использование ЭС в данном приложении может быть возможно, но не оправдано. Применение ЭС может быть *оправдано* одним из следующих факторов:

• решение задачи принесет значительный эффект, например экономический;

• использование человека-эксперта невозможно либо из-за недостаточного количества экспертов, либо из-за необходимости выполнять экспертизу одновременно в различных местах;

• использование ЭС целесообразно в тех случаях, когда при передаче информации эксперту происходит недопустимая потеря времени или информации;

• использование ЭС целесообразно при необходимости решать задачу в окружении, враждебном для человека.

Приложение *соответствует* методам ЭС, если решаемая задача обладает совокупностью следующих характеристик:

1) задача может быть естественным образом решена посредством манипуляции с символами (т.е. с помощью символических рассуждений), а не манипуляций с числами, как принято в математических методах и в традиционном программировании;

2) задача должна иметь эвристическую, а не алгоритмическую природу, т.е. ее решение должно требовать применения эвристических правил. Задачи, которые могут быть гарантированно решены (с соблюдением заданных ограничений) с помощью некоторых формальных процедур, не подходят для применения ЭС;

3) задача должна быть достаточно сложна, чтобы оправдать затраты на разработку ЭС. Однако она не должна быть чрезмерно сложной (решение занимает у эксперта часы, а не недели), чтобы ЭС могла ее решать;

4) задача должна быть достаточно узкой, чтобы решаться методами ЭС, и практически значимой.

При разработке ЭС, как правило, используется концепция "быстрого прототипа". Суть этой концепции состоит в том, что разработчики не пытаются сразу построить конечный продукт. На начальном этапе они создают прототип (прототипы) ЭС. Прототипы должны удовлетворять двум противоречивым требованиям: с одной

стороны, они должны решать типичные задачи конкретного приложения, а с другой - время и трудоемкость их разработки должны быть весьма незначительны, чтобы можно было максимально запараллелить процесс накопления и отладки знаний (осуществляемый экспертом) с процессом выбора (разработки) программных средств (осуществляемым инженером по знаниям и программистом). Для удовлетворения указанным требованиям, как правило, при создании прототипа используются разнообразные средства, ускоряющие процесс проектирования.

Прототип должен продемонстрировать пригодность методов инженерии знаний для данного приложения. В случае успеха эксперт с помощью инженера по знаниям расширяет знания прототипа о проблемной области. При неудаче может потребоваться разработка нового прототипа или разработчики могут прийти к выводу о непригодности методов ЭС для данного приложения. По мере увеличения знаний прототип может достигнуть такого состояния, когда он успешно решает все задачи данного приложения. Преобразование прототипа ЭС в конечный

продукт обычно приводит к перепрограммированию ЭС на языках низкого уровня, обеспечивающих как увеличение быстродействия ЭС, так и уменьшение требуемой памяти. Трудоемкость и время создания ЭС в значительной степени зависят от типа используемого инструментария.

В ходе работ по созданию ЭС сложилась определенная технология их разработки [4], включающая шесть следующих этапов (рис. 1.4): идентификацию, концептуализацию, формализацию, выполнение, тестирование, опытную эксплуатацию (подробнее см. раздел 7.1). На этапе *идентификации* определяются задачи, которые подлежат решению, выявляются цели разработки, определяются эксперты и типы пользователей.

На этапе *концептуализации* проводится содержательный анализ проблемной области, выявляются используемые понятия и их взаимосвязи, определяются методы решения задач.

На этапе *формализации* выбираются ИС и определяются способы представления всех видов знаний, формализуются основные понятия, определяются способы интерпретации знаний, моделируется работа системы, оценивается адекватность целям системы зафиксированных понятий, методов решений, средств представления и манипулирования знаниями.

На этапе *выполнения* осуществляется наполнение экспертом базы знаний. В связи с тем, что основой ЭС являются знания, данный этап является наиболее важным и наиболее трудоемким этапом разработки ЭС. Процесс приобретения знаний разделяют на извлечение знаний из эксперта, организацию знаний, обеспечивающую эффективную работу системы, и представление знаний в виде, понятном ЭС. Процесс приобретения знаний осуществляется инженером по знаниям на основе анализа деятельности эксперта по решению реальных задач.

Рис. 1.4. Технология разработки ЭС

На этапе *тестирования* эксперт (и инженер по знаниям) в интерактивном режиме с использованием диалоговых и объяснительных средств системы проверяет компетентность ЭС. Процесс тестирования продолжается до тех пор, пока эксперт не решит, что система достигла требуемого уровня компетентности.

На этапе *опытной эксплуатации* проверяется пригодность ЭС для конечных пользователей. По результатам этого этапа может потребоваться существенная модификация ЭС.

Процесс создания ЭС не сводится к строгой последовательности перечисленных выше этапов. В ходе разработки приходится неоднократно возвращаться на более ранние этапы и пересматривать принятые там решения.

Контрольные вопросы:

Сформулируйте отличия ЭС от традиционных систем обработки данных.

Назовите примеры успешного применения технологии ЭС.

Объясните основные причины успеха современной технологии ЭС .

Дайте формальное определение продукционной системы (по Е.Посту и А.Ньюэллу).

Охарактеризуйте основные режимы работы ЭС.

Укажите состав и роли участников разработки ЭС.

Перечислите основные компоненты статической ЭС.

Поясните отличия архитектуры динамической ЭС от архитектуры стати-ческой ЭС.

Перечислите и охарактеризуйте основные этапы разработки ЭС

ЛИТЕРАТУРА

1. *Котов В. Е.* **Сети Петри.** - М.: Наука, 1984. -158 с.

2. *Мальцев А.А.* **Алгоритмы и рекурсивные функции.** - М.:Наука, 1965. - 391 с.

3. *Минский М. (Minsky M.)* **Вычисление и автоматы,- М.: Мир, 1971. - 364 с.**

4. *Попов Э.В.* **Экспертные системы. Решение неформализованных задач в** диалоге с ЭВМ.- М.Наука, 1987.-288с.

5. *Давеннорт Т.* **Эпитафия экспертным системам**//Компьютеруик.-1995. -27(185).

6. **Expert system saves 20 million L on pipeline management.**//C&I, 1994, July, p.31.

7. *Feigenbaum E. A.* **The art of artificial intelligence: Themes and case studies of knowledge engineering**//The fifth International Joint Conference on Artificial Intelligence. - Boston: MIT, 1977. - P. 1014-1029.

8. *Feigenbaum E. A.* **Themes and case studies of knowledge engineering**//Expert systenvin micro electronic age. - Edinburgh: Infotach Limited, 1979. - P.3 - 25.

9. *Harmon P.* **The Market for intelligent Software Products Intelligent Software Strategies**, 1992. - V.8. '2. - P.5 -12.

10. *Hayes-Roth F., Jacobstein N.* **The State of Knowledge-Based Systems.** //Communications of the ACM, 19 94, March. - V.37. - N3. - P.27 - 39.

11. *Loofbourrow T.* **Экспертные системы еще живы.** Компьютеруик. - 1995, 5-11 октября. -36(194). - 21 с.

12. *Newell A.* **Production systems: models of comtrol structures**//Visual information processing. - New York: Academic Press, 1973. - P. 463 - 526.

13. *Newell A., Simon M.A.* **Human problem solving.** - Englewood Cliffs, New Jersey: Prentice-Hall, 1972.

14. *Perky D.R.* **Migration to Open Systems. Taming the Tiger.** Mc Graw-Hill, 1993. - P.252.

Лекция 11: Способы характеристики динамических систем

План:

1. Анализ состояния статических экспертных систем
2. Анализ состояния динамических экспертных систем
3. Основные производители ИС для ЭС РВ

1. Анализ состояния статических экспертных систем

С точки зрения введенной в гл. 3 классификации состояние статических ЭС и ИС можно охарактеризовать следующим образом.

По типу приложений большинство ЭС являются интегрированными и открытыми. ЭС, реализуемые на рабочих станциях, как правило, используют архитектуру клиент-сервер. Однако на ПК и больших ЭВМ архитектура клиент-сервер еще не стала доминирующей.

По масштабу ЭС (типу ЭВМ) безусловным лидером, по данным 1995 г. (см. гл. 2), являются ЭС, реализованные на рабочих станциях (более 80% объема продаж), затем идут ЭС на больших ЭВМ (12% объема продаж), ЭС на символьных ЭВМ (6% объема продаж) и, наконец, ЭС на ПК и МАК (2% объема продаж).

По типу проблемной среды статические ЭС разрабатываются в средах типа 1, 2 и 3 (см. гл. 3). При этом в средах типа 1 и 2 используются простые ЭС, реализованные на ПК, а в средах типа 3 - сложные ЭС, реализованные, как правило, на рабочих станциях (следующее место занимают большие ЭВМ и символьные ЭВМ).

ИС для сред типа 1 и 2 представляют данные в виде атрибутов и их значений, а исполняемые утверждения - либо в виде дерева решений (без правил), либо в виде специализированных (частных) правил (см.гл. 6).

Можно сказать, что частные правила являются естественным и наиболее распространенным способом представления поверхностных знаний. При использовании частных правил в процессе решения задачи удается избежать трудоемкой операции сопоставления. Наличие в правиле прямых адресных ссылок позволяет (для повышения эффективности) в режиме приобретения знаний компилировать правила в сеть вывода, определяющую множество допустимых решений, что дает возможность свести процесс решения задачи на этапе консультации не к генерации решения, а к выбору его из множества допустимых.

В принципе с помощью частных правил можно представлять и проблемные среды с общими знаниями. Однако это в ряде случаев будет приводить к значительным неудобствам на этапе приобретения знаний. Действительно, по определению, среды типа 3 характеризуются общими знаниями, т.е. знания эксперта об области экспертизы выражаются общими понятиями, отсутствующими в специализированных правилах. Поэтому для представления общих знаний (например, правила о вычислении некоторой характеристики любого объекта данного класса) придется представлять их в конкретном виде (писать для каждого объекта аналогичное правило, т.е. вместо одного правила вводить столько правил, сколько имеется объектов).

Проблемные среды типа 3 характеризуются изменяемым составом знаний (т. е. количество экземпляров некоторой сущности определяется только в процессе использования ЭС, а не на стадии приобретения знаний) и наличием общих знаний (естественно, могут присутствовать и конкретные знания). При представлении сред типа 3 избежать интерпретации невозможно, т.е. при использовании некоторого общего правила в процессе консультации необходимо выполнять операции сопоставления, в результате которых анализируются все экземпляры упоминаемого в правиле класса объекта.

Подчеркнем, что наличие сопоставления позволяет хранить правила в общем виде, т.е. описывать взаимоотношения между классами, а не между их экземплярами, что значительно уменьшает количество правил, необходимых для описания области.

Для представления сред типа 3 в последнее время используются гибридные ИС, в которых возможности объектно-ориентированного программирования объединяются с общими прави-

лами. Тенденция объединить правила и объекты (фреймы) существовала в традиционных ИС с 1985 г. (например, OPS 5, Personal Consultant Plus, Nexpert Object), однако в этих ИС (по сравнению с современными ИС для областей типа 3) либо реализовано ограниченное сопоставление (например, отсутствует возможность сопоставления в правилах, которые объединяют несколько объектов), либо реализовано ограниченное объектно-ориентированное окружение (отсутствуют классы методов и (или) посылка сообщений), либо они реализованы на диалектах языка Лисп, а не на языках традиционного программирования. Примерами современных гибридных статических ИС являются: ADS (5.1), Level 5 Object, Kappa.

Рассмотрим более подробно ИС, распространенные в США. Сначала проанализируем ИС общего назначения (малые, средние, большие и символьные), а затем проблемно/предметно-ориентированные. Перед тем как перейти к рассмотрению ИС, подчеркнем следующее. Многие ИС реализованы на нескольких типах ЭВМ (например, ПК, рабочие станции, символьные ЭВМ), т.е. их можно было бы отнести к различным типам ИС. Однако обычно ИС относят к тому типу, в котором оно пользуется наибольшим спросом. Например, ИС GURU работает на IBM PC и MicroVax, но его относят к ИС среднего типа, а не малого, так как в США это средство в основном используется на MicroVax.

Ниже приведены характеристики и названия ИС общего назначения по типам используемых ЭВМ.

Характеристики ИС для ЭС (США)

Малые ИС (23 фирмы):

1. Представление: специализированные правила и простые механизмы вывода.
2. Изолированные и интегрированные ЭС.
3. Доступ к БД (dBase III) и интегрированным пакетам (Lotus I-2-3)
4. Оперативная память 512 Кбайт - 4 Мбайта; жесткий диск не обязателен.
5. Встречаются индуктивные методы и гиперсредства.
6. Среда программирования. MS Windows, C, Pascal.
7. Средняя цена: 500 дол

Средние ИС (10 фирм):

1. Представление: гибридные средства; специализированные и общие правила; классы объектов
2. Интегрированные ЭС.
3. Доступ к БД (Oracle, dBase, RDb, DB2, VMS) и Lotus, X Windows
4. Оперативная память 8 - 32 Мбайт; требуется жесткий диск.
5. Встречаются индуктивные методы и гиперсредства.
6 Среда программирования: UNIX, WindowsNT, C, Pascal, Modula II, Fortran.
7. Средняя цена общие ИС - 6500 дол.; ориентированные ИС - 20 000 дол.

Большие ИС (5 фирм):

1 Представление: гибридные средства; специализированные и общие правила, классы объектов
2 Интегрированные ЭС.
3. Доступ к БД (DL/1, DB2, Oracle, Sybase, SQL/DS, ADABAS).
4. Оперативная память 8 - 64 Мбайт; требуется жесткий диск.
5. -
6. Среда программирования: MVS, Unix, C, Pascal, PL/1.
7 Средняя цена: 80 000 дол.

Символьные ИС (5 фирм):

1. Представление: гибридные средства; общие и специализированные правила, классы объектов; альтернативные меры.
2. Интегрированные ЭС.
3. Доступ к БД (dBase, DB2, IMS, Oracle).
4. Оперативная память 8 - 32 Мбайт; требуется жесткий диск.
5. -
6. Среда программирования: MVS, Unix, диалекты Лисп.
7. Средняя цена 12 500 дол.

Наиболее популярные ИС для ЭС (США)

Большие ИС:

1. AionDS 5.1 (Trinzic).
2. KBMS (Trinzic), ART (Inference).

Средние ИС:

1. Nexpert Object (Neuron Data), ProKappa(Intellicorp), Art-IM, ART Enterprise (Inference), Level 5 Object (IBI).

Малые ИС:

1. VP Expert (Paperback Software), 1st Class (1st Class Expert System).
2. Personal Consultant Easy, Procedure Consultant, Crystal.

Символьные ИС:

1. KEE (Intellicorp), ART (Inference).
2. Gold Works (Golden Hill), Mercury (AIT).

Заметное использование в США имеют 23 малые коммерческие ИС. Наиболее популярны следующие ИС: 1st-CLASS FUSION (фирма 1st-CLASS Expert Systems Inc.), VP Expert (Paperback Software), Exsys (Exsys Inc.), Procedure Consultant и Personal Consultant Easy (TI.), Level5 (Information Builders Inc.). Кроме того, распространены такие ИС: Crystal (Intelligent Environment Ltd.), Expert Common, OPS.5, Expertfacts, ExperOPS5 (Expertelligence), Expert Edge (Helix Expert Systems Ltd), KDS 2&3 (KDS Corp.), PC Expert Professional (Software Artistry), Instant Expert+ (Human Intellect Systems), Intelligent Developer (Hyperpress Publishing Corp.) и др.

Большинство этих малых ИС (см. выше) имеют доступ к БД (обычно dBase) и интегрированным пакетам (обычно Lotus 1-2-3), требуют от 512 Кбайт до 4 Мбайт оперативной памяти, жесткий диск не обязателен. Некоторые ИС используют индуктивные методы приобретения знаний (например, 1st-CLASS FUSION, KDS 2&3, Super Expert и др.). Большинство малых ИС, распространенных в США, реализованы на языках Си и Паскаль. Необходимо подчеркнуть, что существуют такие ИС, как Instant Expert+, Intelledgent Developer, Level 5, которые сопряжены со средствами обработки гипертекстов (Hypertext, HyperCard).

Из малых ИС, разработанных в России (табл. 4.1), можно назвать следующие: ЭКО, МОДИС, SIMER+MIR, ЭКСПЕРТ, ЛЭДИ, МЭС, ЭСПЛАН, ФИАКР, ПИЭС, ЗНАТОК, ЭКРАН, ШЕДЛ и др. Надо подчеркнуть, что многие из отечественных ИС (малых, средних и больших) скорее всего не поддерживаются, так как большинство коллективов разработчиков распалось.

Таблица 4.1 - ИС для России

ИС	Предприятие	Язык программирования, ЭВМ, требования ОЗУ
ЭКО	Рос НИИ ИТ и АП, Москва	С, PC, 300 Кбайт
SIMER+MIR	Институт программных систем АН, Переяславль-Залесский	TURBO C, PC
ИНТЕРЭКСПЕРТ	Центр программных систем, Тверь	
ЭКСПЕРТ-МИКРО	Москва	
СПЕЙС	ВНИИСИ, Москва	Lisp, PC
ШЕДЛ	НОВИНТЕХ, Москва	Язык ДЕКЛ, PC, 240 Кбайт
ЭСПЛАН	Москва	Turbo Prolog, PC, 400 Кбайт
ЛЭДИ	ИНКОММЕД, Москва	MuLisp
XSIMP	ЭПСИЛОН-СП, Москва	Lisp, PC, VAX
КРИС	Москва	-
ТЕТ-А-ТЕТ	Москва	PC, 400 Кбайт
OPS/ST	ИНФОТЕХ, Москва	Smalltalk, PC

Аргумент	Интеграл, Москва	Си, PC, PS/2, 300 Кбайт

Средние ИС в США используются в основном на рабочих станциях (Sun, HP, IBM, DEC), хотя встречаются случаи использования на PC и MAC. В США заметное распространение имеют не более 6 ИС [3]. Наиболее популярны следующие средние ИС: Level 5 (Information Builders Inc.), Nexpert Object (Neuron Data.), ART IM (Inference Corp.), ADS (Trinzic), Kappa (фирма Intellicorp.)

Большинство средних ИС имеют доступ к БД (к dBase, DB2, Oracle, IMS и т. п.), интегрированным пакетам (Lotus 1-2-3), требуют от 8 до 32 Мбайт оперативной памяти, жесткий диск обязателен (см. выше). Ряд средних ИС генерирует SQL-коды (Nexpert Object, Nexus). Средние ИС используют гибридные представления, специализированные и общие правила. Среди них есть ИС, ориентированные на индуктивные методы приобретения знаний (TIMM, RuleMaster). Большинство средних ИС написаны на языке С , используются также языки Pascal, Modula II, Fortran.

Необходимо подчеркнуть, что фирмы, разработавшие средние ИС, переносят их на большие ЭВМ (например, Nexpert Object -Neuron Data) и Level 5 Object (Information Builders Inc.), а фирмы, разработавшие большие ИС, переносят их на рабочие станции и PS/2, например ADS-5.1 (Aion Corp.).

Среди отечественных средних ИС можно отметить НЭКС, ЭКСПЕРТ-МИКРО, ЭКСНА, КОНС-ПРОЛОГ, ПРОДУС и др.

Большие ИС до 1990 г. развивались слабо. Так, по состоянию на 1990 г. в США значительное распространение получили только пять ИС: Aion Development System, KBMS (Trinzic), Expert System Environment (ESE), Knowledge Tool, TIRS (IBM), среди которых лидером является Aion Development System (ADS, версия 5.1) (Trinzic). В 1990 г. к списку больших ИС добавился ряд ИС, бывших в перечне средних, среди них Level 5 Object и ART-IM. Большие ИС имеют доступ к нескольким сложным БД (DB2, SQL/DS, DL/1, IDMS, ADABAS), генерируют SQL-коды, требуют память 8 - 64 Мбайт, написаны на языках Си, Паскаль, ПЛ/1.

Среди отечественных больших ИС можно отметить: НЭКС, ОПС-86, МЕДИФОР, КОМФОРТ, КОНСУЛЬТАНТ-2, РЕЛЯП и др.

Символьные ИС написаны на диалектах LISP и часто предназначены для Lisp-машин. Символьные ИС ориентированы на использование большой оперативной памяти (обычно 8 - 32 Мбайт). Многие из них имеют доступ к БД (dBase, DB2, IMS/DB, Oracle и т.п.) и некоторые генерируют SQL-код (см. выше).

Объем продаж символьных ИС с 1990 г. начал заметно сокращаться. Ведущую роль среди ИС этого класса играют KEE (Intellicorp) и его модификация IBM/KEE (IBM и Intellicorp). Цена на символьные ИС и на KEE, в частности, существенно зависит от типа ЭВМ. Значительное распространение получили ИС ART (Inference Corp.), Eloquent (Eloquent Systems Corp.). Необходимо подчеркнуть, что если 5 лет назад символьные ИС выполнялись только на символьных ЭВМ, то теперь почти все ИС этого типа могут выполняться и на многих ЭВМ традиционной архитектуры.

Отечественные символьные ИС, достигшие коммерческой стадии, нам не известны.

В *проблемно/предметно-ориентированных* ИС можно выделить следующие поднаправления:

• ИС для динамических экспертных систем реального времени, используемых в управлении технологическими процессами и имитационном моделировании (см. п. 4.2);

• ИС для систем советчиков (help-desk application) (см. п. 2.2);

• ИС для систем, основанных на прецедентах (см. п. 2.2).

Проблемно/предметно-ориентированные ИС в США активно развиваются. Они разрабатываются для всех типов ЭВМ. Цены на проблемно-ориентированные ИС зависят от сложности задачи, типа ЭВМ. средняя цена приблизительно 20 тыс. дол. Средняя цена на предметно-ориентированные ИС приблизительно 24 тыс, дол. Перечислим те проблемы и области, для которых разработаны ориентированные ИС. Для проблемно-ориентированных ИС - это диагностика оборудованея - Test Bench (TI и Carnegie Group); Diagnostic Reasoning Template (Coherent Thought); интеллектуальный вход-выход к реляционным БД (Genesis V (Help/Systems Inc.);

ProGenisis (Quantum in KNOWvations); поддержка разработчика планов и программ (Service /Maintenance Planner (Carnegie Group) и др.

Предметно-ориентированные ИС разработаны для следующих областей: разработка финансовых приложений - Cogensys Judgement Software (Cogensys Corp.); диагностика электронного оборудования -ICAT (Automated Reasoning Corp.); автоматизация конструирования - ICAD (ICAD Inc.); генерирование планов - Intelligen (CIMTelligence Corp.); поиск повреждений - CAIS (Rosh Intelligent Systems Inc.); управление производством - Flexis ToolSet (Savoir Systems Group) и др.

Кроме перечисленных выше поднаправлений проблемно/предметно-ориентированных ИС можно указать еще коммерческие базы знаний (help-desk application) и ИС, ориентированные на приобретение знаний.

Коммерческие БЗ используются совместно с коммерческими ИС и предлагаются третьими фирмами, не являющимися разработчиками ИС. Эти фирмы обеспечивают разработку, сопровождение, модификацию и маркетинг БЗ. Так, например, фирма ServiceWare предлагает базы знаний, называемые "KnowledgePaks", которые могут быть использованы с продуктами фирмы Inference CBR Express и CasePoint. Каждая БЗ KnowledgePak ориентирована на поиск и устранение конфликтов в программном обеспечении типа Windows, Word, Lotus 1-2-3 и содержит решение от 450 до 500 различных задач.

ИС, ориентированные на приобретение знаний, в настоящее время обычно не выделяются в самостоятельный продукт, а поставляются в составе ИС общего назначения. Тем не менее коммерческие ИС приобретения знаний в США существуют и могут быть охарактеризованы следующим образом.

1. Средства приобретения знаний, основанные на деревьях решений. Средства этого типа обычно включаются в состав ИС общего назначения. Типичными примерами являются ИС общего назначения для ПК Procedural Consultant (фирма TI) и ИС общего назначения для рабочих станций DEC и VAX Dession Expert (фирма DEC).

2. Индуктивные средства приобретения знаний. Эти средства либо включаются в состав ИС общего назначения (например, VP-Expert, 1st-CLASS FUSION, KDS 2&3 и др.), либо распространяются как самостоятельное средство (например, продукт BEAGLE фирмы VRS Consulting для PC и для VAX).

3. Средства приобретения знаний, базирующиеся на психологической теории. Наиболее широкое использование имеет ИС ETS (фирма Boeing), реализованное на символьной ЭВМ и базирующееся на методе репертуарных решеток Дж. Келли. Данное средство фирмой не продается, а используется только для внутренних приложений фирмы (известно несколько сотен приложений средства).

4. Средства приобретения знаний, ориентированные на конкретные ИС общего назначения. Типичным примером является ИС KAT, предназначенное для помощи в создании БЗ для ИС Level 5, и средство Nextra, которое упрощает приобретение знаний для ИС Nexpert Object. Nextra сочетает индуктивный метод и метод репертуарных решеток и реализовано на ПК Macintosh.

5. Средства приобретения знаний общего назначения. Данные средства не ориентируются ни на какое ИС, они используются не только для создания БЗ ЭС. Их цель помочь разработчику в накоплении, редактировании и управлении знаниями о конкретной проблеме. Типичным примером является ИС CAMEO (Arthur D.Little - ADL). Это ИС не продается, но доступно клиентам, работающим совместно с ADL.

2. Анализ состояния динамических экспертных систем

Как видно из данных, приведенных в гл. 2, среди всех видов ИС наиболее динамично развиваются ЭС реального времени. В 1995 г. объем продаж ЭС реального времени составил примерно 70% рынка проблемно/предметно-ориентированных СОЗ и был равен 38 млн дол. (в 1988 г. - 3 млн. дол.). Значимость ИС и ЭС реального времени (РВ) определяется не столько их бурным коммерческим успехом (хотя и это достойно тщательного анализа), но в первую очередь тем, что только с помощью подобных средств создаются стратегически значимые приложения в таких областях, как: управление непрерывными производственными процессами в химии, фармакологии, производстве цемента, питания и т.п.; аэрокосмические исследования, транспорти-

ровка и переработка нефти(газа), управление атомными и тепловыми электростанциями, финансовые операции, связь и многие другие.

В последнее время на основе динамических ИС начинают создаваться ИС для интеллектуального имитационного моделирования, используемые в реинжиниринге (реорганизации) бизнес-процессов (БПР) (см. Приложение 2). Интерес к ИС этого типа инициируется тем, что в отличие от статических ИС и ЭС, используемых, как указано ранее, для БПА, т.е. для автоматизации текущего состояния бизнеса, ИС для БПР используются для решения существенно более значимых и сложных задач, т.е. для "фундаментального переосмысления и радикального перепроектирования деловых процессов для достижения существенных улучшений в главных показателях деятельности компаний, таких, как стоимость, качество, услуги и темпы"[2].

Ниже перечислены некоторые области применения ЭС РВ, разработанных на базе ИС G2 (см. гл. 9). Всего на базе G2 разработано более 700 ЭС РВ, работающих более чем в 30 областях.

Области применения ЭСРВ (перечень фирм и характеристик приложений)

3M (США) - G2 используется на ряде заводов 3M в Миннесоте для управления технологическими процессами и поддержки принятия решений.

Caterpillar (США) - интегрированная система мониторинга и планирования для прокатного стана на базе распределенной системы, включающей G2 и Telewindows.

Camunsa (Испания) - автоматизированный, интеллектуальный гараж в Барселоне, разработанный к летним Олимпийским играм 1992 г. Гараж не требует присутствия людей и размещает 800 машин на том же пространстве, где при обычном подходе размещаются только 300.

Carpenter Technology Corp. (США) - CarTech использует DSP (ИС на базе G2) для моделирования операций горячего прокатного стана и связанных с ним печей. DSP разрабатывает расписание печи и потока материалов, поступающих от печи на дальнейшую обработку.

Forsmark Nuclear Plant (Швеция) - система обеспечения безопасности и моделирования событий для ядерной электростанции. Содержит более 200 правил. Использует более 130 диаграмм различной формы для отображения процесса.

General Electric (США) - GE разработала ряд систем на базе G2: систему для наземных станций слежения за спутниками в GE Aerospace в Филадельфии; систему для производства и тестирования самолетных двигателей в Лин-не; предсказывающую систему для GE Nuclear в Сан Хозе, CA.

IBM (США) - MOM (Measurement of On-line Manufacturing) - система управления, разработанная для улучшения производства блоков памяти и питания на заводе IBM в Торонто и интегрированная в производственный процесс. MOM объединяет системы G2, Serveio's Gemstone OODBMS и последовательную SPS в единую систему управления и контроля за производством печатных плат, повышающую качество, окупаемость и производительность завода.

Intelsat (США) - система диагностики, мониторинга и контроля сети, разработанная за 4 месяца на базе G2. Обеспечивает помощь при восстановлении спутников путем мониторинга критических состояний и диагностики сбоев коммуникационных каналов до и во время их появления.

Lafarge Coppee (США) - 25 установок G2 на цементных заводах, расположенных по всему миру. Lafarge использует возможности нечеткой логики G2 для обеспечения замкнутого цикла управления мельничными установками.

Mrs.Baird's Bakery (США) - самая большая частная пекарня в США использует G2 для планирования и управления всем производственным процессом.

NASA/ Space Shuttle (США) - NASA использует G2 с октября 1988 г. в ряде систем для космических аппаратов, включая управление 38 реактивными двигателями, обеспечивающими маневрирование челнока. G2 обрабатывает данные от 16 000 датчиков в секунду, осуществляя проверку всех параметров от температуры до курса.

ЭС РВ решают следующие классы задач: мониторинг в реальном масштабе времени; системы управления верхнего уровня; системы обнаружения неисправностей; диагностика; составление расписаний; планирование; оптимизация; системы - советчики оператора; системы проектирования и т. п.

Традиционные статические ИС и ЭС не способны решать подобных задач, так как они не выполняют требования, предъявляемые к системам, работающим в реальном времени:

• представлять изменяющиеся во времени данные, поступающие от внешних источников, обеспечивать хранение и анализ изменяющихся данных;

• выполнять одновременно временные рассуждения о нескольких различных асинхронных процессах (задачах), т. е. планировать в соответствии с приоритетами обработку процессов, поступивших в систему;

• обеспечивать механизм рассуждения при ограниченных ресурсах (время, память). Реализация этого механизма предъявляет требования к высокой скорости работы системы, способности одновременно решать несколько задач (т. е. необходимо использовать операционные системы UNIX, VMS, Windows NT, но не MS DOS);

• обеспечивать предсказуемость поведения системы, т.е. гарантию того, что каждая задача будет запущена и завершена в строгом соответствии с временными ограничениями. Например, требование предсказуемости не допускает использования в ЭС РВ механизма сборки мусора, свойственного языку Lisp;

• моделировать "окружающий мир", рассматриваемый в данном приложении, обеспечивать создание различных его состояний;

• протоколировать свои действия и действия персонала, обеспечивать восстановление после сбоя;

• обеспечивать наполнение базы знаний (БЗ) для приложений реальной степени сложности с минимальными затратами времени и труда (необходимо использование объектно-ориентированной технологии, общих правил, модульности и т. п.);

• обеспечивать настройку системы на решаемые задачи (проблемно-предметная ориентация);

• обеспечивать создание и поддержку пользовательских интерфейсов для различных категорий пользователей;

• обеспечивать уровень защиты информации (по категориям пользователей) и предотвращать несанкционированный доступ.

3 . Основные производители ИС для ЭС РВ

ИС для создания ЭС реального времени впервые в 1985 г. выпустила фирма Lisp Machine Inc. Это ИС называлось Picon, и оно исполнялось на символьных ЭВМ Symbolics. Успех этого ИС привел к тому, что группа ведущих разработчиков Picon в 1986 г. образовала частную фирму Gensym, которая, значительно развив идеи, заложенные в Picon, выпустила в 1988 г. ИС под названием G2, версия 1.0. В настоящее время работает версия 4.2 и готовится к выпуску версия 5.0.

С отставанием от Gensym на 2 - 3 года ряд других фирм начал создавать свои ИС для ЭС РВ. В табл. 4.2 [1] приведен достаточно полный перечень всех фирм и объявленных ими продуктов. Следует отметить, что, несмотря на значительное количество объявленных ИС, в этом списке много либо незавершенных ИС, либо ИС, которые только с большой натяжкой могут быть отнесены к ИС для создания ЭС РВ. В настоящее время наиболее продвинутым ИС, безусловно, остается G2 (Gensym, США), следующие места со значительным отставанием (реализовано менее 50% возможностей G2 [1]) занимают RTworks - фирма Talarian, США, COMDALE/C (Comdale Techn., Канада), COGSYS (SC, США), ILOG Rules (ILOG, Франция).

Т а б л и ц а 4.2 – Коммерческие ИС для ЭС РВ (1993 г.)

№	Продукт	Фирма
1	G2	Gensym Corp, USA 100%
2	RTworks (R*Time, L*Star)	Talarian Corp, USA < 50 %
3	COMDALE/C	Comdale Tech, Canada
4	COGSYS	SC Scicon/Cogsys Comp, USA
5	ILOG Rules (XRete)	ILOG, France
6	Activation Framework	Real Time Intelli Systems,

		USA
7	Chronos	S20, France
8	Escort	PA Consultans, UK
9	Expert 90	Bailey, USA
10	Mercury KBE	Intelligence Tech, US A
11	Muse	Cambridge Consultans, UK
12	Montrex	Stone and Webster, USA
13	Promass	Unibit, UK
14	Rocky	Expert Edge, USA
15	RTAC M/Power	Mitech, USA
16	RTES	Knowledge systems Inc, USA
17	RT/AI	Intellisys, USA
18	RT Expert	Integrated Systems Inc, USA
19	SNAP	Template Software, USA
20	TDC Expert	Honeywell, USA

Из приведенного в табл. 4.3 сравнения видно, что G2 значительно превосходит ближайшего конкурента RTworks [4]. Сравнение G2 и RTworks проводилось путем разработки одного и того же приложения на этих ИС двумя организациями: NASA (США) и Storm Integration (США).

Таблица 4.3 Сравнительные характеристики G2 и RT works

Возможности ИС	*ИС*	
	G2	Rtworks
Объектно-ориентированная технология: связи между объектами отношения между объектами иерархия объектов	+ + +	+ - -
Представление знаний: правила (общие и специализированные) Процедуры динамические модели функции в ЕЯ	+ + + +	(Нет общих) - - (Только на Си)
Механизм рассуждений: от данных от цели Сканирование метарассуждения (события, фокусирование на классах объектов или правил) одновременное выполнение правил и (или) процедур	+ + + + +	+ + + - -
Графическое определение объектов	+	-
Клонирование объектов и их групп	+	-
Графические пользовательские интерфейсы для различных категорий пользователей	+	(Нет собственной графики, используется ИС Dataviews)
Многопользовательская кооперативная разработка приложения	+	-
ИС запрограммировано на Си	+	+
Распределенное приложение	+ G2-G2	-

Приведем результаты сравнения G2 со следующими группами программного обеспечения (см. табл. 4.4) по 16 свойствам, характеризующим ЭС РВ [5]:

•статические ЭС (группа А);

• супервизорныё системы управления (группа В);

• ЭС реального времени, исключая G2 (группа С);

• ЭС G2.

Таблица 4.4 – Сравнительная характеристика G2 и программных продуктов групп A, B, C и D

№	***Свойства ЭС РВ***	A	B	C	D (G2)
1	Работа в реальном времени, внутренний планировщик, параллельные процессы рассуждения	-	+	+	+
2	Структурированный естественно-языковый интерфейс с управлением по меню и автоматической проверкой синтаксиса	-	-	-	+
3	Общие правила, уравнения и динамические модели, применимые к классам объектов	-	-	-	+
4	Обратный и прямой вывод, сканирование, фокусирование, использование метазнаний	-	-	-	+
5	Интеграция подсистемы моделирования с динамическими моделями	-	-	-	+
6	Структурирование БЗ, наследование свойств, понимание связей между объектами	-	-	-	+
7	Библиотеки знаний являются ASCII-файлами, портируемыми на любые аппаратные платформы без какого-либо дополнительного программирования	-	-	-	+
8	Развитый редактор для сопровождения базы знаний без программирования	-	-	-	+
9	Средства инспекции базы знаний	+	+	-	+
10	Средства управления доступом с помощью механизма авторизации пользователя и обеспечение желаемого взгляда на приложение	-	-	-	+
11	Средства трассировки и отладки БЗ	+	-	+	+
12	Интерфейс оператора, включающий график, диаграммы, шкалы, кнопки, редактор многослойных пиктограмм	-	+	-	+
13	Исполнение на ряде универсальных ЭВМ, включая рабочие станции DEC, HP, SUN, IBM, SG, Intell	+	+	-	+
14	Кооперация ЭС реального времени по сетевому протоколу TCP/IP или DECnet с другими приложениями	-	-	-	+
15	Удаленные окна, включая интерактивную многопользовательскую работу	-	-	-	+
16	Интерфейсы с источниками данных, обеспечивающие эффективную связь с внешними системами и базами данных	-	+	+	+

Общий итог по результатам сравнения 16 позиций таков:

1) в группе А реализовано 3 свойства из 16 (18% от функциональных возможностей G2);

2) в группе В реализовано 5 свойств из 16 (31% от функциональных возможностей G2);

3) в группе С реализовано 3 свойства из 16 (18% от функциональных возможностей G2).

Контрольные вопросы:

1. Приведите примеры современных гибридных инструментальных средств для статических экспертных систем.

2. Укажите основные характеристики инструментальных средств для каждого типа ЭВМ.

3. Приведите примеры статических и динамических экспертных систем.

4. Назовите основные направления использования проблемно/предметно-ориентированных ИС.

5. Назовите примеры применения технологии динамических экспертных систем.

6. Приведите результаты сравнения наиболее развитой динамической экспертной Системы G2 с другими классами экспертных систем.

7. Выделите, значимые параметры, по которым целесообразно проводить сравнение различных динамических экспертных систем.

ЛИТЕРАТУРА

1. *Clements B.R. and Preto F.* **Evaluting Commercial Real Time Expert System Software for Use in the Process Industries.** - C&I. - 1993. -P. 107-114.

2. *Hammer M. and Champy J.* **Reengineering the Corporation. A Manifesto for Business Revolution.** - New York: Harper Colins. - 1993.

3. **Intelligent Software Strategies.** - N2.- 1996.

4. *Moore B.*, **Memorandum**//Copyright. - 1993, April. Gensym Corporation.

5. Moore B. and others.. **Questions and Answers about G2**//Copyright. -1993. Gensym Corporation. - P. 26 - 28.

Лекция 12: Интеллектуальные операторы, осуществляющие формирование, отображение возможностей и умственных выводов, понятий в процессах понятия

План:

1. Механизмы вывода экспертных систем

2. Стратегии как механизмы управления

Анализ исследований в области искусственного интеллекта, проведенный А. Ньюэллом и М. Саймоном [6], позволил им выделить два основных понятия: символические системы и поиск. *Символическая система* есть набор символов, образующих символические структуры, и набор процессов. Процессы способны производить, разрушать и модифицировать символические структуры. Символ - это первичное понятие. *Символические структуры* могут рассматриваться как типы данных в некотором языке. Они обладают двумя основными свойствами:

• могут обозначать объекты, процессы и другие символические структуры;

• если они обозначают процессы, то они могут быть интерпретированы.

Символическая структура обозначает некоторую сущность (объект, процесс или другую символическую структуру), если символическая система может осуществлять поведение, определяемое данной сущностью, или может воздействовать на эту сущность. Система может интерпретировать символическую структуру, если структура обозначает некоторый процесс, и система может выполнить этот процесс.

А.Ньюэлл и М.Саймон [6] обосновали две гипотезы, на которых базируются исследования по ИИ: *гипотезу символических систем* и *гипотезу поиска.* Согласно первой гипотезе символические системы имеют необходимые и достаточные условия для осуществления интеллектуальных действий. Согласно второй гипотезе символические системы решают задачи с помощью поиска, т.е. они генерируют потенциальные решения и постепенно модифицируют их, пока последние не будут удовлетворять заданным условиям решения. Приведенные гипотезы разделяются большинством специалистов в области ИИ.

Можно утверждать также, что все существующие экспертные системы подтверждают их.

Общая схема функционирования управляющего компонента экспертной системы приведена на рис. 6.1. Управляющий компонент экспертных систем обычно называют *интерпретатором* (механизмом вывода). Задача механизма вывода состоит в том, чтобы на основании текущего состояния рабочей памяти определить, какой модуль и с какими данными будет работать. По окончании работы текущего модуля (правила) механизм вывода проверяет условия окончания задачи, и если они не удовлетворены, то выполняется очередной цикл. Каждый модуль (правило) снабжается образцом, т.е. описанием, указывающим, при выполнении каких условий этот модуль (правило) может приступить к работе.

В общем случае работа механизма вывода в каждом цикле состоит в последовательном выполнении четырех этапов: *выборки, сопоставления, разрешения конфликтов, выполнения* (рис. 6.2). С точки зрения теории работа механизма вывода зависит только от состояния рабочей памяти и от состава базы знаний. На практике обычно учитывается история работы, т.е. поведение механизма вывода в предшествующих циклах. Информация о поведении механизма вывода запоминается в памяти состояний (см. рис. 6.1). Обычно память состояний содержит протокол работы системы.

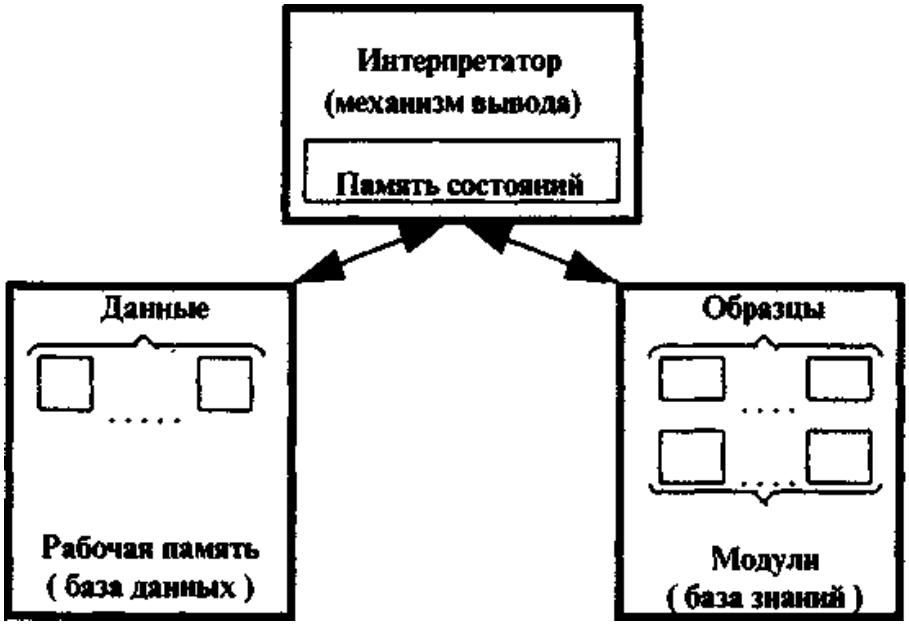


Рис. 6.1. Схема функционирования управляющей компоненты экспертной системы

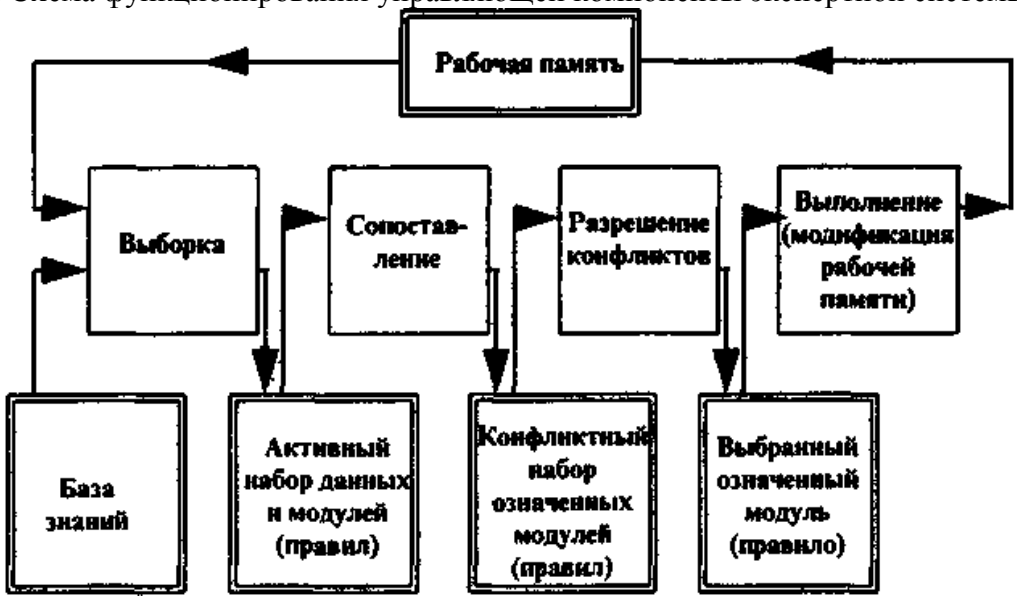


Рис.6.2. Цикл работы механизма вывода (интерпретатора)

В общем случае каждый из этапов использует в своей работе три источника знаний: рабочую память, базу знаний и память состояний. Для повышения эффективности функционирование системы на каждом из этапов направляется *стратегиями управления,* т.е. некоторыми эвристическими правилами. Возможности стратегий зависят от того, какие функции механизма вывода могут изменяться, а какие встроены в него жестко. Встраивание определенных функций в механизм вывода повышает эффективность его работы, но ограничивает степень воздействия на процесс функционирования. Как правило, в механизм вывода встраивают общую схему поиска решения (т.е. метод) (см. п. 6.3), а через стратегии (см. п. 6.2) управляют деталями поиска.

Рассмотрим теперь назначение и основные функции этапов, представленных на рис. 6.2. На этапе выборки осуществляется определение подмножества элементов рабочей памяти и подмножества правил (модулей) базы знаний, которые могут быть использованы в текущем цикле. При реализации этапа выборки обычно используется один из двух подходов. Первый подход, называемый иногда *синтаксической выборкой,* выполняет грубый отбор знаний, данных и (или) правил, которые могут быть полезны в текущем цикле. Основанием для выборки знаний в данном случае являются формальные (синтаксические) знания, встроенные в систему разработчиком.

Второй подход, называемый иногда *семантической выборкой,* осуществляет отбор знаний на основании таких сведений, как модель предметной области, разбиение задачи на подзадачи, текущие цели и т.п. Семантические знания, используемые на этапе выборки, вводятся в систему экспертом, например, в виде *метаправил.* В результате работы этапа выборки происходит вы-

деление активного набора данных и активного набора правил (модулей), т.е. осуществляется фокусирование внимания системы на определенном ограниченном количестве данных и правил (модулей).

На этапе сопоставления определяется, какие активные модули и на каких активных данных готовы к работе. Модуль готов к работе, если среди активных данных есть данные, удовлетворяющие условиям этого модуля, указанным в его образце. Такие модули называются *означенными.* Результатом работы этапа сопоставления является набор означенных модулей. Набор означенных модулей часто называют *конфликтным набором,* подчеркивая этим тот факт, что к работе готовы все модули набора, но механизм вывода не знает еще, какой из них предпочесть. Теоретически сопоставление выполняется в каждом цикле работы механизма вывода над всеми активными знаниями, т.е. образцы всех активных модулей сопоставляются со всеми активными данными. На практике в целях повышения эффективности все означивания не вырабатываются заново на каждом очередном цикле.

На этапе разрешения конфликтов механизм вывода выбирает из конфликтного набора те означивания, которые будут выполняться в текущем цикле. На данном этапе интерпретатор оценивает означенные модули с точки зрения их полезности при достижении текущей цели. Подчеркивая этот факт, данный этап иногда называют этапом планирования.

На этапе выполнения осуществляется исполнение правил (модулей), выбранных на этапе разрешения конфликтов. В ходе этого этапа осуществляется модификация рабочей памяти, выполняются операции ввода-вывода и изменяется память состояний интерпретатора.

На рис. 6.1 и 6.2 приведено обобщенное описание управляющей компоненты ЭС. В настоящее время при реализации этой общей схемы используются две основные архитектуры. Различия в реализации общей схемы являются в первую очередь следствием различной сложности используемых модулей. При одном подходе модулями являются относительно небольшие автономные фрагменты знаний, представляемые в виде правил (в частном случае - в виде продукционных правил), которые понятны пользователю (эксперту), не знакомому с программированием. Этот подход часто называют подходом, использующим *управляемые образцами правила,* а системы, основанные на данном подходе, - *системами, управляемыми правилами.* При втором подходе в качестве модулей используются большие сложные автономные фрагменты знаний, представленные в виде программ, смысл которых, конечно, не может быть понятен непрограммисту. Этот подход называют подходом, использующим *управляемые образцами модули.*

Типичным примером первого подхода являются системы OPS 5, MYCIN и другие, а примером второго подхода - система HEARSAY-II. Оба подхода используют управление, основанное на сопоставлении образцов, т.е. по окончании работы одного модуля его преемником является один из тех модулей, образцы которых будут означены элементами рабочей памяти. В обоих подходах взаимосвязь модулей (информационная и управляющая) осуществляется через общую память. В первом подходе модуль называют правилом, а в остальном терминология и структура построения системы соответствуют схеме, приведенной на рис. 6.1.

Во втором подходе используется другая терминология. Рабочая память называется *"классной доской",* конфликтный набор - *агендой,* программы, разрешающие конфликты, - *политическими модулями,* а модули - *источниками знаний.* Каждый источник знания имеет образец. Если образец некоторого источника знаний сопоставляется с данными на "классной доске", то этот означенный источник знания заносится в агенду. Агенда представляет собой упорядоченный список работ, готовых к выполнению. Под работой понимается источник знания с описанием данных, которые он может обрабатывать в текущий момент. В каждый момент времени с агендой работает один из политических модулей. Выбор политического модуля зависит от обрабатываемой гипотезы. Политический модуль переупорядочивает агенду и выбирает некоторую работу на исполнение. Результатом выполнения работы является изменение содержимого "классной доски". Политические модули, являясь источниками знания, также заносятся в агенду и выбираются на исполнение. Множество политических модулей обеспечивает разнообразие способов выработки управляющих решений. Модификация систем подобного типа достигается за счет независимости источников знания.

Концепция управляемых образцами модулей позволяет решать более сложные задачи и строить более эффективные системы Однако этот подход затрудняет реализацию объяснительных способностей и способностей по приобретению новых знаний. Использование в данном

подходе больших фрагментов знаний связано с разработкой для каждой проблемной области своих политических модулей, осуществляющих детальное планирование и использование знаний. Кроме того, возможности данного подхода к решению задач различных классов ограничены номенклатурой имеющихся модулей и способами их взаимодействия.

Концепция управляемых образцами правил позволяет (за счет ограниченной сложности используемых фрагментов знания и представления их в понятном для пользователя виде) решать разнообразные задачи, обеспечивая развитые объяснительные способности и способности по приобретению знаний.

Итак, оба рассмотренных подхода, несмотря на некоторые различия, являются вариантами одного и того же метода управления. Подход с управляемыми образцами правилами обладает большей декларативностью используемого представления знаний и в связи с этим большей универсальностью.

Рассмотрим более подробно подход управляемых образцами правил, так как он получил наиболее широкое распространение. Механизмы вывода (МВ), используемые в этом подходе, различаются следующим:

• используются только частные знания или допускаются и общие знания;

• каким способом (при наличии общих знаний) обеспечивается сокращение вычислительных затрат на выполнение операций выборки, сопоставления и разрешения конфликтов.

Только простейшие ЭС не используют общих знаний. Механизм вывода в этих инструментальных средствах сводит решение задачи к поиску пути в дискриминационном дереве (графе), которое компилируется на этапе приобретения знаний. Развитые ЭС базируются на технологии объектно-ориентированного подхода (см. Приложение 1) и, как следствие, используют частные и общие знания. Недостатки использования частных знаний сводятся к увеличению в 10 и более раз количества правил по сравнению с использованием общих правил, что приводит к необозримости базы знаний (БЗ) конечным пользователем; увеличению времени разработки и отладки БЗ; усложнению модификации приложения, так как вместо изменения одного общего правила надо изменять десятки подобных частных правил.

Представление исполняемых утверждений (правил, процедур, действий) в общем виде обеспечивает применение одного утверждения к множеству однотипных конкретных объектов, что значительно снижает трудоемкость накопления базы знаний, упрощает сопровождение (модификацию) приложения, минимизирует ошибки при отладке БЗ. Использование общих утверждений при всех их плюсах создает одну серьезную проблему. Исполнение общих правил требует значительных вычислительных затрат, которые минимизируются двумя рассматриваемыми ниже подходами:

• использование синтаксических средств типа алгоритма Rete [1], сокращающих на каждом цикле работы МВ перебор при выполнении операции сопоставления;

• использование семантических средств, сокращающих перебор при выполнении операции выборки и разрешении конфликтов путем использования метапланирования и фокусирования на объектах или правилах [5].

Суть *синтаксических подходов* ускорения процесса сопоставления сводится к следующему. В общем случае в каждом цикле работы МВ для получения означиваний (конфликтного набора) требуется заново сопоставить все правила из БЗ (из ее активного подмножества) со всеми экземплярами объектов из рабочей памяти (из ее активного подмножества). Задача синтаксических методов состоит в том, чтобы избежать на каждом цикле работы МВ повторных сопоставлений правил и данных, так как значительная часть означиваний (i-l)-го цикла может быть использована в i-м цикле.

Наиболее популярным синтаксическим методом, минимизирующим сопоставления, является алгоритм Rete [1]. В данном алгоритме каждое правило рассматривается как один или несколько образцов, с каждым из которых связывается список всех активных сопоставляющихся с ним элементов рабочей памяти (РП). Этот список модифицируется на каждом цикле изменения РП. Если некоторый элемент вводится в РП (или модифицируется), то МВ находит все образцы (правила), которые сопоставляются с этим элементом, и добавляет его к спискам соответствующих образцов. При удалении некоторого элемента из РП МВ устраняет его из списков всех образцов, которые с ним сопоставлялись. Таким образом, МВ, запоминающий указанную информацию, не сопоставляет всю РП со всеми правилами.

Описания изменений рабочей памяти, поступающие в Rete-алгоритм, называются *признаками.* Признак представляет собой упорядоченную пару, состоящую из метки и списка элементов. В простейшем исполнении для Rete-алгоритма необходимы две метки: "+" и "-", означающие соответственно добавление в РП или устранение чего-то из рабочей памяти. Если некоторый элемент модифицируется, то на вход алгоритма поступают два признака: один указывает, что старая форма элемента устраняется из рабочей памяти, а другой - что новая форма добавляется.

Задача алгоритма состоит в том, чтобы определить, какие правила будут удовлетворены поступившими на вход признаками (т.е. изменениями элементов рабочей памяти). Простейшее решение этой задачи состоит в сопоставлении признаков со всеми правилами. При таком подходе на каждом цикле будет множество повторных ("лишних") просмотров. Для того чтобы избежать лишних сопоставлений, образцы правил преобразуются в сетевую структуру, которая выполняет функции индексирования правил. Сеть образцов представляет собой разновидность дискриминационной сети. В вершинах дискриминационной сети проверяются характеристики элементов. В зависимости от результатов проверок признак, поступивший на вход сети, пройдет через сеть по тому или иному пути (путям) и в результате сообщит, какие правила удовлетворяют этому признаку. Сеть составляется специальной программой (компилятором) на основе анализа условий правил, хранимых в рабочей памяти.

Семантические подходы, как правило, применяются на этапе выборки и разрешения конфликтов. Как отмечалось, на этапе выборки механизм вывода фокусируется на определенном подмножестве элементов рабочей памяти и подмножестве правил базы знаний, которые могут быть использованы в текущем цикле.

Обычно выделяют два типа выборки: *простую выборку* и *иерархическую выборку.* Простая выборка характеризуется тем, что выбираемые сущности рассматриваются как сущности одного уровня. Поясним суть простой выборки на примере выборки правил. В данном случае при появлении нового элемента в рабочей памяти те правила, которые содержат этот элемент в условии правила (при поиске от данных), помечаются как активные. При удалении элемента из памяти метки у соответствующих правил снимаются. Выборка в данном случае сводится к выбору из всего множества правил тех, которые помечены.

При *иерархической выборке* объекты (правила, данные) разбиваются на иерархические подмножества (классы). Выборка в данном случае состоит в использовании метаправила для выбора одного из классов. При этом классы могут быть как непересекающимися, так и пересекающимися. Следует отметить, что введение иерархии правил неизбежно влечет за собой и иерархию данных, что, к сожалению, не всегда явно признается разработчиками систем. Действительно, метаправила в отличие от правил применяются не к объектам предметной области, а к метаданным, т.е. уместно говорить о появлении в рабочей памяти данных и метаданных.

Объекты, подлежащие выборке на текущем цикле, задаются либо по имени, либо по описанию свойств. При задании по имени указывается либо перечень объектов (данных, правил), либо перечень имен классов, описывающих объекты. При задании объектов через описания свойств указывают не имена объектов (классов объектов), а перечень свойств, которыми эти объекты должны обладать.

Рассмотрим, какие семантические подходы к сокращению перебора существуют в одной из наиболее популярных на сегодняшний день экспертной системе G2 (см. гл. 9). В G2 используются следующие варианты фокусирования:

• фокусирование на классе объектов или на специфическом (конкретном) объекте, например focus on terminal-1, т.е. фокусировать на терминале-1;

• фокусирование на некотором правиле через заданный временной интервал (длина интервала указывается в атрибуте правила scaninterval (интервал сканирования)); если правило является общим, то G2 возбуждает каждый пример этого правила (каждое означивание) через заданный интервал;

• фокусирование на классе (группе) правил, например invoke safety rules (возбудить правила безопасности), где safety (6езопасность) -название категории правил, на которой разработчик хочет сфокусировать внимание МВ;

• фокусирование на классе правил для конкретного объекта, например invoke safety rules for tank-4 ("возбудить правила безопасности для емкости-4).

Для обеспечения перечисленных выше вариантов фокусирования правила в G2 имеют следующие атрибуты: focal-classes (в качестве значения этого атрибута указываются классы объектов из БЗ, с которыми ассоциируется данное правило); focal-objects (в качестве значения этого атрибута указываются специфические объекты из БЗ, с которыми ассоциируется данное правило); categories (в качестве значений этого атрибута указывается имя класса правил, к которым относится данное правило).

Заметим, что механизм фокусирования внимания на классе объектов и (или) правил может рассматриваться как механизм планирования. Действительно, план можно представить либо декларативно, либо процедурно. При декларативном представлении план задается последовательностью состояний: начальное (начальные) состояние, промежуточные состояния от 1 до k, конечное состояние. В G2 промежуточные состояния могут быть заданы декларативно последовательностью focus, фокусирующейся на соответствующих классах объектов. При процедурном представлении план задается последовательностью исполняемых утверждений (процедур, правил, действий), переводящих начальное состояние в конечное. В G2 последовательность исполняемых утверждений может быть задана последовательностью действий invoke и scan над соответствующими классами правил. Очевидно, что в G2 план можно представить в виде смешанной последовательности действий focus, invoke и scan.

В G2 введены средства, позволяющие, используя семантику приложения, ограничить сферу действия общих правил, т.е. предотвратить применение, общих правил к бесперспективным сущностям. Основа этих средств базируется на использовании в общих правилах связей (connection) и отношений (relation), существующих между сущностями приложения.

2. Стратегии как механизмы управления

Необходимость использовать в экспертных системах нетрадиционные методы управления вызвана в первую очередь неформализованностью решаемых ими задач [1]. Особенности неформализованных задач с точки зрения организации управления приводят к тому, что процесс решения таких задач не удается представить в виде детерминированной последовательности правил (программных модулей). Здесь в некоторый текущий момент к исполнению пригодно несколько правил (или одно правило, но над разными данными), причем не существует надежной информации, позволяющей предпочесть одно правило другому. Задача управляющей компоненты состоит в том, чтобы обеспечить функционирование системы в подобных условиях. Так, например, в относительно простой ЭС MYCIN в любой момент в среднем пригодны к использованию 50 правил из 400 [1].

В традиционном программировании модули (программы) вызываются по имени. Поэтому программист в ходе составления и отладки программы должен выявить множество всех мыслимых ситуаций, которые возникнут в ходе работы общей программы при различных входных данных; в каждой точке, где завершается работа одного модуля, в явном виде (указав имя модуля и перечень используемых им данных) необходимо запрограммировать однозначный переход к очередному модулю. Такая организация управления не позволяет решать неформализованные задачи. Основные отличия управляющей компоненты экспертных систем от традиционных механизмов управления состоят в следующем:

• отдельные модули (правила) вызывают не по имени, а по описанию ситуации;

• способ взаимосвязи модулей (правил) формируется в процессе решения задачи, так как выбор очередного модуля (правила) зависит от текущей ситуации и не может быть сформирован заранее.

Основным механизмом, обеспечивающим разнообразное управление в рамках общей схемы работы интерпретатора, являются *стратегии.* Стратегии можно рассматривать по крайней мере с трех точек зрения: как средство разрешения конфликтов; как способ представления метазнаний и как средство повышения эффективности метода, встроенного в механизм вывода (интерпретатор). Первая точка зрения важна в тех случаях, когда размер конфликтного множества достаточно велик. При этом во избежание слепого исчерпывающего поиска необходимо использовать знания, направляющие процесс выбора текущего модуля. Ясно, что качество сделанного выбора будет сильно влиять на "интеллектуальность" системы. Во многих системах знания, на основании которых осуществляется указанный выбор, не являются явными и не выражены в достаточно общей форме. В первом приближении стратегией можно называть знания

о том, какой модуль (правило) следует выбрать при наличии нескольких модулей, пригодных к работе.

В более общих терминах стратегии можно рассматривать как метазнания о том, как и когда использовать различные источники знаний объектного уровня (т.е. знания о предметной области). Метазнания могут выражать знания о разбиении задачи на подзадачи, знания о кооперации источников знания, знания о наличии различных стратегий поиска. Необходимо подчеркнуть важность явного задания подобных знаний, т.е. задания знаний в такой форме, которую система может анализировать. Явное задание знаний обеспечивает гибкое поведение системы.

Кроме того, стратегии традиционно рассматривают как средство повышения эффективности некоторого общего метода. Таковы, например, стратегии, ограничивающие принцип резолюции в логических системах. В более общих терминах стратегии можно рассматривать как любые знания о том, как, когда и какие модули (правила) использовать, т.е. стратегии можно рассматривать не только как средство оптимизации некоторого метода, но и как средство для выбора (и даже для определения) метода.

Приведем классификацию стратегий, используемых в экспертных системах, по следующим параметрам [1]: общность; явное или неявное задание; содержание.

По принципам общности стратегии можно классифицировать следующим образом: стратегии, не зависящие от способа представления знаний; стратегии, не зависящие от предметной области; стратегии, учитывающие специфику предметной области, и стратегии, учитывающие специфику цели. Примерами общих стратегий, не зависящих от способа представления, являются стратегии поиска от целей или от данных (см. п. 6.3.1). Примером стратегий, не зависящих от предметной области, является стратегия множества поддержки, используемая при доказательстве теорем в исчислении предикатов [2]. Отметим, что данная стратегия зависит от выбранного способа представления, так как она применима только в контексте доказательства теорем и исчисления предикатов. Однако эта стратегия применима к любой области, где можно использовать технику доказательства теорем.

Стратегии, учитывающие специфику области, имеют более ограниченное применение, чем стратегии первых двух типов, однако именно они позволяют использовать знания о конкретной области для получения качественных и эффективных решений. Стратегии, учитывающие специфику цели, позволяют управлять процессом решения в зависимости от текущих задач системы.

Стратегии можно разделить на заданные явно и неявно. Стратегия задана явно, если она может быть идентифицирована как отдельная сущность системы, т.е. смена стратегии проходит для системы безболезненно. Неявные стратегии иногда подразделяют на концептуально неявные и неявные по выполнению. Концептуально неявными являются стратегии, механизм которых рассредоточен по системе. Например, их результат проявляется как побочный эффект выполнения какой-либо другой части системы. Стратегии, не явные по выполнению, - это те стратегии, основные идеи которых выражены явно, но при этом из-за специфики выполнения имеется некоторая вложенность стратегий в другие конструкции системы.

Не касаясь специфики предметной области, содержание стратегии можно охарактеризовать по крайней мере тремя независимыми параметрами: масштабом; составом знаний, используемых стратегией, и полезностью стратегии.

Параметр "масштаб" делит все стратегии на *локальные* и *глобальные.* Локальными называют те стратегии, которые определяют поведение интерпретатора в текущем цикле его работы, в отличие от глобальных стратегий, которые определяют некоторую линию рассуждений интерпретатора, т.е. последовательность выполняемых (предполагаемых к выполнению) циклов (шагов).

Параметр "состав используемых знаний" делит знания на два класса: знания о текущем цикле работы интерпретатора и знания об истории работы интерпретатора. Заметим, что теоретически для работы экспертной системы достаточно только текущих знаний, однако на практике в целях повышения эффективности систем используют и знания об истории работы.

Параметр "полезность" подразделяется на индивидуальную и сравнительную полезность. Индивидуальная полезность характеризует некоторое знание само по себе вне сравнения его с другими знаниями. Сравнительная полезность характеризует ценность некоторого знания по сравнению с другим знанием.

Контрольные вопросы:

1. Сформулируйте основные задачи механизма вывода экспертной системы.
2. Укажите назначение и главные функции четырех этапов работы интерпретатора.
3. Назовите основные различия между подходом, использующим управляемые образцами правила, и подходом, использующим управляемые образцами модули.
4. Дайте определение стратегии управления в экспертных системах и приведите классификацию стратегий.

ЛИТЕРАТУРА

1. *Попов Э.В.* **Экспертные системы. Решение неформализованных задач в диалоге с ЭВМ.** - М: Наука, 1987.
2. *Попов Э.В., Фридман Г. Р.* **Алгоритмические основы интеллектуальных роботов и искусственного интеллекта.** - М.: Наука, 1976. - 455 с.
3. *Минский М.* **На пути к созданию искусственного разума**//Вычислительные машины и мышление. - М.: Мир, 1967. - 552 с.
4. *Нильсон Н.* **Искусственный интеллект. Методы поиска решений. -** М.: Мир, 1973.

Лекция 13: Способы решения задач ДЭС

План:

1. Методы поиска решений в экспертных системах

2. Поиск решений в одном пространстве
3. Поиск в иерархии пространств
4. Поиск в альтернативных пространствах
5. Поиск с использованием нескольких моделей
6. Выбор метода решения задач

1. Методы поиска решений в экспертных системах

Методы решения задач, основанные на сведении их к поиску, зависят от особенностей предметной области, в которой решается задача, и от требований, предъявляемых пользователем к решению. Особенности предметной области с точки зрения методов решения можно характеризовать следующими параметрами:

• размер, определяющий объем пространства, в котором предстоит искать решение;

• изменяемость области, характеризует степень изменяемости области во времени и пространстве (здесь будем выделять статические и динамические области);

• полнота модели, описывающей область, характеризует адекватность модели, используемой для описания данной области. Обычно если модель не полна, то для описания области используют несколько моделей, дополняющих друг друга за счет отражения различных свойств предметной области;

• определенность данных о решаемой задаче, характеризует степень точности (ошибочности) и полноты (неполноты) данных. Точность (ошибочность) является показателем того, что предметная область с точки зрения решаемых задач описана точными или неточными данными; под полнотой (неполнотой) данных понимается достаточность (недостаточность) входных данных для однозначного решения задачи.

Требования пользователя к результату задачи, решаемой с помощью поиска, можно характеризовать количеством решений и свойствами результата и (или) способом его получения. Параметр "количество решений" может принимать следующие основные значения: одно решение, несколько решений, все решения. Параметр "свойства" задает ограничения, которым должен удовлетворять полученный результат или способ его получения. Так, например, для системы, выдающей рекомендации по лечению больных, пользователь может указать требование не использовать некоторое лекарство (в связи с его отсутствием или в связи с тем, что оно противопоказано данному пациенту). Параметр "свойства" может определять и такие особенности, как время решения ("не более чем", "диапазон времени" и т.п.), объем памяти, используемой для получения результата, указание об обязательности (невозможности) использования каких-либо знаний (данных) и т.п.

Итак, сложность задачи, определяемая вышеприведенным набором параметров, варьируется от простых задач малой размерности с неизменяемыми определенными данными и отсутствием ограничений на результат и способ его получения до сложных задач большой размерности с изменяемыми, ошибочными и неполными данными и произвольными ограничениями на результат и способ его получения. Из общих соображений ясно, что каким-либо одним методом нельзя решить все задачи. Обычно одни методы превосходят другие только по некоторым из перечисленных параметров.

Рассмотренные ниже методы могут работать в статических и динамических проблемных средах. Для того чтобы они работали в условиях динамики, необходимо учитывать время жизни значений переменных, источник данных для переменных, а также обеспечивать возможность хранения истории значений переменных, моделирования внешнего окружения и оперирования временными категориями в правилах (см. гл.9).

Существующие методы решения задач, используемые в экспертных системах, можно классифицировать следующим образом:

• методы поиска в одном пространстве - методы, предназначенные для использования в следующих условиях: области небольшой размерности, полнота модели, точные и полные данные;

• методы поиска в иерархических пространствах - методы, предназначенные для работы в областях большой размерности;

• методы поиска при неточных и неполных данных ;

• методы поиска, использующие несколько моделей, предназначенные для работы с областями, для адекватного описания которых одной модели недостаточно.

Предполагается, что перечисленные методы при необходимости должны объединяться для того, чтобы позволить решать задачи, сложность которых возрастает одновременно по нескольким параметрам.

2. Поиск решений в одном пространстве

Методы поиска решений в одном пространстве обычно делятся на поиск в пространстве состояний, поиск методом редукции, эвристический поиск и поиск методом "генерация-проверка" [1].

Поиск в пространстве состояний

Задача поиска в пространстве состояний обычно формулируется в теоретико-графовой интерпретации [1,2].

Пусть задана тройка (S_0, F, S_T), где S_0 - множество начальных состояний (условия задачи), F - множество операторов задачи, отображающих одни состояния в другие; S_T - множество конечных (целевых) состояний (решений задачи).

В этой постановке решить задачу - значит определить такую последовательность операторов, которая преобразует начальные состояния в конечные. Процесс решения можно представить в виде графа G = (X, Y), где X = {x_0, x_1,...} - множество (в общем случае бесконечное) вершин графа, каждая из которых отождествляется с одним из состояний, а Y - множество, содержащее пары вершин (x_i, x_j), (x_i, x_j) $\in X$. Если каждая пара (x_i, x_j) неупорядочена, то ее *называют ребром,* а граф - *неориентированным.* Если для каждой пары (x_i, x_j) задан порядок (направление), то пару (x_i, x_j) называют *дугой* (ориентированным ребром), а граф называют *ориентированным (направленным).* Вершины пары (x_i, x_j) называют *концевыми точками ребра* (дуги).

Поиск в пространстве состояний естественно представить в виде ориентированного графа. Наличие пары (x_i, x_j) свидетельствует о существовании некотррого оператора $f\ (f \in F)$, преобразующего состояние, соответствующее вершине x_i, в состояние x_j. С точки зрения поиска в пространстве состояний для некоторой вершины x_i уместно выделить множество всех направленных пар (x_i, x_j)$\in Y$, т.е. множество дуг, исходящих из вершины x_i (родительской вершины), и множество вершин (называемых дочерними вершинами), в которые эти дуги приводят. Множество дуг, исходящих из вершины x_i, соответствует множеству операторов, которые могут быть применены к состоянию, соответствующему вершине x_i.

В множестве вершин X выделяют подмножество вершин $X_0 \subseteq X$ соответствующее множеству начальных состояний (S_0), и подмножество вершин $X_T \subseteq X$, соответствующее множеству конечных (целевых) состояний (S_T). Множество X_T может быть задано как явно, так и неявно, т.е. через свойства, которыми должны обладать целевые состояния.

Отметим, что граф G может быть задан явно и неявно. Неявное задание графа G состоит в определении множества $X_0 \subseteq X$ (соответствующего множеству начальных состояний) и множества операторов, которые, будучи применимы к некоторой вершине графа, дают все ее дочерние вершины.

Итак, граф G задает пространство состояний, т.е. пространство, в котором осуществляется поиск решения. Построение пространства осуществляется с помощью следующего процесса. Берется некая вершина $x_0 \in X_0$, к ней применяются все возможные операторы, порождающие все дочерние вершины. Этот процесс называют процессом *раскрытия вершин.* Если получена целевая вершина, то она не раскрывается. Процесс построения пространства состояний заканчивается, когда все нераскрытые вершины являются целевыми, или терминальными (т.е. вершинами, к которым нельзя применить никаких операторов). В связи с тем, что пространство состояний может содержать бесконечное количество вершин, на практике процесс порождения пространства ограничивают либо временем, либо объемом памяти.

В практических приложениях часто требуется обеспечить полноту поиска, т.е. организовать поиск так, чтобы все целевые вершины были найдены, если они существуют. Надежным способом обеспечения полноты является полный перебор всех вершин. Для задания процесса перебора необходимо определить порядок, в котором будут перебираться вершины графа. Обычно выделяют два основных способа поиска: *поиск в глубину* и *поиск в ширину* При поиске в глубину сначала раскрывается та вершина, которая была построена самой последней. Глубина вершины в графе определяется так:

глубина начальной вершины равна нулю;

глубина неначальной вершины равна единице плюс глубина наиболее близкой родительской вершины.

При практической реализации поиск в глубину в некотором направлении завершается в следующих случаях:

при достижении целевой вершины;

при достижении терминальной вершины;

при построении в ходе поиска вершины, глубина которой превышает некоторую граничную глубину.

При поиске в ширину вершины раскрываются в том же порядке, в котором они порождаются.

Если в пространство состояний ввести операторы, переводящие состояние S_i в предшествующее состояние S_{i-1}, то поиск можно осуществлять не только в направлении от начального состояния к целевому, но и в обратном направлении. Поиск первого типа называют *поиском от данных,* или *прямым поиском,* а поиск второго типа - *поиском от цели,* или *обратным поиском.* Можно организовать поиск в двух направлениях одновременно. Такой поиск называют *двунаправленным* (или бинаправленным).

На рис. 6.3 приведен пример решения задачи поиском в глубину (рис. 6.3, а) и в ширину (рис. 6.3, б). Вершины пронумерованы в том порядке, в котором они раскрываются (а не порождаются), целевые вершины помечены черными квадратами, а терминальные - белыми квадратами. При использовании каждого из способов могут быть найдены все решения. При переборе всего пространства оба метода будут анализировать одинаковое количество вершин, однако метод поиска в ширину будет требовать существенно больше памяти, так как он запоминает все пути поиска (а не один, как при поиске в глубину).

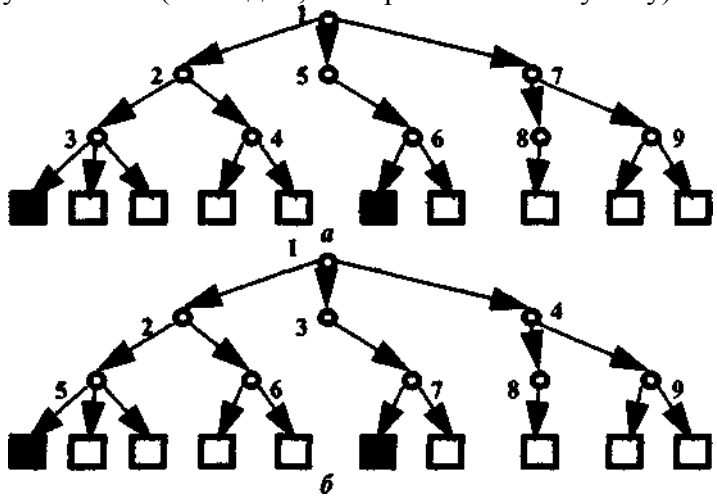


Рис.6.3. Пространство состояний, построенное поиском в глубину (а) и поиском в ширину (б)

Поиск методом редукции

При поиске методом редукции решение задачи сводится к решению совокупности образующих ее подзадач [1]. Этот процесс повторяется для каждой подзадачи до тех пор, пока каждая из полученного набора подзадач, образующих решение исходной задачи, не будет иметь очевидное решение. Подзадача считается очевидной, если ее решение общеизвестно или получено ранее. Процесс решения задачи разбиением ее на подзадачи можно представить в виде специального направленного графа G, называемого И/ИЛИ-графом. Каждой вершине этого графа ставится в соответствие описание некоторой задачи (подзадачи). В графе выделяют два

типа вершин: конъюнктивные вершины и дизъюнктивные вершины. *Конъюнктивные вершины,* или вершины типа "И", вместе со своими дочерними вершинами интерпретируются так: решение задачи сводится к решению всех ее подзадач, соответствующих дочерним вершинам конъюнктивной вершины. *Дизъюнктивные вершины,* или вершины типа "ИЛИ", вместе со своими дочерними вершинами интерпретируются так: решение задачи сводится к решению любой из ее подзадач, соответствующих дочерним вершинам дизъюнктивной вершины. Отметим, что некоторые авторы [4, 7] определяют вершины И и ИЛИ иначе.

Во множестве вершин И/ИЛИ-графа выделяют подмножество начальных вершин, т.е. задач, которые следует решить, и подмножество конечных (целевых) вершин, т.е. заведомо разрешимых задач. Решение задачи при поиске методом редукции (при поиске в И/ИЛИ-графе) сводится к нахождению в И/ИЛИ-графе решающего графа, определение которого будет дано ниже. Заметим, что метод сведения задач к подзадачам является в некотором роде обобщением подхода с использованием пространства состояний. Действительно, перебор в пространстве состояний можно рассматривать как тривиальный случай сведения задачи всегда к одной подзадаче.

Графически для различения дизъюнктивной и конъюнктивной вершин дуги, исходящие из конъюнктивной вершины, соединяются дужкой при вершине. Пример графического представления разбиения задачи на подзадачи приведен на рис. 6.4. Здесь S_0 - исходная задача, для решения которой требуется решить подзадачу S_3 или подзадачи S_1 и S_2. Решение задачи S_1 сводится к решению либо подзадачи S_4, либо подзадачи S_5. Решение подзадачи S_3 сводится к решению подзадач S_6 и S_7. Решение задач S_2, S_5, S_7 предполагается известным, решение задач S_4 и S_6 неизвестно. В приведенном примере задача S_0 может быть решена либо путем решения задачи S_3, либо путем решения задач S_1 и S_2. В связи с тем, что в И/ИЛИ-графе каждая вершина относится только к одному типу (либо И, либо ИЛИ), то для записи графа, изображенного на рис. 6.4 в виде И/ИЛИ-графа, надо ввести дополнительную вершину (вершина R_1 на рис. 6.5). На рис. 6.5 двойными линиями выделен

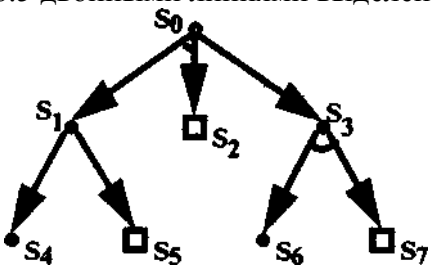


Рис. 6.4. Графическое представление процесса разбиения задачи на подзадачи

решающий граф задачи S_0, а конечные вершины обозначены зачерненными квадратами.

Цель процесса поиска в И/ИЛИ-графе - показать, что начальная вершина разрешима, т.е. для этой вершины существует решающий граф. Определение *разрешимой вершины* в И/ИЛИ-графе можно сформулировать рекурсивно следующим образом:

1. Конечные (целевые) вершины разрешимы, так как их решение известно по исходному предположению.

2. Вершина ИЛИ разрешима тогда и только тогда, когда разрешима по крайней мере одна из ее дочерних вершин.

3. Вершина И разрешима тогда и только тогда, когда разрешима каждая из ее дочерних вершин.

Итак, *решающий граф* определяется как подграф из разрешимых вершин, который показывает, что начальная вершина разрешима (в соответствии с приведенным выше определением). На рис. 6.5 разрешимые вершины зачернены, а неразрешимые оставлены белыми.

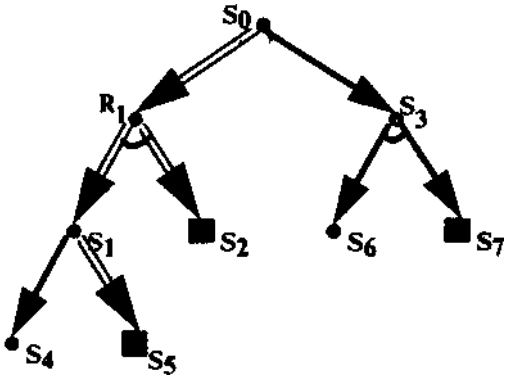


Рис. 6.5. Пример И/ИЛИ-графа

Для графа И/ИЛИ, так же как для поиска в пространстве состояний, можно определить поиск в глубину и поиск в ширину как в прямом, так и в обратном направлении. На рис. 6.6 приведен пример поиска в ширину (рис. 6.6, *а)* и поиска в глубину (рис. 6.6, *б).* На рисунке вершины пронумерованы в том порядке, в котором они раскрывались; конечные вершины обозначены квадратами, разрешимые вершины зачернены, дуги решающего графа выделены двойными линиями.

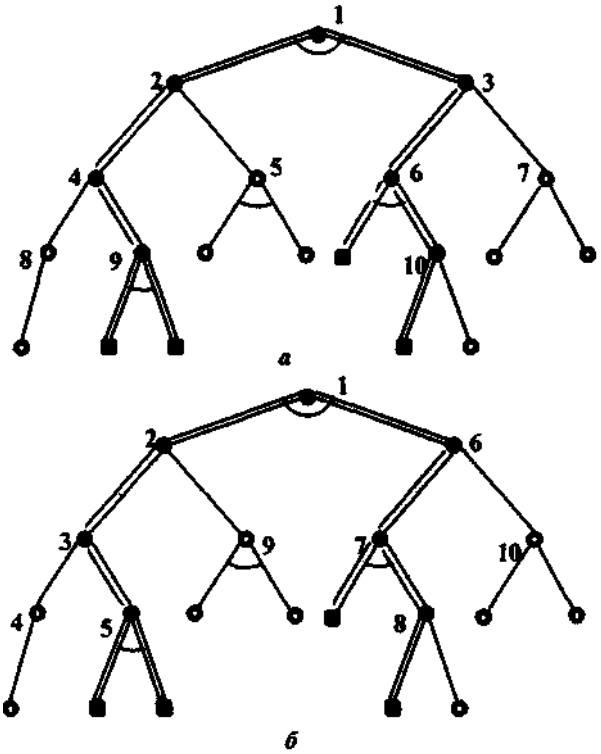


Рис.6.6. Пример разбиения задач на подзадачи при поиске в ширину (а) и при поиске в глубину (б)

Эвристический поиск

Методы поиска в глубину и ширину называют *слепым поиском,* поскольку в этих методах порядок раскрытия вершин предопределен и никак не зависит от расположения цели. При увеличении пространства поиска методы слепого поиска требуют чрезмерных затрат времени и (или) памяти. Стремление сократить время поиска привело к созданию эвристических методов поиска, т.е. методов, использующих некоторую информацию о предметной области для рассмотрения не всего пространства поиска, а таких путей в нем, которые с наибольшей вероятностью приводят к цели. Один способ сокращения перебора состоит в выборе более "информированного" оператора, который не строит так много вершин, не относящихся к делу. Другой способ состоит в использовании эвристической информации для определения на каждом шаге дальнейшего направления перебора. Для этого необходимо ввести меру "перспективности" вершины в виде некоторой оценочной функции. В некоторых случаях удается ввести такую оценочную функцию, что она, сокращая перебор, не теряет свойства полноты. Чаще же используемые эвристики, существенно сокращая перебор, влекут за собой потерю свойства полноты.

Как правило, оценочные функции пытаются количественно оценить расстояние от текущей вершины до конечной. Из двух вершин при одинаковой глубине перспективней та, от которой меньше расстояние до цели. Для многих приложений, в частности для экспертных систем, применение количественных оценок не позволяет эффективно направлять процесс поиска.

Поиск методом "генерация-проверка"

Процесс поиска может быть сформулирован в терминах "генерация-проверка". Действительно, пространство поиска (пространство состояний или И/ИЛИ-граф), как правило, явно не задано. Поэтому для осуществления процесса поиска необходимо генерировать очередное возможное решение (состояние или подзадачу) и проверить, не является ли оно результирующим. Разумно потребовать, чтобы генератор удовлетворял требованиям полноты и неизбыточности. Говорят, что *генератор* является *полным,* если он обеспечивает генерацию всех возможных решений. *Генератор* является *неизбыточным,* если он генерирует каждое решение только один раз. Обеспечение свойства неизбыточности является важным, но трудновыполнимым, так как в соответствии с этим требованием не допускается генерация не только тождественных, но и синонимичных решений. Например, если задача генератора - синтезировать все фразы русского языка, то весьма трудно (если вообще возможно) сделать такой генератор неизбыточным.

При генерации текущего возможного решения (состояния или подзадачи) возникает проблема распределения знаний между генератором и устройством проверки. При слепом и эвристическом поиске генератор имеет минимальные знания об области, достаточные для генерации всех возможных решений (состояний или подзадач), а устройство проверки определяет, не является ли очередное решение целевым. В принципе некоторые знания можно перенести из устройства проверки в генератор, чтобы он не генерировал решения, которые заведомо не могут привести к успеху. Увеличение знаний генератора об области приводит к сокращению пространства, в котором осуществляется поиск. Однако при этом повышаются затраты на генерацию каждого очередного состояния (подзадачи).

Можно выделить важную форму метода "генерация-проверка", называемую "иерархическая генерация-проверка". В этом случае на верхнем уровне генератор вырабатывает не полное, а частично определенное решение (будем для краткости называть такие решения частичными). Каждое *частичное решение* описывает не все состояние, а только его некоторую часть, определяя таким образом класс возможных состояний. Идея состоит в том, что устройство проверки может уже по виду частичного решения определить, что оно (а следовательно, и все полные решения, которые могут быть получены из него) не ведет к успеху. Если же проверка не отвергает частичное решение, то на следующем уровне генератор продолжает вырабатывать из данного частичного решения все полные решения, а устройство проверки определяет, являются ли они целевыми.

3. Поиск в иерархии пространств

Методы поиска в одном пространстве не позволяют решать сложные задачи, так как с увеличением размера пространства время поиска экспоненциально растет. При большом размере пространства поиска можно попробовать разбить общее пространство на подпространства и осуществлять поиск сначала в них. Можно сказать, что в данном случае пространство поиска представлено иерархией пространств. Важность иерархических методов при работе с большими пространствами понята давно. Еще в 1963 г. М.Минский писал, что введение "островков планирования" уменьшает время поиска по экспоненте: "В графе с 10 ребрами, исходящими из каждой вершины, 20-шаговый поиск может потребовать 10^{20} попыток, что нереально реализовать, в то время как введение четырех лемм или последовательных подцелей может уменьшить поиск до 5×10^4 попыток, которые машина может выполнить. Поэтому имеет смысл приложить даже огромные усилия, чтобы выявить такие "островки" при решении сложных задач" [3]. Идею М.Минского о иерархии пространств можно развить, допустив в иерархии не только конкретные, но и абстрактные пространства, т.е. пространства которые имеют описание только наиболее важных сущностей. В качестве классического примера использования абстрактных пространств можно привести задачу определения кратчайшего пути на карте. Пусть требуется переехать из центра города *A* в центр города *B*. Если осуществлять поиск требуемого пути на детальной карте, содержащей все улицы во всех городах, встретившихся по дороге, то задача может стать практически неразрешимой. При определении пути из города *A* в город *B* целесооб-

разно спланировать маршрут по крупномасштабной карте (т.е. осуществить поиск в абстрактном пространстве), а затем по детальной карте спланировать выезд из города *A* и въезд в город *B*. В данном разделе будут рассмотрены методы, использующие общую идею иерархии пространств, но отличающиеся природой пространств.

Методы поиска решения в иерархических пространствах обычно делятся на поиск в факторизованном пространстве, поиск в фиксированном и изменяющемся множестве пространств [1].

Поиск в факторизованном пространстве

Во многих приложениях требуется найти все решения. Примерами таких областей являются интерпретация данных, постановка диагноза и др. Действительно, в случае постановки диагноза нас интересуют все, а не некоторые болезни пациента. Однако пространство поиска в практических приложениях бывает столь велико, что не позволяет применить слепые методы поиска. Применение эвристических методов в данном случае, как правило, также исключено, так как они не обеспечивают получение всех возможных решений. Если пространство поиска удается факторизовать, то поиск даже в очень большом пространстве можно организовать эффективно. *Пространство* называется *факторизованным,* если оно разбивается на непересекающиеся подпространства (классы) *частичными (неполными)* решениями. Причем по виду частичного решения можно определить, что оно не приведет к успеху, т.е. что все полные решения, образованные из него, не приведут к целевым решениям. Поиск в факторизованном пространстве осуществляется на основе метода "иерархическая генерация-проверка" (см. выше). Генератор вырабатывает текущее частичное решение, затем проверяется, может ли это решение привести к успеху. Если текущее частичное решение отвергается, то из рассмотрения без генерации и проверки устраняются все полные решения этого класса. Если текущее частичное решение не отвергается, то генератор вырабатывает на его основе все полные решения, а устройство проверки определяет, являются ли эти решения целевыми.

Поиск в фиксированном множестве пространств

Применение метода факторизации пространства ограничено тем, что для ряда областей не удается по частичному решению сделать заключение о его непригодности. Примерами таких областей являются задачи планирования и конструирования. Действительно, как правило, по фрагменту плана или конструкции нельзя сказать, что этот фрагмент не может являться частью полного решения. В этих случаях могут быть применены методы поиска, использующие идею абстрактного пространства. Методы различаются предположениями о природе этого пространства. Абстракция должна подчеркнуть важные особенности рассматриваемой задачи, позволить разбить задачу на более простые подзадачи и определить последовательность подзадач (план решения), приводящую к решению основной задачи. В простейшем случае пространство поиска разбивается на фиксированную последовательность подзадач (подпространств), с помощью которых можно решить любую исходную задачу.

Подобный метод поиска использован, например, в экспертной системе R1 [1]. На основании заказа покупателя на требуемую ему кон-фигурацию системы VAX система R1 определяет, не содержит ли заказ несовместимых компонентов, выявляет недостающие компоненты и строит диаграммы, изображающие пространственные взаимосвязи компонентов VAX. Система R1 разбивает общую задачу на шесть подзадач Порядок, в котором вызываются эти задачи, зависит от заказанной конфигурации. Действия, выполняемые каждой подзадачей, зависят от комбинации заказанных компонентов и способа их взаимосвязи. В системе каждой подзадаче соответствует свой набор правил, т е. каждая подзадача решается в своем подпространстве. Поиск в R1 осуществляется с помощью безвозвратной стратегии поиска, т е. без использования процедуры бэктрекинга [7] Этот механизм восстанавливает состояние, непосредственно предшествующее текущему, и затем выбирает очередную альтернативу.

Поиск в изменяющемся множестве иерархических пространств

В ряде приложений не удается все решаемые задачи свести к фиксированному набору подзадач. Примерами таких приложений являются задачи планирования перемещений в пространстве. План решения задачи в данном случае должен иметь переменную структуру и не может быть сведен к фиксированному набору подзадач. Для решения подобных задач может быть использован метод нисходящего уточнения (top-down refinement). Для того чтобы упростить процесс решения некоторой задачи в сложном пространстве, целесообразно получить

обобщенное пространство (пространство меньшей размерности) и попробовать получить решение в этом пространстве. Указанный прием можно повторять многократно. При этом полный процесс решения задачи можно представить как нисходящее движение в иерархии пространств от наиболее абстрактного к конкретному, в котором получается окончательное решение. Существенной характеристикой такого процесса являются поиск решения задачи в абстрактном пространстве, преобразование этого решения в решение более низкого уровня и т.д. Причем на каждом уровне вырабатывается окончательное решение и только затем осуществляется переход на следующий, более конкретный уровень. Внутри каждого уровня подзадачи рассматриваются как независимые, что создает частичное упорядочение абстрактных состоянии. Формирование более абстрактного пространства осуществляется путем игнорирования части описаний менее абстрактного пространства (на первом шаге - конкретного пространства). Игнорирование описаний осуществляется на основе ранжирования описаний по степени важности. Часто ранжирование осуществляется на основе учета степени неизменности фактов (наиболее абстрактны те описания, которые не могут изменяться). При этом абстрактные пространства, с одной стороны, должны для упрощения решения задачи обеспечивать значительное упрощение исходного пространства, а с другой стороны, должны быть подобны друг другу и конкретному пространству, чтобы процесс нисходящего переноса решения из более абстрактных пространств в менее абстрактные не требовал больших вычислительных затрат.

Система ABSTRIPS [1] является одним из первых примеров использования метода нисходящего уточнения. ABSTRIPS является программой, составляющей план перемещения роботом объектов (ящиков) между комнатами. Получив задачу, система составляет последовательность действий робота, которая решает эту задачу. Робот действует в мире, содержащем описание комнат, расположение дверей в комнатах, состояние дверей (открыты, закрыты), местонахождение объектов, местонахождение робота. Робот умеет выполнять ряд действий: перемещаться по комнате, переходить из одной комнаты в другую, открывать дверь, толкать объекты и т.п. Возможным действиям робота соответствуют определенные операторы. Каждый оператор представлен наименованием со списком параметров, условиями применимости оператора и преобразованиями, которые он совершает, изменяя пространство. Пространство поиска (конкретное пространство), в котором ищется решение, состоит из возможных состояний мира, получаемых преобразованием исходного состояния путем применения к нему всех возможных операторов. Для того чтобы упростить процесс решения задачи, ABSTRIPS формирует из конкретного пространства иерархию абстрактных пространств. Абстрактные пространства образуются путем упрощения условий применимости операторов, т.е. чем выше уровень абстракции, тем меньше литер (слов) содержит условие применимости каждого оператора. Такой подход позволяет при формировании абстрактного пространства не вычеркивать несущественные детали из описания мира и операторов, а просто не учитывать их при решении. Уровень детальности указывается с помощью веса, связанного с каждой литерой в условии применимости оператора. Основанием для назначения веса служат следующие эвристические соображения. Существование предметов и их свойств (т.е. наличие комнат, дверей, ящиков) является с точки зрения построения плана более важным фактом, чем положение предметов, которые могут передвигаться роботом, и тем более чем положение робота. Поэтому только эти наиболее важные факты должны учитываться в абстрактном пространстве. После построения приблизительного плана его детали уточняются в более конкретных пространствах.

Завершая описание метода нисходящего уточнения, отметим, что абстрактные пространства здесь создаются индивидуально в соответствии с решаемой задачей. Необходимо отметить, что метод базируется на следующих предположениях:

• возможно осуществить частичное упорядочение понятий области, приемлемое для всех решаемых задач;

• решения, принимаемые на верхних уровнях, нет необходимости отменять на более нижних.

Использование ограничений при поиске решения

Ограничения можно рассматривать как способ частичного описания некоторых сущностей. Ограничения могут быть заданы как в числовой, так и в символьной форме. Примером задания ограничения в числовой форме является любая формула, которая накладывает ограничения на соотношение входящих в формулу переменных. Примером задания ограничения в сим-

вольной форме является модель управления любого глагола, задающая семантические категории.

Ограничения могут быть использованы для представления целей в методах поиска в иерархических пространствах. Например, при конструировании топологии электрической схемы инвертор на верхнем уровне абстракции может быть описан как дискретное переключательное устройство с одним входом и несколькими выходами. На этом уровне описания игнорируется такая информация, как геометрия инвертора, источник питания и земля. На более низком уровне абстракции инвертор может быть описан с учетом его геометрии. На этом же уровне могут быть указаны два ограничения, определяющие, какая часть инвертора должна быть связана с питанием, а какая - с землей. Использование ограничений позволяет отложить решение вопроса о том, как именно выглядит маршрут, соединяющий части инвертора с питанием и землей. Эти ограничения могут быть учтены при конструировании других частей схемы. Если ограничения не могут быть учтены, то построенная схема инвертора должна быть пересмотрена.

Использование ограничений вместо получения конкретного решения дает возможность отложить принятие решения. Откладывание решения может быть вызвано рядом причин: нет достаточной информации для того, чтобы определить местонахождение питания и земли; другие соображения при конструировании схемы могут оказаться более важными, чем рассматриваемый инвертор.

Вторая из указанных причин иллюстрирует важный феномен процесса решения задач - взаимодействие подзадач. Если для решения подзадач требуется их незначительная координация, то говорят, что подзадачи почти независимы. Обычно такие подзадачи имеют более одного решения, если при получении решения учитываются только локальные ограничения, т.е. ограничения, вытекающие из самой подзадачи, а не из других подзадач, от которых данная подзадача почти независима. Если получать решение таких подзадач как независимых, то часто при объединении подзадач возникают несоответствия. Введение ограничений позволяет избежать преждевременного получения решений, учитывающих не все, а только локальные ограничения. Использование ограничений позволяет применять принцип наименьших свершений (см. ниже). Этот принцип позволяет переключать внимание с одной подзадачи на другую и избегать преждевременных решений.

Принцип наименьших свершений

Основной недостаток метода нисходящего уточнения состоит в том, что он не имеет обратной связи. Метод предполагает, что одни и те же решения должны приниматься в одинаковых ситуациях при решении любой задачи. При решении ряда задач детализация решения, полученного на абстрактном уровне, оказывается невозможной, так как при построении абстрактного плана были опущены детали, препятствующие его уточнению, т.е. требуется пересмотр абстрактного плана (решения). В подобных ситуациях целесообразно применение *принципа наименьших свершений.* В соответствии с данным принципом решение не строится сразу до конца на верхних уровнях абстракции. Частичное решение детализируется постепенно, по мере появления информации, подтверждающей возможность решения и вынуждающей принять решение. Рассуждение, основанное на использовании принципа наименьших свершений, требует, чтобы система была в состоянии совершить следующие действия:

• определить, когда накопилось достаточно информации для принятия решения;

• приостанавливать работу над некоторой подзадачей, когда для решения нет достаточной информации;

• переходить с одной подзадачи на другую, возобновляя выполнение приостановленной подзадачи при появлении недостающей информации;

• объединять информацию, полученную различными подзадачами.

Принцип наименьших свершений впервые был использован экспертной системой MOLGEN [I], предназначенной для планирования экспериментов по молекулярной генетике. MOLGEN представляет взаимодействие между подзадачами в виде ограничений. Для рассуждений об ограничениях используются операторы метауровня (в противовес операторам предметной области). Система чередует использование принципа наименьших свершений и использование эвристических стратегий. При использовании принципа наименьших свершений выбор осуществляется только тогда, когда ограничения определяют достаточно узкий набор альтернатив.

В противном случае процесс решения задачи приостанавливается (задача переходит в состояние "ожидание ограничений"), и осуществляется переход к другой подзадаче.

Распространение ограничений - механизм для передачи информации между подзадачами. Ограничения, выставленные одной подзадачей, могут существенно сузить набор альтернатив другой подзадачи. MOLGEN строит планы в ответ на распространение ограничений.

Чередование в MOLGEN подхода наименьших свершений и эвристических стратегий иллюстрирует ограниченность принципа наименьших свершений. В связи с тем, что любой решатель имеет неполные знания о проблеме, в процессе использования принципа наименьших свершений может возникнуть следующая ситуация. Необходимо делать выбор, но нет оснований предпочесть одну альтернативу другим. Эта ситуация приводит к остановке процесса и называется тупиком, потому что все подзадачи перешли в состояние "ожидание ограничений". Когда MOLGEN распознает эту ситуацию, она переключается на эвристическую стратегию и делает предположение (угадывание). Во многих случаях угадывание позволяет продолжить процесс поиска решения и довести его до конечного результата. В других случаях угадывание приводит к конфликтам, требующим новых попыток по угадыванию. Конфликт может возникнуть и при работе по принципу наименьших свершений, а именно в том случае, когда цели принципиально недостижимы.

Итак, принцип наименьших свершений координирует процесс поиска решения с наличием необходимой информации и в соответствии с доступной информацией перемещает фокус активности по решению задачи от одной подзадачи к другой. Данный подход непригоден, когда существует много возможностей, но нет надежных оснований для выбора решения. В этих случаях необходимо использовать некоторые формы правдоподобных рассуждений или переходить на использование другой модели (см. ниже).

Метапространства в иерархии пространств

При решении любой задачи многократно возникает вопрос: "Что делать на следующем шаге?". В простейшем случае решение предопределено методом поиска решения. При поиске в абстрактных и конкретных пространствах на каждом шаге решался вопрос о том, какой из операторов, существующих в проблемной области, применить к текущему состоянию проблемной области. Вопрос о том, как решающая программа это сделает, не обсуждался. Можно оказывать не явное, а косвенное влияние на определение того, "что делается на следующем шаге в проблемной области" путем выбора того или иного метода, известного решателю. Подобный подход требует явного разграничения знаний о процессе решения и знаний о проблемной области. Для этого необходимы знания на метауровне. Решатель в метапространстве содержит явное описание процесса организации поиска, т.е. описание состояний, операторов, условий применимости операторов, описание доступных методов (стратегий) поиска и способов их взаимодействия. Получить решение в метапространстве - это значит определить, какой метод (программа) будет применен на следующем шаге, т.е. составить метаплан решения задачи. Заметим, что метаплан в отличие от абстрактного плана выражается не в терминах операторов проблемной области, а в терминах методов (программ), известных решателю. Не существует причин ограничивать метазнания одним уровнем.

По аналогии с факторизацией абстрактного пространства можно говорить о разбиении метапространства на метазадачи (методы, программы). Разбиение на метазадачи является полезным методом организации знаний в экспертных системах, однако в настоящее время еще далеко до общего теоретического осмысления данного вопроса.

Завершая описания методов поиска в иерархии пространств, подчеркнем, что в рассмотренных подходах используются пространства трех видов: конкретные, абстрактные и метапространства, и все они могут использоваться в одной системе.

4. Поиск в альтернативных пространствах

Рассмотренные выше методы поиска исходят из молчаливой предпосылки, что знания о предметной области и данные о решаемой задаче являются точными и полными и для них справедливо следующее:

• все утверждения, описывающие состояние, являются истинными;

• применение оператора к некоторому состоянию формирует некоторое новое состояние, описание которого состоит только из истинных фактов.

Однако при решении любых практических задач и особенно при решении неформализованных задач распространена обратная ситуация. Эксперту приходится работать в условиях неполноты и неточности знаний (данных) и, как правило, в условиях дефицита времени. Когда эксперт решает задачу, он использует методы, отличающиеся от формальных математических рассуждений. В математических рассуждениях каждое заключение должно строго следовать из предыдущей информации. В противоположность этому в правдоподобных рассуждениях, основанных на здравом смысле, заключения основываются на частичной информации. В этом случае эксперт делает правдоподобные предположения, которые он не может доказать; тем самым вопрос об их истинности остается открытым. Все утверждения, полученные на основе этих правдоподобных предположений, также не могут быть доказаны.

Один из способов обоснования предположений заключается в том, чтобы рассматривать их как возможные значения, задаваемые по умолчанию. Например, высказав предположение, что сейчас 14 ч (посмотрев на часы), мы молчаливо предполагаем, что часы идут и идут правильно. Обычно человек знает, что некоторые предположения верны только при определенных условиях. Если информация, указывающая на нарушение этих условий; отсутствует, то предположение может быть высказано. Другое обоснование предположений базируется на рассмотрении рассуждения как процесса с ограниченными ресурсами. Так, можно считать, что предположение (X) имеет место, если, используя ограниченные ресурсы, нельзя доказать истинность противоположного утверждения. Предположение и выводы, сделанные на его основе, должны устраняться, если появилась информация, показывающая ошибочность этого предположения. Этот аспект в построении умозаключений с использованием предположений называется *немонотонностью.*

Любая *формальная система* является *монотонной,* т.е. если A, B и C есть некоторые высказывания, такие, что если *B* выводится из *A,* то *B* будет выводиться и из $A \wedge C$. Система *немонотонна,* если *B* выводится из *A*, но $\neg B$ выводится из $A \wedge C$[8]. Немонотонные рассуждения особенно важны при решении задач планирования и конструирования. В этих задачах пространство поиска иногда очень велико, и нет возможности предвидеть все последствия сделанного выбора. Так, например, конструктор знает, чего он хочет, но не знает, как это сделать. Поэтому при конструировании предположения выступают в виде пробных решений, последствия которых затем анализируются с точки зрения их пригодности (непригодности). Если последствия не противоречат тому, что хотел конструктор, то процесс конструирования продолжается дальше, возможно, с выдвижением новых предположений. В противном случае необходимо устранить все последствия и сделать альтернативное предположение и т.д.

Итак, для того чтобы система могла делать умозаключения, основанные на здравом смысле, при работе с неполными (неточными) данными и знаниями, она должна быть способна делать предположения, а при получении новой информации, показывающей ошибочность предположений, отказываться как от сделанных предположений, так и от умозаключений, полученных на основе этих предположений. Мнение системы о том, какие факты имеют место, изменяется в ходе рассуждения, т.е. можно говорить о *ревизии мнений.* Таким образом, даже если рассматривать проблемную область как статическую, неполнота (и неточность) знаний и данных влечет за собой рассмотрение этой области при различных (и даже противоположных) предположениях, что, в свою очередь, приводит к представлению области в виде альтернативных пространств, соответствующих различным, возможно, противоречивым и (или) взаимодополняющим предположениям и мнениям.

Мнение *B* можно представлять в виде отношения, состоящего из субъекта мнения *a* (индивида или системы, имеющих рассматриваемое мнение), объекта мнения *p* (то, о чем субъект имеет мнение) и обоснования мнения *r* (причины, по которым субъект имеет данное мнение), или в формальном виде - B *(a, p, r) (a* думает, что *p*, так как *r).*

Будем говорить, что множество мнений, свойственных некоторому индивиду (системе), составляет его *систему мнений.* Основываясь на некоторой системе мнений, можно образовать пространство поиска, предназначенное для решения каких-либо задач. В ходе рассуждений человек (система) может менять свои мнения, образуя различные системы мнений. Совокупность мнений, которой система придерживается в текущий момент, будем называть *активной системой мнений.* Каждой из систем мнений соответствует свое пространство поиска, а все вместе они образуют альтернативные пространства. На рис. 6.7 изображены три альтернативных про-

странства *P, Q* и *R*. Пространство *P* образовано исходными посылками C1, B1, предположениями A1 и выводами, сделанными на их основе; пространство *Q* образовано из C1, B1, предположений A2, D1 и выводов, сделанных на их основе; пространство *R* образовано из C1, B1, предположений A2, D2 и выводов, сделанных на их основе.

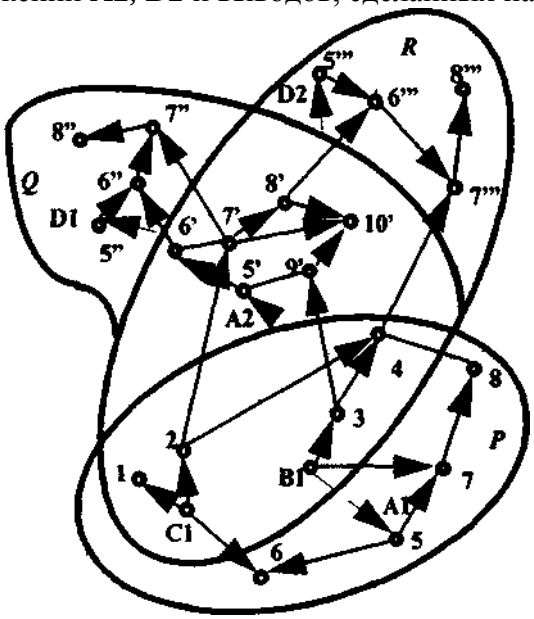


Рис.6.7. Пример альтернативных пространств

Для того чтобы изменить мнение, система должна быть способна рассуждать о зависимостях, существующих в активной системе мнений. Новые мнения могут быть следствием новой информации, полученной извне, или выведенной. Зависимости в системе мнений должны содержать сведения о мнениях, правилах вывода и обоснованиях (поддержках) мнений. Простейшим видом обоснования (justification) может являться информация о том, на каких мнениях основываются данные. Например, предположим, что система имеет следующие утверждения:

Мнение 1: *R* участвует в совещании в среду в 13 ч.

Мнение 2: Если *R* присутствует на совещании, то он занят.

Правило 3: Modus ponens $(A, A \supset B \models B)$.

Из этой информации можно сделать вывод:

Мнение 4: *R* занят в среду в 13 ч.

Обоснованием для этого мнения может быть следующая запись:

Обоснование 1:

Поддержка для: Мнение 4.

Правило вывода: Правило 3.

Посылки: (Мнение 1, Мнение 2).

Обоснование может использоваться для поддержки или изменения текущей системы мнений (ревизии мнений).

Остановимся на особенностях механизма поиска в альтернативных пространствах. В рассмотренных ранее методах поиск в некотором направлении прерывался при достижении целевого или терминального состояния. Если достигалось целевое состояние, то либо работа завершалась (при поиске одного решения), либо продолжался поиск следующего решения. При достижении терминального состояния необходимо вернуться в некоторое предыдущее состояние пространства и продолжить поиск в новом направлении. Обычно при реализации поиска использовался механизм *бэктрекинга* (backtracking). Бэктрекинг работает по принципу "last-in, first-out" ("последним вошел, первым вышел"), т.е. сначала устраняется последнее рассматриваемое состояние, для реализации которого может быть применен механизм стека.

Применение механизма бэктрекинга при поиске в альтернативных мирах будет приводить к излишней неэффективности, так как все неудачи, возникшие при поиске в одном направлении, не запоминаются при переходе к поиску в другом направлении, Та же самая причина неудачи может заново обнаруживаться и на новом направлении. Так, например, если мы взяли стакан правой рукой и из-за того, что он горячий, отпустили его, то нецелесообразно тут же пытаться брать стакан левой рукой (что, образно говоря, будет делать бэктрекинг).

Таким образом, механизм традиционного бэктрекинга отбрасывает слишком много информации. Осуществлять возврат целесообразно не к состоянию, непосредственно предшествующему данному, а к тому состоянию, которое является причиной возникновения неудачи. В используемых нами терминах причиной неудач являются предположения, т.е. недоказуемые утверждения. Поэтому при обнаружении неудачи необходимо возвращаться в состояние, где это предположение было сделано, и испытывать другое предположение. Так, например, если при получении утверждения 8 (см. рис. 6.7) система установила наличие неудачи (противоречия), то возврат нужно делать не к предыдущему шагу (к утверждению 7), а к шагу 5, на котором было сделано предположение A1, и заменять A1 на некоторое новое предположение A2. Таким образом осуществляется переход из пространства *P* в новое пространство *(R* или *Q).* Для выполнения описан-

кого способа поиска можно использовать информацию о зависимости, представленную в том или ином виде. По этой причине данный метод поиска называют *поиском, направляемым зависимостью.*

5. Поиск с использованием нескольких моделей

Все методы поиска, рассмотренные до сих пор, использовали при представлении проблемной области какую-то одну модель, т.е. рассматривали область с какой-то одной точки зрения. При решении сложных задач в условиях ограниченных ресурсов использование нескольких моделей может значительно повысить мощность системы. Объединение в одной системе нескольких моделей дает возможность преодолеть следующие трудности. Во-первых, переход с одной модели на другую позволяет обходить тупики, возникающие при поиске в процессе распространения ограничений. Во-вторых, использование нескольких моделей позволяет в ряде случаев уменьшить вероятность потери хорошего решения (следствие неполного поиска, вызванного ограниченностью ресурсов) за счет конструирования полного решения из ограниченного числа частичных кандидатов путем их расширения и комбинации. В-третьих, наличие нескольких моделей позволяет системе справляться с неточностью (ошибочностью) данных. Следует отметить, что использование нескольких моделей требует дополнительных знаний о том, как создавать и объединять различные точки зрения.

Рассмотрим метод использования нескольких моделей, впервые примененный в экспертной системе SYN [1]. SYN представляет собой программу для синтеза электрических схем. Система определяет значения компонент схемы, форму схемы и некоторые особенности ее работы. Новизна SYN состоит в использовании нескольких моделей, т.е. SYN может рассматривать схему с различных точек зрения, что соответствует идее эквивалентных электрических схем. Например, SYN может рассматривать делитель напряжения как состоящий из двух последовательно соединенных сопротивлений (R_1; R_2), а может рассматривать эти сопротивления как одно ($R = R_1 + R_2$). Так, при анализе делителя напряжения SYN использует вторую точку зрения для вычисления тока, проходящего через делитель ($I = U/R$). Затем для вычисления напряжения в средней точке делителя ($U_1 = I \times R_2$, $U_2 = I \times R_2$) SYN возвращается к первой точке зрения. Идея поочередного использования эквивалентных представлений электрических схем позволяет преодолеть тупики, возникающие при распространении ограничений. Мощность использования нескольких (в описанном случае двух) моделей состоит в том, что этот подход обеспечивает дополнительные пути по распространению поиска (расчета). Используя идею эквивалентных электрических схем, система SYN способна исследовать сложные схемы без трудоемких алгебраических вычислений.

6. Выбор метода решения задач

Выбор метода решения задачи зависит прежде всего от сложности задачи, которая определяется особенностями проблемной области и требованиями, предъявляемыми пользователем к решению задачи. Простые задачи характеризуются небольшой размерностью пространства поиска, точностью и полнотой данных, статичностью области, возможностью адекватного описания области с помощью одной модели. На практике встречается мало приложений, удовлетворяющих перечисленным требованиям.

Сложные задачи характеризуются тем, что значение хотя бы одного из перечисленных параметров оказывается в них более сложным. Для преодоления трудностей, вызванных большим

пространством поиска, используются методы, основанные на введении иерархий пространств (конкретных, абстрактных и метапространств). Простейший из этих методов основывается на факторизуемости пространства решений, что позволяет производить раннее отсечение. Метод обеспечивает получение всех решений. Если пространство поиска не удается факторизовать, но при этом не требуется получать все решения или выбирать лучшее, то могут быть применены методы, использующие иерархию однородных абстрактных пространств. Если пространство поиска таково, что любая задача может быть сведена к известной заранее последовательности подзадач, то используется фиксированное абстрактное пространство.

Эффективность этого метода определяется возможностью использовать безвозвратную стратегию. В тех случаях, когда решение задачи не может быть получено без механизма бэктрекинга, применяются более сложные методы. Метод нисходящего уточнения применим в том случае, когда все задачи не могут быть сведены к фиксированному набору подзадач, однако существует фиксированная упорядоченность понятий области и фиксированный частичный порядок между подзадачами. В случае, если подзадачи взаимозависимы, т.е. для решения некоторой подзадачи может требоваться информация, получаемая Другой подзадачей, и подзадачи не могут быть упорядочены, целесообразно применять принцип наименьших свершений. Этот подход позволяет приостанавливать решение подзадачи, для которой недостает информации, переходить к решению другой подзадачи и возвращаться к исходной задаче, когда отсутствующая информация станет доступной.

Следует отметить, что использование данного подхода требует более разнообразных знаний о решении задачи, чем в предыдущих случаях. При использовании разнообразных знаний о процессе решения становится целесообразным объединять принцип наименьших свершений с методами, использующими метазнания. Принцип наименьших свершений может приводить к образованию тупиков в процессе решения задачи, что препятствует использованию этого принципа в чистом виде. Для преодоления тупиков используют предположения или применяют метод нескольких моделей. Для преодоления трудностей, вызванных неполнотой и (или) неточностью данных (знаний), используют вероятностные, размытые и точные методы. Все эти методы основываются на идее увеличения надежности путем комбинирования фактов и использования метазнаний о возможностях комбинирования фактов. Неточные подходы (вероятностные, псевдовероятностные, размытые) используют разнообразные априорные оценки, условные вероятности и размытые множества; точные подходы используют предположения и ревизию мнений при немонотонных рассуждениях.

Для преодоления неадекватности модели проблемной области используются методы, ориентированные на использование нескольких моделей. Эти методы позволяют объединить возможности различных моделей, описывающих проблемную область с различных точек зрения. Кроме того, использование нескольких моделей позволяет уменьшить вероятность потери хорошего решения, несмотря на неполноту поиска, вызванную ограниченностью вычислительных ресурсов.

Контрольные вопросы:

1. Охарактеризуйте метод поиска решений в одном пространстве.
2. Охарактеризуйте метод поиска решений в иерархии пространств.
3. Охарактеризуйте метод поиска решений в альтернативных пространствах при неполных и неточных данных.
4. Охарактеризуйте метод поиска решений с использованием нескольких моделей.
5. Приведите обоснование выбора метода решений задач в экспертных системах.

ЛИТЕРАТУРА

1. *Попов Э.В.* **Экспертные системы. Решение неформализованных задач в диалоге с ЭВМ.** - М: Наука, 1987.

2. *Попов Э.В., Фридман Г. Р.* **Алгоритмические основы интеллектуальных роботов и искусственного интеллекта.** - М.: Наука, 1976. - 455 с.

3. *Минский М.* **На пути к созданию искусственного разума**//Вычислительные машины и мышление. - М.: Мир, 1967. - 552 с.

4. *Нильсон Н.* **Искусственный интеллект. Методы поиска решений. -** М.: Мир, 1973.

5. *Moore R.* **Expert Systems in Real-Time Applications: Experience and Developments** //Proceedings of the Seventeenth Annual Advance Control Conferenc, 1991, October.

6. *Nevell A., Simon M.A.* **Computer science as empirical enquiry: Symbols and search**//Communications of the ACM. - 1976. - V. 10. - № 3. - P. 133 - 146.

7. *Nilsson N.J.* **Principles of Artificial intelligence. -** Palo Alto; California, Tioga Press, 1980.

8. *Winograd T.* **Extended inference modes in reasoning by computer systems//**Artificial Intelligence. - 1980. - V. 13. - P. 5 - 26.

Лекция 14: Знания в экспертных системах: концептуалые; фактические, предметные; алгоритмические, процедурные

План:

1.Формирование эмпирических знаний, стратегий и эвристик
2.Формирование знаний о динамике нелинейной системы автоматического управления.
3.Знания, передаваемые экспертной системе

1. Формирование эмпирических знаний, стратегий и эвристик

Для обеспечения универсальности экспертного регулятора теоретические знания обязательно должны быть дополнены эмпирическими знаниями о зависимости пространства качества САУ от параметров регулятора

Формирование знании о динамике линейной системы автоматического управления. Передаточная функция ОУ первого порядка

$$W_{OY}(s)=\frac{K}{s+a}, \qquad (37)$$

где K — коэффициент усиления ОУ, $1/a$—постоянная времени апериодического звена $(a \neq 0)$; при $a=0$ ОУ представляет интегрирующее звено.

В рассматриваемом случае синтез может осуществляться только на основе П-регулятора, так как любой иной линейный регулятор переводит систему в класс САУ второго порядка. Хорошо известно, что для улучшения прямых показателей качества следует увеличивать коэффициент П-рсгулятора настолько, насколько позволяют возможности при практической реализации регулятора. Таким образом, правило по синтезу системы первого порядка П-регулятором следующее.

Правило 23. ЕСЛИ (увеличить коэффициент П-регулятора), ТО (прямые показатели качества переходного процесса улучшатся).

Синтез объекта управления второго порядка может осуществляться либо П-регулятором, либо ПД-регулятором. Передаточные функции ОУ второго порядка и замкнутой системы, включая П-рсгулятор, имеют вид

$$W_{OY}(s)=\frac{K}{c_0 s^2+c_1 s+c_2}, \qquad (38)$$

$$W_3(s)=\frac{b_0}{a_0 s^2+a_1 s+a_2}, \qquad (39)$$

ъ

где $c_0=a_0,\ c_1=a_1,\ a_2=c_2+b_0,\ b_0=K K_p;\ K$ — коэффициент усиления ОУ, Kp — коэффициент П-регулятора.

Несложно показать, что при соблюдении условия

$$b_0 \le \frac{c_1^2}{4c_0}-c_2 \qquad (40)$$

в системе гарантируется монотонный переходный процесс. Минимальное время регулирования t_p в классе монотонных процессов будет в том случае, когда выражение (40) превращается в равенство. Этот случай характеризуется наличием кратных корней в знаменателе передаточной функции (39) замкнутой системы.

С другой стороны, при нарушении неравенства (40) переходный процесс сначала переходит в класс апериодических, а затем в класс колебательных процессов. Однако возможно уменьшение по сравнению с t_p времени регулирования монотонных процессов с помощью увеличения коэффициента П-регулятора до тех пор, пока в апериодическом или колебательном движении первый максимум кривой переходного процесса относительно установившегося значения не выйдет за пределы ±Δ, где Δ—заданная малая постоянная величина, представляющая собой допустимую ошибку. При этом допустимое приращение относительно максимально возможного b_0 монотонного процесса будет

$$\Delta b_0 = \left[\frac{\pi a_1}{\ln \Delta}\right]^2 \frac{1}{4a_0}. \qquad (41)$$

Таким образом, знания или правила по настройке П-регулятора системы автоматического управления второго порядка следующие.

Правило 24. ЕСЛИ (требуется, чтобы переходный процесс был МОНОТОННЫМ), ТО (*b_0 следует выбирать исходя из неравенства (40)).

Правило 25. ЕСЛИ (требуется, чтобы переходный процесс был монотонным и с максимальным быстродействием), ТО (*b_0 следует выбирать исходя из равенства в (40)).

Правило 26. ЕСЛИ (требуется повысить быстродействие системы и при этом нет ограничений на вид переходного процесса), ТО (следует увеличить b_0 на величину Δb_0, определяемую из (41)).

Перейдем к синтезу ПД-регулятора для ОУ второго порядка. В этом случае передаточная функция объекта управления описывается выражением (38), а передаточная функция замкнутой системы имеет вид

$$W_3(s) = \frac{b_0 + b_1 s}{a_0 s^2 + a_1 s + a_2}, \qquad (42)$$

где $b = K K_d$; K_d — коэффициент дифференциального канала ПД-регулятора.

В рассматриваемом случае монотонный переходный процесс гарантируется при выполнении условия

$$c_1 - \sqrt{c_1^2 - 4a_0 c_2} \le \frac{2c_2 K_d}{K_p} \le c_1 + \sqrt{c_1^2 - 4a_0 c_2}. \qquad (43)$$

При этом монотонный переходный процесс с максимальным быстродействием будет обеспечиваться при выполнении условия

$$\frac{K_d}{K_p} = \frac{c_1 + \sqrt{c_1^2 - 4a_0 c_2}}{2c_2}. \qquad (44)$$

Равенство (44) интерпретируется следующим образом: если объект управления содержит два апериодических звена, то K_d / K_p равно максимальному значению из постоянных времени этих звеньев; если ОУ содержит апериодическое звено и интегратор, то K_d / K_p равно постоянной времени апериодического звена.

Так же, как в случае с Л-регулятором, возможно уменьшение времени регулирования по сравнению с полученным t_p монотонного процесса с максимальным быстродействием. Однако, в отличие от П-регулятора, для ПД-ре гулятора не существует аналитического выражения допустимого приращения. Поэтому в данном случае следует сначала выбрать Kp исходя из требований по точности, предъявляемой к САУ, а Kd увеличивать до тех пор, пока в апериодическом или колебательном движении первый максимум кривом переходного процесса относительно установившегося значения не выйдет за пределы ±Δ.

Для настройки ПД-регулятора следует сначала выбрать K_p исходя из требований по точности, после чего воспользоваться следующими правилами.

Правило 27. ЕСЛИ (задача обеспечения требуемых точностных характеристик системы не решается), ТО (выбрать K_p произвольно).

Правило 28. ЕСЛИ (требуется, чтобы переходный процесс САУ принадлежал классу монотонных процессов), ТО (Kd следует выбирать из неравенства (43)).

Правило 29. ЕСЛИ (требуется, чтобы переходный процесс САУ был монотонным и, кроме того, с максимальным быстродействием), ТО (Kd следует выбирать из равенства (44)).

Правило 30. ЕСЛИ (требуется повысить быстродействие системы и при этом нет ограничений на вид переходного процесса), ТО (следует увеличивать K_d до тех пор, пока в апериодическом или колебательном движении первый максимум кривой переходного процесса относительно установившегося значения не выйдет за пределы ± Δ).

При формировании БЗ для настройки ПД-регулятора САУ третьего порядка нет необходимости в идентификации параметров объекта управления. В этом случае знания о свойствах пространства качества системы (так же, как и для систем первого и второго порядка) априорны и несколькими пробными изменениями коэффициентов ПД-регулятора можно определить, в какой точке пространства качества находится система и что следует предпринять (изменяя коэффициенты регулятора) для удовлетворения предъявляемых к системе требований.

В рассматриваемом случае передаточные функции объекта управления и замкнутой системы с ПД-регулнтором имеют вид

$$W_{\text{ОУ}}(s) = \frac{K}{s^3 + a_1 s^2 + a_2 s + a_3}, \qquad (45)$$

$$W_3(s) = \frac{K(K_d s + K_p)}{s^3 + a_1 s^2 + a_2 s + a_3 + K(K_d s + K_p)}. \qquad (46)$$

Произведем в (46) замену переменной $s = p a_1$; тогда получим

$$W_3(s) = \tilde{K}\frac{1+(K_p a_1 / K_p)p}{p^3 + p^2 + x\,p + y}, \qquad (47)$$

где $\tilde{K} = K K_p / a_1^3,\ x = K K_d + a_2 / a_1^2,\ y = K K_p + a_3 / a_1^3.$ При этом нормированное характеристическое уравнение системы имеет вид

$$p^3 + p^2 + xp + y = 0, \qquad (48)$$

условия устойчивости по Гурвицу; $x>0, y>0$.

Известно, если x и y — действительные числа, то уравнение (48) имеет или один действительный и два сопряженных комплексных корня, или три действительных корня, но крайней мере два из которых равны, или три различных действительных корня в зависимости от того, будет

$$Q(x,y) = \frac{4x^3 - x^2 + 27y^2 + 4y - 18xy}{108}$$

соответственно положительно, равно нулю или отрицательно.

На рис. 5 показана плоскость (x, y), разбитая на области I, II, III. В областях I и II $Q>0$, а в области III $Q<0$. Линия $Q(x,y) = 0$, обозначенная на рисунке буквой $Г$, является границей между областями III и I, II, а уравнение линии P (равенства действительных частей корней и границы между областями I и II имеет вид

$$y = \frac{x}{3} - \frac{2}{27}, \quad x > \frac{1}{3}. \qquad (49)$$

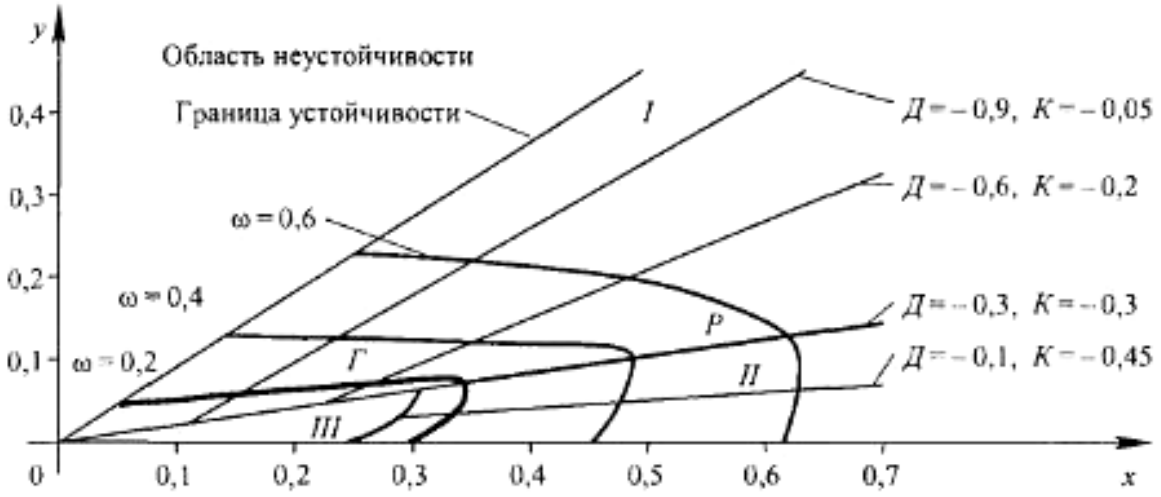


Рис 5. Диаграмма расположения корней характеристического уравнения САУ третьего порядка с ПД-регулятором

На рис. 5 также показано расположение корней внутри каждой из этих областей. Область III, где все корни вещественные, характеризуется апериодическим переходным процессом. В областях I и II , где имеется один веще-

ственный корень и два комплексных, переходный процесс будет соответственно колебательным и монотонным.

Для более точной оценки характера протекания переходного процесса нанесем вспомогательные линии, разбивающие плоскость *(x,y)* на еще более мелкие части, что позволит иметь более полное суждение о быстродействии и запасе устойчивости:

— линии постоянных значений действительных частей корней (обозначение на линии *Д*=-0,15; *К*= -0,7 означает, что если точка попадет на эту линию, то действительный корень уравнения (48) равен -0,15, а действительная часть комплексного корня равна -0,7), определяемые уравнением

$$y = -C_1 x + \frac{1}{3}R - R^3 - \frac{2}{27}, \qquad (50)$$

где $R = C_1 + 1/3$, C_1 — значение вещественного корпя;

—линии постоянства степени колебательности (обозначение на линии со = 0,6 означает, что если точка попадет на эту линию, то мнимая часть корня уравнения (48) равна 0,6), определяемые уравнением

$$y^2 - \frac{2}{3}\left[r + \frac{1}{9}\right]y + \frac{1}{9}\left[r + \frac{1}{9}\right]^2 + \frac{4}{27}r^3 - C_2^2(C_2^2 - r)^2 = 0, \qquad (51)$$

где $r = x - 1/3$, C_2 - — значение мнимой части корня.

Как видно из рис. 5, наибольшая степень устойчивости имеет место в точке с координатами x = 1/3, y = 1/27. Следовательно, эта точка соответствует наилучшим значениям параметров системы с точки зрения величины степени устойчивости. Однако степень устойчивости является приближенной оценкой быстроты затухания переходного процесса. Поэтому при выборе параметров регулятора не обязательно попадать именно в эту точку плоскости *(x,y)*. Можно считать, что наилучшей областью параметров системы будет область, прилегающая к данной точке и находящаяся вблизи линии равенства действительных частей корней, определяемой уравнением (49). Таким образом, при синтезе параметров регулятора следует по форме переходного процесса определить, в какой области плоскости *(x, y)* находится точка, и предпринять соответствующие действия, позволяющие приблизить ее к линии *P*. Сформулируем эти действия в виде правил:

Правило 31. ЕСЛИ (переходный процесс монотонный (корни характеристического уравнения расположены в области II)), ТО (следует увеличить K_p) ИЛИ (уменьшить *Kd)*.

Правило 32. ЕСЛИ (переходный процесс колебательный (корни характеристического уравнения расположены в области I)), ТО (следует уменьшить K_p) ИЛИ (увеличить *Kd)*.

Однако при этом необходимо не забывать, что уменьшение коэффициента *Kp* пропорционального канала ПД-регулятора ведет к ухудшению точностных показателей системы, а увеличение *Kd* — к росту степени колебательности в переходном процессе. Аналогичные рассуждения можно провести для ПИ-регулягора и объекта управления второго порядка.

При формирования эмпирических знаний но настройке параметров ПИД-регулятора простой аналитической зависимости между пространством параметров

регулятора и пространством качества системы уже нет. Поэтому в этом случае знания экспертным регулятором формируются на основе обучения. При этом набор эмпирических знаний для САУ формируется на основе алгоритма обучения или построения и анализа поверхностей качества в окрестности "рабочей точки".

Рассмотрим пространство L = {Параметры регулятора + Параметры объекта управления. Параметры качества переходного процесса}. В общем случае поверхности качества в пространстве L представляют собой сложные многомерные поверхности, например, $\{\sigma = F_1(R,P),\ t_p = F_2(R,P)\}$, где F_1, F_2 — поверхности качества; R, P — соответственно векторы параметров регулятора и объекта управления; σ, t_p — показатели качества (соответственно перерегулирование и время регулирования).

Определим в пространстве L "рабочую точку" как (R',P').

При синтезе параметров регулятора (18) на основе критерия максимальной степени устойчивости (19) корни характеристического уравнения системы обладают следующим уникальным свойством: на вертикальной линии комплексной плоскости, проходящей через точку $(-J_{опт}, j0)$ расположено m корней, а остальные n-m корней расположены левее этой линии.

Указанное свойство позволило выдвинуть следующую гипотезу. В окрестности "рабочих точек" $(R',P')_1$ и $(R',P')_2$ двух систем управления, подобных друг другу в смысле совпадения структуры их передаточных функций и синтезированных на основе критерия (19), соответствующие поверхности σ_1, t_{p1} и σ_2, t_{p2}, качественно подобны.

Предложенная гипотеза иллюстрируется рис. 6, на котором изображены поверхности качества двух систем четвертого порядка с различными значениями параметров объекта управления и коэффициентов ПИД-регулятора.

Рассмотрим теперь получение эмпирических знаний на основе алгоритма обучения. Задачу обучения ЭР можно сформулировать следующим образом. Дано: множество переходных процессов системы F\ пространство параметров системы P, которое состоит из подпространства S параметров объекта управления и подпространства R параметров регулятора; пространство показателен качества системы Q; текущее значение вектора показателей качества системы Q'_f исходное значение вектора параметров регулятора R . Требуется сформировать в качественных категориях ("много", "большой", "мало", "средний" и т. д.) зависимость между подпространством параметров регулятора и подпространством показателей качества переходного процесса в окрестности рабочей (полученной при синтезе) точки (R*, P*). Задача обучения решается специальным блоком ЭР, представляющим собой алгоритмический модуль моделирования переходных процессов САУ. При этом выполняется следующая последовательность действий: осуществляется последовательный перебор значений коэффициентов регулятора в заранее определенных пределах; при каждом изменении параметра рассчитывается и строится переходный процесс САУ; производится анализ изменений показателей качества и на основе его записываются правила.

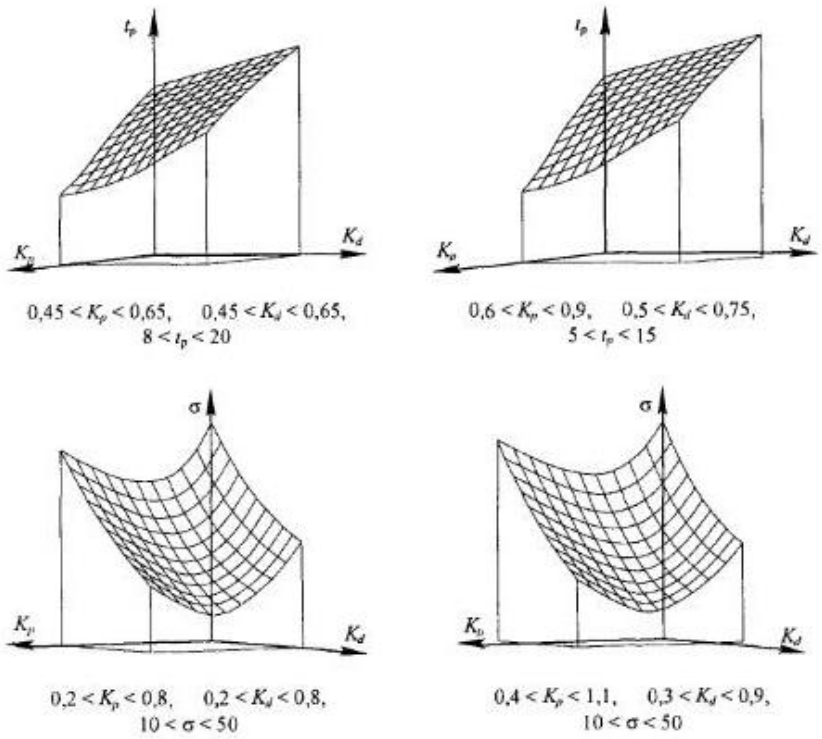


Рис.6 Поверхности качества двух САУ четвертого порядка с различными значениями параметров ОУ и ПИД-регулятора

2. Формирование знаний о динамике нелинейной системы автоматического управления.

Для формирования знаний о динамике нелинейной САУ предлагается использовать диаграммы качества нелинейных колебаний [9].

При построении диаграмм качества предполагается, что приведенная линейная часть системы обладает хорошими фильтрующими свойствами, в системе устанавливается колебательный переходный процесс и решение в первом приближении может быть представлено в виде

$$u(t) = A(t)\sin(\psi(t)), \qquad \frac{dA}{dt} = A\xi(A), \qquad \frac{d\psi}{dt} = \omega(A),$$

где ξ и ω — соответственно показатель затухания и частота, определяемые как функции переменной амплитуды A из характеристического уравнения данной системы после ее гармонической линеаризации; $u(t)$ — сигнал, подаваемый на вход нелинейного элемента.

Диаграммы качества представляют собой семейство линий $\xi = \text{const}$ и линий $\omega = \text{const}$ на плоскости (A,K), где A — амплитуда сигнала, подаваемого на нелинейный элемент, K — один из параметров системы. Эти диаграммы позволяют судить о виде переходного процесса, о быстроте его затухания или о времени установления автоколебаний, а также о частоте колебаний и о характере ее изменения во времени. Таким образом, по диаграммам можно оценить изменения показателей качества при варьировании параметров нелинейной системы и эффективно формировать знания о ее динамике.

1.При формировании эмпирических знаний о динамике нелинейной системы управления, модель которой удовлетворяет гипотезе фильтра, предлагается

следующая последовательность действий. Записать характеристическое уравнение данной гармонически линеаризованной системы.

2.Разложить характеристическое уравнение системы на сомножители, последний из которых соответствует основной паре комплексных корней $\lambda_{1,2} = \xi \pm j\omega$, определяющей колебательный переходный процесс в исследуемой системе.

3. Определить функциональные зависимости

$$\xi = F_1(A, K) \quad \text{и} \quad \omega = F_2(A, K).$$

4. Построить диаграммы качества.

5. С помощью приближенных выражений для перерегулирования σ и времени регулирования t_p соответственно

$$\sigma \approx \exp\frac{\pi\xi_c}{\omega_c}, \quad \xi_c < 0, \quad t_p \approx \frac{1}{\xi_c}\ln\frac{A_k}{A_0},$$

где ξ_c и ω_c—средние значения величин ξ и ω, взятые из диаграмм качества для исследуемого участка, A_k и A_0—соответственно конечное и начальное значения амплитуды, построить функциональные зависимости

$$\sigma = F_1(K), \quad t_p = F_2(K).$$

6. На основе анализа последних функциональных зависимостей сформировать эмпирические знания о динамике нелинейной системы.

3. Знания, передаваемые экспертной системе

Знания, передаваемые экспертной системе, можно разделить на три категории:

Концептуальное (***на уровне понятий***) ***знание*** – это знание, воплощенное в словах или, конкретнее, в научно-технических терминах и, естественно, в стоящих за этими терминами классах и свойствах объектов окружающей среды. Сюда же входят связи, отношения и зависимости между понятиями и их свойствами, причем связи абстрактные, также выраженные словами и терминами. Концептуальное знание – это сфера главным образом фундаментальных наук, если учитывать, что понятие есть высший продукт высшего продукта материи – мозга.

Фактуальное, предметное знание – это совокупность сведений о качественных и количественных характеристиках конкретных объектов. Именно с этой категорией знания связываются термины «информация» и «данные», хотя такое употребление терминов несколько принижает их значение. Любое знание несет информацию и может быть представлено в виде данных; фактуальное знание – это то, с чем всегда имели дело вычислительные машины и с чем они больше всего имеют дело до сих пор.

Алгоритмическое процедурное знание – это то, что принято называть словами «умение», «технология» и др. В вычислительном деле алгоритмическое знание реализуется в виде алгоритмов, программ и подпрограмм, но не всяких, а таких, которые могут передаваться из рук в руки и использоваться без участия авто-

ров. Такая реализация алгоритмического знания называется программным продуктом. Наиболее распространенные формы программного продукта – пакеты прикладных программ, программные системы и другие, ориентированные на конкретную область применения ДЭС. Организация и использование пакетов прикладных программ базируется на концептуальном знании.

Контрольные вопросы:

1.Формирование эмпирических знаний, стратегий и эвристик?
2.Формирование знаний о динамике нелинейной системы автоматического управления?
3.Знания, передаваемые экспертной системе?
4.Знания в экспертных системах: концептуалые; фактические, предметные; алгоритмические, процедурные?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 15: Создания и применение базы знаний в интеллектуальных системах

План:

1. Сущности и иерархия классов
2. Иерархия модулей и рабочих пространств
3. Структуры данных БЗ

Все знания в G2 хранятся в двух типах файлов: базы знаний (БЗ) и библиотеки знаний (БиЗ). В файлах БЗ хранятся знания о приложениях: определения всех объектов, объекты, правила, процедуры и т.п. В файлах БиЗ хранятся общие знания, которые могут быть использованы более чем в одном приложении, например определение стандартных объектов. Файлы БЗ имеют расширение kb (knowledge base), а файлы БиЗ - kl (knowledge libraries). Файлы БЗ могут путем замены расширения преобразоваться в БиЗ и обратно.

В целях обеспечения повторной используемости приложений в G2 реализовано средство, позволяющее объединять ранее созданные kb и kl-файлы с текущим приложением. При этом G2 автоматически выявляет и выводит на дисплей конфликты в объединяемых знаниях.

Знания в G2 структурируются следующими способами: иерархия классов, иерархия модулей, иерархия рабочих пространств. Каждая из указанных иерархий может быть показана на дисплее, используя возможность "Inspect" (см. ниже).

1. Сущности и иерархия классов

Класс является основой представления знаний в G2. Понятие "класс в G2" базируется на объектно-ориентированной технологии (ООТ). Как уже указывалось ранее (см. гл.5), использование ООТ является на текущем уровне развития ИИ и вообще программирования главной тенденцией.

В ООТ структуры данных представляются в виде классов объектов (определений объектов), имеющих определенные атрибуты. Классы наследуют атрибуты от суперклассов и передают свои атрибуты подклассам. Каждый класс (исключая корневой) может иметь конкретные экземпляры класса. В четвертой версии G2 введен механизм множественного наследования. Теперь в системе достаточно легко произвести, например, новый класс саморегулирующихся насосов от классов контроллеров и насосов. В системе достаточно изящно решена проблема конфликтов между именами атрибутов. Использование ООТ обеспечивает следующие преимущества:

1) уменьшает избыточность и упрощает определение классов, так как определяется не весь класс, а только его отличия от суперкласса;

2) позволяет использовать общие правила, процедуры, формулы, что уменьшает их количество;

3) является естественным для человека способом описания сущностей.

Класс в G2 является основой представления знаний. Все, что хранится в БЗ и чем оперирует система, является экземпляром того или иного класса. Все синтаксические конструкции G2 тоже являются классами. Для сохранения общности даже базовые типы данных - символьные, числовые, булевские и истинностные значения нечеткой логики представлены соответствующими классами. Описание класса (тоже экземпляр специального класса) включает ссылку на суперкласс (is-a-иерархия) и перечень атрибутов, специфичных для класса (part-of-иерархия).

Концептуально иерархия классов G2 берет свое начало от корневого класса, именуемого item-or-value (сущность или значение). Класс item-or-value сам по себе не может иметь экземпляров. Однако так как он является корнем всей иерархии классов, он определяет основное поведение всех классов G2. Item-or-value имеет два производных класса - value (хотя концептуально ветвь value представляется классом, в действительности это типы данных G2) и item. Каждый из этих классов имеет свои производные классы. Сущность (item) является корнем разветвленной иерархии классов. Наиболее важные ветви этой иерархии могут быть сгруппированы в небольшое число категорий. Они перечислены ниже (см. п. 9.1.3), в порядке их "видимости" для пользователя, начиная с наиболее "видимых".

2. Иерархия модулей и рабочих пространств

G2-приложение не представляет собой единый блок. Оно структурируется с помощью модулей и рабочих пространств на легко управляемые куски. Несмотря на то, что функции модулей и рабочих пространств похожи, между ними есть существенные различия, отмеченные ниже.

Приложение в G2 может быть организовано в виде одной БЗ или в виде нескольких БЗ, называемых модулями. В последнем случае говорят, что приложение модуляризировано (структурировано на модули). Модули приложения организованы в древовидную иерархию с одним модулем верхнего уровня. Модули следующего уровня состоят из тех модулей, без которых не может работать модуль предыдущего уровня. Эти модули называют "непосредственно требуемые модули". Существуют 2 способа создать G2-приложения.

1. Разрабатывается одномодульное приложение, которое затем при необходимости разделяется на отдельные модули.

2. Приложение изначально создается как состоящее из нескольких модулей. Некоторые из этих модулей разрабатываются впервые, а другие могут выбираться из библиотеки знаний.

Структурирование приложения на модули обеспечивает следующие преимущества:

• позволяет разрабатывать приложение одновременно нескольким группам разработчиков;

• упрощает разработку, отладку и тестирование;

• позволяет изменять модули независимо друг от друга;

• упрощает повторное использование знаний.

Рабочие пространства являются контейнерным классом, в котором размещаются другие классы и их экземпляры, например объекты, связи, правила, процедуры и т.д. Каждый модуль (база знаний) может содержать любое количество рабочих пространств. Рабочие пространства образуют одну или несколько древовидных иерархий с отношением is-a-part-of (является частью). С каждым модулем (базой знаний) ассоциируется одно или несколько рабочих пространств верхнего (нулевого) уровня, каждое из этих рабочих пространств является корнем соответствующей древовидной иерархии. В свою очередь, с каждым объектом (определением объекта или связи), расположенным в нулевом уровне, может быть ассоциировано рабочее пространство первого уровня, связанное с ним отношением "является частью", и т.д.

Различие между модулями и рабочими пространствами состоит в следующем. Модули разделяют приложение на отдельные базы знаний, совместно используемые в различных приложениях. Динамические модули (аналог библиотек динамического связывания) могут подгружаться и вытесняться из оперативной памяти во время исполнения программно и одновременно использоваться несколькими приложениями. Рабочие пространства выполняют свою роль при исполнении приложения. Они содержат в себе (и в своих подпространствах) различные сущности и обеспечивают разбиение приложения на небольшие части, которые легче понять и обрабатывать. Например, весь процесс разбивается на подпроцессы, и с каждым подпроцессом ассоциируется свое подпространство.

Рабочие пространства могут устанавливаться (вручную или действием в правиле-процедуре) в активное или неактивное состояние (т.е. сущности, находящиеся в этом пространстве и в его подпространствах, становятся невидимыми для механизма вывода). Механизм активации (деактивации) рабочих пространств используется, например, при наличии альтернативных групп правил, когда активной должна быть только одна из альтернативных групп.

Кроме того, рабочие пространства используются для задания пользовательских ограничений, определяющих поведение приложения для различных категорий пользователей.

3. Структуры данных БЗ

Глобально сущности в БЗ G2 с точки зрения их использования могут быть разделены на структуры данных и исполняемые утверждения. Примерами первых являются объекты и их классы, связи (connection), отношения (relation), переменные, параметры, списки, массивы, рабочие пространства и т.п. Примерами вторых - правила, процедуры, формулы, функции и т.п.

Опишем наиболее важные ветви иерархии "item".

Объект (object) и его подклассы. Объекты представляют объекты реального мира в приложении. Класс объектов определяет атрибуты, которые позволяют создать пиктограммы для

объектов, определить их положение на схемах и *отростки связей (stubs)* для присоединения их к другим объектам.

Связь (connection). Класс для изображения путей между объектами. Можно создать подкласс связей для указания различных типов потоков, которые могут существовать между объектами на схеме. Например, объекты могут быть соединены водопроводными трубами и (или) проводами, передающими логические сигналы. Определив различные классы для этих связей, можно быть уверенным, что G2 будет их различать и никогда не позволит воде течь по электропроводам.

Рабочее пространство БЗ (Kb-workspace). Класс, определяющий независимый сегмент базы знаний, который может быть активирован или деактивирован. Рабочие пространства отображаются как отдельные, ограниченные рабочие области, в которых можно помещать объекты и объединять их в схемы. Можно создать связи между рабочими пространствами с помощью *точек связи (connection posts).* По сути класс рабочих пространств является развитием концепции рабочей памяти в традиционных системах. Можно сказать, что рабочая память системы G2 строится на основе иерархии рабочих пространств. Иерархия рабочих пространств тесно связана с графическим представлением объектов. Рабочее пространство является контейнерным классом для экземпляров других классов. Каждый экземпляр объекта может обладать своим рабочим пространством, представляющим его внутреннюю структуру. Введение концепции рабочих пространств обеспечивает две важные функции системы G2: возможность осуществлять рассуждения на разных уровнях абстракции и возможность продолжительной (теоретически - бесконечной) работы системы без необходимости "сборки мусора" в пределах отведенного объема оперативной памяти, что очень важно для систем управления непрерывными процессами.

Классы пользовательского интерфейса (user-interface). Определяют такие элементы пользовательского интерфейса, как меню, селективные кнопки (radio button), сообщения (message), шкалы, круговые шкалы и многое другое. Можно определять подклассы класса сообщений, например, для создания сообщений со специальным способом отображения. Из всех классов пользовательского интерфейса только для сообщений есть возможность создавать производные классы.

Классы описаний классов (class definition) определяют классы, экземпляры которых содержат созданные пользователем описания классов и служат шаблонами для создания экземпляров других классов. Эти классы порождены от *класса описание (definition).* Описание имеет три подкласса - *описание объекта (object-definition), описание связи (connection-definition)* и *описание сообщения (message-definition)* в соответствии с классами, которые может определять пользователь.

Классы языка G2 (G2 language): эти классы используются для определения различных элементов языка G2, таких, как правила, отношения, действия и процедуры. Нельзя создать, собственные производные классы от этих классов.

С помощью G2 новые классы могут создаваться не только в процессе разработки, но и динамически, во время работы приложения. Во время исполнения приложения может быть создан, модифицирован или уничтожен экземпляр любого класса или целый класс. Это касается как объектов, так правил и процедур. В этом смысле G2 более объектно-ориентированная система, чем даже C++. Эта возможность является частью общих возможностей G2, дополняющих описания классов и позволяющих создавать новые сущности, включая рабочие пространства, правила, связи и процедуры. G2 обеспечивает операции *create by cloning (создание клонированием)* и *change the text of (изменить текст),* которые используются для клонирования похожего описания класса и последующего редактирования его в соответствии с требованиями новых особенностей. По умолчанию динамически созданные сущности являются *временными (transient),* т. е. они существуют только на протяжении данного сеанса работы и не сохраняются в базе знаний. Однако описание класса должно быть *постоянной (permanent)* сущностью в момент создания экземпляра или производного класса. Можно использовать операцию *make permanent (сделать постоянным)* для преобразования временной сущности -описания класса в постоянную.

Все классы G2 обладают по крайней мере одним общим свойством - их экземпляры имеют графическую форму представления. Используя эти графические образы вместе с классом связей, можно строить схемы систем для любого уровня сложности. Кроме визуализации взаимо-

действия объектов G2 предоставляет синтаксические конструкции, позволяющие осуществлять рассуждения на основе графических схем. Например, можно проверить состояние всех вентилей, соединенных с данной емкостью, или определить температуру объекта, ближайшего к указанному.

Рассмотрим подробнее наиболее важные классы сущностей.

Выделяют объекты (классы), встроенные в систему и вводимые пользователем. При разработке приложения, как правило, создаются подклассы пользовательских и встроенных классов, отражающие специфику данного приложения. Среди встроенных подклассов наибольший интерес представляет подкласс объектов, включающий подклассы переменных и параметров, и подкласс связей (connection) и отношений (relation).

Объекты

Объекты в базе знаний представляют собой отображения элементов реального мира, которые будут применяться при решении поставленной перед ЭС РВ задачи. Выделяют постоянные и временные объекты. *Постоянные объекты* заносятся в БЗ разработчиком ЭС РВ в процессе диалога с системой, в то время как *временные объекты* создаются после выполнения специальных команд в правилах и процедурах. Временные объекты могут существовать в БЗ только в процессе работы ЭС РВ. С каждым объектом ассоциируется таблица атрибутов, в которую заносятся существенные для решаемой задачи свойства объекта. Элемент данной таблицы представляет собой пару "атрибут -значение".

Объекты могут иметь графические образы, отображаемые на экране дисплея, называемые *пиктограммами.* На пиктограммах разработчиком могут быть выделены отдельные участки. Цвет таких участков может изменяться в результате выполнения специальных команд в правилах или процедурах. Таким способом обеспечивается высокая наглядность информации, предоставляемой лицу, работающему с ЭС РВ.

Поскольку реальные приложения могут содержать большое количество объектов, целесообразно предоставлять возможность объединения множества объектов со схожими свойствами в классы. Классы объектов составляют иерархию, в которой определяется отношение "родительский класс - подкласс". Объекты подклассов могут наследовать атрибуты и пиктограммы родительских классов.

Иерархическая упорядоченность классов значительно упрощает задачу определения новых классов в приложении. Например, атрибуты, характеризующие объекты различных классов, могут быть однократно определены в одном классе, являющемся общим родительским классом для них. Такие атрибуты будут автоматически наследоваться объектами, принадлежащими к подклассам, что снимает необходимость их повторного определения. Другим важным достоинством введения классов объектов является возможность составления правил, относящихся ко всем объектам, принадлежащим к некоторому классу (общих правил). Задача разработчика значительно упрощается за счет того, что им может быть составлен ряд общих правил, применимых к различным классам объектов приложения, а результирующая БЗ будет иметь меньший объем по сравнению с БЗ, в которой не могут применяться общие правила.

Особая роль в G2 отводится переменным. В отличие от статических систем переменные в G2 делятся на три вида: собственно переменные, параметры и простые атрибуты. *Параметры* - получают значения в результате работы машины вывода или выполнения какой-либо процедуры. *Переменные* представляют измеряемые характеристики объектов реального мира и поэтому имеют специфические черты: время жизни значения и источник данных. *Время жизни значения переменной* определяет промежуток времени, в течение которого это значение актуально, по истечении этого промежутка переменная считается не имеющей значения. В отличие от переменных параметры всегда имеют значение, так как их значения либо заданы в качестве начальных значений, либо перевычислены механизмом вывода G2.

Поскольку системе может потребоваться текущее значение переменной, для каждой из них должен быть определен *источник данных (сервер данных).* Источником данных для переменной могут служить: машина вывода, подсистема имитационного моделирования или внешний по отношению к G2 источник данных. С переменными могут быть ассоциированы формулы имитационного моделирования, в результате применения которых система также может получать значения переменных. Для параметров указанный механизм получения значений из ис-

точника данных не используется; они получают новые значения после выполнения специальных операторов в заключениях правил или процедур.

При ссылке в правиле или процедуре как для переменных, так и для параметров допустимо использование следующих выражений, отражающих динамику их значений:

• текущее значение;
• значение в заданный момент времени;
• среднее значение за интервал времени;
• интеграл по интервалу времени;
• интерполяция значения в заданный момент времени;
• максимальное (минимальное) значение за интервал времени;
• количество собранных значений за интервал времени;
• скорость изменения значений в течение интервала времени;
• стандартное отклонение в течение интервала времени.

Очевидно, что далеко не для всех используемых в приложении значений нужно применять такой мощный инструментарий, поэтому в целях повышения эффективности функционирования системы в этих случаях используют простые атрибуты.

Связи и отношения

В G2 предусмотрены два вида взаимосвязей между объектами: связи и отношения. Под *связями* понимается взаимосвязь между двумя сущностями, задаваемая разработчиком приложения и имеющая графическое представление. В реальном физическом окружении, описываемом в G2, связи может соответствовать физическая связь между сущностями, такая, как электрическое соединение или трубопровод. В G2 разработчик может задавать классы связей, ссылаться на объекты посредством указания связей, в которых они участвуют, а также делать заключения на основании наличия или отсутствия связей. Отношения, как и связи, представляют взаимосвязи между объектами. Под *отношением* понимается поименованная взаимосвязь между двумя сущностями. G2 предоставляет возможность разработчику задавать различные типы отношений. На основании наличия или отсутствия отношения между объектами могут производиться выводы. Основные отличия связей и отношений сводятся к следующему:

• связи задаются разработчиком в процессе создания ЭС, в то время как отношения устанавливаются динамически после выполнения специальных операторов в правилах или процедурах;

• связи имеют графическое представление, в то время как отношения не отображаются на экране дисплея;

• отношения в отличие от связей нецелесообразно сохранять в качестве постоянной части БЗ.

Исполняемые утверждения БЗ

Основу исполняемых утверждений БЗ составляют правила и процедуры. Кроме того, есть формулы, функции, действия и т.п. Правила в G2 имеют традиционный вид: условие (антецедент) и заключение (консеквент). Кроме if-правила: условие ("if <логическое выражение>") и заключение ("then <действия>") используются еще 4 типа правил: initially, unconditionally, when и whenever.

Способы применения каждого правила определяются его синтаксисом:

```
<правило> ::= {|<префикс for>|
                    {<правило if> | <правило unconditionally>
                    | <правило when> |        <правило whenever>}
                    | <правило initially>}
<префикс for> ::= for {any | the } <item> ...
<правило if> ::= if <логическое выражение> then <список действий>
<правило unconditionally> ::= unconditionally <список действий>
<правило when> ::= when <логическое выражение>
                    then <список действий>
<правило whenever> ::= whenever <описание события>
                    |or <описание события>|
                    |and when <логическое выражение>|
<правило initially> ::= initially
                    |if <логическое выражение> then| <список действий>
```

Каждый из типов правил может быть как *общим,* т.е. относящимся ко всему классу, так и *специализированным,* относящимся к конкретным экземплярам класса. Возможность представлять знания в виде общих, а не только конкретных правил, обеспечивает следующие преимущества:

- минимизируется избыточность БЗ;
- упрощается наполнение БЗ и ее сопровождение;
- минимизируются ошибки при отладке БЗ;
- способствует повторной используемости знаний (так как общие правила запоминаются в библиотеке G2 и могут использоваться в подобных приложениях).

Несмотря на то, что продукционные правила обеспечивают достаточную гибкость для описания реакций системы на изменения окружающего мира, в некоторых случаях, когда необходимо выполнить жесткую последовательность действий (например, запуск или остановку комплекса оборудования), более предпочтительным является процедурный подход. Язык программирования, используемый в G2 для представления процедурных знаний, является достаточно близким родственником Паскаля. Кроме стандартных управляющих конструкций язык расширен элементами, учитывающими работу процедуры в реальном времени: ожидание наступления событий, разрешение другим задачам прерывать выполнение данной процедуры, директивы, задающие последовательное или параллельное выполнение операторов. Еще одна интересная особенность языка - *итераторы,* позволяющие организовать цикл над множеством экземпляров класса. Перечисленные свойства языка позволяют системе одновременно выполнять множество различных процедур или множество копий одной и той же процедуры для множества различных объектов.

Контрольные вопросы:

1. Сущности и иерархия классов
2. Иерархия модулей и рабочих пространств
3. Структуры данных БЗ
4. Охарактеризуйте иерархию классов, иерархию модулей и иерархию рабочих пространств инструментального комплекса G2.
5. Приведите структуру данных в G2.
6. Опишите основные классы исполняемых утверждений G2.

Литература

1. *Кисель Е.Б.* **Сравнительный анализ инструментальных средств для разработки систем управления реального времени.** Материалы семинара "Экспертные системы реального времени". - М.: ЦРДЗ, 1995.

2. Копченова Н.В., Марон И.А. **Вычислительная математика в примерах и задачах.** - М.: Наука, 1972.

3. *Попов Э.В., Фоминых И.Б., Кисель Е.Б.* **Статические и динамические экспертные системы (классификация, состояние, тенденции). Методические материалы.** - М.: ЦРДЗ, 1995.

4. *Попов Э.В.* **Экспертные системы реального времени** //Открытые системы. 1995. - №2.

Лекция 16: Прикладные программы и алгоритмы для решения уравнений

План:

1. Центральная парадигма интеллектуальных технологий - обработка знаний
2. Современное состояние разработок в области ЭС
3. Применение ЭС

1. Центральная парадигма интеллектуальных технологий - обработка знаний

Центральная парадигма интеллектуальных технологий сегодня - это обработка знаний. Системы, ядром которых является база знаний или модель предметной области, описанная на языке сверхвысокого уровня, приближенном к естественному, называют интеллектуальными. Будем называть такой язык сверхвысокого уровня - языком представления знаний (ЯПЗ). Чаще всего интеллектуальные системы (ИС) применяются для решения сложных задач, где основная сложность решения связана с использованием слабо-формализованных знаний специалистов-практиков и где логическая (или смысловая) обработка информации превалирует над вычислительной. Например, понимание естественного языка, поддержка принятия решения в сложных ситуациях, постановка диагноза и рекомендации по методам лечения, анализ визуальной информации, управление диспетчерскими пультами и др.

Фактически сейчас прикладные интеллектуальные системы используются в десятках тысяч приложений. А годовой доход от продаж программных и аппаратных средств искусственного интеллекта еще в 1989 г. в США составлял 870 млн. долларов, а в 1990 г. - 1,1 млрд. долларов. В дальнейшем почти тридцати процентный прирост дохода сменился более плавным наращиванием темпов.

Наиболее распространенным видом ИС являются экспертные системы. Экспертные системы (ЭС) - это наиболее распространенный класс ИС, ориентированный на тиражирование опыта высококвалифицированных специалистов в областях, где качество принятия решений традиционно зависит от уровня экспертизы, например, медицина, юриспруденция, геология, экономика, военное дело и др. ЭС эффективны лишь в специфических "экспертных" областях, где важен эмпирический опыт специалистов. Только в США ежегодный доход от продаж инструментальных средств разработки ЭС составлял в начале 90-х годов 300-400 млн. долларов, а от применения ЭС -80-90 млн. долларов. Ежегодно крупные фирмы разрабатывают десятки ЭС типа "in-house" для внутреннего пользования. Эти системы интегрируют опыт специалистов компании по ключевым и стратегически важным технологиям. В начале 90-х гг. появилась новая наука - "менеджмент знаний" (knowledge management), ориентированная на методы обработки и управления корпоративными знаниями .

Современные ЭС - это сложные программные комплексы, аккумулирующие знания специалистов в конкретных предметных областях и распространяющие этот эмпирический опыт для консультирования менее квалифицированных пользователей. Разработка экспертных систем, как активно развивающаяся ветвь информатики, направлена на использование ЭВМ для обработки информации в тех областях науки и техники, где традиционные математические методы моделиро-

вания малопригодны. В этих областях важна смысловая и логическая обработка информации, важен опыт экспертов.

Приведем некоторые условия, которые могут свидетельствовать о необходимости разработки и внедрения экспертных систем: нехватка специалистов, затрачивающих значительное время для оказания помощи другим; выполнение небольшой задачи требует многочисленного коллектива специалистов, поскольку ни один из них не обладает достаточным знанием; сниженная производительность, поскольку задача требует полного анализа сложного набора условий, а обычный специалист не в состоянии просмотреть (за отведенное время) все эти условия; большое расхождение между решениями самых хороших и самых плохих исполнителей; наличие конкурентов, имеющих преимущество в силу того, что они лучше справляются с поставленной задачей.

Подходящие задачи имеют следующие характеристики: являются узкоспециализированными; не зависят в значительной степени от общечеловеческих знаний или соображений здравого смысла; не являются для эксперта ни слишком легкими, ни слишком сложными. (Время, необходимое эксперту для решения проблемы, может составлять от нескольких часов до нескольких недель.)

Экспертные системы достаточно молоды - первые системы такого рода, MYCIN и DENDRAL, появились в США в середине 70-х гг. В настоящее время в мире насчитывается несколько тысяч промышленных ЭС, которые дают советы: при управлении сложными диспетчерскими пультами, например сети распределения электроэнергии - Alarm Analyser при постановке медицинских диагнозов - ARAMIS, NEUREX, при поиске неисправностей в электронных приборах, диагностика отказов контрольно-измерительного оборудования - Intelligence Ware, Plant Diagnostics, по проектированию интегральных микросхем - DAA , QO, по управлению перевозками - AIRPLAN, по прогнозу военных действий - ANALYST , BATTLE по формированию портфеля инвестиций, оценке финансовых рисков - RAD, налогообложению -RUNE и т. д. Сейчас легче назвать области, где еще нет ЭС, чем те, где они уже применяются. Уже в 1987 г. опрос пользователей, проведенный журналом "Intelligent Technologies" (США), показан на (рис.4).

Главное отличие ИС и ЭС от других программных средств - это наличие базы знаний (БЗ), в которой знания хранятся в форме, понятной специалистам предметной области и могут быть изменены и дополнены также в понятной форме. Это и есть языки представления знаний - ЯПЗ.

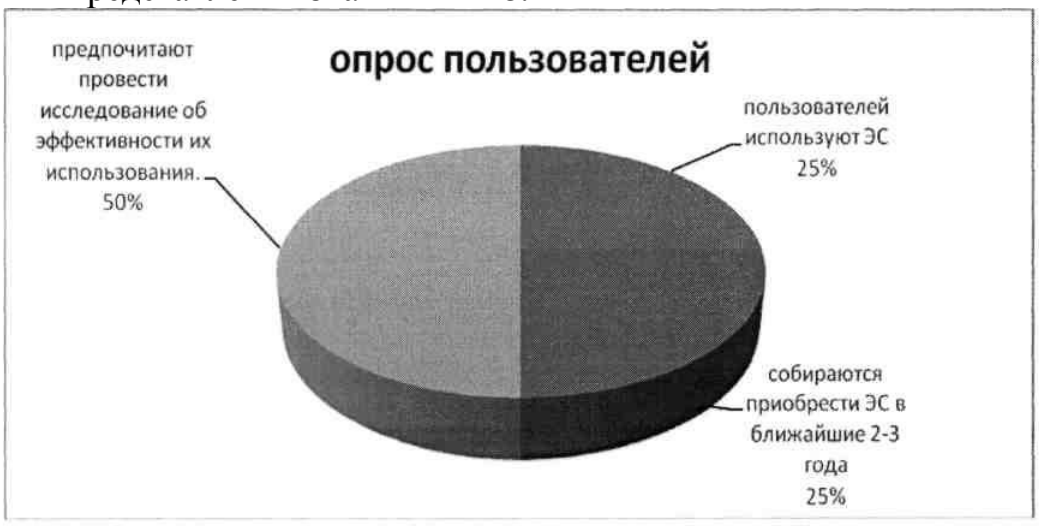


Рис. 4. Опрос пользователей по использованию ЭС

2. Современное состояние разработок в области ЭС

Современное состояние разработок в области ЭС в России можно охарактеризовать как стадию все возрастающего интереса среди широких слоев специалистов - финансистов, топ- менеджеров, преподавателей, инженеров, медиков, психологов, программистов, лингвистов. В последние годы этот интерес имеет пока достаточно слабое материальное подкрепление - явная нехватка учебников и специальной литературы, отсутствие символьных процессоров и рабочих станций, ограниченное финансирование исследований в этой области, слабый отечественный рынок программных продуктов для разработки ЭС.

Поэтому появляется возможность распространения "подделок" под экспертные системы в виде многочисленных диалоговых систем и интерактивных пакетов прикладных программ, которые дискредитируют в глазах пользователей это чрезвычайно перспективное направление. Процесс создания экспертной системы требует участия высококвалифицированных специалистов в области искусственного интеллекта, которых пока выпускает небольшое количество высших учебных заведений страны.

Наибольшие трудности в разработке ЭС вызывает сегодня не процесс машинной реализации систем, а домашинный этап анализа знаний и проектирования базы знаний. Этим занимается специальная наука - инженерия знаний.

В начале восьмидесятых годов в исследованиях по искусственному интеллекту сформировалось самостоятельное направление, получившее название "экспертные системы" (ЭС). Цель исследований по ЭС состоит в разработке программ, которые при решении задач, трудных для эксперта-человека, получают результаты, не уступающие по качеству и эффективности решениям, получаемым экспертом. Исследователи в области ЭС для названия своей дисциплины часто используют также термин "инженерия знаний", введенный Е.Фейгенбаумом как "привнесение принципов и инструментария исследований из области искусственного интеллекта в решение трудных прикладных проблем, требующих знаний экспертов".

Программные средства (ПС), базирующиеся на технологии экспертных систем, или инженерии знаний (в дальнейшем будем использовать их как синонимы), получили значительное распространение в мире. Важность экспертных систем состоит в следующем: технология экспертных систем существенно расширяет круг практически значимых задач, решаемых на компьютерах, решение которых приносит значительный экономический эффект; объединение технологии ЭС с технологией традиционного программирования добавляет новые качества к программным продуктам за счет: обеспечения динамичной модификации приложений пользователем, а не программистом; большей "прозрачности" приложения (например, знания хранятся на ограниченном ЕЯ, что не требует комментариев к знаниям, упрощает обучение и сопровождение); лучшей графики; интерфейса и взаимодействия.

3. Применение ЭС

По мнению ведущих специалистов, в недалекой перспективе ЭС найдут следующее применение:

- ЭС будут играть ведущую роль во всех фазах проектирования, разработки, производства, распределения, продажи, поддержки и оказания услуг;
- технология ЭС, получившая коммерческое распространение, обеспечит революционный прорыв в интеграции приложений из готовых интеллектуально-взаимодействующих модулей.
- высокая стоимость сопровождения сложных систем, которая часто в несколько раз превосходит стоимость их разработки; низкий уровень повторной используемости программ и т.п.;
- ЭС предназначены для так называемых неформализованных задач, т.е. ЭС не отвергают и не заменяют традиционного подхода к разработке программ, ориентированного на решение формализованных задач.

Неформализованные задачи обычно обладают следующими особенностями: ошибочностью, неоднозначностью, неполнотой и противоречивостью исходных данных; ошибочностью, неоднозначностью, неполнотой и противоречивостью знаний о проблемной области и решаемой задаче; большой размерностью пространства решения, т.е. перебор при поиске решения весьма велик; динамически изменяющимися данными и знаниями.

Следует подчеркнуть, что неформализованные задачи представляют большой и очень важный класс задач. Многие специалисты считают, что эти задачи являются наиболее массовым классом задач, решаемых ЭВМ.

Экспертные системы и системы искусственного интеллекта отличаются от систем обработки данных тем, что в них в основном используются символьный (а не числовой) способ представления, символьный вывод и эвристический поиск решения (а не исполнение известного алгоритма).

Экспертные системы применяются для решения только трудных практических (не игрушечных) задач. По качеству и эффективности решения экспертные системы не уступают решениям эксперта-человека. Решения экспертных систем обладают "прозрачностью", т.е. могут быть объяснены пользователю на качественном уровне. Это качество экспертных систем обеспечивается их способностью рассуждать о своих знаниях и умозаключениях. Экспертные системы способны пополнять свои знания в ходе взаимодействия с экспертом. Необходимо отметить, что в настоящее время технология экспертных систем используется для решения различных типов задач (интерпретация, предсказание, диагностика, планирование, конструирование, контроль, отладка, инструктаж, управление) в самых разнообразных проблемных областях, таких, как финансы, нефтяная и газовая промышленность, энергетика, транспорт, фармацевтическое производство, космос, металлургия, горное дело, химия, образование, целлюлозно-бумажная промышленность, телекоммуникации и связь и др.

Коммерческие успехи к фирмам-разработчикам систем искусственного интеллекта (СИИ) пришли не сразу. На протяжении 1960 - 1985 гг. успехи ИИ касались в основном исследовательских разработок, которые демонстрировали пригодность СИИ для практического использования. Начиная примерно с 1985 г. (в массовом масштабе с 1988 - 1990 гг.), в первую очередь ЭС, а в последние годы системы, воспринимающие естественный язык (ЕЯ-системы), и нейронные сети (НС) стали активно использоваться в коммерческих приложениях. Причины, при-

ведшие СИИ к коммерческому успеху, следующие. Разработаны инструментальные средства искусственного интеллекта (ИС ИИ), легко интегрирующиеся с другими информационными технологиями и средствами (с CASE, СУБД, контроллерами, концентраторами данных и т.п.). ИС ИИ разрабатываются с соблюдением стандартов, обеспечивающих открытость и переносимость. Использование языков традиционного программирования и рабочих станций. Переход от ИС ИИ, реализованных на языках ИИ (Lisp, Prolog и т.п.), к ИС ИИ, реализованным на языках традиционного программирования (C, C++ и т.п.), упростил обеспечение интегриро-ванности, снизил требования приложений ИИ к быстродействию ЭВМ и объемам оперативной памяти. Использование рабочих станций (вместо ПК) резко увеличило круг приложений, которые могут быть выполнены на ЭВМ с использованием ИС ИИ. Архитектура клиент-сервер. Разработаны ИС ИИ, поддерживающие распределенные вычисления по архитектуре клиент-сервер, что позволило: снизить стоимость оборудования, используемого в приложениях, децентрализовать приложения, повысить надежность и общую производительность (так как сокращается количество информации, пересылаемой между ЭВМ, и каждый модуль приложения выполняется на адекватном ему оборудовании).

Проблемно/предметно-ориентированные ИС ИИ. Переход от разработок ИС ИИ общего назначения (хотя они не утратили свое значение как средство для создания ориентированных ИС) к проблемно предметно-ориентированным ИС ИИ обеспечивает: сокращение сроков разработки приложений; увеличение эффективности использования ИС; упрощение и ускорение работы эксперта; повторную используемость информационного и программного обеспечения (объекты, классы, правила, процедуры).

Контрольные вопросы:

1. Прикладные программы и алгоритмы для решения уравнений?
2. Какова центральная парадигма интеллектуальных технологий?
3. Каково современное состояние разработок в области ЭС?
4. Области применения ЭС?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 17: База данных

План:

1. Направление в изучении искусственного интеллекта
2. Особенности перехода от Базы Данных к Базе Знаний
3. Методы представления знаний

1. Направление в изучении искусственного интеллекта

В настоящее время в исследованиях по искусственному интеллекту (ИИ) выделились шесть направлений представленных на (рис.3).

В рамках направления "Представление знаний" решаются задачи, связанные с формализацией и представлением знаний в памяти интеллектуальной системы (ИС). Для этого разрабатываются специальные модели представления знаний и языки для описания знаний, выделяются различные типы знаний. Изучаются источники, из которых ИС может черпать знания, и создаются процедуры и приемы, с помощью которых возможно приобретение знаний для ИС. Проблема представления знаний для ИС чрезвычайно актуальна, т.к. ИС - это система, функционирование которой опирается на знания о проблемной области, которые хранятся в ее памяти

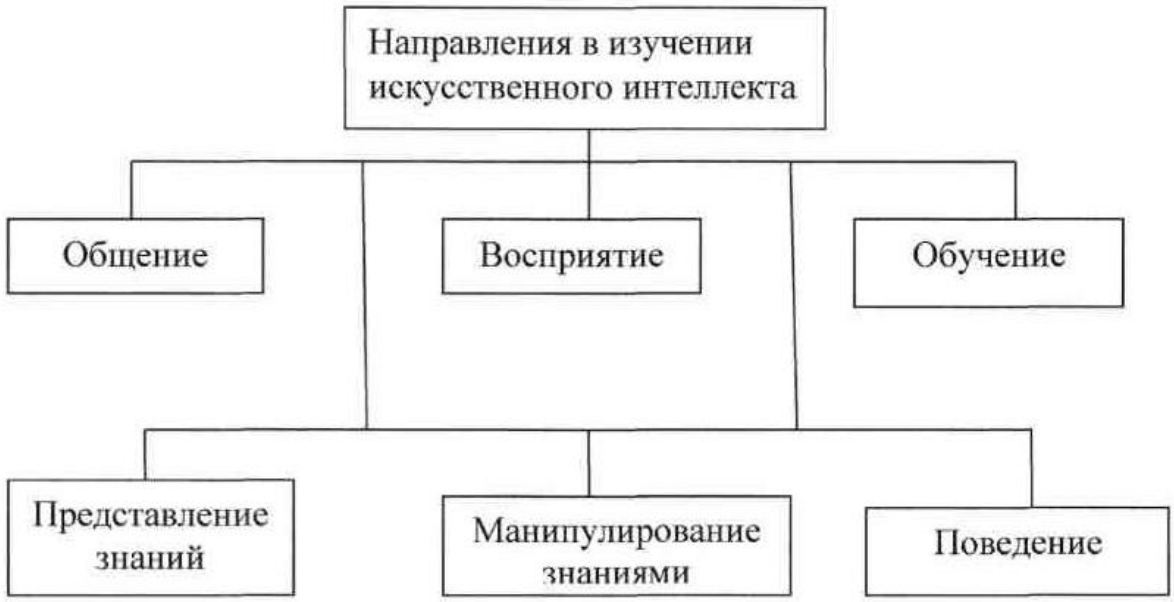


Рис. 3. Основные направления в исследовании ИИ

Информация, с которой имеют дело ЭВМ, разделяется на процедурную и декларативную. Процедурная информация овеществлена в программах, которые выполняются в процессе решения задач, декларативная информация -в данных, с которыми эти программы работают. Стандартной формой представления информации в ЭВМ является машинное слово, состоящее из определенного для данного типа ЭВМ числа двоичных разрядов - битов. Машинное слово для представления данных и машинное слово для представления команд, образующих программу, могут иметь одинаковое или разное число разрядов. В последнее время для представления данных и команд используются одинаковые по числу разрядов машинные слова. Однако в ряде случаев машинные слова разбиваются на группы по восемь двоичных разрядов, которые называются байтами.

Одинаковое число разрядов в машинных словах для команд и данных позволяет рассматривать их в ЭВМ в качестве одинаковых информационных единиц и выполнять операции над командами, как над данными. Содержимое памяти образует информационную базу.

В большинстве существующих ЭВМ возможно извлечение информации из любого подмножества разрядов машинного слова вплоть до одного бита. Во многих ЭВМ можно соединять два или более машинного слова в слово с большей длиной. Однако машинное слово является основной характеристикой информационной базы, т.к. его длина такова, что каждое машинное слово хранится в одной стандартной ячейке памяти, снабженной индивидуальным именем - адресом ячейки. По этому имени происходит извлечение информационных единиц из памяти ЭВМ и записи их в нее.

Параллельно с развитием структуры ЭВМ происходило развитие информационных структур для представления данных. Появились способы описания данных в виде векторов и матриц, возникли списочные структуры, иерархические структуры. В настоящее время в языках программирования высокого уровня используются абстрактные типы данных, структура которых задается программистом. Появление баз данных (БД) знаменовало собой еще один шаг на пути организации работы с декларативной информацией. В базах данных могут одновременно храниться большие объемы информации, а специальные средства, образующие систему управления базами данных (СУБД), позволяют эффективно манипулировать с данными, при необходимости извлекать их из базы данных и записывать их в нужном порядке в базу. По мере развития исследований в области ИС возникла концепция знаний, которые объединили в себе многие черты процедурной и декларативной информации.

В ЭВМ знания так же, как и данные, отображаются в знаковой форме -в виде формул, текста, файлов, информационных массивов и т.п. Поэтому можно сказать, что знания - это особым образом организованные данные. Но это было бы слишком узкое понимание. А между тем, в системах ИИ знания являются основным объектом формирования, обработки и исследования. База знаний, наравне с базой данных, - необходимая составляющая программного комплекса ИИ. Машины, реализующие алгоритмы ИИ, называются машинами, основанными на знаниях, а подраздел теории ИИ, связанный с построением экспертных систем, - инженерией знаний.

2. Особенности перехода от Базы Данных к Базе Знаний

При переходе от Базы Данных к Базе Знаний выделяют следующие особенности:

1. Внутренняя интерпретируемость. Каждая информационная единица должна иметь уникальное имя, по которому ИС находит ее, а также отвечает на запросы, в которых это имя упомянуто. Когда данные, хранящиеся в памяти, были лишены имен, то отсутствовала возможность их идентификации системой. Данные могла идентифицировать лишь программа, извлекающая их из памяти по указанию программиста, написавшего программу. Что скрывается за тем или иным двоичным кодом машинного слова, системе было неизвестно.

Если, например, в память ЭВМ нужно было записать сведения о сотрудниках учреждения, представленные в табл. 1.1, то без внутренней интерпретации в память ЭВМ была бы занесена совокупность из четырех машинных слов, соответствующих строкам этой таблицы.

Таблица 1.1 Данные о сотрудниках учреждения

Фамилия	Год рождения	Специальность	Стаж, число лет
Попов	1965	Слесарь	5

Сидоров	1946	Токарь	20
Иванов	1925	Токарь	30
Петров	1937	Сантехник	25

При этом информация о том, какими группами двоичных разрядов в этих машинных словах закодированы сведения о специалистах, у системы отсутствуют. Они известны лишь программисту, который использует данные табл. 1.1 для решения возникающих у него задач. Система не в состоянии ответить на вопросы типа "Что тебе известно о Петрове?" или "Есть ли среди специалистов сантехник?". При переходе к знаниям в память ЭВМ вводится информация о некоторой протоструктуре информационных единиц. В рассматриваемом примере она представляет собой специальное машинное слово, в котором указано, в каких разрядах хранятся сведения о фамилиях, годах рождения, специальностях и стажах. При этом должны быть заданы специальные словари, в которых перечислены имеющиеся в памяти системы фамилии, года рождения, специальности и продолжительности стажа. Все эти атрибуты могут играть роль имен для тех машинных слов, которые соответствуют строкам таблицы. По ним можно осуществлять поиск нужной информации. Каждая строка таблицы будет экземпляром протоструктуры. В настоящее время СУБД обеспечивают реализацию внутренней интерпретируемости всех информационных единиц, хранящихся в базе данных.

2. Структурированность. Информационные единицы должны обладать гибкой структурой. Для них должен выполняться "принцип матрешки", т.е. рекурсивная вложимость одних информационных единиц в другие. Каждая информационная единица может быть включена в состав любой другой, и из каждой информационной единицы можно выделить некоторые составляющие ее информационные единицы. Другими словами, должна существовать возможность произвольного установления между отдельными информационными единицами отношений типа "часть - целое", "род - вид" или "элемент -класс".
3. Связность. В информационной базе между информационными единицами должна быть предусмотрена возможность установления связей различного типа. Прежде всего эти связи могут характеризовать отношения между информационными единицами. Семантика отношений может носить декларативный или процедурный характер. Например, две или более информационные единицы могут быть связаны отношением "одновременно", две информационные единицы - отношением "причина - следствие" или отношением "быть рядом". Приведенные отношения характеризуют декларативные знания. Если между двумя информационными единицами установлено отношение "аргумент - функция", то оно характеризует процедурное знание, связанное с вычислением определенных функций. Далее будем различать отношения структуризации, функциональные отношения, каузальные отношения и семантические отношения. С помощью первых задаются иерархии информационных единиц, вторые несут процедурную информацию, позволяющую находить (вычислять) одни информационные единицы через другие, третьи задают причинно - следственные связи, четвертые соответствуют всем остальным отношениям.

Между информационными единицами могут устанавливаться и иные связи, например, определяющие порядок выбора информационных единиц из памяти или указывающие на то, что две информационные единицы несовместимы друг с другом в одном описании.

Перечисленные три особенности знаний позволяют ввести общую модель представления знаний, которую можно назвать семантической сетью, представляющей собой иерархическую сеть, в вершинах которой находятся информационные единицы. Эти единицы снабжены индивидуальными именами. Дуги семантической сети соответствуют различным связям между информационными единицами. При этом иерархические

связи определяются отношениями структуризации, а неиерархические связи - отношениями иных типов.

4. Семантическая метрика. На множестве информационных единиц в некоторых случаях полезно задавать отношение, характеризующее ситуационную близость информационных единиц, т.е. силу ассоциативной связи между информационными единицами. Его можно было бы назвать отношением релевантности для информационных единиц. Такое отношение дает возможность выделять в информационной базе некоторые типовые ситуации (например, "покупка", "регулирование движения на перекрестке"). Отношение релевантности при работе с информационными единицами позволяет находить знания, близкие к уже найденным.
5. Активность. С момента появления ЭВМ и разделения используемых в ней информационных единиц на данные и команды создалась ситуация, при которой данные пассивны, а команды активны. Все процессы, протекающие в ЭВМ, инициируются командами, а данные используются этими командами лишь в случае необходимости. Для ИС эта ситуация не приемлема. Как и у человека, в ИС актуализации тех или иных действий способствуют знания, имеющиеся в системе. Таким образом, выполнение программ в ИС должно инициироваться текущим состоянием информационной базы. Появление в базе фактов или описаний событий, установление связей может стать источником активности системы.

Перечисленные пять особенностей информационных единиц определяют ту грань, за которой данные превращаются в знания, а базы данных перерастают в базы знаний (БЗ). Совокупность средств, обеспечивающих работу с знаниями, образует систему управления базой знаний (СУБЗ). В настоящее время не существует баз знаний, в которых в полной мере были бы реализованы внутренняя интерпретируемость, структуризация, связность, введена семантическая мера и обеспечена активность знаний.

3. Методы представления знаний

Существуют два типа методов представления знаний (ПЗ): Формальные модели ПЗ и неформальные (семантические, реляционные) модели ПЗ.

Очевидно, все методы представления знаний, которые рассмотрены выше, включая продукции (это система правил, на которых основана продукционная модель представления знаний), относятся к неформальным моделям. В отличие от формальных моделей, в основе которых лежит строгая математическая теория, неформальные модели такой теории не придерживаются. Каждая неформальная модель годится только для конкретной предметной области и поэтому не обладает универсальностью, которая присуща моделям формальным. Логический вывод - основная операция в СИИ - в формальных системах строг и корректен, поскольку подчинен жестким аксиоматическим правилам. Вывод в неформальных системах во многом определяется самим исследователем, который и отвечает за его корректность. Каждому из методов ПЗ соответствует своя модель описания знаний.

1. Логические модели. В основе моделей такого типа лежит формальная система, задаваемая четверкой вида: $M = <T, P, A, B>$. Множество Г есть множество базовых элементов различной природы, например слов из некоторого ограниченного словаря, деталей детского конструктора, входящих в состав некоторого набора и т.п. Важно, что для множества *Т* существует некоторый способ определения принадлежности или непринадлежности произвольного элемента к этому множеству. Процедура такой проверки может быть любой, но за конечное число шагов она должна давать положительный или отрицательный ответ на вопрос, является ли х элементом множества *Т.* Обозначим эту процедуру *ЩТ).*

Множество Р есть множество синтаксических правил. С их помощью из элементов Т образуют синтаксически правильные совокупности. Например, из слов ограниченного словаря строятся синтаксически правильные фразы, из деталей детского конструктора с помощью гаек и болтов собираются новые конструкции. Декларируется существование процедуры П(Р), с помощью которой за конечное число шагов можно получить ответ на вопрос, является ли совокупность Х синтаксически правильной.

В множестве синтаксически правильных совокупностей выделяется некоторое подмножество А. Элементы А называются аксиомами. Как и для других составляющих формальной системы, должна существовать процедура П(А), с помощью которой для любой синтаксически правильной совокупности можно получить ответ на вопрос о принадлежности ее к множеству А.

Множество В есть множество правил вывода. Применяя их к элементам А, можно получать новые синтаксически правильные совокупности, к которым снова можно применять правила из В. Так формируется множество выводимых в данной формальной системе совокупностей. Если имеется процедура П(В), с помощью которой можно определить для любой синтаксически правильной совокупности, является ли она выводимой, то соответствующая формальная система называется разрешимой. Это показывает, что именно правило вывода является наиболее сложной составляющей формальной системы.

Для знаний, входящих в базу знаний, можно считать, что множество А образуют все информационные единицы, которые введены в базу знаний извне, а с помощью правил вывода из них выводятся новые производные знания. Другими словами формальная система представляет собой генератор
порождения новых знаний, образующих множество выводимых в данной системе знаний. Это свойство логических моделей делает их притягательными для использования в базах знаний. Оно позволяет хранить в базе лишь те знания, которые образуют множество А, а все остальные знания получать из них по правилам вывода.

2. Сетевые модели. В основе моделей этого типа лежит конструкция, названная ранее семантической сетью. Сетевые модели формально можно задать в виде $H = <1, C_ь\ Сг, .., C_п, Г>$. Здесь I есть множество информацион ных единиц; $C_ь\ C_2, ..., C_п$ - множество типов связей между информационны ми единицами. Отображение Г задает между информационными единицами, входящими в I, связи из заданного набора типов связей.

В зависимости от типов связей, используемых в модели, различают классифицирующие сети, функциональные сети и сценарии. В классифицирующих сетях используются отношения структуризации. Такие сети позволяют в базах знаний вводить разные иерархические отношения между информационными единицами. Функциональные сети характеризуются наличием функциональных отношений. Их часто называют вычислительными моделями, т.к. они позволяют описывать процедуры "вычислений" одних информационных единиц через другие. В сценариях используются каузальные отношения, а также отношения типов "средство - результат", "орудие -действие" и т.п. Если в сетевой модели допускаются связи различного типа, то ее обычно называют семантической сетью.

3. Продукционные модели. В моделях этого типа используются неко торые элементы логических и сетевых моделей. Из логических моделей за имствована идея правил вывода, которые здесь называются продукциями, а из сетевых моделей - описание знаний в виде семантической сети. В резуль тате применения правил вывода к фрагментам сетевого описания происходит трансформация семантической сети за счет смены ее фрагментов, наращива ния сети и исключения из нее ненужных фрагментов. Таким образом, в про дукционных моделях процедурная информация явно выделена и описывается

иными средствами, чем декларативная информация. Вместо логического вывода, характерного для логических моделей, в продукционных моделях появляется вывод на знаниях.

4. Фреймовые модели. В отличие от моделей других типов во фреймовых моделях фиксируется жесткая структура информационных единиц, которая называется протофреймом. В общем виде она выглядит следующим образом:

(Имя фрейма:
Имя слота 1 (значение слота 1)
Имя слота 2(значение слота 2)
Имя слота К (значение слота К)).

Значением слота может быть практически что угодно (числа или математические соотношения, тексты на естественном языке или программы, правила вывода или ссылки на другие слоты данного фрейма или других фреймов). В качестве значения слота может выступать набор слотов более низкого уровня, что позволяет во фреймовых представлениях реализовать "принцип матрешки".

При конкретизации фрейма ему и слотам присваиваются конкретные имена и происходит заполнение слотов. Таким образом, из протофреймов получаются фреймы - экземпляры. Переход от исходного протофрейма к фрейму - экземпляру может быть многошаговым, за счет постепенного уточнения значений слотов.

Например, структура табл. 1.1, записанная в виде протофрейма, имеет вид

(Список работников:
Фамилия (значение слота 1);
Год рождения (значение слота 2);
Специальность (значение слота 3);
Стаж (значение слота 4)).

Если в качестве значений слотов использовать данные табл. 1.1, то получится фрейм - экземпляр

(Список работников:
Фамилия (Попов - Сидоров - Иванов - Петров);
Год рождения (1965 - 1946 - 1925 - 1937);
Специальность (слесарь - токарь - токарь - сантехник);
Стаж (5 - 20 - 30 - 25)).

Связи между фреймами задаются значениями специального слота с именем "Связь". Часть специалистов по ИС считает, что нет необходимости специально выделять фреймовые модели в представлении знаний, т.к. в них объединены все основные особенности моделей остальных типов.

Система ИИ в определенном смысле моделирует интеллектуальную деятельность человека и, в частности, - логику его рассуждений. В грубо упрощенной форме наши логические построения при этом сводятся к следующей схеме: из одной или нескольких посылок (которые считаются истинными) следует сделать "логически верное" заключение (вывод, следствие). Очевидно, для этого необходимо, чтобы и посылки, и заключение были представлены на понятном языке, адекватно отражающем предметную область, в которой проводится вывод. В обычной жизни это наш естественный язык общения, в математике, например, это язык определенных формул и т.п. Наличие же языка предполагает, во - первых, наличие алфавита (словаря), отображающего в символьной форме весь набор базовых понятий (элементов), с которыми придется иметь дело и, во - вторых, набор синтаксических правил, на основе которых, пользуясь алфавитом, можно построить определенные выражения.

Логические выражения, построенные в данном языке, могут быть истинными или ложными. Некоторые из этих выражений, являющиеся всегда истинными. Объявляются аксиомами (или постулатами). Они составляют ту базовую систему посылок, исходя из

которой и пользуясь определенными правилами вывода, можно получить заключения в виде новых выражений, также являющихся истинными.

Если перечисленные условия выполняются, то говорят, что система удовлетворяет требованиям формальной теории. Ее так и называют формальной системой (ФС). Система, построенная на основе формальной теории, называется также аксиоматической системой.

Формальная теория должна, таким образом, удовлетворять следующему определению: всякая формальная теория F = (A, V, W, R), определяющая некоторую аксиоматическую систему, характеризуется:

- наличием алфавита (словаря), A;
- множеством синтаксических правил, V;
- множеством аксиом, лежащих в основе теории, W;
- множеством правил вывода, R.

Исчисление высказываний (ИВ) и исчисление предикатов (ИП) являются классическими примерами аксиоматических систем. Эти ФС хорошо исследованы и имеют прекрасно разработанные модели логического вывода -главной метапроцедуры в интеллектуальных системах. Поэтому все, что может и гарантирует каждая из этих систем, гарантируется и для прикладных ФС как моделей конкретных предметных областей. В частности, это гарантии непротиворечивости вывода, алгоритмической разрешимости (для исчисления высказываний) и полуразрешимости (для исчислений предикатов первого порядка).

ФС имеют и недостатки, которые заставляют искать иные формы представления. Главный недостаток - это "закрытость" ФС, их негибкость. Модификация и расширение здесь всегда связаны с перестройкой всей ФС, что для практических систем сложно и трудоемко. В них очень сложно учитывать происходящие изменения. Поэтому ФС как модели представления знаний используются в тех предметных областях, которые хорошо локализуются и мало зависят от внешних факторов.

Центральная парадигма интеллектуальных технологий сегодня - это обработка знаний. Системы, ядром которых является база знаний или модель предметной области, описанная на языке сверхвысокого уровня, приближенном к естественному, называют интеллектуальными. Будем называть такой язык сверхвысокого уровня - языком представления знаний (ЯПЗ). Чаще всего интеллектуальные системы (ИС) применяются для решения сложных задач, где основная сложность решения связана с использованием слабо-формализованных знаний специалистов-практиков и где логическая (или смысловая) обработка информации превалирует над вычислительной. Например, понимание естественного языка, поддержка принятия решения в сложных ситуациях, постановка диагноза и рекомендации по методам лечения, анализ визуальной информации, управление диспетчерскими пультами и
др.

Фактически сейчас прикладные интеллектуальные системы используются в десятках тысяч приложений. А годовой доход от продаж программных и аппаратных средств искусственного интеллекта еще в 1989 г. в США составлял 870 млн. долларов, а в 1990 г. - 1,1 млрд. долларов. В дальнейшем почти тридцати процентный прирост дохода сменился более плавным наращиванием темпов.

Контрольные вопросы:

1. Направление в изучении искусственного интеллекта?
2. Особенности перехода от Базы Данных к Базе Знаний?
3. Методы представления знаний?

Литература

1. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
2. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
3. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 18: Этапы решения задач: построения абстрактных программ для решения

План:

1 Принятие решения

Принятие решения включает формирование промежуточных целей для выполнения поставленной задачи.

Но все вышесказанное не означает, что роль человека будет состоять только в том, чтобы наслаждаться работой робототехнических систем, пребывая в полной бездеятельности. Напротив его ответственность возрастает и потребуется колоссальная нагрузка на человека, чтобы управлять сложными системами, создавать новые механизмы и обезопасить себя от любых техногенных катастроф.

Современная быстродействующая вычислительная техника позволила качественно изменить структуру технологического оборудования. **Во-первых**, благодаря высокому быстродействию вычислений появилась возможность осуществлять управление механизмами, в которых перемещения не совпадают с координатами обрабатываемой детали. Например, высоко скоростные прямолинейные перемещения можно выполнять с помощью вращательных пар. **Во-вторых**, быстродействующие средства контроля дали возможность построить системы оперативной настройки режимов обработки, получая информацию об обрабатываемой поверхности.

В *робототехнике* системы *очувствления* и *искусственного интеллекта* нашли достаточно широкое применение. Следует выделить следующие направления развития интеллектуальных роботов:

1. Промышленные роботы, работающие в производственной сфере и заменяющие человека при выполнении технологических операций. Интеллект указанных роботов заключается в их способности автоматически распознавать качество обработанной поверхности, контролировать режимы обработки и корректировать их в зависимости от поставленной цели, например, минимизировать погрешности, уменьшать энергозатраты, выбирать технологию обработки в зависимости от типа детали и требований к ее выходным характеристикам. В настоящее время это, пожалуй, основной класс роботов, которому должно быть уделено особое внимание, так как замена человека в сфере производства качественно изменит его жизнедеятельность
2. Безусловно к сугубо интеллектуальным роботам следует отнести робото-тележки, перемещающиеся по космическим планетам в условиях непредсказуемой обстановки и выполняющие операции сбора информации о местности, на основе которой они определяют направление своего движения.

3. Игровые роботы, предназначенные для тренировки спортсменов. Роботы, играющие в гольф, теннис, шахматы, соревнующиеся друг с другом, на первый взгляд указанные роботы не предназначены для замены человека на производстве. Однако, как и в человеческой деятельности, при выполнении игровых задач отрабатывается структура искусственных машин, их силовые возможности, быстродействие и интеллектуальные способности.

4. Специальные роботы, способные работать в военной обстановке, а также в условиях особо опасных для жизнедеятельности человека.

ИРС для выполнения производственных задач, так называемые *роботы-станки*, являются устройствами, которые полностью автоматизируют производство по выпуску определенного вида продукции. Данное оборудование оснащается системами контроля технологических и выходных параметров обрабатываемого изделия.

В станочном оборудовании предъявляются достаточно высокое требования к точности, надежности и ответственности выполняемой операции. При выполнении операций обработки и сборки сложных деталей невозможно требовать вероятностного результата. Как правило, такие операции строго детерминированы. Поэтому вероятностные поисковые методы возможны только на стадии обработки результатов. Принятие окончательного решения должно обеспечивать детерминированный результат, обеспечивающий поставленную цель.

Особенно высокие требования предъявляются при обработке поверхностей сложной формы. В этом случае необходимо более точное выполнение режимов обработки, контроль износа инструмента в процессе обработки и обеспечение одновременно нескольких параметров детали. В частности, для каждой точки поверхности нужно одновременно обеспечивать до шести геометрических параметров, не считая качества поверхности. Для сложных поверхностей, кроме требований к самим координатам, накладываются условия и на их производные.

Для соблюдения высоких требований к точности изготовления деталей необходимо осуществлять постоянный контроль геометрических параметров станка, размеров звеньев, температурных изменений и других параметров. Применение механизмов параллельной структуры также качественно меняет подход к проектированию станочного робототехнического оборудования.

Понятие *робот-станок* было введено в 1992 году при описании станочного оборудования, построенного на механизмах параллельной структуры и позволяющего посредством одного и того же механизма выполнять транспортные операции и операции обработки. Данные механизмы позволяют расширить функциональные возможности станочного оборудования и при наличии системы управления, оснащенной элементами *искусственного интеллекта*, делает данное оборудование близким к интеллектуальным роботам.

Совмещение функций особенно актуально для сложных высокоточных операций, когда требуется обработка детали от одной базы. В данном случае получаем универсальное оборудование, позволяющее выполнять несколько различных технологических операций для широкой номенклатуры изделий.

Главной отличительной особенностью *робота-станка* от обрабатывающего центра является универсальность, точнее, более богатые кинематические возмож-

ности перемещения механизма. Безусловно, из набора *роботов-станков* можно построить распределенный обрабатывающий центр. Механизмы параллельной структуры расширили возможности исполнительных механизмов станков, сделали их более облегченными и универсальными. Наличие параллельных кинематических цепей позволяет управлять одним выходным звеном по нескольким параллельным каналам, обеспечивая одновременное управление по положению, скорости, более высоким производным, а также по силе. В работе приведены, хотя и не в полном объеме, механизмы параллельной структуры, которые с успехом можно применить в станочном оборудовании.

Последующие лекции ставят своей целью показать место интеллектуальных систем в сфере промышленной *робототехники*. При этом роботы представляются в виде технологических систем, непосредственно выполняющих операцию обработки. Мы их называем *роботами-станками*, так как их кинематическая схема позволяет выполнять транспортные операции и непосредственно обработку. Применение механизмов параллельной структуры уже на низшем уровне позволяет расширить интеллектуальные возможности технологических машин.

2 Структура и состав интеллектуальной робототехнической системы.

Интеллектуальная робототехническая система включает объект управления совместно со средой, в которой она работает. Объект управления представляет непосредственно механизмы перемещения инструмента и изделия. В состав манипуляторов входят исполнительные двигатели, которые осуществляют их перемещение по заданным законам $R_Д$ и $R_И$. Информация о положении выходных звеньев манипуляторов определяется датчиками, расположенными в шарнирах звеньев манипуляторов, которые получают информации о выходных координатах механизмов перемещения, их скоростях, ускорениях и силах. Основная функция системы управления манипуляторами состоит в формировании законов перемещения исполнительными механизмами манипуляторов в реальном времени $U_И(t)$ и $U_Д(t)$. Данные системы обычно работают в следящем режиме, обеспечивающем выполнение каждой степенью подвижности манипуляторов заданной траектории перемещения с требуемыми точностью, скоростью и усилием. Выходными координатами манипуляторов являются $R_Д$ и $R_И$. В результате взаимодействия инструмента с деталью создается усилие $P(t)$, которое воздействует на исполнительные органы манипуляторов. Применительно к рассматриваемой системе в качестве объекта управления и внешней среды следует рассматривать манипуляторы перемещения изделия, инструмента и непосредственно сам технологический процесс.

На рисунке 8.1 не раскрывается состав подсистемы управления высшего уровня. Ее структура и выполняемые функции подробно описаны в лекции 1. Общим информационным управляющим каналом на систему управления низшего уровня является канал передачи управляющих сигналов $U(t)$ и обратной связью от системы низшего уровня - сигнал $R(t)$. Управляющее воздействие $U(t)$ представляет выбранную программу действия из некоторого множества $U \in U(t)$ и соответствующую заданной детали, либо обработке заданной поверхности детали. Какую из программ следует выбрать, решается системой высшего уровня как на основе информации от системы распознавания поверхности, так и

на основе указаний оператора, управляющего робототехнической системой. Выбранная программа U(t) задается непрерывно в реальном масштабе времени.

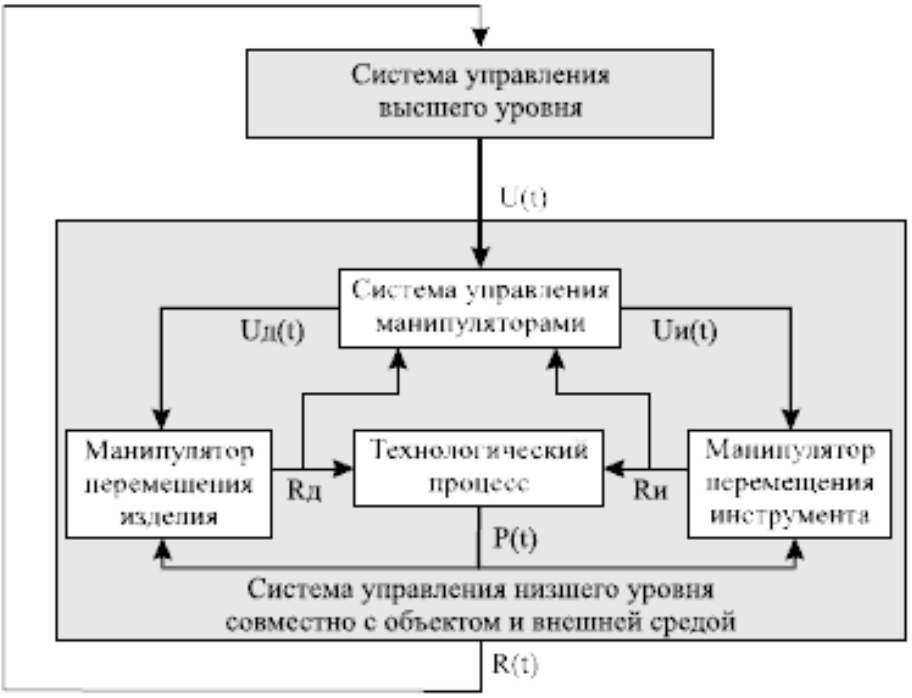


Рис. 8.1. Структура робототехнической системы

Обратная связь R(t) может нести полную информацию о работе системы управления низшего уровня в виде логических сигналов о ее состоянии, непрерывную информацию о геометрических размерах, качестве обработки поверхности детали и информацию о состоянии внешней среды, например, о температуре окружающей среды или двигателей, о состоянии сопутствующих обработке других устройств.

Представленная на рисунке 8.1 система является обобщенной для технологических машин широкого назначения. Более детальное представление данной системы рассмотрим на примере системы управления робота-станка (рис. 8.2). Отличительной особенностью рассматриваемой следящей системы управления от существующих станочных систем является наличие главной обратной связи по результату обработки поверхности (вычисление $^{д}A_i^*(t)$).

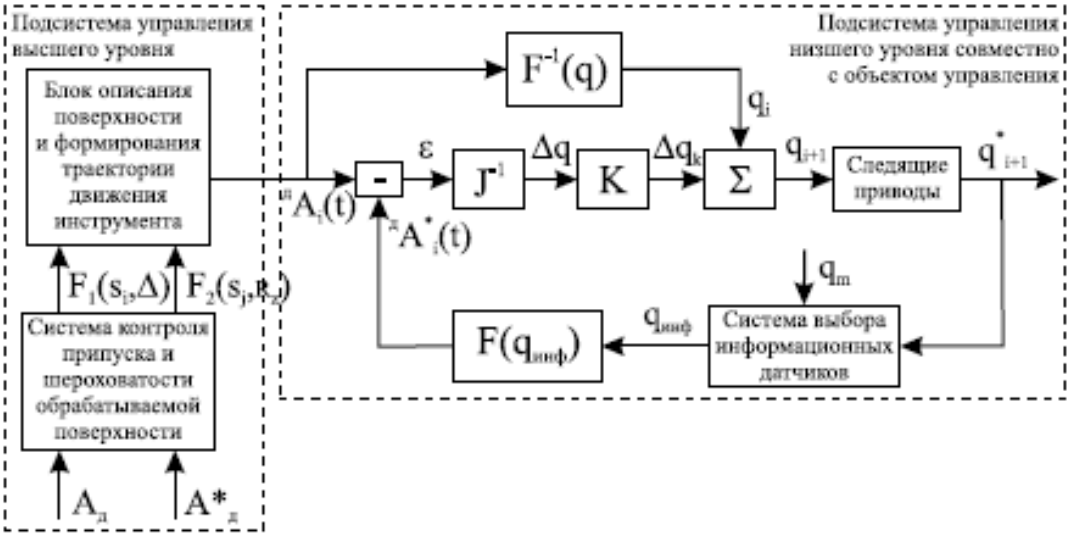


Рис. 8.2. Система управления робота-станка.

Вычисление $^{Д}A_i^*(t)$ осуществляется в системе координат детали решением прямой задачи о положении $F(q_{инф.})$ по информации датчиков, располагаемых в сочленениях звеньев механизма. Погрешность ε вычисляется сравнением программного значения управляющего воздействия $^{Д}A_i(t)$ и вычисленного реального его значения $^{Д}A_i^*(t)$. Обратный Якобиан J^{-1} и устройство K выполняют функции преобразования и решения линейной задачи вычисления приращений обобщенных координат q_i. Суммируя приращения на каждом шаге вычисления с предыдущим значением, формируется управляющее воздействие на исполнительные приводы q_i.

В качестве электродвигателей приводов манипуляторов применяются безредукторные и высокомоментные электродвигатели. Это требует применения методики синтеза приводов с учетом переменности моментов инерции, а для многостепенной механической системы требуется также учитывать взаимовлияние приводов по степеням подвижности.

Подсистема управления высшего уровня выполняет следующие функции. Получая информацию от оптической системы о состоянии обрабатываемой поверхности и ее геометрических размерах, данная подсистема выбирает требуемую программу обработки из некоторого детерминированного множества программ либо при ее отсутствии на основе анализа принимает наиболее близкую по критерию точности воспроизведения требуемой поверхности.

Оптические средства контроля геометрических размеров припуска и качества обработки (шероховатости) поверхности детали позволяют оптимизировать режимы резания. В работе приведено описание оптической системы, построенной с применением специальной решетки и источника монохроматического света. В настоящем курсе лекций дается описание данной системы, рассматриваются вопросы построения системы распознавания зон с заданным качеством обработки и формирования на этой основе новой программы обработки поверхности.

Формирование программной траектории перемещения инструмента относительно обрабатываемой поверхности $^{Д}A_i(t)$, производится на основе информации, полученной от оптической системы контроля поверхности и экспертной оценки при выборе режимов обработки. (В лекции 7 был рассмотрен пример выбора режимов и программы обработки в среде CLIPS). Информация о геометрических размерах полученной после обработки поверхности контролируется оптической системой контроля. Эта система формирует также данные о качестве обрабатываемой поверхности. В зависимости от этой информации выбирается ограниченная область обработки поверхности.

Математическая модель объекта управления совместно с окружающей средой, формируемая в системе высшего уровня на основе информации, получаемой от датчиков, включает: чертеж детали с реальными геометрическими размерами, чертеж требуемой идеальной детали и набор параметров, определяющих режимы обработки. Указанная модель позволяет, проигрывая различные ситуации, представляющие набор процедур для выполнения обработки, выбирать цель и формировать программу обработки $U(t)$.

3 Пример робота-станка

Пример робота-станка, построенного на механизмах параллельной структуры и оснащенного интеллектуальной системой обработки информации и управления, показан на рисунке 8.3. Исполнительный механизм робота-станка включает манипулятор перемещения изделия, представляющий пятизвенник, состоящий из звеньев 10, 11, 12, 13 и основания, манипулятор перемещения инструмента, который представляет собой два звена, управляемых двигателями 1 и 4 с вертикальной осью вращения. Манипулятор перемещения изделия осуществляет управляемое перемещение по четырем координатам с помощью четырех исполнительных приводов 2, 3, 8 и 14. Обработка выполняется путем взаимного перемещения инструмента 6 относительно изделия 9. Для стабилизации и удержания веса манипулятора перемещения изделия применено пневматическое устройство 16. Бабка для вращения изделия 15 и бабка для инструмента 5 содержат исполнительные приводы для вращения инструмента 7 и изделия 8. В целом механизм относительного перемещения робота-станка позволяет выполнять взаимное перемещение инструмента и изделия по шести координатам.

В механизмах параллельной структуры имеются кинематические пары, которые выполняют функции преобразования движения и не содержат исполнительных силовых элементов (пятизвенник в манипуляторе перемещения изделия). В сочленениях данных пар возможна установка дополнительных датчиков, позволяющих повысить точность контроля положения выходного звена. Кроме того, установка в этих сочленениях дополнительных приводов, управляемых, к примеру, по силе, разгружает основные приводы, выполняющие перемещения по заданным координатам, и позволяет по одной и той же координате управлять положением, скоростью и силой.

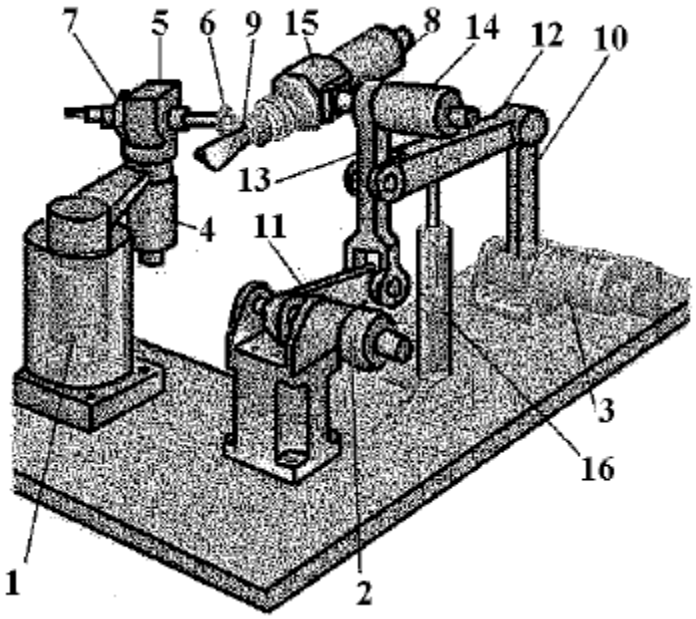


Рис. 8.3. Робот-станок

Пример кинематической схемы робота-станка приведен для лучшего понимания работы реальной интеллектуальной робототехнической системы, чтобы показать место установки датчиков и дополнительных приводов в механизме.

В рассматриваемом курсе лекций мы не рассматриваем вопросы работы **мехатронных** элементов в составе интеллектуальной робототехнической системы. Безусловно, аппаратная часть системы управления робота-станка содержит мехатронные элементы. Это непосредственно оптическая система, которая включает механические элементы преобразования оптического изображения и цифровую систему обработки изображения. Встраиваемые исполнительные приводы совместно с датчиками положения также представляют мехатронные системы восприятия и преобразования информации.

Контрольные вопросы:

1 Этапы решения задач: построения абстрактных программ для решения?

2 Принятие решения?

3 Структура и состав интеллектуальной робототехнической системы?

4 Пример робота-станка?

Литература

4. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
5. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
6. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 19: Перевод задачи на машинный язык; трансляция и выполнения программ.

План:

1. Распознавание изображений
2. Преобразование изображений в цифровой код
3. Экспертные системы
4. Машинный перевод и понимание текстов на естественном языке

Для решения трудно формализуемых задач и, в частности, для работы со знаниями были созданы языки программирования для задач ИИ: LISP (1958 год, J. MacCatthy), Пролог (1975-79 годы, D. Warren, F. Pereira), ИнтерLISP, FRL, KRL, SMALLTALK, OPS5, PLANNER, QA4, MACSYMA, REDUCE, РЕФАЛ, CLIPS. К числу наиболее популярных традиционных языков программирования для создания ИС следует также отнести C++ и Паскаль.

1. Распознавание изображений

Рождение робототехники выдвинуло задачи машинного зрения и распознавания изображений в число первоочередных.

В традиционном распознавании образов появился хорошо разработанный математический аппарат, и для не очень сложных объектов оказалось возможным строить практически работающие системы классификации по признакам, по аналогии и т. д. В качестве признаков могут рассматриваться любые характеристики распознаваемых объектов. Признаки должны быть инвариантны к ориентации, размеру и вариациям формы объектов. Алфавит признаков придумывается разработчиком системы. Качество распознавания во многом зависит от того, насколько удачно придуман алфавит признаков. Распознавание состоит в априорном получении вектора признаков для выделенного на изображении отдельного распознаваемого объекта, и лишь затем в определении того, какому из эталонов этот вектор соответствует.

П. Уинстон в начале 80-х годов обратил внимание на необходимость реализации целенаправленного процесса машинного восприятия. Цель должна управлять работой всех процедур, в том числе и процедур нижнего уровня, т. е. процедур предварительной обработки и выделения признаков. Должна иметься возможность на любой стадии процесса в зависимости от получаемого результата возвращаться к его началу для уточнения результатов работы процедур предшествующих уровней. У П. Уинстона, так же как и у других исследователей, до решения практических задач дело не дошло, хотя в 80-е годы вычислительные мощности больших машин позволяли начать решение подобных задач. Таким образом, ранние традиционные системы распознавания, основывающиеся на последовательной организации процесса распознавания и классификации объектов, эффективно решать задачи восприятия сложной зрительной информации не могли.

2. Преобразование изображений в цифровой код

Для того чтобы ввести изображение в машину, нужно перевести его на машинный язык, т.е. закодировать, представить в виде некоторой комбинации символов, которыми может оперировать машина. Кодирование плоских фигур можно осуществить самым различным образом. Лучше стремиться к наиболее " естественному" кодированию изображений. Будем рисовать фигуры на некотором поле, разбитом вертикальными и горизонтальными прямыми на одинаковые элементы - квадратики. Элементы, на которые упало изображение, будем сплошь зачернять, остальные - оставлять белыми. Условимся обозначать черные элементы единицей, белые - нулем. Введем последовательную нумерацию всех элементов поля, например, в каждой строке слева направо и по строкам сверху вниз. Тогда каждая фигура, нарисованная на таком поле, будет однозначно отображаться кодом, состоящим из стольких цифр (единиц и нулей), сколько элементов содержит поле.

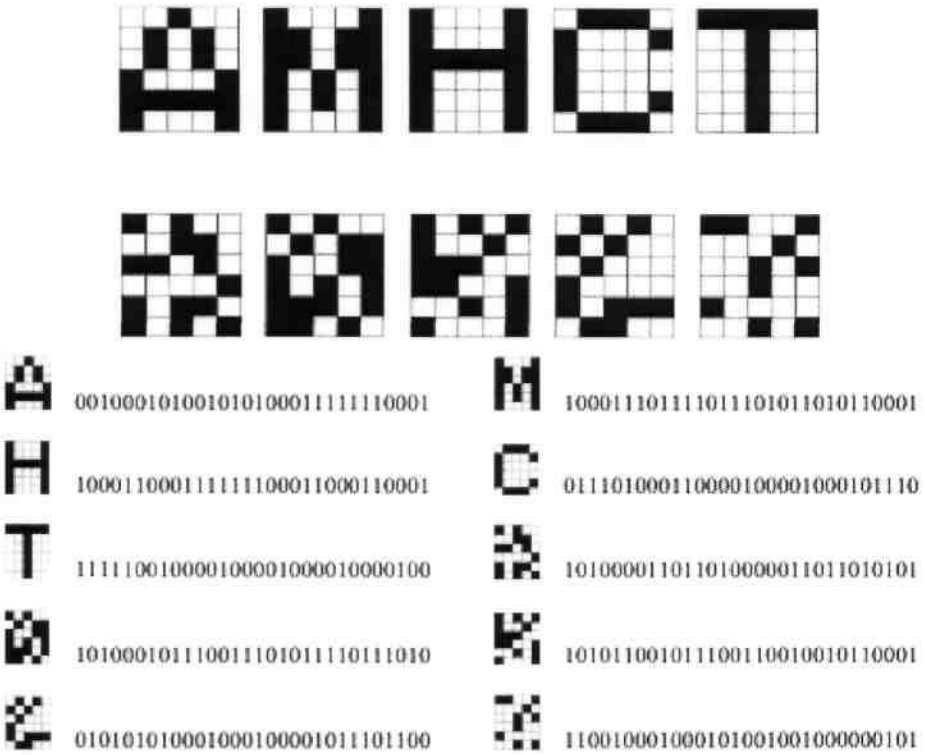


Рис. 7. Примеры проецирования и кодирования изображений.

Такое кодирование (рис. 7). считается " естественным" потому, что разбиение изображения на элементы лежит в основе работы нашего зрительного аппарата. Действительно, сетчатка глаза состоит из большого числа отдельных чувствительных элементов (так называемых палочек и колбочек), связанных нервными волокнами со зрительными отделами головного мозга. Чувствительные элементы сетчатки передают по своим нервным волокнам в головной мозг сигналы, интенсивность которых зависит от освещенности данного элемента. Таким образом, изображение, спроектированное оптической системой глаза на сетчатку, разбивается палочками и колбочками на отдельные участки, и по элементам в некотором коде передается в мозг. Отдельные элементы поля называются рецепторами, а само поле - полем рецепторов.

Совокупность всех плоских фигур, которые можно изобразить на поле рецепторов, составляет некое множество. Каждая конкретная фигура из этой совокупности есть объект этого множества. Любому их таких объектов соответствует определенный код. Точно также любому коду соответствует определенное изоб-

ражение на поле рецепторов. Взаимно однозначное соответствие между кодами и изображениями позволит оперировать только кодами, помня о том, что изображение всегда может быть воспроизведено по его коду.

Емкость ИНС - число образов, предъявляемых на входы ИНС для распознавания. Для разделения множества входных образов, например, по двум классам достаточно всего одного выхода. При этом каждый логический уровень - "1" и "О" - будет обозначать отдельный класс. На двух выходах можно закодировать уже 4 класса и так далее. Для повышения достоверности классификации желательно ввести избыточность путем выделения каждому классу одного нейрона в выходном слое или, что еще лучше, нескольких, каждый из которых обучается определять принадлежность образа к классу со своей степенью достоверности, например: высокой, средней и низкой. Такие ИНС позволяют проводить классификацию входных образов, объединенных в нечеткие (размытые или пересекающиеся) множества. Это свойство приближает подобные ИНС к условиям реальной жизни.

3. Экспертные системы

Методы ИИ нашли применение при создании автоматических консультирующих систем. До 1968 года исследователи в области ИИ работали на основе общего подхода - упрощения комбинаторики, базирующегося на уменьшении перебора альтернатив исходя из здравого смысла, применения числовых функций оценивания и различных эвристик.

В начале 70-х годов произошел качественный скачок и пришло понимание, что необходимы глубокие знания в соответствующей области и выделение знаний из данных, получаемых от эксперта. Появляются экспертные системы (ЭС), или системы, основанные на знаниях.

ЭС DENDRAL (середина 60-х годов, Стэнфордский университет) расшифровывала данные масс-спектрографического анализа.

ЭС MYCIN (середина 70-х годов, Стэнфордский университет) ставила диагноз при инфекционных заболеваниях крови.

ЭС PROSPECTOR (1974-1983 годы, Стэнфордский университет) обнаруживала полезные ископаемые.

ЭС SOPHIE обучала диагностированию неисправностей в электрических цепях. ЭС XCON помогала конфигурировать оборудование для систем VAX фирмы DEC, ЭС PALLADIO помогала проектировать и тестировать СБИС-схемы.

ЭС JUDITH помогает специалистам по гражданским делам и вместе с юристом и с его слов усваивает фактические и юридические предпосылки дела, а затем предлагает рассмотреть различные варианты подходов к разрешению дела.

ЭС LRS оказывает помощь в подборе и анализе информации о судебных решениях и правовых актах в области кредитно-денежного законодательства, связанного с использованием векселей и чеков.

ЭС "Ущерб" на основе российского трудового законодательства обеспечивает юридический анализ ситуации привлечения рабочих и служащих к материальной ответственности при нанесении предприятию материального ущерба действием или бездействием.

Список созданных ЭС можно перечислять очень долго. Были разработаны и внедрены тысячи реально работающих экспертных систем. Об этом мы будем говорить подробнее в 6 и 7 лекциях.

Разработка инструментальных средств для создания ЭС ведется постоянно. Появляются экспертные системы оболочки, совершенствуются технологии создания ЭС. Язык Пролог (1975-79 годы) становится одним из основных инструментов создания ЭС. Язык CLIPS (C Language Integrated Production System) начал разрабатываться в космическом центре Джонсона NASA в 1984 году [6]. Язык CLIPS свободен от недостатков предыдущих инструментальных средств для создания ЭС, основанных на языке LISP. Появляется инструментарий EXSYS, ставший в начале 90-х годов одним из лидеров по созданию ЭС [7]. В начале XXI века появляется теория интеллектуальных агентов и экспертных систем на их основе [8]. Web-ориентированный инструментарий JESS (Java Expert System Shell), использующий язык представления знаний CLIPS, приобрел достаточную известность в настоящее время [9]. Среди отечественных инструментальных средств следует отметить веб-ориентированную версию комплекса АТ-ТЕХНОЛОГИЯ, разработанного на кафедре Кибернетики МИФИ. В этом комплексе вся прикладная логика как комплекса в целом, так и разработанных в нем веб-интегрированных ЭС, сосредоточена на стороне сервера [10].

Практика внедрения ЭС показала, что нет чудодейственных рецептов - нужна кропотливая работа по вводу в ЭВМ опыта и знаний специалистов всех областей науки.

4. Машинный перевод и понимание текстов на естественном языке

Началом работ по машинному переводу следует считать 1954 год, когда в США с помощью ЭВМ было переведено шестьдесят фраз. Этот известный "Джорджтаунский эксперимент" произвел неизгладимое впечатление на специалистов. Тогда казалось, что достаточно создать большие хранилища словарей для перевода с одного языка на другой, разработать правила перевода - и проблема будет решена. Когда выяснилось, что проблема не так проста, был создан язык-посредник, облегчающий сопоставление фраз на разных языках. Во второй половине семидесятых годов этот язык-посредник превратился в семантическую модель представления смысла переводимых текстов. Таким образом, достижения в области машинного перевода оказались очень полезными для ИИ, так как они показали, что анализ естественно-языковых конструкций невозможен без создания семантической модели, в которой интерпретируется смысл слов. Создание семантической модели привело к пониманию необходимости создания языка для внутреннего представления знаний. И, наконец, важным достижением явилось понимание того, что анализ текстов на естественном языке (ЕЯ) состоит из четырех основных этапов: морфологический анализ, синтаксический анализ, семантический анализ, прагматический анализ [11].

Задача разбора предложения на естественном языке даже с ограниченным словарем в сотню слов является типичной задачей ИИ. Для достаточно формальных предметных областей и конкретных наборов предметов, обладающих однозначными, недвусмысленными характеристиками и названиями, такая система была создана Т. Виноградом в 1971 году. Его знаменитая программа SHRDLU

[12] работает в абстрактном мире, состоящем из разноцветных кубиков и пирамид. В этом мире функционирует "однорукий" робот Робби, способный общаться с человеком на естественном (английском) языке и совершать манипуляции с объектами этого мира. Этот робот также способен понимать достаточно сложные выражения и самостоятельно планировать свою деятельность и объяснять мотивы тех или иных действий.

Вот небольшой пример - фрагмент реального диалога с этой системой:

- Найди блок, который выше, чем тот, что в твоей руке, и положи его в коробку.

- ОК.

- Поставь, пожалуйста, один из красных блоков на другой, а сверху либо зеленый блок, либо пирамиду.

- ОК.

- Служит ли самая короткая вещь, которая опирается на опору самой высокой пирамиды, опорой для чего-то еще?

- Да, для зеленой пирамиды.

- Когда ты брался за зеленую пирамиду?

- В то время, когда я строил башенку из красного куба, большого красного блока и большого зеленого куба.

- Почему?

- Чтобы от нее избавиться.

- Почему?

- Чтобы освободить красный куб.

Многократное повторение вопроса "Почему?" в итоге приведет к ответу: "Потому что вы меня об этом попросили". Самым интересным здесь является то, что программа не отвечает на заранее заданные типовые фразы. SHRDLU, можно сказать, "понимает", о чем ее спрашивают, и отвечает достаточно разумно. Сам алгоритм Винограда настолько элегантен, что занимает всего несколько сотен строк кода на языке LISP, любимом языке разработчиков ИИ, занимающихся анализом ЕЯ. Этот пример с роботом Робби весьма показателен и мы будем обращаться к нему в разных лекциях.

Надо отметить, что даже для английского языка, который служит основой для всех современных языков программирования в силу своей лаконичности и достаточно формальной семантики, до сего дня не удалось создать более-менее эффективную программную систему, способную адекватно понимать СМЫСЛ фраз из достаточно больших областей знаний, например, нашего обыденного мира.

В разборе и понимании естественного русского языка массу проблем создает сложная падежная система, склонения, времена, отсутствие формального порядка следования членов предложения. Тем не менее российскими учеными созданы эффективные системы разбора фраз ограниченного естественного языка (ОЕЯ) [13], [14], [15].

Контрольные вопросы:

1. Порядок распознавания изображений?
2. Как преобразуется изображений в цифровой код?
3. Применение экспертных системх при создании автоматических консультирующих систем

4. Как осуществляется машинный перевод и понимание текстов на естественном языке?

Литература

7. Интеллектуальные системы автоматического управления /Под ред. И. М. Макарова, В. М. Лохина. – М.: Физматлит, 2001
8. Методы робастного, нейро-нечеткого и адаптивного управления. Под.ред. Н.Д.Егупова. М.: Изд-во МГТУ им. Н.Д.Баумана, 2002
9. Емельянов В.В, Ясинский С.И. Введение в интеллектуальное моделирование сложных дискретных систем и процессов. Язык РДО. – М.: АНВИК, 1998

Лекция 20: Создание единой программной среды и синтез алгоритмов для непосредственной задачи

План:

1. Области использования динамических ЭС
2. Моделирование гидравлических режимов каскадов водохранилищ на базе G2
3. Имитационная модель автоматизированного грузового комплекса на базе Rethink

1. Области использования динамических ЭС

Опыт применения динамических ЭС опишем на примере программных продуктов фирмы Gensym, занимающей первое место в мире среди фирм, производящих интеллектуальные продукты. Интерес к этой фирме вызван и тем, что с 1994 г. ее продукты доступны на рынке СНГ (и в частности, России). Интересы Gensym в СНГ представляет АО Аргуссофт Компани. К настоящему моменту, несмотря на высокую стоимость этих продуктов, во всем мире их продано более 5000 копий; все 25 самых крупных компаний мира используют систему G2.

На базе G2 созданы сотни промышленных систем. В частности, эта система успешно применяется в нефтяной промышленности (26 систем); космосе, авиации и обороне (31); производстве цемента (35); химии (35); энергетике (32); финансовых системах (22); металлургии (20); пищевой промышленности (16); телекоммуникации и связи (13); сборочном производстве (11); целлюлозно-бумажной промышленности (11); машиностроении (10); в горном деле (9); на транспорте (9); микроэлектронике (8); правительственных лабораториях (8); фармацевтическом производстве (7); биохимии (4); судостроении (6); для безопасности офисов (5); медицине (4); сельском хозяйстве (2); для учебных целей в университетах (108).

Широкое распространение G2 предопределило создание международного общества пользователей фирмы Gensym, а опыт ее использования освещается в многочисленных докладах и публикациях.

Перечислим наиболее характерные прикладные системы, разработанные на базе продуктов фирмы Gensym.

Нефть и газ: системы управления газопроводами, системы переработки нефти.

Финансы и бизнес: реинжиниринг банков и компаний (см. Приложение 2).

Космос: управление двигателями космических аппаратов (в частности, NASA широко использует G2 с октября 1988 г.); мониторинг критических состояний и диагностика сбоев коммуникационных каналов, обеспечивающих связь со спутниками.

Металлургия и машиностроение: контроль за состоянием доменной печи; управление оборудованием по производству сплавов; моделирование операций прокатного стана и связанных с ним печей, составление расписания обработки для печей и потока обрабатываемых материалов в целях наиболее полного использования оборудования.

Пищевая промышленность: планирование всего производственного процесса крупной пекарни, слежение за ним и управление; динамическое планирование загрузки линий по упаковке напитков.

Электроника: диагностика и выявление неисправностей в линиях по производству печатных плат в целях сокращения брака.

Транспорт: планирование загрузки контейнеров при авиационных, железнодорожных и автомобильных перевозках.

Ниже приведено более подробное описание российского опыта построения динамических ЭС на примере приложений для ЦДУ РАО ЕЭС России и АО "Шереметьево-Карго". Следует отметить, что в России, кроме описываемых ниже, в настоящее время более 15 предприятий разрабатывают приложения на базе продуктов фирмы Gensym в таких областях, как: финансы, телекоммуникация, контроль и диагностика космической аппаратуры, управление производством удобрений и т.п. Кроме того, продукты фирмы Gensym установлены в трех вузах России: Московском энергетическом институте (МЭИ), Московском государственном инженерно-физическом институте (МИФИ) и Государственной академии нефти и газа им. И.М. Губкина (ГАНГ), где они используются в учебном процессе. На базе этих продуктов разработаны следующие приложения: советчик оператора атомной электростанции, система управления ускорителем, система управления газопроводом.

2. Моделирование гидравлических режимов каскадов водохранилищ на базе G2

Для поддержки принятия решений по оперативному управлению гидравлическими режимами каскадов водохранилищ гидроэлектростанций (ГЭС) на базе инструментального комплекса G2 (фирма Gensym, США) специалистами ЦДУ РАО ЕЭС (Единая ЭнергоСеть) России, ВНИИЭ и АО Аргуссофт Компани была разработана система, позволяющая создавать и использовать имитационные модели каскадов. Актуальность решения данной задачи вызвана следующим.

При оперативном управлении режимами каскадов водохранилищ ГЭС, помимо требований эффективности работы станций, необходимо учитывать ограничения неэнергетических водопользователей. Эти ограничения по уровням и расходам в нижнем и верхнем бьефах ГЭС определяются условиями рыболовства, навигации и сельского хозяйства, условиями надежности гидросооружений и затопляемости территорий и т.д. Эффективное управление, обеспечивающее выполнение требуемых ограничений, основывается на анализе неустановившегося движения воды, позволяющем прогнозировать уровни и расходы на протяжении русла и, таким образом, оценивать принимаемые решения. В формальных моделях волнового перемещения масс воды по бьефам гидроузлов естественное русло реки представляется в виде схематизированного русла, состоящего из отдельных призматических участков, называемых первичными участками и отличающихся друг от друга по таким характеристикам, как поперечное сечение, уклон дна, коэффициент шероховатости и т.д. Формализованная таким образом задача из-за низкой точности исходной гидрологической и гидравлической информации не обеспечивает необходимой точности результатов, поэтому для формирования ре-

шений необходим анализ фактических данных, а также экспертные решения пользователей.

Модели имеют иерархическую структуру. В описаниях верхнего уровня, представленных картами-схемами, указываются географические расположения водохранилищ и гидроузлов. На более детальном уровне задаются описания гидроузлов (справочная информация и архивные данные по уровням и расходам для каждого гидроузла) и водохранилищ (гидравлические и морфометрические характеристики первичных участков русла, а также расположение удаленных контролируемых створов - створов выборочной выдачи - с архивными данными по расходам и уровням). На базе описания верхнего уровня формируются модели конкретных каскадов, в рамках которых и проводятся расчеты.

Разработчикам моделей система предоставляет следующие возможности:

• формировать и корректировать карты-схемы, содержащие информацию о географическом расположении водохранилищ и ГЭС;

• просматривать, вводить и корректировать справочную информацию о характеристиках водохранилищ и ГЭС;

• использовать в картах-схемах географические карты, подготовленные в виде графических файлов;

• формировать и корректировать данные о фрагментах русла -первичных участках - непосредственно на схематическом описании русла;

• формировать и корректировать данные о створах выборочной выдачи, а также подключать архивную информацию (в виде таблиц и графиков) по расходам и уровням.

Конечные пользователи имеют следующие возможности:

• получать автоматически сформированные модели каскадов по картам-схемам (в каскад включаются гидроузлы и водохранилища, отмеченные пользователем на карте-схеме);

• вести архивы фактических данных по расходам и уровням на периоды половодья;

• задавать и использовать в расчетах как фактические, так и экспериментальные параметры русла;

• задавать и корректировать граничные и начальные условия расчетов;

• проводить оперативный расчет неустановившегося движения воды с определением уровней во всех первичных участках и автоматическим выявлением нарушений допустимых пределов по уровням;

• получать фактическую информацию по уровням с автоматическим выявлением существенных отклонений от расчетных данных;

• проводить перерасчет с учетом полученных фактических данных и (или) откорректированных граничных условий.

Основным режимом работы системы является режим имитационного моделирования. В рабочее окно системы выводится автоматически сформированная схема русла участка реки с указанием створов и ГЭС. Выбрав любой из объектов модели, пользователь может просмотреть, ввести и (или) откорректировать архивную информацию по фактическим режимам, подготовить исходные данные для расчета, а также ознакомиться с текущими результатами имитационного моделирования.

При подготовке исходных данных для расчетной модели указываются начальные и граничные условия в виде уровенных и расходных режимов по первичным участкам на начальный момент расчета (задаются в виде гистограмм, которые пользователь может корректировать с помощью мыши), а также допустимые пределы по уровням для контролируемых створов. Затем можно запускать процедуру, имитирующую посуточный просчет режимов неустановившегося движения воды с заданным временным шагом (на

пример, 3 с соответствуют 1 суткам). Текущие результаты расчета для уровней отображаются на графике. Кроме этого можно просмотреть результаты расчетов по суткам для любого из створов выборочной выдачи как в графическом, так и в табличном виде.

В ходе расчета контролируется соблюдение ограничений на уровни в створах выборочной выдачи. В случае нарушения ограничений цветом выделяется изображение створа, в котором выявлено нарушение, и выдается предупреждение. Можно остановить расчет, откорректировать исходные данные и повторить процедуру моделирования.

Получив расчетные результаты, можно перейти в режим, в ходе которого имитируется ежесуточное поступление фактических данных и сравнивается поступающая информация с результатами расчета. В случае отклонения расчетных данных от фактических более чем на 15 % система выдает соответствующее предупреждение.

Низкая точность исходной гидрологической и гидравлической информации приводит к недостаточной точности результатов расчетов, и поэтому встает вопрос о настройке расчетной модели, т.е. построении уточненных характеристик участков русла исходя из анализа архива фактических данных и экспертных оценок пользователя.

В системе можно реализовать различные способы формирования экспериментальных характеристик. В текущей версии имеется возможность строить экспериментальные оценки зависимостей модулей расходов от глубин $K_i(h)$ для первичных участков на основе архивной информации о фактических расходах за те дни, когда имел место статический режим расходов.

Получение экспертных оценок параметров, используемых в расчетной модели, может дать хорошие результаты только в тех случаях, когда принятые в модели допущения соответствуют реальным условиям задачи. Чтобы расширить область эффективного использования системы, в настоящее время исследуется возможность применения методов нейронных сетей для получения значений уровней и расходов по начальным и граничным условиям.

Система поддержки оперативного управления гидравлическим режимом каскадов водохранилищ функционирует на рабочих станциях Sun и RS 6000. Расчетный модуль реализован на языке Си. Система включает около 80 классов и более 1700 объектов языка представления знаний комплекса G2, а также более 200 процедур и более 30 общих правил. В настоящее время система используется для работы с каскадом Нижне-Волжских ГЭС.

3. Имитационная модель автоматизированного грузового комплекса на базе Rethink

Общие характеристики модели. Имитационная модель технологического процесса обработки груза на типовом грузовом предприятии от приема груза с транспортного средства до выдачи его клиенту разработана для апробации методики и программных средств, предлагаемых АО "Аргуссофт Компани", при проведении комплексных аналитических исследований эффективности функционирования подразделений и служб АО "Шереметьево-Карго". Модель реализована на Rethink и исполняется на Pentium под операционной системой WindowsNT. Модель включает 236 объектов, представляющих элементарные технологические операции, 14 классов для описания предметов труда и используемых ресурсов, 6 процедур и 8 продукционных правил, расширяющих возможности базовой среды моделирования.

Содержание прототипа модели. Типовое грузовое предприятие использует наличные ресурсы в ходе исполнения трех основных технологических процессов:

1. *Импорт.* Получение груза для клиента, извещение клиента, временное хранение груза до прихода клиента, выдача груза клиенту.

2. *Экспорт.* Прием груза от клиента, временное хранение груза, отправка груза в пункт назначения.

3. *Трансфер.* Получение груза для дальнейшей транспортировки, временное хранение груза, отправка груза в пункт назначения.

Модель отображает использование ресурсов предприятия при совместном исполнении перечисленных процессов и обеспечивает:

• отображение и сбор статистики по выбранной системе технико-экономических показателей;

• введение новых показателей;

• модификацию модели для отображения новых технологических схем исполнения технологического процесса;

• интерфейс с пользователем для изменения значений регулируемых параметров.

В состав регулируемых параметров модели включены;

• характеристики грузопотока;

• количество и структура людских ресурсов;

• количество и номенклатура используемого оборудования;

• структура затрат на исполнение моделируемого процесса.

Выходные характеристики модели, позволяющие оценивать результаты деятельности предприятия, должны включать:

• сводные показатели эффективности исполнения технологического процесса;

• себестоимость обработки грузов в ходе исполнения моделируемого процесса;

• коэффициенты использования оборудования и людских ресурсов;

• время обработки грузов.

Модель обеспечивает возможность, варьируя значения регулируемых параметров, получать сводные характеристики исполнения моделируемых процессов.

Экспериментальная апробация модели позволила уточнить структуру и состав необходимых параметров для разработки полной модели АГК, включающей:

• комплекс технико-экономических показателей, позволяющих объективно оценивать использование ресурсов и оборудования, а также удовлетворять требования клиентов АГК в условиях моделируемой организационной структуры;

• набор регулируемых параметров модели, отображающих как внешние условия функционирования предприятия (расписание прибытия и отправки авиарейсов, налоговые ставки и т.п.), так и стратегические управляющие воздействия (тарифные ставки, запасы ресурсов).

Структура модели:

1. ***Рабочие объекты (предметы труда).*** Рабочие объекты, используемые в модели, являются производными от класса **bpr-object** и наследуют от него стандартные стоимостные и временные характеристики. Общим для представителей класса bpr-object является накопление в атрибуте TOTAL-COST подтаблицы COST-SUBTABLE затрат на исполнение всех операций, в которых участвовал объект, т.е. себестоимости его обработки. Ниже приводятся основные рабочие объекты модели АГК с краткой характеристикой их назначения и специфических атрибутов.

Самолет. Характеризуется номером авиарейса, временем прибытия и вылета, средним и максимальным количеством доставляемого груза. Характеристики авиарейсов, используемые в модели, задаются в двух файлах. В первом из них содержится информация о рейсе, а во втором - интервалы времени между прибытием рейсов в Москву. Для учета того, что информация о прибытии рейса поступает на АГК за два часа до реального прибытия и для учета разницы между прибытием последнего и первого рейсов в расписании, сдвиг первого рейса установлен в 1 час. С другой стороны, для учета недельного цикла расписания стартовое время модели должно устанавливаться в 21.30 любого воскресенья.

Информация о рейсе. Содержит данные о реальном объеме груза, который прибывает с данным авиарейсом на АГК, и перроне, на который прибывает рейс. Данная информация служит основой для формирования сцепки тележек, подаваемых на соответствующий перрон под разгрузку.

Команда на вылет. Формируется в момент прибытия рейса на АГК и запускает таймер, гарантирующий отбытие рейса из Москвы в соответствии с расписанием. Таймер работает параллельно с операциями разгрузки и погрузки самолета и поэтому позволяет зафиксировать отклонения от графика работ, ведущие к задержке вылета рейса.

Авианакладная. Является основным документом, сопровождающим груз во время обработки. Общий вес груза на накладную разыгрывается по статистикам, характерным для реального грузопотока на АГК.

Груз. Является элементом обработки и хранения в ячейках складов АГК. Характеризуется весом и ассоциирован с накладной, по которой он прибыл.

Клиент. В явном виде в модели представлен только клиент - отправитель груза. После прибытия клиента в блок бронирования в модели появляется авианакладная, клиент покидает модель, и дальнейшие операции осуществляются с накладной.

Тележка. Вместе с палетой используется для перевозки грузов к самолету и от самолета на склад.

Полета. Вместе с тележкой используется для перевозки грузов к самолету и от самолета на склад. Вместе с грузом и комплектом авиа-накладных отправляется в пункт назначения и прибывает на АГК с самолетом.

Комплект. Абстрактный объект, служащий для объединения авианакладных, лежащих на одной палете во время транспортировки.

Сцепка. Абстрактный объект, служащий для объединения от одной до шести тележек, перевозимых трактором *по* территории АГК и между АГК и перроном Ш-2.

2. Ресурсы (средства труда). Ресурсы являются производными от класса *bpr-resource* и предназначены для ограничения исполняемых операций на основе объема и состава наличных ресурсов. Каждый из ресурсов характеризуется стоимостью его использования, переносимой в процессе моделирования в затраты на исполнение соответствующих операций и себестоимость обработки рабочих объектов.

Трактор. Служит для транспортировки сцепки тележек по территории АГК и между АГК и перроном Ш-2.

Автотрак. Служит для транспортировки грузов с авианакладными для дальнейшего следования с отправкой из других аэропортов Москвы.

Грузчик. Привлекается для выполнения операций, связанных с погрузкой (разгрузкой) транспортных средств и помещением (выдачей) грузов на хранение.

Ячейка. Единица хранения в складских помещениях.

Методика проведения экспериментов. На верхнем уровне моделирования (уровень детализации 0) АГК представляется в виде единого блока с двумя входами и тремя выходами. Ряд экономических показателей блока АГК характеризуют затратные статьи, не зависящие от технологических характеристик модели. Показатель ВЕС-ОБРАБОТАННОГО-ГРУЗА накапливает нарастающим итогом вес грузов, покинувших АГК за время с начала моделирования. Показатель ТАРИФ устанавливает расценки АГК на килограмм груза, оплачиваемого клиентами, для определения дохода предприятия. Подтаблица COST-SUBTABLE (как и у других блоков модели) содержит стоимостные статистики, рассчитываемые в ходе моделирования. Наибольший интерес для исследования представляет характеристика TOTAL-COST, содержащая текущий уровень затрат на предприятии, исчисляемый из затрат на использование ресурсов при исполнении отдельных технологических операций,

Входы модели:

1. *Прибывающие авиарейсы.* Данный вход разделяется на два потока:

• авиарейсы, поступающие на перроны АГК и Шереметьево-2;

• информации о рейсах, поступающих на АГК за два часа до прибытия соответствующего рейса.

2. *Клиенты* - отправители грузов, являющиеся источником информации об объеме отправляемого груза. ***Выходы модели:***

1. *Пункт назначения.* Самолет, покидая блок АГК, попадает в пункт назначения, где из него извлекается экспортный груз. Самолеты и грузы покидают модель. Дополнительная функция, исполняемая данным блоком - генерация грузов, отправляемых на АГК зарубежными клиентами.

2. *Отправка траков.* Импортные грузы, предназначенные для дальнейшей транспортировки внутренними авиарейсами, покидают АГК на автотраках.

3. *Клиент-грузополучатель.* Через этот блок покидают модель грузы, получаемые клиентами в Москве.

Порядок запуска модели. Ниже перечислен типичный порядок запуска и экспериментирования с моделью.

1. *Установка начальных значений параметров моделирования*.

На данном уровне детализации АГК характеризуется объемами грузов, поступающими на входы и покидающими модель через перечисленные выходы. Экспериментатор имеет возможность устанавливать напряженность грузопотока, изменяя расписание рейсов и интервалы времени между поступлением экспортных и импортных грузов. В модели предусмотрены два способа задания расписания авиарейсов:

• на основе данных, записанных в текстовых файлах SCHEDULE.TXT (характеристики 230 реальных рейсов, прибывающих на перроны Шереметьево-2 и АГК в течение недели) и DURATION.TXT (временные интервалы между прибытием соответствующих рейсов);

• на основе графика интенсивности прилетов, находящегося в подпространстве сценария модели. Как в первом, так и во втором случае характеристики рейсов (средний вес привозимых грузов, место разгрузки, длительность стоянки перед обратным вылетом) берутся из файла SCHEDULE.TXT.

В подпространстве сценария находятся также полозковые регуляторы, позволяющие установить интервал между поступлением импортных и экспортных клиентов. Предполагается, что клиенту соответствует одна авианакладная, вес грузов и количество мест на накладную разыгрываются в соответствии со статистиками реального грузопотока.

Установив исходные характеристики грузопотока, не покидая подпространства сценария, экспериментатор имеет возможность изменить количество ресурсов, привлекаемых для выполнения процессов обработки грузов: количество автотраков; количество палет на складе; количество складских тракторов; количество тележек; количество складских тракторов.

2. *Запуск модели.* Следующим этапом после установки начальных значений является запуск модели. Запуск может быть осуществлен в одном из трех режимов: с переменным шагом (Jump); пошаговый (Step); синхронный (Synch);

3. *Наблюдение за ходом моделирования.* Система ReThink и лежащий в ее основе инструментальный комплекс G2 обеспечивают широкий спектр форм представления информации для наблюдения за ходом моделирования.

Причинно-следственные взаимосвязи, взаимовлияние параллельных потоков событий и порядковые отношения между наступлением событий во времени экспериментатор имеет возможность отслеживать непосредственно на диаграммах модели, наблюдая за перемещением рабочих объектов модели между блоками и за занятием и освобождением соответствующих ресурсных объектов.

Динамику изменения различных показателей показывают графики, помещенные в рабочем пространстве ОТЧЕТЫ. Здесь присутствуют графики удельной загрузки складов, коэффициентов использования тракторов, выходного грузопотока, коэффициента использования тележек и график экономических показателей. Графики удельной загрузки складов и коэффициентов использования тракторов реализованы полностью на базе средств системы ReThink.

4. *Моментальный снимок текущего состояния модели.* Кнопка *Моментальный снимок* служит для архивирования текущего состояния модели, с возможностью в дальнейшем "теплого пуска" процесса моделирования с данной контрольной точки. Моментальный снимок состояния сохраняется в файле SNAPSHOT.KB в текущем каталоге модели. Переписывая данный файл или переименовывая его, можно создать набор контрольных точек, наиболее интересных с точки зрения анализа протекания моделируемых бизнес-процессов.

Перспективы дальнейшего развития модели. В дальнейшем предполагаются развитие и детализация модели в направлении повышения ее адекватности реальному процессу обработки грузов АО "Шереметьево-Карго", что позволит оценивать результаты перераспределения ресурсов и структурной реорганизации работы предприятия до их внедрения, а также эффективность работы с клиентами (в частности, тарифной политики) в условиях современного динамично развивающегося рынка транспортных и складских услуг. В более отдаленной перспективе планируется преобразование модели в систему поддержки принятия решений диспетчерским персоналом АГК, интегрированную к АСУ "Шереметьево-Карго" и призванною оптимально ращпремелять поступающие самолеты и грызы, а тикже оптимально выделять наличнын ресурсы рабочей силы и техники(в реальном масштабе времени.

Контрольные вопросы:

1. Области использования динамических ЭС?
2. Моделирование гидравлических режимов каскадов водохранилищ на базе G2?
3. Имитационная модель автоматизированного грузового комплекса на базе Rethink?

Литература

1. *Едельштайн Н.* **Интеллектуальные средства анализа, интерпретации и представления данных в информационных хранилищах** // Компьютеруик. -1996. - №16.
2. *Hall C.* **The devil s in the details: techniques, tools, and applications for database mining and knowledge discovery** //Intelligent Software Strategies. - P. I. V XI. -№9 - 1995, September.
3. *Hall C.* **The devil s in the details: techniques, tools, and applications for database mining and knowledge discovery**//Intelligent Software Strategies - P.II. V. XI. -№9. - 1995, October.
4. *Inmon W.H.* **Building the Data Warehouse.** - NY: John Wiley & Sons, Inc., 1992. - 298 P.

ЛИТЕРАТУРА

1. Дракин В.И., Попов Э.В., Преображенский А.Е. Общение конечных пользователей с системами обработки данных. - М.: Радио и связь, 1988.- 287 с.
2. Информационные системы в экономике: Учебник / Под ред. проф. В.В. Дика. - М.: Финансы и статистика, 1996. - 272с.
3. Искусственный интеллект. Книга 1. Системы общения и экспертные системы./ Под ред. проф. Э.В.Попова.- М.: Радио и связь, 1990. - 461с.
4. Искусственный интеллект. Книга 2. Модели и методы. / Под ред. проф. Д.А.Поспелова. - М.: Радио и связь, 1990. - 304 с.
5. Искусственный интеллект. Книга 3. Программные и аппаратные средства. / Под. ред. В.Н.Захарова, В.Ф.Хорошевского. - М.: Радио и связь, 1990. - 320с.
6. Калянов Г.Н. Консалтинг при автоматизации предприятий: Научно-практическое издание. Серия «Информатизация России на пороге XXI века». - М.: СИНТЕГ, 1997. - 316 с.
7. Левин Р., Дранг В., Эделсон Б. Практическое введение в технологию искусственного интеллекта и экспертных систем с иллюстрациями на Бэйсике./ Пер. с англ.- М.: Финансы и статистика, 1991.- 239с.
8. Мишенин А.И. Теория экономических информационных систем. М.: Финансы и статистика, 1993 - 166 с.
9. Обработка знаний / Пер. с япон.: Под ред. С.Осуга. - М.: Мир, 1989. - 292 с.
10. Ойхман Е.Г., Попов Э.В. Реинжиниринг бизнеса: Реинжиниринг организаций и современные информационные технологии. - М.: Финансы и статистика , 1997. -336с.: ил.
11. Попов Э.В., Шапот М.Д., Кисель Е.Б., Фоминых И.Б.. Статические и динамические экспертные системы, М: Финансы и статистика, 1996. -320с. : ил.
12. Поспелов Д.А. Моделирование рассуждений. Опыт анализа мыслительных фактов. - М.: Радио и связь, 1989.- 184 с.
13. Приобретение знаний / Пер. с япон.: Под ред. С.Осуга., Ю. Саэки - М.: Мир, 1990. - 292 с.
14. Представление и использование знаний / Пер. с япон.; Под ред. Х.Уэно, М.Исидзука. - М.: Мир, 1990. - 220 с.
15. Таунсенд К., Фохт Д. Проектирование и программная реализация экспертных систем на персональных ЭВМ. - М.: Финансы и статистика, 1990.- 319с
16. Тельнов Ю.Ф., Скорова А.А., Андреева Н.В. Проектирование баз знаний. Учебное пособие.- М.: МЭСИ, 1992.-100с.
17. Тельнов Ю.Ф., Диго С.М., Полякова Т.М. Интеллектуальные системы обработки данных. Учебное пособие.- М.: МЭСИ, 1989.-102с.
18. Тельнов Ю.Ф. Реинжиниринг бизнес-процессов и проектирование информационных систем. - В кн.: «Реинжиниринг бизнес-процессов предприятий на основе современных информационных технологий». Сб. научных трудов 2-й Российской научно-практической конференции. - М.: МЭСИ, 1998. - с.28 - 34.
19. Уоссермен Ф. Нейрокомпьютерная техника. Теория и практика. / Пер с англ. Ю.А. Зуева М.:Мир, 1992 -237 с.
20. Уотерман Д. Руководство по экспертным системам. / Пер. с англ.; Под. ред. Стефанюка В.Л. - М.: Мир, 1989.- 388 с.

21.Буч Г. Объектно-ориентированное проектирование с примерами применения: Пер. с англ. - М.: Конкорд, 1992. - 519с.
22.Ин Ц., Соломон Д. Использование Турбо-Пролога. Пер. с англ. - М.: Мир, 1993. - 608с.
23.Марселлус Д. Программирование экспертных систем на ТУРБО ПРОЛОГЕ. Пер. с англ. - М.: Финансы и статистика, 1994. -256с.
24.Нечеткие множества в моделях управления и искусственного интеллекта/ Под ред. Поспелова Д.А. - М.: Наука, 1986 - 312 стр.
25.Попов Э.В., Шапот М.Д., Кисель Е.Б., Фоминых И.Б. Статические и динамические экспертные системы. - М: Финансы и статистика, 1996. -320с.
26.Построение экспертных систем / Под ред. Ф. Хейос-Рот, Д.Уотерман, Д.Ленат / Пер. с англ. - М.: Мир, 1987. - 441 с.
27.Представление и использование знаний / Пер. с япон.; Под ред. Х.Уэно, М.Исидзука. - М.: Мир, 1989. - 220 с.
28.Российский софт 97. Справочник по программному обеспечению. М.: Метод, 1997. - 160с.
29.Системы управления базами данных и знаний: Справочное издание/ Наумов А.Н., Вендров А.М., Иванов В.К. и др./ Под ред. Наумова А.Н. - М.: Финансы и статистика, 1991 - 180 стр.
30.Таунсенд К., Фохт Д. Проектирование и программная реализация экспертных систем на персональных ЭВМ. Пер. с англ. - М.: Финансы и статистика, 1990.-319с.
31.Тельнов Ю.Ф., Скорова А.А., Андреева Н.В. Проектирование баз знаний. Учебное пособие.- М.: МЭСИ, 1992.-100с.
32.Тельнов Ю.Ф., Диго С.М., Полякова Т.М. Интеллектуальные системы обработки данных. Учебное пособие.- М.: МЭСИ, 1989.-102с.
33.Уотерман Д. Руководство по экспертным системам. / Пер. с англ.; Под. ред. Стефанюка В.Л. - М.: Мир, 1989.- 388 с.
34.Форсайт Р. Экспертные системы: принципы и примеры. / Пер. с англ - М.: Радио и связь, 1987.
35.Цикритзис Д., Лоховски Ф. Модели данных. /Пер. с англ. - М.: Финансы и статистика, 1985. -344с.
36.Шлеер С., Меллор С. Объектно-ориентированный анализ: Моделирование мира в состояниях. Пер. с англ. - Киев: Диалектика, 1993. -240с.
37.Элти Дж., Кумбс М. Экспертные системы: концепции и примеры /Пер. с англ. - М.: Финансы и статистика, 1987.- 191 с.
38.Durkin J. Expert Systems: a view of the field. IEEE Expert, 1996, No2 ,p. 56- 63
39.Harmon P. The intelligente software development tools market// Part I. Intelligent Software Strategies. - 1995, Vol. 11, No 2. p. 1 - 12.
40.Harmon P. The intelligente software development tools market// Part II. Intelligent Software Strategies. - 1996, Vol. 12, No 3. p. 1 - 16.
41.Martinson, F.R. Schindler. Organizational visions for technology assimilation: the strategic road to knowledge-based systems success. IEEE Transactions on engineering management, 1995, Vol 42, No 1, p 10 - 18.
42.Ross R.G. The Business Rule Book. Classifying, Defining and Modelling Rules. Data Base Research Group, Inc. -1997, 394 p.

43.Decision support systems. Putting theory into practice. Edited by R.H. Sprague, H. Watson. Prentice-Hall, 1993. -437p.
44.Gevarter W.B.. The Nature and Evaluation of Commertial Expert Systems. Building Tools.- Computer, May, 1987. p 24-41.
45.Martinson, F.R. Schindler. Organizational visions for technology assimilation: the strategic road to knowledge-based systems success. IEEE Transactions on engineering management, 1995, Vol 42, No 1, p 10 - 18.
46.Talebzadeh, Mandutianu S., Winner C.F. Countrywide Loan-Underwriting Expert System. AI Magazine, 1995, april, p. 51 - 64.

ТАШКЕНТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

КАФЕДРА “АВТОМАТИЗАЦИЯ ПРОИЗВОДСТВЕННЫХ ПРОЦЕССОВ”

ПРАКТИЧЕСКИЕ РАБОТЫ ПО ДИСЦИПЛИНЫ “ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМ УПРАВЛЕНИЕ И ПРИНЯТЬ РЕШЕНИЯ ”

ТАШКЕНТ – 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В последние годы за рубежом существенно повысился интерес к исследованию прикладных интеллектуальных управляющих систем, их разработке и внедрению в промышленную и непромышленную сферы.

Интерес к интеллектуальным системам управления (ИСУ) объясняется рядом причин. Первая из них состоит в том, что традиционные технологии уже не могут обеспечить повышение качества управления, поскольку не учитывают всех неопределенностей, воздействующих на систему

Методическая пособие предназначена для практических занятий по предмету «Интеллектуальные систем управление и принять решения». В методическом пособие приведены практические занятия, рассчитанные на 40 часов.

Практическая работа 1

Типы динамических экспертных систем

Цель работы: изучение типов динамических экспертных систем

Для классификации ЭС выберем такие параметры, которые удовлетворяют двум условиям. Во-первых, выбирая значение этих параметров, пользователь, не являющийся специалистом в ЭС, должен быть способен характеризовать особенности своего приложения. Это позволит разработчику ЭС выбрать ИС, адекватное данному приложению. Во-вторых, параметры и их различные значения должны обеспечивать разработчика ЭС информацией, достаточной для ответа на стратегические вопросы, возникающие у пользователя на различных этапах существования приложения.

Примерами вопросов, стоящих перед пользователем, являются следующие: "Сможет ли создаваемая ЭС использовать созданные ранее программы?", "Будет ли ЭС работать с разнородной программно-технической средой пользователя?", "Насколько создаваемая ЭС будет критична к предполагаемой смене платформ (ЭС с операционной системой)?", "Сможет ли ЭС решать все задачи данного приложения или часть останется, например, за экспертом?", "Каковы сроки окупаемости ЭС?", "Адекватны ли выбранные разработчиком ИС задачам пользователя?", " Когда ЭС будет использоваться для решения практических задач пользователя, а не просто будет сдана пользователю?", "Какова стоимость разработки, использования и сопровождения (модификации) ЭС?"

Будем классифицировать приложения с ЭС по следующим параметрам:

- тип приложения;
- стадия существования;
- масштаб;
- тип проблемной среды.

Тип приложения

Тип приложения характеризуют следующие наборы параметров.

1. Возможность взаимодействия приложения с другими программными средствами:

• *изолированное приложение,* состоящее из ЭС, не способной взаимодействовать с другими программными системами, используемыми конечным пользователем (например, с БД, электронными таблицами, пакетами прикладных программ, контроллерами, системой датчиков и т. п.);

• *интегрированное приложение,* состоящее из ЭС и других программных систем, с которыми ЭС взаимодействует в ходе работы.

Подчеркнем, что большинство современных (особенно динамических) ЭС, используемых для решения практически значимых задач, являются интегрированными.

2. Возможность исполнять приложение на разнородной аппаратуре и переносить его на различные платформы:

• *закрытые приложения,* которые исполняются только в программной среде данной фирмы и могут быть перенесены на другие платформы только путем перепрограммирования приложения;

• *открытые приложения,* которые ориентированы на исполнение в разнородном программно-аппаратном окружении и в идеале могут быть перенесены на другие платформы без перепрограммирования.

3. Архитектура приложения:

• приложение реализуется как *централизованное,* на базе центральной ЭВМ, с которой связаны терминалы;

• децентрализованное *распределенное приложение;* в настоящее время обычно используется архитектура клиент-сервер.

Стадия существования

Стадия существования характеризует степень проработанности и отлаженности ЭС. Обычно выделяют следующие стадии:

• исследовательский прототип;

• действующий прототип;

• промышленная система;

• коммерческая система.

Исследовательским прототипом называют систему, которая решает представительный класс задач приложения, но может быть неустойчива в работе и не полностью проверена. При наличии развитых инструментальных средств (ИС) для разработки исследовательского прототипа требуется примерно 2 - 4 месяца. Исследовательский прототип обычно имеет в базе знаний не больше 50 общих исполняемых утверждений; при использовании только частных утверждений их количество возрастает в 3 - 10 раз.

Действующий прототип надежно решает все задачи, но для решения сложных задач может требовать чрезмерно много времени и (или) памяти. Доведение системы от начала разработки до стадии действующего прототипа требует примерно 6 - 9 месяцев, при этом количество исполняемых утверждений в базе знаний увеличивается до 100.

ЭС, достигшая стадии *промышленной системы,* обеспечивает высокое качество решений всех задач при минимуме времени и памяти. Обычно процесс преобразования действующего прототипа в промышленную систему состоит в расширении базы знаний (до 150 исполняемых утверждений) и ее тщательной отладке. Доведение ЭС от начала разработки до стадии промышленной системы на развитом ИС требует примерно 12 - 18 месяцев.

Обобщение задач, решаемых ЭС на стадии промышленной системы, позволяет перейти к стадии *коммерческой системы,* т.е. к системе, пригодной не только для собственного использования, но и для продажи различным потребителям. Доведение системы до коммерческой стадии требует примерно 1,5 - 2 года. Приведенные выше сроки справедливы для ЭС средней сложности.

Масштаб ЭС (тип ЭВМ)

Многие специалисты классифицируют ЭС (приложения) по их сложности (типу используемой ЭВМ) на малые, средние, большие и символьные.

Малые ЭС реализуются на ПК типа PC или Macintosh, часто являясь изолированными ЭС. Малые ЭС обычно используются в целях первичного обучения или для исследования возможности использования технологии ЭС в данной области.

Средние ЭС реализуются на рабочих станциях. Они бывают изолированными и интегрированными с БД и электронными таблицами. Данные приложения охватывают весь спектр использования ЭС.

Большие ЭС реализуются на рабочих станциях или ЭВМ общего назначения (mainframe). Они, как правило, имеют доступ к огромным БД.

Символьные ЭС реализуются на символьных ЭВМ или с использованием ИС типа Lisp и Prolog. Эти ЭС, как правило, являются исследовательскими и не используются для решения реальных задач.

Тип проблемной среды

Понятие "проблемная среда" включает предметную область (множество сущностей, описывающих область экспертизы, т.е. множество объектов, значений их характеристик и связывающих их отношений) и решаемых в предметной области задач. Иначе говоря, проблемная среда включает сущности (структуры данных) и решаемые над ними задачи, представляемые в виде исполняемых утверждений (в виде правил, процедур, формул и т. п.). В связи с этим проблемная среда определяется характеристиками соответствующей предметной области и характеристиками типов решаемых в ней задач. Заметим, что наряду с понятием "проблемная среда" используется синонимичный ему термин "проблемная область".

Характеристики предметной области определяются следующим набором параметров:

1) тип предметной области:

статический, т.е. входные данные не изменяются за время сеанса работы приложения, значения других (не входных) данных изменяются только ЭС;

динамический, т.е. входные данные, поступающие от внешних источников, изменяются во времени, значения других данных изменяются ЭС или подсистемой моделирования внешнего окружения;

2) способ описания сущностей предметной области: совокупность атрибутов и их значений (фиксированный состав сущностей);

совокупность классов (объектов) и их экземпляров (изменяемый состав сущностей);

3) способ организации сущностей в БЗ:

неструктурированная БЗ;

структурирование сущностей БЗ по различным иерархиям (наиболее распространенные иерархии: "общее/частное", "часть/целое"), что обеспечивает наследование свойств сущностей, представляемых в БЗ.

Характеристиками типов задач являются:

1) тип решаемых задач:

задачи анализа и (или) синтеза;

статические или динамические задачи;

2) частность (общность) исполняемых утверждений (правил, процедур, формул и т. д.):

частные (специализированные, конкретные) исполняемые утверждения;

общие исполняемые утверждения.

Наиболее естественным для человека способом описания *сущностей предметной области* является соотнесение с ними в памяти ЭВМ объектов, состоя-

щих из атрибутов со значениями. Обычно вводится описание объекта некоторого типа, в соответствии с которым создаются конкретные экземпляры объектов этого типа. При этом количество экземпляров объекта никак не ограничивается, т.е. состав представляемых сущностей при таком представлении проблемной области является *изменяемым.*

Однако для простых приложений при малом количестве объектов (от 1 до 5) иногда применяют упрощенное представление в виде атрибут/значение без упоминания объекта, которому эти атрибуты принадлежат. Следствием этого упрощения явилось то, что реально существующие объекты (сущности) предметной области стали представляться в виде фиксированного количества размноженных имен соответствующих атрибутов. Например, вместо ссылки на атрибут i (1*K) объекта X (l÷N) использовали ссылку на атрибут j (l÷(KxN)). Таким образом, вместо K атрибутов вводили KxN атрибутов. Достоинства этого подхода состоят в том, что обеспечивается прямая ссылка на атрибут (объект). Недостатки в общем случае весьма значительны. Они заключаются в следующем:

1) невозможно использовать "общие" исполняемые утверждения (правила, процедуры, функции и т. п.), ссылающиеся на произвольное количество сущностей; приходится использовать частные (специализированные) утверждения, что увеличивает их количество. Кроме того, при изменении состава сущностей в БЗ приходится вводить новые соответствующие им специализированные утверждения;

2) устранение объектов не позволяет явно определить естественные взаимосвязи между атрибутами одного объекта (все атрибуты являются как бы независимыми);

3) с устранением объектов исчезает возможность определить иерархию классов объектов.

Следующим параметром, характеризующим предметную область, является *наличие (отсутствие) структурирования БЗ.* Смысл структурирования БЗ состоит в следующем:

1) ограничить круг сущностей, которые должны рассматриваться механизмом вывода, и таким образом сократить перебор при выборе решения;

2) обеспечить *наследование свойств сущностей,* т.е передачу свойств вышестоящих в иерархии сущностей нижестоящим, что значительно упрощает процесс приобретения и использования знаний. Например, общие свойства класса "автомобиль" автоматически наследуются всеми подклассами автомобилей и конкретными экземплярами этих подклассов. В подобной иерархии наследуется отношение "являться подмножеством (экземпляром)". Кроме того, широко применяется и другая иерархия - "является частью".

По типу решаемых задач в первую очередь все задачи целесообразно разделить на задачи анализа и синтеза.

В *задаче анализа* задана модель сущности (объекта), и требуется в результате анализа этой модели определить некоторые неизвестные характеристики (функции) модели. В *задаче синтеза* задаются условия, которым должны удовлетворять характеристики (функции) некоторой "неизвестной" модели сущности, и требуется построить модель этой сущности. Решение задачи синтеза представляет собой итерационный процесс, состоящий из следующих шагов:

1) создание исследовательской модели сущности;

2) анализ этой модели (т. е. решение задачи анализа);

3) сравнение результатов анализа с условиями задачи.

Таким образом, задача синтеза включает в себя анализ. Отметим, что в процессе создания конкретной ЭС, решающей задачу анализа, разработчик, создавая БЗ (модель области экспертизы), решает задачу синтеза, а построенная ЭС будет решать задачу анализа.

Задачи синтеза и анализа могут решаться в статических и динамических областях. Если ЭС базируется на предположении, что исходная информация о предметной области (об окружающем мире), на основе которой решается задача, не изменяется за время решения задач, то говорят о *статической предметной области* (точнее о статическом представлении области в ЭС); если информация о предметной области изменяется за время решения задач, то говорят о *динамической предметной области.* При представлении динамической области возникает задача моделирования окружающего мира, в частности моделирования активных агентов.

Если задачи, решаемые ЭС, явно не учитывают фактор времени и (или) не изменяют в процессе своего решения данные (знания) об окружающем мире, то говорят, что ЭС решает *статические задачи;* если задачи учитывают фактор времени и (или) изменяют в процессе решения данные об окружающем мире, то говорят о решении *динамических задач.* Таким образом, ЭС работает в статической проблемной среде, если она использует статическое представление и решает статические задачи. ЭС работает в динамической проблемной среде, если она использует динамическое представление и (или) решает динамические задачи.

Учитывая значимость времени в динамических проблемных средах, многие специалисты называют их приложениями (ЭС), работающими в реальном времени. Обычно выделяют следующие системы реального времени: *"псевдореального"* времени, *"мягкого"* реального времени и *"жесткого"* реального времени. Системы псевдореального времени, как следует из названия, не являются системами реального времени, однако они в отличие от статических систем получают и обрабатывают данные, поступающие от внешних источников. Системы псевдореального времени решают задачу быстрее, чем происходят значимые изменения информации об окружающем мире.

Системы "мягкого" реального времени работают в тех приложениях, где допустимо время реакции на события более 0,1 - 1 с. К этому диапазону относятся почти все существующие ЭС реального времени. Системы "жесткого" реального времени должны обеспечивать время реакции быстрее 0,1 - 0,5 с. Для достижения такого быстродействия они используют не стандартные операционные системы (ОС) типа Unix и Windows NT, а специализированные ОС и специализированные бортовые ЭВМ, обеспечивающие быстрое время реакции. В настоящее время ЭС, работающие в "жестком" реальном времени, нам не известны.

Задачи, решаемые ЭС, различают тем, как представляются исполняемые утверждения. Используются как *частные (специализированные) утверждения,* т.е. утверждения, содержащие ссылки на конкретные сущности (объекты), так и *общие утверждения,* относящиеся к любым сущностям заданного типа (вне зависимости от их числа и имени). Использование общих утверждений позволяет значительно лаконичнее представлять знания. Однако так как общие утверждения не содержат явных ссылок на конкретные сущности, для их использования требуется

затратить значительную работу по определению тех сущностей, к которым они должны применяться, т.е. выполнить, как говорят специалисты, операцию сопоставления.

Не все сочетания перечисленных выше параметров, характеризующих проблемную среду, встречаются на практике. Выделим несколько наиболее часто встречающихся типов проблемных сред.

Тип 1. Статическая проблемная среда: статическая предметная область; сущности представляются как совокупность атрибутов и их значений; состав сущностей неизменяемый; БЗ не структурирована; решаются статические задачи анализа, используются только специализированные исполняемые утверждения.

Тип 2. Статическая проблемная среда: статическая предметная область; сущности представляются в виде атрибутов со значениями или вырожденных объектов (фреймов); состав сущностей неизменяемый; иерархия БЗ либо отсутствует, либо слабо выражена (нет наследования свойств); решаются статические задачи анализа, используются специализированные исполняемые утверждения.

Тип 3. Статическая проблемная среда: статическая предметная область; сущности представляются в виде объектов; состав сущностей изменяемый; БЗ структурирована; решаются статические задачи анализа и синтеза, используются общие и специализированные исполняемые утверждения.

Тип 4. Динамическая проблемная среда: динамическая предметная область; сущности представляются совокупностью атрибутов и их значений; состав сущностей неизменяемый; БЗ не структурирована; решаются динамические задачи анализа, используются специализированные исполняемые утверждения.

Тип 5. Динамическая проблемная среда: динамическая предметная область; сущности представляются в виде объектов; изменяемый состав сущностей; БЗ структурирована; решаются динамические задачи анализа и синтеза; используются общие и специализированные исполняемые утверждения.

Контрольные вопросы:

1. Сформулируйте параметры классификации экспертных систем.
2. Определите понятия интегрированного приложения, открытого приложения и распределенного приложения.
3. Поясните отличия коммерческой системы от промышленной и действующего прототипа от исследовательского
4. Перечислите основные параметры, определяющие свойства предметной области.

Практическая работа 2

Структура I типа динамических экспертных систем

Цель работы: изучение структуры I типа динамических экспертных систем

В начале восьмидесятых годов в исследованиях по искусственному интеллекту сформировалось самостоятельное направление, получившее название "экспертные системы" ОС). Цель исследований по ЭС состоит в разработке программ, которые при решении задач, трудных для эксперта-человека, получают результаты, не уступающие по качеству и эффективности решениям, получаемым экспертом. Исследователи в области ЭС для названия своей дисциплины часто используют также термин "инженерия знаний", введенный Е. Фейгенбаумом как "привнесение принципов и инструментария исследований из области искусственного интеллекта в решение трудных прикладных проблем, требующих знаний экспертов".

Программные средства (ПС), базирующиеся на технологии экспертных систем, или инженерии знаний (в дальнейшем будем использовать их как синонимы), получили значительное распространение в мире. Важность экспертных систем состоит в следующем:

• технология экспертных систем существенно расширяет круг практически значимых задач, решаемых на компьютерах, решение которых приносит значительный экономический эффект;

• технология ЭС является важнейшим средством в решении глобальных проблем традиционного программирования: длительность и, следовательно, высокая стоимость разработки сложных приложений; высокая стоимость сопровождения сложных систем, которая часто в несколько раз превосходит стоимость их разработки; низкий уровень повторной используемости программ и т.п.;

• объединение технологии ЭС с технологией традиционного программирования добавляет новые качества к программным продуктам за счет: обеспечения динамичной модификации приложений пользователем, а не программистом; большей "прозрачности" приложения (например, знания хранятся на ограниченном ЕЯ, что не требует комментариев к знаниям, упрощает обучение и сопровождение); лучшей графики; интерфейса и взаимодействия.

ЭС предназначены для так называемых неформализованных задач, т.е. ЭС не отвергают и не заменяют традиционного подхода к разработке программ, ориентированного на решение формализованных задач. Следуя А.Ньюэллу и М.Саймону, к неформализованным (ill-structured) будем относить такие задачи, которые обладают одной или несколькими из следующих характеристик:

• задачи не могут быть заданы в числовой форме;

• цели не могут быть выражены в терминах точно определенной целевой функции;

• не существует алгоритмического решения задач;

• алгоритмическое решение существует, но его нельзя использовать из-за ограниченности ресурсов (время, память).

В зависимости от цели, которая стоит перед ИС, база знаний, алгоритмы решения задачи, принятия решения, выработки управления могут, естественно, иметь различное представление, зависящее, в свою очередь, от характера решения

задач. Соответственно этому можно видеть три типа ДЭС. Структура ДЭС первого типа приведена на рисунке 1.

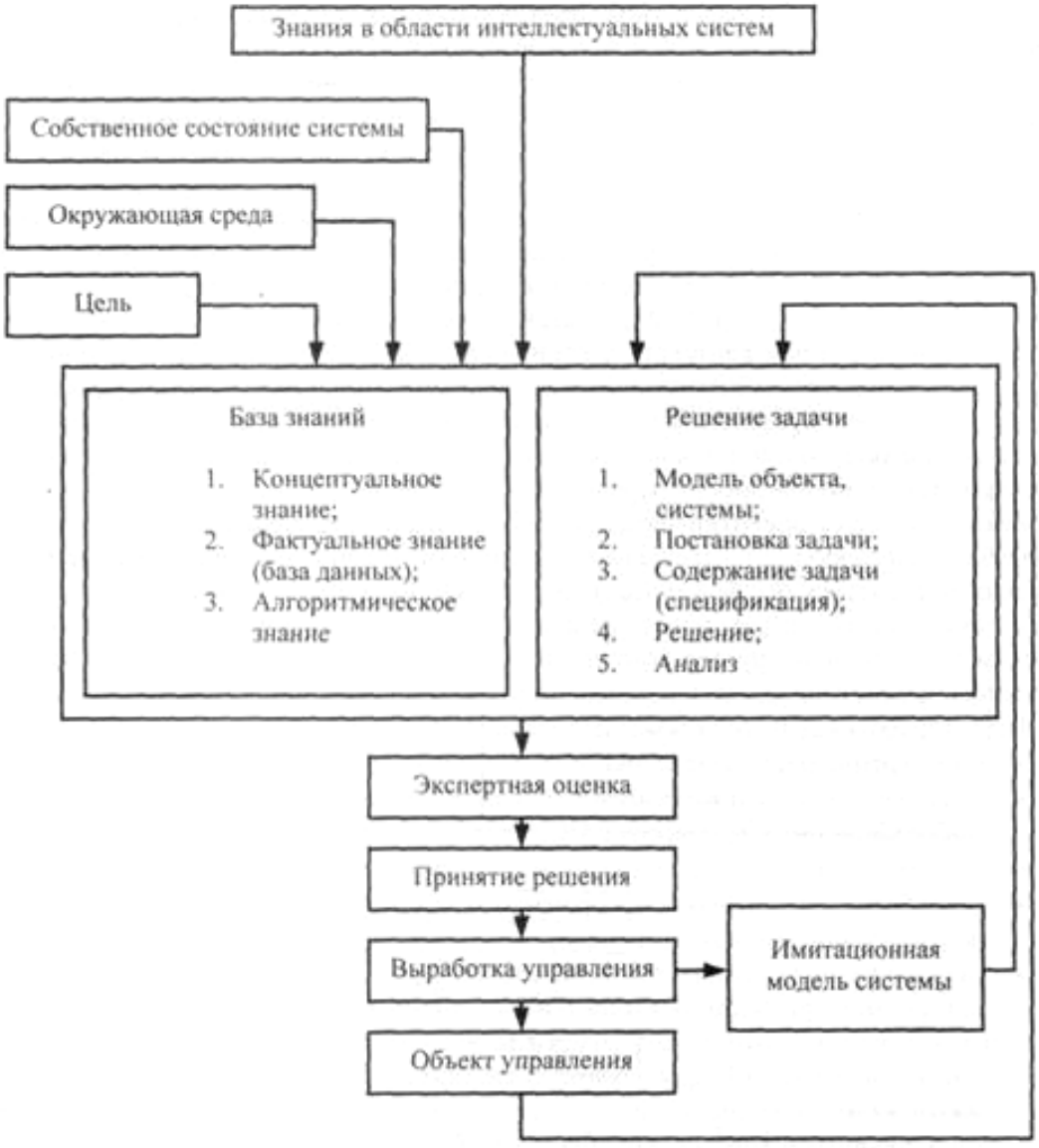


Рис. 1 – Структура ДЭС первого типа

Здесь предполагается, что концептуальные и фактуальные знания точно отражают процессы и сведения, относящиеся к некоторой предметной области.

Тогда решение задачи, возникающей в этой области, будет получено на основе строгих математических методов, в соответствии с постановкой и спецификацией. Результаты исследования решения и прогноз используются для получения экспертной оценки и принятия решения о необходимости управления. Затем на основе подходящего алгоритма управления, имеющегося в базе знаний, формируется управляющее воздействие.

Эффективность и непротиворечивость этого воздействия, прежде чем оно поступит на объект управления, оценивается с помощью имитационной математической модели. Оценка должна выполняться быстрее реальных процессов в ИС.

Однако ДЭС, реализующие принятие решения, представляют собой сложные программные комплексы, предназначенные для автоматического принятия решения или для помощи лицам, принимающим решения, и при оперативном управлении сложными системами и процессами, как правило, работают в условиях жестких временных ограничений.

Контрольные вопросы:

1. Сформулируйте отличия ЭС от традиционных систем обработки данных.
2. Назовите примеры успешного применения технологии ЭС.
3. Объясните основные причины успеха современной технологии ЭС.
4. Охарактеризуйте основные режимы работы ЭС.

Практическая работа 3

Структура II типа динамических экспертных систем

Цель работы: изучение структуры II типа динамических экспертных систем

Экспертные системы и системы искусственного интеллекта отличаются от систем обработки данных тем, что в них в основном используются символьный (а не числовой) способ представления, символьный вывод и эвристический поиск решения (а не исполнение известного алгоритма).

Специфика приложений экспертных систем по сравнению с другими системами искусственного интеллекта состоит в следующем. Экспертные системы применяются для решения только трудных практических (не игрушечных) задач. По качеству и эффективности решения экспертные системы не уступают решениям эксперта-человека. Решения экспертных систем обладают *"прозрачностью"*, т.е. могут, быть объяснены пользователю на качественном уровне (в отличие от решений, полученных с помощью числовых алгоритмов, и в особенности от решений полученных статистическими методами). Это качество экспертных систем обеспечивается их способностью рассуждать о своих знаниях и умозаключениях. Экспертные системы способны пополнять свои знания в ходе взаимодействия с экспертом.

Экспертная система работает в двух режимах: режиме приобретения знаний и в режиме решения задачи (называемом также режимом консультации или режимом использования ЭС).

В режиме приобретения знаний общение с ЭС осуществляет (через посредничество инженера по знаниям) эксперт. В этом режиме эксперт, используя компонент приобретения знаний, наполняет систему знаниями, которые позволяют ЭС в режиме решения самостоятельно (без эксперта) решать задачи из проблемной области. Эксперт описывает проблемную область в виде совокупности данных и правил. Данные определяют объекты, их характеристики и значения, существующие в области экспертизы. Правила определяют способы манипулирования с данными, характерные для рассматриваемой области.

Отметим, что режиму приобретения знаний в традиционном подходе к разработке программ соответствуют этапы алгоритмизации, программирования и отладки, выполняемые программистом. Таким образом, в отличие от традиционного подхода в случае ЭС разработку программ осуществляет не программист, а эксперт (с помощью ЭС), не владеющий программированием.

В режиме консультации общение с ЭС осуществляет конечный пользователь, которого интересует результат и (или) способ его получения. Необходимо отметить, что в зависимости от назначения ЭС пользователь может не быть специалистом в данной проблемной области (в этом случае он обращается к ЭС за результатом, не умея получить его сам), или быть специалистом (в этом случае пользователь может сам получить результат, но он обращается к ЭС с целью либо ускорить процесс получения результата, либо возложить на ЭС рутинную работу). Следует подчеркнуть, что термин "пользователь" является многозначным, так как использовать ЭС кроме конечного пользователя может и эксперт, и инженер по знаниям, и программист. Поэтому когда хотят подчеркнуть, что речь идет о том, для кого делалась ЭС, используют термин "конечный пользователь".

В основе ИС лежат объектно-ориентированная база знаний (ОО-технология БЗ) и механизм вывода, способный оперировать с правилами, в которых явным образом отражено время (РВ - механизм вывода). Во внутреннем кольце расположены компоненты, обеспечивающие моделирование, анимацию, активную графику, механизм общих правил и т.д. Во внешнем кольце отражены технологии и требования, обязательные в современных ИС для создания ЭС.

Механизм вывода для динамических проблемных сред дополнительно содержит:

• планировщик, обеспечивающий в соответствии с приоритетами всю деятельность ЭС;

• средства, гарантирующие получение лучшего решения в условиях ограниченности ресурсов;

• систему поддержания истинности значений переменных, изменяющихся во времени.

Динамические ИС по отношению к системе моделирования могут быть охарактеризованы следующим образом:

1) система моделирования отсутствует;

2) существует система моделирования общего назначения, являющаяся частью ИС;

3) существует специализированная система моделирования, являющаяся внешней по отношению к ИС, на котором реализуется ЭС.

Системы моделирования общего назначения используют для описания модели алгебраические, разностные и дифференциальные уравнения (обычно первого порядка). Как правило, в систему моделирования включается несколько альтернативных методов. Например, по выбору пользователя уравнение решается либо методом Рунге-Кутта, обеспечивающим лучшую точность, либо методом Эйлера, обеспечивающим меньшую точность, но более быстрое получение решения. Специализированные системы моделирования учитывают специфику приложения (например, модель химического предприятия).

В отличие от ДЭС первого типа, предназначенных для поиска оптимального решения и базирующихся на строгих математических методах и моделях оптимизации, ДЭС второго типа в основном ориентированы на решение трудно формализуемых задач в отсутствие полной и достоверной информации (рис. 2). Здесь используются экспертные модели, построенные на основе знаний экспертов — специалистов в данной проблемной области, и эвристические методы поиска решения.

Одной из основных проблем при проектировании ДЭС второго типа является выбор формального аппарата для описания процессов принятия решений и построение на его основе модели принятия решений, адекватной проблемной области (семантически корректной). В качестве такого аппарата обычно используют продукционные системы. Однако основные исследования ведутся в контексте алгоритмической (детерминированной) трактовки продукционной системы с присущей ей последовательной схемой поиска решения.

Получающиеся в результате модели зачастую неадекватны реальным проблемным областям, характеризующимся недетерминизмом процесса поиска решения. Выход из такого положения — параллелизм при поиске.

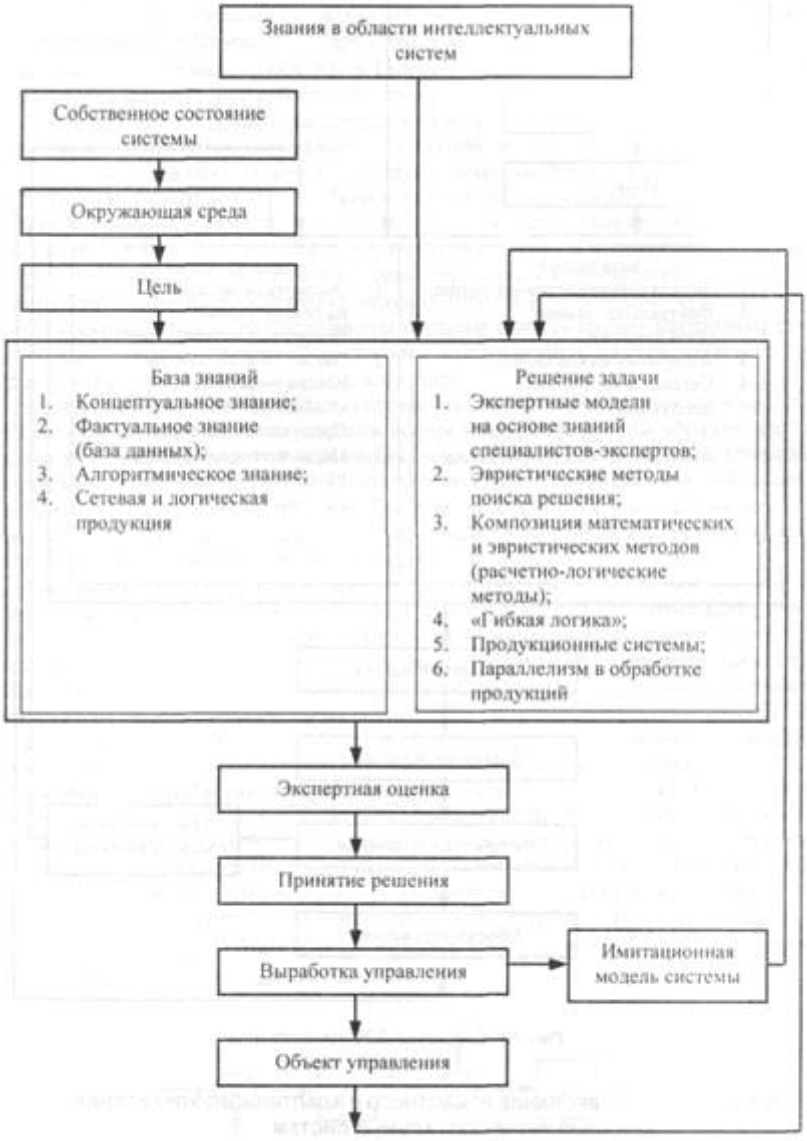


Рис. 5.8. Структура ДЭС второго типа

Рис. 2 – Структура ДЭС второго уровня

Контрольные вопросы:

Сформулируйте отличия ЭС от традиционных систем обработки данных.
Назовите примеры успешного применения технологии ЭС.
Объясните основные причины успеха современной технологии ЭС.
Охарактеризуйте основные режимы работы ЭС.

Практическая работа 4

Расчетно-логический динамических экспертных систем III типа

Цель работы: изучение расчетно-логических динамических экспертных систем III типа

ИС, ориентированные на приобретение знаний, в настоящее время обычно не выделяются в самостоятельный продукт, а поставляются в составе ИС общего назначения. Тем не менее коммерческие ИС приобретения знаний в США существуют и могут быть охарактеризованы следующим образом.

1. Средства приобретения знаний, основанные на деревьях решений. Средства этого типа обычно включаются в состав ИС общего назначения. Типичными примерами являются ИС общего назначения для ПК Procedural Consultant (фирма TI) и ИС общего назначения для рабочих станций DEC и VAX Dession Expert (фирма DEC).

2. Индуктивные средства приобретения знаний. Эти средства либо включаются в состав ИС общего назначения (например, VP-Expert, 1st-CLASS FUSION, KDS 2&3 и др.), либо распространяются как самостоятельное средство (например, продукт BEAGLE фирмы VRS Consulting для PC и для VAX).

3. Средства приобретения знаний, базирующиеся на психологической теории. Наиболее широкое использование имеет ИС ETS (фирма Boeing), реализованное на символьной ЭВМ и базирующееся на методе репертуарных решеток Дж. Келли. Данное средство фирмой не продается, а используется только для внутренних приложений фирмы (известно несколько сотен приложений средства).

4. Средства приобретения знаний, ориентированные на конкретные ИС общего назначения. Типичным примером является ИС KAT, предназначенное для помощи в создании БЗ для ИС Level 5, и средство Nextra, которое упрощает приобретение знаний для ИС Nexpert Object. Nextra сочетает индуктивный метод и метод репертуарных решеток и реализовано на ПК Macintosh.

5. Средства приобретения знаний общего назначения. Данные средства не ориентируются ни на какое ИС, они используются не только для создания БЗ ЭС. Их цель помочь разработчику в накоплении, редактировании и управлении знаниями о конкретной проблеме. Типичным примером является ИС CAMEO (Arthur D.Little - ADL). Это ИС не продается, но доступно клиентам, работающим совместно с ADL.

В последнее время на основе динамических ИС начинают создаваться ИС для интеллектуального имитационного моделирования, используемые в реинжиниринге (реорганизации) бизнес-процессов (БПР). Интерес к ИС этого типа инициируется тем, что в отличие от статических ИС и ЭС, используемых, как указано ранее, для БПА, т.е. для автоматизации текущего состояния бизнеса, ИС для БПР используются для решения существенно более значимых и сложных задач, т.е. для "фундаментального переосмысления и радикального перепроектирования деловых процессов для достижения существенных улучшений в главных показателях деятельности компаний, таких, как стоимость, качество, услуги и темпы".

Реально следует ориентироваться на объединение ДЭС первого и второго типа в расчетно-логическую ДЭС третьего типа, где база знаний сочетает описание в виде строгих математических формул с информацией экспертов, а также со-

ответственно — математические методы поиска решения с нестрогими эвристическими методами, причем вес того или другого компонента определяется возможностью адекватного описания предметной области и способом отыскания решения (рис. 3).

При разработке ДЭС возникают следующие проблемы:

- определение состава базы знаний и ее формирование;
- разработка новых и использование известных теорий и методов для описания информационных процессов в ИС;
- разработка способов представления и организации использования знаний;
- разработка алгоритмов и программного обеспечения с распараллеливанием и использованием «гибкой логики»;
- отыскание подходящих вычислительных сред для реализации параллельных алгоритмов при формировании ДЭС.

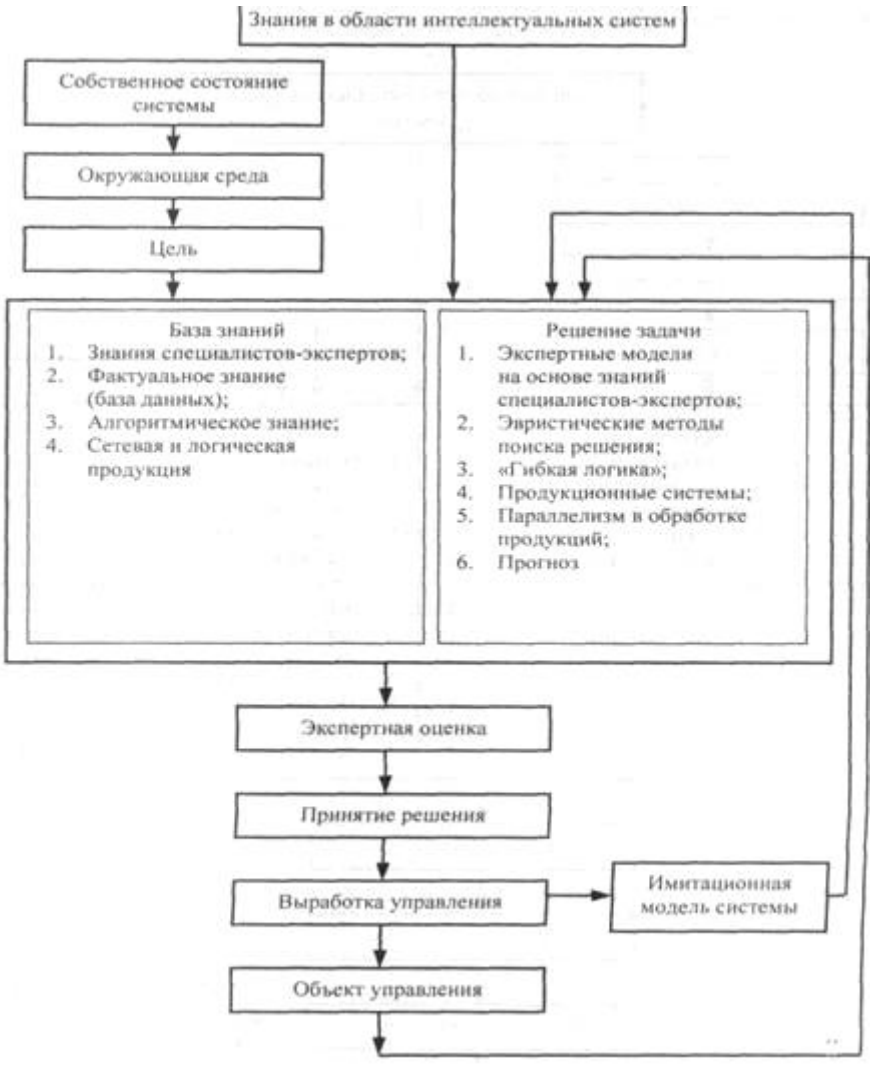


Рис. 3 – Структура ДЭС третьего уровня

Наряду с изложенным важно отметить, что ДЭС должны обладать свойством адаптации к динамической проблемной области, способностью ввода новых элементов и связей в описание ситуаций, изменения правил и стратегии функционирования объектов в процессе принятия решения и выработки управления, работы с неполной, нечеткой и противоречивой информацией и т.д.

Динамические экспертные системы функционируют в составе ИС, имеющих обратные связи, и поэтому важно обеспечить устойчивую работу таких ИС.

С традиционных позиций можно считать, что длительность реакции ДЭС на входные воздействия, т.е. время, затрачиваемое на обработку входной информации и выработку управляющего воздействия, есть чистое запаздывание. На основе частотного анализа можно оценить изменение фазовых свойств системы и тем самым определить запас устойчивости. При необходимости можно произвести коррекцию системы посредством фильтров.

Однако с точки зрения классической теории управления ИС являются многообъектными многосвязными системами, анализ устойчивости которых обычными способами весьма затруднителен.

Контрольные вопросы:

1. На какие типы подразделяются динамическая экспертная систем?
2. Чем отличается типы ДЭС?
3. Какие проблемы при разработке ДЭС возникают?

Практическая работа 5

Проблемы создания динамических экспертных систем. Определение состав и формирование базы знаний

Цель работы: изучение проблемы создания динамических экспертных систем. Определение состав и формирование базы знаний.

ИС можно классифицировать по тому, какую технологию разработки ЭС допускает это инструментальное средство. В отличие от распространенного мнения о наличии единой технологии создания ЭС можно выделить, по крайней мере, четыре значительно отличающиеся технологии:

1) подход, базирующийся на поверхностных знаниях;

2) структурный подход;

3) подход, базирующийся на глубинных знаниях;

4) смешанный подход, базирующийся на использовании поверхностных и глубинных знаний.

Подход, базирующийся на поверхностных знаниях, применяется к сложным задачам, которые не могут быть точно описаны. Этот подход заключается в получении от эксперта фрагментов знаний (часто эвристических), которые релевантны решаемой задаче. При этом не предпринимается никаких попыток систематического или глубинного изучения области, что предопределяет использование поиска в пространстве состояний в качестве универсального механизма вывода. Обычно в ЭС, использующих данный подход, в качестве способа представления выбираются правила. Условие каждого правила определяет образец некоторой ситуации, при соблюдении которой правило может быть выполнено. Поиск решения состоит в выполнении тех правил, образцы которых сопоставляются с текущими данными. При этом предполагается, что в процессе поиска решения последовательность формируемых таким образом ситуаций не оборвется до получения решения, т.е. не возникнет неизвестной ситуации, которая не сопоставится ни с одним правилом. Данный подход с успехом применяется к широкому классу приложений, однако он оказывается неэффективным в тех приложениях, когда задача может быть заранее структурирована или при решении задачи может быть использована Некоторая модель.

Структурный подход к построению ЭС обусловлен тем, что для ряда приложений применение только техники поверхностных знаний не обеспечивает решения задачи. Действительно, использование поиска в качестве механизма вывода в неструктурированной базе знаний может приводить к. ненадежным и (или) некачественным решениям. Структурный подход к построению ЭС подобен структурному программированию. Однако применительно к ЭС речь не идет о том, что структурирование должно довести задачу до алгоритма (как в традиционном программировании), а предполагается, что часть задачи решается с помощью поиска. Структурный подход в различных приложениях целесообразно сочетать с поверхностным или глубинным.

В ***глубинном подходе*** компетентность ЭС базируется на модели той проблемной среды, в которой эта ЭС работает. Модель может быть определена различными способами (декларативно, процедурно). Необходимость в ряде приложений использовать модели вызвана стремлением исправить несовершенство по-

верхностного подхода, возникающего при отсутствии правил, удовлетворяющих текущей ситуации в рабочей памяти. Глубинные ЭС кроме возможностей поверхностных ЭС обладают способностью при возникновении неизвестной ситуации определить с помощью некоторых общих принципов, справедливых для области экспертизы, какие действия следует выполнить.

Глубинный (модельный) подход требует явного описания структуры и взаимоотношений между различными сущностями области. При этом подходе необходимо использовать ИС, обладающие мощными моделирующими возможностями: объекты с присоединенными процедурами, иерархическое наследование свойств, активные знания (программирование, управляемое данными), передача сообщений объектам (объектно-ориентированное программирование) и т.п.

Смешанный подход в общем случае может сочетать поверхностный, структурный и глубинный подходы. Например, поверхностный подход может быть использован для поиска адекватных знаний, которые затем используются некоторой глубинной моделью.

В соответствии с введенными выше параметрами классификации проблемных сред и ИС можно выделить следующие типы ИС, соответствующие типам проблемных сред.

Тип 1. ИС применяется для создания ЭС в статических проблемных средах типа 1. ИС, как правило, являются оболочкой (средой) и характеризуются следующими особенностями: сущности представляются совокупностью атрибутов и значений, БЗ не структурирована, ИС ориентировано на решение статических задач анализа. Используется программирование, ориентированное на правила. Правила конкретные. Правила и данные вводятся экспертом или создаются автоматически из данных индуктивными методами. Сеть или дерево вывода компилируется в режиме приобретения знаний. Направление поиска решения: от цели к данным. Сопоставление не используется. При разработке ЭС используется технология, базирующаяся на поверхностных знаниях. Примеры ИС: 1-st class, ИЛИС и др.

Тип 2. ИС применяется для создания ЭС в статических средах типа 2. ИС, как правило, - оболочка (среда) со следующими особенностями: сущности представляются в виде атрибутов или вырожденных объектов (фреймов), структурирование БЗ слабо выражено, ориентация на решение статических задач анализа. Используются программирование, ориентированное на правила, и зачатки объектно-ориентированного программирования. Правила обычно конкретные, но в ряде ИС данного типа используются общие правила. Правила и структуры данных вводятся экспертом. В ИС этого и последующих типов индуктивные методы обычно не применяются, так как с их помощью можно получать только простые правила. Процесс получения решения обычно состоит в генерации сети вывода в режиме приобретения знаний. Направление поиска решения: от цели к данным. Сопоставление обычно не используется. При разработке ЭС используются поверхностный и структурный подходы.

Примеры ИС данного типа: ЭКО, Guru, Leonardo и др.

Тип 3. ИС применяется для создания ЭС в статических средах типа 3. ИС, как правило, - оболочка (среда) со следующими особенностями: сущности представляются в виде объектов, БЗ структурирована, ориентация на статические задачи анализа и синтеза. Используется программирование, ориентированное на правила,

объектно-ориентированное и процедурное программирование. Правила конкретные и общие, т.е. используется операция сопоставления. Процесс получения решения включает: генерацию сети вывода и поиска или выработку предположений, генерацию сети вывода и поиска. Направление поиска решения по выбору: от цели к данным и от данных к целям. Используется (по выбору) поиск в ширину или в глубину. При разработке ЭС используются структурный, поверхностный и глубинный подходы. Данные и правила вводятся на ограниченном ЕЯ с использованием изображений.

Примеры ИС данного типа: Nexpert Object, ART, Level 5 Object, ProKappa, ADS.

Тип 4. ИС применяется для создания ЭС в динамических средах типа 4. ИС представляет собой инструментарий, т.е. совокупность компонентов, из которых программируется ЭС. Сущности представляются в виде атрибутов и значений, БЗ не структурирована, решаются динамические задачи анализа в реальном времени, правила конкретные. Используется как поиск от цели, так и от данных. Сопоставление не используется. Система моделирования отсутствует. При разработке ЭС используются поверхностный и структурный подходы.

Примеры ИС данного типа: TDC Expert, Activation Frame Work, Rocky.

Тип 5. ИС применяется для создания ЭС в динамических средах типа 5 ИС, как правило, - оболочка (среда) со следующими особенностями сущности представляются классами объектов и их экземплярами, БЗ структурирована, решаются динамические задачи анализа и синтеза. Используется программирование, ориентированное на правила, объектно-ориентированное программирование и процедурное программирование. Правила конкретные и общие. Процесс получения решения состоит из генерации сети вывода и поиска. Направление поиска задается пользователем (от данных или от целей). Используется поиск в глубину и ширину.

Большинство существующих инструментальных средств для создания ЭС предоставляют разработчику свободу в выборе процедурного или декларативного подхода к представлению знаний. Процедура описывает строгую последовательность действий - по преимуществу линейный алгоритм решения задачи. В случае, когда операции строго регламентированы и носят явно последовательный порядок исполнения, процедурный подход является более предпочтительным. Однако во многих системах реального времени в ответ на текущий анализируемый набор данных может возбуждаться большое количество управляющих последовательностей, что делает целесообразным применение продукционных правил.

В объектно-ориентированных расширениях процедурных языков достаточно развита концепция функций-методов для ассоциации поведенческих аспектов объектов с иерархией классов. Методы позволяют инкапсулировать специфику исполнения общих операций над классом в его описании, что является следующим шагом в локализации внесения изменений при необходимости перепрограммирования или расширения системы по сравнению с модульной схемой организации системы. Современные средства для создания ЭС в полной мере обладают указанными возможностями и распространяют этот подход на продукционные правила экспертной системы.

Ассоциирование продукционных правил с иерархией классов осуществляется за счет использования общих правил, в качестве префикса которых используется

ссылка на класс, к которому данное правило применимо. Указанный префикс с точки зрения декларативного представления знаний семантически близок к квантору всеобщности в исчислении предикатов. В процедурной интерпретации наличие префикса, связывающего продукцию с классом, вызывает необходимость перебора всех экземпляров указанного в префиксе класса и его подклассов и проверки истинности антецедента для атрибутов каждого из экземпляров. Очевидно, что применение общих правил на стадии исполнения увеличивает накладные расходы и поэтому следует как можно более точно описывать область действия общих утверждений за счет дополнительных ограничивающих условий.

Ввод общих знаний средствами системы К-ЭКО

Комплекс ЭКО включает три компонента.

Ядром комплекса является *интегрированная оболочка* экспертных систем ЭКО, которая обеспечивает быстрое создание эффективных приложений для решения задач анализа в статических проблемных средах типа 1 и 2.

При разработке средств представления знаний оболочки преследовались две основные цели: эффективное решение достаточно широкого и практически значимого класса задач средствами персональных компьютеров; гибкие возможности по описанию пользовательского интерфейса и проведению консультации в конкретных приложениях. При представлении знаний в оболочке используются специализированные (частные) утверждения типа "атрибут - значение" и Частные правила, что позволяет исключить ресурсоемкую операцию сопоставления по образцу и добиться эффективности разрабатываемых приложений. Выразительные возможности оболочки удалось существенно расширить за счет интегрированности, обеспечиваемой путем вызова внешних программ через сценарий консультации и стыковки с базами данных (ПИРС и dBase IV) и внешними программами. В оболочке ЭКО обеспечивается слабая структуризация БЗ за счет ее разделения на отдельные компоненты для решения отдельных подзадач в проблемной среде - модели (понятию "модель" ЭКО соответствует понятие "модуль" базы знаний системы G2).

С точки зрения технологии разработки ЭС оболочка поддерживает подходы, основанные на поверхностных знаниях и структурировании процесса решения.

Существенным расширением возможностей оболочки ЭКО являются средства конкретизации знаний, предоставляемые системой К-ЭКО. Система обеспечивает диалоговый ввод и тестирование общих и конкретных знаний с последующей их компиляцией в формат БЗ оболочки ЭКО (без сценариев). К-ЭКО позволяет использовать оболочку ЭКО для решения задач в проблемных средах типа 2.

Представление знаний основано на фреймовой модели. Общие знания о сущностях реального мира представляются с помощью структур данных, называемых в системе *классами объектов* (аналоги фреймов). Класс представляет собой структуру, объединяющую атрибуты (характеристики) некоторой сущности, множество (объединение сущностей) и методы их определения (т.е. правила вывода). Конкретные знания представляются с помощью экземпляров объектов (в дальнейшем - просто объектов) и называются *фактами.* Знания о значениях атрибутов объектов - это *утверждения* в терминах оболочки ЭКО. Модель в системе

К-ЭКО включает базу знаний и базу фактов. В базе знаний представлены общие знания, а в базе фактов - конкретные.

БЗ содержит описание предметной области в терминах классов. Каждый класс описывается следующим набором:

• множеством вложенных конкретных объектов;

• множеством имен атрибутов (слотов) с описанием их типа и значения;

• множеством имен множеств с описанием их типа и значения;

• множеством общих правил.

Вложенные конкретные объекты имеют тот же синтаксис, что и конкретные объекты в базе фактов. При создании конкретного объекта, в общем описании которого есть вложенные объекты, вложенные объекты размножаются. Вложенные объекты "видны" только лишь внутри того объекта, в котором они описаны.

При описании типа значения атрибута возможны следующие варианты:

• множество символов (утверждений);

• число целого или вещественного типа;

• множество альтернативных утверждений или ссылок;

• множество дистрибутивных утверждений.

Допускается присваивание значения атрибута внутри описания класса.

Общие правила имеют те же типы, что и в оболочке ЭКО. Особенностью общего правила являются список описаний переменных атрибутов (имена и типы), используемых в правиле, и наличие этих переменных в условии применения.

База фактов содержит список конкретных объектов, каждому из которых соответствует один из общих объектов, являющийся прототипом для его создания.

В ходе компиляции общих знаний система К-ЭКО осуществляет частичный вывод решения, если для этого имеются соответствующие факты

Формирование баз знаний на основе обучающей выборки средствами системы ИЛИС

Система ИЛИС ориентирована на решение задач эвристической классификации в проблемных средах типа 1 на основе индуктивного вывода по примерам обучающей выборки. Входное множество примеров отражает опыт эксперта и соответствует множеству разумных решений некоторой задачи в проблемной области (например, диагностики по симптомам).

Примеры представляются в виде обычных таблиц показателей, сходных с таблицами СУБД реляционного типа (атрибут, кортеж, значение) В ходе индуктивного вывода они анализируются и обобщаются для построения правил проблемной области, организованных в виде дерева решений, при этом используется метод машинного обучения, основанный на CLS-подходе.

Примеры вводятся в таблицу, столбцам которой соответствуют атрибуты проблемной области. Один из столбцов описывается как целевой (например, "диагноз"). Подобная таблица называется *базой опыта*. Ввод осуществляется в диалоговом режиме, при этом система проверяет примеры на корректность и непротиворечивость.

После ввода имеющихся примеров (или расширения существующей базы опытов) строится (модифицируется) БЗ, т.е. происходит процесс обучения. БЗ может строиться как в автоматическом, так и в ручном режиме. При этом в обоих

режимах гарантируется, что получаемые знания не будут противоречить обучающей выборке.

Знания формируются в виде дерева решений, которое представляет собой сеть вывода для консультации. Дерево решений в системе ИЛИС - это помеченный связанный граф без циклов, внутренние вершины которого помечаются именами нецелевых атрибутов, дуги -именами значений этих атрибутов, листьевые вершины - значениями целевого атрибута. Множество значений атрибутов, которым соответствует путь от корня дерева к листу, назовем *правилом.* Любому примеру из обучающей выборки соответствует правило, причем разным примерам, принадлежащим к различным классам, не может соответствовать одно правило. В ходе построения дерева система может давать диагностику о неполноте генерируемой модели.

С помощью разнообразных опций система ИЛИС обеспечивает различные стратегии обучения, например наращивание дерева решений в глубину или в ширину, выбор атрибута классификации по критерию энтропии или в соответствии с предпочтениями эксперта и т.д.

Для проверки адекватности модели необходимо провести ее тестирование в режиме консультации средствами системы ИЛИС или в режиме просмотра базы знаний.

В ходе тестирования в режиме консультации экспертная система должна правильно решить достаточное количество контрольных задач, предложенных экспертом. Если эксперт не удовлетворен удельным весом правильных решений в контрольном наборе задач, он должен продолжить процесс накопления опыта, включив в базу опыта примеры, которые были решены неправильно, а затем повторить процесс обучения системы.

В ходе тестирования в режиме просмотра БЗ эксперт должен проанализировать правильность созданных системой правил БЗ и составить контрпримеры для правил, корректность которых вызывает сомнения. Затем необходимо ввести эти контрпримеры в базу опыта и заново обучиться.

Процесс тестирования повторяется циклично до тех пор, пока не достигается приемлемая достоверность консультации (базы знаний).

Модели системы ИЛИС могут быть автоматически переведены в формат оболочки ЭКО. Подобный перевод необходим в тех случаях, когда получаемые модели являются составными частями более сложных прикладных ЭС, создаваемых на базе комплекса ЭКО. Кроме того, это позволяет воспользоваться широкими и гибкими возможностями по описанию пользовательского интерфейса, которые предоставляет оболочка ЭКО.

Контрольные вопросы:

1. Сформулируйте параметры классификации экспертных систем.
2. Укажите главные характеристики типов задач, решаемых экспертной системой.
3. Назовите основные типы проблемных сред и ИС
4. Назовите состав структуры базы знаний оболочки ЭКО.
5. Приведите пример сети вывода в оболочке ЭКО.
6. Охарактеризуйте стратегии управления, применяемые в оболочке ЭКО.

7. Объясните механизм решения задач в ЭКО.

8. Опишите средства ввода общих знаний в комплексе ЭКО.

9. Объясните механизм формирования баз знаний на основе обучающей выборки в системе ИЛИС.

Практическая работа 6
Характеристики информационных процессов в интеллектуальных системах и разработка новых методов

Цель работы: изучение характеристик информационных процессов в интеллектуальных системах и разработка новых методов

Бизнес-процесс "реинжиниринг", или БПР (в оригинале - "Business process reengineering"), начиная с 1990 г., вызывает активный интерес специалистов в области менеджмента и информационных технологий (ИТ). С 1994 г. в США проводятся ежегодные конференции по БПР. В настоящее время БПР взят на вооружение почти всеми ведущими компаниями мира. В частности, по данным Ernst & Young, 100 крупнейших банков Северной Америки затратят в 1997 г. около 2,9 млрд. дол. на реинжиниринг своих подразделений. За последние полтора года правительство США начало более 200 проектов по реинжинирингу. М. Хаммер и Дж. Чампи определяют реинжиниринг как "фундаментальное переосмысление и радикальное перепроектирование бизнес-процессов компаний для достижения коренных улучшений в основных актуальных показателях их деятельности: стоимость, качество, услуги и темпы". Подчеркнем, что речь идет не о небольшом усовершенствовании бизнес-процессов компаний, таком, например, как на 10 - 100%, а о кардинальном повышении их эффективности в десятки раз. При этом реинжиниринг рассматривается как необходимое условие выживания современных компаний в условиях жесткой конкурентной борьбы на мировом рынке.

Решением проблемы являются смена базовых принципов организации компаний и переход к ориентации не на функции, а на процессы. Из всех концепций менеджмента, основанных на процессах, БПР рассматривается как наиболее эффективная концепция: М. Хаммер, автор термина "реинжиниринг", считает появление БПР революцией в бизнесе, которая знаменует отход от базовых принципов построения компаний, предложенных 200 лет назад А.Смитом, и превращает конструирование бизнеса в инженерную деятельность. Возможность такой революции обусловлена в первую очередь новейшими достижениями в области информационных технологий, в частности технологии динамических экспертных систем.

На самом деле информационные технологии позволяют изменить базовые правила организации работы (табл. 1).

Можно выделить два вида влияния информационных технологий на перестройку деятельности компаний и соответственно две группы технологий, имеющих пересечение (рис. 1). Технологии первой группы обеспечивают проведение БПР за счет автоматизации работ по реинжинирингу. Технологии второй группы обеспечивают появление новых процессов, позволяющих перейти к новым правилам работы в организациях.

Таблица 1 – Влияние информационных технологий на переход к новым правилам работы компаний

Прежнее правило	Технология	Новое правило
Информация может появляться в одно время в одном месте	Распределенные базы данных	Информация может появляться одновременно в разных местах тогда, когда она необходима
Сложную работу могут выполнять только эксперты	Экспертные системы	Работу эксперта может выполнять специалист по общим вопросам
Необходимо выбирать между централизацией и децентрализацией бизнеса	Телекоммуникационные сети	Бизнес может пользоваться преимуществами централизации и децентрализации одновременно
Все решения принимают менеджеры	Средства поддержки принятия решений, доступ к базам данных, средства моделирования	Принятие решений становится частью работы каждого сотрудника (иерархическое принятие решений)
Для получения, хранения, поиска и передачи информации требуется офис	Беспроводная связь и переносимые компьютеры	Сотрудники могут посылать и получать информацию из того места, где они находятся
Лучший контакт с потенциальным покупателем - личный контакт	Интерактивный видеодиск	Лучший контакт с потенциальным покупателем - эффективный контакт
Чтобы найти некоторый объект, необходимо знать, где он находится	Автоматическое индексирование и отслеживание	Объекты сами информируют о своем местонахождении
Планы работ пересматриваются и корректируются периодически	Высокопроизводительные компьютеры	Планы пересматриваются и корректируются оперативно, по мере необходимости

Чтобы пояснить, каким образом проведение БПР повышает эффективность работы компании, рассмотрим, как реинжиниринг изменяет перепроектируемые бизнес-процессы.

Несколько рабочих процедур объединяются в одну. Наиболее характерным свойством перепроектированных процессов является отсутствие технологии "сборочного конвейера", в рамках которой на каждом рабочем месте выполняются простые задания, или рабочие процедуры. Вместо этого процедуры, выполнявшиеся различными сотрудниками, интегрируются в одну - *горизонтальное сжатие процесса.*

На практике далеко не всегда удается сжать все шаги процесса к одной работе, выполняемой одним сотрудником. В этом случае создается команда, которая несет ответственность за данный процесс. Наличие в команде нескольких человек неизбежно приводит к некоторым задержкам и ошибкам, возникающим при передачи работы между членами команды. Однако потери здесь значительно меньше, чем при традиционной организации работ, когда исполнители процесса подчиняются различным подразделениям компании (возможно, располагающимся на различных территориях). Кроме того, при традиционной организации работ трудно (а иногда и невозможно) определить ответственного за быстрое и качественное выполнение работы.

Рис.1. Информационные технологии в БПР

Сравнительные оценки, выполненные компаниями, которые провели реинжиниринг, показывают, что переход от традиционной организации работ к выполнению процесса одним сотрудником уменьшает количество людей и ускоряет выполнение процесса примерно в 10 раз. К другим достоинствам горизонтального сжатия процессов относятся следующие:

• уменьшается количество ошибок и отпадает необходимость в специальной группе сотрудников для устранения этих ошибок;

• улучшается управляемость за счет уменьшения количества людей и четко распределенной ответственности между ними.

Исполнители принимают самостоятельные решения. В ходе реинжиниринга компании осуществляют не только горизонтальное, но и *вертикальное сжатие процессов.* Вертикальное сжатие происходит за счет того, что в тех точках процесса, где при традиционной организации работ исполнитель должен был обращаться к управленческой иерархии для принятия решений, он принимает решения самостоятельно.

При традиционной организации работ, ориентированной на выпуск массовой продукции, исходили из предположения, что исполнители не имеют ни времени, ни склонности, ни глубоких и всесторонних знаний, необходимых для принятия решений. Реинжиниринг отбрасывает это предположение, что представляется естественным при отказе от массового производства и при современном уровне образования. Наделение сотрудников большими полномочиями и увеличение роли каждого из них в работе компании приводят к значительному повышению их отдачи.

Шаги процесса выполняются в естественном порядке. Реинжиниринг процессов освобождает от линейного упорядочивания рабочих процедур, свойственного традиционному подходу, позволяя распараллеливать процессы там, где это возможно.

Процессы имеют различные варианты исполнения. Традиционный процесс ориентирован на производство массовой продукции для массового рынка, поэтому он должен исполняться единообразно независимо от исходных условий

(т.е. при всех возможных входах процесса). Высокая динамичность рынка приводит к тому, что процесс должен иметь различные версии исполнения в зависимости от конкретной ситуации, состояния рынка и т.д.

Новые (перепроектированные) процессы, имеющие различные версии исполнения, начинаются с некоторого проверочного шага, определяющего, какая версия процесса наиболее подходит в данном конкретном случае (например, в простом случае выполняется автоматизированная процедура, в нетривиальном случае привлекается специалист и в сложном случае специалист приглашает экспертов).

Традиционные процессы обычно оказываются довольно сложными, так как они учитывают различные исключения и частные случаи. Новые процессы в отличие от традиционных являются более ясными и простыми, так как каждый вариант ориентирован только на одну соответствующую ему ситуацию.

Работа выполняется в том месте, где это целесообразно. В традиционных компаниях работа организуется по функциональным подразделениям: отдел заказов, транспортный отдел и т.п., и если конструкторскому отделу требуется новый карандаш, то он обращается с заявкой в отдел заказов. Отдел заказов находит производителя, договаривается о цене, размещает заказ, осматривает товар, оплачивает его и передает конструкторам. Описанный процесс является достаточно расточительным и медленным. Проведенный в одной из компаний США анализ показал [3], что при традиционном распределении работ внутренние затраты компании на приобретение батарейки стоимостью 3 дол. составили 100 дол. Кроме того, было установлено, что 35% всех заказов составляют заказы стоимостью менее 500 дол. После проведения реинжиниринга отделы перешли к самостоятельному заказу дешевых товаров. Итак, реинжиниринг распределяет работу между границами подразделений, устраняя излишнюю интеграцию, что приводит к повышению эффективности процесса в целом.

Уменьшается количество проверок и управляющих воздействий. Проверки и управляющие воздействия непосредственно не производят материальных ценностей, поэтому задача реинжиниринга - сократить их до экономически целесообразного уровня. Традиционные процессы насыщены подобными шагами, единственное назначение которых контроль за соблюдением исполнителями предписанных правил. Так, например, перед выполнением заказа соответствующий отдел компании проверяет право клиента сделать данный заказ, а также подлинность подписи клиента и финансовую состоятельность его подразделения или организации. При общей целесообразности проверок многие компании не задумываются над тем, сколько стоит проведение этих проверок. На практике довольно часто оказывается, что стоимость проверок и управляющих воздействий превосходит стоимость потерь, которые бы имели место при отсутствии проверок.

Реинжиниринг предлагает более сбалансированный подход. Вместо проверки каждого из выполняемых заданий перепроектированный процесс часто объединяет эти задания и осуществляет проверки и управляющие воздействия в отложенном режиме, что заметно сокращает время и стоимость проверок.

Минимизируется количество согласований. Еще один вид работ, не производящих непосредственных ценностей для заказчика, - это согласования. Задача реинжиниринга состоит в минимизации согласований путем сокращения внешних точек контакта и, как следствие, стирание граней между функциональными подразделениями.

CASE-технологии использовались в реинжиниринге практически с самого начала. Однако их ориентация на разработчиков информационных систем привела к тому, что в настоящее время их начинают объединять с другими современными технологиями, в первую очередь с объектно-ориентированными.

Имитационное моделирование обеспечивает наиболее глубокое представление моделей для непрограммирующего пользователя, а также наиболее полные средства анализа таких моделей. Модели создаются в виде потоковых диаграмм, в которых представлены основные рабочие процедуры в компании и описано их поведение, а также информационные и материальные потоки между ними.

Чтобы преодолеть эти проблемы, в настоящее время начинают использовать методы *инженерии знаний.* С их помощью можно непосредственно представлять в моделях плохо формализуемые знания менеджеров о бизнес-процессах, в частности рабочих процедурах. Кроме того, решается проблема создания интеллектуального интерфейса конечного пользователя со сложными средствами анализа моделей. *Средства быстрой разработки приложений* позволяют сокращать время создания поддерживающих информационных систем и, следовательно, необходимы не только в ходе реинжиниринга компании, но и на этапе эволюционного развития, сопровождающегося постоянными модификациями и улучшениями информационных систем компании.

Контрольные вопросы:

1. Как информационные технологии позволяют изменить базовые правила организации работы?
2. Какие виды влияния информационных технологий на перестройку деятельности компаний и соответственно группы технологий можно выделить?

Практическая работа 7
Разработка методов отражения и организация использования знаний

Цель работы: изучение порядка разработки методов отражения и организации использования знаний

Состав и организация знаний в экспертных системах

Стремление выделить организацию знаний в самостоятельную задачу вызвано, в частности, тем, что эта задача возникает для любого языка представления и способы решения этой задачи являются одинаковыми (либо сходными) вне зависимости от используемого формализма.

Итак, в круг вопросов, решаемых при представлении знаний, будем включать следующие:

- определение состава представляемых знаний;
- организацию знаний;
- представление знаний, т.е. определение "модели представления.

Состав знаний ЭС определяется следующими факторами:

- проблемной средой;
- архитектурой экспертной системы;
- потребностями и целями пользователей;
- языком общения.

В соответствии с общей схемой статической экспертной системы для ее функционирования требуются следующие знания:

• знания о процессе решения задачи (т.е. управляющие знания), используемые интерпретатором (решателем);

• знания о языке общения и способах организации диалога, используемые лингвистическим процессором (диалоговым компонентом);

• знания о способах представления и модификации знаний, используемые компонентом приобретения знаний;

• поддерживающие структурные и управляющие знания, используемые объяснительным компонентом.

Для динамической ЭС, кроме того, необходимы следующие знания:

1) знания о методах взаимодействия с внешним окружением;

2) знания о модели внешнего мира.

Зависимость состава знаний от требований пользователя проявляется в следующем:

• какие задачи (из общего набора задач) и с какими данными хочет решать пользователь;

• каковы предпочтительные способы и методы решения;

• при каких ограничениях на количество результатов и способы их получения должна быть решена задача;

• каковы требования к языку общения и организации диалога;

• какова степень общности (конкретности) знаний о проблемной области, доступная пользователю;

• каковы цели пользователей.

Состав знаний о языке общения зависит как от языка общения, так и от требуемого уровня понимания.

С учетом архитектуры экспертной системы знания целесообразно делить на *интерпретируемые* и *неинтерпретируемые*. К первому типу относятся те знания, которые способен интерпретировать решатель (интерпретатор). Все остальные знания относятся ко второму типу. Решатель не знает их структуры и содержания. Если эти знания используются каким-либо компонентом системы, то он не "осознает" этих знаний. Неинтерпретируемые знания подразделяются на *вспомогательные* знания, хранящие информацию о лексике и грамматике языка общения, информацию о структуре диалога, и *поддерживающие* знания. Вспомогательные знания обрабатываются естественно-языковой компонентой, но ход этой обработки решатель не осознает, так как этот этап обработки входных сообщений является вспомогательным для проведения экспертизы. Поддерживающие знания используются при создании системы и при выполнении объяснений. Поддерживающие знания выполняют роль описаний (обоснований) как интерпретируемых знаний, так и действий системы. Поддерживающие знания подразделяются на *технологические* и *семантические.* Технологические поддерживающие знания содержат сведения о времени создания описываемых ими знаний, об авторе знаний и т.п. Семантические поддерживающие знания содержат смысловое описание этих знаний. Они содержат информацию о причинах ввода знаний, о назначении знаний, описывают способ использования знаний и получаемый эффект. Поддерживающие знания имеют описательный характер.

Интерпретируемые знания можно разделить на *предметные знания, управляющие знания* и *знания о представлении.* Знания о представлении содержат информацию о том, каким образом (в каких структурах) в системе представлены интерпретируемые знания.

Предметные знания содержат данные о предметной области и способах преобразования этих данных при решении поставленных задач. Отметим, что по отношению к предметным знаниям знания о представлении и знания об управлении являются *метазнаниями.* В предметных знаниях можно выделить описатели и собственно предметные знания. Описатели содержат определенную информацию о предметных знаниях, такую, как коэффициент определенности правил и данных, меры важности и сложности. Собственно предметные знания разбиваются на *факты* и *исполняемые утверждения.* Факты определяют возможные значения сущностей и характеристик предметной области. Исполняемые утверждения содержат информацию о том, как можно изменять описание предметной области в ходе решения задач. Говоря другими словами, исполняемые *утверждения* - это знания, задающие процедуры обработки. Однако мы избегаем использовать термин "процедурные знания", так как хотим подчеркнуть, что эти знания могут быть заданы не только в процедурной, но и в декларативной форме.

Управляющие знания можно разделить на *фокусирующие* и *решающие.* Фокусирующие знания описывают, какие знания следует использовать в той или иной ситуации. Обычно фокусирующие знания содержат сведения о наиболее перспективных объектах или правилах, которые целесообразно использовать при проверке соответствующих гипотез. В первом случае внимание фокусируется на элементах рабочей памяти, во втором - на правилах базы знаний. Решающие знания содержат информацию, используемую для выбора способа интерпретации

знаний, подходящего к текущей ситуации. Эти знания применяются для выбора стратегий или эвристик, наиболее эффективных для решения данной задачи.

Качественные и количественные показатели экспертной системы могут быть значительно улучшены за счет использования *метазнаний,* т.е. знаний о знаниях. Метазнания не представляют некоторую единую сущность, они могут применяться для достижения различных целей. Перечислим возможные назначения метазнаний:

1) метазнания в виде стратегических метаправил используются для выбора релевантных правил;

2) метазнания используются для обоснования целесообразности применения правил из области экспертизы;

3) метаправила используются для обнаружения синтаксических и семантических ошибок в предметных правилах;

4) метаправила позволяют системе адаптироваться к окружению путем перестройки предметных правил и функций;

5) метаправила позволяют явно указать возможности и ограничения системы, т.е. определить, что система знает, а что не знает.

Вопросы организации знаний необходимо рассматривать в любом представлении, и их решение в значительной степени не зависит от выбранного способа (модели) представления. Выделим следующие аспекты проблемы организации знаний :

- организация знаний по уровням представления и по уровням детальности;
- организация знаний в рабочей памяти;
- организация знаний в базе знаний.

Организация знаний в рабочей памяти

Рабочая память (РП) экспертных систем предназначена для хранения данных. Данные в рабочей памяти могут быть однородны или разделяются на уровни по типам данных. В последнем случае на каждом уровне рабочей памяти хранятся данные соответствующего типа. Выделение уровней усложняет структуру экспертной системы, но делает систему более эффективной. Например, можно выделить уровень планов, уровень агенды (упорядоченного списка правил, готовых к выполнению) и уровень данных предметной области (уровень решений).

В современных экспертных системах данные в рабочей памяти рассматриваются как изолированные или как связанные. В первом случае рабочая память состоит из множества простых элементов , а во втором - из одного или нескольких (при нескольких уровнях в РП) сложных элементов (например, объектов). При этом сложный элемент соответствует множеству простых, объединенных в единую сущность. Теоретически оба подхода обеспечивают полноту, но использование изолированных элементов в сложных предметных областях приводит к потере эффективности.

Данные в РП в простейшем случае являются *константами* и (или) *переменными.* При этом переменные могут трактоваться как характеристики некоторого объекта, а константы - как значения соответствующих характеристик. Если в РП требуется анализировать одновременно несколько различных объектов, описывающих текущую проблемную ситуацию, то необходимо указывать, к каким объектам относятся рассматриваемые характеристики. Одним из способов решения

этой задачи является явное указание того, к какому объекту относится характеристика.

Если РП состоит из сложных элементов, то связь между отдельными объектами указывается явно, например заданием семантических отношений. При этом каждый объект может иметь свою внутреннюю структуру. Необходимо отметить, что для ускорения поиска и сопоставления данные в РП могут быть связаны не только логически, но и ассоциативно.

Организация знаний в базе знаний

Показателем интеллектуальности системы с точки зрения представления знаний считается способность системы использовать в нужный момент необходимые *(релевантные)* знания. Системы, не имеющие средств для определения релевантных знаний, неизбежно сталкиваются с проблемой "комбинаторного взрыва". Можно утверждать, что эта проблема является одной из основных причин, ограничивающих сферу применения экспертных систем. В проблеме доступа к знаниям можно выделить три аспекта: *связность знаний и данных, механизм доступа к знаниям* и *способ сопоставления.*

Связность (агрегация) знаний является основным способом, обеспечивающим ускорение поиска релевантных знаний. Большинство специалистов пришли к убеждению, что знания следует организовывать вокруг наиболее важных объектов (сущностей) предметной области. Все знания, характеризующие некоторую сущность, связываются и представляются в виде отдельного объекта. При подобной организации знаний, если системе потребовалась информация о некоторой сущности, то она ищет объект, описывающий эту сущность, а затем уже внутри объекта отыскивает информацию о данной сущности. В объектах целесообразно выделять два типа связок между элементами: *внешние* и *внутренние.* Внутренние связки объединяют элементы в единый объект и предназначены для выражения структуры объекта. Внешние связки отражают взаимозависимости, существующие между объектами в области экспертизы. Многие исследователи классифицируют внешние связки на *логические* и *ассоциативные.* Логические связки выражают семантические отношения между элементами знаний. Ассоциативные связки предназначены для обеспечения взаимосвязей, способствующих ускорению процесса поиска релевантных знаний.

Основной проблемой при работе с большой базой знаний является проблема поиска знаний, релевантных решаемой задаче. В связи с тем, что в обрабатываемых данных может не содержаться явных указаний на значения, требуемые для их обработки, необходим более общий механизм доступа, чем метод прямого доступа (метод явных ссылок). Задача этого механизма состоит в том, чтобы по некоторому описанию сущности , имеющемуся в рабочей памяти, найти в базе знаний объекты, удовлетворяющие этому описанию. Очевидно, что упорядочение и структурирование знаний могут значительно ускорить процесс поиска.

Нахождение желаемых объектов в общем случае уместно рассматривать как двухэтапный процесс. На первом этапе, соответствующем процессу выбора по ассоциативным связкам, совершается предварительный выбор в базе знаний потенциальных кандидатов на роль желаемых объектов. На втором этапе путем выполнения операции сопоставления потенциальных кандидатов с описаниями кандидатов осуществляется окончательный выбор искомых объектов. При организации

подобного механизма доступа возникают определенные трудности: Как выбрать критерий пригодности кандидата? Как организовать работу в конфликтных ситуациях? и т.п.

Операция сопоставления может использоваться не только как средство выбора нужного объекта из множества кандидатов; она может быть использована для классификации, подтверждения, декомпозиции и коррекции. Для идентификации неизвестного объекта он может быть сопоставлен с некоторыми известными образцами. Это позволит классифицировать неизвестный объект как такой известный образец, при сопоставлении с которым были получены лучшие результаты. При поиске сопоставление используется для подтверждения некоторых кандидатов из множества возможных. Если осуществлять сопоставление некоторого известного объекта с неизвестным описанием, то в случае успешного сопоставления будет осуществлена частичная декомпозиция описания.

Операции сопоставления весьма разнообразны. Обычно выделяют следующие их формы: *синтаксическое, параметрическое, семантическое* и *принуждаемое сопоставления*. В случае *синтаксического сопоставления* соотносят формы (образцы), а не содержание объектов. Успешным является сопоставление, в результате которого образцы оказываются идентичными. Обычно считается, что переменная одного образца может быть идентична любой константе (или выражению) другого образца. Иногда на переменные, входящие в образец, накладывают требования, определяющие тип констант, с которыми они могут сопоставляться.

Контрольные вопросы:

1. *Каков состав и организация знаний в экспертных системах?*
2. Как организованы знания в рабочей памяти?
3. Как осуществляется организация знаний в базе знаний?

Практическая работа 8
Использование «гибкой логики» и разработка программного обеспечения и алгоритмов за счет параллельного использования

Цель работы: изучение использования «гибкой логики» и разработки программного обеспечения и алгоритмов за счет параллельного использования

Интерес к объектно-ориентированным технологиям значительно возрос в последние годы прошлого десятилетия, когда в центре внимания разработчиков программного обеспечения оказались сложные системы, не поддающиеся программированию "в лоб". Создание подобных систем требует выполнения ряда этапов, предшествующих программированию.

Традиционно проектирование сложных систем основывалось на декомпозиции систем, т.е разбиении их на составные части, каждая из которых рассматривалась отдельно от других. Классический подход к разработке сложных систем представляет собой структурное проектирование, при котором осуществляется алгоритмическая декомпозиция системы по методу "сверху-вниз". Жизненный цикл разработки прикладной системы в этом случае складывается из этапов анализа, проектирования, программирования, тестирования и сопровождения, которые выполняются последовательно. Такой метод, называемый *каскадным,* имеет следующие отличительные особенности:

• линейность выполнения этапов жизненного цикла разработки;

• четкое разделение данных и процессов их обработки;

• использование процедурных языков программирования. Недостатки каскадного метода очевидны. Главный из них - последовательное выполнение этапов. Например, программирование можно начать только по завершении анализа и проектирования. Это приводит к большим потерям времени, не позволяет быстро разрабатывать прототипы программной системы. Каскадный принцип не согласуется с итеративным характером разработки программной системы, поскольку на последних этапах может выясниться необходимость внесения изменений в решения, принятые на предыдущих этапах.

Особенность процесса разработки современных сложных программных систем состоит в том, что центр тяжести смещается от программирования к более ранним этапам - анализу и проектированию, поэтому эффективность принятых методик анализа и проектирования имеет определяющее значение для судьбы проекта.

Достоинствами объектно-ориентированного подхода являются следующие.

Распараллеливание работ. Как отмечалось выше, программирование и тестирование отдельных компонентов системы возможно до завершения проектирования, что экономит время разработки. При программировании может возникнуть необходимость внесения изменений в существующие классы или потребоваться введение новых объектов или классов. В этом случае, вернувшись к этапу проектирования или даже к анализу, можно внести изменения и дополнений, не подвергая проект полной переработке.

Упрощение внесения изменений. В отличие от структурного подхода в объектно-ориентированном внесение изменений в проект имеет более локальный характер. В тех случаях, когда изменение носит характер уточнения, вводятся новые

классы, наследующие поведение ранее созданных. Наследование (одно из основных свойств классов) позволяет в этих случаях не только не пересматривать ранее созданные объекты и классы, но даже обойтись без их повторной трансляции. В более сложных случаях, когда меняются методы, определяющие интерфейс классов, изменения в проекте будут более значительными, но и тогда они будут локализованы, затрагивая лишь классы, использующие эти методы.

Переносимость и гибкость архитектуры. Объектно-ориентированная декомпозиция, в результате которой приложение представляется в виде совокупности классов и объектов, обеспечивает гибкость архитектуры системы. В клиент-серверной системе объекты могут размещаться как на клиентских местах, так и на серверах. В гетерогенных сетях возможна реализация классов на компьютерах разных типов, а фиксированный интерфейс каждого класса, определяемый набором его методов, обеспечит правильность функционирования системы. Изменения конфигурации оборудования не потребуют внесения изменений в проект.

Повторное использование программных компонентов. Разрабатываемые в рамках проекта классы обычно отражают типовые проектные решения, поэтому их использование возможно и в других проектах. Возможность повторного использования программных компонентов - одно из наиболее привлекательных свойств объектно-ориентированного подхода. Библиотеки классов, отражающие программистский опыт в определенной области, позволяют значительно снизить объем программирования при разработке новых проектов. При наличии развитых библиотек классов проектирование и программирование новых приложений будет в основном сводиться к сборке системы из готовых компонентов.

Иерархический характер сложных программных систем позволяет значительно повысить эффективность повторного использования компонентов. При этом чем более высокого уровня объекты можно повторно использовать, тем большего эффекта можно достичь. Для того чтобы повторное использование компонентов приносило свои плоды, разработчики программных систем должны:

• осознавать выгоды такого подхода;

• знать, какие части задачи могут быть решены с применением уже существующих программных средств;

• заниматься поиском подходящих для повторного использования программ;

• стремиться непременно найти такие программы;

• использовать их даже в том случае, если они лишь частично совпадают с тем, что программист написал бы сам.

Следует отметить, что основные свойства классов и объектов -инкапсуляция, наследование и полиморфизм - полностью отвечают задаче повторного использования.

Естественность описания. Объектно-ориентированный подход позволяет описывать как статические, так и динамические отношения между объектами модели. По описанию предметной области, выполненному на естественном языке, легко выделить объекты и статические связи между ними. Объекты соответствуют существительным, а связи - глаголам и отглагольным формам. Например, фраза "фирмы выполняют заказы" позволяет выделить классы объектов "фирма" и "заказ" и отношение "выполнять" между ними типа M:N (многие к многим), так как фирма может выполнять много заказов, а заказ может быть выполнен разными фирмами.

Кроме того, свойства наследования и инкапсуляции позволяют каждому участнику проекта рассматривать модель на удобном для него уровне детализации. Руководители проекта могут работать с верхним уровнем модели, где отражаются только основные классы, объекты и связи. Другие разработчики или эксперты имеют возможность опускаться до более мелких, терминальных объектов, их свойств, связей, методов.

Среда разработки

Развитая система встроенных текстовых и графических редакторов системы G2 и средств визуализации знаний приближает ее по возможностям к современным CASE-средствам. Упрощение взаимодействия разработчика с системой достигается за счет оригинального подхода, реализованного в текстовом редакторе. Процесс редактирования все время направляется процедурой грамматического разбора, что гарантирует введение только синтаксически правильных конструкций языка. В окне редактирования появляется динамически изменяемая подсказка, указывающая, какие языковые конструкции пользователь может вводить, начиная с текущей позиции курсора. Разработчик может набирать вводимый текст на клавиатуре или выбирать подходящие шаблоны из подсказки. При редактировании доступны клавиатурные команды и контекстно-зависимое меню операций редактирования.

Система RTworks не обладает встроенными средствами редактирования базы знаний. Приложение должно быть сначала записано в виде ASCII-файла и затем подвергнуто грамматическому разбору средствами RTworks. Фирма Talarian представляет такой подход как возможность пользоваться "Вашим любимым текстовым редактором". Очевидно, что отсутствие интерактивных средств разработки увеличивает стоимость и продолжительность этапа создания приложения.

Создание приложения в TDC Expert заключается в заполнении таблиц, представляющих перечень атрибутов объектов, описывающих анализируемые "ситуации" и выражения, которые должны вычисляться в процессе функционирования приложения.

Интерфейс с конечным пользователем

Система G2 предоставляет разработчику богатые возможности для формирования простого, ясного и выразительного графического интерфейса с пользователем с элементами мультипликации. Предлагаемый инструментарий позволяет наглядно отображать технологические процессы практически неограниченной сложности на разных уровнях абстракции и детализации. Кроме того, графическое отображение взаимосвязей между объектами приложения может напрямую использоваться в декларативных конструкциях языка описания знаний.

RTworks не обладает собственными средствами для отображения текущего состояния управляемого процесса. Разработчик приложения вынужден использовать систему Dataview фирмы VI Corporation, что в значительной степени ограничивает его возможности.

Интерфейс с пользователем TDC Expert ограничен возможностями системы TDC 3000, т.е. взаимодействие с конечным пользователем ограничивается текстовым режимом работы.

Архитектура приложения

Система RTworks базируется на возможностях операционной системы UNIX для организации распределенной обработки (рис. 1).

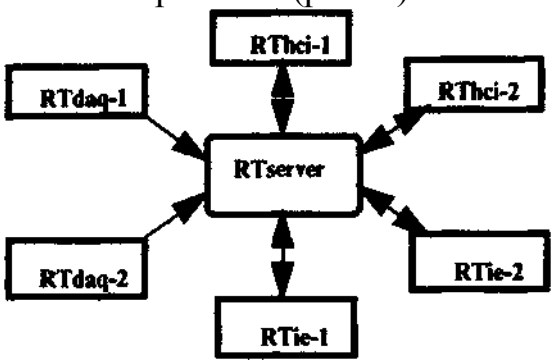


Рис. 1. Подсистемы типичного приложения на базе Rtworks

Приложения на базе RTworks имеют модульную структуру, которая включает следующие подсистемы:

- коммуникационный сервер (RTserver);
- подсистему получения данных (RTdaq);
- подсистему логического вывода (RTie);
- человеко-машинный интерфейс (RThci).

Наличие интерфейса с внешними процедурами, написанными на Си, и использование среды Unix для поддержки распределенной обработки обеспечивают открытость системы Rtworks.

G2 предоставляет разработчику гораздо более гибкие и мощные средства для формирования распределенных приложений на базе архитектуры клиент-сервер (рис.2). В зависимости от требований конкретной задачи можно построить систему как содружество автономных интеллектуальных агентов на базе интерфейса G2↔G2. При этом обмен данными осуществляется на уровне переменных через протокол ICP (Intelligent Communication Protocol). Для организации обмена необходимо в описании переменной, получающей значения от другого G2-процесса, просто указать номер сетевого порта источника. С другой стороны, можно разрабатывать приложение как иерархическую систему. Для этого фирмой Gensym разработана клиентная система Telewindows, обеспечивающая множественный доступ к централизованной базе знаний и групповую работу с приложением.

Связь с внешними источниками данных строится на основе библиотеки стандартных интерфейсов и сервера GSI (G2 Standard Interface). Подсистема GSI работает параллельно функционированию прикладной системы как независимый обработчик событий и обеспечивает ее двустороннее (в отличие от RTworks) взаимодействие с широким спектром программируемых контроллеров ведущих фирм (Alien Bradley, GE-Fanuc, AEG Modicon), систем сбора данных (ABB, Fisher, Siemens, Yokogawa, Foxboro, ORSI), концентраторов данных (DEC BASEstar, Alien Bradley Pyramid Integrator, SETPOINT SETCIM) и развитых СУБД (Oracle, Sybase, Informix, DEC Rdb). Библиотека GSI и так называемые G2 Bridge products позволяют легко интегрировать G2-приложения в существующие АСУТП.

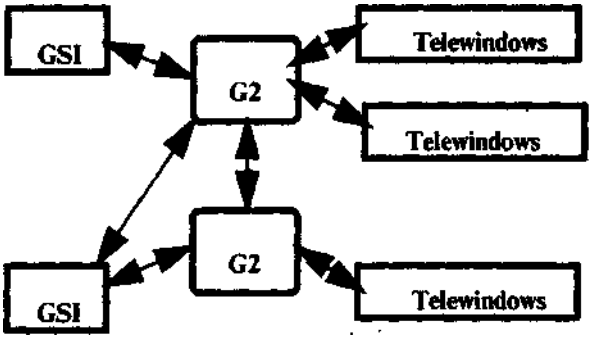


Рис. 2. Организация распределенной разработки средствами G2

В системе TDC Expert специальных средств для распределенной обработки не предусмотрено. Средства связи с управляемым процессом обеспечиваются комплексом TDC 3000. Структура типичного приложения на базе TDC Expert представлена на рис. 3.

Рис. 3. Структура системы на базе TDC Expert

Контрольные вопросы:

1. Что представляет с собой среда разработки?
2. Каков интерфейс с конечным пользователем?
3. Каковы особенности архитектуры приложения?

Практическая работа 9
Поиск средств расчета и параллельного алгоритма формирования динамических экспертных систем

Цель работы: изучение поиска средств расчета и параллельного алгоритма формирования динамических экспертных систем

Представление знаний *в системах ИИ*

Система ИИ – это система, оперирующая знаниями в проблемной области. Без Базы знаний систем ИИ не существует. Для формализации и представления знаний разрабатываются специальные модели представления знаний и языки для описания знаний, выделяются различные типы знаний Изучаются источники, из которых ИС может черпать знания, и создаются процедуры и приемы, с помощью которых возможно приобретение знаний для ИС

Данные и знания

Параллельно с развитием структуры ЭВМ происходило развитие информационных структур для представления данных. Появились способы описания данных в виде векторов и матриц, возникли списочные структуры, иерархические структуры. В настоящее время в языках программирования высокого уровня используются абстрактные типы данных, структура которых задается программистом. Появление баз данных (БД) знаменовало собой еще один шаг на пути организации работы с декларативной информацией. В пазах данных могут одновременно храниться большие объемы информации, а специальные средства, образующие систему управления базами данных (СУБД), позволяют эффективно манипулировать с данными, при необходимости извлекать их из базы данных и записывать их в нужном порядке в вазу

Особенности знаний

В общем случае данные в СИИ должны отвечать следующим 5 условиям.

1. Внутренняя интерпретируемость. Каждая информационная единица должна иметь уникальное имя, по которому СИИ находит ее, а также отвечает на запросы, в которых это имя упомянуто.

При переходе к знаниям в память ЭВМ вводится информация о некоторой протоструктуре информационных единиц.

2. Структурированность. Информационные единицы должны обладать гибкой структурой. Должна существовать возможность произвольного установления между отдельными информационными единицами отношений типа «часть – целое», «род – вид» или «элемент - класс».

3. Связность. В информационной базе между информационными единицами должна быть предусмотрена возможность установления связей различного типа. Прежде всего, эти связи могут характеризовать отношения между информационными единицами. Семантика отношений может носить декларативный или процедурный характер. Например, две или более информационные единицы могут быть связаны отношением 'одновременно*, две информационные единицы - отношением «причина – следствие» или от-

ношением «быть рядом». Приведенные отношения характеризуют декларативные знания. Если между двумя информационными единицами установлено отношение «аргумент - функция», то оно характеризует процедурное знание, связанное с вычислением определенных функций.

Между информационными единицами могут устанавливаться и иные связи, например, определяющие порядок выбора информационных единиц из памяти или указывающие на то, что две информационные единицы несовместимы друг с другом в одном описании.

Перечисленные три особенности знаний позволяют ввести общую модель представления знаний, которую можно назвать семантической сетью, представляющей собой иерархическую сеть, в вершинах которой находятся информационные единицы. Эти единицы снабжены индивидуальными именами. Дуги семантической сети соответствуют различным связям между информационными единицами.

Отношения могут быть самого разного типа, что позволяет в достаточной мере обеспечить в семантической сети такой признак знаний, как связность. В общем случае это означает, что в виде семантической сети можно отобразить знания, составленные на естественном языке

На примере семантической модели общего вида можно установить различие между базой данных и Базой знаний. Предметная область есть множество допустимых состояний своих компонентов. Это множество, представленное через общие понятия и отношения между ними, образует базу знаний - в виде интенсиональной семантической сети. С другой стороны, в зависимости от ситуации компоненты предметной области будут иметь конкретные значении, свойства, характеристики. Все эти конкретные данные о предметной области будут отображаться в так называемой экстенсиональной семантической сети или базе данных сетевой структуры.

4. Семантическая метрика. На множестве информационных единиц в некоторых случаях полезно задавать отношение, характеризующее ситуационную (контекстную) близость информационных единиц, т.е. силу ассоциативной связи между информационными единицами. Его можно было бы назвать отношением релевантности для информационных единиц. Такое отношение дает возможность выделять в информационной базе некоторые типовые ситуации или контексты интерпретации (например, «покупка», «регулирование движения на перекрестке»). Отношение релевантности при работе с информационными единицами позволяет находить знания, близкие к уже найденным.

5. Активность. С момента появления ЭВМ и разделения используемых в ней информационных единиц на данные и команды создалась ситуация, при которой данные пассивны, а команды активны. Все процессы, протекающие в ЭВМ. инициируются командами, а данные используются этими командами лишь в случае необходимости. Для СНН эта ситуации не приемлема. Как и у человека, в СИИ актуализации тех или иных действий способствуют знании, имеющиеся в системе. Таким образом, выполнение программ в СИИ должно инициироваться текущим состоянием информационной базы. Появление в базе фактов или описаний событий, установление связей может стать источником активности системы.

Перечисленные пять особенностей информационных единиц определяют ту грань, за которой данные превращаются в знания, а базы данных перерастают в базы знаний (БЗ). Совокупность средств, обеспечивающих работу с знаниями, образует систему управлении базой знаний (СУБЗ).

Логические модели представлении знаний

Постановка и решение любой всегда начинается с определения множества объектов (предметов и событий) предметной области.

Образы объектов предметной области (первичные и вторичные, а. следовательно, и абстрактные) принято называть сущностями.

Отношения между объектами и свойства объектов (одноместные отношения) можно рассматривать как сущности и включать в предметную область.

Вводя различные отношения подобия между сущностями, можно разбить предметную область на классы подобных сущностей, которые в свою очередь, являются сущностями данной предметной области.

Отношения между сущностями выражаются с помощью суждений, которым в языке соответствуют предложения.

Представление предметной области логическим языком называется логической моделью.

Этапы разработки экспертных систем

Разработка ЭС имеет существенные отличия от разработки обычного программного продукта. Опыт создания ЭС показал, что использование при их разработке методологии, принятой в традиционном программировании, либо чрезмерно затягивает процесс создания ЭС, либо вообще приводит к отрицательному результату. Перед тем как приступить к разработке ЭС, инженер по знаниям должен рассмотреть вопрос, следует ли разрабатывать ЭС для данного приложения. В обобщенном виде ответ может быть таким; использовать ЭС следует только тогда, когда разработка ЭС *возможна, оправдана* и методы инженерии знаний *соответствуют* решаемой задаче. Ниже будут уточнены использованные понятия "возможно", "оправдано", "соответствует". Чтобы разработка ЭС была *возможной* для данного приложения, необходимо одновременное выполнение по крайней мере следующих требований:

1) существуют эксперты в данной области, которые решают задачу значительно лучше, чем начинающие специалисты;

2) эксперты сходятся в оценке предлагаемого решения, иначе нельзя будет оценить качество разработанной ЭС;

3) эксперты способны вербализовать и объяснить используемые ими методы, в противном случае трудно рассчитывать на то, что знания экспертов будут "извлечены" и вложены в ЭС;

4) решение задачи требует только рассуждений, а не действий;

5) задача не должна быть слишком трудной;

6) задача хотя и не должна быть выражена в формальном виде, но все же должна относиться к достаточно "понятной" и структурированной области, т.е. должны быть выделены основные понятия, отношения и известные способы получения решения задачи;

7) решение задачи не должно в значительной степени использовать "здравый смысл", так как подобные знания пока не удается (в достаточном количестве) вложить в системы искусственного интеллекта.

Использование ЭС в данном приложении может быть возможно, но не оправдано. Применение ЭС может быть *оправдано* одним из следующих факторов:

• решение задачи принесет значительный эффект, например экономический;

• использование человека-эксперта невозможно либо из-за недостаточного количества экспертов, либо из-за необходимости выполнять экспертизу одновременно в различных местах;

• использование ЭС целесообразно в тех случаях, когда при передаче информации эксперту происходит недопустимая потеря времени или информации;

• использование ЭС целесообразно при необходимости решать задачу в окружении, враждебном для человека.

Приложение *соответствует* методам ЭС, если решаемая задача обладает совокупностью следующих характеристик:

1) задача может быть естественным образом решена посредством манипуляции с символами, а не манипуляций с числами, как принято в математических методах и в традиционном программировании;

2) задача должна иметь эвристическую, а не алгоритмическую природу, т.е. ее решение должно требовать применения эвристических правил. Задачи, которые могут быть гарантированно решены с помощью некоторых формальных процедур, не подходят для применения ЭС;

3) задача должна быть достаточно сложна, чтобы оправдать затраты на разработку ЭС. Однако она не должна быть чрезмерно сложной, чтобы ЭС могла ее решать;

4) задача должна быть достаточно узкой, чтобы решаться методами ЭС, и практически значимой.

Процесс создания ЭС не сводится к строгой последовательности перечисленных выше этапов. В ходе разработки приходится неоднократно возвращаться на более ранние этапы и пересматривать принятые там решения.

Контрольные вопросы:

1. Представление знаний *в системах ИИ?*

2. Данные и знания, их особенности?

3. Логические модели представлении знаний?

4. Перечислите и охарактеризуйте основные этапы разработки ЭС

Практическая работа 10
Интеллектуальные системы управления и возможности и перспективы прикладного применения

Цель работы: изучение интеллектуальных систем управления и возможностей и перспектив прикладного применения

Среди всех видов ИС, наиболее динамично развиваются ЭС реального времени. В 1995 г. объем продаж ЭС реального времени составил примерно 70% рынка проблемно/предметно-ориентированных СОЗ и был равен 38 млн дол. (в 1988 г. - 3 млн. дол.). Значимость ИС и ЭС реального времени (РВ) определяется не столько их бурным коммерческим успехом (хотя и это достойно тщательного анализа), но в первую очередь тем, что только с помощью подобных средств создаются стратегически значимые приложения в таких областях, как: управление непрерывными производственными процессами в химии, фармакологии, производстве цемента, питания и т.п.; аэрокосмические исследования, транспортировка и переработка нефти(газа), управление атомными и тепловыми электростанциями, финансовые операции, связь и многие другие.

В последнее время на основе динамических ИС начинают создаваться ИС для интеллектуального имитационного моделирования, используемые в реинжиниринге (реорганизации) бизнес-процессов (БПР) (см. Приложение 2). Интерес к ИС этого типа инициируется тем, что в отличие от статических ИС и ЭС, используемых, как указано ранее, для БПА, т.е. для автоматизации текущего состояния бизнеса, ИС для БПР используются для решения существенно более значимых и сложных задач, т.е. для "фундаментального переосмысления и радикального перепроектирования деловых процессов для достижения существенных улучшений в главных показателях деятельности компаний, таких, как стоимость, качество, услуги и темпы"[2].

Ниже перечислены некоторые области применения ЭС РВ, разработанных на базе ИС G2 (см. гл. 9). Всего на базе G2 разработано более 700 ЭС РВ, работающих более чем в 30 областях.

Области применения ЭСРВ (перечень фирм и характеристик приложений)

3М (США) - G2 используется на ряде заводов 3М в Миннесоте для управления технологическими процессами и поддержки принятия решений.

Caterpillar (США) - интегрированная система мониторинга и планирования для прокатного стана на базе распределенной системы, включающей G2 и Telewindows.

Camunsa (Испания) - автоматизированный, интеллектуальный гараж в Барселоне, разработанный к летним Олимпийским играм 1992 г. Гараж не требует присутствия людей и размещает 800 машин на том же пространстве, где при обычном подходе размещаются только 300.

Carpenter Technology Corp. (США) - CarTech использует DSP (ИС на базе G2) для моделирования операций горячего прокатного стана и связанных с ним печей. DSP разрабатывает расписание печи и потока материалов, поступающих от печи на дальнейшую обработку.

Forsmark Nuclear Plant (Швеция) - система обеспечения безопасности и моделирования событий для ядерной электростанции. Содержит более 200 правил. Использует более 130 диаграмм различной формы для отображения процесса.

General Electric (США) - GE разработала ряд систем на базе G2: систему для наземных станций слежения за спутниками в GE Aerospace в Филадельфии; систему для производства и тестирования самолетных двигателей в Линнеи; предсказывающую систему для GE Nuclear в Сан Хозе, CA.

IBM (США) - MOM (Measurement of On-line Manufacturing) - система управления, разработанная для улучшения производства блоков памяти и питания на заводе IBM в Торонто и интегрированная в производственный процесс. MOM объединяет системы G2, Serveio's Gemstone OODBMS и последовательную SPS в единую систему управления и контроля за производством печатных плат, повышающую качество, окупаемость и производительность завода.

Intelsat (США) - система диагностики, мониторинга и контроля сети, разработанная за 4 месяца на базе G2. Обеспечивает помощь при восстановлении спутников путем мониторинга критических состояний и диагностики сбоев коммуникационных каналов до и во время их появления.

Lafarge Coppee (США) - 25 установок G2 на цементных заводах, расположенных по всему миру. Lafarge использует возможности нечеткой логики G2 для обеспечения замкнутого цикла управления мельничными установками.

Mrs.Baird's Bakery (США) - самая большая частная пекарня в США использует G2 для планирования и управления всем производственным процессом.

NASA/ Space Shuttle (США) - NASA использует G2 с октября 1988 г. в ряде систем для космических аппаратов, включая управление 38 реактивными двигателями, обеспечивающими маневрирование челнока. G2 обрабатывает данные от 16 000 датчиков в секунду, осуществляя проверку всех параметров от температуры до курса.

ЭС РВ решают следующие классы задач: мониторинг в реальном масштабе времени; системы управления верхнего уровня; системы обнаружения неисправностей; диагностика; составление расписаний; планирование; оптимизация; системы - советчики оператора; системы проектирования и т. п.

Традиционные статические ИС и ЭС не способны решать подобных задач, так как они не выполняют требования, предъявляемые к системам, работающим в реальном времени:

• представлять изменяющиеся во времени данные, поступающие от внешних источников, обеспечивать хранение и анализ изменяющихся данных;

• выполнять одновременно временные рассуждения о нескольких различных асинхронных процессах (задачах), т. е. планировать в соответствии с приоритетами обработку процессов, поступивших в систему;

• обеспечивать механизм рассуждения при ограниченных ресурсах (время, память). Реализация этого механизма предъявляет требования к высокой скорости работы системы, способности одновременно решать несколько задач (т. е. необходимо использовать операционные системы UNIX, VMS, Windows NT, но не MS DOS);

• обеспечивать предсказуемость поведения системы, т.е. гарантию того, что каждая задача будет запущена и завершена в строгом соответствии с временными

ограничениями. Например, требование предсказуемости не допускает использования в ЭС РВ механизма сборки мусора, свойственного языку Lisp;

• моделировать "окружающий мир", рассматриваемый в данном приложении, обеспечивать создание различных его состояний;

• протоколировать свои действия и действия персонала, обеспечивать восстановление после сбоя;

• обеспечивать наполнение базы знаний (БЗ) для приложений реальной степени сложности с минимальными затратами времени и труда (необходимо использование объектно-ориентированной технологии, общих правил, модульности и т. п.);

• обеспечивать настройку системы на решаемые задачи (проблемно-предметная ориентация);

• обеспечивать создание и поддержку пользовательских интерфейсов для различных категорий пользователей;

• обеспечивать уровень защиты информации (по категориям пользователей) и предотвращать несанкционированный доступ.

Области использования динамических ЭС

Опыт применения динамических ЭС опишем на примере программных продуктов фирмы Gensym, занимающей первое место в мире среди фирм, производящих интеллектуальные продукты. Интерес к этой фирме вызван и тем, что с 1994 г. ее продукты доступны на рынке СНГ (и в частности, России). Интересы Gensym в СНГ представляет АО Аргуссофт Компани. К настоящему моменту, несмотря на высокую стоимость этих продуктов, во всем мире их продано более 5000 копий; все 25 самых крупных компаний мира используют систему G2.

Широкое распространение G2 предопределило создание международного общества пользователей фирмы Gensym, а опыт ее использования освещается в многочисленных докладах и публикациях.

Перечислим наиболее характерные прикладные системы, разработанные на базе продуктов фирмы Gensym.

Нефть и газ: системы управления газопроводами, системы переработки нефти.

Финансы и бизнес: реинжиниринг банков и компаний (см. Приложение 2).

Космос: управление двигателями космических аппаратов (в частности, NASA широко использует G2 с октября 1988 г.); мониторинг критических состояний и диагностика сбоев коммуникационных каналов, обеспечивающих связь со спутниками.

Металлургия и машиностроение: контроль за состоянием доменной печи; управление оборудованием по производству сплавов; моделирование операций прокатного стана и связанных с ним печей, составление расписания обработки для печей и потока обрабатываемых материалов в целях наиболее полного использования оборудования.

Пищевая промышленность: планирование всего производственного процесса крупной пекарни, слежение за ним и управление; динамическое планирование загрузки линий по упаковке напитков.

Электроника: диагностика и выявление неисправностей в линиях по производству печатных плат в целях сокращения брака.

Транспорт: планирование загрузки контейнеров при авиационных, железнодорожных и автомобильных перевозках.

Разработчикам моделей система предоставляет следующие возможности:

• формировать и корректировать карты-схемы, содержащие информацию о географическом расположении водохранилищ и ГЭС;

• просматривать, вводить и корректировать справочную информацию о характеристиках водохранилищ и ГЭС;

• использовать в картах-схемах географические карты, подготовленные в виде графических файлов;

• формировать и корректировать данные о фрагментах русла -первичных участках - непосредственно на схематическом описании русла;

• формировать и корректировать данные о створах выборочной выдачи, а также подключать архивную информацию (в виде таблиц и графиков) по расходам и уровням.

Конечные пользователи имеют следующие возможности:

• получать автоматически сформированные модели каскадов по картам-схемам (в каскад включаются гидроузлы и водохранилища, отмеченные пользователем на карте-схеме);

• вести архивы фактических данных по расходам и уровням на периоды половодья;

• задавать и использовать в расчетах как фактические, так и экспериментальные параметры русла;

• задавать и корректировать граничные и начальные условия расчетов;

• проводить оперативный расчет неустановившегося движения воды с определением уровней во всех первичных участках и автоматическим выявлением нарушений допустимых пределов по уровням;

• получать фактическую информацию по уровням с автоматическим выявлением существенных отклонений от расчетных данных;

• проводить перерасчет с учетом полученных фактических данных и (или) откорректированных граничных условий.

Основным режимом работы системы является режим имитационного моделирования. В рабочее окно системы выводится автоматически сформированная схема русла участка реки с указанием створов и ГЭС. Выбрав любой из объектов модели, пользователь может просмотреть, ввести и (или) откорректировать архивную информацию по фактическим режимам, подготовить исходные данные для расчета, а также ознакомиться с текущими результатами имитационного моделирования.

В ходе расчета контролируется соблюдение ограничений на уровни в створах выборочной выдачи. В случае нарушения ограничений цветом выделяется изображение створа, в котором выявлено нарушение, и выдается предупреждение. Можно остановить расчет, откорректировать исходные данные и повторить процедуру моделирования.

Получив расчетные результаты, можно перейти в режим, в ходе которого имитируется ежесуточное поступление фактических данных и сравнивается поступающая информация с результатами расчета. В случае отклонения расчетных данных от фактических более чем на 15 % система выдает соответствующее предупреждение.

Получение экспертных оценок параметров, используемых в расчетной модели, может дать хорошие результаты только в тех случаях, когда принятые в модели допущения соответствуют реальным условиям задачи. Чтобы расширить область эффективного использования системы, в настоящее время исследуется возможность применения методов нейронных сетей для получения значений уровней и расходов по начальным и граничным условиям.

Система поддержки оперативного управления гидравлическим режимом каскадов водохранилищ функционирует на рабочих станциях Sun и RS 6000. Расчетный модуль реализован на языке Си. Система включает около 80 классов и более 1700 объектов языка представления знаний комплекса G2, а также более 200 процедур и более 30 общих правил.

Контрольные вопросы:

1. Интеллектуальные системы управления и возможности и перспективы прикладного применения?
2. Каковы области применения ЭСРВ (перечень фирм и характеристик приложений)?
3. В каких областях используются динамические ЭС?

ЛИТЕРАТУРА

1. Афонин В. Л., Макушкин В. А. Предмет исследования искусственного интеллекта. Сайт "Интернет университет"
2. Афонин В. Л., Макушкин В. А. Интеллектуальные робототехнические системы. - М. : Интернет-университет информационных технологий - ИНТУИТ.ру, 2005. - 235 с.
3. Гаврилова Т. А., Хорошевский В. Ф. Базы знаний интеллектуальных систем: Учебник для вузов. Сайт "Введение в интеллектуальные системы".

4.Минаев В.А. Прогноз финансовых рисков. Сайт "Интеллектуальные системы обработки данных и проблемы информационой безопасности в интернет"

4. Кудрявцев Б.А. COD:NET все для программиста. Сайт "Системы распознавания образов (идентификации)"
5. Сотник С. Л., Искусственный интеллект. Сайт "Системы распознавания образов (идентификации)"
6. Терехов А.С, Лекции по теории и приложениям искусственных нейронных сетей. Сайт "Биологический нейрон и его кибернетическая модель."
7. Цыганков В. Д., Конспект лекций. Сайт "Тема 13. Интеллектуальные системы управления"
8. Эндрю А.Л. Искусственный интеллект. - М.: Мир, 1985. - 267с.
9. Винер Н.Н. Кибернетика.- М.: Наука, электронная версия, 1998.- 189с.
10. Соколов Е. Н., Вайткявичус Г.Г. Нейроинтеллект: от нейрона к нейрокомпьютеру. - М.: Наука, 1989. - 83с.
11. Люгер Д. В. Искусственный интеллект. - М. : Мир, 2003. - 195 с.

Перечень тем самостоятельных работ

1. Нечеткие множества
2. Форма представления нечетких множеств и их компьютерная реализация
3. Свойства нечетких множеств
4. Лингвистические модификации нечетких множеств
5. Нечеткая логика в задаче фильтрации случайных возмущений
6. Многослойные нейронные сети и их аппроксимирующие свойства
7. Основные этапы процедуры идентификации
8. Выбор модальной структуры
9. Планирование и проведение эксперимента
10. Оптимизация параметров нейросетевой модели

ВОПРОСНИК ДЛЯ ПК, ТК И ИК

1. **Оценка информации полученной от систем датчиков**
2. Интерес к интеллектуальным системам управления (ИСУ)
3. Базовые интеллектуальные технологии
4. Машинная форма представления информации
5. Практическая реализация концепции ситуационного управления на основе современных интеллектуальных технологий
6. **Синтез цели и принятие решения для начала движение**
7. Фундаментальные и прикладные работы по созданию интеллектуальных систем управления
8. Модели УО
9. **Активные оценки информации**
10. Основные блоки концептуальной архитектуры интеллектуальной системы
11. Аппаратно-реализованные блоки ИС
12. **Предсказание результатов движение и разработка управление**
13. Принципы оценки поведения интеллектуальных систем управления
14. Степени интеллектуальности
15. Модели и методы вывода в условиях неопределенности используемой информации
16. Уровни иерархии интеллектуальной системы управления и степень интеллектуальности
17. **Преобразование управление в физические сигналы**
18. Физические и математические основы оптических систем контроля
19. Контроль качества поверхности (ее шероховатости)
20. Контроль изделий сложной формы в составе технологических систем.
21. **Цепь обратной связи**
22. Прямая и обратная цепочки рассуждений
23. Основные понятия теории вероятностей
24. Нечеткая логика в экспертных системах
25. **Эмоциональные оценки полученных результатов**
26. Основные направления интеллектуализации прикладных систем и систем принятия решений
27. Методы искусственного интеллекта в прикладных системах и системах принятии решений
28. Интеллектуальные информационные технологии в прикладных системах и системах принятия решений
29. **Коррекция управления**
30. Процесс создания прототипа
31. Система управления
32. Система оптического контроля
33. **Синтез новых целей и их организация**
34. Неформализованные задачи
35. Сущность механизма афферентного синтеза
36. Структура процесса получения решения
37. **Динамические экспертные системы**
38. Назначение экспертных систем

39. Формальные основы экспертных систем
40. Архитектура статических и динамических экспертных систем
41. Этапы разработки экспертных систем
42. Способы характеристики динамических систем
43. Анализ состояния статических экспертных систем
44. Анализ состояния динамических экспертных систем
45. Основные производители ИС для ЭС РВ
46. Интеллектуальные операторы, осуществляющие формирование, отображение возможностей и умственных выводов, понятий в процессах понятия
47. Механизмы вывода экспертных систем
48. Стратегии как механизмы управления
49. Способы решения задач ДЭС
50. Методы поиска решений в экспертных системах
51. Поиск решений в одном пространстве
52. Поиск в иерархии пространств
53. Поиск в альтернативных пространствах
54. Поиск с использованием нескольких моделей
55. Выбор метода решения задач
56. Знания в экспертных системах: концептуалые; фактические, предметные; алгоритмические, процедурные
57. Формирование эмпирических знаний, стратегий и эвристик
58. Формирование знаний о динамике нелинейной системы автоматического управления.
59. Знания, передаваемые экспертной системе
60. Создания и применение базы знаний в интеллектуальных системах
61. Сущности и иерархия классов
62. Иерархия модулей и рабочих пространств
63. Структуры данных БЗ
64. Прикладные программы и алгоритмы для решения уравнений
65. Центральная парадигма интеллектуальных технологий - обработка знаний
66. Современное состояние разработок в области ЭС
67. Применение ЭС
68. База данны
69. Направление в изучении искусственного интеллекта
70. Особенности перехода от Базы Данных к Базе Знаний
71. Методы представления знаний
72. Этапы решения задач: построения абстрактных программ для решения
73. Принятие решения
74. Структура и состав интеллектуальной робототехнической системы.
75. Пример робота-станка
76. Перевод задачи на машинный язык; трансляция и выполнения программ.
77. Распознавание изображений
78. Преобразование изображений в цифровой код

79. Экспертные системы
80. Машинный перевод и понимание текстов на естественном языке
81. **Создание единой программной среды и синтез алгоритмов для непосредственной задачи**
82. Области использования динамических ЭС
83. Моделирование гидравлических режимов каскадов водохранилищ на базе G2
84. Имитационная модель автоматизированного грузового комплекса на базе Rethink

ГЛОССАРИЙ

Интеллектуальная информационная система (ИИС) - это ИС, которая основана на концепции использования базы знаний для генерации алгоритмов решения экономических задач различных классов в зависимости от конкретных информационных потребностей пользователей.

Важнейшие признаки классификации ИИС: развитые коммуникативные способности, сложность (плохая формализуемость алгоритма), способность к самообучению, адаптивность.

Основные подклассы ИИС: интеллектуальные базы данных, в т.ч. с интерфейсами, использующими естественный язык, гипертекст и мультимедиа, когнитивную графику; статические и динамические экспертные системы; самообучающиеся системы на принципах индуктивного вывода, нейронных систем, поиска прецедентов, организации информационных хранилищ; адаптивные информационные системы на основе использования CASE-технологий и/или компонентных технологий.

Система с интеллектуальным интерфейсом - это ИИС, предназначенная для поиска неявной информации в базе данных или тексте для произвольных запросов, составляемых, как правило, на ограниченном естественном языке.

Экспертная система (ЭС) - это ИИС, предназначенная для решения слабоформализуемых задач на основе накапливаемого в базе знаний опыта работы экспертов в проблемной области.

Участники процесса разработки и эксплуатации ЭС: эксперты, инженеры по знаниям, пользователи.

Эксперт - специалист, знания которого помещаются в базу знаний.

Инженер по знаниям - специалист, который занимается извлечением знаний и их формализацией в базе знаний.

Пользователь - специалист, интеллектуальные способности которого расширяются благодаря использованию в практической деятельности ЭС.

Основные составные части архитектуры ЭС: база знаний, механизмы вывода, объяснения, приобретения знаний, интеллектуальный интерфейс.

База знаний - это центральный компонент ЭС, который определяет ценность ЭС и с которым связаны основные затраты на разработку.

База знаний - это хранилище единиц знаний, описывающих атрибуты и действия, связанные с объектами проблемной области, а также возможные при этом неопределенности.

Единица знаний - это элементарная структурная единица, (описание одного объекта, одного действия), которая имеет законченный смысл. В качестве единиц знаний обычно используются **правила** и/или **объекты.**

Неопределенность знаний - это или неполнота, или недостоверность, или многозначность, или качественная (вместо количественной) оценка единицы знаний.

Механизм вывода - это обобщенная процедура поиска решения задачи, которая на основе базы знаний и в соответствии с информационной потребностью пользователя строит цепочку рассуждений (логически связанных единиц знаний), приводящую к конкретному результату.

Дедуктивный вывод (от общего к частному)- вывод частных утверждений путем подстановки в общие утверждения других известных частных утверждений. Различают **прямую** (от данных к цели) и **обратную** (от цели к данным) **цепочки рассуждений (аргументации).**

Индуктивный вывод (от частного к общему) - вывод (обобщение) на основе множества частных утверждений общих утверждений (из примеров реальной практики правил).

Абдуктивный вывод (от частного к частному) - вывод частных утверждений на основе поиска других аналогичных утверждений (прецедентов).

Механизм приобретения знаний - это процедура накопления знаний в базе знаний, включающая ввод, контроль полноты и непротиворечивости единиц знаний и, возможно, автоматический вывод новых единиц знаний из вводимой информации.

Механизм объяснения - это процедура, выполняющая обоснование полученного механизмом вывода результата.

Интеллектуальный интерфейс - это процедура, выполняющая интерпретацию запроса пользователя к базе знаний и формирующая ответ в удобной для него форме.

Назначение экспертной системы: консультирование и обучение неопытных пользователей, ассистирование экспертам в решении задач, советы экспертам по вопросам из смежных областей знаний (интеграция источников знаний).

Статическая экспертная система - это ЭС, решающая задачи в условиях не изменяющихся во времени исходных данных и знаний.

Динамическая экспертная система - это ЭС, решающая задачи в условиях изменяющихся во времени исходных данных и знаний.

Аналитическая экспертная система - это ЭС, осуществляющая оценку вариантов решений (проверку гипотез).

Синтетическая экспертная система - это ЭС, осуществляющая генерацию вариантов решений (формирование гипотез).

Классы решаемых задач в экспертной системе: интерпретация, диагностика, прогнозирование, проектирование, планирование, мониторинг, коррекция, управление.

Самообучающаяся система - это ИИС, которая на основе примеров реальной практики автоматически формирует единицы знаний.

Система с индуктивным выводом - это самообучающаяся ИИС, которая на основе обучения по примерам реальной практики строит деревья решений.

Нейронная сеть - это самообучающаяся ИИС, которая на основе обучения по примерам реальной практики строит ассоциативную сеть понятий (нейронов) для параллельного поиска на ней решений.

Система, основанная на прецедентах, - это самообучающаяся ИИС, которая в качестве единиц знаний хранит собственно прецеденты решений (примеры) и позволяет по запросу подбирать и адаптировать наиболее похожие прецеденты.

Информационное хранилище (Data Warehouse) - это самообучающаяся ИИС, которая позволяет извлекать знания из баз данных и создавать специально-организованные базы знаний.

Этапы создания экспертных систем: идентификация, концептуализация, формализация, реализация, тестирование, внедрение.

Прототип экспертной системы - это расширяемая (изменяемая) на каждом последующем этапе версия базы знаний с возможной модификацией программных механизмов. Различают прототипы: демонстрационный, исследовательский, действующий, промышленный, коммерческий.

Этап идентификации проблемной области - определение требований к разрабатываемой ЭС, контуров рассматриваемой проблемной области (объектов, целей, подцелей, факторов), выделение ресурсов на разработку ЭС.

Этап концептуализации проблемной области - построение концептуальной модели, отражающей в целостном виде сущность функционирования проблемной области на объектном (структурном), функциональном (операционном), поведенческом (динамическом) уровнях.

Объектная модель - отражение на семантическом уровне фактуального знания о классах объектов, их свойств и отношений. Элементарная единица объектного знания - это **триплет:** "объект - свойство (отношение) - значение" или двухместный предикат.

Функциональная модель - отражение зависимостей фактов, определяющих условия образования одних фактов из других. Элементарная единица функционального знания - **импликация** фактов.

Дерево целей (граф "И"-"ИЛИ") фиксирует зависимость целевого предиката (переменной) от множества факторов - определяющих предикатов (переменных).

Дерево решений фиксирует зависимость значений целевого предиката от комбинации значений факторов.

Поведенческая модель - отражение выполняемых действий над объектами (фактами) в зависимости от происходящих во времени событий.

Этап формализации базы знаний - выбор метода представления знаний, в рамках которого проектируется логическая структура базы знаний. **Методы представления знаний** различаются характером представления объектного, функционального, поведенческого видов знаний и реализацией неопределенностей, т.е. ориентацией на определение структуры объектов или действий над ними, детерминированность или неопределенность, статику или динамику проблемной области.

Метод представления (модель) знаний - это совокупность средств структурирования и обработки единиц знаний. Методы представления знаний различаются характером представления объектного, функционального, поведенческого видов знаний и реализацией неопределенностей, т.е. ориентацией на определение структуры объектов или действий над ними, детерминированность или неопределенность, статику или динамику проблемной области.

Логическая модель - это модель, в которой область определения предиката задается либо перечислением фактов, либо в виде импликаций (правил).

Продукционная модель - факты - значения переменных, операции над фактами - правила. Правила выбираются из конфликтных наборов с помощью задаваемых эвристических критериев: приоритетов, достоверности, стоимости и т.д.

Простые правила - обрабатывают отдельные значения переменных.

Обобщенные правила - обрабатывают классы объектов.

Правила, управляемые данными:

Если <условие> То <заключение>

Правила, управляемые событиями:

Всякий раз, как <событие> То <действие>

Обработка неопределенностей знаний основана на использовании условных вероятностей или нечеткой логики.

Семантическая сеть отражает как объектное, так и операционное знание в виде двухместных предикатов (бинарных отношений). Различают типизированные отношения “род” -”вид”, “целое” - “часть”, “причина” - “следствие” и др.

Фреймовая модель - это семантическая сеть с N-арными отношениями и присоединенными процедурами. Используются **механизмы наследования свойств** по иерархии классов объектов и вызова процедур в зависимости от происходящих событий.

Объектно-ориентированная модель предусматривает **инкапсуляцию** процедур (методов) в структуры данных классов объектов, к которым разрешен доступ только через эти методы. Механизм наследования свойств распространяется и на методы, обеспечивая свойство **полиморфизма** процедур.

Этап реализации ЭС представляет отображение структуры базы знаний в среде выбранного инструментального средства, а также настройка и/или доработка программных механизмов. Различают программные оболочки, инструментальные среды и языки представления знаний; универсальные инструментальные, проблемно-ориентированные и предметно-ориентированные инструментальные программные средства.

Алгоритм выбора инструментального средства. Требования класса решаемых задач в части реализации объектов, операций и неопределенностей налагаются на возможности инструментальных средств по представлению выявленных особенностей знаний, в результате чего формируется ранжированный список инструментальных средств.

Этап тестирования оценивает экспертную систему с позиции двух основных групп критериев: точности и полезности. **Точность** работы: правильность заключений, адекватность базы знаний проблемной области, соответствие методов решения проблемы экспертным. **Полезность:** ответы на запросы пользователя; удобство интерфейса; объяснение получаемых результатов; надежность, адаптирумость, производительность и стоимость эксплуатации.

Этап внедрения и опытной эксплуатации - это переход от тестовых примеров к решению реальных задач.

ПРИЛОЖЕНИЯ

Новые информационные технологии, интегрируемые с технологией экспертных систем

ПРИЛОЖЕНИЕ 1

Объектно-ориентированная технология

Особенности сложных программных систем

Интерес к объектно-ориентированным технологиям значительно возрос в последние годы прошлого десятилетия, когда в центре внимания разработчиков программного обеспечения оказались сложные системы, не поддающиеся программированию "в лоб". Создание подобных систем требует выполнения ряда этапов, предшествующих программированию.

Традиционно проектирование сложных систем основывалось на декомпозиции систем, т.е разбиении их на составные части, каждая из которых рассматривалась отдельно от других. Классический подход к разработке сложных систем представляет собой структурное проектирование, при котором осуществляется алгоритмическая декомпозиция системы по методу "сверху-вниз". Жизненный цикл разработки прикладной системы в этом случае складывается из этапов анализа, проектирования, программирования, тестирования и сопровождения, которые выполняются последовательно. Такой метод, называемый *каскадным,* имеет следующие отличительные особенности:

• линейность выполнения этапов жизненного цикла разработки;

• четкое разделение данных и процессов их обработки;

• использование процедурных языков программирования. Недостатки каскадного метода очевидны. Главный из них - последовательное выполнение этапов. Например, программирование можно начать только по завершении анализа и проектирования. Это приводит к большим потерям времени, не позволяет быстро разрабатывать прототипы программной системы. Каскадный принцип не согласуется с итеративным характером разработки программной системы, поскольку на последних этапах может выясниться необходимость внесения изменений в решения, принятые на предыдущих этапах [1,11].

Для устранения этого недостатка Б.Боэм [10] предложил *спиральный* подход. Он заключается в том, что разработка проекта ведется как бы по спирали, причем на каждом ее витке выполняются последовательно перечисленные выше этапы, на которых уточняется проект [7]. Этот подход дополняет каскадный метод элементами итеративности. Но и для него характерен ряд существенных недостатков, к числу которых можно отнести [7]:

• трудоемкость внесения изменений;

• большой объем документации по проекту, затрудняющий программирование;

• серьезные ограничения возможностей сборки системы из готовых компонентов;

• сложность переноса на другие платформы.

Для того чтобы раскрыть сущность *объектно-ориентированного* подхода к разработке приложений [9], рассмотрим главные особенности сложных систем, определяющие требования к методикам и инструментальным средствам, поддерживающим жизненный цикл их разработки.

Иерархичность. Описывая характерные черты сложных систем, Г. Буч [1,11] особое внимание уделяет их иерархическому характеру. Иерархическое построение таких систем облегчает понимание их человеком, возможности которого по восприятию информации весьма ограничены. В иерархических структурах человек может ограничиваться рассмотрением только определенного уровня, не вдаваясь в детали реализации. Для сложной системы целесообразно моделировать два типа иерархии - *типовую* и *структурную.* Типовая иерархия отражает взаимосвязи "общее/частное". В объектно-ориентированном подходе ей соответствует иерархия классов. Структурная иерархия показывает связи типа "часть/целое". При объектно-ориентированном подходе ей соответствует иерархия объектов, образуемая атрибутами контейнерных классов.

Групповая разработка. Разработка сложной программной системы не может быть прерогативой одного человека. Для этой цели формируется группа, в которой каждый выполняет свои определенные функции. Иерархический характер сложных систем хорошо согласуется с принципом групповой разработки. В этом случае деятельность каждого участника проекта ограничивается соответствующим иерархическим уровнем. Применяемые инструментальные средства должны поддерживать групповую разработку. Для этого современные программные средства реализуются на комплексах с архитектурой клиент-сервер. В них должны быть предусмотрены возможности интеграции результатов работы отдельных участников проекта и защиты их от несанкционированного доступа.

Модифицируемость проекта. Сложные системы, имеющие достаточно долгое время жизни, обычно подвергаются многократной модификации. Это связано как с устранением ошибок, выявленных в процессе разработки, отладки или эксплуатации, так и с необходимостью внесения изменений и дополнений, вызванных изменениями внешних условий и требований к системе. Очевидно, что модификация сложных приложений может столкнуться с существенными трудностями ввиду значительного объема таких систем и большого числа взаимосвязей между их компонентами.

Сборочное **проектирование.** При разработке больших программных систем широко используется концепция сборочного проектирования, основанная на идее повторно используемых компонент [6]. Сборка прикладной системы из готовых компонент позволяет значительно сократить время разработки.

Использование стандартных СУБД. Современные интегрированные программные системы обычно используют в работе стандартные СУБД в основном реляционного типа, причем реализация таких систем обычно осуществляется в клиент-серверной среде. Интеграция прикладной системы с базой данных (БД) ставит перед разработчиками ряд дополнительных задач. Главной из них является преемственность, т.е. возможность использования в разрабатываемом приложении данных, накопленных ранее в БД. Кроме того, при разработке приложения в большинстве случаев возникает необходимость проектирования логической структуры новой БД. Для интегрированных систем с архитектурой клиент-сервер используются специальные инструментальные средства.

Особенности объектно-ориентированного подхода

Стремление избавиться от недостатков традиционного структурного подхода привело к развитию идей, основанных на объектной декомпозиции. Такой подход к разработке программных систем получил название объектно-ориентированного. В основе его лежат понятия "объект" и "класс" [1, 4, 11]. В реальном мире, а точнее в интересующей разработчика проблемной среде, в качестве объектов могут рассматриваться конкретные предметы, а также абстрактные или реальные сущности. Например, объектами могут быть покупатель, фирма, производящая определенные товары, банк, заказ на поставку.

Объект обладает индивидуальностью и поведением, имеет атрибуты, значения которых определяют его состояние. Так, конкретный покупатель, делая заказ, может оказаться в состоянии, когда денег на его счете не хватает для оплаты, а его поведение в этом случае заключается в обращении в банк за кредитом.

Каждый объект является представителем некоторого класса однотипных объектов. Класс определяет общие свойства для всех его объектов. К таким свойствам относятся состав и структура данных, описывающих атрибуты класса и соответствующих объектов, и совокупность методов - процедур, определяющих взаимодействие объектов этого класса с внешней средой. Например, описание класса "магазины" может включать такие атрибуты, как название и адрес, которые индивидуальны для каждого объекта этого класса - конкретного магазина; штат сотрудников; размер текущего счета, определяющий состояние объектов; методы: формирование заказов на поставку товаров, передача товара со склада в торговую секцию и т.д. Объекты и классы обладают характерными свойствами, которые активно используются при объектно-ориентированном подходе и во многом определяют его преимущества.

Инкапсуляция - скрытие информации [3]. При объектно-ориентированном программировании имеется возможность запретить любой доступ к атрибутам объектов, кроме как через его методы. Внутренняя структура объекта в этом случае скрыта от пользователя, объекты можно считать самостоятельными сущностями, отделенными от внешнего мира. Для того чтобы объ-

ект произвел некоторое действие, ему извне необходимо послать сообщение, которое инициирует выполнение нужного метода. Инкапсуляция позволяет изменять реализацию любого класса объектов без опасения, что это вызовет нежелательные побочные эффекты в программной системе. Тем самым упрощается процесс исправления ошибок и модификации программ.

Наследование - возможность создавать из классов новые классы по принципу "от общего к частному". Наследование позволяет новым классам при сохранении всех свойств классов-родителей (называемых в дальнейшем суперклассами) добавлять свои характеристики, отражающие их индивидуальность [3]. Сообщения, обработка которых не обеспечивается собственными методами класса, передаются суперклассу. Наследование позволяет создавать иерархии классов, являясь эффективным средством внесения изменений и дополнений в программные системы.

Полиморфизм - способность объектов выбирать метод на основе типов данных, принимаемых в сообщении [2]. Каждый объект может реагировать по-своему на одно и то же сообщение. Полиморфизм позволяет упростить исходные тексты программ, обеспечивает их развитие за счет введения новых методов обработки.

Объектно-ориентированная декомпозиция заключается в представлении системы в виде совокупности классов и объектов предметной области. При этом иерархический характер сложной системы отражается в виде иерархии классов, а ее функционирование рассматривается как взаимодействие объектов. Такой подход позволяет описать сложную систему наиболее естественным образом.

Жизненный цикл разработки приложения при использовании объектно-ориентированного подхода

Жизненный цикл объектно-ориентированной разработки программных систем содержит несколько этапов, но в отличие от структурного подхода в нем нет строгой последовательности их выполнения. Процесс принципиально носит итеративный характер, что полностью отвечает потребностям разработчиков (рис. П1.1).

Разработка начинается с *этапа обследования* - объектно-ориентированного анализа. Здесь определяются требования к системе и осуществляется анализ проблемной среды, в ходе которого определяются основные классы и объекты, которые составляют словарь проблемной среды. Результат обследования должен представлять достаточно полную модель системы.

После обследования начинается *объектно-ориентированное проектирование,* в ходе которого детализируется представление классов и объектов, полученных на этапе анализа. Определяются структуры данных, методы, отношения между классами, разрабатываются сценарии взаимодействия объектов. При проектировании системы могут вводиться новые классы и объекты, если это требуется для решения поставленных проблем. Результатом проектирования должны быть детальная модель системы, спецификации объектов, классов и отношений, достаточные для их программирования.

Рис.П1.1. Цикл разработки программного обеспечения с использованием объектно-ориентированного подхода

Программирование, тестирование и сборку системы Г.Буч [11] рассматривает как единый этап, называемый *эволюцией системы.* При объектно-ориентированном подходе имеется возможность быстрого создания прототипов проектируемой системы, постепенное развитие которых приводит к конечному результату. На этом этапе также возможно введение новых классов, изменение структур данных, добавление новых методов. Следует отметить, что программирование и тестирование отдельных компонентов системы возможно до завершения проектирования, что экономит время разработки. Современные объектно-ориентированные инструменталь-

ные средства, применяемые при разработке программных систем, обычно обладают достаточными возможностями по автоматизации действий, выполняемых на этом этапе. В частности, существует возможность автоматической генерации кодов программ.

Модификация системы может рассматриваться как отдельный этап. Для сложных систем возможность внесения изменений является естественным свойством, обеспечивающим их развитие. При объектно-ориентированном подходе модификация не требует полного пересмотра проекта, затрагивая лишь необходимые для этого классы и объекты.

Главная особенность жизненного цикла при объектно-ориентированном подходе заключается в том, что нет строгой последовательности выполнения отдельных этапов. При проектировании может выясниться необходимость дополнительного обследования, программирование и последующее тестирование могут потребовать возврата к проектированию. Такой метод, названный Г.Бучем *возвратным,* отражает итеративный характер процесса проектирования.

Преимущества и недостатки объектно-ориентированного подхода

Особенность процесса разработки современных сложных программных систем состоит в том, что центр тяжести смещается от программирования к более ранним этапам - анализу и проектированию, поэтому эффективность принятых методик анализа и проектирования имеет определяющее значение для судьбы проекта.

Достоинствами объектно-ориентированного подхода являются следующие.

Распараллеливание работ. Как отмечалось выше, программирование и тестирование отдельных компонентов системы возможно до завершения проектирования, что экономит время разработки. При программировании может возникнуть необходимость внесения изменений в существующие классы или потребоваться введение новых объектов или классов. В этом случае, вернувшись к этапу проектирования или даже к анализу, можно внести изменения и дополнений, не подвергая проект полной переработке.

Упрощение внесения изменений. В отличие от структурного подхода в объектно-ориентированном внесение изменений в проект имеет более локальный характер. В тех случаях, когда изменение носит характер уточнения, вводятся новые классы, наследующие поведение ранее созданных. Наследование (одно из основных свойств классов) позволяет в этих случаях не только не пересматривать ранее созданные объекты и классы, но даже обойтись без их повторной трансляции. В более сложных случаях, когда меняются методы, определяющие интерфейс классов, изменения в проекте будут более значительными, но и тогда они будут локализованы, затрагивая лишь классы, использующие эти методы.

Переносимость и гибкость архитектуры. Объектно-ориентированная декомпозиция, в результате которой приложение представляется в виде совокупности классов и объектов, обеспечивает гибкость архитектуры системы. В клиент-серверной системе объекты могут размещаться как на клиентских местах, так и на серверах. В гетерогенных сетях возможна реализация классов на компьютерах разных типов, а фиксированный интерфейс каждого класса, определяемый набором его методов, обеспечит правильность функционирования системы. Изменения конфигурации оборудования не потребуют внесения изменений в проект.

Повторное использование программных компонентов. Разрабатываемые в рамках проекта классы обычно отражают типовые проектные решения, поэтому их использование возможно и в других проектах. Возможность повторного использования программных компонентов - одно из наиболее привлекательных свойств объектно-ориентированного подхода. Библиотеки классов, отражающие программистский опыт в определенной области, позволяют значительно снизить объем программирования при разработке новых проектов. При наличии развитых библиотек классов проектирование и программирование новых приложений будет в основном сводиться к сборке системы из готовых компонентов.

Иерархический характер сложных программных систем позволяет значительно повысить эффективность повторного использования компонентов. При этом чем более высокого уровня объекты можно повторно использовать, тем большего эффекта можно достичь [12]. Для того чтобы повторное использование компонентов приносило свои плоды, разработчики программных систем должны [8]:

• осознавать выгоды такого подхода;

• знать, какие части задачи могут быть решены с применением уже существующих программных средств;

• заниматься поиском подходящих для повторного использования программ;

• стремиться непременно найти такие программы;

• использовать их даже в том случае, если они лишь частично совпадают с тем, что программист написал бы сам.

Следует отметить, что основные свойства классов и объектов -инкапсуляция, наследование и полиморфизм - полностью отвечают задаче повторного использования.

Естественность описания. Объектно-ориентированный подход позволяет описывать как статические, так и динамические отношения между объектами модели. По описанию предметной области, выполненному на естественном языке, легко выделить объекты и статические связи между ними. Объекты соответствуют существительным, а связи - глаголам и отглагольным формам. Например, фраза "фирмы выполняют заказы" позволяет выделить классы объектов "фирма" и "заказ" и отношение "выполнять" между ними типа M:N (многие к многим), так как фирма может выполнять много заказов, а заказ может быть выполнен разными фирмами.

Кроме того, свойства наследования и инкапсуляции позволяют каждому участнику проекта рассматривать модель на удобном для него уровне детализации. Руководители проекта могут работать с верхним уровнем модели, где отражаются только основные классы, объекты и связи. Другие разработчики или эксперты имеют возможность опускаться до более мелких, терминальных объектов, их свойств, связей, методов.

Недостатки объектно-ориентированного подхода лежат в области программирования. *Динамическое связывание,* предполагающее поиск метода в классе, которому принадлежит получающий сообщение объект, приводит к тому, что обращение к методу занимает в 1,75 -2,5 раза больше времени, чем в обычной подпрограмме [1,11]. Это, конечно, замедляет работу приложения. Однако, как указывает Г.Буч, динамическое связывание при использовании строго типизированных языков применяется примерно в 20% случаев от общего числа вызовов методов. Это позволяет снизить непроизводительные потери времени. В приложениях, где такие потери критичны, приходится прибегать к специальным программистским приемам.

Другой недостаток связан с излишней многочисленностью методов и их вызовов. Он вытекает из того, что для доступа ко многим атрибутам объектов (а к защищенным - всегда) используются отдельные методы Вызов метода высокого уровня абстракции приводит к тому, что в системе происходит каскад вызовов - от методов более высоких уровней иерархии к методам более низких уровней. Если время является ограничивающим фактором, такая ситуация может оказаться неприемлемой. Выходом может служить оптимизация начального варианта системы, связанная с уменьшением числа вызовов. Например, защищенные переменные можно сделать общедоступными и обращаться к ним напрямую, уменьшая тем самым число вызовов.

На компьютерах с сегментированной организацией памяти объектно-ориентированные системы при работе могут осуществлять интенсивный межсегментный обмен, что сказывается на их производительности. Это связано с тем, что классы обычно объявляются в разных файлах и соответственно реализуются в разных сегментах. Решение этой проблемы заключается в перераспределении классов по модулям. При этом логическое описание модели не изменяется.

Для задач реального времени, выполняющихся в высоком темпе, нежелательным является динамическое создание и удаление объектов, что также активно используется в объектно-ориентированных языках. В [1] предлагается выполнять размещение таких объектов априорно, в процессе создания программы, а не во время работы критичных по времени алгоритмов. Преодоление перечисленных затруднений связано с дополнительной работой программистов, но в то же время не требует очень больших усилий. В большинстве случаев действия, которые надо предпринять, достаточно очевидны. Кроме того, подобные проблемы возникают весьма редко. Следует также заметить, что объектно-ориентированные языки включают средства, позволяющие достичь более высокого быстродействия программ по сравнению с традиционными языками [1,11]. Таким образом, следует признать, что недостатки объектно-ориентированного подхода с лихвой *компенсируются* его достоинствами.

Проблемы, связанные с переходом к объектно-ориентированным технологиям, состоят в следующем.

Отсутствие немедленной отдачи Существует достаточно распространенное мнение, что объектно-ориентированный подход труден для понимания, поэтому переход на объектно-ориентированные технологии связан с большими затратами, которые не окупаются. В действи-

тельности дело обстоит по-другому. Традиционная и объектно-ориентированная технологии с точки зрения получаемых результатов по-разному ведут себя по отношению к затратам на их освоение. При использовании традиционных технологий некоторые результаты можно получить и при сравнительно небольших затратах, однако на определенной стадии наступает насыщение, когда даже значительные дополнительные затраты не приводят к существенному повышению эффективности. Объектно-ориентированные технологии не дают немедленной отдачи. Эффект от их применения начинает сказываться после разработки двух-трех проектов и накопления повторно используемых компонентов, отражающих типовые проектные решения в данной области.

На рис. П1.2 показана диаграмма роста эффективности разработок в зависимости от затрат для структурного и объектно-ориентированного подходов. При объектно-ориентированном подходе с приобретением опыта разработок кривая эффективности резко растет вверх

Рис. П1.2. Рост эффективности разработок по отношению к затратам при традиционном и объектно-ориентированном программировании

за счет рассмотренных выше преимуществ, в особенности из-за возможности сборки систем из готовых программных компонентов.

Диаграмма на рис. П 1.3 демонстрирует сокращение сроков разработок проектов В обоих случаях есть стремление к определенным пороговым уровням Но если для традиционного подхода снижение времени разработки связано в основном с ростом квалификации участников проектов, то при объектно-ориентированном подходе к этому прибавляется опыт использования типовых проектных решений.

Рис. П1.3. Снижение сроков разработки при традиционном и объектно-ориентированном подходах.

Психологические трудности. Переход на объектно-ориентированные технологии связан с преодолением психологических трудностей. Разработчикам и программистам приходится отвыкать от традиционных способов мышления, изучать новые языки программирования. Внедрение объектно-ориентированных технологий может натолкнуться на сопротивление некоторых участников проектов. В связи с этим необходимо уделять значительное внимание мерам, предпринимаемым для достижения этой цели [5].

ЛИТЕРАТУРА

1. *Буч Г.* **Объектно-ориентированное проектирование с примерами применения.** Пер с англ. - М.: Конкорд, 1992. - 519 с.

2. *Дункан Р.* **Замещение операторов и функций в Си и Си++** //PC Magazine /USSR/. - 1991. - №3. - С. 89 - 92.

3. *Дункан Р.* **Инкапсуляция данных и наследование свойств в Си++** //PC Magazine /USSR/. - 1991. - №3. - С. 99 - 104.

4. *Дункан Р.* **Си++ - новое мышление в программировании** //PC Magazine /USSR/.- l991. - №3. - С. 93 - 97.

5. **Как внедрить объектно-ориентированный подход.** The OOP Survial Guide./Agila C.A.//Computerworld-Moscow. - 1995. - №15. - С. 31.

6. *Липаев В.В., Позин Б.А., Штрик А.А.* **Технология сборочного программирования.** /Под ред. В.В.Липаева. - М.: Радио и связь, 1992 - 272 с.

7. **Метод "по спирали" быстро ведет к цели** //Деловой мир. - 1995. - № 23 - 24.

8. **Программы многократного использования становятся реальностью. Making reuse a reality**./Tibbetts J.,Bernstein B.//Компьютеруик. - 1995. - № 18. - С. 21, 30.

9. *Новоженов Ю.В.* **Объектно-ориентированный подход к разработке прикладных программных систем** //PC magazine. - 1995. - № 12.

10. *Boehm B.* **A spiral model of software development and enhancement** //IEEE Computer. - 1988. - № 25(5). - P. 61 - 72.

11. *Booch G.* **Object-Oriented Analysis and Design with Applications** // Bengamin/Cummings, Redword City, CA, USA, 1994.

12. *Mood J.* **Object Methods Tame Reengeneering Madness.** - Datamation. - 1995, May. - P. 43, 44, 48.

ПРИЛОЖЕНИЕ 2

Бизнес-Процесс "РЕИНЖИНИРИНГ" и интеллектуальное моделирование компаний

Бизнес-процесс "реинжиниринг", или БПР (в оригинале - "Business process reengineering"), начиная с 1990 г., вызывает активный интерес специалистов в области менеджмента и информационных технологий (ИТ). С 1994 г. в США проводятся ежегодные конференции по БПР. В настоящее время БПР взят на вооружение почти всеми ведущими компаниями мира. В частности, по данным Ernst & Young, 100 крупнейших банков Северной Америки затратят в 1997 г. около 2,9 млрд. дол. на реинжиниринг своих подразделений. За последние полтора года правительство США начало более 200 проектов по реинжинирингу. М. Хаммер и Дж. Чампи определяют реинжиниринг как "фундаментальное переосмысление и радикальное перепроектирование бизнес-процессов компаний для достижения коренных улучшений в основных актуальных показателях их деятельности: стоимость, качество, услуги и темпы" [5]. Подчеркнем, что речь идет не о небольшом усовершенствовании бизнес-процессов компаний, таком, например, как на 10 -100%, а о кардинальном повышении их эффективности в десятки раз. При этом реинжиниринг рассматривается как необходимое условие выживания современных компаний в условиях жесткой конкурентной борьбы на мировом рынке.

Необходимость реинжиниринга связывается с высокой динамичностью современного делового мира. Непрерывные и довольно существенные изменения в технологиях, рынках сбыта и потребностях клиентов стали обычным явлением, и компании, стремясь выжить и сохранить конкурентоспособность, вынуждены непрерывно перестраивать свою стратегию и тактику. БПР подверг ревизии принципы организации бизнес-процессов на основе разделения труда, предложенные А.Смитом в "Богатстве наций", и показал, что они неадекватны современным условиям. Дело в том, что принципы разделения труда, послужившие базой для успешного развития бизнеса в течение последних двухсот лет (эффективная организация железных дорог в Северной Америке в 1820-х годах; конвейер Генри Форда; принципы управления большими компаниями Альфреда Слоуна, внедренные в Дженерал Моторс, и др.), исходят из предположения об относительной *стабильности* существующих технологий, а также *постоянно растущем спросе на товары и услуги,* при котором *потребитель не имеет широкого выбора* и довольствуется уже самим наличием продукции. В подобных условиях наиболее эффективными оказались компании с иерархической пирамидальной структурой, организованные по функциональному признаку. Управление строится исходя из административно-командных принципов. При этом клиентам отводится самый нижний уровень иерархии, где они представлены безликим "массовым потребителем". Однако развитие современных технологий привело к *исчезновению стабильности,* а рост конкуренции - к *изменению роли потребителя.* Соревнование между производителями привело к дроблению массового рынка на относительно небольшие ниши, в которых уже потребитель диктует свои условия производителям, а не наоборот. Потребитель в настоящее время имеет существенно больший выбор не только товаров и услуг, но даже технологий (например, он может приобрести настольный издательский комплекс). В результате производитель вынужден непрерывно приспосабливаться как к новым технологиям, так и к постоянно меняющимся запросам своих клиентов: изменение бизнес-процессов превращается в практику повседневной жизни „компании. В этих условиях *инерционность пирамидальной структуры оказалась тормозом* на пути к выживанию компаний.

Решением проблемы являются смена базовых принципов организации компаний и переход к ориентации не на функции, а на процессы. Из всех концепций менеджмента, основанных на процессах, БПР рассматривается как наиболее эффективная концепция: М. Хаммер, автор термина "реинжиниринг", считает появление БПР революцией в бизнесе, которая знаменует отход от базовых принципов построения компаний, предложенных 200 лет назад А.Смитом, и превращает конструирование бизнеса в инженерную деятельность. Возможность такой революции обусловлена в первую очередь новейшими достижениями в области информационных технологий, в частности технологии динамических экспертных систем.

Каким же может быть вклад информационных технологий в организацию деятельности компании? Очевидный ответ - это автоматизация бизнес-процессов (БПА), в оригинале - *busi-*

ness process automation. Но автоматизация приводит лишь к ускорению существующих процессов, которое не может в большинстве случаев привести к тому многократному улучшению эффективности, которое предусматривает подлинный реинжиниринг (как правило, БПА дает улучшение на десятки процентов, в то время как БПР позволяет достичь выигрыша в сотни процентов). На самом деле информационные технологии позволяют изменить базовые правила организации работы (табл. П2.1).

Можно выделить два вида влияния информационных технологий на перестройку деятельности компаний и соответственно две группы технологий, имеющих пересечение (рис. П2.1). Технологии первой группы обеспечивают проведение БПР за счет автоматизации работ по реинжинирингу. Технологии второй группы обеспечивают появление новых процессов, позволяющих перейти к новым правилам работы в организациях.

Таблица П2.1

Влияние информационных технологий на переход к новым правилам работы компаний

Прежнее правило	Технология	Новое правило
Информация может появляться в одно время в одном месте	Распределенные базы данных	Информация может появляться одновременно в разных местах тогда, когда она необходима
Сложную работу могут выполнять только эксперты	Экспертные системы	Работу эксперта может выполнять специалист по общим вопросам
Необходимо выбирать между централизацией и децентрализацией бизнеса	Телекоммуникационные сети	Бизнес может пользоваться преимуществами централизации и децентрализации одновременно
Все решения принимают менеджеры	Средства поддержки принятия решений, доступ к базам данных, средства моделирования	Принятие решений становится частью работы каждого сотрудника (иерархическое принятие решений)
Для получения, хранения, поиска и передачи информации требуется офис	Беспроводная связь и переносимые компьютеры	Сотрудники могут посылать и получать информацию из того места, где они находятся
Лучший контакт с потенциальным покупателем - личный контакт	Интерактивный видеодиск	Лучший контакт с потенциальным покупателем - эффективный контакт
Чтобы найти некоторый объект, необходимо знать, где он находится	Автоматическое индексирование и отслеживание	Объекты сами информируют о своем местонахождении
Планы работ пересматриваются и корректируются периодически	Высокопроизводительные компьютеры	Планы пересматриваются и корректируются оперативно, по мере необходимости

Рис.П2.1. Информационные технологии в БПР

Чтобы пояснить, каким образом проведение БПР повышает эффективность работы компании, рассмотрим, как реинжиниринг изменяет перепроектируемые бизнес-процессы.

Несколько рабочих процедур объединяются в одну. Наиболее характерным свойством перепроектированных процессов является отсутствие технологии "сборочного конвейера", в рамках которой на каждом рабочем месте выполняются простые задания, или рабочие процедуры. Вместо этого процедуры, выполнявшиеся различными сотрудниками, интегрируются в одну - *горизонтальное сжатие процесса.*

На практике далеко не всегда удается сжать все шаги процесса к одной работе, выполняемой одним сотрудником. В этом случае создается команда, которая несет ответственность за данный процесс. Наличие в команде нескольких человек неизбежно приводит к некоторым задержкам и ошибкам, возникающим при передачи работы между членами команды. Однако потери здесь значительно меньше, чем при традиционной организации работ, когда исполнители процесса подчиняются различным подразделениям компании (возможно, располагающимся на различных территориях). Кроме того, при традиционной организации работ трудно (а иногда и невозможно) определить ответственного за быстрое и качественное выполнение работы.

Сравнительные оценки, выполненные компаниями, которые провели реинжиниринг, показывают, что переход от традиционной организации работ к выполнению процесса одним сотрудником уменьшает количество людей и ускоряет выполнение процесса примерно в 10 раз. К другим достоинствам горизонтального сжатия процессов относятся следующие:

• уменьшается количество ошибок и отпадает необходимость в специальной группе сотрудников для устранения этих ошибок;

• улучшается управляемость за счет уменьшения количества людей и четко распределенной ответственности между ними.

Исполнители принимают самостоятельные решения. В ходе реинжиниринга компании осуществляют не только горизонтальное, но и *вертикальное сжатие процессов.* Вертикальное сжатие происходит за счет того, что в тех точках процесса, где при традиционной организации работ исполнитель должен был обращаться к управленческой иерархии для принятия решений, он принимает решения самостоятельно.

При традиционной организации работ, ориентированной на выпуск массовой продукции, исходили из предположения, что исполнители не имеют ни времени, ни склонности, ни глубоких и всесторонних знаний, необходимых для принятия решений. Реинжиниринг отбрасывает это предположение, что представляется естественным при отказе от массового производства и при современном уровне образования. Наделение сотрудников большими полномочиями и увеличение роли каждого из них в работе компании приводят к значительному повышению их отдачи.

Шаги процесса выполняются в естественном порядке. Реинжиниринг процессов освобождает от линейного упорядочивания рабочих процедур, свойственного традиционному подходу, позволяя распараллеливать процессы там, где это возможно.

Процессы имеют различные варианты исполнения. Традиционный процесс ориентирован на производство массовой продукции для массового рынка, поэтому он должен исполняться единообразно независимо от исходных условий (т.е. при всех возможных входах процесса). Высокая динамичность рынка приводит к тому, что процесс должен иметь различные версии исполнения в зависимости от конкретной ситуации, состояния рынка и т.д.

Новые (перепроектированные) процессы, имеющие различные версии исполнения, начинаются с некоторого проверочного шага, определяющего, какая версия процесса наиболее подходит в данном конкретном случае (например, в простом случае выполняется автоматизированная процедура, в нетривиальном случае привлекается специалист и в сложном случае специалист приглашает экспертов).

Традиционные процессы обычно оказываются довольно сложными, так как они учитывают различные исключения и частные случаи. Новые процессы в отличие от традиционных являются более ясными и простыми, так как каждый вариант ориентирован только на одну соответствующую ему ситуацию.

Работа выполняется в том месте, где это целесообразно. В традиционных компаниях работа организуется по функциональным подразделениям: отдел заказов, транспортный отдел и т.п., и если конструкторскому отделу требуется новый карандаш, то он обращается с заявкой в отдел заказов. Отдел заказов находит производителя, договаривается о цене, размещает заказ, осматривает товар, оплачивает его и передает конструкторам. Описанный процесс является достаточно расточительным и медленным. Проведенный в одной из компаний США анализ показал [3], что при традиционном распределении работ внутренние затраты компании на приобретение батарейки стоимостью 3 дол. составили 100 дол. Кроме того, было установлено, что 35% всех заказов составляют заказы стоимостью менее 500 дол. После проведения реинжиниринга отделы перешли к самостоятельному заказу дешевых товаров. Итак, реинжиниринг распределяет работу между границами подразделений, устраняя излишнюю интеграцию, что приводит к повышению эффективности процесса в целом.

Уменьшается количество проверок и управляющих воздействий. Проверки и управляющие воздействия непосредственно не производят материальных ценностей, поэтому задача реинжиниринга - сократить их до экономически целесообразного уровня. Традиционные процессы насыщены подобными шагами, единственное назначение которых контроль за соблюдением исполнителями предписанных правил. Так, например, перед выполнением заказа соответствующий отдел компании проверяет право клиента сделать данный заказ, а также подлинность подписи клиента и финансовую состоятельность его подразделения или организации. При общей целесообразности проверок многие компании не задумываются над тем, сколько стоит проведение этих проверок. На практике довольно часто оказывается, что стоимость проверок и управляющих воздействий превосходит стоимость потерь, которые бы имели место при отсутствии проверок.

Реинжиниринг предлагает более сбалансированный подход. Вместо проверки каждого из выполняемых заданий перепроектированный процесс часто объединяет эти задания и осуществляет проверки и управляющие воздействия в отложенном режиме, что заметно сокращает время и стоимость проверок.

Минимизируется количество согласований. Еще один вид работ, не производящих непосредственных ценностей для заказчика, - это согласования. Задача реинжиниринга состоит в минимизации согласований путем сокращения внешних точек контакта и, как следствие, стирание граней между функциональными подразделениями.

"Уполномоченный" менеджер обеспечивает единую точку контакта. Механизм "уполномоченного" менеджера применяется в тех случаях, когда шаги процесса либо сложны, либо распределены таким образом, что их интеграция силами небольшой команды невозможна. "Уполномоченный" менеджер играет роль буфера между сложным процессом и заказчиком. Менеджер ведет себя с заказчиком так, как если бы он был ответственным за весь процесс. Чтобы выполнить эту роль, менеджер должен быть способен отвечать на вопросы заказчика и

решать его проблемы. Поэтому менеджер должен иметь доступ ко всем информационным системам, используемым в процессе, и ко всем исполнителям.

Преобладает смешанный централизованно-децентрализованный подход. Современные технологии дают возможность компаниям действовать полностью автономно на уровне подразделений, сохраняя при этом возможность пользоваться централизованными данными.

Важность объединения достоинств централизации и децентрализации можно проиллюстрировать на примере работы банков. При работе с крупными корпорациями многие банки осуществляют с одним и тем же клиентом независимые финансовые отношения через свои различные подразделения. Подобный децентрализованный подход может приводить к хаосу, так как каждое подразделение отслеживает только ту часть рынка, которая соответствует его профилю. В [5] описана реальная ситуация, в которой банк установил для одного из своих клиентов максимальный кредит в размере 20 млн дол. Вследствие децентрализованности этого банка каждое из его подразделений выдало этому клиенту по 20 млн. дол., т.е. клиент получил кредит, в несколько раз больший, чем планировал банк, что выяснилось только после банкротства клиента.

В определении реинжиниринга подчеркивается решающая роль радикального перепроектирования бизнес-процессов. Однако необходимо помнить, что реинжиниринг начинается с перепроектирования бизнес-процессов, но не заканчивается на этом. Дело в том, что фундаментальное изменение бизнес-процессов оказывает воздействие почти на все аспекты компании.

Компания может быть представлена в виде ромба с вершинами (см. рис. П2.2): бизнес-процессы; работы и структуры; системы управления и оценок; системы убеждения и ценностей. Вершина 1 ромба соответствует бизнес-процессам компании, т.е. способу, которым работа делается. Вершина 1 определяет вершину 2, которая характеризует природу выполняемых работ и то, как люди организованы для выполнения работ. В традиционной компании процессы разбиты на простые работы, выполняемые функциональными подразделениями. В новой компании процесс разбивается на сложные (многоплановые) работы, выполняемые командами процессов.

Рис.П2.2. Компоненты бизнес-системы

Для того чтобы люди, выполняющие работу, были заинтересованы в удовлетворении потребностей клиентов в товарах или услугах, необходимы продуманные системы управления и оценок, а также механизмы формирования системы ценностей и убеждений сотрудников. Системы управления и оценок, определяют, как оценивается эффективность работы и как работа оплачивается. Ценности и убеждения сотрудников должны способствовать эффективному выполнению процессов. Например, процесс исполнения заказов клиента, спроектированный так, что он быстро и точно исполняется, не будет эффективным, если исполнители убеждены, что наиболее важны скорость и точность. Таким образом система убеждений и ценностей оказывает влияние на процессы компании (вершина 1).

Итак, для успешного функционирования компании все четыре аспекта бизнес-системы должны быть согласованы. Рассмотрим теперь более подробно последствия реинжиниринга в контексте описанных выше четырех аспектов.

Переход от функциональных подразделений к командам процессов. По сути реинжиниринг объединяет в единое целое процессы, которые по предложению Адама Смита много лет назад были разбиты на отдельные простые части. В традиционно организованной компании люди распределяются по отделениям, отделам, лабораториям, группам и т.п., в которых они выполняют предписанные им функции (части процессов). Эта фракционность создает множество проблем, в частности проблему несогласованности и даже противоречивости целей различных групп людей. Реинжиниринг предлагает альтернативный подход, состоящий не в разделении людей по подразделениям, а в объединении людей в команды процессов, т.е. в группы людей, выполняющих совместно законченную часть работы - процесс. Команды процессов заменяют старые функциональные подразделения. В зависимости от сути выполняемых работ ис-

пользуются различные типы команд процессов. Рассмотрим наиболее часто встречающиеся типы команд.

Один тип команды объединяет некоторое число совместно работающих людей различных специальностей, выполняющих рутинную, повторяющуюся работу. В связи с тем, что в данном случае команда выполняет повторяющуюся работу, члены команды объединяются на длительное время. Такой тип команды используется компанией Bell Atlantic [5].

Другой тип команды объединяет людей для решения некоторой эпизодической и, как правило, сложной задачи. В этом случае команда создается на время решения задачи. Команды подобного типа называют *виртуальными командами.* При завершении проекта команды этого типа расформировываются, а их члены переходят в другие проекты и команды. Один человек может быть одновременно членом нескольких виртуальных команд, распределяя свое время между несколькими проектами.

Третий тип команды подобен первому описанному нами типу, но состоит из одного человека.

Работа исполнителя изменяется от простой к многоплановой. Люди, работающие в команде, отмечают, что их работа значительно отличается от работы, которую они исполняли в функциональном подразделении. Член команды в отличие от сотрудника традиционного подразделения, отвечающего за отдельные задания (части процесса), несет совместно с другими членами команды ответственность за весь процесс, что требует умения не только выполнять свое задание, но и понимать весь процесс в целом и уметь при необходимости выполнять не одно, а несколько заданий. Работа члена команды становится более содержательной, так как из нее в ходе реинжиниринга устраняются лишние проверки, согласования, ожидания, вызванные преодолением границ между подразделениями традиционной организации. Члены команды фокусируют свои усилия на потребностях пользователей, а не на потребностях начальства.

Требования к работникам изменяются: от контролируемого исполнения предписанных заданий к принятию самостоятельных решений. Традиционная компания требует, чтобы ее работники следовали предписанным правилам. Компания, завершившая реинжиниринг, требует, чтобы ее сотрудники не следовали предписанным правилам, а предлагали свои правила, т.е. члены команды уполномочены принимать самостоятельные решения. Если исполнители должны ждать указаний по их работе, то они не являются членами команды. Таким образом, передача полномочий исполнителям является обязательным условием проведения реинжиниринга. Проведение реинжиниринга влечет за собой изменение требований к сотрудникам, принимаемым на работу.

Изменяются требования к подготовке сотрудников: от курсов обучения к образованию. Традиционные компании готовят своих сотрудников на обучающих курсах, цель которых обучить, как выполнять некоторую конкретную работу или как управлять той или другой специфической ситуацией. В связи с многоплановостью и изменяемостью работ, ориентированных на процессы, компании должны заботиться не только о проведении обучающих курсов, но и о непрерывном образовании своих сотрудников. Действительно, при непрерывно изменяющемся окружении невозможно нанять людей, которые уже знают все, что от них может потребоваться.

Изменяются оценка эффективности работы и оплата труда: от оценки деятельности к оценке результата. В традиционной компании схема оплаты довольно прямолинейна: людям платят за отработанное время. Понятно, что это далеко не самый эффективный способ оплаты, однако при разбиении работы на простые задания компания не имеет возможности оценить эффективность узкого задания. Кроме того, увеличение эффективности узкоопределенного задания не всегда приводит к увеличению эффективности всего процесса. После проведения реинжиниринга команда отвечает за результаты процесса, и в этом случае компания может измерить эффективность работы команды и оплатить ее в соответствии с полученным результатом. Реинжиниринг приводит к тому, что компании пересматривают базовые предположения об оплате труда, свойственные традиционному подходу:

• эффективность работы сотрудника в текущем году не является гарантией его эффективной работы в следующем году. По этой причине базовая зарплата сотрудника меняется мало, награду за высокую эффективность он получает в виде премий;

• жалованье сотрудника определяется не столько временем, проведенным на работе, важностью выполняемой работы, трудовым стажем, количеством подчиненных и занимаемой должностью, сколько эффективностью его работы, оцениваемой по конечному результату.

Критерий продвижения в должности изменяется: от эффективности выполнения работы к способности (умению) выполнять работу. Одним из последствий реинжиниринга является проведение четкого различия между продвижением сотрудника и эффективностью его работы. Наградой за эффективность работы должна быть премия, а не продвижение по службе. Продвижение по службе есть функция от способностей сотрудника, а не от эффективности его работы, т.е. принцип компаний должен быть таков: "платим за эффективность, продвигаем за способности". Несмотря на очевидность этого принципа, он часто нарушается. Типичным является такое рассуждение: если N хороший программист, то он подходит на должность руководителя лаборатории программистов. Подобный вывод часто оказывается ошибочным, и компания получает плохого руководителя за цену хорошего программиста.

Изменяется цель работ: от удовлетворения потребностей начальника к удовлетворению потребностей клиентов. Реинжиниринг вызывает существенный сдвиг в культуре компании. Реинжиниринг требует от исполнителей убежденности, что они работают для клиентов, а не для своих начальников. Исполнители будут верить этому в той степени, в которой практика работы компании подтверждает это. Так, например, руководство фирмы Xerox Corporation не только говорит своим сотрудникам, что зарплату им платят клиенты, но и реализует это высказывание следующим образом: основная часть премии менеджеров зависит от степени удовлетворения ими клиентов.

Функции менеджеров изменяются от контролирующих к тренерским. В результате реинжиниринга бизнес-процессы становятся проще, а отдельные задания (шаги) процесса, выполняемые исполнителем, становятся сложнее. Усложнение работ, выполняемых исполнителями, приводит к тому, что уменьшается работа менеджеров по контролю за ходом выполнения процесса. Кроме того, в связи с тем, что команда процесса полностью отвечает за выполнение своего процесса, устраняются управляющие воздействия на исполнителей со стороны менеджеров. Функции менеджеров изменяются, их задача теперь состоит не в выдаче управляющих и контролирующих воздействий, а в помощи членам команды решать проблемы, возникающие у них в ходе выполнения процесса. Таким образом, менеджер выполняет функции тренера, который непосредственно не участвует в работе команды, но помогает команде выполнить ее работу с минимальными непроизводительными затратами. Именно этот вид деятельности требует от менеджера подлинного профессионализма.

Традиционная практика недооценивает как работу исполнителя, так и работу менеджера. Недооценка роли исполнителя выражается в том, что для него вершина успеха состоит в переходе из исполнителей в менеджеры, т.е. фактически традиционные компании оценивают управленческую деятельность выше, чем деятельность по производству товаров и услуг. Недооценка роли менеджера состоит в утверждении, что любой хороший исполнитель может стать хорошим менеджером. Ошибочность этого утверждения очевидна любому спортивному болельщику, знающему, что не любой, даже выдающийся спортсмен, может стать хорошим тренером.

Организационная структура компании изменяется от иерархической (многоуровневой) к "плоской". В традиционной компании организационная структура играет важную роль, так как она является механизмом, с помощью которого решаются основные проблемы компании. Действительно, в традиционной организации основной единицей является функциональное подразделение - совокупность людей, объединенных по подобию выполняемых ими задач (заданий). При этом компания как целое состоит из функциональных подразделений, организованных тем или иным способом. В так называемых функциональных компаниях все связанные функциональные отделы объединяются в единое функциональное отделение. Например, все отделы продаж объединяются в отделение продаж и т.д. Возможно объединение отделов по территориальному принципу, например Западное отделение компании.

Организационные структуры устанавливают границы взаимодействия между подразделениями и определяют иерархию принятия решений. Таким образом, процесс разбивается на отдельные части, выполняемые в различных подразделениях. При этом работа менеджеров в значительной степени состоит в контроле за исполнителями и в "склеивании" отдельных работ в единый процесс.

После проведения реинжиниринга значительно сокращается работа, выполняемая менеджерами (т. е. уменьшается требуемое количество менеджеров), и меняется ее характер (от контролирующей к тренерской). Действительно, менеджер, осуществляющий контролирующие функции, обычно не может работать более чем с семью подчиненными (менеджерами или исполнителями). Менеджер, осуществляющий тренерские функции, может работать примерно с тридцатью людьми. Изменение соотношения от 1 к 7 на 1 к 30 приводит к тому, что значительно сокращается количество управляющих уровней в иерархической структуре, в связи с чем важность организационной структуры уменьшается.

Административные функции изменяются от секретарских к лидирующим. В традиционной компании администрация оторвана от непосредственных исполнителей и клиентов, она выполняет функции секретаря, а не руководителя. Одним из последствий реинжиниринга является изменение роли руководящей администрации. Уменьшение количества управляющих уровней в иерархической структуре приближает администрацию к непосредственным исполнителям и клиентам. В перепроектированной компании успешное выполнение работы в основном зависит от членов команды, а не от функциональных менеджеров. Следовательно, администрация должна исполнять функции лидера, способствующего словом и делом укреплению убеждений и ценностей исполнителей.

Администрация несет общую ответственность за перепроектированный процесс, но не имеет непосредственного воздействия на людей, выполняющих этот процесс, так как члены команды и их тренер работают довольно автономно. Администрация осуществляет влияние на эффективное исполнение процессов за счет того, что при проектировании процессов с помощью систем управления компании обеспечивается мотивация членов команды.

Реинжиниринг компаний стал неотъемлемой особенностью нашего времени, что отразилось на жизненном цикле современной компании. Цикл начинается с реинжиниринга - кардинальной и революционной перестройки бизнес-процессов компании, сопровождающейся переходом на новые принципы построения организации. Этот вид деятельности требует выполнения специального проекта и создания команды по реинжинирингу, включающей как сотрудников компании, так и приглашенных консультантов. По достижении намеченных целей завершаются работы по проекту, и компания переходит к другому периоду своего развития - эволюционному, называемому усовершенствованием бизнеса. Этот этап характеризуется постоянными небольшими усовершенствованиями в бизнесе, выполняемыми в ходе текущей работы. После того как возможности эволюционного развития исчерпываются, компания вновь проводит реинжиниринг (как правило, проект охватывает уже не всю компанию целиком, а несколько функциональных подразделений). Таким образом, изменения организации работ в компании становятся частью ее повседневной жизни как реакция на постоянные изменения во внешнем окружении (рынок, уровень технологий, потребности клиентов, конкуренция).

Как отмечалось выше, успешный реинжиниринг приводит к многократному повышению производительности процессов, причем речь идет не о 10 - 20%-ном, а о десятикратном и более улучшении показателей. Но, к сожалению, около 50% попыток реинжиниринга ранее (до появления развитых инструментальных средств) заканчивались неудачей. Для выявления причин неудач и факторов риска был проведен ряд серьезных исследований [4,7]. Перечислим наиболее важные факторы, определяющие успех реинжиниринга.

Точность понимания задачи. Распространенная ошибка заключается в следующем: реинжинирингом считают то, чем он на самом деле не является. Естественно, что результаты автоматизации, реорганизации или уменьшения размерности не соответствуют тем, которые можно получить от реинжиниринга.

Мотивация. Следует с самого начала четко и ясно сформулировать основные цели реинжиниринга компании. Важен реалистичный взгляд на ожидаемые результаты и требуемые затраты времени и финансов. Сотрудники компании должны быть заинтересованы в проведении реинжиниринга: в этом смысле предпочтительнее проекты, которые рассматриваются с точки зрения роста и расширения фирмы, а не сокращения размеров и расходов, поскольку первые не вызывают большого сопротивления вводимым новшествам со стороны сотрудников.

Приверженность руководства компании идее реинжиниринга. Проект должен реализовываться под контролем одного из высших руководителей, глубоко заинтересованного в успешном его осуществлении. Руководство при этом должно быть готово пойти на риск.

Хорошо поставленное управление деятельностью компании. Отмечается, что наибольших успехов добиваются те компании, которые могли бы обойтись и без реинжиниринга за счет налаженного стратегического планирования, контроля финансов, освоения новых технологий и т.д. Опыт показывает, что реинжиниринг нередко проводится (и почти всегда успешно) в благополучных компаниях - компаниях, которые занимают прочное положение в текущей конкурентной борьбе и которым, казалось бы, нет смысла бороться за выживание. Проведение реинжиниринга для них означает переход на новейшие прогрессивные принципы организации своего бизнеса, то есть закладку прочного фундамента для последующего успешного развития. Отметим также, что проект не может осуществляться на основе самофинансирования - он должен иметь собственный бюджет.

Твердая методологическая основа при проведении БПР. Успеха достигали только те команды по реинжинирингу, которые следовали отработанным методам его проведения. Здесь важно выделить такие условия, как: твердое руководство проектом, четкое распределение ролей и ответственности между членами команды, использование адекватной технологической поддержки, параллельная разработка новой структуры предприятия и поддерживающей информационной системы, привлечение экспертов.

К сожалению, до сих пор последнему фактору уделялось недостаточное внимание. В настоящее время имеются несколько методологий БПР, разработанных различными консалтинговыми фирмами, и целый ряд инструментальных средств их поддержки. Однако после пятилетнего опыта проведения реинжиниринга специалисты сходятся во мнении, что не существует универсальной методологии, как не существует единого лекарства от всех болезней. В настоящее время ведущие консалтинговые фирмы располагают интегрированными совокупностями методов и приемов, а выбор тех или иных методов определяется особенностями конкретного проекта по реинжинирингу. Чем шире диапазон методов, тем больше у проекта шансов на успех.

Рие.П2.3. Этапы БПР

Выделяются шесть фаз (этапов) реинжиниринга (рис. П2.3).

1. Постановка задачи реинжиниринга - спецификация основных целей компании исходя из ее стратегии, потребностей клиентов, общего уровня бизнеса в отрасли (определяется на основе анализа какой-либо из ведущих фирм смежной отрасли, не являющихся конкурентами и готовых представить необходимую информацию о себе) и текущего состояния компании.

2. Создание модели существующей компании (называемое также обратным, или ретроспективным, инжинирингом). На этой фазе менеджеры с участием разработчиков информационных систем должны разработать детальное описание существующей компании, идентифицировать и документировать ее основные бизнес-процессы, а также оценить их эффективность.

3. Перепроектирование бизнес-процессов. Создание более эффективных рабочих процедур (элементарных заданий, из которых строятся бизнес-процессы), определение способов использования информационных технологий, идентификация необходимых изменений в работе персонала.

4. Разработка бизнес-процессов компании на уровне трудовых ресурсов. Здесь проектируются различные виды работ, подготавливается система мотивации, организуются команды по выполнению работ и труппы поддержки качества, создаются программы подготовки специалистов и т.д.

5. Разработка поддерживающих информационных систем. На этой фазе определяются имеющиеся ресурсы (оборудование, программное обеспечение) и реализуется специализированная информационная система (или системы) компании,

6. Внедрение перепроектированных процессов. Интеграция и тестирование разработанных процессов и поддерживающей информационной системы, обучение сотрудников, установка информационной системы, переход к новой работе компании.

Обратим внимание на то, что в проведение реинжиниринга вовлекаются специалисты двух типов - профессионалы в области реконструируемого бизнеса и разработчики информационных систем. Опыт реинжиниринга показывает, что по-настоящему успешное и новаторское внедрение информационных технологий является уникальным и творческим процессом: управляющие компаниями и специалисты-технологи, знакомясь с методами информационных технологий (ИТ), сами делают открытия относительно возможностей их использования в своем конкретном бизнесе [5,7]. В то же время создание высококачественных информационных систем требует участия профессионалов в области ИТ. Возникает проблема нахождения общего языка. Решение этой проблемы стоит на пути интеграции таких современных технологий, как объектно-ориентированное программирование, CASE-технологии, инженерия знаний, имитационное моделирование процессов и средства быстрой разработки приложений (в оригинале - *rapid application development, RAD).*

Большинство современных консалтинговых фирм основывают свои подходы к реинжинирингу исходя из CASE-технологии разработки информационных систем. Здесь можно отметить такие известные фирмы, как Gemini Consulting (методология Consruct, инструментальное средство BusinessWorks, построенное в среде VisualWorks Smalltalk) и Andersen Consulting (методология Eagle и набор инструментариев, обеспечивающих поддержку всех фаз проекта, за исключением четвертой). П. Хармон, рассматривая методологии этих фирм в своем обзоре [6], отмечает их ориентацию на профессионалов в области ИТ и направленность на разработку поддерживающих информационных систем.

Интересная методология предложена И.Якобсоном в его объектно-ориентированном подходе, основанном на примерах использования [7]. Ранее И.Якобсон разработал подход, известный как объектно-ориентированный инжиниринг программного обеспечения. Отметив аналогию между конструированием технических систем, информационных систем и бизнес-процессов крупных компаний, И.Якобсон разработал собственную методологию параллельного создания двух взаимосвязанных систем - бизнес-системы и поддерживающей ее информационной системы компании. Предусматривается создание по-следовательности моделей, описывающих обе системы как с точки зрения их использования (в первом случае - клиентами компании, во втором - пользователями информационной системы), так и с точки зрения их внедрения. При построении моделей используется общая методологическая база: модели первого типа описываются в терминах примеров использования (use case), а модели второго типа раскрывают особенности реализации этих примеров в терминах объектно-ориентированного моделирования. Объектно-ориентированные модели описываются на различных уровнях детализации. Совместная разработка моделей обеих систем при общей методологической базе позволяет естественным образом учесть взаимосвязь этих систем и осуществить параллельное и согласованное их создание и последующее развитие. Для поддержки реинжиниринга разработана объектно-ориентированная программная среда разработки Objectory с элементами CASE-технологии. Методология И.Якобсона и среда Objectory взяты на вооружение рядом консалтинговых фирм и многими разработчиками инструментариев поддержки БПР. Однако модели, создаваемые в соответствии с этой методологией, довольно сложны, и маловероятно, что управ-

ляющие компаниями могут работать с ними так же естественно и легко, как профессионалы в области ИТ.

Еще один известный подход, предложенный Дж.Мартином и Дж.Оделлом, был использован в ряде инструментариев, в том числе в системе OMW (Object Management Workbench) фирмы IntelliCorp. Его особенность состоит в сочетании CASE-технологии, объектно-ориентированного программирования и статических экспертных систем. Подход предусматривает создание диаграмм, представляющих потоки работ, структуры данных, взаимосвязи объектов, состояния и переходы в описании процессов. В отличие от всех предыдущих подходов здесь поддерживается процесс разработки программного обеспечения от диаграмм, описывающих модель бизнеса, до работающего кода. Тем не менее даже этот подход, как и все предыдущие, ориентирован на разработчика информационных систем, а не на менеджеров компаний, в которых проводится реинжиниринг.

Как уже отмечалось, для обеспечения активного участия менеджеров в проведении реинжиниринга целесообразно объединить ключевые достижения современных информационных технологий - *объектно-ориентированного программирования, CASE-технологии, имитационного моделирования процессов, инженерии знаний и средств быстрой разработки приложений.* Именно такая тенденция и наблюдается в настоящее время в развитии методологий и инструментальных средств БПР (см. также [2]).

Объектно-ориентированное моделирование в настоящее время признано базовой методологией БПР. Его особая роль объясняется следующим. Традиционно при создании информационных систем компаний разработчики отталкивались от данных. В результате используемые ими подходы к моделированию систем были ориентированы на описание данных о сущностях реального мира и их взаимосвязей, но не на поведение этих сущностей. Поскольку реинжиниринг ориентирован на процессы, а не на данные, традиционные подходы оказались неадекватны. Объектно-ориентированный подход является в на-стоящее время единственным подходом, позволяющим описывать как данные о сущностях, так и их поведение. Кроме того, он обеспечивает создание прозрачных, легко модифицируемых моделей бизнеса и информационных систем, допускающих повторное использование отдельных компонентов.

CASE-технологии использовались в реинжиниринге практически с самого начала. Однако их ориентация на разработчиков информационных систем привела к тому, что в настоящее время их начинают объединять с другими современными технологиями, в первую очередь с объектно-ориентированными.

Имитационное моделирование обеспечивает наиболее глубокое представление моделей для непрограммирующего пользователя, а также наиболее полные средства анализа таких моделей. Модели создаются в виде потоковых диаграмм, в которых представлены основные рабочие процедуры в компании и описано их поведение, а также информационные и материальные потоки между ними. Однако построение реальных имитационных моделей является довольно трудоемким процессом, а их детальный анализ (выходящий за рамки простого сбора статистики по срокам и стоимостям) зачастую требует от пользователя специальной подготовки. Для описания рабочих процедур может понадобиться дополнительное программирование. Таким образом, при попытке привлечь менеджеров к непосредственному использованию средств имитационного моделирования возникают определенные проблемы.

Чтобы преодолеть эти проблемы, в настоящее время начинают использовать методы *инженерии знаний.* С их помощью можно непосредственно представлять в моделях плохо формализуемые знания менеджеров о бизнес-процессах, в частности рабочих процедурах. Кроме того, решается проблема создания интеллектуального интерфейса конечного пользователя со сложными средствами анализа моделей. *Средства быстрой разработки приложений* позволяют сокращать время создания поддерживающих информационных систем и, следовательно, необходимы не только в ходе реинжиниринга компании, но и на этапе эволюционного развития, сопровождающегося постоянными модификациями и улучшениями информационных систем компании.

В настоящее время переход к использованию интегрированных методологий и средств только начинается. В числе консалтинговых фирм, поддерживающих интегрированные методологии, следует указать компанию Coopers & Lybrand (США). Предложенная ею методология SPARKS основана на применении баз знаний о типовых бизнес-процессах, которые могут ис-

пользоваться непосредственно менеджерами компаний. Компания разработала собственное инструментальное средство поддержки реинжиниринга на базе инструментального комплекса G2 фирмы Gensym, что позволило ей объединить возможности объектно-ориентированного программирования, анимации и имитационного моделирования с CASE-технологией.

Современные инструментальные средства можно разделить на 5 категорий.

1. Средства создания диаграмм и инструментарии низкого уровня (Micrografx: ABC Flowcharter; Scitor: Process Charter; High Performance Systems: iThink). Они являются дешевыми средствами, предназначенными для автоматизации первой и, возможно, второй фазы реинжиниринга. Чаще всего используются заинтересованными бизнесменами для описания существующего состояния компании и ее будущего. Не имеют связей со средствами быстрой разработки приложений; иногда включают элементы имитационного моделирования, но на довольно низком уровне.

2. Средства описания потоков работ (Action Technologies: Action-Workflow Analyzer; Viewstar: Process Architect). Позволяют проектировать планы работы над проектами; просты в использовании, но средства анализа получаемых планов довольно слабые.

3. Средства имитационного моделирования / анимации (CASI: Modsim; Systems Modeling: Arena; ProModel: ProModel; Gensym: Re-Think). Довольно дорогостоящие средства. Предлагают имитационное моделирование с помощью графических средств, библиотек специализированных подпрограмм и специализированных языков; используются для выполнения особо сложных проектов, в крупных фирмах или на уровне нескольких организаций.

4. CASE, объектно-ориентированные инструментарии и средства быстрой разработки приложений (Ptech: Framework, Oracle: Designer 2000; Popkin: Systems Architect). Многие разработчики CASE-средств и объектно-ориентированных средств начинают предлагать дополнения к своим инструментариям, позволяющие применять их в БПР. Эти инструментарии ориентированы исключительно на разработчиков информационных систем.

5. Многофункциональные средства, автоматизирующие основные этапы проведения БПР (Meta Software: Workflow Analyzer; IDS Prof. Scheer: ARIS Toolset; Interfacing Technologies: FirstStep; Gensym: Re-Think + G2). Фирмы-поставщики предлагают методологическую поддержку, организацию многопользовательского доступа к инструментарию, стыковку со средствами быстрой разработки приложений и даже возможности имитационного моделирования и анимации. Использование этих средств требует специальной подготовки. Бизнесмены не могут использовать их без посредничества специалистов в области ИТ и БПР.

Помимо специализированных ИС в ходе реинжиниринга используются и средства более общего назначения - инструментарии поддержки коллективных разработок, средства управления проектами, менеджеры процессов и др.

Как показано на рис. П2.4, ни одна из категорий специализированных ИС не позволяет охватить весь процесс реинжиниринга - от определения целей и перспектив до реализации процессов и генерации кодов информационной системы.

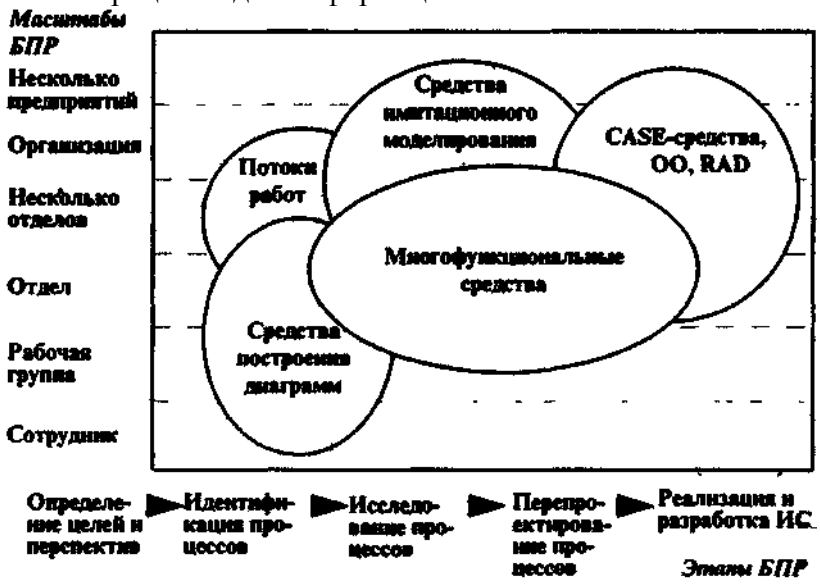


Рис.П2.4. Категории ИС поддержки БПР

Дешевые средства создания диаграмм и описания потоков работ (категории 1 и 2) рекомендуется использовать в начале реинжиниринга, пока не выявлены особенности конкретного проекта и не получен материал, достаточный для принятия решения о закупке более дорогого специализированного ИС. К достоинствам этих средств можно отнести их простоту и ясность, благодаря которым с ними могут работать непрограммирующие бизнесмены. Средства имитационного моделирования и анимации (категория 3) обеспечивают наиболее полный анализ динамики бизнес-процессов, а также прозрачность представления моделей бизнеса. При использовании этих средств сначала создаются детальные модели бизнес-процессов организации. Имитационные модели описывают не только потоки сущностей, информации и управления, но и различные метрики, например, частоту появления заявок, время выполнения каждой рабочей процедуры (возможно, с учетом случайных отклонений). Затем модели "проигрываются" в сжатом времени или пошаговом режиме. При отсутствии анимации модели могут создаваться как графически, так и аналитически. Если, есть анимация, то модели представляется в виде диаграмм процессов; в ходе "проигрывания" модели эти диаграммы, очереди, а также поведение системы в целом визуализируются, и благодаря этому пользователь может получать полное представление о работе исследуемой системы.

Развитые средства имитационного моделирования пришли в БПР из промышленности и космических исследований. Работа с ними требует от пользователей определенной математической подготовки, поэтому на практике поставщики подобных средств всегда предоставляют консалтинговые услуги. Прозрачность представления моделей, возможность глубокого их изучения (особенно в случае сложных ответственных проектов) делают методы имитационного моделирования и анимации одним из перспективных направлений БПР. Возможности имитационного моделирования (в том или ином объеме) включают и в инструментарий, относящийся почти ко всем категориям ИС БПР. Увеличивается число фирм, использующих методы имитационного моделирования при проведении реинжиниринга.

Включение методов БПР в традиционные средства разработки программного обеспечения (ИС категории 4) представляется нетривиальной задачей. Если базой для моделей, создаваемых в ходе реинжиниринга, являются процессы, то традиционный инжиниринг программного обеспечения основывается на данных. Наилучшим, решением проблемы оказалось использование объектно-ориентированного подхода к разработке программного обеспечения (Object-Oriented Information Engineering - OOIE), который позволяет описывать в объектах как данные, так и поведение (процессы).

Многофункциональные средства (категория 5) поддерживают наибольший объем функций, используемых при проведении БПР. Кроме этого многие из них обеспечивают хорошую методологическую поддержку, модульность, средства коллективного доступа к моделям и нередко стыковку со средствами разработки приложений. Поэтому при реализации больших проектов по реинжинирингу рекомендуется использовать именно эти средства.

Рассмотрим основные возможности, предоставляемые многофункциональными ИС. Все ИС, относящиеся к категории 5, имеют хорошие средства спецификации процессов (поддержка методологии IDEF, потоки работ в сочетании с объектной ориентацией и т.д.). Реализованы средства оценивания процессов. Некоторые из рассматриваемых ИС включают средства имитационного моделирования. Реализованы, хотя и не в полной мере, средства оценивания по рабочим процедурам. Почти во всех ИС имеются средства анализа критического пути. Методологическая поддержка (в том или ином объеме) присутствует во многих ИС. Почти всеми ИС поддерживается режим коллективной разработки моделей. Предусматривается стыковка со средствами разработки приложений. По степени реализации перечисленных функций наиболее удачными являются такие системы, как Re-Think и G2 (Gensym), Workflow Analyzer (Meta Software) и Process Wise (ICL). Однако лидер в этой группе средств еще не определился.

Безусловно, средства именно этой категории рекомендуется использовать в сложных проектах по реинжинирингу. Однако следует учитывать, что в настоящее время идет активное их развитие - в части простоты использования, полноты средств имитационного моделирования, стыковки со средствами разработки приложений и т.д. В этом секторе особый интерес представляет разработка фирмы Gensym, предназначенная для поддержки БПР, - инструментальное средство ReThink [1]. В этой системе объединены возможности ключевых современных информационных технологий: графический объектно-ориентированный язык для описания моделей и

проектов, средства анимации и имитационного моделирования реконструируемых процессов, методы искусственного интеллекта для полного и адекватного представления экспертных знаний о процессах. Все это открыло доступ к непосредственному моделированию и реконструированию бизнес-процессов новой группе пользователей - менеджерам. Сочетание прозрачных средств интерактивной графики с мощными возможностями моделирования процессов в реальном времени позволяет им самостоятельно, без помощи программистов воплощать свои идеи в виде работающих моделей процессов.

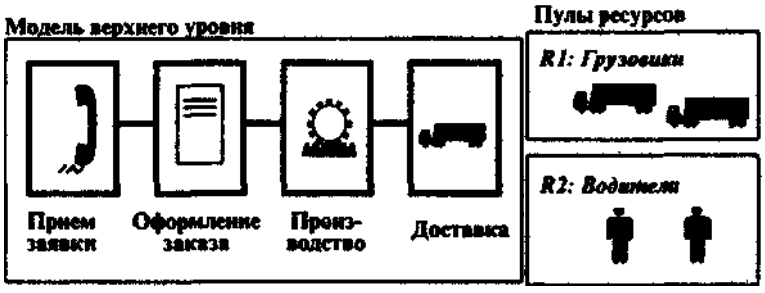
Система ReThink построена на базе инструментального комплекса G2 (см. гл. 9), Таким образом, она является проблемно-ориентированным приложением комплекса G2, которое позволяет разработчикам использовать не только специализированные средства моделирования бизнес-процессов, но и универсальные средства комплекса по созданию интеллектуальных объектно-ориентированных систем управления реального времени.

Для представления моделей бизнес-процессов используются диаграммы, состоящие из блоков и соединений. Блоки представляют задачи в бизнес-процессах, а соединения - потоки сущностей: документов, информации, а также предметов, фигурирующих в бизнесе (например, запасных частей или упаковок с отпускаемой продукцией). В системе реализован ряд стандартных блоков, которые могут быть использованы в качестве сборочных элементов для построения работающих моделей любых процессов, например: источник заявок, принятие решения, обработка задания. Свойства и поведение блоков могут описываться как точными, так и случайными величинами. В случае необходимости разработчик может переопределять поведение блоков или задавать новые их классы с помощью базовых средств комплекса G2. В системе ReThink реализованы средства анимации моделей.

Объектная ориентация системы ReThink позволяет создавать понятные и наглядные модели бизнес-процессов, что существенно упрощает освоение и использование системы непрограммирующими пользователями. Объекты, построенные в результате моделирования бизнес-процессов, являются естественной основой для проектирования информационных систем поддержки этих процессов. В этом смысле средства системы ReThink могут рассматриваться как развитие CASE-средств. ReThink поддерживает анимацию потоков работ в ходе моделирования деятельности компании. Благодаря этому менеджер имеет возможность непосредственно наблюдать функционирование моделей, что повышает степень его доверия к результатам моделирования.

ReThink поддерживает создание иерархических моделей, позволяющих описывать процессы с различной степенью детализации. Это обеспечивает простоту и естественность при создании сложных моделей больших компаний (рис. П2.5). Все элементы моделей, включая ресурсы процессов, могут модифицироваться непосредственно во время исполнения. Результаты изменений можно увидеть сразу же после их введения.

ReThink позволяет формировать стоимостные и временные характеристики различных проектов для объективного их сравнения, а также проверять гипотезы "Что, если". Для анализа работы моделей предусмотрен целый набор инструментариев: блоки-датчики для сбора данных, блоки-установщики значений атрибутов сущностей, графики для наглядного отображения результатов моделирования, всевозможные просмотровые табло из стандартных средств комплекса G2. С помощью датчиков можно снимать такие показатели, как длительность



Подмодель "Доставка"

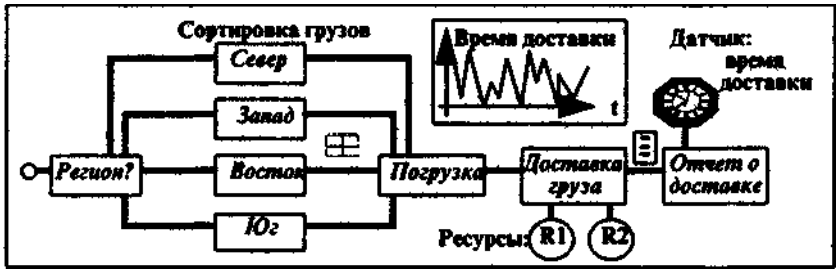


Рис.П2.5. Пример иерархической модели в системе ReThink

цикла обработки сущности на том или ином этапе, стоимость обработки, а также любы е другие свойства, определенные разработчиком модели. Для отсева шумов и выявления тенденций можно использовать специальные блоки-фильтры.

Для проверки гипотезы "Что, если" в системе реализован механизм, сценариев. Сценарии позволяют исследовать зависимость поведения одной и той же модели от поведения внешнего мира (например, частоты поступления заявок, сложности этих заявок и т.д.) и каких-либо параметров этой модели (например, количества транспортных средств или численности служащих, занятых оформлением заказов). Варьируемые параметры и измеряемые показатели выносятся на отдельное окно сценария, после чего в результате прогона модели автоматически формируется отчет. Кроме этого ReThink позволяет использовать сценарии для объективного сравнения альтернативных проектов: один и тот же сценарий, описывающий некоторое заранее заданное поведение внешнего мира, может использоваться для прогона различных моделей. Результаты прогона, вынесенные в отчет, являются основой для сопоставления и оценки этих моделей.

ReThink поддерживает коллективную работу с приложениями на основе архитектуры клиент-сервер с помощью клиентной системы Telewindows комплекса G2. Коллективная разработка и использование приложений имеют принципиальное значение при проведении глобального реинжиниринга крупной компании или объединения, например нескольких компаний в рамках отрасли.

Как и инструментальный комплекс G2, система ReThink функционирует на большинстве рабочих станций в среде Unix, системах OPEN VMS, а также на Pentium PC в среде Windows NT, Windows 95 и Windows 3.1. Система Telewindows позволяет обращаться к системе с персональных компьютеров Intel 386/486 в среде Windows 3.1.

При создании системы ReThink фирма Gensym не ставила своей целью предложить какую-либо конкретную методологию реинжиниринга. Ее задача - создание удобного универсального средства для реализации различных методологий. Система ReThink адресована в первую очередь консалтинговым фирмам и информационным подразделениям крупных компаний для воплощения их оригинальных идей в области реинжиниринга. Система предоставляет возможность развития средств инструментального комплекса G2 - вплоть до реализации новых нестандартных средств моделирования и анализа. Особый интерес представляет создание предметно-ориентированных баз знаний о типовых бизнес-процессах. Например, большое значение имеет реализация стандартных средств представления финансово-экономической деятельности организаций в нашей стране.

Система ReThink успешно используется в ряде компаний, в том числе в патентном ведомстве США и компании Xerox, проведшей реинжиниринг отделения по закупке сопутствующих материалов с годовым оборотом в 3 млрд. дол. В компании Xerox при проведении реинжиниринга сначала использовался пакет ABC Flowcharter. Построенная модель работы отделения включала 17 процессов и 314 рабочих процедур. Анализ модели показал, что 70 % процедур оказались непроизводительными. Затем была разработана новая модель процессов закупки, включающая всего 42 рабочие процедуры. Столкнувшись с таким существенным сокращением количества процедур, руководство компании поставило вопрос о работоспособности новой организации: не встанут ли перед компанией серьезные непредвиденные проблемы после того, как она сделает основные капиталовложения в реконструкцию отделения? Чтобы обосновать предложенный проект, было решено использовать систему ReThink, с помощью которой предполагалось исследовать имитационную модель предлагаемой организации работы отделения. В результате несколько процессов пришлось снова перепроектировать. Таким образом, использо-

вание мощных средств моделирования привело к явному выигрышу в качестве проекта, следовательно, снизило риск неудачи при проведении реинжиниринга.

Как научно-практическое направление БПР впервые появился в США и за пять лет превратился в одну из ведущих и активно развивающихся отраслей информатики. В настоящее время начинается продвижение консалтинговых услуг и инструментариев по БПР на российский рынок. Применение мирового опыта построения эффективных компаний представляет огромную ценность для нашей страны, проводящей глобальную экономическую реформу и активно внедряющейся в мировую экономическую систему. Практика БПР показала, что реинжиниринг не только необходим, но и возможен. Но для успешного его проведения необходимо использование обоснованных методологий и современных инструментальных средств, адекватных решаемым задачам.

ЛИТЕРАТУРА

1. *Попов Э.В.. Шапот М.Д.* **Реинжиниринг бизнес-процессов и информационные технологии** // Открытые системы. - 1996. - № 1.

2. *Шапот М.Д.* **Инструментальные средства поддержки реинжиниринга биз-нес-процессов //** Материалы семинара "Динамические интеллектуальные системы в управлении и моделировании". - М.: ЦРДЗ, 1996.

3. *Davenport T.H.* **Business Innovation, Reengineering Work through Information Technology.** - Boston: Harvard Business School Press, 1993.

4. *Flynn K.* **Critical Success Factors for a Successful Business Reengineering Project** //CASE World Conference Proceedings. - Boston, 1993, October.

5. Hammer M. and Champy J. **Reengineering the Corporation: A Manifesto for Business Revolution.** - New York: HarperCollins, 1993.

6. *Harmon P.* **Business Process Reengineering with Objects - Part 2 //**Object-Oriented Strategies. - 1995. - Vol. 5. - № 1.- P.1 -13.

7. *Jacobson I., Ericsson M., Jacobson A.* **The Object Advantage: Business Process Reengineering with Object Technology** //ACM Press. - Addison-Wesley Publishing, 1995.

ПРИЛОЖЕНИЕ 3

Нейросетевая технология

Особенности нейросетей

Причины шумного успеха искусственных нейронных сетей во многом остаются загадочными. Отбросим момент рекламы, амбиции исследователей и попытаемся выяснить, имеются ли у нейронных сетей реальные преимущества перед традиционными методами обработки информации. Одним из главных преимуществ нейронных сетей всегда считалась возможность распараллеливания вычислений. Однако в последние годы нейронные сети эмулируются с помощью обычных последовательных машин не только для исследовательских целей, но и для практического применения. Очевидно, это преимущество не является столь уж важным, если от него так легко отказываются. Не является таким преимуществом и возможность обучения на примерах обучаться могут и последовательные машины. Здесь, однако, следует заметить, что для нейросети исходная информация может быть значительно меньше - нейросеть способна начинать обучение буквально с нуля при минимуме сведений о свойствах объекта. Благодаря этому программирование сводится к выбору конфигурации сети (числа нейронов в каждом слое) и начальных значений весовых коэффициентов; все остальное достигается обучением.

Возможно, главное достоинство нейросетей в том, что они предоставляют в руки пользователю некий *универсальный нелинейный элемент с* возможностью широкого изменения и настройки его характеристик [10, 11]. Располагая своего рода конструктором из таких элементов и соединяя их в сеть, пользователь, с одной стороны, получает возможность широкого изменения ее характеристик, а с другой - может особенно не задумываться над процессами, происходящими в этой сети. Им заранее гарантированы целенаправленность и оптимальность, приводящие в конечном итоге к достаточно приемлемому результату. Чем-то нейросеть напоминает язык программирования высокого уровня, да по сути и является разновидностью такого языка, освобождающего пользователя от необходимости вникать в детали производимых операций. Появление нейросетей укладывается в общую для всей информационной индустрии тенденцию - переход от деталей к крупноблочному строительству (Case-системы, объектно-ориентированные технологии и т.п.).

Набор нелинейных адаптивных элементов позволяет моделировать любое нелинейное преобразование и настраивать его на различные задачи автоматически путем изменения параметров в процессе обучения. Причем в последнее время наблюдается тенденция использовать для настройки не эмпирически найденные приемы (типа правила Хебба, обратного распространения ошибки и т.п.), а универсальные и хорошо отработанные математические методы поиска экстремума целевой функции в пространстве параметров. Это касается и выбора целевой функции: переход от частных эмпирически найденных форм (аналог энергии в сетях Хопфилда, суммарная квадратичная ошибка в методе обратного распространения) к более общим.

Место нейронных сетей в системах обработки информации можно указать по аналогии со структурой человеческой психики: оно соответствует низшему интуитивному уровню реакции, когда требуется быстрый ответ на достаточно стандартную ситуацию. Если ответ не найден или система сомневается в его правильности, то управление передается более высокому логическому уровню. Ему соответствует экспертная система, располагающая широкой базой знаний и способная делать более обоснованные выводы.

Нейронные сети способны решать такие задачи, как распознавание образов, выделение сигнала на фоне шума, исправление ошибок, управление сложной адаптивной системой управления при невозможности формализовать экспертные знания или при отсутствии таковых и т.п. Все это уже находит широкое практическое применение (некоторые примеры приведены ниже). Нейросеть может запоминать действия опытного оператора, управляющего сложной системой, а затем воспроизводить их, проявляя необходимую гибкость, сменяя образцы поведения и выбирая среди них тот, который наиболее близок и адекватен текущей ситуации. При этом нет необходимости алгоритмизировать деятельность оператора, чтобы затем на ее основе строить программу управления: система схватывает формы поведения целостно как неразложимое целое и создает для их реализации соответствующие структуры.

В общем случае в поведении такой системы следует различать три задачи [9]:

• обучение и запоминание поведенческих образцов (эталонов), задаваемых внешними условиями. При этом происходят образование и модификация связей между элементами;

• распознавание внешней ситуации, отнесение ее к одному из запомненных эталонов, выбор соответствующего поведенческого образца;

• реализация выбранного эталона поведения, поддержание эталонных значений переменных, возвращение к ним после возмущений, исправление ошибок и нейтрализация помех, создаваемых внешней средой. В частном случае третья задача может отсутствовать и работа системы может завершаться распознаванием ситуации.

Свойства нейрона

С конструктивной точки зрения *нейрон,* являющийся основным элементом нейросети, - это устройство для получения нелинейной функции нескольких переменных x_i с возможностью настройки его параметров c_j в достаточно широком диапазоне [8]:

$$y = f(x_1, x_2, \ldots, x_m, c_1, c_2, \ldots, c_n) \quad (1)$$

Однако традиционно нейрон описывается в терминах, заимствованных из физиологии. Согласно этим представлениям нейрон имеет один выход s_i и несколько входов (синапсов), на которые поступают внешние воздействия x_i (от рецепторов и других нейронов). Он умножает входное воздействие на весовой коэффициент c_{ij} (проводимость синапса) и суммирует взвешенные входы:

$$s_j = \sum c_{ij} x_i + c_{0j} \quad (2)$$

Выходная величина y_j является некоторой функцией от этой суммы: $y_j = f(s_j)$. Ее называют *функцией активации* или *передаточной функцией.* Вид этой функции является важнейшей характеристикой нейрона. В простейшем случае - это *линейная* зависимость (рис. П3.1, а):

$$y_j = k s_j = k(\sum c_{ij} x_i + c_{0j}) \quad (3)$$

Такая зависимость использовалась в первых моделях персептрона Ф.Розенблатта [17]. Несмотря на ряд первоначальных успехов, теоретический анализ возможностей персептрона, проведенный М.Минским и С. Пейпертом [16], показал, что персептрон не является универсальным устройством для распознавания и, в частности, принципиально неспособен решить целый ряд весьма простых задач. Причиной этого является именно линейный характер активационной функции.

Еще в работе У.Мак-Каллока и У.Питтса [15] использовалась *ступенчатая* функция активации: если сумма s_j выше некоторого порогового значения c_{0j}, то выход y_j равен единице, в противном случае - минус единице (или нулю). Формально это можно описать с помощью следующей зависимости (рис. П3.1,б):

$$y_j = sgn(s_j) = sgn(\sum c_{ij} x_i + c_{0j}) \quad (4)$$

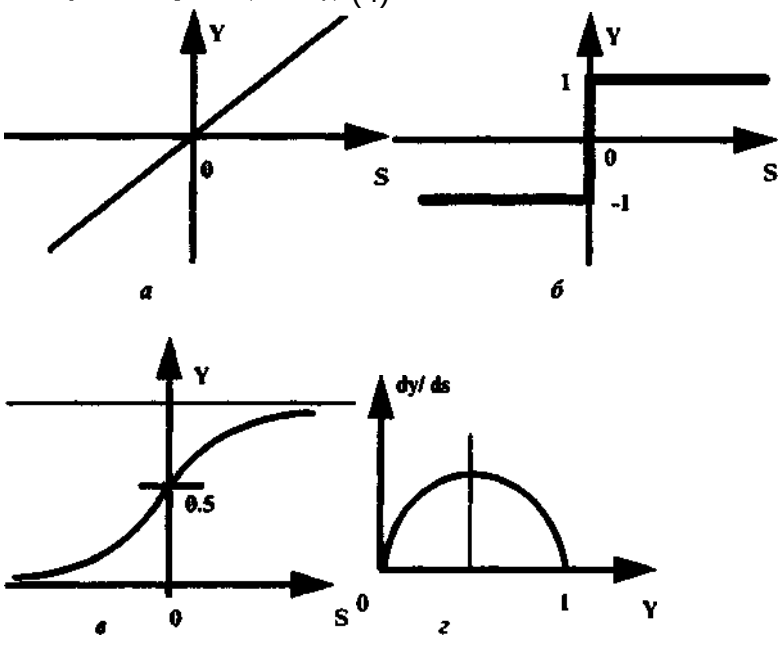


Рис.П3.1. Функции активации нейронной сети

а - линейная функция; *б* - ступенчатая функция,

в - сигмоидальная функция; *г* - производная от сигмоидальной функции

В настоящее время в качестве активационной функции чаще используют близкую к ступенчатой, но более гладкую зависимость, которую называют *сигмоидальной, или логистической,* функцией (рис. П3.1, в). Обычно она описывается следующим выражением:

$y = 1/(1+ e^{-ks})$. (5)

Встречаются и другие выражения, например,

$y = s/(1+k|s|)$, (6)

где $|s|$ - абсолютная величина s, $k > 0$.

Параметр k задает крутизну зависимости y от s: чем больше k, тем ближе сигмоида к пороговой функции; чем меньше k, тем ближе она к линейной. Таким образом, сигмоида является некоторым компромиссом между линейной и ступенчатой функцией, сохраняющим достоинства обеих. Подобно ступенчатой функции, она нелинейна, и это важно, поскольку только нелинейные функции позволяют вычленять в пространстве признаков множества сложной формы, в том числе невыпуклые и несвязные. Но в то же время сигмоида в отличие от ступенчатой функции переходит от одного значения к другому без разрыва, как это имеет место и в линейной функции. Это обстоятельство оказывается чрезвычайно важным при поиске экстремума целевой функции в пространстве нейронных параметров: в этом случае зависимость целевой функции от параметров также оказывается гладкой, и в каждой точке пространства может быть вычислен градиент целевой функции, указывающий направление поиска экстремума.

Производная от сигмоидальной функции, характеризующая силу связи между s и y, также имеет простой вид:

$dy/ds = ky(1-y)$. (7)

Эта величина обращается в нуль на границах Диапазона изменения y при y=0 и y=1 и достигает максимума в середине диапазона, т. е. связь между переменными наиболее сильна в середине диапазона и ослабевает к его краям (рис. П3. 1, *г*).

Нейроны организуются в сеть (рис. П3.2) за счет того, что выход i-го нейрона (y_i) соединяется с одним из входов (x_i) другого j-го нейрона. При этом выходная переменная y_i отождествляется с входной переменной x_i. Поэтому в дальнейшем будем использовать оба обозначения в зависимости от того, рассматривается ли данная i-я переменная как входная или как выходная. Весовой коэффициент c_{ij} ("синаптический вес") характеризует знак и силу связи между переменными x_i и x_j. Возможна и обратная связь, при которой выход j-го нейрона соединяется с j-м входом j-го нейрона. На рис.П3.2 эти связи не представлены. В общем случае коэффициент связи c_{ji} не обязательно равен c_{ij}.

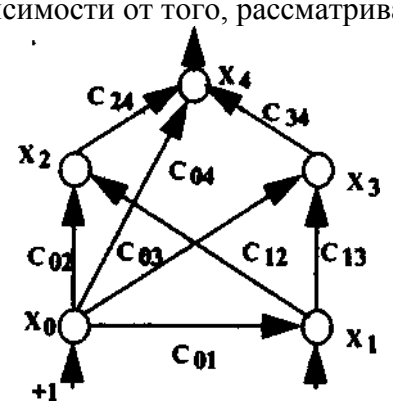


Рис. П3.2. Пример нейронной сети

Важнейшим свойством нейрона является его *пластичность* - возможность изменять параметры в процессе обучения. В ранних работах по нейросетям обычно различали два типа пластичности: синаптическую (изменение c_{ij}) и нейронную (изменение высоты порога нейрона c_{0j}). В настоящее время пороговую пластичность обычно сводят к синаптической с помощью следующего приема. К числу входов j-го нейрона добавляют еще один фиктивный x_0, не связанный ни с каким реальным рецептором (см. рис.П3.2). На этот вход подают постоянный сигнал, равный +1. Весовой коэффициент этого входа c_{0j} модифицируют в процессе обучения по общим правилам. Модификация этого коэффициента равносильна смещению порога нейрона.

Еще в 1949 г. Д. Хеббом [19] было предложено естественное *правило модификации весовых коэффициентов: если два нейрона возбуждаются вместе, то сила связи между ними воз-*

растает; если они возбуждаются порознь, то сила связи между ними уменьшается. Правило оказалось настолько удачным, что до сих пор используется в различных моделях нейронных систем. Формально это правило может быть описано следующим образом. Пусть время обучения разбито на такты и в *k-м* такте две переменные нейросети (состояния двух нейронов) имели значения x^k_i и x^k_j. Тогда вес связи между переменными возрастает на величину

$$\Delta c^k_{ij} = x^k_i x^k_j \quad (8)$$

В случае двоичных переменных приращение равно либо +1 (при совпадении знаков x^k_i и x^k_j), либо -1 (когда знаки различны). Если начальный вес связи был равен нулю, то вес связи к *p*-му такту равен:

$$c_{ij} = \sum_{1}^{p} x^k_i x^k_j \quad (9)$$

где x^k_i, x^k_j - состояния двух нейронов в *k*-м такте; *p* - число тактов обучения.

Использование нелинейных элементов

Один из самых неожиданных результатов анализа М.Минского и С.Пейперта состоял в том, что персептрон, построенный на линейных функциях активации, не может воспроизвести такую простую логическую функцию, как исключающее ИЛИ (XOR). Это функция двух аргументов $y(x_1, x_2)$, каждый из которых может быть нулем или единицей. Всего, следовательно, возможны четыре комбинации значений аргументов: 00, 01, 10, 11. В пространстве признаков они расположены по углам единичного квадрата (рис. П3.3). Функция $y = (x_1$ XOR $x_2)$ равна единице, когда равен единице один из аргументов, но не оба сразу. Таким образом, множество точек (01, 10) относится к классу, где $y = 1$, а множество (00,11) - к классу $y = 0$. В пространстве признаков элементы этих классов лежат на противоположных углах квадрата и никакая линейная функция $y(x_1, x_2)$ от этих признаков не способна разделить эти два класса. В этом случае не помогает и использование второго слоя нейронов, так как произведение двух линейных преобразований снова дает линейное преобразование, обладающее теми же недостатками. Выходом является использование нелинейных элементов.

Для иллюстрации этого вывода рассмотрим простой пример, когда пространство признаков является одномерным [11]. Даже в этом случае легко построить задачу, которая не может быть решена с использованием только линейных функций активации.

Пусть в одномерном пространстве признаков x_1 элементы одного класса (нулики) расположены вокруг элементов другого класса (крестиков) (рис. П3.4), т.е. множество нуликов является несвязным. Необходимо построить такую функцию $f(x_1)$, которая принимала бы положительное значение на крестиках и отрицательное - на нуликах. Это позволит разделить все пространство *x* на *три* области и затем крайние области (отрицательные значения $f(x_1)$) объединить в один класс, а среднюю (положительные значения *a*) отнести к другому. Желаемая (идеальная) зависимость $y'(x)$, решающая эту задачу, показана на рис. П3.4,*а*. Как видно из рисунка, она является нелинейной: нужна кривая по меньшей мере второго порядка,

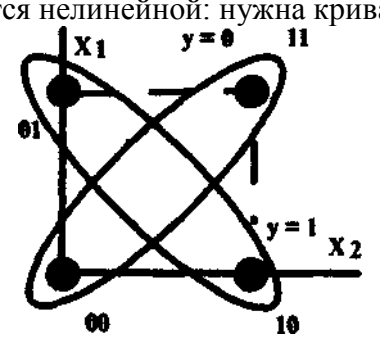


Рис. П3.3. Пространство признаков для функции $y = x_1$ XOR x_2

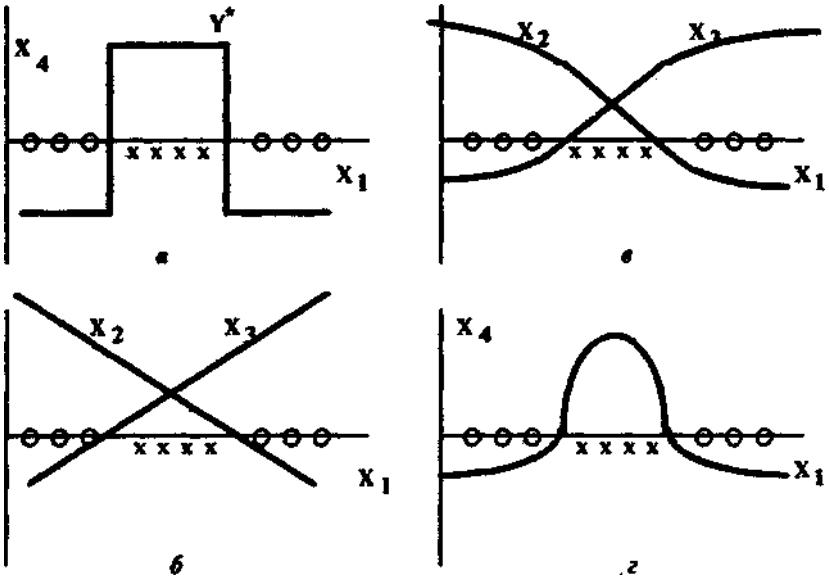


Рис.П3.4. Использование нелинейных функций активации:

а - идеальная зависимость; б - линейные зависимости; *в* - сигмоидальные зависимости; г - решение задачи с помощью нейронной сети с сигмоидальными зависимостями

чтобы она пересекла ось *x* дважды и разделила ее на три части. Из рис. П3.4,*б* следует, что никакие линейные зависимости или их комбинации (в одномерном пространстве это просто прямые) не позволяют решить эту задачу: любая комбинация прямых дает снова прямую, а она пересекает ось *x* только в одной точке и делит все пространство *x* только на *две* области. Иное дело - нелинейные, например ступенчатые или сигмоидальные, функции (рис. П3.4,*в*).

Для реализации требуемой зависимости необходима сеть, состоящая из пяти нейронов (рис. П3.2). Нейрон x_1 является рецептором, он воспринимает значение "признак x_1 и просто транслирует его дальше. Нейрон x_0 является "псевдонейроном" - его задача создавать постоянный сигнал +1, который, умноженный на веса c_{0i}, формирует пороговые значения для других нейронов. Нейроны x_2 и x_3 -основные рабочие нейроны сети. Их активационными функциями являются сигмоиды (рис. П3.4,*в*). Наклон характеристики - положительный для x_3 или отрицательный для x_2 - задается весовыми коэффициентами c_{13} и c_{12}, а положение на оси *x* - порогами c_{03} и c_{02} соответственно:

$x_2 = f(c_{12}x_1 + c_{02})$; $x_3 = f(c_{13}x_1 + c_{03})$.

Наконец, выходной нейрон x_4 просто суммирует эти сигмоиды - с какими-то порогами и весами:

$x_4 = c_{24}x_2 + c_{34}x_3 + c_{04}$.

В результате получается зависимость выхода сети x_4 от входа x_1, решающая задачу разделения крестиков и нуликов: она положительна для крестиков и отрицательна для нуликов (рис. П3.4,*г*).

Одна из тенденций в развитии нейронных сетей состоит в переходе к более гибким и универсальным нелинейным функциям. Суть тенденции может быть понята из следующего рассуждения [10]: *целью* обучения обычно является выделение *областей,* занимаемых различными классами. *Средством* же является проведение *границ,* поскольку пороговый элемент - линейный или нелинейный - определяет именно границу в пространстве признаков. И только на следующем уровне иерархии с помощью границ выделяются области. Такой путь неудобен, особенно если область, занимаемая классом, имеет сложную форму, является многосвязной (как в случае исключающего ИЛИ). По-видимому, будет лучше (по крайней мере в некоторых случаях), если с элементом связывается не граница, а сразу некоторая стандартная элементарная область, которая может служить базисом для построения более сложных областей. Например, можно описывать нейрон ступенчатой активационной функцией $y(x)$, положительной в некоторой области пространства признаков и отрицательной - во всех остальных областях этого пространства (рис. П3.4,*а*). Настраиваемыми параметрами при этом могут быть размеры области и ее положение в пространстве признаков. Тогда задачу о разделении крестиков и нуликов можно было решить с помощью единственного нейрона. Суммируя выходы нескольких таких нейронов, можно легко выделить область самой сложной формы. Ступенчатую функцию при этом

можно, конечно, сгладить (например, как на рис. П3.4,*г*), чтобы иметь возможность использовать градиентные методы поиска экстремума.

Из классических методов распознавания наиболее близок к этому известный метод потенциальных функций [3], [2]. В последние годы все чаще появляются нейросети, использующие именно такого рода функции (радиальные базисные функции, р-функции и т.п.). Так, сферическая радиальная базисная функция *i*-го нейрона может задаваться выражением, аналогичным выражению для нормального распределения. Комбинация элементов такого или подобного типа способна аппроксимировать любую нелинейную зависимость и, следовательно, выделить в пространстве признаков области самой сложной формы - невыпуклые, многосвязные и т.п.

В целом архитектура нейросети может быть задана матрицей весовых коэффициентов c_{ij}, характеризующих силу связей между элементами сети. В общем случае все элементы связаны со всеми, но матрица связей несимметрична, $c_{ij} \neq c_{ji}$. Некоторые коэффициенты связей могут оставаться свободными, незаданными и тогда возможно их изменение - обучение сети.

Таким образом, налагая условия на значения c_{ij}, предопределяется конфигурация сети. При этом из множества возможных конфигураций получили распространение и достаточно хорошо исследованы лишь некоторые. К числу важнейших относятся две конфигурации [11]:

1) однослойная сеть Хопфилда;

2) трехслойная сеть с промежуточным слоем "скрытых" нейронов.

Сеть Хопфилда

В 1982 г. появилась работа Дж. Хопфилда [20], которая вызвала лавину теоретических и экспериментальных исследований и оживила угасавший интерес к нейронным сетям. Неожиданный успех работы объясняется использованием простого и эффективного математического аппарата, который позволил увидеть новые грани проблемы и получить ряд новых результатов чисто теоретическим путем. Сходство этой сети с некоторыми хорошо исследованными физическими моделями (модель Изинга, спиновые стекла и пр.) позволило использовать для анализа готовый и хорошо отработанный аппарат статистической термодинамики.

Сеть Хопфилда получается, если наложить на веса связей следующие условия:

1) все элементы связаны со всеми,

2) $c_{ij} = c_{ji}$ - прямые н обратные связи симметричны,

3) $c_{ij} = 0$ - диагональные элементы матрицы связей равны нулю.

Последнее условие обычно (хотя и не всегда) добавляется, чтобы исключить непосредственную обратную связь с выхода нейрона на вход.

Одно из достоинств симметричной квадратной матрицы связей, характерной для сети Хопфилда, состоит в том, что поведение сети можно описать через стремление к минимуму простой целевой функции

$$E = -\sum_{i \neq j} c_{ij} x_i x_j = \min \qquad (10)$$

Обычно Е интерпретируется как некоторая обобщенная энергия [11]. Такая интерпретация берет начало от известной модели Изинга, в которой совокупность взаимодействующих магнитных диполей (спинов) стремится занять такую конфигурацию, в которой суммарная энергия будет минимальна. Модель Хопфилда обобщает модель Изинга в двух отношениях:

• коэффициенты связей могут принимать любые значения, как положительные, так и отрицательные;

• эти значения не заданы раз и навсегда, а меняются в процессе обучения.

Поведение системы в пространстве состояний напоминает движение шарика, который стремится скатиться в точку минимума некоторого потенциального рельефа. Характер рельефа определяется видом целевой функции Е и формируется в процессе обучения сети. Обучение производится путем демонстрации эталонных образов, которые сеть должна запоминать, хранить и потом воспроизводить (узнавать). Алгоритм обучения (формирование весовых коэффициентов c_{ij}) основывается на правиле Хебба.

Замечательное свойство такой сети (несколько напоминающее голограмму) состоит в том, что одна и та же сеть с одними и теми же весами связей может хранить и воспроизводить несколько различных эталонов. Каждый эталон является аттрактором [11], вокруг которого существует область притяжения. Любая система с несколькими аттракторами, к которым она тяготеет, может рассматриваться как содержательно-адресуемая память, т.е. память, из которой ин-

формация об эталоне извлекается путем задания нескольких признаков эталона. Если системе задается некоторое начальное состояние, отличное от эталонного, то это равносильно заданию частичной информации об эталоне. Если начальное состояние достаточно близко к эталону и попадает в область его притяжения, то система начинает двигаться к этому эталону - "вспоминает" его. Это выглядит как восстановление неверно заданных или отсутствующих признаков эталонного образа, отыскание полной информации о нем. Если одним из признаков, предъявлявшихся при обучении, является имя класса, то его восстановление будет равносильно отнесению образа к определенному классу, т.е. распознаванию.

Обратим внимание на следующий факт. Если взять одну из точек минимума энергии E в пространстве признаков X и поменять значения всех признаков на противоположные, то величина E, как видно из выражения (10), не изменится, т.е. останется минимальной. Следовательно, "негатив" эталона является таким же аттрактором, как и сам эталон, а значит, будет притягивать к себе близкие состояния, "узнаваться". Возможно, в этом лежит объяснение того психологического факта, что негатив обычно узнается человеком без всякого обучения - достаточно запомнить лишь позитив. Другими словами, образы, хранящиеся в памяти нейросети, обладают инвариантностью по отношению к позитивно-негативным преобразованиям. В естественных условиях обитания это свойство вряд ли могло принести какую-то пользу, поскольку в природе таких преобразований не бывает. Однако в искусственном мире человеческой цивилизации оно нашло себе применение: мы одинаково хорошо узнаем знаки, написанные чернилами на белой бумаге и мелом на черной доске, черные и белые контурные рисунки и т.п., поскольку важнейшие отношения между элементами образа при таком преобразовании сохраняются.

Далее, если какие-то два фрагмента эталона независимы, то один из них можно поменять на негативный и такая комбинация негатива и позитива снова будет точкой минимума E, т.е. аттрактором. В сетях с большим количеством элементов всегда много достаточно независимых фрагментов. Позитивно-негативные комбинации таких фрагментов могут порождать ложные эталоны, "призраки", которые никогда не предъявлялись при обучении, но тем не менее являются аттракторами и способны притягивать к себе близкие изображения, т.е "узнаваться". Проектировщикам нейросетей эти призраки только мешают, но для психолога они представляют определенный интерес [11]. Не эти ли призраки порождают известный феномен "ложного узнавания", когда человек переживает ситуацию как знакомую, хотя точно знает, что никогда в ней не был?

Хотя сети Хопфилда получили применение на практике (часто как составная часть более сложных систем), однако им свойственны определенные недостатки, ограничивающие возможности их применения:

• модель Хопфилда предполагает симметрию связей между элементами; без этого условия понятие энергии не может быть введено, и эта простая физическая метафора, которой модель во многом обязана своим успехом, перестает работать;

• условность понятия энергии заставляет относиться к нему с осторожностью. Это только метафора, красивая, но искажающая суть происходящих процессов. Нейронная (и ее прототип - нервная) сеть не является устройством для минимизации энергии; это устройство для запоминания и обработки информации. Экономия энергии играет в этих процессах вспомогательную роль. По мнению специалистов [10], именно информация должна занять место энергии как целевой функции сети.

Необходимость иерархии. Многослойные сети

Сеть Хопфилда поддерживает множество лишних, неэффективных связей, по существу дублирующих друг друга. В реальных нервных системах поддержание таких связей требует определенных затрат и потому невыгодно. Поэтому в ходе эволюции нервной системы происходило освобождение от части связей за счет централизации системы связей. Подобная централизация является общесистемной закономерностью и наблюдается во многих системах - биологических, технических, социальных. Например, первые телефонные сети непосредственно связывали абонентов друг с другом. Однако с ростом числа абонентов число связей N росло приблизительно пропорционально квадрату числа абонентов n:

$$N = n(n-1)/2, \qquad (11)$$

и сети быстро усложнялись. Тогда были введены центральные телефонные станции, так что каждый абонент теперь соединялся непосредственно только с телефонной станцией и уже через нее - с другими абонентами. Число связей резко уменьшилось:

$N = n,$ (12)

а с ними уменьшились и затраты на их поддержание.

Подобную эволюцию проделала и нервная система животных [10] - от диффузной у простейших к центральной нервной системе и головному мозгу у высших млекопитающих. Поэтому связь многих элементов с одним, центральным следует считать более высоким принципом организации, чем связь "всех со всеми". Множество центральных элементов образует новый уровень или слой, для которого, в свою очередь, справедлив тот же принцип организации. Так возникает многослойная иерархическая система связей. Склонность к такой организации обнаруживают и системы обработки информации, как естественные, так и искусственные. Их можно найти уже в традиционных системах распознавания образов - в виде иерархической организации системы признаков, когда из простых признаков строятся более сложные, а из них уже - фрагменты образов и далее - сами образы.

В качестве примера такого подхода рассмотрим метод группового учета аргументов, предложенный А.Г. Ивахненко [13], или его аналог - метод Φ-функций [12], основная идея которых состоит в следующем.

Любую нелинейную функцию n признаков x_i, задающую желаемое отображение входного пространства в выходное, можно аппроксимировать с помощью полинома

$$y = a_0 + \Sigma a_i x_i + \Sigma_i \Sigma_j a_{ij} x_i x_j + \ldots + \Sigma_i \Sigma_j \ldots \Sigma_n a_{ij \ldots n} x_i x_j \ldots x_n \quad . (13)$$

Коэффициенты полинома должны быть подобраны так, чтобы обеспечить желаемую зависимость y от x и распознавание с минимальной ошибкой. Для точного отыскания коэффициентов можно использовать систему нормальных уравнений Гаусса. Однако здесь мы сталкиваемся с известным "проклятием размерности": пря сколько-нибудь высокой степени полинома и при достаточном количестве признаков размеры матриц этой системы уравнений растут катастрофически. Так, при десяти признаках матрица содержит $2*10^5*2*10^5$ элементов. Этот процесс проявляет себя и при использовании адаптивных методов отыскания коэффициентов полиномов: время обучения оказывается неприемлемо большим.

В математике давно уже найден выход из этой ситуации. Он состоит в использовании различных систем стандартных функций, таких, как полиномы Чебышева, Эрмита, гармонические функции и пр. Сущность подхода - в иерархической организации сложных зависимостей. Стандартные функции подбираются так, что они, с одной стороны, сами уже обладают достаточно сложными и интересными свойствами, с другой - их можно достаточно просто комбинировать для аппроксимации еще более сложных функций. Иными словами, полином строится не из простейших - степенных функций, а из более сложных. При этом оказывается, что невозможно предложить единую систему стандартных функций, пригодную на все случаи жизни, - для каждой области приложений наиболее подходящей оказывается своя система стандартных функций.

В методе группового учета аргументов сложный полином вида (13) заменяется несколькими более простыми, учитывающими только некоторые признаки ("группы аргументов"). При этом каждый упрощенный полином рассматривается как самостоятельный независимый классификатор, коэффициенты которого определяются путем решения системы нормальных уравнений Гаусса малой размерности (т.е. точным методом). После обучения отбирается несколько наилучших (в смысле результатов классификации) полиномов, и их левые части y_j ("сложные признаки") используются в качестве аргументов для построения более сложного полинома. Практически использовались различные попарные объединения признаков, что давало возможность строить в качестве границ прямые и кривые второго порядка.

Таким образом, иерархическая организация признаков - общий путь, обеспечивающий компромисс между желаемой точностью и приемлемыми затратами на поиск или обучение. Возможность именно такой организации и предоставляет пользователю нейронная сеть с ее готовым набором стандартных нелинейных функций. Если раньше в качестве такого стандарта выступала пороговая зависимость, позволяющая проводить границы в пространстве признаков, то сейчас в качестве альтернативы применяются уже и другие стандартные наборы (потенци-

альные функции, радиальные базисные функции и пр.), оперирующие не с границами, а непосредственно с областями.

Необходимость иерархии во многом и определяет структуру большинства современных нейронных сетей. Важнейшим нововведением и главной отличительной особенностью этой структуры является наличие промежуточного слоя (или нескольких слоев) "скрытых нейронов". Скрытые элементы не являются узкоспециализированными, подобно "сенсорным" или "моторным" нейронам; они не связаны жестко ни с входными образцами, ни с выходными реакциями, эта свобода и придает нейросети необыкновенную гибкость, вычислительную мощность и способность адаптироваться к самым разным комбинациям входов и выходов. Наличие скрытых элементов позволяет нейросети выполнять действия, сходные с теми операциями по преобразованию и сокращению исходных данных, которые давно уже используются в многомерной статистике. Вот некоторые очевидные аналоги функций, выполняемых скрытыми элементами:

• "главные компоненты" в факторном анализе;

• "координаты" в многомерном шкалировании;

• "дискриминантные функции" в дискриминантном анализе.

Сеть со скрытыми элементами может реорганизовать пространство входных признаков в простые области, затем объединить их в более сложные (невыпуклые, несвязные) и, наконец, ассоциировать их с выходными категориями.

Другими факторами, определяющими архитектуру нейросети, являются условия связи с внешней средой: сеть должна иметь число входных элементов, равное размерности пространства признаков; число выходных - размерности пространства ответов. Число промежуточных (скрытых) элементов определяется сложностью задачи, требуемым объемом памяти и допустимой ошибкой распознавания.

Динамика обучения и поведения

В механике динамика в отличие от статики и кинематики предполагает наличие двух моментов: изменение переменных во времени и обусловленность этих изменений силами. Эти два момента имеют смысл применительно и к нейронным сетям. Сила здесь не просто метафора. Ей можно дать точное определение и количественное выражение. Это делает ее полезным инструментом для описания поведения системы. Но для этого сначала необходимо формально описать цель поведения системы.

При описании поведения системы одним из наиболее общих и плодотворных подходов [10] является обращение к экстремальным принципам, когда цель поведения задается в виде стремления к максимуму или минимуму некоторой целевой функции (функционала потенциала).

$$V(x) = \max. \qquad (14)$$

Часто в роли такого потенциала выступает квадратичная функция от переменных, характеризующих систему ("энергия" Хопфилда, суммарный квадрат ошибки в методе обратного распространения и т.п.). Вид целевой функции наряду с конфигурацией сети является важнейшим фактором, определяющим характер поведения системы.

Определим обобщенную силу F_x, действующую на переменную x и ответственную за ее изменения, как частную производную от целевой функции по этой переменной:

$$F_x = \partial V / \partial x \qquad (15)$$

Понятие силы удобно, потому что обычно оно определяется таким образом, что обладает свойством аддитивности. Пусть на одну переменную действует несколько факторов (например, она связана с несколькими другими переменными, как в нейросетях). Если необходимо определить результат их совместного действия, то следует охарактеризовать эти факторы через силы и затем найти равнодействующую этих сил; вследствие аддитивности она будет равна векторной сумме этих сил. В общем случае картина поведения выглядит следующим образом: задаются целевая функция и условия, наложенные на переменные. В частности, некоторые переменные могут быть фиксированными. Тогда остальные свободные переменные начинают меняться в сторону увеличения целевой функции. Процесс продолжается до тех пор, пока не будет достигнут возможный в этих условиях максимум целевой функции (условный максимум). Поскольку поведение свободной переменной всегда направлено на увеличение целевой функции, то для него справедливо следующее выражение:

$$dV/dt = \partial V/\partial x \; dx/dt = F_x \; dx/dt > 0 \qquad (16)$$

Отсюда следует, что знак обобщенной силы F_x и знак реакции dx/dt в свободном поведении всегда совпадают: либо оба положительны, либо оба отрицательны - только так можно обеспечить положительность dV/dt.

Таким образом, основное уравнение динамики системы, связывающее скорость изменения свободной переменной dx/dt с действующей на нее силой F_x, можно написать в следующем виде [11]:

$$\mathrm{sgn}(dx/dt) = \mathrm{sgn}(F_x) = \mathrm{sgn}(\partial V/\partial x) \quad (17)$$

Это уравнение является чрезвычайно общим. Все известные уравнения, описывающие процессы обучения или поведения нейросетей, являются его частными случаями и различаются между собой либо характером переменных, либо видом целевой функции.

В частном случае, когда переменные являются непрерывными и зависимость между ними линейная, уравнение (17) может быть переписано в следующем виде:

$$T dx/dt = F_x. \quad (18)$$

Здесь коэффициент пропорциональности T имеет смысл постоянной времени, характеризующей инерционность переменной. Практически в нейросетях ее задают произвольно, руководствуясь желанием обеспечить, с одной стороны, достаточно высокую скорость, с другой - устойчивость процесса. В дискретном случае задание T определяет величину шага процесса Δx, т.е. изменение переменной за один такт. Чем больше T, тем меньше шаг и тем медленнее протекает процесс. Основное уравнение динамики в этом случае можно записать в виде:

$$T \Delta x = \partial V/\partial x \quad (19)$$

В случае двоичных переменных этот шаг предопределен характером переменной: он равен либо 1 (если переменная принимает значения 0 и 1), либо 2 (если ее значения +1 и -1). В последнем случае уравнение динамики может быть записано в виде:

$$x = \mathrm{sgn}(\partial V/\partial x) \quad (20)$$

Все эти типы уравнений динамики, а также различные их комбинации встречаются сегодня в описаниях нейросетей.

В качестве примера обратимся еще раз к сети Хопфилда [10]. Целевая функция здесь может быть записана в виде

$$V = -E = -\Sigma_{ij} c_{ij} x_i x_j = \max$$

Переменные x_i обычно являются двоичными, весовые коэффициенты c_{ij} могут рассматриваться как дискретные или непрерывные переменные. Предположим, что значения всех переменных x_i фиксированы, свободными переменными являются только веса межнейронных связей c_{ij}. Изменение этих переменных означает обучение сети. Основное уравнение динамики в этом случае выглядит так:

$$T_c \, dc_{ij}/dt = F_c = \partial V/\partial c_{ij} = x_i x_j \quad (21)$$

Если c_{ij} - дискретные переменные, то (21) принимает вид:

$$T_c \Delta c_{ij} = F_c = \partial V/\partial c_{ij} = x_i x_j \quad (22)$$

Это есть не что иное, как правило Хебба для формирования весовых коэффициентов в процессе обучения (с точностью до константы T_c).

Предположим теперь, что фиксированы значения весовых коэффициентов c_{ij} (сеть обучена) и некоторые из признаков x_j (задана частичная информация об эталоне). Для оставшихся свободными признаков основное уравнение динамики принимает вид:

$$T_x \, dx_j/dt = F_j = \partial V/\partial x_j = \Sigma_i c_{ij} x_i \quad (23)$$

Здесь $F_j = \Sigma_i c_{ij} x_i$ и есть равнодействующая сил вида: $F_i = c_{ij}x_i$, действующих на переменную x_j со стороны переменных x_i через связи c_{ij}.

Если x_I - двоичные переменные, принимающие значения +1, -1, то (23) превращается для них в выражение:

$$x_j = \mathrm{sgn} \Sigma_i c_{ij} x_i \quad (24)$$

В таком виде это уравнение динамики первоначально и было написано Хопфилдом.

Обучение многослойных сетей

Преимущества многослойных сетей были поняты достаточно рано. Ясно было также, что для использования этих преимуществ преобразование при переходе от одного слоя к другому должно быть нелинейным: последовательность линейных преобразований дает снова линейное преобразование со всеми его недостатками. Однако развитию многослойных сетей препятство-

вало то, что не было теоретически обоснованного алгоритма обучения таких сетей. Неясно было, по какому правилу следует модифицировать связи нейронов промежуточных слоев, чтобы получить на выходе нужный результат.

Такой алгоритм был предложен в ряде работ [22, 5, 4, 21]. Сейчас он известен как метод "обратного распространения ошибки". Покажем, что и этот метод является частным случаем основного уравнения динамики.

Сущность его в следующем. Если известна целевая функция системы $V(x)$, то можно найти силы, действующие на любые переменные системы и вызывающие их изменения в сторону максимизации целевой функции. Если эти переменные непосредственно входят в целевую функцию, то эти силы вычисляются по формуле (15). Таковы, например, выходные переменные системы, когда целевая функция непосредственно зависит только от них.

Если же интересующие нас переменные x_i непосредственно не входят в целевую функцию, но связаны с переменными x_j, входящими в нее, то действующие на x_i силы могут быть вычислены с помощью "закона передачи силы" - обобщения известного закона рычага на немеханические системы. Закон, как известно, гласит: что выигрывается в силе (F), то проигрывается в расстоянии (Δx). Это можно записать так:

$$F_i\ \Delta x_i = F_j\ \Delta x_j \quad \text{или} \quad F_i / F_j = \Delta x_i / \Delta x_j = k_{ij}. \qquad (25)$$

Здесь $\Delta x_i, \Delta x_j$ - изменения двух связанных переменных; F_i, F_j - действующие на них силы; k_{ij} - передаточный коэффициент от x_i к x_j.

В нейронных сетях к числу переменных, не входящих непосредственно в целевую функцию, относятся параметры нейронов промежуточных ("скрытых") слоев. Однако переменные i-го уровня связаны с переменными следующего j-го уровня. Передаточный коэффициент k_{ij} определяется прежде всего весом синаптической связи c_{ij}. Кроме того, если связь нелинейная, то в него должна входить сомножителем производная от активационной (передаточной) функции, зависящая от значения выходной переменной. В случае сигмои-дальной функции эта производная равна выражению (7). Согласно закону передачи силы справедливо следующее соотношение между силами, действующими на выходные величины y_i и y_j двух соседних слоев :

$$F_i = F_j\ k_{ij}. \qquad (26)$$

Подставляя сюда выражения для силы и для передаточного коэффициента и учитывая, что переменная одного слоя связана с несколькими переменными другого и что действующая на нее сила должна быть равнодействующей суммы сил, получаем следующее рекуррентное соотношение, позволяющее находить производную n - 1-го слоя по производной n-го слоя :

$$\partial V / \partial y_i^{(n-1)} = \Sigma_j\ \partial V / \partial y_j^{(n)}\ y_j^{(n)}\ (1 - y_j^{(n)})\ c_{ij} \qquad (27)$$

Зная силу, действующую на переменную, и используя основное уравнение динамики (17), можно написать закон изменения этой переменной. Если переменная - один из весовых коэффициентов c_{ij}, *то* это и будет закон обучения.

В многослойных сетях, как правило, воздействие распространяется только в одном направлении - от i-го слоя к j-му. Следовательно, матрица связей несимметрична: только веса c_{ij} могут быть отличны от нуля, тогда как c_{ji} заданы равными нулю.

Рассмотрим в качестве примера один из известных алгоритмов обучения - "δ-правило". В простейшем виде оно использовалось уже при обучении персептрона. Суть его в следующем. Пусть в качестве желаемого ("идеального") значения выходной величины задано значение y^*_j . Действительное значение y_j получается путем преобразования входных значений предыдущего слоя x и не обязательно совпадает с желаемым:

$$y_j = f(s_j) = f(\Sigma_i\ c_{ij}\ x_i) \qquad (28)$$

Здесь $f(s_j)$ - активационная функция; $s_j = \Sigma_i\ c_{ij}\ x_i$

Цель поведения состоит в минимизации квадрата ошибки:

$$V = -E = -\Sigma_j (y_j - y^*_j)^2/2 = -\Sigma_j [y(\Sigma_i\ c_{ij}\ x_i) - y^*_j]^2/2 = \max \qquad (29)$$

Тогда сила, действующая на переменную c_{ij}, равна:

$$F_c = -\partial E / \partial c_{ij} = -\partial E / \partial y_j\ dy_j / ds_j\ \partial s_j / \partial c_{ij} = \delta_j\ df/ds\ x_i \qquad (30)$$

Здесь через $\delta_j = (y^*_j - y_j)$ обозначена разность между идеальным (желаемым) и действительным значениями выходной величины. Отсюда и название алгоритма - δ-правило. Скорость модификации веса определяется основным уравнением динамики в форме (18).

$$T_c\ dc_{ij}/dt = F_c = \delta_j\ dy_j / ds_j\ x_i \qquad (31)$$

Это и есть δ-правило. Если $f(s_j)$ - сигмоидальная функция вида (5), то получается один из вариантов алгоритма обратного распространения ошибки.

$$T_c\, dc_{ij}/dt = \delta_j y_j (1 - y_j)\, x_i = (y_j - y_j^*)\, y_j (1 - y_j)\, x_i. \qquad (32)$$

Согласно этому алгоритму скорость модификации весового коэффициента c_{ij} пропорциональна трем факторам :

• "ошибке" - разности между действительным и желаемым значениями выходной величины $(y_j - y_j^*)$;

• производной от функции активации $y_j (1 - y_j)$;

• входной величине x_i.

Если зависимость от первого фактора является полезной, то два последних фактора служат источником различного рода неприятностей, возникающих в процессе обучения.

В настоящее время обычно используют различные модификации этого алгоритма, имеющие целью улучшить устойчивость процесса обучения и разрешить ряд других проблем. Например, вводят в алгоритм "память" о предыдущем шаге, что придает ему определенную инерционность и устойчивость к помехам.

Проблемы и перспективы

Остановимся на трудностях, связанных с обучением нелинейных нейронных сетей. Основные из них следующие [9].

Медленная сходимость процесса обучения. Строго сходимость доказана для дифференциальных уравнений, т.е. для бесконечно малых шагов в пространстве весов. Но бесконечно малые шаги означают бесконечно большое время обучения. При конечных шагах сходимость не гарантируется, но даже если она имеет место, то потребное для этого время может быть слишком большим, сравнимым с временем жизни пользователя.

"Ловушки", создаваемые локальными минимумами. Детерминированный алгоритм обучения не в силах обнаружить глобальный минимум или покинуть локальный минимум. Одним из приемов, позволяющих обходить ловушки, является расширение размерности пространства весов за счет увеличения числа нейронов второго слоя. Некоторые новые возможности открывают стохастические методы. Но все это достигается ценой дополнительных затрат времени обучения.

"Паралич" сети. Сигмоидальный характер передаточной функции нейрона приводит к тому, что если в процессе обучения несколько весов стали слишком большими, то нейрон попадает на горизонтальный участок функции в область насыщения. При этом изменения других весов, даже достаточно большие, практически не сказываются на величине выхода нейрона, а значит, и на величине целевой функции. Из выражения для производной от передаточной функции (7) видно, что она стремится к нулю, когда y приближается к нулю или единице. Это значит, что связь между соседними слоями практически разрывается, и процесс обучения блокируется.

Неудачный выбор диапазона входных переменных - достаточно элементарная, но часто совершаемая ошибка. Если x_i - двоичная переменная со значениями 0 и 1, то примерно в половине случаев она будет иметь нулевое значение: $x_i = 0$. Поскольку x входит сомножителем в выражение для модификации веса (32), то эффект будет тот же, что при насыщении: модификация соответствующих весов прекратится, и обучение будет блокировано. Правильный диапазон для входных переменных должен быть симметричным, например от + 1 до -1.

"Перетренировка". Следует иметь в виду, что излишне высокая точность, полученная на обучающей выборке, может обернуться неустойчивостью результатов на тестовой выборке. Здесь действует общий закон: чем лучше система адаптирована к данным конкретным условиям, тем меньше она способна к обобщению и экстраполяции, тем скорее она может оказаться неработоспособной при изменении этих условий. А такие изменения от выборки к выборке неизбежны, особенно если выборки имеют небольшие размеры. Расширение объема обучающей выборки позволяет добиться большей устойчивости, но за счет увеличения времени обучения.

Проблема объема памяти. Емкость памяти нейросети, ее способность хранить и воспроизродить информацию являются одной из важнейших характеристик нейросети. Однако если в традиционных последовательных машинах характеристики памяти достаточно понятны и до-

ступны оценке, то в нейросетях дело обстоит намного сложнее. До сих пор нет единого подхода даже к определению емкости памяти [14, 18].

Стохастические методы обучения. Детерминистский метод обучения производит модификацию весов сети только на основе информации о направлении градиента целевой функции в пространстве весов. Такой метод способен привести к локальному экстремуму, но не способен вывести из него, поскольку в точке экстремума сила обращается в нуль и причина движения исчезает (как это видно из уравнения динамики (17)). Чтобы заставить сеть покинуть локальный экстремум и отправиться на поиски глобального, нужно создать дополнительную силу, которая зависела бы не от градиента целевой функции, а от каких-то других факторов. Выбор этих факторов, более или менее оправданный различными эвристическими соображениями, и составляет основу различных методов преодоления локальных ловушек. Один из простейших методов состоит в том, чтобы просто создать случайную силу и добавить ее к детерминистической. Само присутствие такого рода случайных факторов: "шума", "температуры" приводит к *"усреднению,* сглаживанию, размыванию" потенциальных барьеров. Мелкие гребни и впадины исчезают, и если в пространстве параметров есть глобальный экстремум, то выявляется сила, действующая в направлении этого экстремума. Правда, сила эта имеет случайный характер: она только в среднем направлена в сторону этого экстремума. По мере приближения к нему эта средняя регулярная составляющая уменьшается, приближаясь к нулю, и остается только случайная. Даже достигнув глобального экстремума, система будет продолжать колебаться около него с достаточно большой амплитудой. Поэтому обычно поступают таким образом: по мере приближения к экстремуму амплитуду случайной составляющей постепенно снижают. Такая процедура напоминает отжиг металла, когда для достижения оптимальной энергетической структуры металла его сначала нагревают, а потом медленно и постепенно охлаждают. Этот метод получил название "метод имитации отжига".

Применение нейросетевой технологии

Новые идеи в области нейросетей довольно быстро нашли практическое применение и вызвали к жизни новый тип микроэлектронной техники - нейропроцессоры и нейрокомпьютеры (НК). В 1986 г. калифорнийский биофизик Т. Сеймовский уже смог создать техническое устройство подобного типа. Система начинала работать "в полном невежестве", но если ей указывали на ошибки, она их больше не повторяла. Эти идеи были подхвачены американской компанией TRW, ее исследовательским центром во главе с Р.Хехт-Нильсеном.

Во многих странах создаются новые исследовательские центры, тратятся десятки миллионов долларов. В разработку НК включились в Америке - IBM, ATT, "Texas Instuments"; в Японии - "Ниппон Электрик", "Фудзизу"; в Западной Европе - "Бюль", "Томсон", "Рон-Пуленк" и множество других известных компаний. Возникли новые фирмы, специализирующиеся на нейросетевой технологии.

Современные нейросети включают возможности: читать цифры и слова; узнавать лицо человека по небольшому фрагменту фотографии; по обрывкам сведений восстанавливать всю информацию, относящуюся к делу; вести разведку на поле боя; обнаруживать малозаметные летательные аппараты; распознавать цели; вести общее руководство боевыми действиями. В области технологии они позволяют управлять технологическими процессами, перенимая опыт квалифицированных операторов, обнаруживать неисправности в сложных системах, предвидеть и предотвращать возможные ошибки и аварии. Вот ряд конкретных примеров.

Фирма NEC (Япония) объявила, что ею было создано устройство для визуального распознавания букв. Точность распознавания превысила 99%. Успех был достигнут за счет интеграции обычных алгоритмов с нейросетью, работающей по методу обратного распространения ошибки.

В университете Дж.Гопкинса (США) создана нейронная сеть "Net-Talk", предназначенная для чтения вслух печатного текста (300 нейронов, 10 000 связей, слова создаются синтезаторами). За восемь дней сеть освоила 20 000 английских слов. По свидетельству очевидцев, звучание текста очень напоминает голос ребенка на различных этапах обучения речи.

Фирма "Белл" реализовала нейросеть в виде микросхемы (54 простейших процессора, 114 400 нейронов, образованных в светочувствительной пленке из аморфного кремния на стеклянной подложке). После тренировки может распознавать изображение по его части.

Во всех приложениях отмечается высокая надежность нейросетей: сеть продолжает работать, даже если 15% ее элементов вышли из строя.

Вот еще ряд быстроразвивающихся направлений применения нейросетевой технологии.

Чтение печатного текста (фирмы Sharp Corp., Mitsubishi Electric Corp., VeriFon Inc., Hecht-Nielsen Corp., Nestor Inc. и др.). Оптическая система распознавания (Optical Character Recognition) фирмы Sharp используется для распознавания японских иероглифов; содержит порядка 10 млн связей и использует разновидность алгоритма LVQ Кохонена; превосходит существующие системы по скорости и точности. Система Onyx Check Reader фирмы VeriFon обеспечивает точное и недорогое считывание чисел на чеках, используя стандартный аналоговый нейрочип фирмы Synaptics. Фирма Calera Recognition Systems продает систему FaxGrabber, которая автоматически превращает поступающий факс в текст, используя в качестве алгоритма модификацию радиальной базисной функции. Фирма Audre Recognition Systems использует вариант алгоритма обратного распространения в устройстве Audre Neural Network, который не только читает стандартный буквенно-цифровой текст, но может быть обучен распознаванию специальных символов, используемых в технических чертежах.

Распознавание ручного печатного шрифта. Система Quickstrokes Automated Data Entry System (Hecht-Nielsen Corp., США) была использована для обработки чеков. Компания Wyoming до этого теряла примерно 300 000 дол. в год из-за задержек на этой операции. Фирма Poqet Computer использует сеть NestorWriter для распознавания рукописных символов на персональных компьютерах с перьевым вводом.

Контроль качества на производстве: анализ спектроскопических данных в химической промышленности [6, 7], классификация дефектов громкоговорителей (CTS Electronics) [1], оценка чистоты апельсинового сока (Florida Departamen of Citrus).

Идентификация событий в ускорителях частиц (CERN и ряд других исследовательских организаций). Быстрая аналоговая нейросеть используется в реальном времени для включения детекторов частиц. Это позволяет отобрать из огромного числа событий приемлемое множество интересных событий, заслуживающих дальнейшего изучения. Аналогичная работа проводится в Fermi National Accelerator Laboratory (США) с использованием разработанного в фирме Intel высокоскоростного аналогового нейрочипа ETANN.

Разведка нефти. Нефтяные компании Arco, Texaco и другие используют нейросети для поиска месторождений нефти и газа.

Борьба с наркотиками. Система на базе ПЭВМ, эмулирующая нейросеть, в Nort Carolina State Bureau of Investigation (США) помогает идентифицировать образцы кокаина, имеющие одинаковое происхождение. Это позволяет выявить группы связанных друг с другом распространителей наркотиков.

Медицинские приложения. Фирма Neuromedical Systems Inc. предлагает электроэнцефалографы, аппаратуру для скриннинга рака и другое оборудование, основанное на нейросетевой технологии. Система Papnet способна помочь цитологу обнаружить раковые клетки; используется в US Food and Drug Administration (США).

Финансовый анализ и прогнозирование. Нейросети используются для этих целей многими инвестиционными фирмами (Merrill Lynch & Co., Salomon Brothers, Shearson Lehman Brothers Inc., Citibank, World Bank). Фирма Promised Land Technologies предлагает недорогой пакет, обещающий существенное улучшение эффективности инвестиций. Chase Manhatten Bank использует гибридную систему распознавания образов с нейросетью для оценки риска при выдаче займов. Фирма Foster Ousley Conley использует систему, разработанную в компании Hecht-Nielsen Corp., для оценки стоимости собственности в Калифорнии. Система Target Marketing System используется компанией Veratex Corp. (США) для оптимизации рыночной стратегии. Фирма Spiegel Inc использует программы, созданные компанией Neural-Ware Inc., для определения потенциальных покупателей; ожидается экономия по крайней мере 1 млн дол. в год за счет увеличения продаж и сокращения затрат на бесперспективных покупателей.

Управление и оптимизация. Интеллектуальный контроллер на основе нейросетевой технологии для управления дуговой печью, установленный фирмой Neural Application Corp., позволяет сберечь миллионы долларов в год на один агрегат. Фирма Copin Corp. использует нейросеть в производстве солнечных элементов. Фирма Pavilion Technologies разработала нейросеть, которая используется в ряде компаний для управления качеством продукции. Техас-

ская фирма Puget Sound Refinery включила нейросети в систему управления очисткой нефти. Одна из таких сетей используется в управлении дебутанайзером - системой, которая разделяет углеводороды по их молекулярным весам. Это требует точного управления температурами, давлениями и скоростями потоков. Семнадцатичасовой цикл подвержен постоянной нестабильности. Нейросеть из семи входных и двух выходных нейронов была обучена на 1500 примерах и способна предупреждать ошибки до того, как они появляются, обеспечивая высокое качество продукта в периоды нестабильности. Фирма Nippon Steel Corp. (США) использует нейросеть для предотвращения нарушений в процессе выплавки стали. Система обучалась методом обратного распространения ошибки и успешно работает с 1990 г. В химической и пищевой промышленности нейросеть CAD/Chem фирмы AI Ware (США) используется для оптимизации рецептуры производимых продуктов.

Военные приложения. Фирма US Naval Air Warfare Center (США) использует нейросети для управления снарядами и других военных приложений. Установлено, что там, где требуются быстрые решения, нейросети имеют огромные преимущества перед обычными методами. Фирма Lockheed (США) разрабатывает систему управления воздушным боем для истребителя, основанную на прогнозировании возможных действий противника. Система использует нейросеть для интеграции многоканальных данных об образцах полета и воздушного боя.

ЛИТЕРАТУРА

1. *Абу-Мостафа Я., Псалтис Д.* **Оптические нейронно-сетевые компьютеры** //В мире науки.-1987, № 5. - С. 42-50.

2. *Айзерман М.А., Браверман Э.М., Розоноэр Л.И.* **Метод потенциальных функций в теории обучения машин.** - М: Наука, 1970. - 383с.

3. *Айзерман М.А., Браверман Э.М., Розоноэр Л.И.* **Теоретические основы метода потенциальных функций в задаче об обучении автоматов разделению входных ситуаций на классы** // Автоматика и телемеханика. - 1964. - № 6.

4. *Барцев С.И., Гилев С.Е., Охонин В.А.* **Принцип двойственности в организации адаптивных сетей обработки информацик**//Динамика хим. и биол. систем. - 1955. - №6.

5. *Барцев С.И., Охонин В.А.* **Адаптивные сети обработки информации.** - Красноярск: Ин-т физики СО АН СССР, 1986, Препринт 59Б. - 20с.

6. *Бонгард М.М.* **Проблема узнавания.** - М.: Наука, 1967. - 320с.

7. *Вапник В.Н. Червоненкис А.Н.* **Об одном классе персептронов** //Изв. АН СССР, Тех. кибернетика. - 1964. - № 1.

8. *Голицын Г. А.* **Применение нейросетевой технологии в ЭС**// Материалы семинара "Экспертные системы реального времени". - М., РДЗ, 1995.

9. *Голицын Г.А., Фоминых И.Б.* **Интеграция нейросетевой технологии с экспертными системами** // Труды 5 Национальной конференции по ИИ. - Казань, 1996.

10. *Голицын Г.А., Петров В.М.* **Гармония и алгебра живого.** - М.: Знание, 1990. - 128с.

11. *Голицын Г.А., Петров В.М.* **Информация - поведение - творчество.** - М.: Наука, 1991. - 224с.

12. *Дуда Р., Хорт П.* **Распознавание образов и анализ сцен.** - М.: Мир, 1976. -512с.

13. *Ивахненко А.Г.* **Персептроны.** - Киев: Наукова думка, 1974.

14. *Лоскутов А.Ю., Михайлов А.С.* **Введение в синергетику.** - М.: Наука, 1990, - 272с.

15. *Маккаллок У.С., Питтс У.* **Логическое исчисление идей, относящихся к нервной деятельности.** - М.**:** Иностр. лит., 1956.

16. *Минский М.Л., Пейперт С.* **Персептроны.** - М.: Мир, 1971.

17. *Розенблатт Ф.* **Принципы нейродинамики.** - М.: Мир, 1965. - 480 с.

18. *Уоссермен Ф.* **Нейрокомпьютерная техника.** - М.**:** Мир, 1992.

19. *Hebb D.O.* **The Organization of Behavior.** - NY.: Wiley, 1949.

ПРИЛОЖЕНИЕ 4

Системы поддержки принятия решений, хранилища данных и извлечение знаний

При возникновении баз данных (БД) считалось, что они откроют самые широкие возможности для построения систем поддержки принятия решений (СППР). Однако по ряду объективных и субъективных причин до настоящего времени этот аванс не был реализован. Ограниченные возможности компьютеров приводили к тому, что они в первую очередь ориентировались на решение операционных (рутинных) задач, таких, как движение товаров на складе, расчеты с поставщиками, работа с кадрами и т.п., а не на решение задач принятия решений, требующих хранения огромных объемов информации и нетривиальных алгоритмов ее обработки.

В начале 90-х годов тенденция начала меняться. Стало очевидно, что, помимо транзакционной обработки данных, организациям необходима аналитическая обработка накопленных данных. По оценкам International Data Corp., рынок СППР в 1994 г. составлял 339 млн. дол., в то время как в 1997 г. этот рынок будет составлять более 1 млрд. дол.

В США до активного распространения архитектуры клиент-сервер выделяли два типа СППР: информационные системы для руководства (Executive Information System - EIS) и системы поддержки решений (Decision Support System - DSS). EIS, как правило, выполнялись на больших ЭВМ и предназначались для руководства верхнего уровня. DSS выполнялись на рабочих станциях и предназначались для руководства среднего звена. Однако в связи с тем, что в последние годы передовые компании активно проводят реинжиниринг (см. Приложение 2), что влечет за собой делегирование полномочий по принятию решений среднему и нижнему звену в управленческой иерархии, различия между EIS и DSS стираются.

СППР обычно используются при решении следующих задач:

• определение и анализ тенденций;

• измерение ключевых соотношений и слежение за ними;

• детализирующий анализ (drill down analysis);

• анализ "что-если" ("what if");

• анализ конкурентоспособности;

• мониторинг задач (problem monitoring).

В общем виде можно сказать, что качество СППР зависит от данных, на основании которых принимаются решения; используемых аналитических методов и моделей обработки и анализа данных; адекватности используемых инструментальных средств (ИС) задачам принятия решений. Данные, используемые для принятия решений в СППР, можно классифицировать по степени полезности для лица, принимающего решение (ЛПР) в порядке увеличения полезности, следующим образом:

• примитивные, операционные данные без временной привязки (см. ниже);

• производные данные (данные, соотнесенные с информационной потребностью пользователя, т.е. предметно-ориентированные данные) с историей их изменения во времени (см. ниже);

• события, т.е. не отдельные данные (информация), а их привязанная ко времени совокупность, обычно выражаемая ЛПР некоторым понятием, обобщающим всю совокупность данных (например, "выпуск нового вида продукции в I кв.", "проведение активной рекламной кампании по некоторому виду продукции в течение I и II кв." и т.п.). Следует подчеркнуть, что в этом случае уместно говорить об иерархии данных (данные, ситуации 1-го уровня,..., ситуации i-го уровня);

• данные (информация) и события о деятельности компании на различных уровнях, генерируемые с помощью моделей и средств моделирования.

Подчеркнем, что в первых трех случаях данные о деятельности компании появляются в результате их сбора в ходе реальной деятельности компании В последнем случае, т.е. при наличии моделей различного уровня, осуществляется генерация данных в ходе гипотетических рассуждений о деятельности компании. Ниже приведены результаты сравнения операционных и аналитических данных.

Примитивные операционные данные	Производные данные, данные для принятия решений
Детализированы	Обобщены либо очищены
Точны в момент доступа	Представляют значения на указанное время
Обслуживают сообщество клерков	Обслуживают сообщество работников управления
Могут корректироваться	Не корректируются
Обрабатываются многократно	Обрабатываются эвристически
Требования к способам обработки выясняются в первую очередь	Требования к способам обработки не имеют, первостепенного значения
Строятся на основе обычного цикла разработки систем	Совершенно иной жизненный цикл
Чувствительны к производительности	Мягкие требования к производительности
Обрабатывается один элемент данных за один запрос	Обрабатывается множество элементов данных за один запрос
Управляются транзакциями	Управляются аналитическими запросами
Ориентированы на приложения	Ориентированы на анализ
Управление обновлением - ключевой момент	Управление обновлением не используется
Высокая степень доступности	Относительная доступность
Контролируется целостность всех данных	Контролируется целостность подмножества данных
Неизбыточны	Избыточны
Статическая структура, произвольное содержание	Гибкая структура
Массивы данных мало используются в процессе обработки	Массивы данных широко используются в процессе обработки
Поддерживают ежедневные операции	Поддерживают нужды управления
Высокая вероятность возникновения запроса	Низкая, умеренная вероятность возникновения запроса

Методы и модели, используемые в СППР для обработки и анализа данных, можно классифицировать по степени полезности для ЛПР (в порядке увеличения полезности) следующим образом:

• используются аналитические методы обработки и анализа;

• объединение аналитических методов и моделей обработки и анализа.

• Уточним, что мы в данном контексте вкладываем в понятие "метод" и "модель". Методы позволяют на основании исторических данных о реальной деятельности компании выводить (вычислять) общую или детализированную информацию. Например:

• по данным о ежемесячных выпусках продукции делать вывод о том, что за последний квартал текущего года ежемесячный рост составил 10%;

• по данным о снижении добычи газа в стране за декабрь текущего года сделать вывод о том, что основной причиной этого является резкий спад добычи газа в некотором регионе.

Модели в отличие от методов с помощью имитационного моделирования и соответствующих ИС позволяют моделировать гипотетические события и их последствия.

Традиционные (операционные) базы данных оказались не пригодны для решения задач аналитической обработки по следующим причинам:

• существует физическое различие между объектами, на которые направлена операционная активность организации, и объектами, необходимыми для анализа, планирования и принятия решений ;

• существующие технологии обеспечения операционной обработки фундаментально отличаются от технологий поддержки принятия решений;

• характер запросов пользователей, обслуживающих операционные системы, совершенно иной, чем характер запросов работников управления.

По этим и многим другим причинам широкое распространение получили *программы извлечения данных* (см. ниже), позволяющие перекачивать различные выборки данных из опера-

ционных баз данных в дополнительные. Существуют по крайней мере две причины, по которым работа с дополнительными базами данных более подходит работникам управления, чем работа с операционными базами данных:

• работа с данными в дополнительных базах данных не сказывается на производительности основных операционных баз данных;

• работники управления могут полностью управлять данными в дополнительных базах данных в режиме "Что, если...".

Этими характеристиками не обладают, да и не могут обладать операционные базы данных. Есть ряд недостатков, которые вовсе не позволяют использовать эти базы данных в процессе принятия решений: недостоверность данных; низкая производительность при нестандартных запросах; невозможность преобразования разнородных данных в единую информацию.

Недостоверность данных можно проиллюстрировать следующим примером. Два подразделения одной организации готовят отчет о текущей прибыли. Первое докладывает, что прибыль выросла на 10%, второе - что прибыль упала на 15%. Работник управления, получив два отчета, не знает, что и думать. Существует несколько причин такой нестыковки:

• данные не имеют меток времени;

• алгоритмы подготовки отчетов различны;

• существует несколько этапов извлечения данных;

• отчеты строятся на основе не только внутренних, но и внешних данных;

• одни и те же отчеты строятся на основе разных источников данных.

Отчет, сделанный в понедельник, как правило, не совпадает с отчетом, выполненным в четверг. Данные организации постоянно меняются. Лишь установка меток времени может обеспечить в четверг составление отчета по состоянию на понедельник. Различные специалисты по управлению используют отличающиеся методики построения отчетов. В одном подразделении анализируются все счета, в другом - только самые крупные. Это влечет за собой различные алгоритмы и, разумеется, различные отчеты.

Как правило, данные для отчетов собираются из множества источников данных. При этом необходимо пройти несколько этапов по извлечению данных. Прохождение этого процесса может занять длительное время. В результате может произойти разбалансировка данных как по времени, так и из-за отличий в алгоритмах извлечения данных. Дополнительную путаницу вносят внешние источники данных. Обычно подразделения ведут две или более баз данных, причем часть информации в них относится к одним и тем же характеристикам одного и того же объекта. При построении отчета один аналитик предпочитает пользоваться данными из базы данных А, другой - из Б. Никакой согласованности между А и Б нет.

Недостоверность данных не единственный недостаток традиционных систем. Многие специалисты по управлению столкнулись с *проблемой затрачиваемого времени* на подготовку даже недостоверного отчета. При создании отчета можно выделить три проблемы:

• найти, где находятся данные, необходимые для отчета;

• обработать и проанализировать данные для отчета;

• привлечь программиста и аналитика для выполнения вышеперечисленных работ.

В подразделении обычно существует множество наборов данных. Один может содержать элемент с именем BALANCE, другой - CUR-BAL, третий - INVLEVEL. В то же время все эти имена указывают на одно и то же. Это существенно затрудняет обнаружение данных для отчета. Таким образом, чтобы обработать эти данные, необходимы программы извлечения данных для каждого источника данных; каждая из программ должна согласовывать форматы данных с другими программами под управлением пользователя; программы должны функционировать на имеющихся программно-аппаратных платформах. Подготовка такой технологической цепочки под каждый запрос может привести к тому, что требуемый отчет может появиться через месяцы, а то и годы.

Тем не менее проблема производительности не последняя в списке проблем, связанных с традиционной архитектурой информационных систем. Существует сложность с формированием информации на основе существующих данных. Как правило, данные группируются относительно тех приложений, которыми они используются. Существуют рабочие места кассира, бух-

галтера, начальника отдела кадров и т.п. Каждое рабочее место может поддерживать свою базу данных.

Необходимо *из разрозненных данных* собрать *осмысленную информацию.* Это довольно сложно, поскольку приложения не были разработаны с целью взаимной интеграции. Кроме того, может оказаться, что базы данных не содержат предыстории, т. ё. на одном рабочем месте присутствует информация за последние два года, на другом – за один, на третьем - только за текущий месяц. Невозможно проследить тенденцию развития организации за какой-то период, если за этот период соответствующие данные отсутствуют.

Подводя итоги рассмотрения проблем, возникающих с традиционным подходом к автоматизации информационной деятельности организации, можно отметить, что, несмотря на обилие данных, возможностей их сбора и хранения, организации до сих пор испытывают существенный недостаток в информации, необходимой для стратегического и оперативного управления своей деятельностью. Существующие системы сбора и обработки корпоративных данных в принципе не пригодны для использования в процессе принятия управленческих решений. Данные разрознены, разнотипны и распределены как внутри организации, так и за ее пределами. Работникам управления Приходится принимать решения не только в условиях неполной, но и зачастую недостоверной и противоречивой информации. К тому же не всегда удается получить требуемую информацию вовремя и в наглядном виде. В результате - неудачные решения, в некоторых случаях - даже крах организации. Ниже приведена характеристика данных по уровням.

Уровни данных	**Характеристика данных**
Операционный уровень	Данные детализированы. Приложения нацелены на обработку ежедневных операций. Хранятся только текущие значения. Высокая вероятность возникновения запросов. Данные ориентированы на использующие их приложения
Корпоративный уровень	Данные обобщены. Все значения имеют метки времени. Данные интегрированы и предметно ориентированы
Уровень подразделения	Собраны данные, имеющие отношение к данному подразделению. Частично данные относятся к примитивным, частично - к производным
Индивидуальный уровень	Данные временны. Запросы нестандартные. Сбор данных происходит эвристически. Операции нерутинные

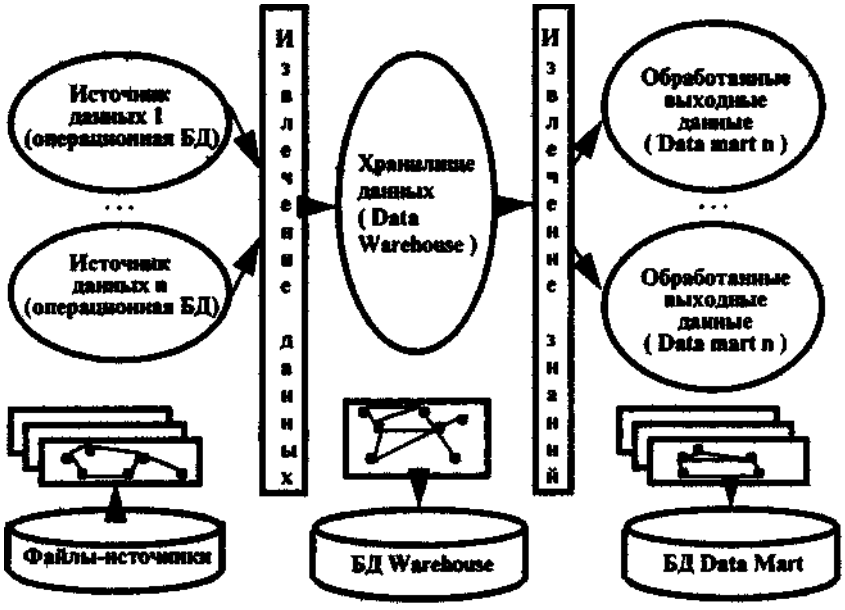


Рис.П4.1. Архитектура хранилища данных (Data Warehouse)

Чтобы вырваться из такого драматического положения, организациям предлагается воспользоваться современной концепцией создания "единого источника правды" для руководящего персонала. Такой источник был назван *"хранилище данных" (Data Warehouse)* [4]. По определению, Data Warehouse - это *предметно-ориентированная, интегрированная, некорректируемая, зависимая от времени* коллекция данных, предназначенная *для поддержки принятия управленческих решений* (рис. П4.1). Хранилище данных должно предложить такую среду

накопления данных, которая оптимизирована для выполнения сложных аналитических запросов управленческого персонала. Эти запросы могут быть достаточно индивидуальны для каждой организации, каждого подразделения и даже отдельного руководителя.

Хранилище данных должно автоматически собирать операционные данные, согласовывать их и объединять в предметно-ориентированный формат, который нужен работникам управления. Данные в хранилище данных не предназначены для модификации.

Предметная ориентация означает, что данные объединены в категории и хранятся в соответствии с теми областями, которые они описывают, а не с приложениями, в которых они используются. Например, информация об оптовых заказчиках хранится в одной базе данных в окружении хранилища данных и может собираться по множеству файлов и баз данных операционного окружения.

Интегрированность определяет данные сразу таким образом, чтобы они удовлетворяли требованиям всего предприятия (в его развитии), а не единственной функции бизнеса. Тем самым хранилище данных гарантирует, что одинаковые отчеты, сгенерированные для разных аналитиков, будут содержать одинаковые результаты.

Некорректируемость заключается в том, что данные в хранилище данных не создаются (они поступают от операционных или внешних источников), не корректируются и не удаляются. *Зависимость от времени* подразумевает, что хранилище данных предназначено для анализа данных во времени. Важно знать не только значения данных, но и время их появления. Кроме того, данные в хранилище данных должны быть согласованы во времени. Нельзя допустить, чтобы данные из различных источников считывались по состоянию на разные моменты времени. *Направленность на принятие управленческих решений* гарантирует правильное использование хранилища данных для анализа и поддержки принятия решений, а не для обработки транзакций.

При реализации хранилища данных особое значение приобретают следующие процессы работы с данными: извлечение; преобразование; анализ; представление. Операция извлечения данных перемещает информацию из источников данных (см.рис. П4.1) в отдельную базу, специально созданную для хранилища данных. При извлечении данные приводятся к единому формату и сочетаются так, как нужно организации.

Основными источниками данных для хранилища данных, как правило, являются эксплуатируемые уже многие годы системы регистрации операций. Это могут быть системы учета движения товаров на складе, учета наличности в кассе, начисления заработной платы, регистрации клиентов, сделок, партнеров и т.д. К источникам данных можно отнести также отдельные документы и наборы данных, предоставляемые специализированными компаниями. Источники данных могут быть классифицированы по территориальному, административному расположению, степени достоверности, частоте обновляемо-сти, количеству пользователей, секретности и используемым системам хранения и управления данными.

Очевидно, что каждый источник данных имеет соответствующие характеристики для подобной классификации. Например, база данных по продажам товара может вестись в файлах стандарта xBase или на сервере баз данных InterBase. Территориально она может находиться в главном офисе или непосредственно в подразделении по продажам. Административно она отнесена к подразделению по продажам, а не к подразделению информационных технологий. Конфиденциальной информацией обеспечивается меньшее количество пользователей, чем общедоступной. Вся эта информация составляет основу *словаря метаданных* хранилища данных. В словарь метаданных автоматически включаются словари данных источников данных. Здесь же хранятся форматы данных для их последующего согласования. В хорошем словаре метаданных отслеживаются периодичность обновления данных и согласованность их во времени. Если источники данных расположены на разных платформах и обслуживаются различными системами управления, то это также должно быть отражено в словаре метаданных. Задача словаря метаданных в таком случае состоит в том, чтобы освободить разработчиков от необходимости стандартизировать источники данных, а возложить это на хранилище данных.

Создание хранилища данных не должно противоречить действующим системам сбора и обработки информации. Специальные компоненты хранилища данных должны обеспечить своевременное извлечение из них данных и преобразование к единому формату на основе информации из словаря метаданных. Словарь метаданных призван обеспечить корректную перио-

дическую актуализацию хранилища данных. Хотя источниками данных для хранилища данных и являются системы оперативной работы, логическая структура данных в хранилище данных радикально отличается от структур данных в таких системах. Данные, собранные для операционных систем, фундаментально отличаются от той. информации, которая нужна лицам, принимающим решения. Причина проста: операционные функции в принципе отличаются от функций по управлению организацией. Следовательно, должны быть коренные различия и в подходах к системам оперативной работы и к хранилищу данных.

Процесс преобразования данных должен обеспечивать подготовку информации к хранению в том виде, который оптимизирован для быстрого исполнения запросов, необходимых для принятия именно тех решений, которые существенны для увеличения конкурентоспособности, доли рынка и прибыли. Преобразование данных заключается в анализе необработанных корпоративных данных и решении, как они будут представлены конечному пользователю. Для разработки эффективного процесса преобразования необходимы хорошо проработанная модель корпоративных данных и модель технологии принятия решений.

Чтобы наиболее полно учесть требования пользователей по подготовке отчетов и аналитических сводок, необходимо как можно раньше вовлечь их в тестирование развивающейся системы.

Ведущие производители серверов баз данных, например Oracle и Informix, выпустили новые версии своих продуктов для обработки колоссальных объемов информации, до сотен гигабайт, хранимых в хранилище данных. Специфика принятия деловых решений требует достаточно быстрого отклика на аналитические запросы, и новые продукты обеспечивают это. Хорошо организованное хранилище данных оптимизирует существующие инвестиции организации в данные и оборудование. Хранилище данных обеспечивает организацию структурными рамками для установления более надежных взаимоотношений между подразделениями и аппаратом управления. Оно является ключом к развитию динамичных, быстрорастущих организаций.

ИС, реализующие аналитические методы анализа и обработки данных, классифицируются по способу представления данных. Производители ИС этого типа единодушны в том, что способ представления данных, принятый в реляционных СУБД, плохо подходит для решения аналитических задач как с точки зрения пользователя (данные представляются в неудобном виде, плохо соответствующем решаемым задачам), так и с точки зрения эффективности обработки. Данные для пользователя удобно представлять в многоразмерных базах данных ("гиперкубах"), где в качестве размерностей выступают такие интересующие руководителей атрибуты, как время, цена, географический регион и т.п. В связи с тем, что представление базы данных в многоразмерном виде на физическом уровне требует значительных затрат памяти, производители инструментальных средств для создания систем СППР используют различные способы для работы с данными.

Выделяются следующие типы ИС:

1. ИС, хранящие данные в реляционном виде, но имитирующие многоразмерность для пользователя: Find Out (фирма Open Data Corp. - США); Forest and Trees (Trinzic Corp. - США).

2. ИС, хранящие данные в многоразмерных базах: Power Play (Cognos Corp. - США); Brio (Brio Technology Inc. - США); Muse (Occam Research. - США).

3. ИС, хранящие данные как в реляционном виде, так и в многоразмерных базах: SAS datasets (SAS Institute - США); Microsoft Access (MicroStrategy - США).

Представляется, что ИС последнего типа являются более предпочтительными, так как они по желанию пользователя позволяют выбрать между экономичностью представления данных и эффективностью обработки.

Помимо извлечения данных из операционных БД для принятия решений весьма актуален *процесс извлечения знаний (data mining)* в соответствии с информационными потребностями пользователя (рис. П4.2). Тема извлечения знаний не нова для искусственного интеллекта (ИИ). Она является основной при наполнении базы знаний экспертной системы (ЭС). Однако в ЭС основное внимание уделялось проблеме извлечения знаний от экспертов, а не из базы данных (БД). В последнее время интерес к этой теме возрос не в связи с внутренними проблемами ИИ, а в связи с глобальными проблемами современных СУБД, рассмотренными выше.

С точки зрения пользователя в процессе извлечения знаний из БД должны решаться следующие задачи преобразования [2]: данных (т.е. неструктурированных наборов чисел и символов) в информацию (т.е. описание обнаруженных закономерностей); информации в знания

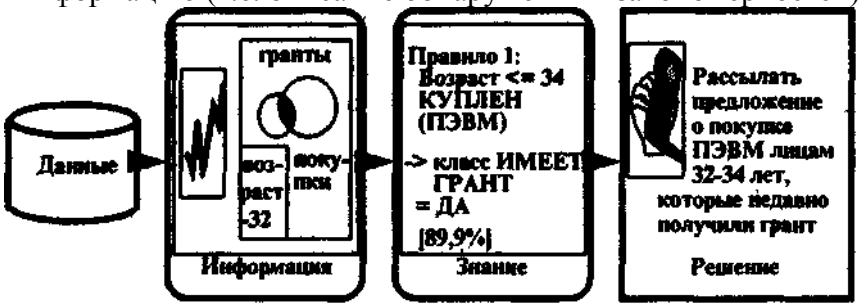


Рис.П4.2. Организация процесса извлечения знаний

(значимые для пользователя закономерности); знаний в решения (последовательность шагов, направленная на достижение информационных потребностей пользователя).

На рис. П4.2 этот процесс проиллюстрирован на примере формирования решения о рассылке предложений покупки ПЭВМ для лиц, имеющих возраст 32 - 34 года и недавно получивших гранты. На первом этапе происходит визуализация данных из некоторой операционной БД и определяются места наибольших пересечений между атрибутами грантов, покупкой ПЭВМ и возрастом. На втором этапе происходят анализ уже только выделенной, ограниченной по размеру информации и формирование правила, описывающего некоторую выявленную закономерность. На третьем этапе происходит формирование решения, адекватного выявленной закономерности.

Интеллектуальные средства извлечения информации позволяют почерпнуть из БД намного более глубокие сведения, чем традиционные системы оперативной обработки транзакций (OLTP) и оперативной аналитической обработки (OLAP). Такие инструменты поиска и анализа выявляют закономерности и выводят из них правила. Эти закономерности и правила можно использовать для принятия решений и прогнозирования их последствий. Кроме того, подобные средства способны ускорять анализ за счет акцентирования внимания на самых важных переменных. Конечно, можно и вручную найти такие закономерности, обратившись к данным с последовательностью запросов, однако рассматриваемые методы извлечения знаний позволяют значительно расширить спектр возможных вариантов, а также работать с огромными массивами данных, что вручную не представляется возможным [1].

Интеллектуальные средства анализа и представления данных получили развитие по нескольким причинам. Организации постепенно накапливают множество данных, связанных с производством или бизнесом. Значительное снижение стоимости систем хранения приводит к тому, что становится легче обеспечить оперативный доступ к огромным информационным массивам. Источниками некоторых таких данных являются традиционные системы OLTP, но большая часть информации представляет собой результат функционирования приложений, появившихся за последние годы. Они регистрируют все детали транзакций и помогают фирмам лучше понять, чего действительно хотят и что делают их клиенты (а не то, о чем они заявляют).

Распространению подобных средств способствовало развитие технологии хранилища данных (см. выше). Если ранее необходимо было сначала собрать данные, проверить и объединить их, то сегодня это уже сделано - данные находятся в информационных хранилищах и дело лишь за тем, чтобы наиболее рационально ими воспользоваться.

Существует несколько интеллектуальных методов выявления и анализа знаний [1]: ассоциация, последовательность, классификация, кластеризация и прогнозирование. *Ассоциация* имеет место в том случае, если несколько событий связаны друг с другом. Например, исследование, проведенное в магазине, может показать, что 55% купивших пиво берут также и сушеную соленую рыбу, а при наличии скидки за такой комплект пиво приобретают в 75% случаев. Располагая этими сведениями, менеджерам легко оценить, насколько действенна предоставляемая скидка. Если существует цепочка связанных во времени событий, то говорят *о последовательности.* Вероятно, наиболее распространенной сегодня операцией интеллектуального анализа знаний является *классификация. С* ее помощью выявляются признаки, характеризующие группу, к которой принадлежит тот или иной объект. Это делается посредством анализа уже

классифицированных объектов и формулирования некоторого набора правил. Например, достаточно болезненной проблемой в бизнесе считается потеря постоянных клиентов. Классификация может помочь выявить характеристики «неустойчивых» покупателей и создать модель, способную предсказать, кто именно склонен уйти к другому поставщику. Использовав ее, можно определить объективные виды скидок и других выгодных предложений, которые будут наиболее действенны для тех или иных типов покупателей.

Кластеризация аналогична классификации, но отличается от нее тем, что сами группы еще не сформированы. С помощью кластеризации интеллектуальные ИС собственно и выделяют различные группы данных. Подобную процедуру можно применять в задачах выявления производственных дефектов или поиска родственных групп клиентов среди обладателей банковских карточек.

Прогнозирование отличается от рассмотренных методов выявления закономерностей тем, что здесь на основе особенностей поведения данных оцениваются будущие значения непрерывно изменяющихся переменных.

Средства извлечения знаний относятся к классу систем, основанных на знаниях, и включают в себя следующие основные механизмы: нейронные сети, деревья решений, индуктивное обучение, визуализацию данных, нечеткие множества и нечеткую логику, статистические методы и их комбинацию.

Нейронные сети (см. Приложение 3) представляют собой совокупность связанных друг с другом узлов, получающих входные данные, осуществляющих их обработку и генерирующих на выходе некоторый результат. Между узлами видимых входного и выходного уровней может находиться какое-то число скрытых уровней обработки. Такая сеть способна обучаться. Для нее имеется специальный набор данных, совокупность входных значений которых порождает заранее установленное множество выходных. Для каждого сочетания обучающих данных на входе выходные значения сравниваются с известным результатом. Если они различаются, то вычисляется корректирующее воздействие, учитываемое при обработке в узлах сети. Указанные шаги повторяются, пока не выполнится условие останова, например, необходимая коррекция не будет превышать заданной величины. Нейронные сети реализуют непрозрачный процесс. Это означает, что построенная в итоге модель не имеет четкой интерпретации, т.е. далеко не всегда понятно, на основе каких логических выводов получаются результаты.

Деревья решений разбивают данные на группы на основе значений тех или иных переменных, используя подход, напоминающий игру в «вопросы». В результате получается иерархия операторов «ЕСЛИ → ТО», которые классифицируют данные. Приведем пример. Если абонент в течение полугода каждый месяц делает на 25% меньше звонков по сотовому телефону, чем за предыдущий, то, с вероятностью 60%, он вскоре откажется от услуг сотовой связи. В настоящее время наблюдается повышение интереса к продуктам, применяющим деревья решений. В основном это объясняется тем, что многие коммерческие проблемы решаются ими быстрее, чем алгоритмами нейронных сетей. К тому же они более просты и понятны для пользователей. В то же время нельзя сказать, что деревья решений всегда действуют безотказно: для определенных типов данных, например при обработке непрерывных величин, они могут оказаться неприемлемыми. Помимо того, набор операторов «ЕСЛИ → ТО» иногда бывает столь же непонятным, как и нейронная сеть, особенно если список условий длинный и сложный.

Программы *визуализации данных* обеспечивают выявление в БД образцов, аномалий и т. д.; они не являются в полном смысле средствами анализа информации, поскольку только представляют ее пользователю в графическом виде. Тем не менее визуальное представление сразу нескольких переменных (например, пяти) достаточно выразительно обобщает очень большие объемы данных.

Индуктивное обучение - это процесс получения знаний путем выполнения индуктивного вывода из фактов, предоставляемых учителем или окружением. Существуют два различных способа индуктивного обучения: обучение по примерам, использующее "учителя", и обучение по наблюдениям (концептуальная кластеризация), которое имеет дело только с описанием исследуемой области и, возможно, ее контекстом. Индуктивные системы, так же как и нейронные сети, поддерживают процесс автоматического выявления закономерностей в БД.

Нечеткие множества и механизмы нечеткой логики обеспечивают представление и использование ненадежных и слабо формализованных данных.

Статистические методы поддерживают процессы классификации, кластеризации и выявления образцов и аномалий в данных и включают набор процедур для проведения кластерного, дискриминантного, дисперсионного и факторного анализа, многомерного шкалирования, различных видов регрессии и т.п.

В настоящее время рассматриваемое направление активно развивается. Так, в Северной Америке существует примерно 20 фирм, занимающихся разработкой ИС извлечения знаний. Ведущие позиции занимают следующие фирмы: AbTech Corp.(AIM), Reduct Systems (DataLogic), Teranet IA Inc.(ModelWare), Attar Software (XpertRule Analyzer), Agnoss Software (KnowledgeSEEKER), Data Patterns (PC-MARS).

Для выбора требуемого ИС необходимо ответить на ряд вопросов, основные из которых следующие [1]:

• Какого рода задачи данное ИС решает ?

• Какую операционную систему и аппаратные средства использует?

• Требует ли выделения подмножества данных или работает со всей информацией БД непосредственно?

• Какой пользовательский интерфейс оно применяет для ввода и интерпретации данных?

• Каково максимальное число обрабатываемых переменных и записей?

• Какие подходы используются для моделирования данных (статистический анализ, нейронная сеть, дерево решений, визуализация и т.д.)?

• В какой мере ИС чувствительно к искажениям данных?

• Насколько понятны результаты? и т.д.

Обычно ИС извлечения знаний классифицируются по методам, которые используются в ИС для анализа и извлечения знаний: классификация, кластеризация, визуализация, нечеткая логика, статистические методы и, наконец, комбинированные методы. Примеры существующих ИС в области извлечения знаний приведены в табл. П4.1.

Таблица П4.1 Примеры ИС для извлечения знаний

Методы классификации	AC2 (ISOFT, США) AIM (AbTech Corp., США) C4.5 (Morgan Kaufmann Publishers, США) DataLogic/R (Reduct Systems, США) IND (COSMIC, США) IDIS (Intelligence Ware, США) KATE (AcknoSoft, США)
Методы кластеризации	Autoclass III (COSMIC, США) DBProfile (Advanced Software Applications, США) ModelMax (Advanced Software Applications, США)
Методы визуализации	NetMAP (ALTA Analytics, Inc., США) Win Viz (Information Technology Institute, США)
Методы нечеткой логики	DataEngine (Mgmnt. Intelligenter Technologien, США) Level5 Quest (Information Builders inc., США)
Статистические методы	Cornerstone (BBN Software Products, США) DATA (TreeAge Software, США) JMP(SAS Institute, США) SAS (SAS Institute, США)
Комбинированные методы	Clementine (Integral Solutions Ltd., США) Data Mariner (Logica UK Ltd., США) Database Mining Workstation (HNC Software Inc., США) Information Harvester (Information Harvesting, Inc., США) Recon (Lockheed Martin Product & Services, США)

На базе этих ИС разработан ряд приложений [3] в области финансов (Accounts Receivable Classifier: Internal Revenue Service, США; Data Cleaning: Lockheed, США; Data Verification for Foreign Prices:

Reuters, США), маркетинга (Ad Tracking System: AdTrack, Inc., США; Marketing Research: Dickinson Direct, США), здравоохранения (Desease Modeling, Severity Outcomes, Data Cleanup: Med-AI, Inc., США; KEFIR: GTE Labs, США), производства товаров и продуктов, в науке, обра-

зовании и исследованиях. Для иллюстрации возможностей современных ИС в области извлечения знаний ниже приведены примеры некоторых приложений в области финансов.

Система Accounts Receivable Classifier обеспечивает классификацию чеков, допустимых к приему. Выделяет чеки с высокой вероятностью оплаты. Использует архивные БД о тысячах оплаченных и неоплаченных налоговых квитанциях для выделения моделей неплательщиков. Поддерживает такие методы выделения признаков, как описательная статистика, алгоритмы кластеризации, полиномиальные сети. Цель - улучшение распределения ресурсов сбора налогов.

Система Data Cleaning обеспечивает чистку базы данных архивной финансовой информации, используемой аналитиками для построения и прогона финансовых моделей для принятия решений об инвестировании, для прогнозирования и т.д. Средства визуализации данных, дедуктивная база данных и методы индукции были использованы для чистки БД, содержащей информацию о 2200 связях мексиканского и британского правительств.

Система Data Verification for Foreign Prices предназначена для верификации данных по зарубежным ценам. Система обнаруживает ошибки в поступающих в реальном времени данных о курсах обмена иностранных валют; использует технику нейронных сетей и индукции; модели осуществляют грубое прогнозирование цен на основе данных об их последних изменениях; если поступающие данные сильно отклоняются от предсказанного значения, они помечаются как подозрительные. Обнаружение ошибок осуществляется на основе знаний, автоматически выведенных из легкодоступных данных, а не полученных от экспертов; система может адаптироваться к изменяющейся обстановке через обучение на новых данных.

Система Forecasting Arrears Problems предназначена для прогнозирования неплатежей. Предсказывает задолженность с помощью анализа данных методами индукции среди 500 000 закладных чеков.

Система Mining for Underwriting Rules использует методы извлечения в страховании. В частности, она использует средства визуализации и методы индукции для профилирования информации о потенциальных клиентах.

ЛИТЕРАТУРА

1. *Едельштайн Н.* **Интеллектуальные средства анализа, интерпретации и представления данных в информационных хранилищах** // Компьютеруик. -1996. - №16.

2. *Hall C.* **The devil s in the details: techniques, tools, and applications for database mining and knowledge discovery** //Intelligent Software Strategies. - P. I. V XI. -№9 - 1995, September.

3. Hall C. **The devil s in the details: techniques, tools, and applications for database mining and knowledge discovery**//Intelligent Software Strategies - P.II. V. XI. -№9. - 1995, October.

4. *Inmon W.H.* **Building the Data Warehouse.** - NY: John Wiley & Sons, Inc., 1992. - 298 P.