Лекция 1

Э’кспертная система (ЭС) - система ИИ, включающая в себя знания о слабоструктурируемой и слабоформализуемой предметной области, способная осуществлять принятие решений в определённых ситуациях

ЭС используется для решения следующих задач -

1. Данные и целевая функция выражаются в символьной форме, а не числовой
2. Нет однозначного алгоритма решения (решение определяется из цепочки выводов)

Данные характеризуются ошибочностью и неполнотой, что требует оценивать риски ошибки.

Области применения ЭС

1. Диагностика ситуации
2. Интерпретация результатов по описанию ситуации
3. Задача планирования вычислений
4. Использование в контуре управления

Классификация знаний (по Лаврову)

1. Понятийные (или концептуальные) знания. Эти знания определяют систему как интеллектуальную. В отличие от обычных систем обработки, где данные числовые, в интеллектуальных системах обрабатываются концептуальные знания.
2. Процедурные знания. Алгоритмы и библиотеки программ. Если в библиотеках программ используется планирование, то она становится интеллектуальной
3. Фактографические знания. Числовые и качественные характеристики объектов.

Две основные модели систем

1. Продукционная
2. Логическая

Они взаимосвязаны и лежат в основе других. Следующие модели это семантические сети и фреймы (сейчас используются только в гибридных моделях, в которых они объединяются с правилами продукций)

Арихитиктура экспертных систем

Два приницпиальных блока:

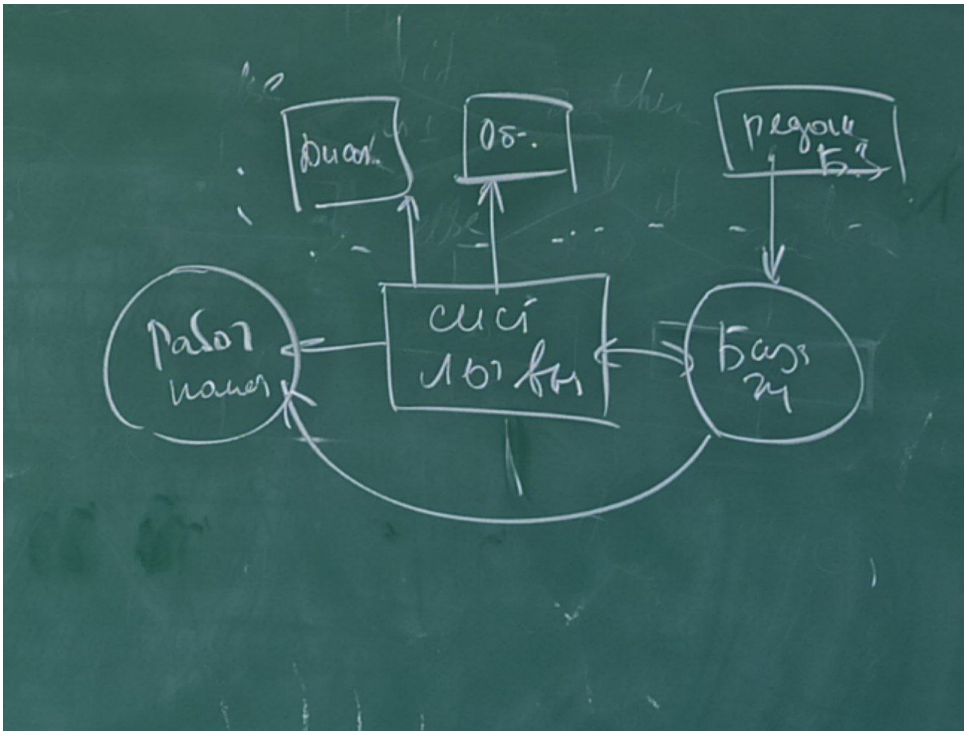
- База знаний (БЗ)
- Система логического вывода (“Решатель”)

Также есть блок рабочей памяти. Рабочая память создаётся в процессе вывода, дополняет БЗ и представляется как текущее решение состояние задачи.

И ещё три блока, которые дополняют:

- Блок наполнения БЗ
- Диа’логовый компонент
- Объяснительный компонент

Архитектура статической экспертной системы.



В динамической БЗ модифицируется в реальном времени. В этом курсе она не рассматривается.

Графы

Методы поиска в графах пространств состояний. При описании экспертных систем используются направленные графы. Ребро соответствует правилу продукции. Введём следующие структуры

- Вершина. Поля:
  - Номер (от 1 до 100)
  - Признак
- Ребро графа. Поля:
  - Начальная вершина
  - Конечная вершина
  - Метка ребра

Граф будем описывать списком рёбер. Последовательность записи определяет ход вывода. Обычно направленный граф описывается рёбрами (вносят в граф в порядке слева-направо)

Основные методы поиска в графах:

- в ширину / в глубину
- прямой / обратный

Будем использовать подход, основанный на обработке списков закрытых и открытых вершин. Вершина будет открыта, если мы не нашли её потомков. Для направленного графа потомки вершины раскрытия или по цели это конечные вершины всех рёбер инцедентных данных ребру.

По другой терминологии это чёрные и белые вершины. В процессе поиска вершина считается серой. Организация метода поиска в ширину и в глубину определяется стратегиями поиска основанного на формировании очереди и стека.

Стратегия стека это метод поиска в глубина. Стратегия очереди это метод поиска в ширину

Метод поиска в глубину (прямой)

Вход: Начальная и целевая вершина. Пространство состояний - начальная вершина одна

Используем стратегию формирования стека. Стек назовём **список открытых вершин**. Начальную вершину помещаем в стек. В процессе поиска мы должны найти **одного первого потомка** и записывать его в стек. Если у вершины нет потомков, то эту вершину надо записать в список закрытых (запрещённых) вершин.

Два метода класса:

1. Метод Найти потомка для текущий подцели - “поиск по образцу”.
2. Метод который организует поиск

Поля класса:

1. Два списка
2. Целевая вершина

Алгоритм поиска по образцу

Цель: найти ребро инцедентное текущей подцели. Если оно найдено - мы сразу выходим из цикла просмотра графа. В результате мы добавляем в голову стека потомка (если найдём), либо определяем что потомков нет и тогда мы просмотрим весь граф. И третий вариант - мы найдём ребро инцедентное с целевой вершиной.

В этом методе просматриваем в цикле while список рёбер. Пока не конец списка и число потомков равно 0. Три условных оператора:

1. Если начальная вершина ребра совпадает с текущей подцелью, конечная является целевой, то флаг решения = 0, число потомков j = 1. Иначе ищем просто инцедентное ребро.
2. Если подцель равна начальной вершине и метка ребра не 0, то конечную вершину записываем в голову стека. j = 1.
3. Возможно, была просмотрена вся база и ничего не нашли и j = 0

Метод в котором организуется поиск. Во всех методах используем два флага.

```
flag y = 0
j = 1
```

В цикле пока оба флага 1 выполнить:

1. Вызываем метод потомки
2. Если в этом методе флаг решения сброшен в 0 ⇒ return, решение найдено
3. Иначе если потомков нет (j=0) и список открытых вершин не пуст мы должны подцель из головы стека переписать в список запрещённых вершин (т.е. вершина не закрывается). Удаляем подцель из головы стека.
4. Иначе если потомков нет и стек пуст, то решения нет, j = 0, выход из цикла. На следующий шаг подцелью будет потомок, которого мы добавили в стек либо если мы его записали в запрещённые, то следующая вершина.