Национальный исследовательский университет «Ташкентский институт инженеров ирригации и механизации сельского хозяйства»

Мухамедиева Дилноз Тулкуновна

4-тема

Модели представления знаний

Знания – база знаний

- *Знания* – закономерности предметной области

(принципы, связи, законы), полученные в результате практической деятельности в некоторой области, позволяющие ставить и решать задачи в этой области.

Знания – совокупность данных и метаданных, то есть:

- Данные о предметной области
- Указания на смысловое значение этих данных
- Указания на взаимозависимости между
- данными и смысловыми значениями

Особенности знаний

1. Внутренняя интерпретируемость.

Каждая информационная единица должна иметь уникальное имя, по которому система находит ее, а также отвечает на запросы, в которых это имя упомянуто.

2. Структурированность.

Информационные единицы должны обладать гибкой структурой. Для них должен выполняться «ПРИНЦИП МАТРЕШКИ», то есть, рекурсивная вложенность одних информационных единиц в другие.

3. Связность.

В информационной базе между информационными единицами должна быть предусмотрена возможность установления связей различного типа.

Связи могут характеризовать ОТНОШЕНИЯ между информационными единицами.

Семантика отношений может носить ДЕКЛАРАТИВНЫЙ (описательный) или ПРОЦЕДУРНЫЙ характер.

Две и более информационных единицы могут быть связаны ОТНОШЕНИЕМ:

- **«ОДНОВРЕМЕННО» - временная связь (декларативное знание);**
- **«ПРИЧИНА-СЛЕДСТВИЕ» - причинно-следственная (каузальная) связь (декларативное знание);**
- **«БЫТЬ РЯДОМ» - пространственная связь (декларативное знание);**
- **«АРГУМЕНТ-ФУНКЦИЯ»**

Особенности знаний

4. Семантическая метрика

Характеризует ситуационную близость информационных единиц, то есть, силу ассоциативной связи (отношение релевантности) между информационными единицами. Отношение релевантности позволяет находить знания, близкие к уже найденным.

5. Активность

Выполнение программ в Интеллектуальных системах должно инициироваться ТЕКУЩИМ СОСТОЯНИЕМ информационной базы. Появление в базе новых фактов или описаний событий, установление связей могут стать источником активизации системы.

Особенности 1÷5 информационных единиц определяет ту грань,

за которой данные превращаются в знания,

а БАЗА ДАННЫХ перерастает в БАЗУ ЗНАНИЙ.

Классификация знаний

2. Знания как элементы семиотической системы

Знания представляются некоторой знаковой (семиотической) системой.

В любой семиотической системе выделяют три аспекта:

Три типа знания:

- **СИНТАКСИЧЕСКИЕ - характеризуют синтаксическую структуру описываемого объекта или явления, не зависящую от смысла и содержания используемых при этом понятий;**
- **СЕМАНТИЧЕСКИЕ - содержат информацию, непосредственно связанную со значениями и смыслом описываемых явлений и объектов;**
- **ПРАГМАТИЧЕСКИЕ - описывают объекты и явления с точки зрения решаемой задачи.**

3. Процедурные и декларативные знания

МОДЕЛИ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ

1. Эвристические модели (интуиция, опыт и мастерство разработчика)

- **фреймовые модели представления знаний**
- **семантические сети**

2. Логические модели (исчисление предикатов)

- **Логические (Prolog)**
- **Продукционные (CLIPS)**

Понятие фрейма

- **Фрейм** (*frame* — «каркас» или «рамка») — способ представления знаний в искусственном интеллекте, представляющий собой схему действий в реальной ситуации. Первоначально термин «фрейм» ввёл Марвин Минский в 70-е годы XX века для обозначения структуры знаний для восприятия пространственных сцен.

Понятие фрейма

- Фрейм — это модель абстрактного образа, минимально возможное описание сущности какого-либо объекта, явления, события, ситуации, процесса.
- Фреймы используются в системах искусственного интеллекта (например, в экспертных системах) как одна из распространенных форм представления знаний.

Фреймовые модели представления знаний

Теория фреймов (автор Минский):

«Когда человек сталкивается с новой ситуацией (или существенно меняет точку зрения на прежнюю задачу), он извлекает из памяти определенную структуру, называемую фреймом. Эту хранящуюся в памяти систему следует при необходимости привести в соответствие с реальностью путем изменения ее деталей».

Фрейм - это структура данных, предназначенная для представления знаний о стереотипной ситуации, причем детали фрейма с изменением текущей ситуации могут меняться.

Структура фрейма

Фреймовая модель состоит из двух частей:

- **набора фреймов, составляющих библиотеку внутри представляемых знаний,**
- **механизмов преобразования фреймов, их связывания и т.д.**

Общая организация:

имя_фрейма: имя_слота1, значение_слота1
имя_слота2, значение_слота2
…….
имя_слотак, значение_слотак

слоты - это некоторые незаполненные подструктуры фрейма, заполнение которых приводит к тому, что данный фрейм ставится в соответствие некоторой ситуации, явлению или объекту.

Структура фрейма

Незаполненный фрейм (оболочка, образец, прототип фрейма), в котором отсутствуют заполнители слотов, называется протофреймом.

Наличие такой оболочки позволяет осуществить процедуру внутренней интерпретации, благодаря которой данные в памяти системы не безлики, а имеют вполне определенный, известный системе смысл (обладают семантикой).

Протофрейм представляет интенсиональное описание.

Заполненный фрейм (экземпляр, пример прототипа) называется экзофреймом. экзофрейм соответствует экстенсиональному представлению протофрейма.

Структура фрейма

Имя фрейма

Имя_слота	Указатель наследования	Указатель атрибутов данных	Значения слота	Присоединенная процедура
Слот 1				
Слот 2				
…				
Слот к				

Свойства фрейма

1. Каждый фрейм должен иметь уникальное имя в данной фреймовой системе.

2. Фрейм состоит из произвольного количества слотов.

Некоторые из них обычно определяются самой системой для выполнения специфических функций, а остальные - самим пользователем.

Фреймы могут представлять иерархические структуры, то есть реализовывать принцип наследования. Реализация механизма наследования основана на использовании системных слотов. Так, в число системных слотов входит слот, указывающий на фрейм-родитель и слот-указатель на дочерние фреймы.

3. Указатель наследования.

Эти указатели касаются только фреймовых систем иерархического типа. Они показывают, какую информацию об атрибутах слотов во фрейме верхнего уровня наследуют слоты-потомки.

Типичными указателями наследования являются:

- **U (Unique) - уникальный. Фреймы-потомки должен иметь различные уникальные значения этого слота.**
- **S (Same) - такой же. Значение слота у всех потомков должно быть равным значению соответствующего слота фрейма-прародителя.**
- **R (Range) - интервал. Значение слота лежит в некоторых границах.**
- **O (Override) - игнорировать. Одновременное выполнение функций указателей U и S. При отсутствии значения слота у фрейма-потомка этим значением становится значение слота фрейма верхнего уровня (S), но допустимо и указание нового значения слота у фрейма-потомка (U).**

Свойства фрейма

4. Указатель атрибутов (типов) данных.

К возможным типам данных относятся:

a) Литеральные константы - INTEGER, REAL, BOOL, CHAR, …

b) TEXT, LIST (список), TABLE, EXPRESSION, …

c) LISP (присоединенная процедура),

d) FRAME (фрейм)

и др.

5. Значение слота.

a) Тип значения должен совпадать с типом указателя атрибута данного.

b) В качестве значений могут выступать выражения, содержащие обращения к функциям, имена таблиц, списков, других фреймов.

6. Присоединенная процедура.

Выделяют два типа присоединенных процедур - процедуры-слуги и процедуры-демоны. Процедуры-слуги активизируются только при выполнении условий, определенных при создании фрейма.

Процедуры-демоны активизируются при каждой попытке обращения к слоту. Среди разновидностей демонов можно отметить следующие:

- **«ЕСЛИ-НУЖНО» - активизируется, если в момент обращения к слоту его значение не было задано.**
- **«ЕСЛИ-ДОБАВЛЕНО» - запускается при занесении в слот значения.**
- **«ЕСЛИ-УДАЛЕНО» - запускается при стирании значения слота.**

Достоинства фреймовых моделей

1. Представление знаний, основанное на фреймах, дает возможность хранить родовую иерархию понятий в Базе знаний в явной форме.

2. Принцип наследования позволяет экономно расходовать память, проводить анализ

ситуации при отсутствии ряда деталей.

3. Фреймовая модель является достаточно универсальной, поскольку позволяет отобразить все многообразие знаний о реальном мире через:

- **фреймы-структуры, использующиеся для обозначения объектов и понятий (залог, вексель);**
- **фреймы-роли (клиент, менеджер);**
- **фреймы-сценарии (банкротство, собрание);**
- **фреймы-ситуации (авария, рабочий режим устройства);**

и др.

4. С помощью присоединенных процедур фреймовая система позволяет реализовать любой механизм управления выводом.

Недостатки фреймовых моделей

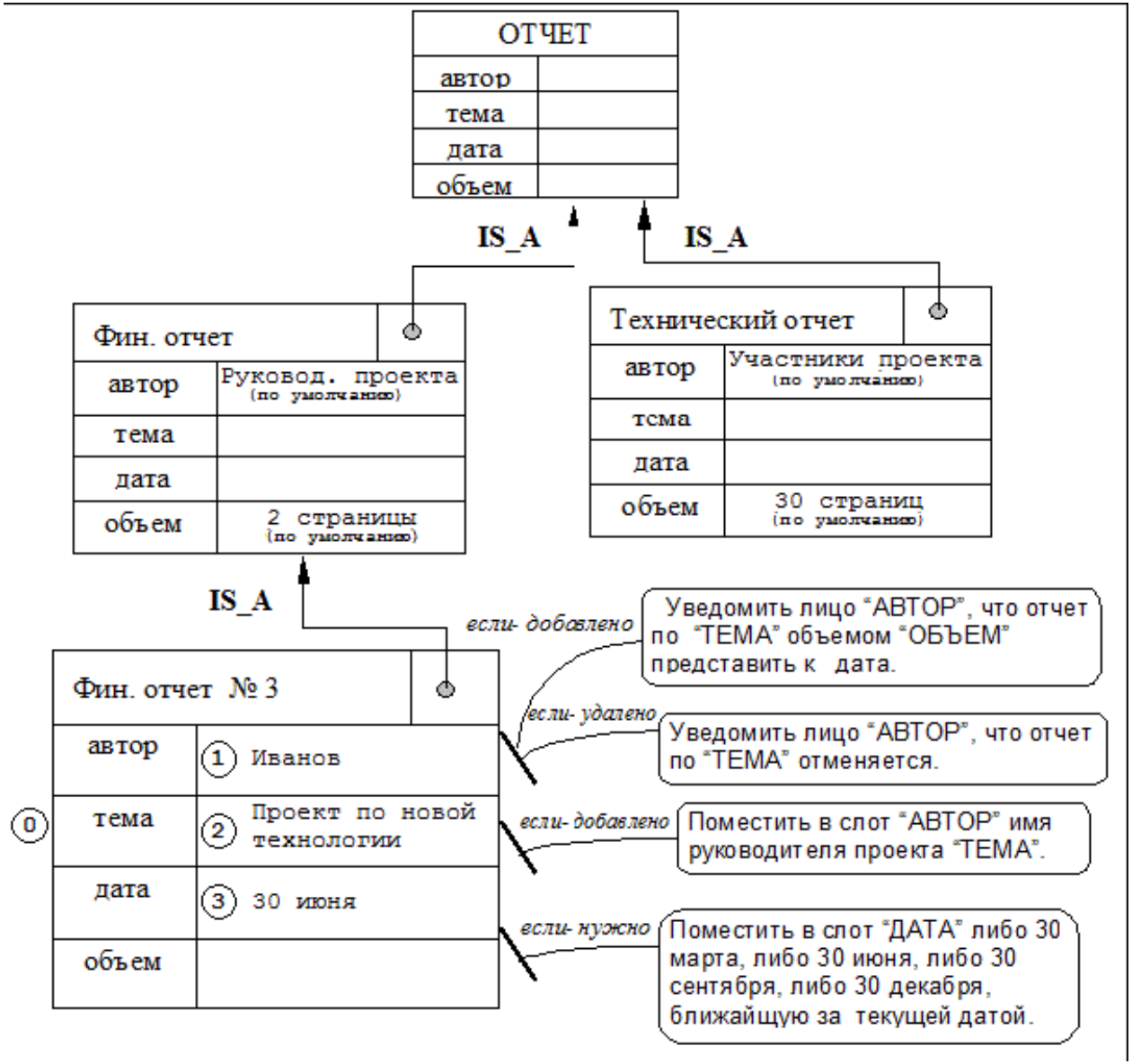
1. Относительно высокая сложность фреймовых систем, что проявляется в снижении скорости работы механизма вывода и в увеличении трудоемкости внесения изменений в родовую иерархию.

2. Во фреймовых системах затруднена обработка исключений. Наиболее ярко достоинства фреймовых систем представления знаний проявляется в том случае, если родовидовые связи изменяются нечасто и предметная область насчитывает немного исключений.

3. Разрозненные части информации, объединенные во фрейм, не могут быть выстроены в последовательность высказываний, иначе говоря, языки описания знаний во фреймовской модели не являются языками, родственными естественным, а ближе к изобразительным средствам.

4. Отсутствует специальный механизм управления выводом, поэтому он реализуется с помощью присоединенных процедур.

Пример реализации фреймовой модели



Пример реализации фреймовой модели

Запрос: «Нужен финансовый отчет о выполнении проекта по новой технологии».

Анализируя структуру модели и на основе фрейма–образца «Фин. отчет» добавим в свою структуру новый пустой фрейм-экземпляр «Фин. отчет №_» (узел №3), а после его создания в слот «ТЕМА» этого фрейм-экземпляра, на основании исходного запроса, внесем текст «Проект по новой технологии».

Далее срабатывают **присоединенные процедуры**:

1. Процедура «ЕСЛИ–ДОБАВЛЕНО», связанная со слотом «**ТЕМА**», осуществляет поиск по своей базе данных руководителя этого проекта (например это Иванов). Процедура вписывает его фамилию в слот «АВТОР» финансового отчета №3. Если руководитель этой темы не будет найден, в слот «АВТОР» будет наследовано значение класса, а именно текст «РУКОВОДИТЕЛЬ ПРОЕКТА».

2. Процедура «ЕСЛИ–ДОБАВЛЕНО», связанная со слотом «**АВТОР**», начинает выполняться, т.к. в слот только что было вписано новое значение. Эта процедура начинает составлять сообщение, чтобы отправить его Иванову, но тут же обнаруживает, что нет нужной даты исполнения.

3. Процедура «ЕСЛИ–ДОБАВЛЕНО», просматривая слот «**ДАТА**» и найдя его пустым, активирует процедуру «если–нужно», связанную с этим слотом. Эта процедура, анализируя текущую дату, например 09.02.17, решит, что «30 июня» ближайшее к ней окончание финансового квартала и впишет эту дату в слот «ДАТА».

4. Процедура «ЕСЛИ–ДОБАВЛЕНО», связанная со слотом «**АВТОР**», найдет, что еще одно значение, которое нужно включить в сообщение, т.е. объем отчета, отсутствует. Слот «ОБЪЕМ» не связан с процедурами и ничем помочь не может. Однако выше узла № 3 существует узел общей концепции финансового отчета, содержащий значение объема. Процедура, используя концепцию наследования свойств класса, использует значение объема и составляет следующее сообщение: «Господин Иванов, подготовьте финансовый отчет по проекту новой технологии к 30 июня объемом 2 страницы».

Пример реализации фреймовой модели

FRL (Frame Representation Language)

```
(frame СТОЛ
      (purpose (value(размещение предметов для деятельности рук)))
      (type (value(письменный)))
      (colour (value(коричневый)))
)
```

KRL (Knowledge Representation Language),
Фреймовая оболочка Карра,

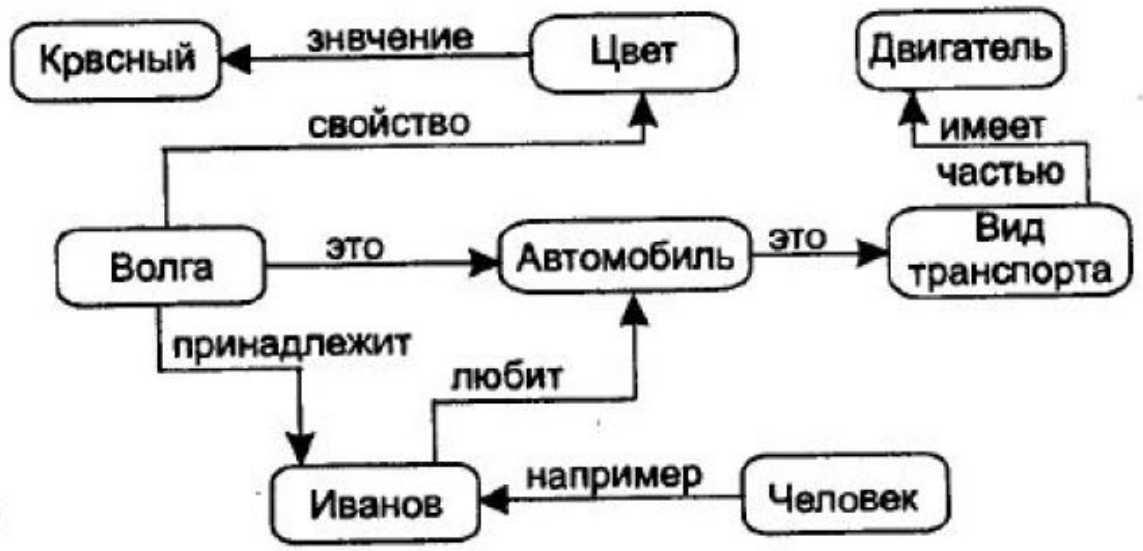
PILOT/2

```
[ Person is_a prototype;
      Name string, if_changed ask_why();
      Age int, restr_by >=0;
      Sex string, restr_by (==”male” || ==”female”), by_default “male”;
      Children {frame} ];
[ John is_a Person; if_deleted bury();
      Name = ”Johnson”;
      Age = 32;
      Children = {Ann, Tom} ];
[ Mary is_a Person;
      without Age;
      Name = ”Smirnova”;
      Sex = ”female”;
      Children = empty ];
```

Определение семантической сети

- Семантическая сеть- ориентированный граф, вершины которого- понятия, а дуги- отношения между ними

Пример семантической сети



Семантическая сеть

СЕМАНТИЧЕСКАЯ СЕТЬ - множество вершин, каждая из которых соответствует определенному **ПОНЯТИЮ, ФАКТУ, ЯВЛЕНИЮ ИЛИ ПРОЦЕССУ**; а между вершинами заданы различные отношения, представляемые дугами.

Вершины помечены именами и описателями, содержащими нужную для понимания семантики вершины информацию.

Дуги также снабжены именами и описателями, задающими семантику отношений.

S = <I, G1, G2,…, GN, R>

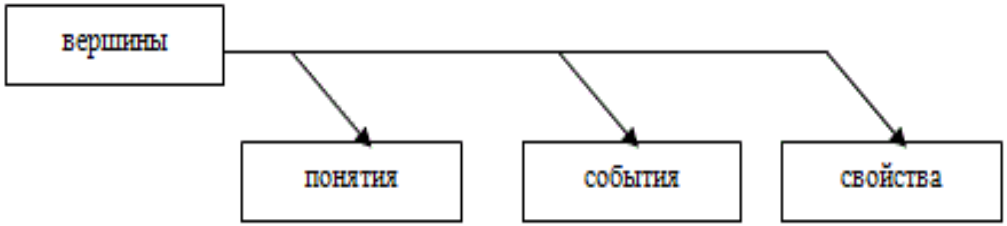
I - множество информационных единиц, представленных вершинами сети;

G1, G2,…,GN - заданный набор типов отношений между информационными единицами;

R - отображение, задающее между информационными единицами, входящими в **I**, связи из заданного набора типов связей.

Элементы семантической сети

Вершины семантической сети



Понятия — представляют собой сведения об абстрактных или физических объектах предметной области или реального мира.

События — представляют собой действия происходящие в реальном мире и определяются:

- указанием типа действия;
- указанием ролей, которые играют объекты в этом действии.

Свойства — используются для уточнения понятий и событий. Применительно к понятиям они описывают их особенности и характеристики (цвет, размер, качество), а применительно к событиям — продолжительность, время, место.

Достоинства и недостатки семантических сетей

Достоинства:

1. Большие выразительные возможности, естественность и наглядность системы знаний, представленной графически.

2. Близость структуры сети, представляющей систему знаний, семантической структуре фраз естественного языка.

3. Данная модель более других соответствует современным представлениям об организации долговременной памяти человека.

Недостатки:

1. Громоздкость и неэффективность представления знаний только аппаратом семантической сети.

2. Сложность организации процедуры поиска нужного знания (как фрагмента сети).

Логические модели

Логическая (формальная) модель представления знаний - **совокупность фактов и правил (утверждений)**.

Факты (правила) представляются как **формулы в некоторой логической системе.**

База Знаний - совокупность формул.

Формулы неделимы и при модификации БЗ могут лишь добавляться и удаляться.

Для представления знаний в математической логике пользуются **исчислением предикатов, которое имеет** ясную формальную семантику и для него разработаны механизмы вывода (**метод резолюций**).

Исчисление предикатов

n-местным предикатом $P(x_1, x_2, ..., x_n)$ называется функция P: $\boldsymbol{M}^n \to \boldsymbol{B}$, где $\boldsymbol{M}$ — произвольное множество, а $\boldsymbol{B} = \{1, 0\}$.

$\boldsymbol{M}$ называется предметной областью, x_i — предметными переменными.

Кванторы

Пусть $P(x_1, x_2, ..., x_n)$ — предикат, определенный на $\boldsymbol{M}^n$. Предметные переменные $x_1, x_2, ..., x_n$ называют *свободными*.

Выражение

$$\forall x_i P(x_1, x_2, ..., x_n)$$

означает: "Для всех $x_i \in \boldsymbol{M}$ предикат $P(x_1, x_2, ..., x_n)$ принимает значение истина".

Выражение

$$\exists x_i P(x_1, x_2, ..., x_n)$$

означает: "Существует $x_i \in \boldsymbol{M}$ такое, что $P(x_1, x_2, ..., x_n)$ принимает значение истина".

Пример предиката

Основное тригонометрическое тождество: $\sin^2(x) + \cos^2(x) = 1$.

Эта формула является теоремой тригонометрии (основное тригонометрическое тождество).

$\forall x$*Равно*(*сумма*(*квадрат*(*sin*(*x*)), *квадрат*(*cos*(*x*))), 1)

Здесь *x* – переменная для обозначения некоторого произвольного числа из множества вещественных чисел, "1" – предметная константа.

Равно – имя предиката, *сумма*, *квадрат*, *sin*, *cos* – имена функций.

Пример программы на Прологе

Предложение - фраза Хорна (Хорновский дизъюнкт), в которой посылки связаны между собой логическим «И», а дизъюнкты- логическим «ИЛИ».

Простейшим типом предложения является факт.

Факт является простым предикатом, который записывается в виде функционального терма

Цель – это средство формулировки задачи, которую должна решать программа

мать(мария, анна).
мать(мария, юлия).
мать(анна, петр).
отец(иван, анна).
отец(иван, юлия).

Вопрос*: Является ли* ***иван дедом петра****, можно задать в виде следующей цели:*
отец(иван, X), мать(X, петр).

Решение: X связывается с константой ***анна***