УЗБЕКСКОЕ АГЕНТСТВО СВЯЗИ И ИНФОРМАТИЗАЦИИ
ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Кафедра «Информационные технологии»

Методические указания к практическим и лабораторным работам по курсу «интеллектуальные системы»

Для студентов направления
5521900 «Информатика и информационные технологии»

Ташкент - 2008

Содержание

Введение

Настоящие методические указания по курсу «Интеллектуальные системы» предназначены для подготовки студентов по направлению «Информатика и информационные технологии». По своему содержанию соответствует типовой программе данного направления. В методические указания включены 8 работ. Выполнение каждой из них рассчитано на 4 часа, из них: 2 часа на практические и 2 часа на лабораторные занятия.

Описание каждой работы включает:

– теоретическая часть;
– порядок выполнения работы;
– варианты заданий;
– требования к содержанию отчета;
– контрольные вопросы

Студент по каждой работе обязан представить отчет и ответить на контрольные вопросы.

Требования к содержанию отчета

Отчет должен содержать:

– название темы варианта;
– цель работы
– теоретическую часть
– схему
– программный код
– интерфейс пользователя

Студент, не выполнивший и не защитивший две работы в установленный кафедрой срок не допускается к дальнейшему выполнению работ до тех пор пока не ликвидирует задолжность.

Работа № 1
Использование семантических сетей
для представления знаний

Цель работы: Научиться использовать семантические сети для представления знаний в интеллектуальных системах.

1. Теоретическая часть

Семантическая сеть – это один из способов представления знаний. Изначально семантическая сеть была задумана как модель представления долговременной памяти в психологии, но впоследствии стала одним из способов представления знаний в ЭС.

Семантика – означает общие отношения между символами и объектами из этих символов.

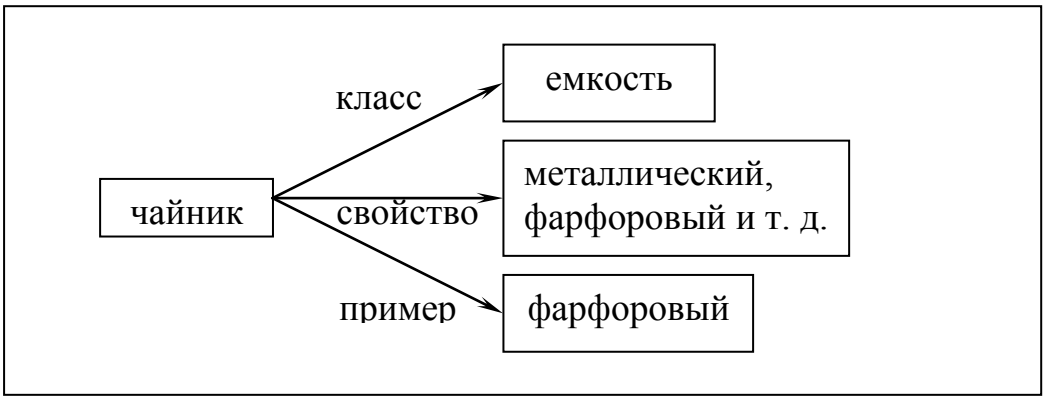

Рис.1. Простейший образец семантической сети.

Вершины – это объекты, дуги – это отношения. Семантическая модель не раскрывает сама по себе каким образом осуществляется представление знаний. Поэтому семантическая сеть рассматривается как метод представления знаний и структурирования знаний. При расширении семантической сети в ней возникают другие отношения:

IS –A (принадлежит) и PART OF (является частью) отношение:

целое → часть.

Ласточка IS – A птица, «нос» PART OF «тело». Например:

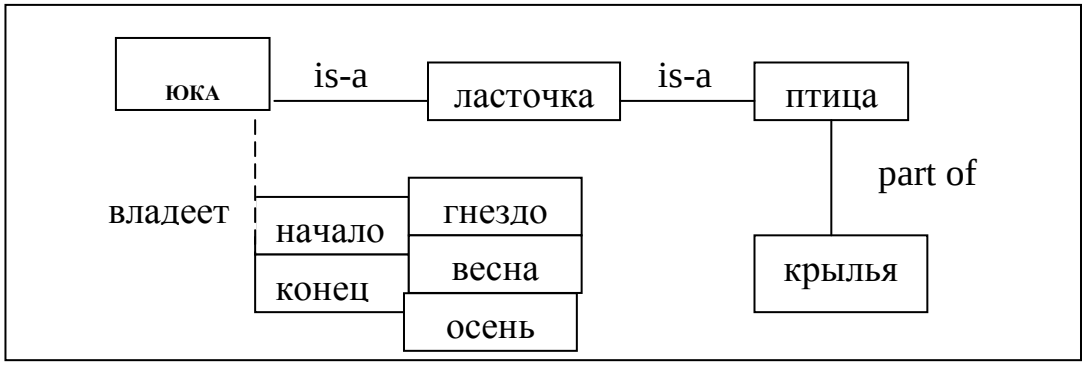

Рис.2. Расширение семантической сети

Могут быть и другие отношения: владеет. Тогда семантическая сеть расширяется иерархически (вершина имеет две ветви). Кроме того, можно расширить сеть и другим отношением:

период → «весна – лето».

Получается иерархическая структура понятия ЮКО. Можно разбить на подсхемы. Большой проблемой для семантических сетей является то, что результат вывода не гарантирует достоверности, так как вывод есть просто наследование свойств ветви is-a.

Для отображения иерархических отношений между объектами и введения единой семантики в семантические сети было предложено использовать процедурные сети. Сеть строится на основе класса (понятия); вершины, дуги и процедуры представлены как объекты.

2. Порядок выполнения работы

1. Изучить теоретическую часть по приведенным выше данным и дополнительной литературе;
2. Просмотреть демонстрационный пример;
3. Получить у преподавателя вариант задания для выполнения;
4. Построить семантическую модель заданного объекта;
5. Реализовать программу с использованием семантической модели

3. Варианты заданий

Используя соответствующие дуги построить семантическую сеть, касающуюся:

1. географии какого-либо региона. Дуги: государство, страна, континент, широта.
2. диагностики глазных заболеваний. Дуги: категории болезней, патофизиологическое состояние, наблюдения, симптомы.
3. распознавания химических структур. Дуги: формула вещества, свойства вещества, область применения, меры предосторожности.
4. процедуры поиска полезных ископаемых. Дуги: наименование ископаемого, расположение месторождения, глубина залегания, методы добычи.
5. судебной процедуры. Дуги: юридическое лицо, событие, меры воздействия, способы расследования.
6. распределения продуктов по магазинам. Дуги: источник снабжения, наименование продукта, способ транспортировки, конечный пункт транспортировки.
7. определения принадлежности животного к определенному виду, типу, семейству. Дуги: место обитания, строение, особенности поведения, вид питания.
8. классификации пищевых продуктов. Дуги: наименование продукта, составляющие части, способ приготовления, срок хранения.

9. распознавания типа компьютера. Дуги: страна изготовитель, стандартная конфигурация, область применения, используемое программное обеспечение.
10. иерархической структуры БД. Дуги: система, состояние, назначение, взаимодействие составляющих.

4. Контрольные вопросы

1. Что такое семантическая сеть и для чего ее применяют?
2. В чем состоит идея создания семантической сети?
3. Каким образом представляются данные в семантической сети?
4. Существуют ли ограничения на число связей элементов, свойств и сложность при построении семантической сети?
5. Какие отношения предложены в качестве операторов отношения для группировки вершин?

Работа № 2
Использование фреймов для представления знаний

Цель работы: Научиться использовать фреймы для представления знаний в интеллектуальных системах

1. Теоретическая часть

Фреймы - один из распространенных формализмов представления знаний в ЭС. Фрейм можно представить себе как структуру, состоящую из набора ячеек - слотов. Каждый слот состоит из имени и ассоциируемых с ним значений. Значения могут представлять собой данные, процедуры, ссылки на другие фреймы или быть пустыми. Такое построение оказывается очень удобным для моделирования аналогий, описания областей с родовидовыми связями понятий и т.п.

Любой фрейм состоит из некоторых составляющих, имена и содержание которых описано ниже:

1. Имя фрейма. Это идентификатор, присваиваемый фрейму, фрейм должен иметь имя уникальное в данной фреймовой системе.
2. Имя слота. Это идентификатор, присваиваемый слоту; слот должен иметь уникальное имя во фрейме, к которому он принадлежит. Обычно имя слота не несет никакой смысловой нагрузки и является лишь идентификатором данного слота.
3. Указатели наследования. Эти указатели касаются только фреймовых систем иерархического типа, основанные на отношениях “абстрактное-конкретное”, они показывают, какую информацию об атрибутах слотов во фрейме верхнего уровня

наследуют слоты с такими же именами во фрейме нижнего уровня. Типичные указатели наследования Unique (U: - уникальный), Same (S: такой же), Range (R: установление границ), Override (O: игнорировать) и т.п. U показывает, что фрейм может иметь слоты с разными значениями: S - все слоты должны иметь одинаковые значения, R - значение слотов фрейма нижнего уровня должны находиться в пределах, указанных значениями слотов фрейма верхнего уровня, O - при отсутствии указания значение слота фрейма верхнего уровня становится значением слота фрейма нижнего уровня, но в случае определения нового значения слотов фреймов нижних уровней указываются в качестве значений слотов.

4. Указание типа данных. указывается, что слот имеет численное значение, либо служит указателем другого фрейма. К типам данных относятся:
   FRAME (указатель), INTEGER (целый), REAL (действительный), BOOL (булев), LISP (присоединенная процедура), TEXT (текст), LIST (список), TABLE (таблица), EXPRESSION (выражение) и др.
5. Значение слота. Пункт ввода значения слота. Значение слота должно совпадать с указанным типом данных этого слота, кроме того должно выполняться условие наследования.
6. Демон. Здесь дается определение демонов типа IF-NEEDED, IF-ADDED, IF-REMOVED и т.д. Демоном называется процедура, автоматически запускаемая при выполнении некоторого условия. демоны запускаются при обращении к соответствующему слоту. Кроме того, демон является разновидностью присоединенной процедуры.
7. Присоединенная процедура. В качестве значения слота можно использовать программу процедурного типа. Когда мы говорим, что в моделях представления знаний фреймами объединяются процедурные и декларативные знания, то считаем демоны и присоединенные процедуры процедурными знаниями.

Особенностью иерархической структуры является то, что информация об атрибутах фрейма на верхнем уровне совместно используется всеми фреймами нижних уровней, связанных с ним.

Например: Фреймовое представление конференции.

Иерархические фреймовые структуры базируются на отношениях IS – A между фреймами, описывающими некоторую конференцию. Все фреймы должны содержать информацию о ДАТЕ, МЕСТЕ, НАЗВАНИИ ТЕМЫ, ДОКЛАДЧИКЕ. Таким образом, на самом верхнем уровне определен фрейм КОНФЕРЕНЦИЯ.

Конференции разделяются на коммерческие и по развитию. Они составляют дочерние фреймы. В них могут быть добавлены слоты: объем торговли и бюджет.

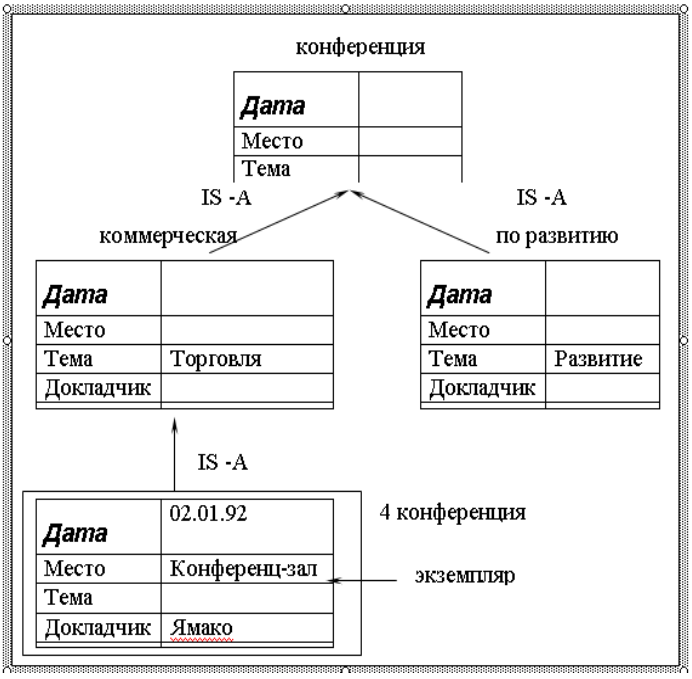

Рис.3. Пример фреймовой модели

2. Порядок выполнения работы:

1. Изучить теоретическую часть по приведенным выше данным и дополнительной литературе.
2. Просмотреть демонстрационный пример.
3. Получить у преподавателя вариант задания для выполнения.
4. Построить фреймовую модель заданного объекта;
5. Реализовать программу с использованием фреймовой модели

3. Варианты заданий

Используя фреймовую модель представления знаний реализовать структуру отношений, описывающие следующие ситуации:

1. экзамен по дисциплине за семестр у преподавателя при составляющих: семестр, экзамен, преподаватель, оценка, студент, получать.
2. ведомость при составляющих: дисциплина, студент, экзамен, семестр, преподаватель, оценка.
3. конференция по коммерческим вопросам при составляющих: дата, место проведения, тема, цель выступающие.
4. получение оценки при составляющих: преподаватель, студент, оценка, получать.
5. использования изделия при составляющих: организация, разработка технологического решения, исследование «физического эффекта», методы создания изделия.

6. информационная структура БД в машиностроении при составляющих: физические эффекты, технические решения, изделия, объект поставки изделия, приборы и стенды, нормативы.
7. классификация продукта при составляющих: название, область применения, способ хранения, способ транспортировки.
8. аудитория (описание) при составляющих: вместимость, назначение, составляющие, местонахождение.
9. животный мир при составляющих: вид, тип, среда обитания, особенности поведения.

4. Контрольные вопросы

1. Что представляет из себя фрейм, его составные части?
2. Что такое слот и из каких частей он состоит?
3. Для чего служат имя фрейма и имя слота?
4. Для чего служат указатели наследования?
5. для чего служат указание типа данных, демон?
6. Для чего служат присоединенная процедура и значение слота?

Работа №3
Описание предметной области.
разработка базы фактов и правил интеллектуальной системы

Цель работы: Научиться строить модель предметной области, описывать решаемую задачу правилами продукционной системы и формализовать используемые знания.

1. Теоретическая часть

В данной работе мы рассмотрим построение базы знаний на основе сведений, полученных от эксперта. Процесс ее построение состоит из двух этапов:

- описание предметной области;
- выбор метода и модели представления знаний;

Инженер знаний должен корректно сформулировать задачу. В то же время он должен уметь распознать, что задача не структурирована, и в этом случае воздержаться от попыток ее формализовать или применить систематические методы решения. Главная цель начального этапа построения базы знаний - определить, как будет выглядеть описание предметной области на различных уровнях абстракции. Экспертная система включает базу знаний, которая создается путем формализации некоторой предметной области, а та в свою очередь является результатом абстрагирования определенных сущностей реального мира.

После того как предметная область выделена, инженер знаний должен

ее формально описать. Для этого ему необходимо выбрать какой-либо способ представления знаний о ней (модель представления знаний). В. настоящее время отсутствует общий способ представления знаний, который бы годился для формализации предметных областей любой природы. Инженер знаний должен воспользоваться той моделью, с помощью которой можно лучше всего отобразить специфику предметной области. Когда будет создана общая теория представления знаний (если это вообще когда-нибудь произойдет), ее можно будет применять для формализации новых предметных областей без учета их особенностей.

Определение характера решаемых задач

Обратимся к примеру из медицинской практики. Предположим, что мы хотим построить экспертную систему, предназначенную для обработки результатов химического анализа крови, выполненного в лаборатории. Инженер знаний прежде всего обязан провести опрос эксперта и только потом приступать к построению системы. Эксперт, безусловно, должен быть специалистом в той области, в которой будет работать система. Первым делом необходимо определить целевое назначение системы. Какие, собственно 'задачи предстоит решать системе, основанной на знаниях? Цели разработки системы следует сформулировать точно, полно и непротиворечиво. Например, для диагностической системы это может быть получение ответов на такие вопросы:

1. Здоров ли пациент (исправна ли система)? Если нет, то какое именно у него заболевание? Если имеете» несколько заболеваний, то какое из них наиболее опасно?

2. Какие изменения в диете и рационе питания следует рекомендовать и, какие из них считаются особенно важными?

3. Какие лабораторные исследования необходимо провести дополнительно и, какие из них являются первоочередными?

4. Как нужно изменить образ жизни пациента или климатические условия, в которых он находится?

5. Нужно ли направить пациента для обследования к врачам-специалистам и если да, то к каким именно? Подумайте, на какие еще вопросы должна уметь отвечать наша диагностическая система?

После того как цель разработки системы определена, инженер знаний приступает к формулированию подцелей. Это поможет ему установить иерархическую структуру системы и разбить ее на модули. Введение тех или иных подцелей обусловливается наличием связей между отдельными фрагментами знаний. Проблема сводится к разбиению задачи на две или несколько подзадач меньшей сложности и последующему поиску их решений. При необходимости, полученные в результате разбиения подзадачи могут дробиться и дальше.

Выявление объектов предметной области

Следующим шагом построения базы знаний является выделение объектов предметной области, или в терминах теории систем установление границ системы. Как и формальная система, *совокупность* выделенных

понятий должна быть точной, полной и непротиворечивой. Итак, какие конкретно лабораторные анализы необходимо провести? Следует ли обратиться к истории болезни пациента и если да, то какие данные в ней наиболее важны? Какие еще сведения о пациенте могут представлять интерес (например, отмечались ли раковые заболевания у родственников)? Нужно ли учитывать лекарства, которые больной принимал ранее, а также предыдущие назначения врачей? Играет ли какую-нибудь роль род занятий и образ жизни больного, климатические условия и режим питания? Какие симптомы у него наблюдаются (головные боли, жар и т.д.)?

Установление взаимосвязей между объектами

После выявления объектов предметной области необходимо установить, какие между ними имеются связи. Например, низкое содержание тиреотропного гормона в крови может свидетельствовать о повышенной активности поджелудочной железы, но может означать и нечто другое. Следует стремиться к выявлению как можно большего количества связей, в идеале - всех, которые существуют в предметной области.

Формализация знаний

Полученное качественное описание предметной области должно быть представлено средствами какого-либо формального языка, чтобы привести это описание к виду, позволяющему поместить его в базу знаний системы. Для решения этой задачи выбирается подходящая модель представления знаний, с помощью которой сведения о предметной области можно выразить формально.

Рассмотрим пример.

Подходящей задачей, при решении которой можно использовать продукционную модель, может быть задача, вытекающая из следующей ситуации: к директору крупной технической фирмы пришёл человек, желающий устроиться на работу. Директор располагает сведениями о его квалификации, о потребностях фирмы в специалистах и общем положении дел в фирме. Ему нужно решить, какую должность в фирме может занять посетитель.

Рассмотрим модель «Посетитель», выявим необходимые атрибуты для принятия решения о приеме на работу.

Объект: посетитель.

Атрибуты:

1. наличие ученого звания
2. стаж работы по специальности
3. посетитель сделал важное открытие
4. средний бал посетителя за время учебы

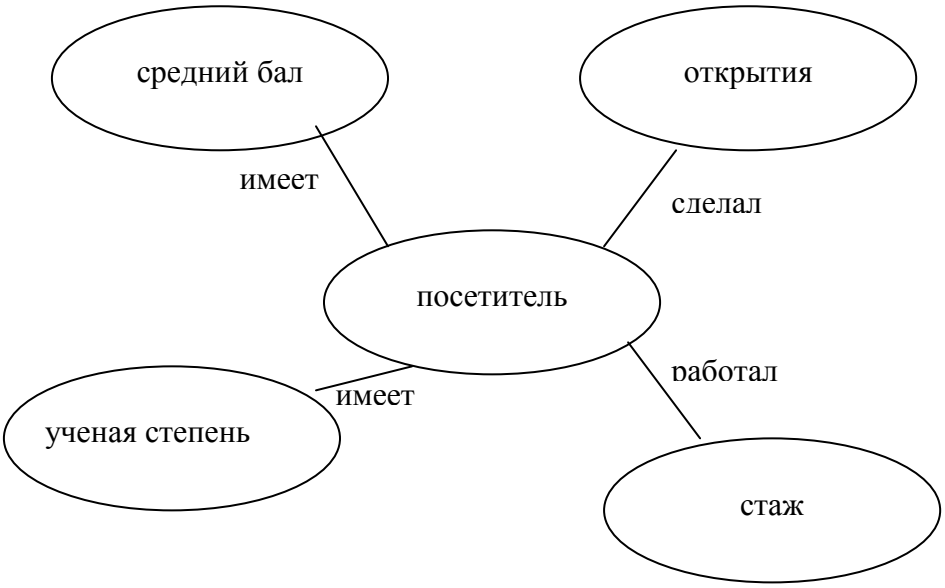

Рис.4. Модель предметной области

2. Порядок выполнения работы

1. Проанализировать полученное задание
2. Определить характер решаемой задачи.
3. Выделить объекты предметной области.
4. Выбрать атрибуты, свойства характеризующие объекты.
5. Установить связи между объектами в виде правил продукционной системы

3. Варианты заданий

Описать предметную область для следующих задач:

1. диагностика неисправностей электронной аппаратуры
2. диагностика неисправностей автомобиля
3. диагностика заболеваний (по выбору)
4. прогнозирование (по выбору)
    a. спортивных мероприятий
    b. телепередач
    c. природных катаклизмов

    и т.п.
5. классификация объектов (по выбору)
6. задачи информационно-советующего характера (по выбору)
    a. помощник заведующего склада
    b. помощник аптекаря
    c. помощник оператора справочной службы
    d. выбор должности
    e. проведение отпуска

    и т.п.

4. Контрольные вопросы

1. Какие модели представления знаний используются?
2. Типы задач экспертных систем?
3. Чем характеризуются объекты предметной области?
4. Как могут быть представлены факты в ЭС?

Работа № 4
Использование правил продукции для представления знаний. прямая цепочка рассуждений

Цель работы: Научиться использовать метод правил продукции для представления знаний на основе прямой цепочки рассуждений.

1. Теоретическая часть

Представление знаний с помощью правил продукции – самая распространенная форма реализации БЗ. С помощью продукции можно описать практически любую систему знаний.

Правила продукций представлены в виде импликации:

$$p_i : s_i \rightarrow d_i ,$$

где p_i - правило продукции,
s_i - условие применения правила,
d_i - результат применения правила.

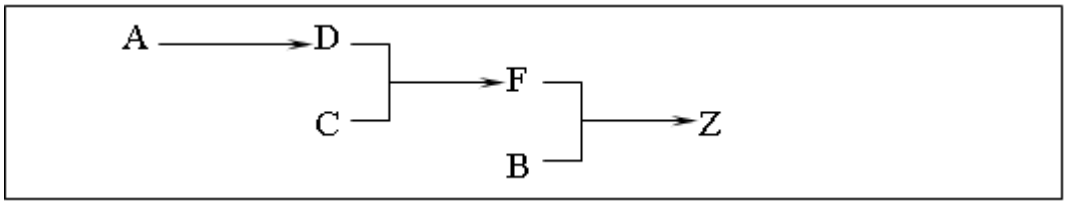

Рис.5. Пример использования правил продукции:

1. Если есть цены на выпускаемые изделия (A) - завод отпускает продукцию (D).
2. Если завод выпускает продукцию и выполняет план по ее реализации (C) - рабочие получают премию (F).
3. Если рабочие получают премию и растет производительность производства (B)- завод производит продукцию сверх плана (Z).

Рассмотрим цепочки выводов.

Прямой способ рассуждения.

По известным фактам отыскивается заключение, которое следует из этих фактов и накапливается рабочая память.

Это приводит к выполнению 2 правила.

C & D→ F, и факт «F» помещается в рабочую память. Тогда опять проверяются правила из базы. Первое правило выполняется F & B → Z , вследствие этого Z заносится в рабочую память. А так как Z является целью, то поиск заканчивается. Этот метод называется прямой цепочкой рассуждений, поскольку поиск новой информации происходит в направлении стрелок, разделяющих левые и правые части правил.

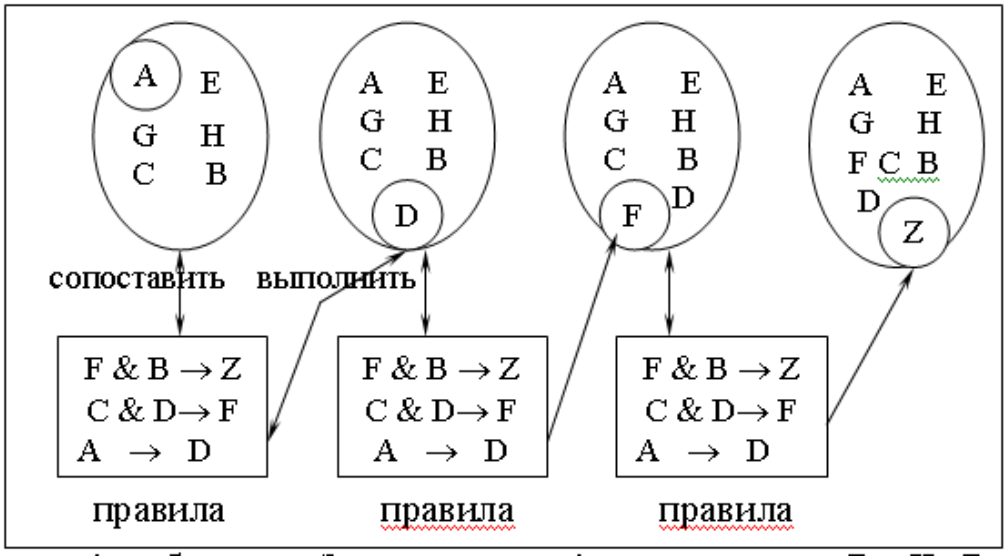

Рис.6. Пример реализации прямой цепочки рассуждений

Обобщённый алгоритм работы системы, реализующий прямую цепочку рассуждений, можно свести к следующему :

1. Определить исходное состояние.
2. Занести переменную условия в очередь переменных логического вывода, а её значение - в список переменных.
3. Просмотреть список переменных и найти ту переменную, имя которой стоит в начале очереди переменных логического вывода. Если переменная найдена, записать в указатель переменных условия номер правила и число 1. Если переменная не найдена, перейти к шагу 6.
4. Присвоить значения не проинициализированным переменным условной части найденного правила (если такие есть). Имена переменных содержатся в списке переменных условия. Проверить все условия правила и в случае их истинности обратиться к части ТО правила.
5. Присвоить значение переменной, входящей в часть ТО правила, и поместить её в конец очереди переменных логического вывода.
6. Удалить переменную, стоящую в начале очереди переменных логического вывода, если она больше не встречается в условной части какого-либо правила.

Закончить процесс рассуждений, как только опустеет очередь переменных логического вывода. Если же в очереди ещё есть переменные, вернуться к шагу 3.

2. Порядок выполнения работы:

1. Изучить теоретическую часть по приведенным выше данным и дополнительной литературе.
2. Просмотреть демонстрационный пример.
3. Получить у преподавателя вариант задания для выполнения.
4. Построить прямую цепочку рассуждений
5. Реализовать программу для прямой цепочки рассуждений

3. Варианты заданий

Реализовать прямую цепочку рассуждений для следующих задач:

1. прогнозирование неисправностей электронной аппаратуры
2. прогнозирование неисправностей автомобиля
3. прогнозирование заболеваний (по выбору)
4. прогнозирование (по выбору)
    a. спортивных мероприятий
    b. телепередач
    c. природных катаклизмов
    и т.п.
5. классификация объектов (по выбору)
6. задачи информационно-советующего характера (по выбору)
    a. помощник заведующего склада
    b. помощник аптекаря
    c. помощник оператора справочной службы
    d. выбор должности
    e. проведение отпуска
    и т.п.

4. Контрольные вопросы

1. Что такое правила продукции и в чем их сущность?
2. В чем отличие прямой цепочки рассуждений от обратной цепочки рассуждений?
3. Из каких частей состоит продукционная система?
4. Значение и применение частей продукционной системы для представления знаний?

Работа № 5
Использование правил продукции для представления знаний обратная цепочка рассуждений

Цель работы: Научиться строить дерево целей и разрабатывать алгоритм Решений на основе обратной цепочки рассуждений.

1. Теоретическая часть

Прямой метод рассуждений имеет следующий недостаток. При большом количестве правил, чтобы найти информацию, связанную с Z, нужно выполнить много правил, не связанных с Z. При этом метод оказывается напрасной тратой времени и денег.

В таких ситуациях более рентабельной является обратная цепочка рассуждений.

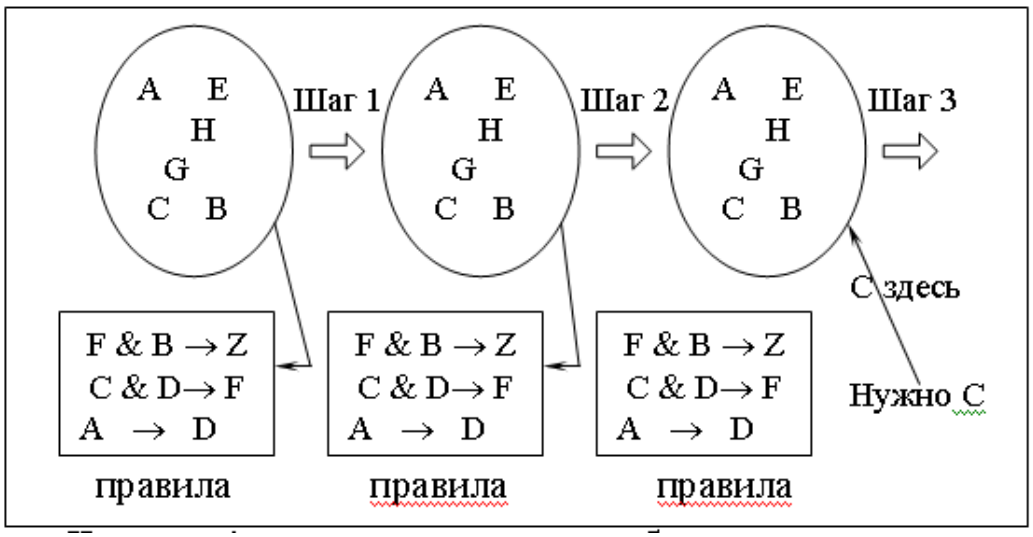

Рис.7 Пример реализации обратной цепочки рассуждений

При этом методе система начинает с того, что нужно доказать, например, что ситуация Z существует, и нужно выполнить только те правила, которые относятся к установлению этого факта.

На шаге 1 системе говорится, чтобы она установила, что ситуация Z существует. Она ищет Z в базе, а если его нет, будет искать правило, приводящее к установлению Z. Она находит правило

$F \& B \rightarrow Z$

и решает, что надо установить F и B.

На шаге 2 система пытается найти факт F или в базе данных или среди правил. Находит правило $C \& D \rightarrow F$ и решает, что необходимо установить существование фактов C и D.

На шагах 3-5 система находит C, затем находит A прежде, чем получит заключение о D.

На шагах 6-8 система выполняет третье правило, чтобы установить D, затем исполняет второе правило, чтобы установить F и наконец – первое правило, чтобы установить основную цель – факт существования Z.

Теперь нужно наглядно её представить. Для описания подобных задач обычно используются диаграммы, которые называются ***деревьями решений.*** Деревья решений дают необходимую наглядность и позволяют проследить ход рассуждений.

Диаграммы называются деревьями решений потому, что, подобно настоящему дереву, имеют ветви. Ветви деревьев решений заканчиваются логическими выводами. Для рассматриваемого примера вывод заключается в том, предложит ли директор должность поступающему на работу, и если да, то какую. Многие задачи сложны, и их непросто представить (или для их решения не собираются использовать ЭС). Дерево решений помогает преодолеть эти трудности.

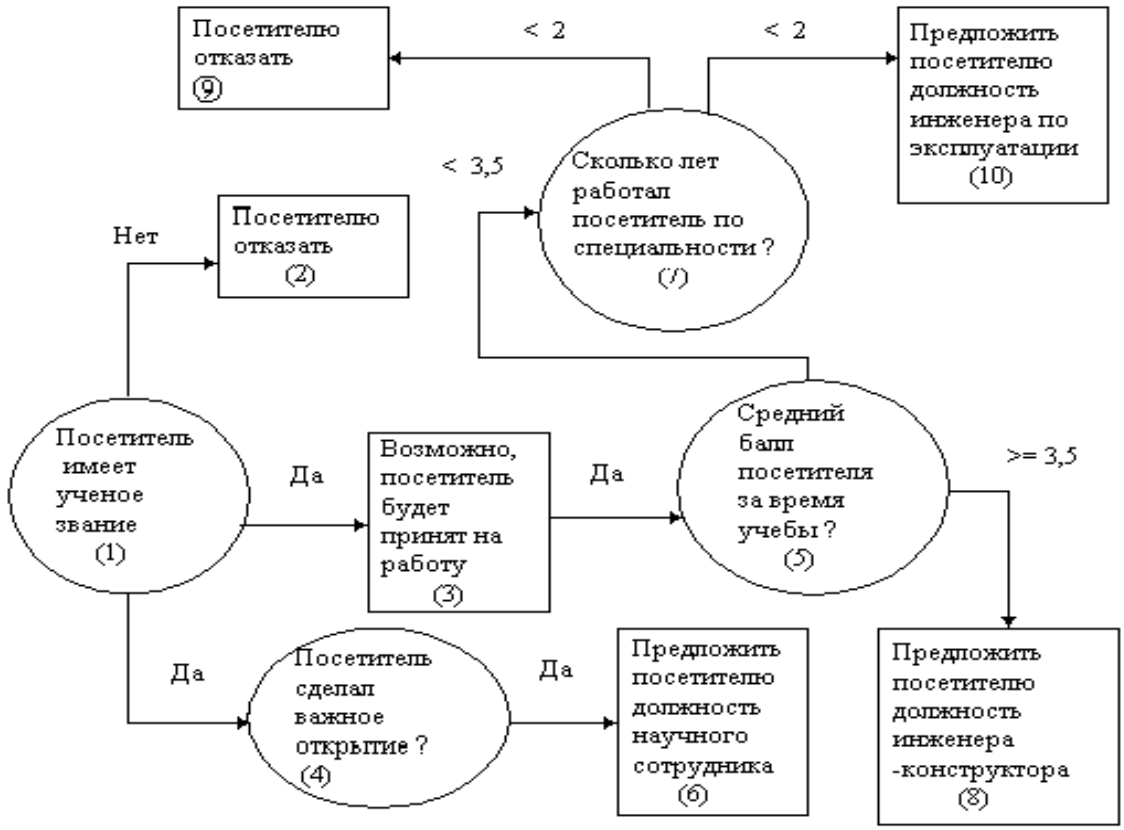

Рис. 8. Дерево решений для выбора должности.

На рис.8 показано дерево решений для примера с приёмом на работу. Видно, что диаграмма состоит из кружков и прямоугольников, которые называются вершинами. Каждой вершине присваивается номер. На вершины можно ссылаться по этим номерам. Линии, соединяющие вершины, называются дугами или ветвями. Кружки, содержащие вопросы, называются вершинами решений. Прямоугольники содержат цели диаграммы и означают логические выводы. Линии показывают

направление диаграммы. Многие вершины имеют сразу по несколько ветвей, связывающих их с другими вершинами. Выбор выходящей из вершины ветви определяется проверкой условия, содержащегося в вершине.

Например, вершина 5 (см. Рис.8) содержит вопрос, на который есть два возможных ответа, и поэтому у неё два пути в зависимости от среднего балла посетителя за время учёбы, то есть возможен выбор одной из двух ветвей. Если средний балл равен 3.1, то будет выбран первый путь, так как 3.1 меньше 3.5. В программе под средний балл сначала отводится переменная, а затем ей присваивается значение. Можно сказать, что вершины содержат переменные, а пути - это условия, в соответствии с которыми переменным присваиваются значения. После того как для проблемной области сформулированы правила, эти условия становятся условными частями (ЕСЛИ) правила. Прямоугольники содержат частные или общие выводы. Например, прямоугольник на рис.1 может содержать ответ на вопрос, будет ли посетителю предложена работа. Общая цель системы, в которой реализованы обратные рассуждения, - получить окончательный ответ. Локальной целью может быть содержащийся в прямоугольнике на рис.8 ответ на вопрос, будет ли посетителю предложена должность. Однако эта вершина имеет и исходящие ветви, и, следовательно, через неё может проходить путь к следующему логическому выводу. В последнем случае, поскольку исходящая ветвь не содержит условия и она только одна, говорят, что вершина содержит локальный вывод для другой цели. Локальный вывод - это также составляющая условной части правила.

Обобщённый алгоритм работы системы с обратной цепочкой выводов.

Система, реализующая обратную цепочку рассуждений, должна выполнять следующие шаги :

1. Определить переменную логического вывода.
2. В списке логических выводов искать первое вхождение этой переменной. Если переменная найдена, в стек логических выводов поместить номер соответствующего правила и установить номер условия равным 1. Если переменная не найдена, сообщить пользователю, что ответ найти невозможно.
3. Присвоить значения всем переменным условия из данного правила.
4. Если в списке переменных указано, что какой-либо переменной условия не присвоено значение и её нет среди переменных логического вывода (её нет в списке логических выводов), запросить её значение у пользователя.
5. Если какая-либо переменная условия входит в переменные логического вывода, поместить в стек номер правила, в логический вывод которого она входит, и вернуться к шагу 3.

6. Если из правила нельзя определить значение переменной, удалить соответствующий ему элемент из стека и в списке логических выводов продолжить поиск правила с этой переменной логического вывода.
7. Если такое правило найдено, перейти к шагу 3.
8. Если переменная не найдена ни в одном из оставшихся правил в логическом выводе, правило для предыдущего вывода не верно. Если предыдущего вывода не существует, сообщить пользователю, что ответ получить невозможно. Если предыдущий вывод существует, вернуться к шагу 6.
9. Определить значение переменной из правила, расположенного в начале стека; правило из стека удалить. Если есть ещё переменные логического вывода, увеличить значение номера условия и для проверки оставшихся переменных вернуться к шагу 3. Если больше нет переменных логического вывода, сообщить пользователю окончательный вывод.

2. Порядок выполнения работы*:*

1. Выбрать по заданной теме 15-20 правил принятия решения;
2. Упорядочить их по степени важности;
3. Построить дерево принятия решения;
4. Построить список переменных логических выводов;
5. Написать программу для реализации обратной цепочки рассуждений

3. Варианты заданий

Реализовать прямую цепочку рассуждений для следующих задач:

1. диагностика неисправностей электронной аппаратуры
2. диагностика неисправностей автомобиля
3. диагностика заболеваний (по выбору)
4. Анализ объекта (по выбору)
    a. спортивных мероприятий
    b. телепередач
    c. природных катаклизмов
    и т.п.
5. задачи информационно-советующего характера (по выбору)
    a. помощник заведующего склада
    b. помощник аптекаря
    c. помощник оператора справочной службы
    d. выбор должности
    e. проведение отпуска
    и т.п.

4. Контрольные вопросы

1. Чем отличаются «прямая» и «обратная» цепочки рассуждений?

2. Какие виды правил существуют?
3. Как контролируется вывод правил из БЗ?
4. Как учитывается достоверность заключительной части правила?

Работа № 6

Использование теории Байеса при проектировании интеллектуальных систем

Цель работы: Научиться использовать формулы условной вероятности при построении базы знаний

1. Теоретическая часть

В основе многих эвристических правил лежит вероятность появления определенного события, вычислить которую может только эксперт, т.е. эксперт делает обоснование предположения в своей проблемной области. В действительности это означает, что существуют статистические данные, позволяющие делать какие-либо предположения. Это могут быть, например, медицинский диагноз, который врач ставит на основании своих наблюдений над пациентом. Опыт врача во многих случаях с большей точностью позволяет определить заболевание пациента. Конечно, есть вероятность, что врач ошибся, поэтому часто рассматриваются и другие диагнозы.

Байес разработал вероятностную методику, основанную на утверждении, что какое-то событие произойдет, потому что раньше уже произошло какое-то другое событие. В экспертных системах широко применяются статистические решения, опирающиеся на теорию Байеса.

Основные понятия теории вероятности

Теория вероятностей изучает случайные события. Очень часто человек, сам того не замечая, высказывает предположение или делает вывод, пользуясь терминологией теории вероятности.

Вероятность можно определить следующим образом:

$$P = \frac{число\,экспериментов,\,исходом\,которого\,являются\,события}{общее\,число\,экспериментов}, \qquad 0 \le P \le 1$$

Вероятность Байеса

Байес занимался разработкой теории условной вероятности. Условная вероятность учитывает уже известные исходы экспериментов.

Условная вероятность – это вероятность наступления какого-то события *S* при условии, что уже наступило какое-то другое событие *L*.

Условная вероятность обозначается *P(S/L)*.

Вероятность наступления двух событий вычисляется следующим образом:

$P(L \quad и \quad S) = P(S/L) \times P(L)$, т.е.

Вероятность того, что произойдут два события *S и L*, причем *L* произойдет первым, равна вероятности наступления события *S*, если известно, что произошло событие *L,* умноженного на вероятность появления события *L*.

В экспертных системах используется еще одно уравнение условной вероятности:

$$P(S) = P(S/l) \times P(l) + P(S/NOT \quad l) \times P(NOT \quad l) \qquad (1)$$

Вероятность появления события S равна вероятности появления события S при условии появления события l *(P(S/L))* умноженной на вероятность появления события l (P(l)) плюс вероятность появления события S при условии, что событие l не произошло *P(S/NOT l)* умноженная на вероятность, что событие l не произошло *P(S/NOT l)*.

Пример.

Рассмотрим использование условной вероятности на примере правил, описывающих экспертную систему фондовой биржи.

П1: ЕСЛИ проц_ставки==падают,
ТО уровень_цен=растет
П2: ЕСЛИ проц_ставки==растут,
ТО уровень_цен=падает
П3: ЕСЛИ вал_курс_доллара==падает,
ТО проц_ставки=растут
П4: ЕСЛИ вал_курс_доллара==растет,
ТО проц_ставки=падают

Надо определить вероятность повышения уровня цен.

Цель примера не в описании реальной ситуации, а в иллюстрации подхода к решению задачи.

Система, реализующая обратные рассуждения в части ТО правил, будет искать вывод *уровень_цен=растет*. Подойдет правило1 при условии, что *проц_ставки=падают*. Используя (1) условной вероятности, можно оценить эти условия.

Заменяя S на STOCK=растет и l на INT=падают, тогда получим:

$$P(STOCK = PACTET) = P(STOCK = PACTET / INT = ПАДАЕТ) \times P(INT = ПАДАЕТ) + P(STOCK = PACTET / INT = НЕ\_ПАДАЕТ \quad) \times P(INT = НЕ\_ПАДАЮТ) \qquad (2)$$

Для того, чтобы определить, присвоено ли переменной INT значение ПАДАЮТ надо вернуться к правилу 4:

ЕСЛИ DOLLAR=PACTET,
ТО INT=ПАДАЮТ

Правило 4 преобразуется в уравнение (3)

$$P(INT = ПАДАЮТ) = P(INT = ПАДАЮТ / DOLLAR = PACTET) \times P(DOLLAR = PACTET) + P(INT = ПАДАЮТ / DOLLAR = НЕ\_PACTET) \times P(DOLLAR = НЕ\_PACTET) \qquad (3)$$

Поскольку ни в одном из правил части ТО нет переменной DOLLAR, т.е. значение вероятности Р для нее определить нельзя, то значение должно быть введено пользователем. По этой же причине пользователем должна быть задана условная вероятность.

Пусть *P(DOLLAR=PACTET)=0,6*

Согласно теории вероятностей сумма вероятности появления и непоявления какого-либо события равна 1, следовательно

P(DOLLAR=HE_PACTET)=1- P(DOLLAR=PACTET)=1-0,6=0,4

Присвоим значения всем условным вероятностям

P(INT=ПАДАЕТ/DOLLAR=PACTET)=0,8

P(INT=ПАДАЕТ/DOLLAR=HE_PACTET)=0,1

(сумма условной вероятности противоположных событий не равна 1)

Подставим эти значения в (3)

*P(INT=ПАДАЕТ)=0,8*0,6+0,1*0,4=0,52*

P(INT=НЕ_ПАДАЕТ)=1- P(INT=ПАДАЕТ)=1-0,52=0,48

Для того, чтобы найти *P(STOCK=PACTET)* пользователем должны быть заданы значения условной вероятностей.

P(STOCK=PACTET/INT=ПАДАЕТ)=0,85

P(STOCK=PACTET/INT=НЕ_ПАДАЕТ)=0,1

Тогда согласно (2)

P(STOCK=PACTET)*=0,85*0,52+0,1*0,48=0,49* или *49%*

Получив все значения вероятностей, пользователь может определить свою политику на бирже.

2. Порядок выполнения работы*:*

1. Написать правила, использую теорию Байеса;
2. Подсчитать вероятность наступления события;
3. Реализовать «дружественный интерфейс» пользователя;
4. Организовать вывод информации пользователю;
5. Описать программное обеспечение реализации Вашей задачи.

3. Варианты заданий

Описать правила и реализовать программный код, используя теорию Байеса для следующих задач:

1. диагностика неисправностей электронной аппаратуры
2. диагностика неисправностей автомобиля
3. диагностика заболеваний (по выбору)
4. прогнозирование (по выбору)
    a. спортивных мероприятий
    b. телепередач
    c. природных катаклизмов

    и т.п.

5. классификация объектов (по выбору)
6. задачи информационно-советующего характера (по выбору)
    a. помощник заведующего склада
    b. помощник аптекаря
    c. помощник оператора справочной службы
    d. выбор должности
    e. проведение отпуска
    и т.п.

4. Контрольные вопросы

1. Какие данные являются исходными для расчета вероятности наступления события?
2. Какие данные вводятся экспертом?
3. В чём отличие создания правил при использовании теории вероятностей?
4. В чем отличие программной реализации при использовании диалога с пользователем?

Работа № 7

Использование коэффициента уверенности при проектировании интеллектуальных систем с нечеткой логикой

Цель работы: Научиться использовать формулы для вычисления коэффициента уверенности на практических примерах.

1. Теоретическая часть

Не всегда можно описать событие с помощью точно определенных правил. Люди не всегда могут ответить на вопросы точно. Можно ли узнать, какая у человека температура, если он говорит, что слегка заболел? Скорее всего нет! Такие слова, как «высокий», «горячий» и «легкий», «растет» или «падает», представляют собой *лингвистические переменные*, которые нельзя определить одним значением. Использование этих понятий при формировании правил называется *нечеткой логикой*.

Понятие «падает» - также *лингвистическая переменная*, использующаяся в правилах, описывающих фондовую биржу. Применяя *лингвистические переменные* можно вычислить значения некоторых вероятностей, не обременяя пользователя лишними вопросами. Для этого необходимо конкретизировать *лингвистические переменные*. Пользователю экспертной системы нужно позволить добавлять к этим переменным определения, например «маленький» или «средний».

Пользователь может задать маленькое повышение курса доллара и экспертная система должна точно знать, что под этим подразумевается.

КУ используется в области математики, называемой *нечеткой логикой.*. КУ может иметь значение от *-1* до *1*. Отрицательное значение КУ показывает степень уверенности в том, что правило не верно, а положительное значение – что верно. Правила, для которых КУ=-1, рассматривать нет смысла.

Прежде всего сформулируем общие принципы вычисления КУ правила:

1. Выбрать максимальное значение КУ из КУ для условий правила, разделенных логическим оператором И.
2. Если в правиле есть оператор ИЛИ, выбрать максимальное значение из КУ для всех условий правила, разделенных оператором И для всех условий, связанных оператором ИЛИ.
3. Умножить выбранный КУ на КУ правила.
4. Если существует несколько правил с одинаковым логическим выводом, выбрать из всех полученных КУ максимальный.

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, 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, значит, работа с базой знаний продолжается. Если при работе с базой знаний встретился тот же самый логический вывод, 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, выполняется логический вывод, в противном случае, работа с базой знаний продолжается. Вышесказанное можно записать с помощью равенства:

*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).

2. Порядок выполнения работы :

1. Для выбранного варианта написать правила, использую коэффициенты уверенности
2. Реализовать «дружественный интерфейс» пользователя;
3. Организовать вывод информации пользователю;
4. Описать программное обеспечение реализации задачи.

3. Варианты заданий

Описать правила и реализовать программный код, используя теорию нечеткой логики для следующих задач:

1. диагностика неисправностей электронной аппаратуры
2. диагностика неисправностей автомобиля
3. диагностика заболеваний (по выбору)
4. прогнозирование (по выбору)
    a. спортивных мероприятий
    b. телепередач
    c. природных катаклизмов
    и т.п.
5. классификация объектов (по выбору)
6. задачи информационно-советующего характера (по выбору)
    a. помощник заведующего склада
    b. помощник аптекаря
    c. помощник оператора справочной службы
    d. выбор должности
    e. проведение отпуска
    и т.п.

4. Контрольные вопросы

1. Какие переменные называются лингвистическими?
2. В каких случаях используются КУ?
3. В каких пределах изменяется КУ?
4. В чём отличие создания правил при использовании КУ?

Работа № 8
Разработка самообучающихся систем

Цель работы: Научиться разрабатывать алгоритмы и программы реализации самообучающихся систем

1. Теоретическая часть

Какой бы метод ни использовался при обучении, человек всегда пополняет свои знания, сталкиваясь с чем-то новым. Такая форма приобретения знаний называется обратной связью. Человек, благодаря механизму обратной связи и уже, имеющихся знаний, приобретает новые.

Приблизительно также «обучается» компьютерная программа. В ней заложен алгоритм хранения фактов и выполнения логических выводов. Логические выводы связаны в программе с постоянно пополняющими ее новыми фактами.

Система будет самообучаться, только в том случае, если она соприкоснулась с чем-то противоречащим ранее известному.

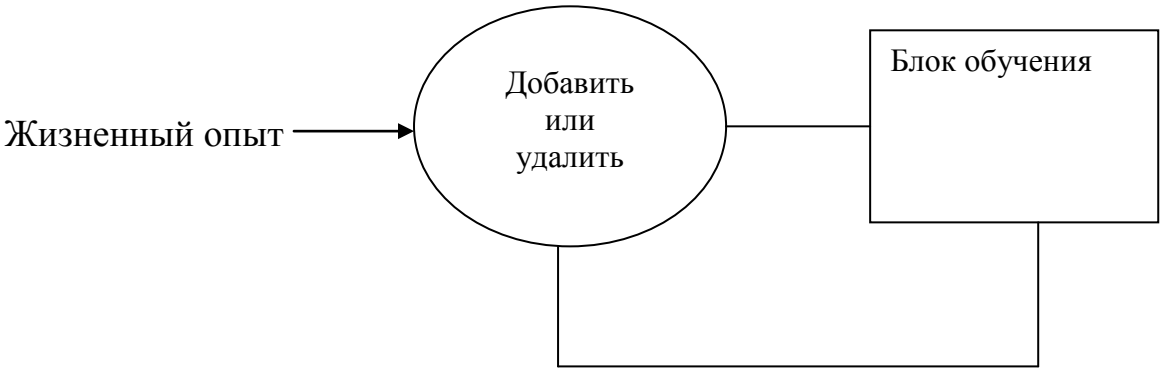

Рис.9. Механизм обучения системы

Основные шаги обучения системы:

1. Система вносит изменения только в том случае, если встречаются противоречия. Это обеспечивается с помощью механизма обратной связи.
2. Система сравнивает введенные атрибуты с атрибутами, хранящимися в ней, пытаясь их идентифицировать и выбрать наиболее похожие. Для этого используются специальные оценки. Вся работа выполняется с помощью логического вывода.

2. Порядок выполнения работы:

1. Выбрать предметную область
2. Создать базу знаний для правил и фактов
3. Реализовать «дружественный интерфейс» пользователя;
4. Организовать вывод информации пользователю;
5. Описать программное обеспечение реализации Вашей задачи.

3. Варианты заданий

Реализовать самообучающуюся систему, используя механизм обучения для конкретных предметных областей:

- География
- История
- Математика
- Физика

и т.д.

4. Контрольные вопросы

1. В чем суть самообучающейся системы?
2. Какая форма приобретения знаний называется обратной связью.
3. Как происходит обучение программы?
4. Каковы основные шаги при разработке обучающейся системы?

Основная литература

1. Гаврилова Т.А., Хорошевский В.Ф. Базы знаний интеллектуальных систем. – С-Пб: Питер, 2001.
2. Змитрович А. И. Интеллектуальные информационные системы.- Мн: ТетраСистемс, 1997.
3. Джексон П. Введение в экспертные системы. - М.: «Вильямс», 2001.
4. Нильсон Н. Принципы искусственного интеллекта. - М.: Радио и связь, 1985
5. Таунсенд К. Проектирование и программная реализация экспертных систем на ПЭВМ. – М. «Финансы и статистика», - 1990. -319с.
6. Левин Р, Дранг Д. Практическое введение в технологию искусственного интеллекта и экспертных систем. М. «Финансы и статистика», - 1990. -235с.

Дополнительная литература

1. Орифжонов М., Бекмурадов Т., Хожиматова Г., “Эксперт системалар”, - Тошкент – “Фан”- 1991й, 60б
2. Корнеев В.В., Гарев А.Ф. и др. Базы данных. Интеллектуальная обработка информации. – М.: «Нолидж», 2000
3. Зыков В.В. Основы информационной культуры. – Тюмень: ТГУ, 1996.

Методические указания к практическим и лабораторным работам по курсу «Интеллектуальные системы»

Составители: доц. Хачатурова Е.М.
ст.преп. Каримова В.А.

Рассмотрено на заседании кафедры «Информационные технологии» ТУИТ (протокол № 25 от __26.02.2008_) и рекомендовано к печати

Рекомендовано для размножения НИС ТУИТ (протокол №_8 от «17_»_апреля_2008 г.

Ответственный редактор: проф. Косимов С.С.

Корректор: