Системы поддержки принятия решений

10. Интеллектуальные системы поддержки принятия решений

Инженерия знаний

- Инженерия знаний – это область информационной технологии, цель которой – превращение знаний, накапливать и применять которые на практике до сих пор мог только человек, в объект обработки на компьютерах. Для этого необходимо проанализировать знания и особенности их обработки человеком и компьютером, а также предложить их машинное представление.
- Цель ИЗ – обеспечить возможность использования информации в компьютерах на более высоком уровне.
- **Что такое знания?**

ПОНЯТИЕ ЗНАНИЯ

- *«Результат, полученный познанием»*
- *«Система суждений с принципиальной и единой организацией, основанная на объективной закономерности»*
- *«Формализованная информация, на которую ссылаются или используют в процессе логического вывода»*

Связь между знаниями и выводом при решении интеллектуальной проблемы

Знания

Информация называется **знанием**, если имеется:

1) внутренняя интерпретируемость (наличие уникального имени, т.е. возможность идентификации каждой информационной единицы);
2) структурированность (наличие гибкой структуры информационных единиц, возможность их рекурсивной вложимости);
3) связность (наличие связей между информационными единицами);
4) семантическая метрика (наличие отношения, характеризующего ситуационную близость информационных единиц, т.е. силу ассоциативной связи между ними);
5) активность (выполнение программ инициируется текущим состоянием информационной базы).

Знания. Внутренняя интерпретируемость

- ***Внутренняя интерпретируемость*.** Каждая информационная единица должна иметь уникальное **имя**, по которому ИС находит её, 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.

Знания. Структурированность

- ***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 её информационные единицы.
  Или: должна существовать возможность произвольного установления между отдельными информационными единицами отношений типа "часть – целое", "род – вид" или "элемент – класс".

Знания. Связность

Связность. 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, информационные единицы могут быть связаны отношением "одновременно", "причина – следствие", "быть рядом" и т.д. Эти отношения характеризуют **декларативные** знания.

Если между двумя информационными единицами установлено отношение "аргумент – функция", то оно характеризует **процедурное** знание, связанное с вычислением определенных функций.

Различают *отношения*

- структуризации (иерархия информационных единиц),
- функциональные отношения (процедурная информация),
- каузальные отношения (причинно-следственные связи),
- семантические отношения (все остальные отношения).

Знания. Метрика и активность

- ***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. 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.
- ***Активность***. 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.

МОДЕЛИ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ

ЛОГИЧЕСКАЯ МОДЕЛЬ

Используется для представления знаний в системе логики предикатов первого порядка и выделения заключений с помощью силлогизма. Пример представления фактов с помощью предикатов (атомарные формулы):

- *СТОЛИЦА(Россия, Москва) :*

  Москва – столица России

- *ЧИНОВНИК(Иванов) :*

  Иванов – чиновник

Пример представления фактов с помощью логических формул:

($\forall x$) [СЛОН (x) $\Rightarrow$ ЦВЕТ(x, СЕРЫЙ)] : все слоны имеют серую окраску

ПРОДУКЦИОННАЯ МОДЕЛЬ

Знания представлены в виде правил вида "ЕСЛИ-ТО".

- Пример:

может_получить_зачет(x) if явл_студентом(x) AND знает_предмет(x).

явл_студентом(x) if сдал_вступит_экзамены(x) AND попал_в_приказ(x).

знает_предмет(x) if ответил_на_вопрос1(x) AND ответил_на_вопрос2(x).

2 диаметрально противоположных типа:

- с прямым выводом (MYCIN, решение задач диагностического характера)
- с обратными выводом (OPS, решение задач проектирования).

МЕХАНИЗМ ВЫВОДА

Задача организации процесса применения правил в определенной последовательности. Существуют два вида механизма вывода, которые основаны:

- на модели продукций;
- на модели логического программирования.

МОДЕЛЬ СИСТЕМЫ ПРОДУКЦИЙ

Правила скомпонованы в **список**. В простейшем варианте происходит просмотр списка до тех пор, пока не будет найдено правило, для которого условие выполнения будет истинным. Тогда оно и выполняется. Далее происходит либо просмотр следующих в списке правил, либо возврат назад. Во втором варианте происходит предварительная выборка готовых к выполнению правил и их компоновка в "конфликтный набор", из которого в соответствии с заданными приоритетами выбирается и исполняется одно из правил.

- Характерной чертой системы продукция является цикл: поиск возбужденного правила - выполнение его.
- Управление в модели СП имеет **линейный** характер.

if (присутствует_хищник) then (спасаться_бегством)

if (необх_спариваться) and (присутствует_партнер) then (спариваться)

if (необх_спариваться) and not(присутствует_партнер) then (искать_партнера)

if (голоден) and not (присутствует_пища) then (искать_пищу)

if (голоден) and (присутствует_пища) then (питаться)

if not(необх_спариваться) and not(голоден) and not(присутствует_хищник) then (ничего_не_делать)

Достоинства и недостатки СП

Недостатки модели СП:

- Затруднены итерация и рекурсия, из чего следует, что они плохо подходят для кодирования стандартных алгоритмов.
- Невозможно статистически проанализировать систему продукций и предсказать ее поведение, откуда следует следующий стиль создания программ: (1) быстро создать систему, (2) испытать ее, (3) продолжить её модификацию (изменить продукцию), пока она наконец не заработает как надо.
- Система продукций работает крайне медленно.
- Неясность взаимных отношений правил.
- Сложность оценки целостного образа знаний.
- Крайне низкая эффективность обработки.
- Отличие от человеческой структуры знаний.
- Отсутствие гибкости в логическом выводе.

Сильные стороны:

- простота создания и понимания отдельных правил;
- простота пополнения и модификации;
- простота механизма логического вывода.

Вывод: СП хорошо работает для небольших задач.

МОДЕЛЬ ЛОГИЧЕСКОГО ПРОГРАММИРОВАНИЯ

- В модели ЛП пытаются использовать высокоуровневое правило путем проверки истинности его предпосылок. Система проверяет истинность таких предпосылок, исследуя, истинны ли предпосылки этих предпосылок и т.п. В Прологе – это процесс поиска в глубину и просмотра слева направо внутри каждого логического предложения. Процесс поиска возвращается назад, если он заходит в тупик.

Пример:

может_получить_зачет(x) if явл_студентом(x) AND знает_предмет(x).

явл_студентом(x) if сдал_вступит_экзамены(x) AND попал_в_приказ(x).

знает_предмет(x) if ответил_на_вопрос1(x) AND ответил_на_вопрос2(x).

ОСНОВНЫЕ МЕХАНИЗМЫ ДЕДУКЦИИ (ЛОГИЧЕСКОГО ВЫВОДА)

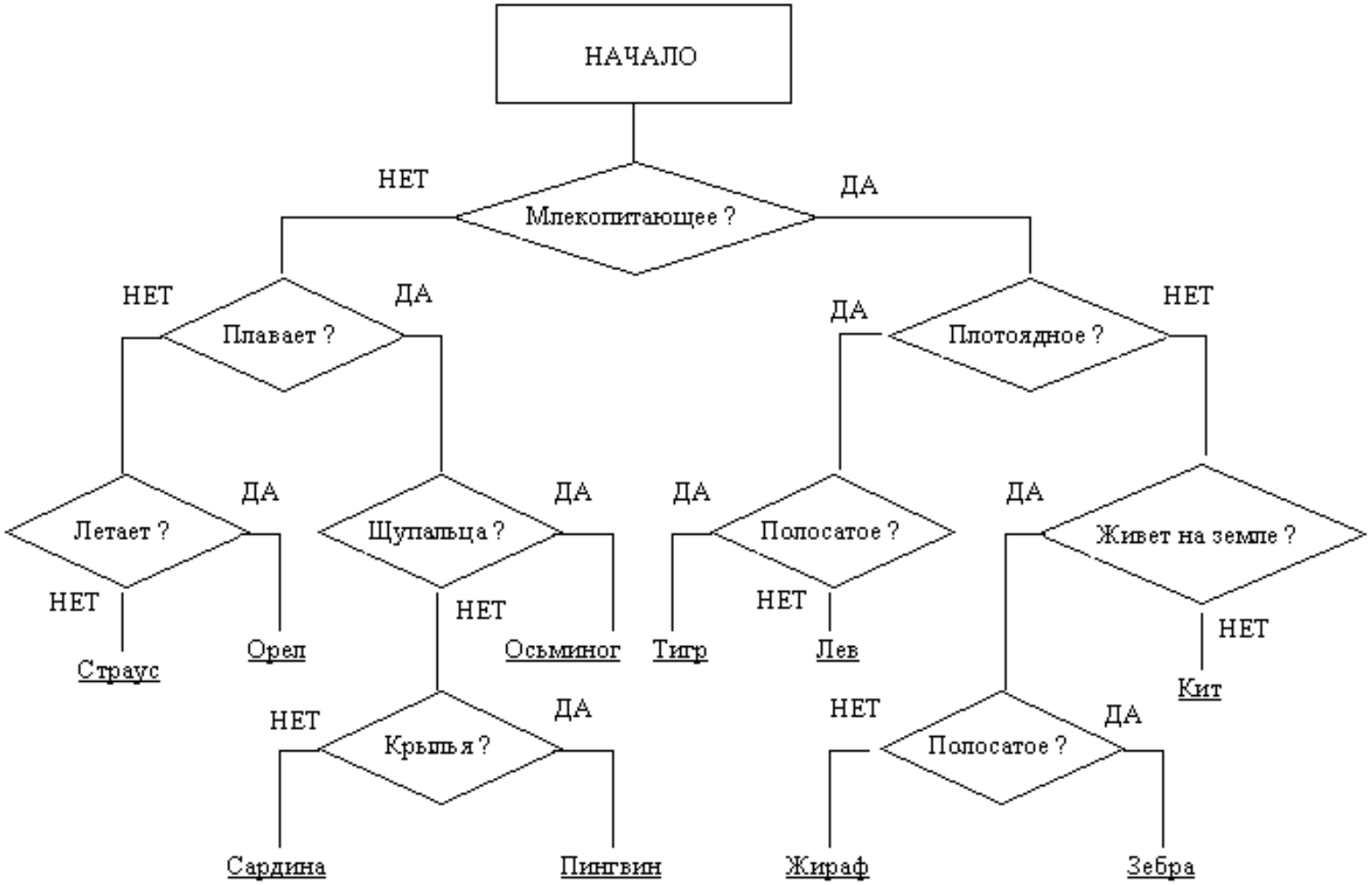
Логический вывод имеет два аспекта:

- Использование рассуждений для нахождения разумных предположений, которые обусловлены имеющимися фактами и правилами (прямая цепочка рассуждений);
- Изучение заключений, которые представляют интерес и могут быть (а могут и не быть) истинными (обратная цепочка рассуждений).

ПРЯМАЯ ЦЕПОЧКА РАССУЖДЕНИЙ

- Суть метода заключается в формировании множества вопросов, позволяющих на каждом шаге отбросить как можно большее число возможных ответов. При этом задаваемые при каждой проверке вопросы целиком зависят от возможных ответов.
- При прямом выводе отправной точкой служат предоставленные данные, причем в качестве заключения (если не все дерево пройдено) используется гипотеза, соответствующая самому верхнему уровню дерева (корню).
- Для такого вывода характерно большое количество данных, а также оценок дерева, не имеющих прямого отношения к заключению (что излишне)

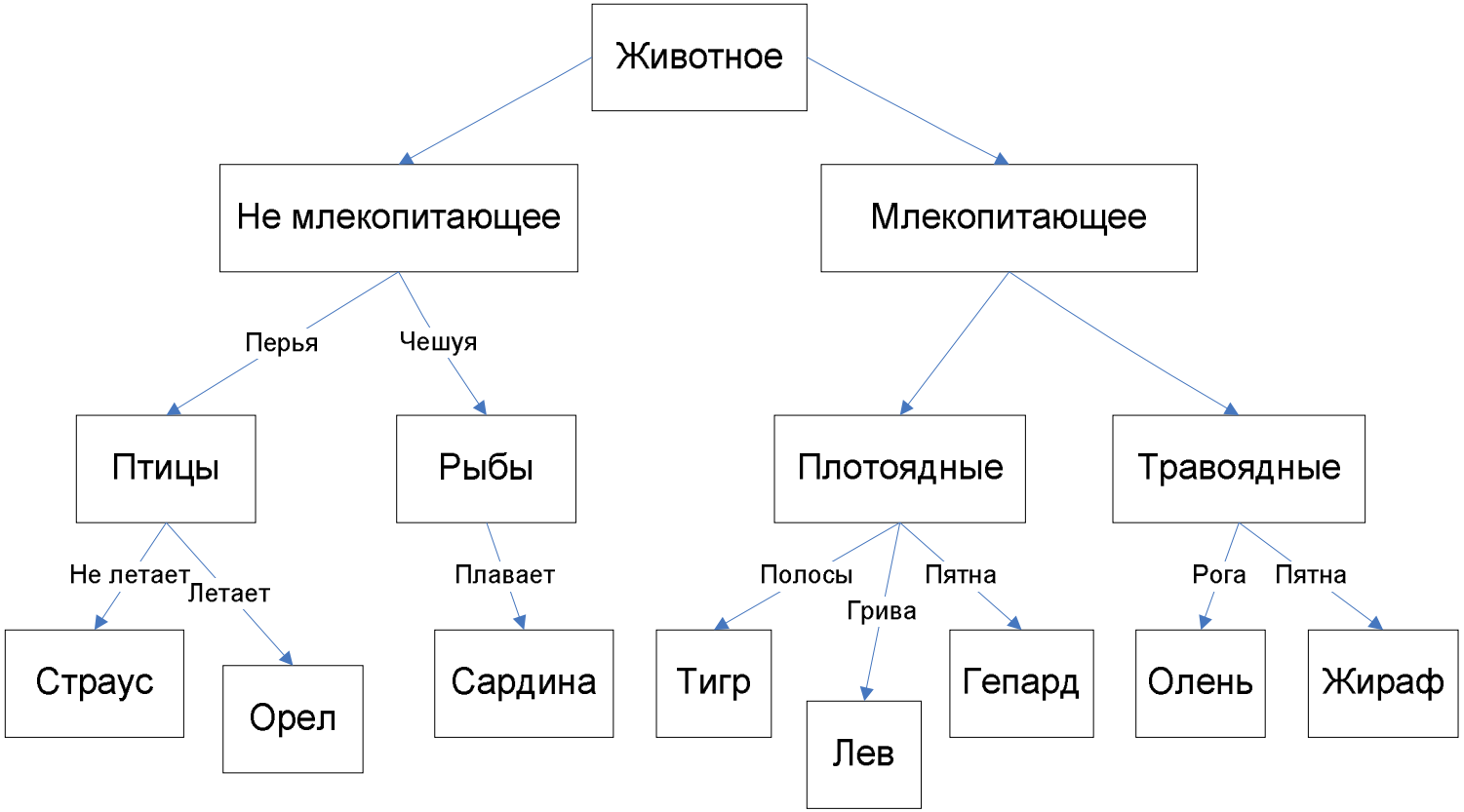
Пример прямой цепочки



ОБРАТНАЯ ЦЕПОЧКА РАССУЖДЕНИЙ

- Начинают с заключения, которое представляет интерес и не является истинным. Механизм вывода определяет все правила, которые приводят к данному факту как к заключению. Затем рассматриваются посылки этих правил. (В Прологе механизм вывода основан именно на обратной цепочке рассуждений)
- Вводится группа правил высокого уровня. Каждое правило описывает одну категорию, четко указывая, какая информация нужна системе, чтобы прийти к выводу, что именно эта категория является искомым ответом. Система пытается по очереди установить истинность или ложность каждого из правил высокого уровня.

Пример обратной цепочки



"Страус" ЕСЛИ ("Не летает") И ("Птица")
"Птица" ЕСЛИ ("Перья") И ("Не млекопитающее")
Начинаем с правила высокого уровня
identify("Страус") :- …
identify("Жираф") :- …

НЕЧЕТКИЕ ЗНАНИЯ

Знания не всегда могут быть описаны точно – часто встречаются так называемые "нечеткие" знания.

В инженерии знаний нечеткости можно классифицировать следующим образом:

- недетерминированность выводов;
- многозначность;
- ненадежность;
- неполнота;
- нечеткость или неточность.

УСЛОВНАЯ ВЕРОЯТНОСТЬ

P(A и B) = P(A|B)·P(B)

Интерпретация импликации с точки зрения теории вероятностей:

Импликация B→A (Если B то A)

Пусть P(B) = 0.9

Уверенность в правиле P(A|B) = 0.95

- P(A) = ?

P(A) = P(A|B)·P(B)+ P(A|¬B)·P(¬B), P(¬B)=1-P(B)

Громоздкость вероятностной схемы для сложных правил.

Дано правило: *Если (A или B), то (C)*
Пусть известно, что
p(C | A и (не B)) = 0.7
p(C | (не A) и B) = 0.8
p(C | A и B) = 0.95
p(A) = 0.8
p(B) = 0.8
p(C) = p(C | A и B) *p(A и B) + p(C | A и (не B)) *p(A и (не B)) +
p(C | (не A) и B) *p((не A) и B) + p(C | (не A) и (не B)) *p((не A) и (не B))
При этом многие из величин нам неизвестны. И мы допускаем, что, например,
p(C | (не A) и (не B) = 0.

ПРИБЛИЖЕННЫЕ РАССУЖДЕНИЯ

Пусть имеется импликация B→A (Если B то A).

Введем понятие **коэффициента уверенности** Ct – субъективной оценки достоверности того или иного факта, наблюдения и т.п. (от 0 до 1)

- Ct(заключение)=Ct(посылка)·Ct(импликация)
- Ct(e1 & e2) = min(Ct(e1),Ct(e2))
- Ct(e1 | e2) = max(Ct(e1),Ct(e2))
- Если 2 правила подтверждают одно заключение, то

$Ct_{R1,R2}$=Ct(R1)+Ct(R2)-Ct(R1)·Ct(R2)

Биполярная схема

- Ct от 0 до 1 имеет один недостаток: если определить отрицание как 1-Ct, то на практике получаются не те результаты, какие ожидались бы.
- Трактовка значения Ct, равного 0.5 как полной неуверенности несостоятельно. Значение 0.5 не является выделенной особой точкой. Лучше выбрать значение Ct, находящееся в интервале, скажем, от –1 до +1.
- Все рассуждения остаются в силе за исключением двух моментов. Во-первых, вычисление Ct для двух подтверждающих правил
- $Ct1>0$ и $Ct2>0$

  $Ct=Ct1+Ct2-Ct1\cdot Ct2$
- $Ct1<0$ и $Ct2<0$

  $Ct=Ct1+Ct2+Ct1\cdot Ct2$
- $Ct1<0$ или $Ct2<0$

  $Ct=(Ct1+Ct2)/(1-\min(|Ct1|,\cdot|Ct2|))$

Обратимые и необратимые правила

- Обратимость правила означает: справедливо ли оно (имеет смысл) при отрицательном значении посылки, т.е. при CtA<0.
- Обратимое правило работает со всеми значениями Ct посылки.
- Необратимые правила работают только при положительных значениях Ct посылки. Если Ct посылки <0, то необратимое правило применять нельзя.

Примеры необратимых правил:

- *Если <грипп>, то <температура высокая>*
- *Если <жираф>, то <есть пятна>*

Если гриппа нет, то это не означает, что у человека не может быть высокой температуры. То же касается и жирафа.

Обратимое правило не должно терять смысл при отрицании посылки и заключения. Пример подобрать сложнее. Обычно правила являются необратимыми.

СЕМАНТИЧЕСКАЯ СЕТЬ

- Семантическая сеть – структура данных, имеющая определенный смысл как сеть.
- Под СС обычно подразумевают систему знаний, имеющую смысл в виде целостного образа сети, узлы которой соответствуют понятиям и объектам, а дуги – отношениям между объектами. Следовательно, всевозможные сети можно рассматривать как сети, входящие в состав семантической сети.
- Формально СС задается в виде

  H = <I, C1, C2, ..., Cn, Г>.

I есть множество информационных единиц;

C1, C2, ..., Cn – множество типов связей между информационными единицами.

Отображение Г задает между информационными единицами, входящими в *I*, связи из заданного набора типов связей.

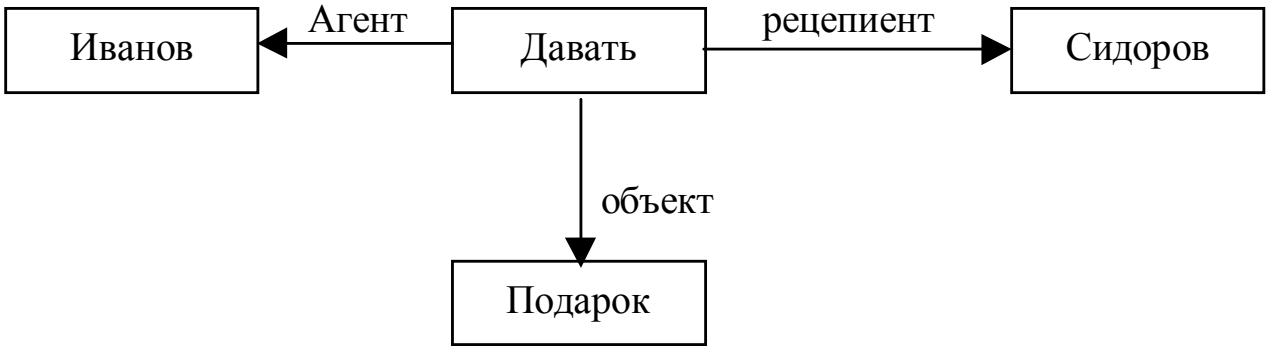
Известно около 200 видов отношений, например таких:

- Род – вид; причина – следствие; вид – род; следствие – причина; целая – часть; часть – целая и т.д..
- Временные отношения, которые можно объединить в группу, например: *протекать параллельно, быть раньше, быть позже*.
- Группа пространственных отношений, например: *над, под, рядом, близко*.

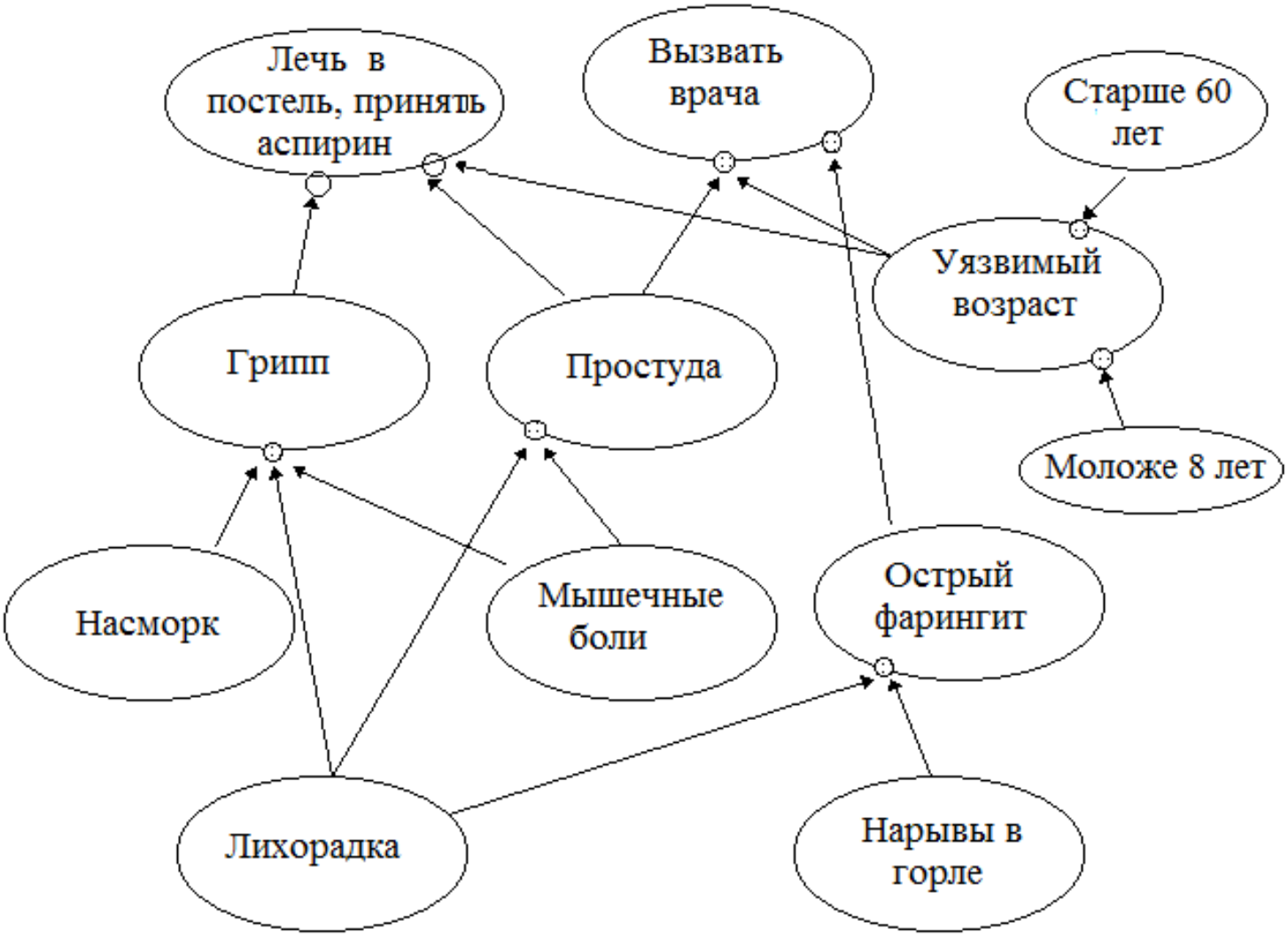
Примеры сетей

"Сократ - человек" "Каждый человек смертен"

"Сократ - смертен"



Еще один пример сети



ФРЕЙМОВАЯ МОДЕЛЬ

- Фреймовая модель (М.Минский) – систематизированная в виде единой теории психологическую модель памяти человека и его сознания.
- Важным моментом в этой теории является понятие фрейма – структуры данных для представления некоторого концептуального объекта.
- Информация, относящаяся к этому фрейму, содержится в слоте (составляющей фрейма). Все фреймы взаимосвязаны и образуют единую фреймовую систему, в которой органически объединены декларативные и процедурные знания.
- *"Фрейм – это структура данных, представляющая стереотипную ситуацию, вроде нахождения внутри некоторого рода жилой комнаты, или сбора на вечеринку по поводу рождения ребенка. К каждому фрейму присоединяется несколько видов информации. Часть этой информации – о том, как использовать фрейм. Часть о том, что следует делать, если эти ожидания не подтвердятся."*

Фреймы и сети

- Фрейм во многом похож на семантическую сеть (семантические сети иногда относят к системам, основанным на фреймах). Фрейм – это сеть узлов и отношений, организованных иерархически, где верхние узлы представляют общие понятия, а нижние – более частные случаи этих понятий.
- Понятие в каждом узле определяется набором атрибутов (слотов). Каждый слот может быть связан с **процедурами** (демонами), которые вызываются при изменении данных.

Например:

- Процедура «если-добавлено». Выполняется при помещении новой информации в слот.
- Процедура «если-удалено». Выполняется при удалении информации из слота.
- Процедура «если-нужно». Выполняется при запросе информации из пустого слота.

Пример фрейма – экземпляра:

(Список работников:

Фамилия (Попов - Сидоров - Иванов - Петров);

Год рождения (1965 - 1946 - 1925 - 1937);

Специальность (слесарь - токарь - токарь - сантехник);

Стаж (5 - 20 - 30 - 25)).

ЭКСПЕРТНЫЕ СИСТЕМЫ

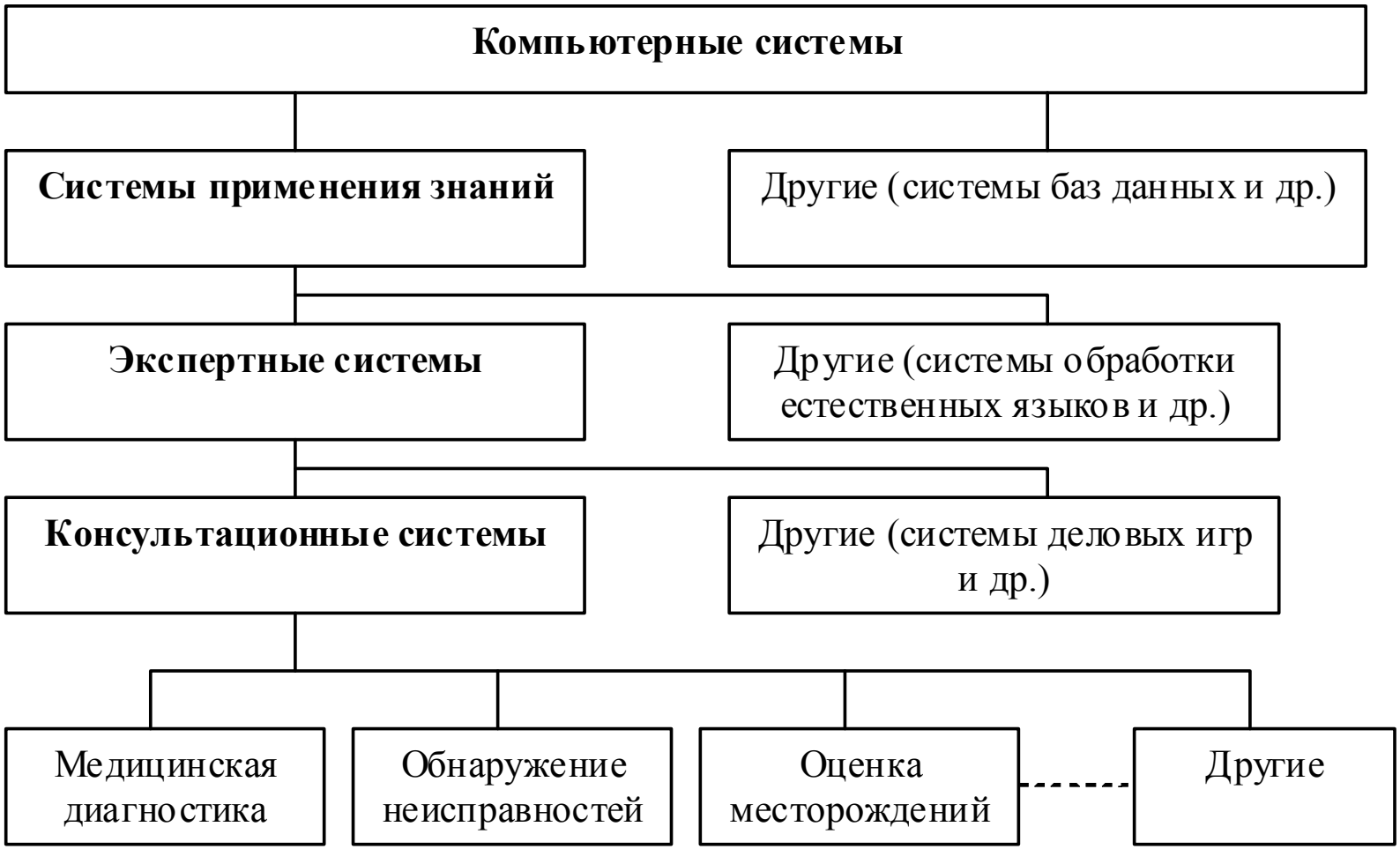
Исторически считалось, что ЭС (или ИЗ в целом) – это одно из направлений ИИ.

ЭС:

- *Система ИИ, созданная для решения задач в конкретной проблемной области*.
- *Программное воплощение специфических знаний и представлений человека-эксперта*.

Место ЭС в компьютерных системах

- ЭС – это разновидность систем, *основанных на знаниях*



Еще раз об определениях

- ЭС – это программа, в которую заложены теоретические и практические знания высококвалифицированных специалистов в некоторой конкретной проблемной области и которая способна давать рекомендации по проблемам в этой области с высокой степенью надежности на уровне этих специалистов.
- ЭС – это формализованное представление знаний о некоторой предметной области, реализованное в виде программно-аппаратного комплекса и опосредованное через некоторый набор математических и методологических процедур.
- *Под ЭС понимается система, объединяющая возможности компьютера со знанием и опытом эксперта в такой форме, что система может предложить разумный совет или осуществить разумное решение поставленной задачи. Дополнительно желаемой характеристикой такой системы, которая многими рассматривается как основная, является способность системы пояснять, по требованию, ход своих рассуждений в понятной для спрашивающего форме.*

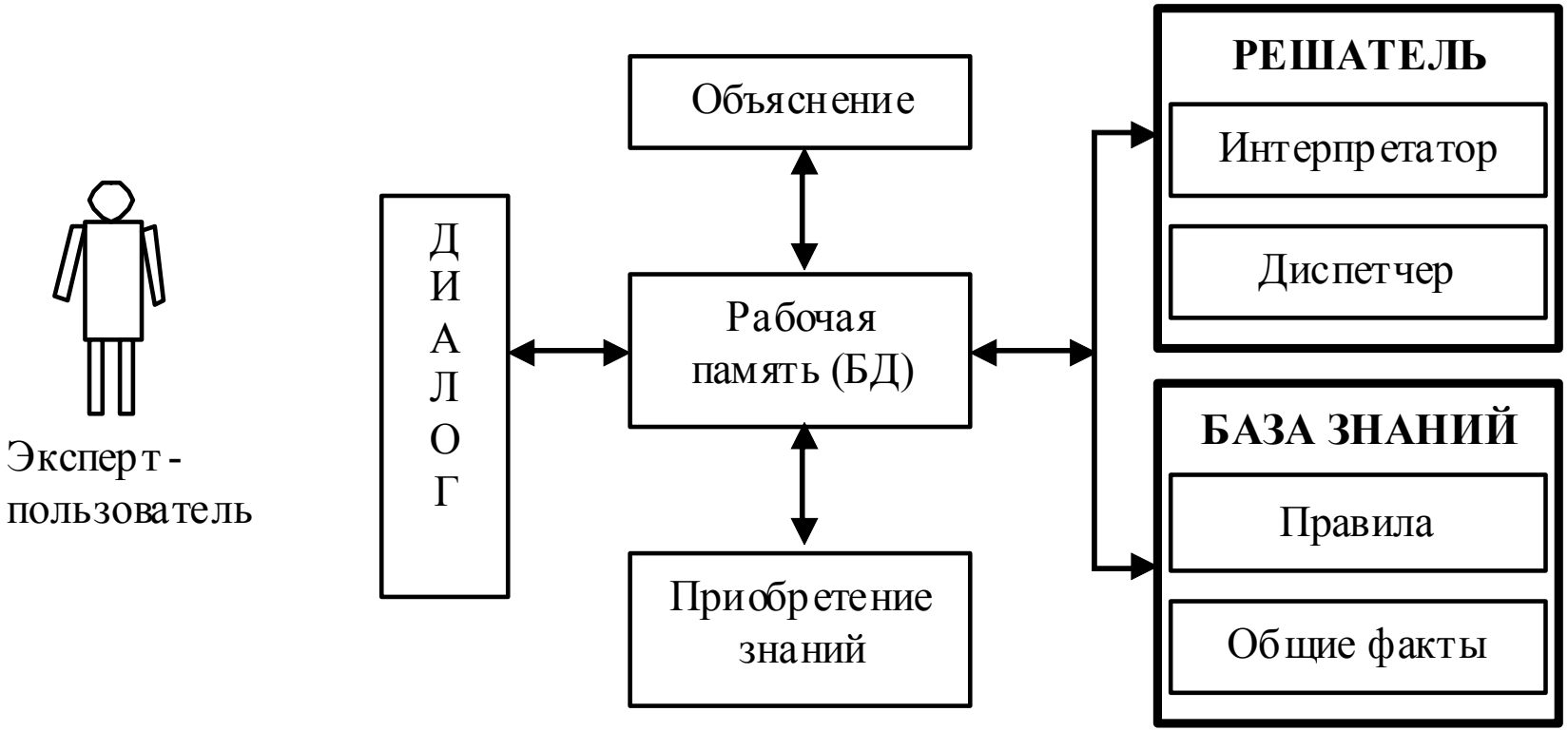
Примечание: ЭС – это не система ИИ. ЭС может включать в себя методы ИИ, не более того.

Отличие ЭС от «традиционных» систем

ЭС не отвергают и не заменяют традиционного подхода к программированию. Они отличаются от традиционных программ тем, что ориентированы на решение неформализованных задач и обладают следующими особенностями:

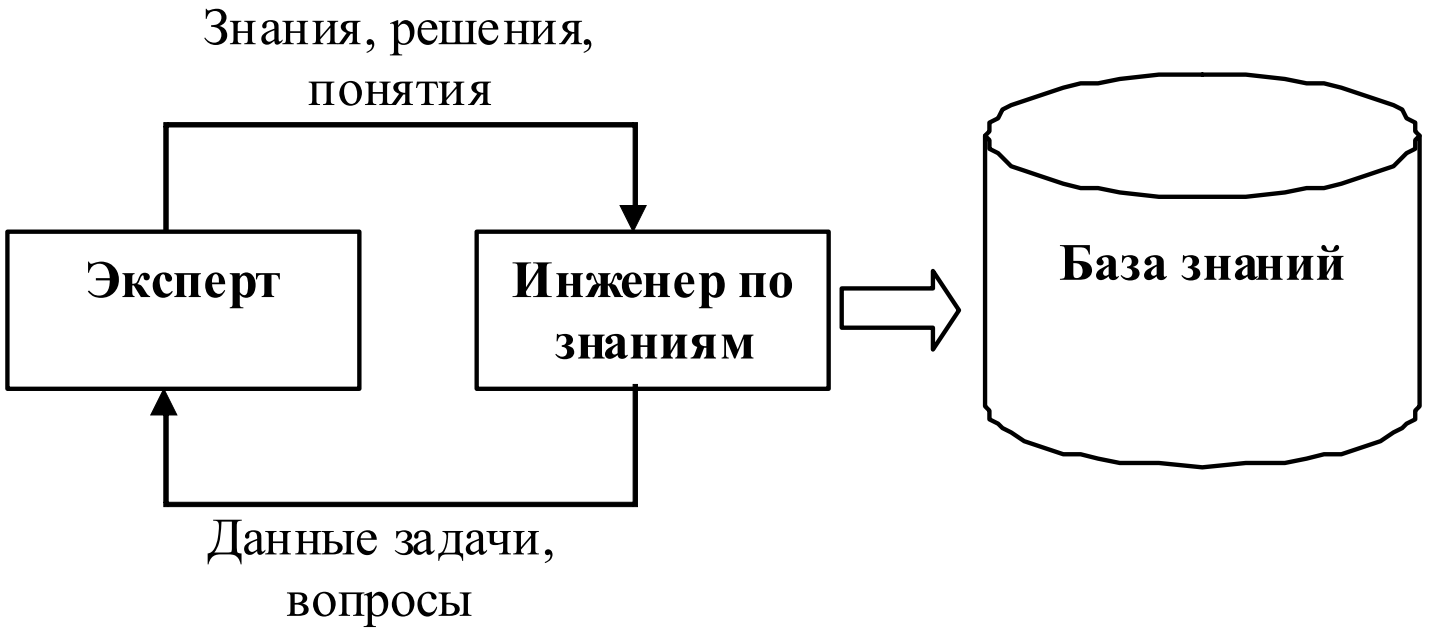
- алгоритм решения неизвестен заранее, а строится самой ЭС с помощью символических рассуждений, базирующихся на эвристических приемах;
- ясность полученных решений, т.е. система "осознает" в терминах пользователя, как она получила решение;
- способность анализа и объяснения своих действий и знаний;
- способность приобретения новых знаний от пользователя-эксперта, не знающего программирования, и применения в соответствии с ними своего поведения;
- обеспечение "дружественного", как правило, естественно - языкового (ЕЯ) интерфейса с пользователем.

СТРУКТУРА И РЕЖИМЫ РАБОТЫ ЭКСПЕРТНОЙ СИСТЕМЫ



ПРИОБРЕТЕНИЕ ЗНАНИЙ

- Взаимодействие инженера по знаниям с экспертом



Некоторые приемы по извлечению знаний из эксперта

Приемы	Описание
1. Наблюдение	Инженер наблюдает (не вмешиваясь) за тем, как эксперт решает реальную задачу
2. Обсуждение задачи	Инженер на представительном множестве задач неформально обсуждает с экспертом данные, знания и процедуры решения
3. Описание задачи	Эксперт описывает типичные задачи для основных типов ответов
4. Анализ задачи	Эксперт решает “вслух” реальные задачи, детализируя ход рассуждений
5. Проверка системы (прототипа)	Эксперт предлагает инженеру перечень задач для решения (от простых до сложных), которые (с использованием приобретенных знаний) перед решением системы инженер решает вручную
6. Исследование системы	Эксперт исследует и критикует правила и механизмы вывода системы
7. Оценка системы	Инженер предлагает другим экспертам оценить решения разработанной системы (прототипа) и решения эксперта, наполнявшего систему

Временные затраты

Для создания экспертной системы требуется от 5 до 10 человеко-лет (для несложной задачи).

- Умеренно трудная (2-4 человека) – 2-7 человеко-лет (0.5-1.5 года при 2-4 разработчиках);
- Трудная (3-5 человек) – 5-15 человеко-лет (1-3 года при 3-5 разработчиках);
- Очень трудная (4-6 человек) –12-30 человеко-лет (3-5 лет при 4-6 разработчиках).

Проект, в котором участвуют меньше 2 человек, работающих полный день, с трудом набирает и поддерживает нужный темп разработки. Если в проекте участвуют больше 6 сотрудников (с полным рабочим днем), то трудно координировать работу, возникают простои.

Типичный проект по разработке ЭС для "несложной" проблемной области:

- Трудоемкость: 6 человеко-лет,
- Время создания: 2 года

Штат	**Занятость (доля раб. времени)**
Старший инженер знаний	0.25
Младший инженер знаний	1.00
Системный программист	1.00
Эксперт	0.75

Итого: специалистов с полным рабочим днем - 3 человека

Фазы разработки или стадии существования ЭС

Бригада разработчиков: один-два инженера по знаниям, один-два программиста в области ИИ и один эксперт.

N	Наименование стадий	Характер стадий	Время работ на стадии	Мощность базы знаний
1	Демонстрационный прототип	демонстрация жизнеспособности основной идеи и алгоритма	1-2 месяца	50-100 правил.
2	Исследовательский прототип	решение всех задач предметной области, но недостаточен ее - вычислительный эффект	3-6 месяцев	200-500 правил
3	Действующий прототип	Надежное решение всей задачи, но для решения сложных задач может потребоваться чрезмерно много времени и/или памяти	6-12 месяцев	500-10000 правил
4	Промышленная система	всестороннее тестирование на реальных задачах, переписывание системы с помощью эффективных инструментальных средств	1-1.5 года	80-1000 правил
5	Коммерческая система	составление подробной документации (5-25 томов)	1-1.5 года	>>1000 правил
Стоимость создания экспертной системы до 1-1.5 млн. долларов				

Примечания:

•на стадии 2 заботимся только о том, чтобы задача решалась в принципе, а не сколько времени тратится на решение.

•на стадии 3 переписывание обязательно, с целью ускорения работы ЭС (выбирается наиболее эффективный язык при решении в этой области).