Модели представления знаний

Лекция 4

Методы представления

- Логические модели
  (в их основе – формальная система)
- Сетевые модели
  (в их основе – семантическая сеть)
- Продукционные модели
  (использует некоторые элементы логических и сетевых моделей)
- Фреймовые модели
  (использует жесткую структуру фиксированных информационных единиц)

Продукционные модели

- под продукцией понимается выражение:

$$(i);\ Q;\ P;\ A{=>}B;\ N$$

где i – имя продукции, с помощью которого данная продукция выделяется из всего множества продукций;

Q – сфера применения продукции;

A=>*B* – ядро продукции;

P – условие применимости ядра.

Классификация ядер

ядро: *A*=>*B*

- ЕСЛИ *A*, ТО ВОЗМОЖНО *B*
- ЕСЛИ *A*, ТО С ВЕРОЯТНОСТЬЮ *p* РЕАЛИЗОВАТЬ *B*
- ЕСЛИ *A*, ТО С БОЛЬШЕЙ ДОЛЕЙ УВЕРЕННОСТИ *B*
- ЕСЛИ *A*, ТО С ВЕРОЯТНОСТЬЮ *p* МОЖНО ОЖИДАТЬ *B*

Представление простых фактов

- Представление – это действие, делающее некоторое понятие воспринимаемым посредством фигуры, записи, языка или формализма
- Представление знаний – формализация истинных убеждений посредством фигур, записей или языков

Синтаксис логики предикатов

- Константы
  - служат именами индивидуумов (в отличие от имен совокупностей): объектов, людей или событий
- Переменные
  - обозначают имена совокупностей, таких как человек, книга, посылка, событие
- Предикатные имена
  - задают правила соединения констант и переменных (предикатные константы)
- Функциональные имена
  - представляют такие же правила, как и предикаты (функциональные константы)

Исчисление предикатов

- Алфавит . Знаки пунктуации ( ) , . ;
- Пропорциональные связи $\neg \cap \cup \supset$
- Знаки кванторов $\forall \exists$
- Символы переменных a_k , $k \in 1,2,3,\ldots$
- Символы функциональных букв P_k^n
- Выражения. Множества выражений строятся над множеством символов алфавита с помощью термов и элементарных формул
- Теоремы. Символы переменной или константы, например $X_1, X_2, \ldots, X_n$ или $f_k^n (X_1, X_2, \ldots, X_n)$.
- Элементарные формулы. Совокупность предикатных букв, термов и знаков пунктуации, например $P_k^n (X_1, X_2, \ldots, X_n)$.
- Правильно построенные формулы. Совокупность элементарных формул и пропорциональных
- связок, например $P_k^n (X_1, X_2, \ldots, X_n) \supset P_l^n (X_1, X_2, \ldots, X_n)$.

Примеры применения

Жак посылает книгу Мари

- Посылка (Жак_2, Мари_4, Книга_22)

Каждый человек прогуливается

- $\forall$ (Человек(x) $\supset$ Прогуливается(x))

Некоторые люди прогуливаются

- $\exists$ x (Человек(x) $\cap$ Прогуливается(x))

Ни один человек не прогуливается

- $\neg$ ($\exists$ x (Человек(x) $\cap$ Прогуливается(x)))

Фреймовые модели

- фиксирует жесткую структуру информационных единиц, называемых протофреймом

$$\{ n, (V_1, G_1, P_1), (V_2, G_2, P_2), \ldots , (V_k, G_k, P_k)\}$$

где n - имя фрейма; V_j - имя слота;

G_j - значение слота; P_j – процедура

- заполняя значения слотов от протофрейма переходим к фрейму экземпляру

Фреймы-экземпляры

(Список работников:	
Фамилия	(значение слота 1);
Год рождения	(значение слота 2);
Специальность	(значение слота 3);
Стаж	(значение слота 4).

(Список работников:	
Фамилия	(Попов - Сидоров - Иванов - Петров);
Год рождения	(1965 - 1946 - 1925 - 1937);
Специальность	(слесарь - токарь - токарь - сантехник);
Стаж	(5 - 20 - 30 - 25).

Модель представления знаний

- фреймы соответствуют понятиям, отражающим объекты, явления, характеристики предметной области
- модель представления знаний:
  - набор фреймов, образующих библиотеку внутреннего представления знаний
  - механизм их преобразования, связывание и т. д.
- содержит как информационные, так и процедуральные элементы

Два типа фреймов

- профрейм - это интенсиональное описание некоторого множества фреймов-примеров
- фрейм-пример - это экстенсиональное описание соответствующего фрейма-прототипа

< фрейм > ::= [< имя >] < ссылка на прототип > < слот > {< слот >} < слот > ::= < имя слота > < значение > [{< процедура >}]

Процедура в фрейме

< ДАТА > (<МЕСЯЦ> <ИМЯ>**) (<ДЕНЬ> <** целые числа {1,2,...31} **>) (<ГОД>** <функция>**) (<ДЕНЬ НЕДЕЛИ> <**перечень{ПНД,ВТР,...ВСК}**>** <функция>**)**

< ISA ДАТА > (< МЕСЯЦ >< май >)(< ДЕНЬ >< 13 >)

Процедуры-демоны активизируются автоматически каждый раз, когда данные попадают в соответствующий фрейм-пример или удаляются из него

Процедуры-слуги активизируются только по запросу

Фрейм – ориентированный граф

- Одна из вершин выделена для предикатного (функционального) символа, остальные - для аргументов выделенного символа. Для каждой аргументной вершины задана область допустимых значений, что позволяет интерпретировать данную вершину как слот.

Процессы обработки

- информационно-вычислительный процесс организуется пользователем с привлечением какого-либо языка программирования
- для систем фреймов вводится единый информационно-вычислительный процесс, основой которого является выбор фреймов, управляющих дальнейшими вычислениями
- выделяются подклассы систем фреймов, для которых разрабатываются специфические алгоритмы, опирающиеся на индивидуальные свойства этих подклассов

Теория формальных грамматик

- Пусть **V** - непустое конечное множество символов (элементов), называемых словарем (алфавитом)
- Произвольную конечную последовательность символов □ называют *цепочкой* в *словаре* **V** (цепочки получаются с помощью операции *конкатенации* - соединения)
- Множество всевозможных цепочек в словаре **V** называют *языком*
- Различают *распознающие*, *порождающие* и *преобразующие* грамматики

Порождающие грамматики

$$G = (T, N, P, S)$$

- где T - конечное непустое множество символов, называемых терминальным (основным) словарем;
- N - конечное непустое множество символов, называемых нетерминальным (вспомогательным) словарем;
- P - конечное непустое множество правил вывода (правил переписывания, продукций);
- S - начальный символ

Пример

Пусть грамматика задана следующим образом:

T = { a , b } , N = { S ,A , B } , S = S ,

P = { 1. **S → aB ;** 2. **S → bA ;** 3. **A → aS ;**

4. **A → bAA ;** 5. **A → a ;** 6. **B → bS ;**

7. **B → aBB ;** 8. **B → b }.**

Типичные выводы предложений :

1) **S → aB → ab**
2) **S → aB → abS → abbA → abba**
3) **S → bA → ba**
4) **S → bA → bbAA → bbaA → bbaa**

Пример

Пусть грамматика задана следующим образом :

T = { усиливает, генерирует, сигнал, импульс, прибор, электронный, полупроводниковый, электронно-полупроводниковый **} ;**

N = { С - (сказуемое), П - (подлежащее), О - (определение), Д - (дополнение), ГП - (группа подлежащего), ГС - (группа сказуемого), ПР - (предложение) **} ;**

S = { ПР - (предложение) **} ;**

P = { 1. ПР $\rightarrow$ (ГП) (ГС) ; 2. ГП $\rightarrow$ (О) (П) ; 3. ГС $\rightarrow$ (С) (Д) ;

4. О $\rightarrow$ (электронный, полупроводниковый, ЭПП) ;

5. П $\rightarrow$ (прибор) ; 6. С $\rightarrow$ (усиливает, генерирует) ;

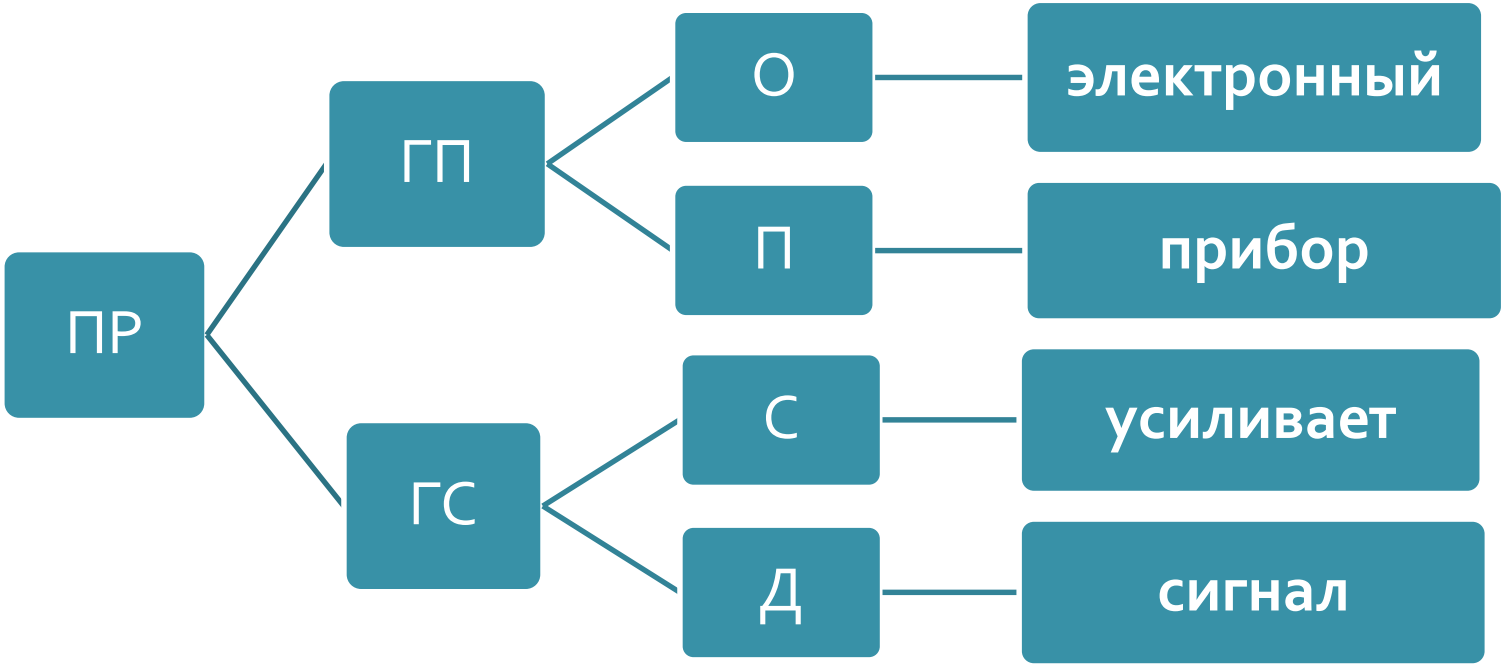
7. Д $\rightarrow$ (сигнал, импульс) ; **}**

Пример

- ПР → (ГП)(ГС) → (О)(П)(ГС) → (О)(П)(С)(Д) → электронный(П)(С)(Д) электронный(П)(С)(Д) → электронный прибор(С)(Д) → электронный прибор усиливает(Д) → электронный прибор усиливает сигнал
- Аналогично можно вывести цепочку: “полупроводниковый прибор генерирует импульс” и ряд других

Структурное дерево

- Каждая вершина дерева имеет в качестве метки символ из V;
- корень дерева, т.е. вершина, в которую не входит ни одна дуга, имеет метку S;
- если вершина с некоторой меткой D имеет хотя бы одну подчиненную вершину, то **D** $\in$ **N**;
- если некоторые n вершин с метками D1,D2,…Dn подчинены непосредственно вершине с меткой D, то в P существует правило D $\rightarrow$ D1…Dn.


ПР
ГП
ГС
О
П
С
Д
электронный
прибор
усиливает
сигнал

Типы формальных грамматик

- тип 0 - это грамматики, в которых не накладывается никаких ограничений на правила вывода $\varphi \to \psi$, где φ и ψ могут быть любыми цепочками из **V**
- типа 1 - это грамматики, в которых содержатся правила $\varphi \to \psi$, удовлетворяющие условиям $\varphi = \varphi_1 A \varphi_2$, $\psi = \varphi_1 \omega \varphi_2$, где A - нетерминальный символ, а ω, φ, φ_1, φ_2, ψ - цепочки из **V**

Типы формальных грамматик

- тип 2 - это грамматики, в которых допустимы лишь правила вида **A** $\rightarrow \omega$, где **A** $\in$ **N** , а ω - любая непустая цепочка из **V**
- тип 3 имеют правила вида **A** $\rightarrow$ **aB** либо **A** $\rightarrow$ **b**, где **A,B** $\in$ **N ;** а **b** $\in$ **T**. Заметим, что **A,B,a,b** являются одиночными символами (не цепочками) соответствующих словарей

Типы формальных грамматик

- *Неукорачивающие грамматики* - это грамматики, в которых для любого правила $\varphi \to \psi$ справедливо соотношение $|\varphi| \leq |\psi|$
- Грамматика *непосредственно-составляющих* имеет правила вида $\varphi A \psi \to \varphi \omega \psi$ или $A \to \omega$, где $A \in N$; ω, φ, ψ - произвольные цепочки
- Грамматики, использующие правила перехода от одних выражений к другим, несущим ту же информацию, называются *трансформационными*

Трансформационная грамматика

- Первая компонента - НС – грамматика
- Вторая (Т – компонента) образована правилами трансформации $p_1,\ldots p_n \rightarrow l$, где $n \geq 1$, а $p_1,\ldots p_n, l$ - не цепочки символов, а С – маркеры

Таким образом, вывод в трансформационной грамматике есть отображение набора

С - маркеров в новый С – маркер, представляющий поверхностную структуру предложения

Цикл лекций подготовлен в 2011/2012 уч. году Кузнецовым Игорем Ростиславовичем, доцентом кафедры радиоэлектронных средств Санкт-Петербургского Государственного электротехнического университета «ЛЭТИ»

Прочитан в дисциплине
«Информационные технологии»

© Кузнецов И.Р.