Представление знаний для решения интеллектуальных проблем

Концепция

- База знаний описывается как отображение объектов и отношений предметной области в вычислительные объекты и программные отношения;
- Результаты вывода в базе знаний должны соответствовать результатам действий или наблюдений в мире (в предметной области).

Формальные системы

- Формальная постановка задачи – выражение знаний о среде, необходимых для решения задачи, на некотором формальном языке;
- Синтаксис языка определяет допустимые в языке представления, состоящие из цепочек символов некоторого множества – алфавита;
- Семантика языка – определяет смысл предложений, сопоставляя символам объекты реального мира, а предложениям – отношения между объектами.

К формальным системам можно отнести языки программирования, рассматривая программы как описание некоторых систем реального мира. Однако императивные языки программирования (C++, Java и проч.) не определяют общие правила оперирования моделью – необходимо не просто описать некоторую систему на языке программирования, нужно также реализовать еще и действия с этой моделью – собственно алгоритмы программ.

Логическое исчисление

Логическое исчисление, или просто исчисление, задано, если заданы:

1. Алфавит – совокупность используемых символов;
2. Синтаксис – правила построения формул на основе алфавита;
3. Аксиомы – множество общезначимых исходных формул;
4. Правила вывода по аксиомам производных формул или теорем.

Пропозициональная логика (логика нулевого порядка)

Грамматика высказываний пропозициональной логики:

- *Sentence* – *AtomicSentence* | *ComplexSentence*
- *AtomicSentence* – **True** | **False** | *Symbol*
- *Symbol* – P | Q | R | …
- *ComplexSentence* – ¬*Sentence* |
  (*Sentence* ^ *Sentence*) |
  (*Sentence* v *Sentence*) |
  (*Sentence* ⇔ *Sentence*) |
  (*Sentence* ⇒ *Sentence*)

Формально True и False в синтаксис не входят, однако их часто используют. Также часто используются следующие символы: = (равенство), ≡ (тождество), ⊕ (исключающее или), при этом они

Понятие о логическом выводе

- Задана некоторая база знаний, определяющая модель – Knowledge Base (KB). Как правило, база знаний представлена в виде набора высказываний.
- Задача вывода – определить, является ли истинным KB ⊢ Z (из KB следует Z) для некоторого заданного высказывания Z.
- Простейший способ – перебрать все комбинации значений переменных, на которых все аксиомы KB истинны, и для каждого случая установить, истинно ли Z. Если всегда истинно, то Z следует из KB.
- Требования к алгоритмам вывода – непротиворечивость и полнота. *Каждый известный алгоритм логического вывода для пропозициональной логики в худшем случае имеет сложность, экспоненциально зависящую от размера ввода.*

Принципы

- Логическая эквивалентность – эквивалентность двух высказываний на множестве моделей;
- Логическая допустимость – высказывание истинно во всех моделях (тавтология);
- Выполнимость – существуют модели, в которых высказывание истинно.

Стандартные логические эквивалентности (тавтологии)

$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$	коммутативность связки $\wedge$
$(\alpha \vee \beta) \equiv (\beta \vee \alpha)$	коммутативность связки $\vee$
$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$	ассоциативность связки $\wedge$
$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$	ассоциативность связки $\vee$
$\neg(\neg \alpha) \equiv \alpha$	устранение двойного отрицания
$(\alpha \Rightarrow \beta) \equiv (\neg \beta \Rightarrow \neg \alpha)$	контрапозиция
$(\alpha \Rightarrow \beta) \equiv (\neg \alpha \vee \beta)$	устранение импликации
$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$	устранение двухсторонней импликации
$\neg(\alpha \wedge \beta) \equiv (\neg \alpha \vee \neg \beta)$	правило де Моргана
$\neg(\alpha \vee \beta) \equiv (\neg \alpha \wedge \neg \beta)$	правило де Моргана
$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$	дистрибутивность связки $\wedge$ по $\vee$
$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$	дистрибутивность связки $\vee$ по $\wedge$

Правила вывода

1. Modus ponens (правило отделения): «*если истинна формула A и истинно, что из A следует B, то истинна и формула B*»;
2. Modus tollendo tollens: «*Если из A следует B и B ложно, то и A ложно*»;
3. Modus ponendo tollens: «*Если A и B не могут одновременно быть истинными и A истинно, то B ложно*».
4. Modus tollendo ponens: «*Если либо A, либо B является истинным и A не истинно, то B истинно*».

Правила вывода (продолжение)

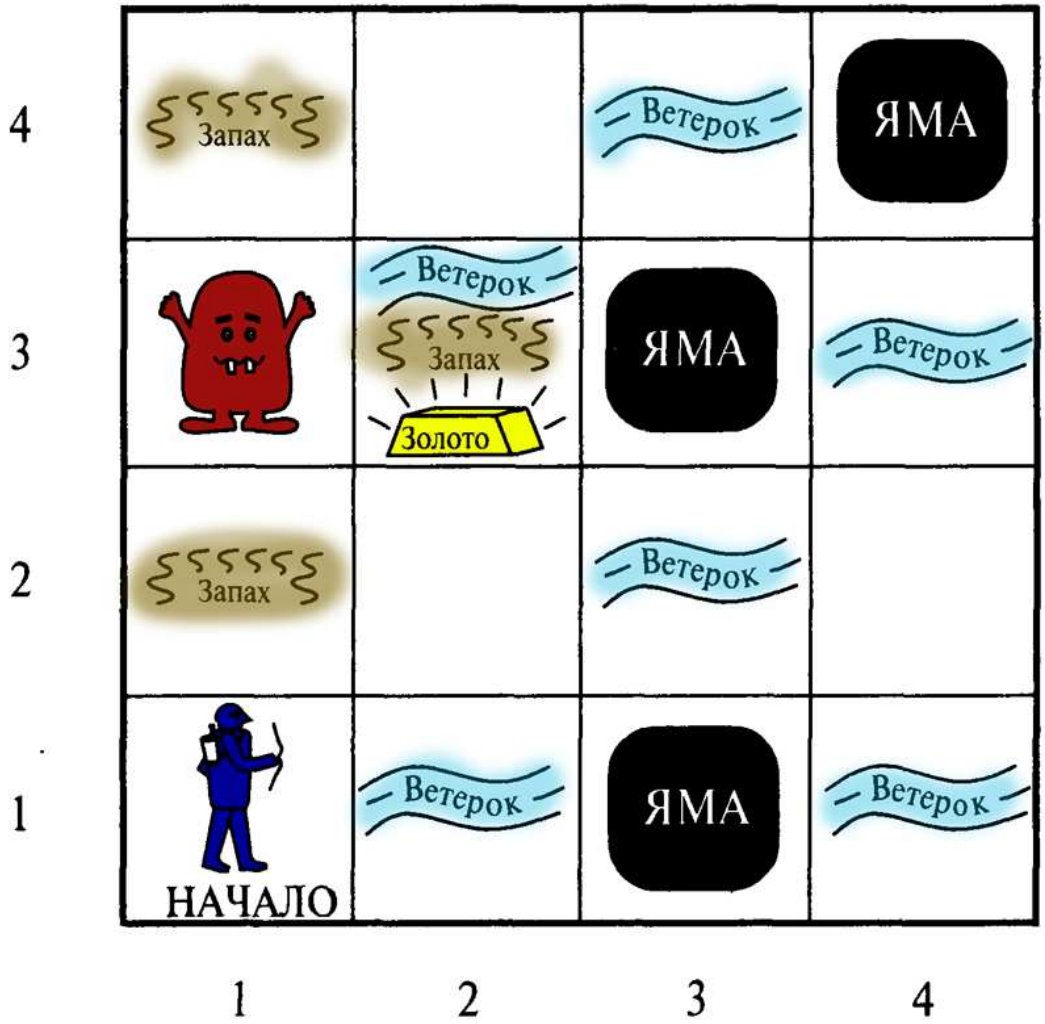
5. Правило силлогизма: $\dfrac{A \to B, B \to C}{A \to C}$

6. Закон контрапозиции: $\dfrac{A \to B}{\overline{B} \to \overline{A}}; (A \to B) \to (\overline{B} \to \overline{A})$

7. Закон исключения третьего: $\dfrac{A \vee \overline{A}}{истина}$

8. Закон «гибельная дилемма»: $\dfrac{A \to B, B \to D, \overline{C} \vee \overline{D}}{\overline{A} \vee \overline{B}}$

Мир Вампуса



Игра написана Грегори Йобом (Gregory Yob) в 1972 году, предложена Майклом Генезеретом (Michael Genesereth) как испытательная среда для интеллектуальных агентов.

Одно из текущих состояний

После нескольких шагов определено множество аксиом:

$$R_1 : \neg P_{1,1}$$

$$R_2 : B_{1,1} \Leftrightarrow P_{1,2} \vee P_{2,1}$$

$$R_3 : B_{2,1} \Leftrightarrow P_{1,1} \vee P_{2,2} \vee P_{1,3}$$

$$R_4 : \neg B_{1,1}$$

$$R_5 : B_{2,1}$$

Аксиомы R2 и R3 описывают общие факты (верны для любой игровой конфигурации), остальные правила описывают текущую конфигурацию.

Принципы резолюции

- Рассматриваем два высказывания в КНФ, содержащие пару противоположных литералов;
- Выполняем «склеивание» высказываний, удаляя противоположные литералы;
- Выполняем факторизацию - удаляем копии литералов.

Полученное выражение называется дизъюнктом, оно помещается в множество аксиом.

Алгоритм резолюции в пропозициональной логике

1. Добавить к множеству аксиом отрицание доказываемого факта;
2. Привести KB к КНФ;
3. «Обнулить» NewSet – множество резольвент;
4. Для каждой пары (Ci, Cj) аксиом из KB:
   1. Построить множество резольвент RSet;
   2. Если RSet содержит пустой дизъюнкт – выход (успех);
   3. Добавить RSet к NewSet;
5. Если NewSet подмножество KB – выход (неудача), Иначе перейти на шаг 2

Логика первого порядка

Логика первого порядка (*исчисление предикатов,* FOL – First-Order Logic или FOPC – first-Order Predicate Calculus) — формальное исчисление, допускающее высказывания относительно переменных, фиксированных функций, и предикатов. Расширяет логику высказываний. В свою очередь является частным случаем логики высшего порядка.

- Пример большого класса формальных языков – языки программирования, но не обладают общим механизмом логического вывода.

Синтаксис FOPC

$$
\begin{aligned}
\textit{Sentence} &\rightarrow \textit{AtomicSentence} \\
&\mid \textit{(Sentence Connective Sentence)} \\
&\mid \textit{Quantifier Variable,... Sentence} \\
&\mid \neg \textit{Sentence}
\end{aligned}
$$

$$
\textit{AtomicSentence} \rightarrow \textit{Predicate(Term,...)} \mid \textit{Term} = \textit{Term}
$$

$$
\begin{aligned}
\textit{Term} &\rightarrow \textit{Function(Term,...)} \\
&\mid \textit{Constant} \\
&\mid \textit{Variable}
\end{aligned}
$$

$$
\begin{aligned}
\textit{Connective} &\rightarrow \Rightarrow \mid \wedge \mid \vee \mid \Leftrightarrow \\
\textit{Quantifier} &\rightarrow \forall \mid \exists \\
\textit{Constant} &\rightarrow A \mid X_1 \mid \textit{John} \mid \ldots \\
\textit{Variable} &\rightarrow a \mid x \mid s \mid \ldots \\
\textit{Predicate} &\rightarrow \textit{Before} \mid \textit{HasColor} \mid \textit{Raining} \mid \ldots \\
\textit{Function} &\rightarrow \textit{Mother} \mid \textit{LeftLeg} \mid \ldots
\end{aligned}
$$

Метод резолюций Робинсона

Является *полным в смысле опровержения* – если в базе знаний существует противоречие, то оно будет найдено;

Является вариацией на тему «доказательство от противного»;

Лежит в основе работы таких систем, как PROLOG - декларативных;

Существенный недостаток – сложность интерпретации промежуточных состояний экспертом.

Преобразования в логике первого порядка

$$P \Leftrightarrow Q \rightarrow (\neg P \wedge \neg Q) \vee (P \wedge Q)$$
$$P \Rightarrow Q \rightarrow (\neg P \vee Q)$$
$$\neg(\forall x)P(x) \rightarrow (\exists x)\neg P(x)$$
$$\neg(\exists x)P(x) \rightarrow (\forall x)\neg P(x)$$

$$(\exists x)P(x) \rightarrow P(something) \text{ или } (\exists x)P(x) \rightarrow P(f())$$
$$(\exists x)P(x,y) \rightarrow P(somethinng, y) \text{ или } (\exists x)P(x,y) \rightarrow P(f(), y)$$
$$(\exists y)P(w,x,y,z) \rightarrow P(w,x,f(w,x),z)$$

Метод резолюций

Непосредственно перед началом работы выполняем следующие действия:

1. Преобразовать эквиваленции;
2. Преобразовать импликации;
3. Преобразовать отрицания – внести под кванторы;
4. Выполнить стандартизацию переменных – привести к префиксной форме, когда кванторы находятся перед предикатами;
5. Исключение квантора существования – процесс *сколемизации*;
6. Отбросить кванторы всеобщности;
7. Выражения вида R1^R2 разбить на отдельные дизьюнкты;
8. Разделить переменные – переименовать так, чтобы одна переменная встречалась только в одном дизъюнкте;

Алгоритм унификации

Пытаемся определить эквивалентность двух выражений. Переменные заменяем некоторыми другими термами.

Ограничения:

- Нельзя унифицировать константы (в том числе функцию от константы);
- Нельзя унифицировать переменную, уже входящую в другое выражение;
- Нельзя пытаться заменить переменную на функцию от этой переменной.

Стратегии в методе резолюций

- Исключение тавтологий;
- Исключение предложений, содержащих уникальные литералы;
- Сначала используем самые короткие предложения для склейки.

Пример

для самостоятельного изучения

1. Все небедные и умные люди счастливы;
2. Человек, читающий книги, – неглуп;
3. Джон умеет читать и является состоятельным человеком
4. Счастливые люди живут интересной жизнью

Существует ли человек, живущий интересной жизнью?

Продукционные модели

Знания представляются в виде «Если (условие) то (действие)» либо «Если (условие) то (следствие)».

"Условие" содержит описание ситуации, в которой применима продукция. Это описание задается в виде условий, называемых *посылками продукции*.

"Действие" — это набор инструкций, подлежащих выполнению в случае применимости продукции.

Структура продукционной системы

- База правил – набор продукций;
- База знаний (рабочая область) – набор известных или доказанных на данный момент фактов;
- Исполнительная логика – алгоритмы работы с базой правил и базой знаний.

Замечание: в некоторых источниках используется другая терминология. Базой знаний (Knowledge base) называют набор продукций, а множество всех доказанных фактов хранится в рабочей памяти (Working memory).

Пример продукционной системы

	Посылка	Следствие
1	**A, B**	**F**
2	**B, C, D**	**J**
3	**A, F**	**D**
4	**B**	**C**
5	**A, C, D**	**E**
6	**E, F**	**K**
7	**A, C, D**	**H**

Допустим, нам известно, что факты A и B достоверны, требуется доказать факты E и K.

Общие принципы работы

Продукционные системы легко реализуют системы с «исполнительной» логикой, в которых нужно определить реакцию на окружение;

Критерий работы (вывода):

- пока выводятся новые факты (пока множество применимых правил не пусто);
- пока не «поставлен диагноз» - выведен факт из некоторого целевого множества (например, в диагностических системах);
- постоянно – например, системы управления. При этом отслеживается предметная область, устанавливаются новые факты.

Продукции аналогичны по своей сути логическим правилам. При использовании логики первого порядка легко провести интеграцию логической и продукционной модели (то, чего не хватало Вампусу).

Алгоритмы работы (вывода)

- Методы прямого вывода (forward chaining, reasoning);
- Методы обратного вывода (backward chaining, reasoning).

Методы обратного вывода более предпочтительны, так как отрабатывают меньшее число продукций, обладают свойством направленности. Однако в некоторых случаях необходимо использовать методы прямого вывода.

Требования к системам

Одно из основных требований:

«*Предоставить эксперту возможность в любой момент остановить работу (процесс вывода), и пояснить эксперту, какие продукции и почему реализовывались*».

Требование из опыта использования экспертных систем, связанное с возникновением неверного вывода, постановки неверных диагнозов и прочего. В общем случае применение продукций меняет состояние рабочей области, и в сложных системах отслеживать адекватность вывода становится сложно. В любом случае, *вывод должен аргументироваться*.

Продукция

Продукция в общем виде:

(i); Q;P;A⇒B;N

- i – имя продукции;
- Q – сфера применения продукции;
- P – условие применимости (предикат);
- A⇒B – ядро продукции;
- N – постусловие, выполняемое после реализации ядра продукции.

Классификация ядер продукций

- Детерминированные – если может быть исполнена, то будет исполнена. Недетерминированные – может быть будет выполнена (вероятностные);
- Однозначные и альтернативные – справа в ядре могут присутствовать несколько вариантов выбора;
- Прогнозирующие: ЕСЛИ А, ТО с вероятностью Р ожидается В.

Пример конфликта

Номер продукции	Посылка	Действие
1	Растение впереди	Шаг вперед
2	Растение слева	Повернуть налево
3	Хищник слева	Повернуть направо
4	Хищник справа	Повернуть налево

Игра: прямоугольное поле, в клетке – трава, травоядное либо хищник. Циклы ход – восприятие, текущая ситуация (*растение слева, хищник справа, хищник слева*).

Фронт продукций (конкурирующие продукции): 2,3 и 4. Что делать?

Проблема выбора – основная проблема таких систем. Она пересекается с проблемой выявления неявных конфликтов, которые зачастую можно выявить только в процессе эксплуатации системы, и поиском решения конфликтных ситуаций.

Пример конфликта 2

Id	Посылка	Действие
1	**ОценкаГОС(Студент(X)) < ОценкаГОС(Студент(Y)) И СреднийБалл(Студент(X)) > СреднийБалл(Студент(Y))**	Магистр(X)
2	**ОценкаГОС(Студент(X)) > ОценкаГОС(Студент(Y)) И СреднийБалл(Студент(X)) < СреднийБалл(Студент(Y))**	Магистр(Y)
3	**ОценкаГОС(Студент(X)) > ОценкаГОС(Студент(Y)) И СреднийБалл(Студент(X)) > СреднийБалл(Студент(Y))**	Магистр(X)
4	**ОценкаГОС(Студент(X)) > ОценкаГОС(Студент(Y)) И СреднийБалл(Студент(X)) > СреднийБалл(Студент(Y)) И Публикации(Студент(X)) << Публикации(Студент(Y))**	Магистр(Y)

Игра: Поступи в магистратуру. Правила: нечеткие

Предметная область:

```
Собственно, простой пример. Есть три студента:
   Иванов - ИТ - 4 по госу - средний балл 4.5 - публикаций нет
   Сидоров - ИТ - 4 по госу - средний балл 4 - одна хорошая публикация
   Петров - ПМ - 4 по вступительному - средний балл 4.2 - три хороших публикации

   Иванов > Сидоров (по среднему баллу, в результате множественной
   диспетчеризации)
   Сидоров > Петров (по публикациям, без учета средних баллов)
   Петров > Иванов (по публикациям, без учета средних баллов)
```

Проблемы

1. Сложность представления функций, нечетких знаний и проч. – система недостаточно гибкая, хотя превосходит классическую логику;
2. Проблема дублирующих продукций: в примере продукции 1 и 2 согласованно (с точностью до равенства) представляют одно и то же правило;
3. Проблема учета исключений из правил: продукция 4 – правило учета публикаций (частный случай?);
4. Если Knowledge Base не согласована, то обосновать можно противоречивые факты. Так, интерпретатор обратного вывода аргументировано объяснит, почему именно студент X должен быть принят в аспирантуру, а студент Y – в картотеку биржи труда.

Разрешение конфликтов

- Различные алгоритмы разрешения конфликтов:
  - Назначение весов продукциям – при конфликте сравниваем веса, выбираем на основе некоторого критерия;
  - Выбор правил с наиболее жесткой левой частью (более специфицированное правило).
- Установки ограничений на конфигурации конфликтных наборов – разбиение правил на группы – использование P.

Семантические сети

Семантическая сеть представляет собой направленный граф, в котором вершины – объекты предметной области, а дуги – отражение связей между объектами.

Особенности:

- Графовая модель;
- Имеют давнюю историю (Чарльз Сандерс Пирс, Otto Selz и др.);
- Реализуют идеи множественного наследования.

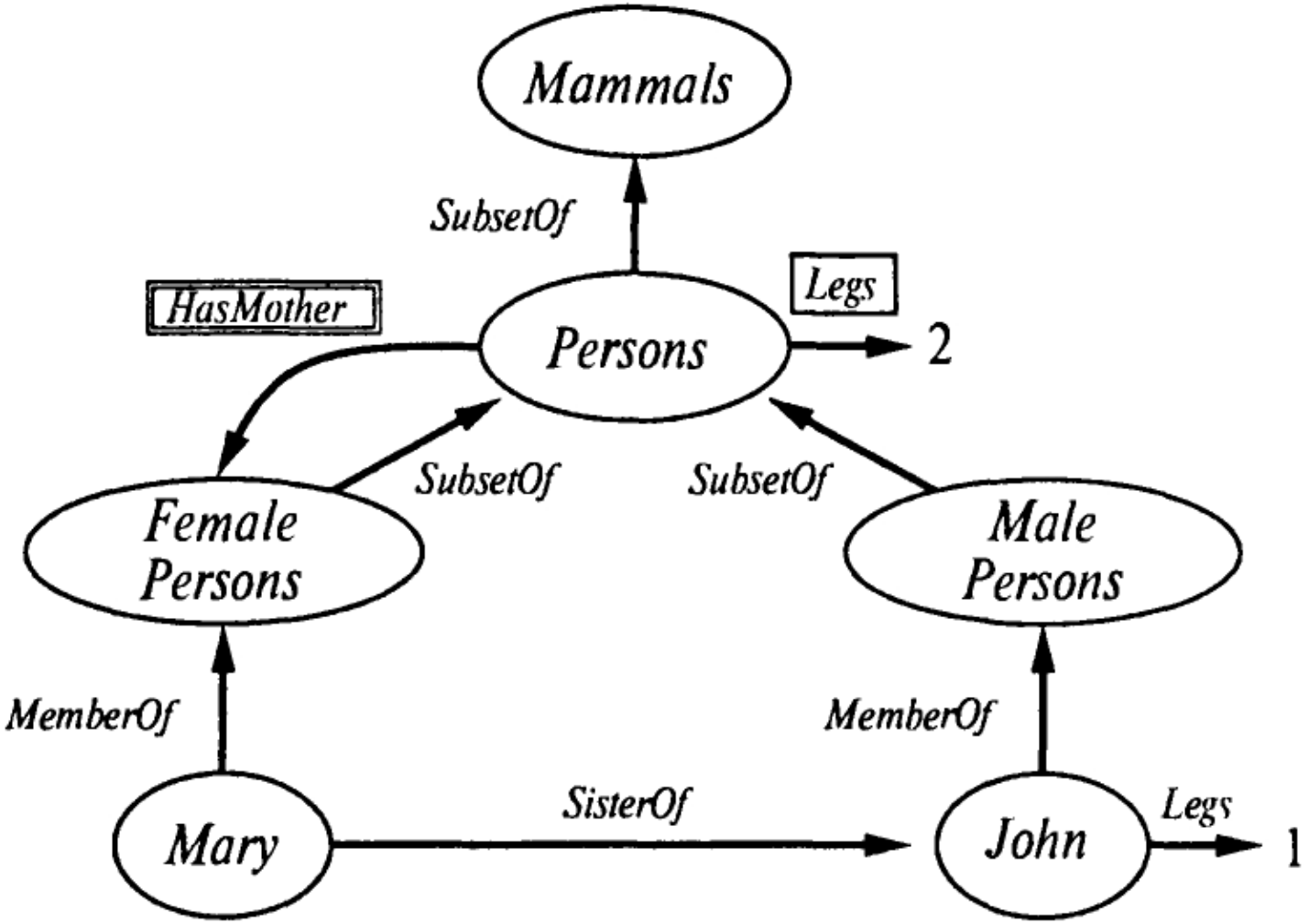
Хранение информации человеком

Коллинс и Квиллан, 1969

Общие особенности

1. Узлы семантических сетей представляют собой концепты предметов, событий, состояний;
2. Каждый концепт встречается только один раз;
3. Дуги семантических сетей создают отношения между узлами-концептами (пометки над дугами указывают на тип отношения);
4. Некоторые отношения между концептами представляют собой лингвистические падежи, такие как агент, объект, реципиент и инструмент (другие означают временные, пространственные, логические отношения и отношения между отдельными предложениями;
5. Концепты организованы по уровням в соответствии со степенью обобщенности так как, например, сущность, живое существо, животное, плотоядное.

Пример семантической сети



Классификация вершин

- Общие – вершины отражают понятия, события и общие законы, действующие в мире;
- Фактуальные – вершины отражают конкретные проявления общих объектов.

Виды связей

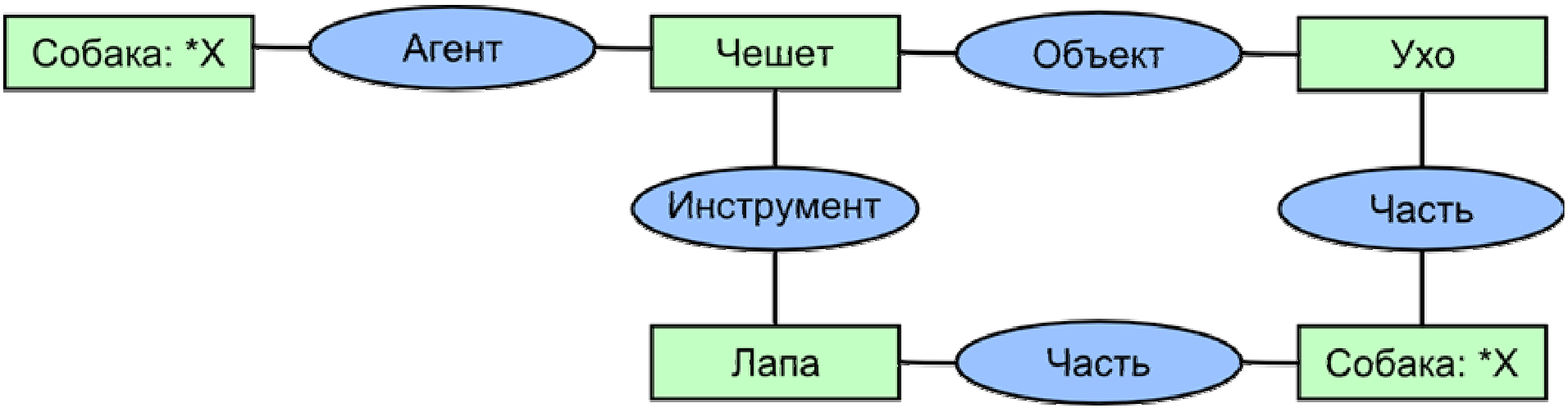
- Лингвистические (аттрибутивные) – «вода является мокрой», «птица умеет летать», «канарейка умеет петь», «канарейка имеет желтый цвет»;
- Теоретико-множественные («является частью», «является подмножетсвом», «имеет своей частью»);
- Логические;
- Квантификационные.

Процесс построения сетей

- Построить концептуальные графы для каждого объекта (концептуальный граф – граф, соответствующий некоторому высказыванию);
- Построить концептуальные графы для объектов на периферии;
- Объединить все построенные графы в сеть с помощью правил конъюнкции и упрощения.

Концептуальные графы

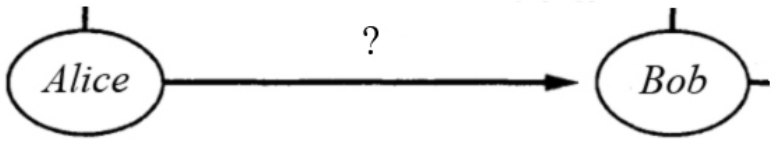
- Конечный связанный двудольный граф, метки дуг не используются,
- Узлы графа – *понятия*, или *концептуальные отношения*;
- Отношения между понятиями – узлы концептуальных отношений;
- Узлы понятий представляют абстрактные либо конкретные отношения.



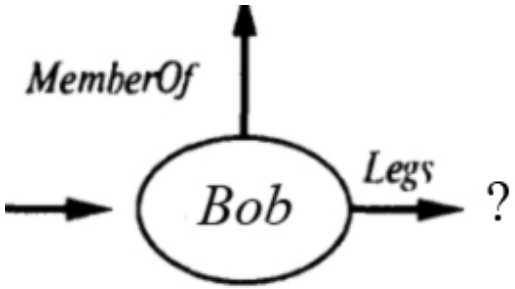
Методы вывода на СС

Различают две основных группы методов:

- *Методы изоморфного сопоставления* фрагментов семантической сети. При этом вопросы, ставящиеся к базе знаний, представляют в виде подграфа, и выполняется поиск подобных конструкций.



- *Методы распространения активности и техники пересечений*. Применимы, когда нам необходимо узнать, в каком отношении находятся два объекта.



Фреймовая модель

- Предложена Марвином Мински, первая публикация на эту тему – 1974;
- Является вариацией на тему семантических сетей;
- Мински определяет фрейм как «рамку», единицу представления знаний, запомненную в прошлом, детали которой при необходимости могут изменены согласно текущей ситуации.
- Другое определение фрейма: фрейм – структура для представления знаний, которая при ее заполнении соответствующими значениями превращается в описание конкретного факта, события или ситуации.

Пример фрейма

Фрейм «Стол»				
Слоты	Указатель наследования	Атрибуты слота	Значение	Демон
Вес	О	INTEGER	30	
Материал	О	FRAME	<указание на фрейм материала>	
Цвет	О	TEXT	Белый	
…	…	…	…	…
Слот n				

Описание

- Фрейм состоит из произвольного числа слотов, несколько из них обычно определяются самой системой для выполнения специфических функций, остальные – пользователем. Есть слот IS-A – фрейм-родитель данного фрейма, слот указателей дочерних фреймов, слот для ввода имени пользователя, даты определения, даты изменения, текста комментария и другие слоты.
- Указатели наследования Unique (U: уникальный), Same (S: такой же), Range (R: установление границ), Override (O: игнорировать) и т. п.
- *Демон*. Здесь дается определение демонов типа IF-NEEDED, IF-ADDED, IF-REMOVED и т. д. Демоном называется процедура, автоматически запускаемая при выполнении некоторого условия. Запускаются при обращении к слоту. Например, демон IF-NEEDED запускается, если в момент обращения к слоту его значение не было установлено, IF-ADDED запускается при подстановке в слот значения, IF-REMOVED запускается при стирании значения слота. Кроме того, демон является разновидностью присоединенной процедуры.
- *Присоединенная процедура*. В качестве значения слота можно использовать программу процедурного типа, называемую служебной (servant) (в языке Лисп) или методом (в языке Смолток).

Литература

1. Люгер Дж. Ф. Искусственный интеллект. Стратегии и методы решений сложных проблем, 4-е изд., – М.: «Вильямс», 2003.
2. Нильсон Н. Искусственный интеллект. Методы поиска решений – М.: «Мир», 1973;
3. Рассел С., Норвиг П. Искусственный интеллект. Современный подход, 2-е изд., – М.: «Вильямс», 2006.
4. Хант Э. Искусственный интеллект. – М.: «Мир», 1978;