

Лекция 1

Введение в представление знаний

mouromtsev@mail.ifmo.ru

Предпосылки появления СОЗ

- Первые исследования в области искусственного интеллекта
- Разнообразие трудноформализуемых задач
- Развитие когнитивной психологии, формальных логик и человеко-машинных интерфейсов и баз знаний

Методы принятия решений

- Только человек - теория принятия решений
- Только машина - теория автоматического управления
- Интеллектуальное поведение машины не означает полную замену оператора

Феномен знаний

- Справедливо ли говорить о знаниях как о техническом объекте?
- Можно ли выразить знания моделями и алгоритмами?
- Какова природа знаний?

Относительность наших знаний

- Любыe ли знания можно использовать для построения формальных моделей?
- Любыe ли проблемы позволяют ставить научные теории?
- Возможно ли решать задачи без моделирования?

Предмет изучения СОЗ

- Представление в памяти (репрезентация) знаний и когнитивные процессы.
- Моделирование процесса рассуждений.
- Проблема истинности и непротиворечивости решений.
- Проблемы нечетких знаний.
- Вопросы моделирования структуры и функций объектов реального мира НЕ рассматриваются.

Что есть «знания»?

- Группы информации:
 - контент (неупорядоченная и неструктурированная информация);
 - данные (имеют тип, характеризуются моделью данных);
 - знания.
- Знания — это закономерности предметной области (принципы, связи, законы), полученные в результате практической деятельности и профессионального опыта, позволяющие специалистам ставить и решать задачи в этой области.

Основные свойства знаний

- Знания субъективны — они зарождаются в памяти человека как результат мышления.
- Могут быть зафиксированы на материальных носителях и переданы от одного человека к другому.
- Оперируют связанным набором понятий (определяемых через интенсионал и / или экстенсионал).

Инженерия знаний

- Инженерия знаний — это ветвь информатики, изучающая модели и методы извлечения, структурирования и формализации (представления) знаний для их обработки в интеллектуальных и информационных системах.
- Этапы:
 - **извлечение** - протоколы и интервью, стенограммы, документы, фото и пр.,
 - **концептуализация (структурение)** - поле знаний: интелект-карты, концепт-графы, таблицы решений,
 - **формализация** - фреймы, семантические сети, продукции.

Языки представления знаний

- табличные,
- текстовые,
- графические (графовые),
- фреймы (объектные структуры),
- логические,
- (онтологии).

Представление знаний (1)

- Представление знаний — это множество синтаксических и семантических соглашений, которое делает возможным описание знаний о каком-либо предмете или задаче в памяти вычислительной машины.

Неоднозначность представления

- «Молотком стукнули по графину, и он разбился»
- «Графином стукнули по кирпичу, и он разбился»
- Что именно разбилось?

Сложные рассуждения

$$(\alpha \wedge (\beta \rightarrow \gamma)) \rightarrow ((\delta \vee \alpha) \wedge (\neg \alpha \rightarrow \neg \beta)).$$

- Является ли теоремой логики высказываний эта формула?

Соглашения о представлении

- *Что же все-таки разбилось, X или Y?*
- «X-ом стукнули по Y-ку, и он разбрёлся»

Представление знаний (2)

- Представление знаний — это высокоуровневое описание понятий, их взаимосвязи и действий, необходимых для решения проблемы на формальном языке, пригодном для построения компьютерной системы с одной стороны, и легком для понимания человеком, с другой.

Табличные формы представления знаний



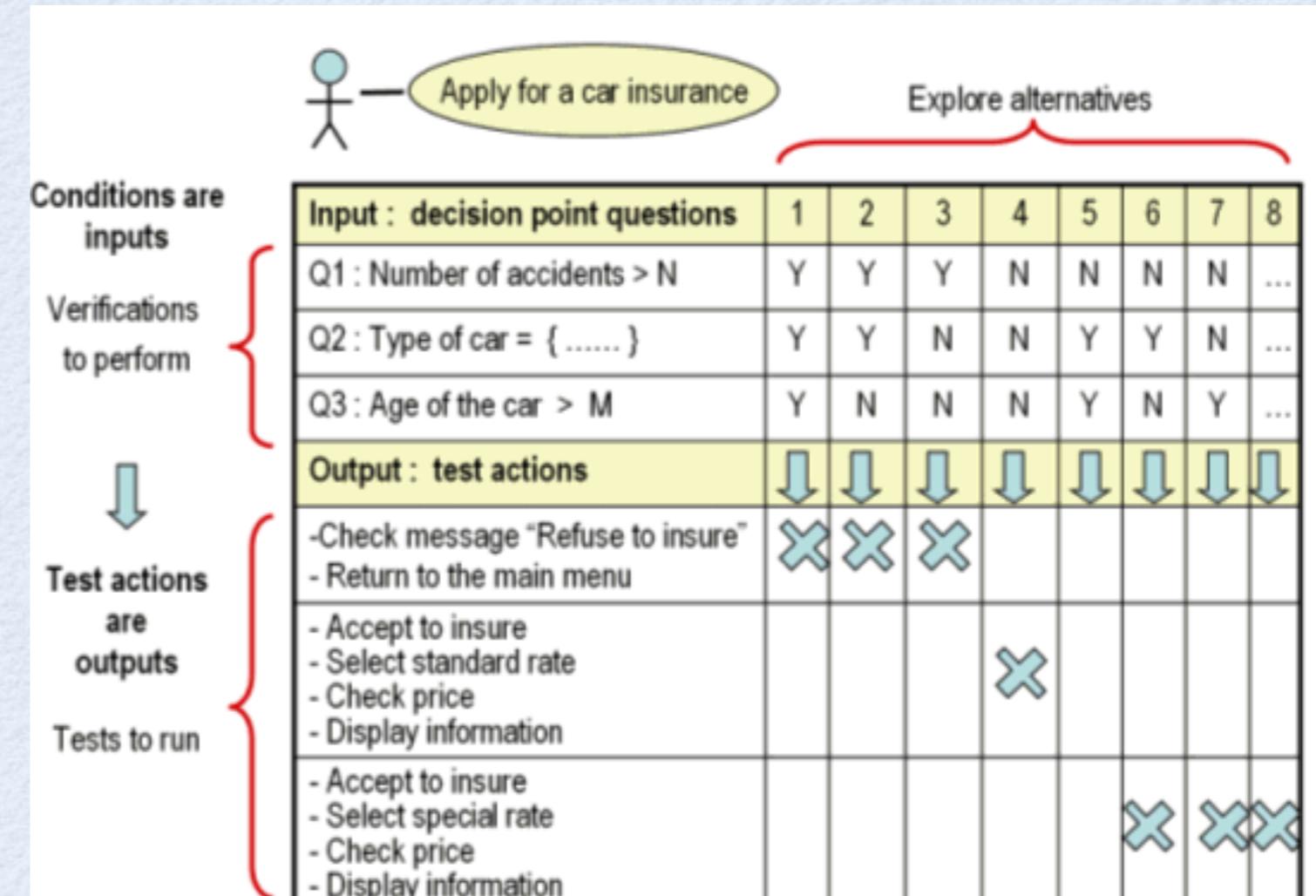
DecisionTableCreator

File View Setting

Conditions	1	2	3	4
Request login	0	1	1	1
Valid user name entered	X	0	1	1
Valid password entered	X	X	0	1

Actions	1	2	3	4
Offer recovery credentials	0	1	1	0
Activate entrybox user name	0	1	1	0
Activate entrybox password	0	0	1	0
Enter privileged area	0	0	0	1

Coverage 4 2 1 1 = 8 of 8



<http://decisiontables.wikispaces.com/Software+History>

Таблицы решений (decision table)

- логические таблицы
- отображают соотношения между комбинациями условий и комбинациями действий

Условия	Ситуации			
	c1	c2	...	c4
y1	I	O		I
...				
y2	I	I		O
Действия				
D1	O	I		I
...				
D2	I	O		I

Компоненты таблицы решений

- в ячейках таблицы решений могут использоваться любые символы (обычно 1/0, Да/Нет, Y/N)
- можно использовать символ «—» («не имеет значения»)
- структура таблицы модифицируется под задачу, т.к. в процессе использования добавление новых строк (условий и действий) затруднительно
- Алгоритм поиска решения прост в реализации

Столбец условий	Матрица соответствия условий
Столбец действий	Матрица возможных действий

Методика формализации знаний с помощью таблиц решений

1. Описание всех требуемых действий.
2. Выявление перечня условий, необходимых для выполнения указанных действий.
3. Определение ключевого набора условий при решении проблемы.
4. Поочередное описание ситуаций для частных решений проблемы.

	1	2	3	4	5	6	...
Есть движущаяся машина (1 – да, 0 – нет)	1	1	1	0	0	1	
Расстояние до машины (1 – близко, 2 – далеко)	1	2	2			2	
Свет светофора (1 – зеленый, 2 – желтый, 3 – красный)			3	не 1	1	2	
Пешеход спешит (1 – да, 0 – нет)				1			
Пешеход склонен к риску (1 – да, 0 – нет)			1				
Стоять на месте	1		1			1	
Перейти дорогу		1		1	1		

Пример таблицы решений “Чем заняться сегодня”

- Пример таблицы решений со всеми возможными сочетаниями вариантов условий

	1	2	3	4	5	6	7	8
Сегодня рабочий день?	1	1	1	1	0	0	0	0
Сегодня выходной?	0	0	1	1	1	1	0	0
Идет дождь?	1	0	1	0	1	0	1	0
Отправиться на работу	1	1						
Поехать на пикник					1	1	1	1
Смотреть телевизор			1		1		1	

Избыточность примера

- Первая и вторая ситуации рекомендуют одно и то же, а отличаются только одним условием: идет или не идет дождь
- аналогично ситуации 3 и 5, 4 и 6,
- условия можно объединить символом “-” (не имеет значения)

Оптимизированный пример

- Таблица решений из примера после оптимизации (устранения дублирующихся рекмендаций)

	1	2	3	4	5	6	7	8
Сегодня рабочий день?	1	-	-	0	0			
Сегодня выходной?	0	1	1	0	0			
Идет дождь?	-	1	0	1	0			
Отправиться на работу	1							
Поехать на пикник			1		1			
Смотреть телевизор		1		1				

Пример таблицы решений для проектирования объективов

	1	2	3	...
J - светосила системы	не 2	-	-	
W - угловое поле	не 2	не 0	-	
F - фокусное расстояние,	-	-	-	
L – хроматический диапазон	-	-	-	
Q – качество изображения,	-	-	-	
S - задний фокальный отрезок,	-	-	не 2	
D - положение входного зрачка	не 2	2	не 1	
Оптический элемент В	1P1A			
Оптический элемент К			303I	
Оптический элемент С				
Оптический элемент Y		2P2P		

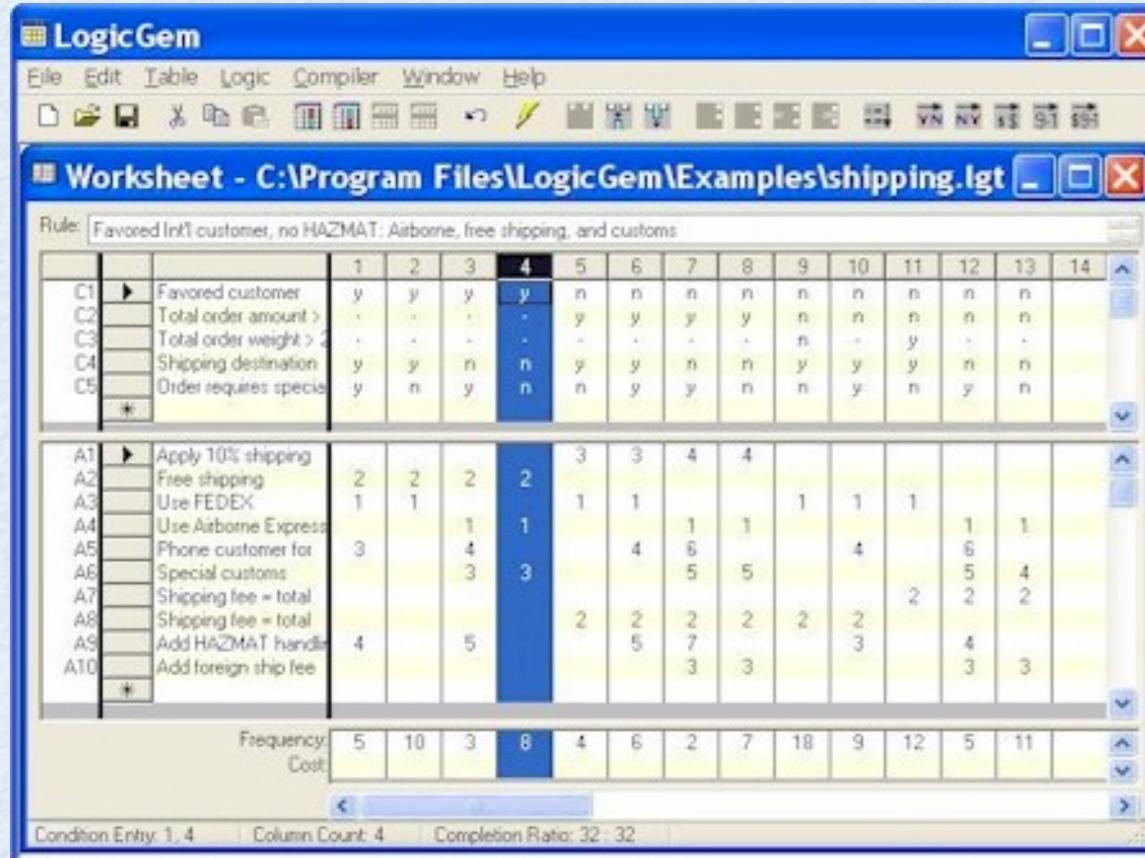
- Условия — параметры объектива, описываемые обобщенными значениями 1, 2 и 3
- Действия — рекомендации по установке конкретных линз (оптических элементов), представленных обозначениями оптических поверхностей

Достоинства и недостатки таблиц решений

- высокая степень формализации, наглядности процесса принятия решений,
- строятся регулярным образом и могут наращиваться практически до бесконечности,
- являются универсальным средством решения задач, для которых возможно описание ситуаций с помощью ограниченного набора условий,
- если различные ситуации характеризуются разными условиями, то таблицы решений становятся сильно разреженными и малоэффективными.

LogicGem 3.0

<http://www.catalyst.com/products/logicgem/>



```

Procedure: shipping.lgt

1
If Favored customer is true and
    Shipping destination inside U.S. is true and
        Order requires special HAZMAT (hazardous materials) packing
is true , then
    Use FEDEX,
    Free shipping,
    Phone customer for special handling,
    Add HAZMAT handling fee $45.00,

2
If Favored customer is true and
    Shipping destination inside U.S. is true and
        Order requires special HAZMAT (hazardous materials) packing
is not true , then
    Use FEDEX,
    Free shipping,

```

<http://books.ifmo.ru/file/pdf/497.pdf>

Drools Expert

<http://www.jboss.org/drools/drools-expert.html>

The screenshot shows the Drools Expert interface with a decision table for mortgages. The table has columns for Description, amount min, amount max, period, deposit max, income, Application Date, Loan approved, and LMI. The table contains three rows with data: row 1 (amount min: 100001, amount max: 130000, period: 20, deposit max: 3000, income: Job, Application Date: 2011-02-17, Loan approved: true, LMI: 10), row 2 (amount min: 10000, amount max: 100000, period: 20, deposit max: 2000, income: Job, Application Date: 2011-02-16, Loan approved: true, LMI: 0), and row 3 (amount min: 131000, amount max: 200000, period: 30, deposit max: 20000, income: Asset, Application Date: 2011-02-15, Loan approved: true, LMI: 0).

#	Description	amount min	amount max	period	deposit max	income	Application Date	Loan approved	LMI
1		100001	130000	20	3000	Job	2011-02-17	true	10
2		10000	100000	20	2000	Job	2011-02-16	true	0
3		131000	200000	30	20000	Asset	2011-02-15	true	0

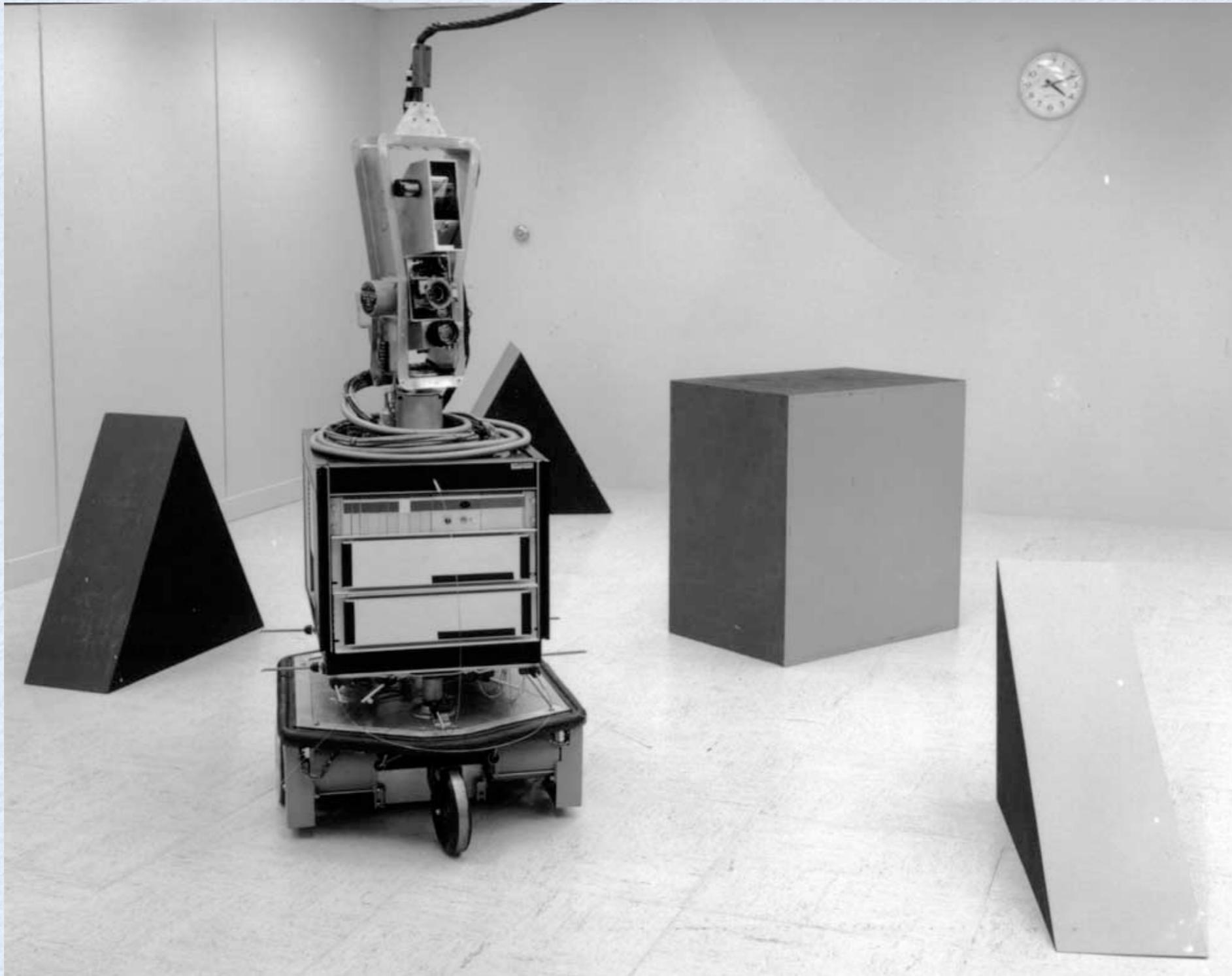
<http://window.edu.ru/resource/741/79741/files/itmo1080.pdf>

Таблицы операторов

- Предложены в системе STRIPS (сокр. от STanford Research Institute Problem Solver).
- Разработаны в 1971 году: Ричардом Файксом (Fikes) и Нильсоном Нильсоном (Nilsson)*.
- Предназначались для решения проблемы формирования плана поведения мобильного робота Shakey

* Fikes R.E. and Nilson N.J. (1971). STRIPS: a new approach to the application of theorem proving to problem solving. Artificial Intelligence, 2, 1971, p. 189-208
(Русский перевод: Файкс Р., Нильсон Н., Система STRIPS – новый подход к применению методов доказательства теорем при решении задач. – в сб. Интеллектуальные роботы, вып. 1. Под. Ред. Г.Е. Поздняка. – с. 382 – 403.)

Shakey, 1967



Shakey за работой. 1969



Shakey передан в музей, 1983

<http://www.computerhistory.org/>



Формализация задачи I

- Текущее состояние окружающей среды — помещений и предметов в них — представляется набором выражений предикат-аргумент:

`<предикат> ::= <предикатный символ>
(<аргумент1>, ..., <аргументn>) ;`

`at(robot, roomA) .`

Формализация задачи II

- совокупность (список или множество) предикатов образуют модель мира:

$$W^1 = \{ \text{at}(\text{robot}, \text{roomA}), \text{at}(\text{box1}, \text{roomB}), \\ \text{at}(\text{box2}, \text{roomC}) \};$$

- Конечная ситуация также задается множеством предикатов:

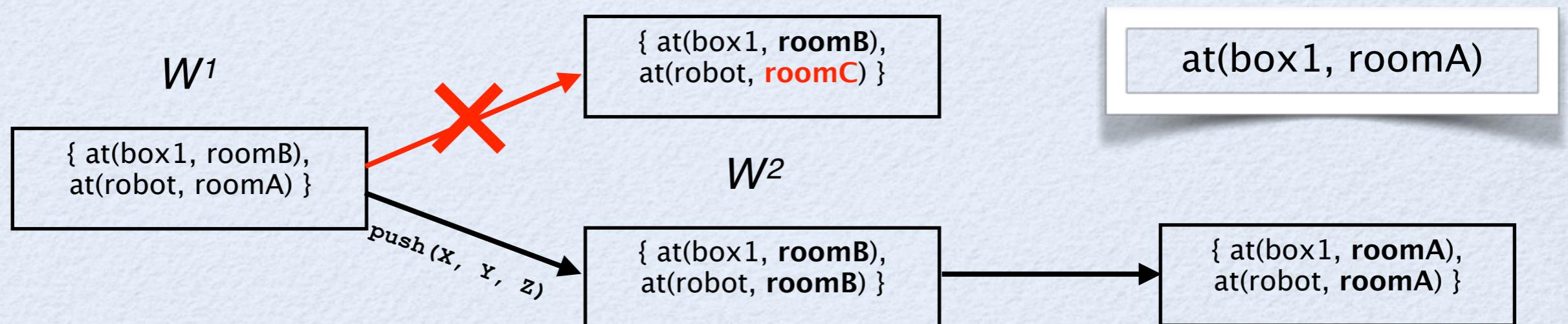
$$W^K = \{ \text{at}(\text{box1}, \text{roomA}), \text{at}(\text{box2}, \text{roomB}) \}.$$

Пример таблицы операторов

Оператор:	move(X, Y)	push(X, Y, Z)
Предварительные условия:	$\{ \text{at}(\text{robot}, X) \}$	$\{ \text{at}(\text{robot}, Y), \text{at}(X, Y) \}$
Список удалений:	$\{ \text{at}(\text{robot}, X) \}$	$\{ \text{at}(\text{robot}, Y), \text{at}(X, Y) \}$
Список добавлений:	$\{ \text{at}(\text{robot}, Y) \}$	$\{ \text{at}(\text{robot}, Z), \text{at}(X, Z) \}$

Метод «анализа целей и средств»

- Последовательный перебор операторов таблице невозможен из-за экспоненциального роста числа вариантов перебора.
- На каждом шаге отличие между текущим состоянием и целевым должно уменьшаться.



Достоинства и недостатки таблиц операторов

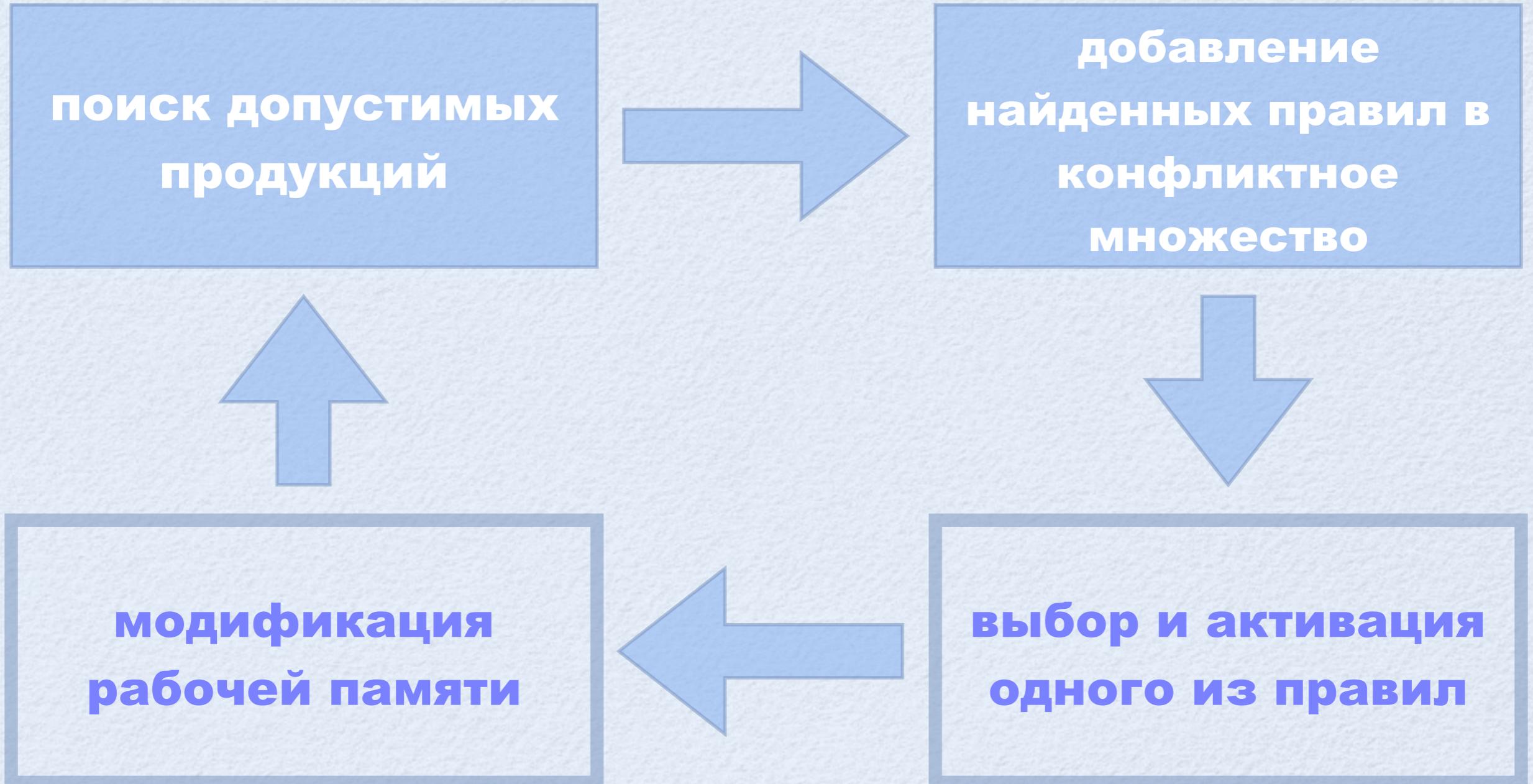
- Таблица операторов, модель мира и цели представлены с помощью одного и того же синтаксиса в виде конструкций предикат–аргумент.
- Простой алгоритм поиска необходимых для достижения цели операций.
- Однако существуют ситуации, в которых система STRIPS оказывается не способной решать поставленные задачи. Одна из таких ситуаций получила название аномалия Зюссмана (Sussman Anomaly).

Продукционная система (production system)

- Модель вычислений, основанная на продукционных правилах (production rule), использует правила вида «ЕСЛИ условие, ТО действие» (IF... THEN...).
- Впервые идея появилась в работе Эмиля Поста (Emil Leon Post), 1943*, посвященной исследованию частного случая канонической системы Поста.
- Управление продукционными системами основано на Марковских алгоритмах.
- Пост доказал, что любая система в математики или логике может быть представлена в виде продукционных правил. То есть система продукции эквивалентна машине Тьюринга.

* Post E. (1943) Formal reduction of the general combination problem. American journal of Mathematics, 65: 197-268, 1943

Цикл распознавание-действие и разрешение конфликтов



Стратегии разрешения конфликтов

Пример реализации стратегий разрешения конфликтов в системе OPS5* (модификации системы OPS - "Official Production System», разработанной в конце 1970-х Чарльзом Форги):

- *Рефракция (refraction)* для предотвращения зацикливания.
- *Новизна (recency)* позволяет сосредоточить поиск на одной линии рассуждения.
- *Специфичность (specify)* отдает предпочтение более конкретным правилам перед более общими

* Brownston L., Farrel R., Kant E., Martin N. (1985) Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming. Reading, MA: Addison-Wesley, 1985.

Разрешение конфликтов в DROOLS

В современной системе DROOLS имеется значительное количество реализованных стратегий, а также можно определить свою*:

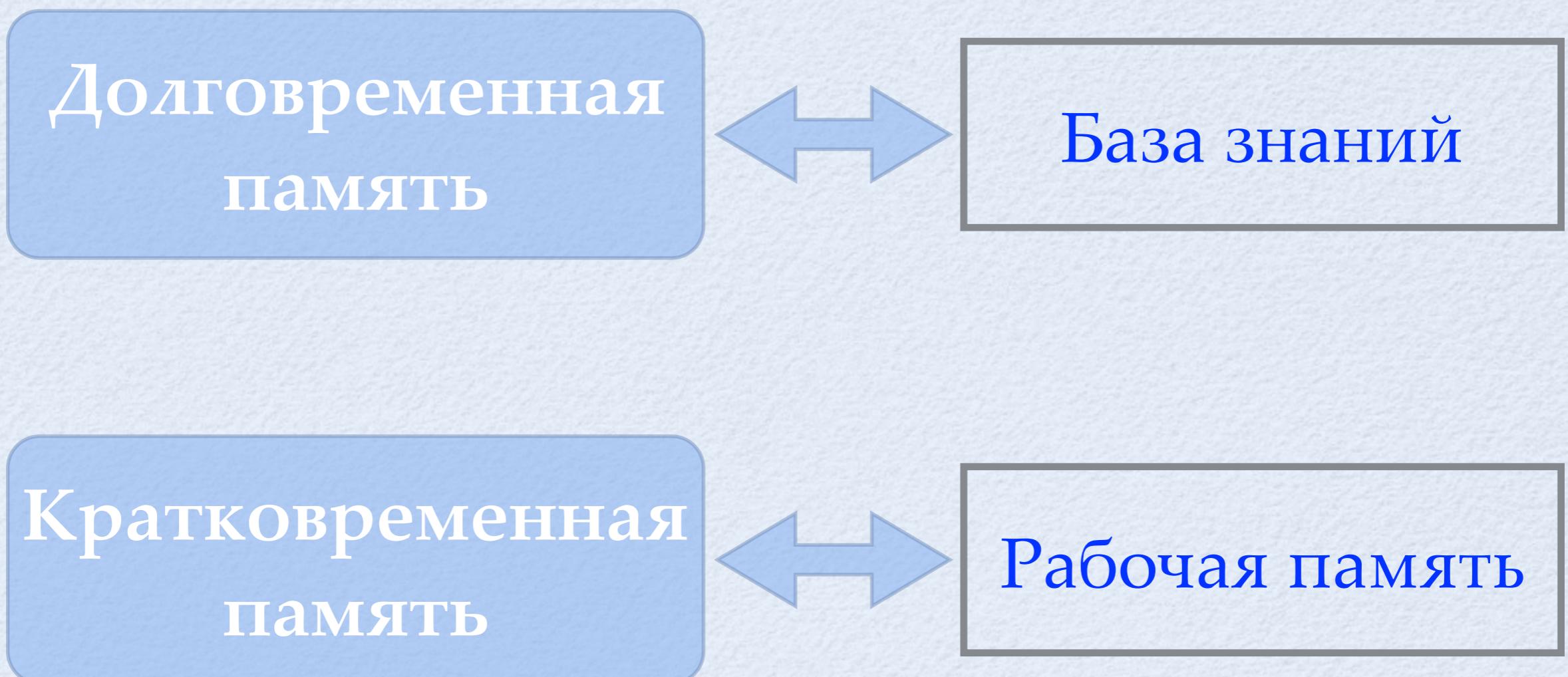
- *Приоритет* («выпуклость» - Salience).
- *Новизна* (Recency).
- *Первенство* (Primacy).
- *FIFO* (First In First Out).
- *LIFO* (First In First Out).
- *Сложность* (Complexity).
- *Простота* (Simplicity).
- *Порядок загрузки* (LoadOrder).
- *Случайность* (Random).

* <http://legacy.drools.codehaus.org/Conflict+Resolution>

Механизм возврата

- Чистая производственная модель не предусматривает выхода из тупиковых ситуаций в процессе поиска.
- Гораздо более эффективным является модификация цикла управления с механизмом *возврата* (cycle back) в предыдущее состояние модели мира.

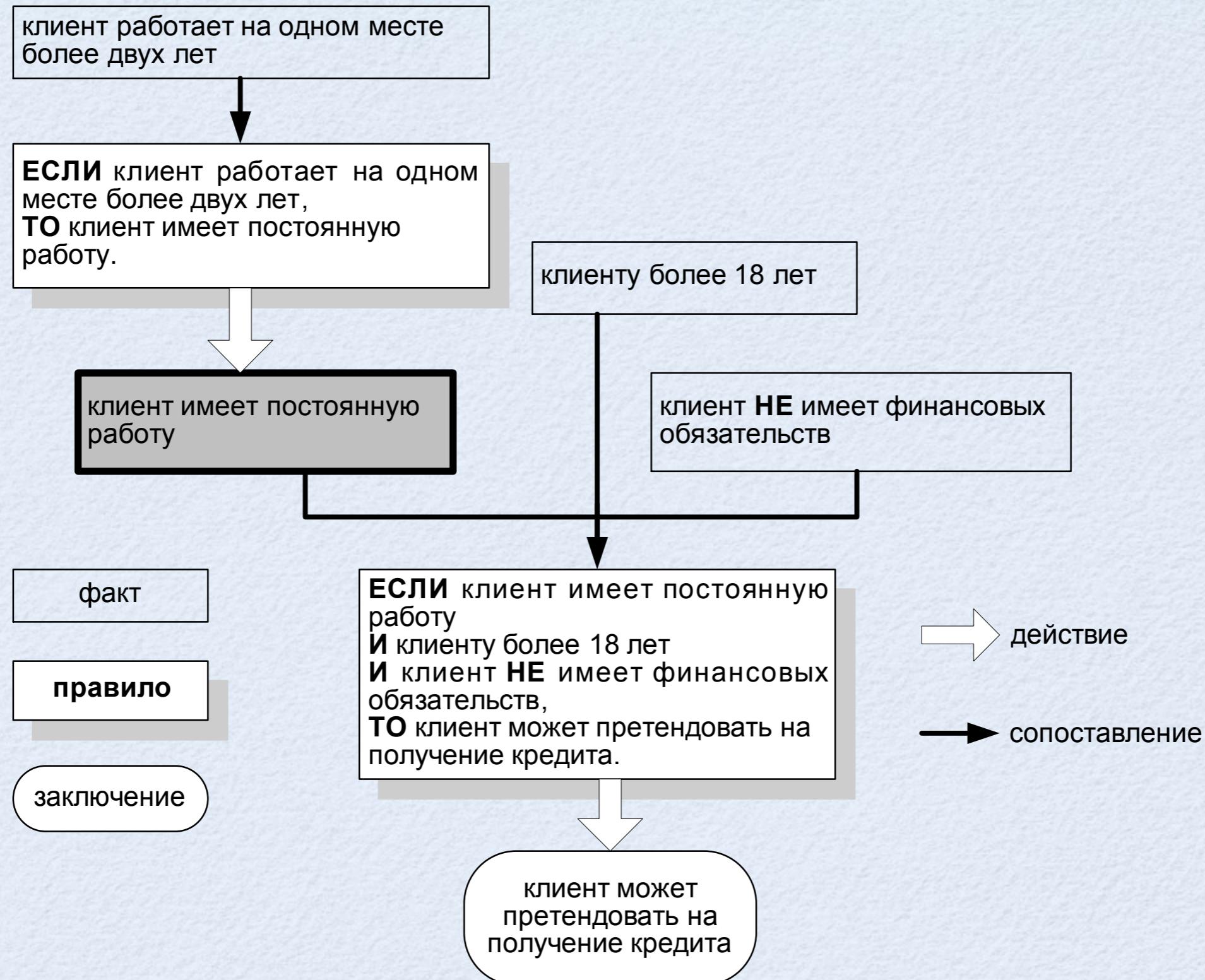
Аналогия с памятью человека



Примеры продукции

- П1: **ЕСЛИ** клиент работает на одном месте более двух лет, **ТО** клиент имеет постоянную работу.
- П2: **ЕСЛИ** клиент имеет постоянную работу **И** клиенту более 18 лет **И** клиент **НЕ** имеет финансовых обязательств, **ТО** клиент может претендовать на получение кредита.

Цепочка вывода (reasoning)



Эвристические знания и метаправила

- Иногда для получения решения требуется вмешательство в стандартный процесс вывода.
- Метаправила не принимают непосредственного участия в процессе формирования рассуждений, а определяют приоритет выполнения или исключают из рассмотрения обычных правила и выполняются в первую очередь.

Пример метаправила (I)

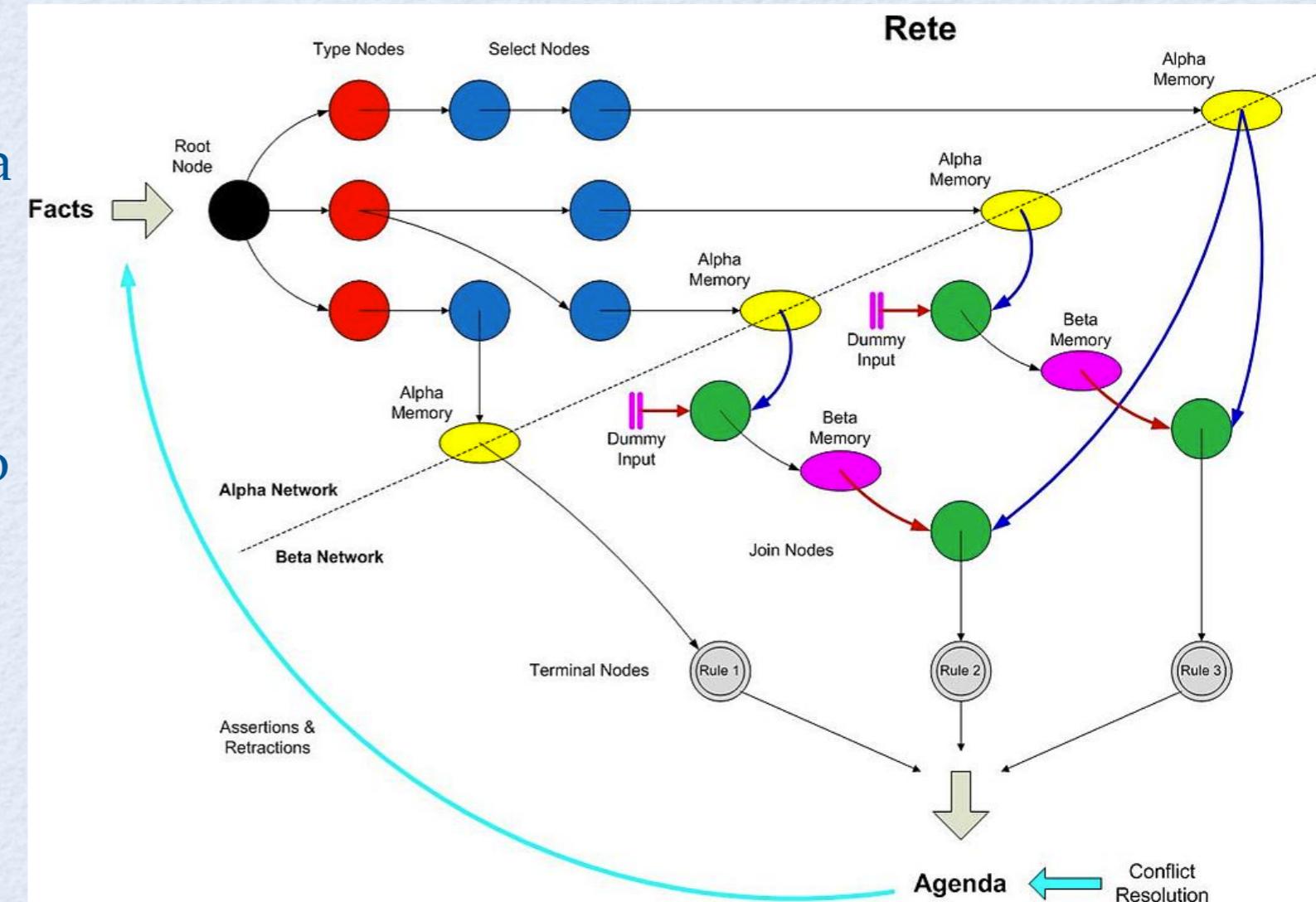
ЕСЛИ кредитный рейтинг клиента высокий **И**
клиент является клиентом банка, ТО сначала
применить правила для льготных условий
предоставления кредита.

Пример метаправила (II)

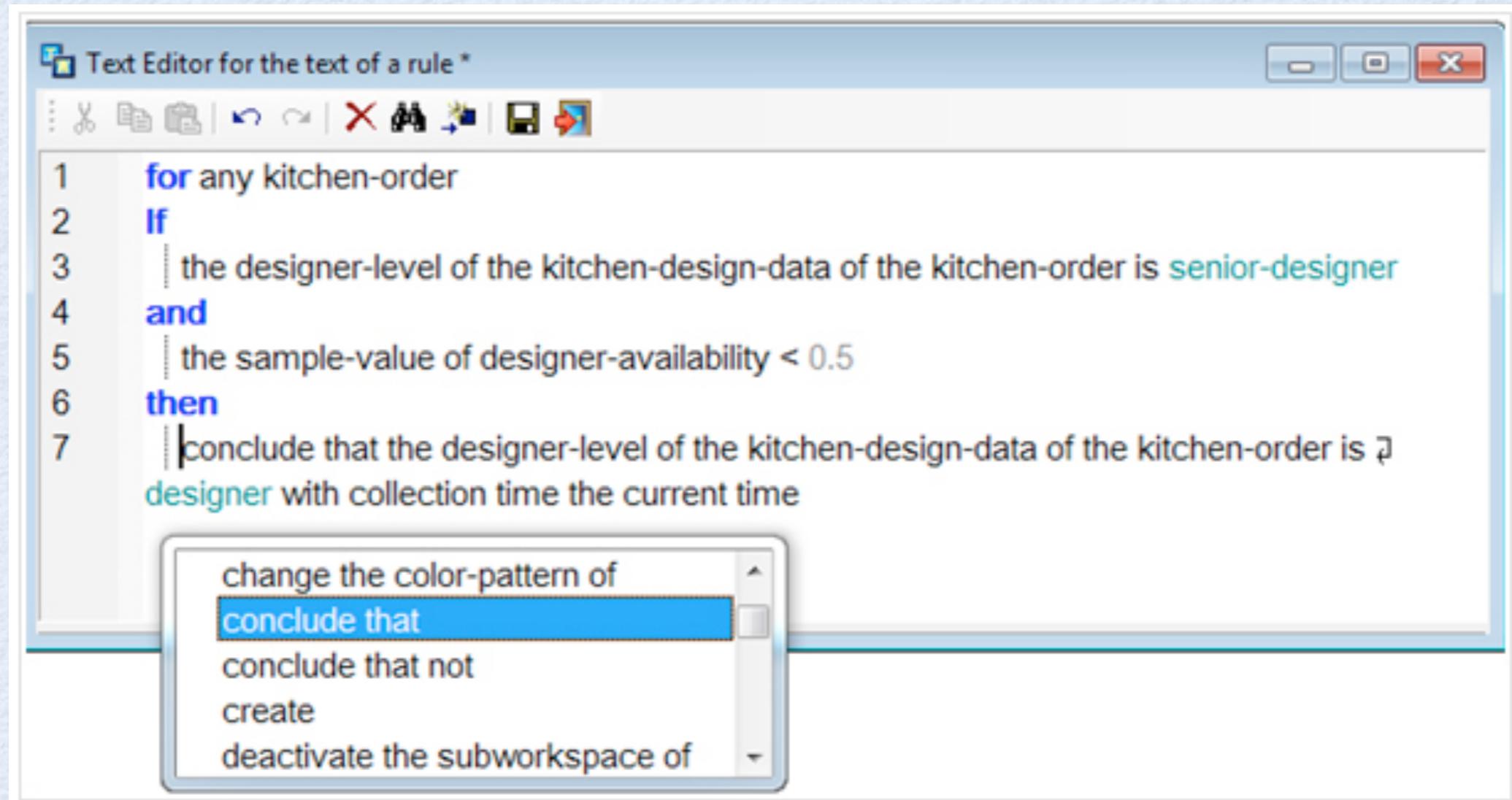
- **ЕСЛИ** существуют правила, в условиях которых не упоминается текущая цель **И** существуют правила, в условиях которых упоминается текущая цель, **ТО** сначала следует активизировать первые из перечисленных правил.

Алгоритм Rete для оптимизации поиска правил

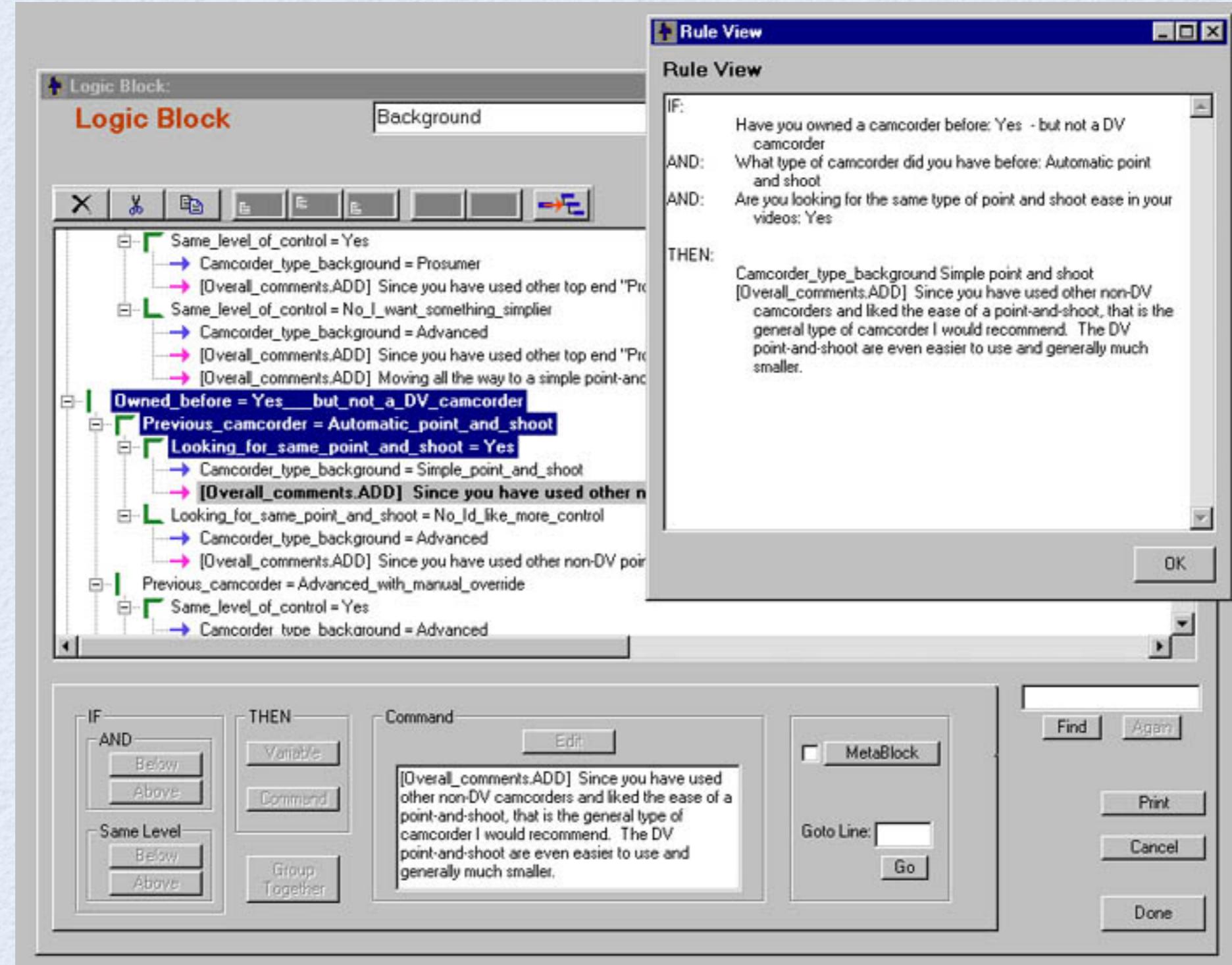
По существу, алгоритм RETE интегрирует правила в сетевую структуру, позволяющую системе сопоставлять правила с данными, непосредственно указывающими на правило. Этот алгоритм значительно ускоряет выполнение поиска, особенно, если число правил велико.



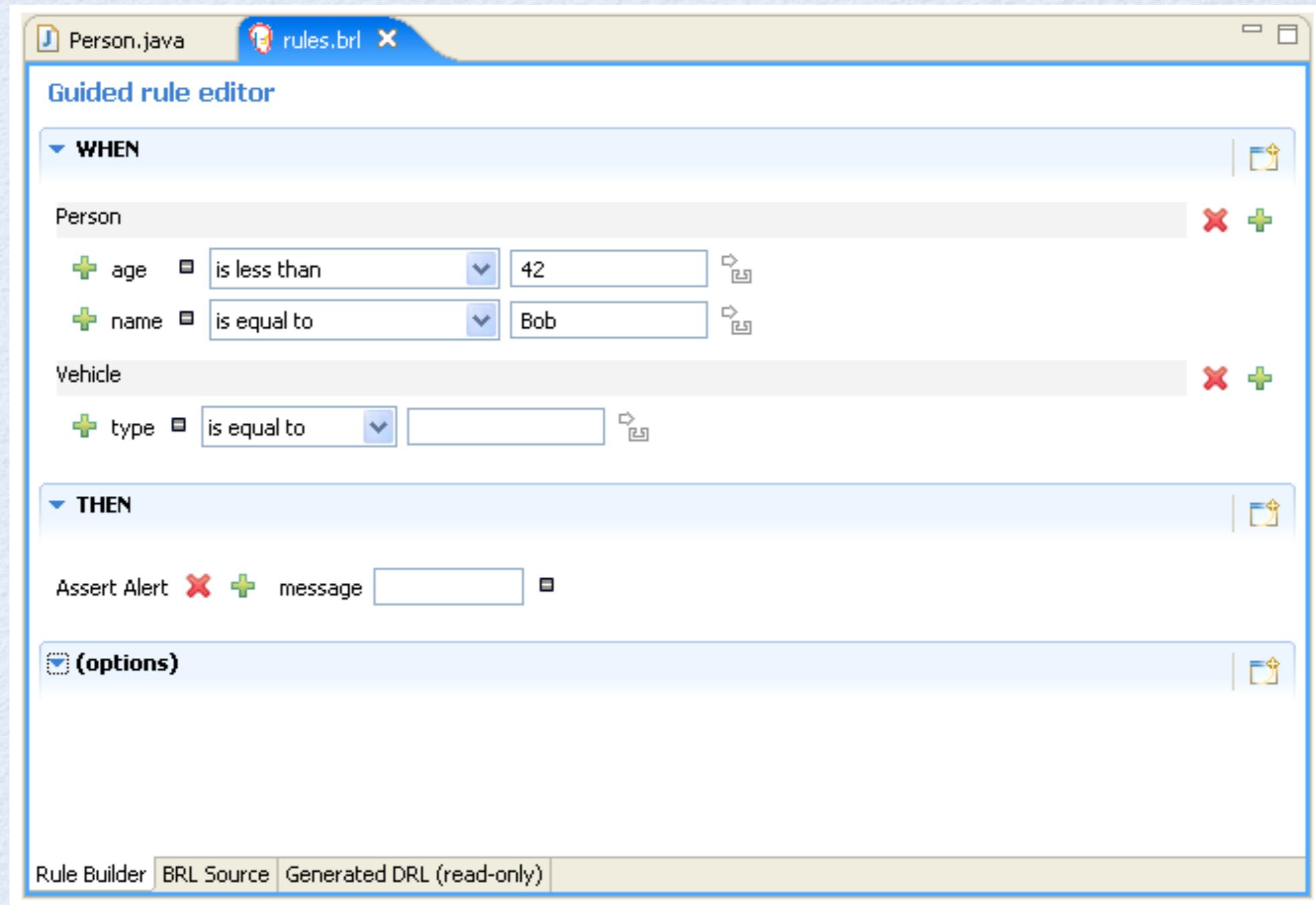
Примеры систем: G2



Примеры систем: Exsys



Примеры систем: Drools



Пример правила в Drools

**IF the entrance pupil of an optical system is removed forward
THEN exclude the all base elements with exception of B3A3P
element**

```
rule "The B3A3P element"
when
    Classification( D == 2 )
    $elements := ArrayList() from
    collect( OpticalElement() )
then
    modify( $elements ) {
        clear()
        add( new OpticalElement( type == "B",
                                surfaces == "3A3P" ) )
    }
end
```

Пример метаправила в Drools

IF the optical system is not fast

THEN a fast element isn't needed at the system

rule "No fast"

when

Classification(J == 1)

\$conds := SystemConditions()

then

modify(\$conds) {

setIsFastNeeded(false)

}

end

Семантическая сеть как метод представления знаний

- *Семантическая сеть*, этот метод представления знаний позволяет описывать объекты, явления и понятия предметной области с помощью сетевых структур, основанных на теории графов.
- *Семантика* — это наука, устанавливающая отношения между символами и объектами, которые они обозначают, или наука, определяющая смысл знаков.
- *Сеть* — разновидность графа.

Первые исследования в области графических языков

- Экзистенциальные графы (*existential graph*), Чарльз Пирс (Charles Sanders Peirce), 1909 год - основа графической логики, которую он называл «логикой будущего»;
- Теория *схематического упреждения* (*schematic anticipation*), Отто Зельц (Otto Selz), 1922 год - целенаправленный метод фиксации мыслительного процесса при поиске ассоциаций и обобщенных понятий;
- Ньюэлл и Саймон адаптировали метод Зельца для изучения процесса решения проблем человеком;
- Росс Квиллиан (Ross Quillian) использовал комбинацию сетей Зельца и семантических сетей для построения системы машинного перевода;
- *Графы концептуальной зависимости*, Шенк и Теслер (Schank, Tesler), 1969;
- *Сети структурного наследования*, Бракман (Brachman), 1979.

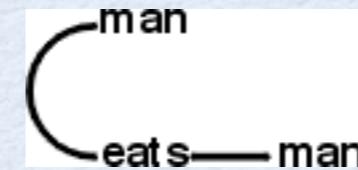
Экзистенциальные графы

	Peirce's Notation		Peano's Notation	
Operation	Symbol	Explanation	Symbol	Explanation
Disjunction	+	Logical sum	∨	v for <i>vel</i>
Conjunction	×	Logical product	∧	Upside down v
Negation	−	−1=0 and −0=1	~	Curly minus sign
Implication	→	Equal or less than	⊃	C for <i>consequentia</i>
Existential Quantifier	Σ	Iterated sum	Ǝ	E for <i>existere</i>
Universal Quantifier	Π	Iterated product	()	O for <i>omnis</i>

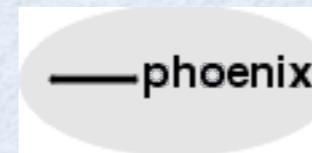
Existential Graphs. MS 514 by Charles Sanders Peirce with commentary by John F. Sowa <http://www.jfsowa.com/peirce/ms514.htm>

Экзистенциальные графы

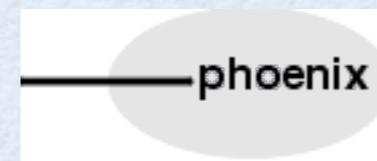
Some man eats a man



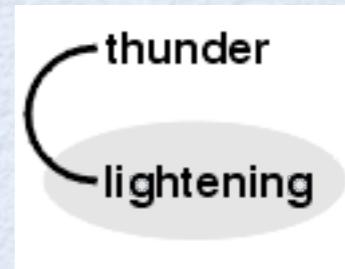
It is false that there is a phoenix



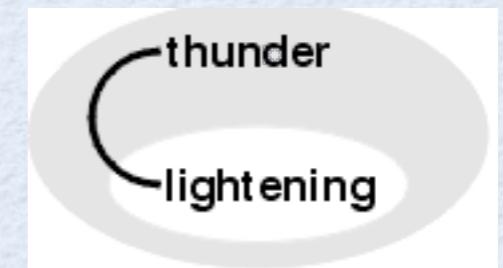
$(\exists x) \sim \text{phoenix}(x)$.



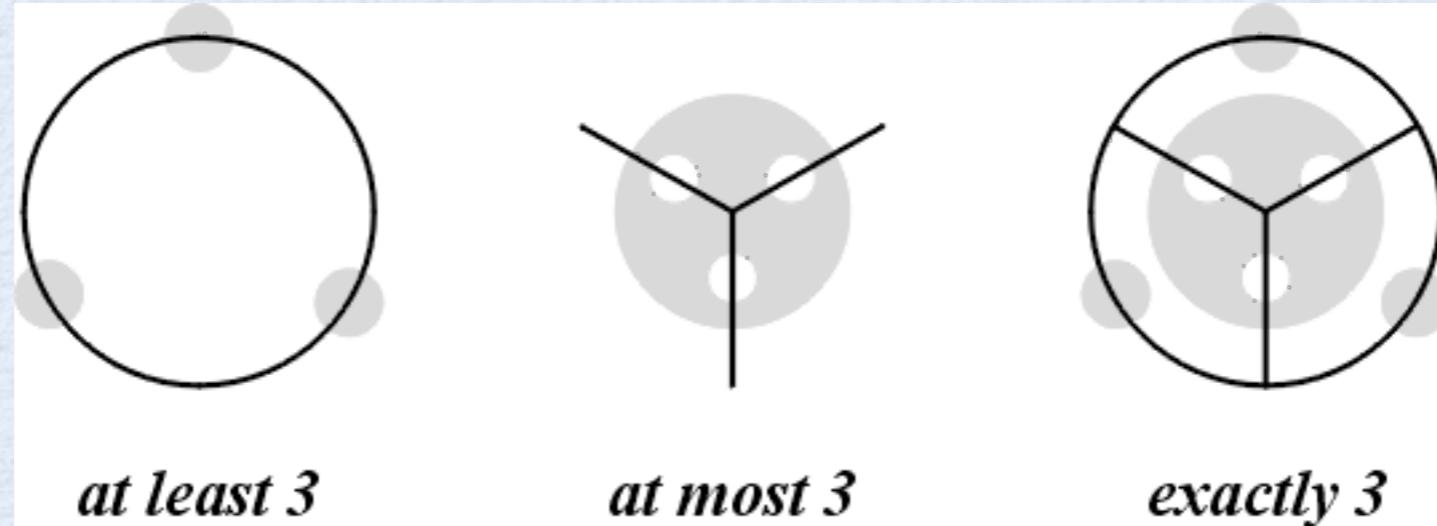
$(\exists x) (\text{thunder}(x) \wedge \sim \text{lightening}(x))$.



$\sim (\exists x) (\text{thunder}(x) \wedge \sim \text{lightening}(x))$.



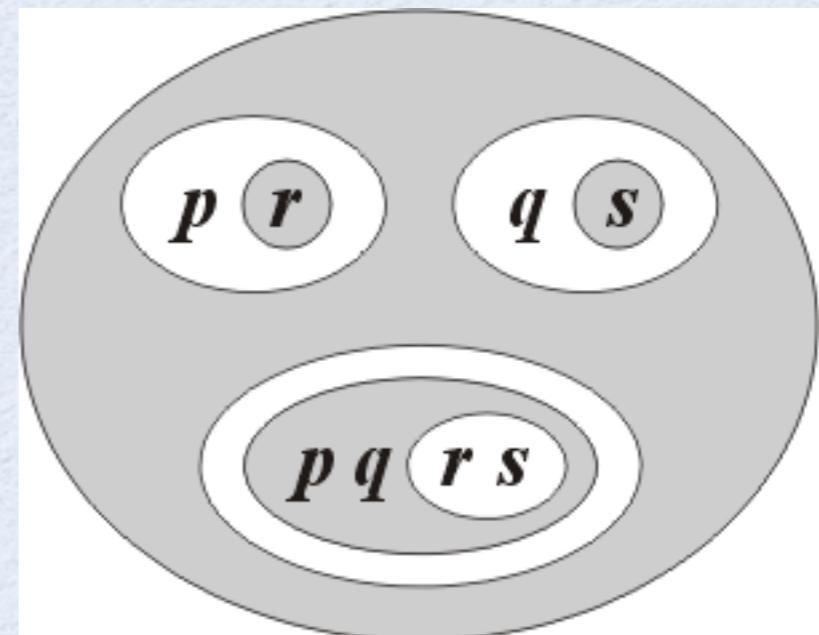
Экзистенциальные графы



Leibniz's *Praeclarum Theorema*

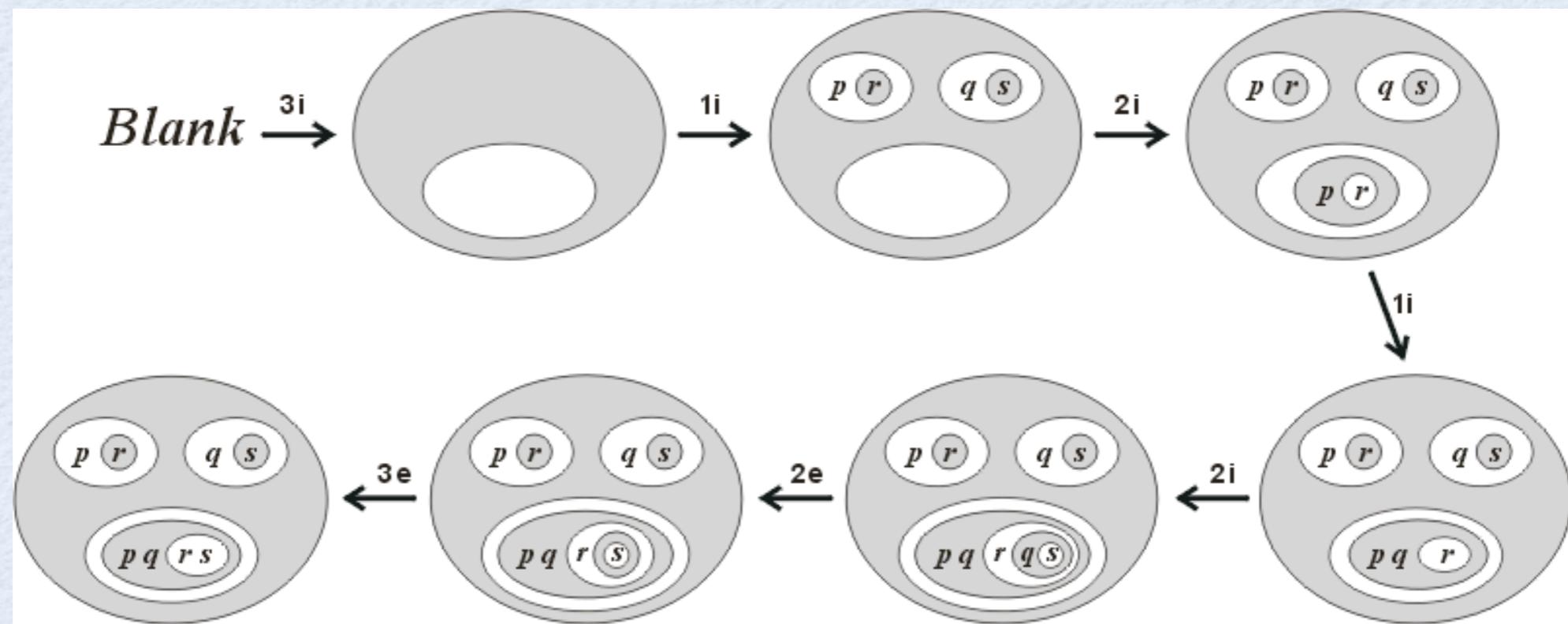
$$((p \supset r) \wedge (q \supset s)) \supset ((p \wedge q) \supset (r \wedge s)).$$

<https://planetmath.org/praeclarumtheorema>



Экзистенциальные графы

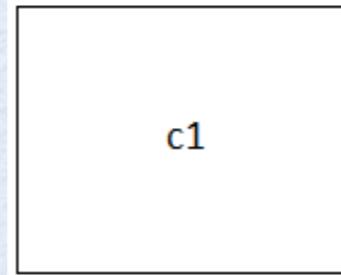
- Графическое доказательство теоремы Лейбница



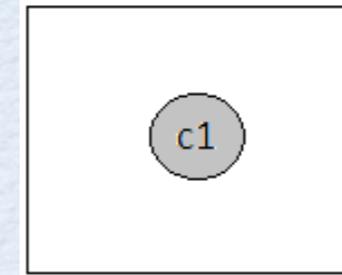
Existential Graphs. MS 514 by Charles Sanders Peirce with commentary by John F. Sowa <http://www.jfsowa.com/peirce/ms514.htm>

Экзистенциальные графы

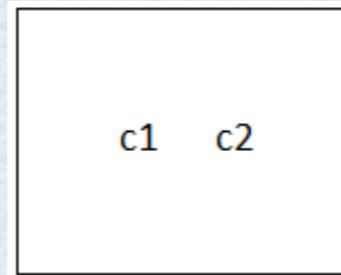
1. $c1$



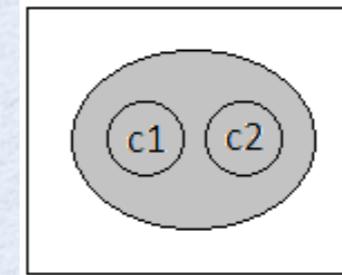
2. $\neg c1$



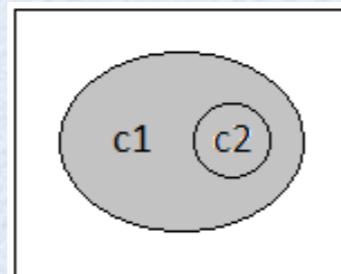
3. $c1 \sqcap c2$



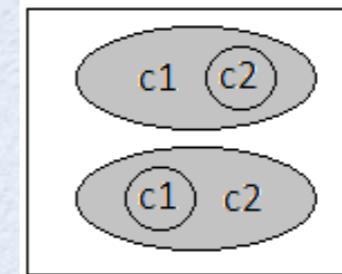
4. $c1 \sqcup c2$



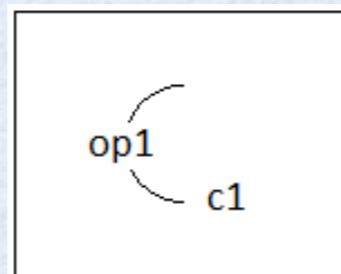
5. $c1 \sqsubseteq c2$



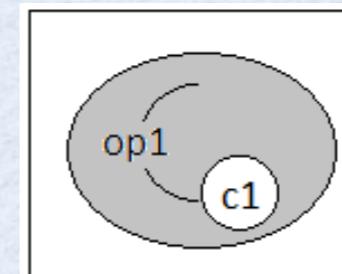
6. $c1 \equiv c2$



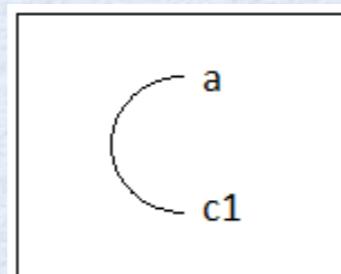
7. $\exists op1.c1$



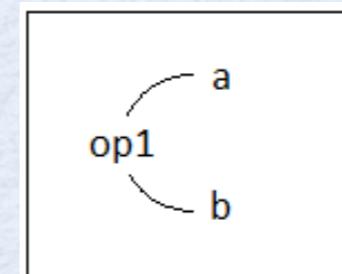
8. $\forall op1.c1$



9. $a : c1$



10. $(a, b) : op1$



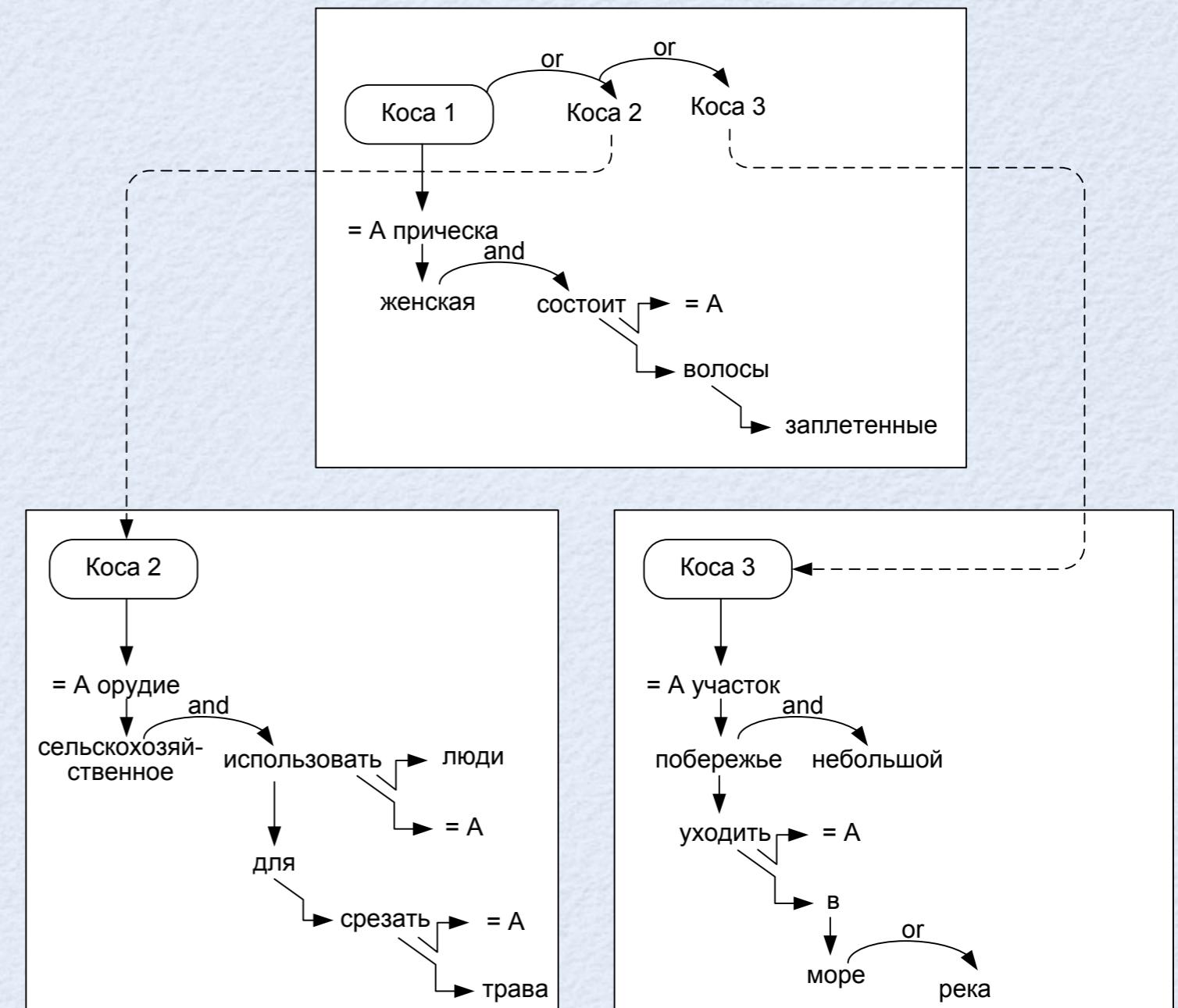
Ассоциативная модель памяти и представление смысла

- Росс Квиллиан (Ross Quillian): в основе восприятия текста человеком лежит «создание некоторого рода мысленного символического представления»;
- Моделирования человеческой памяти с помощью сетевых структур, в которых узлы соответствуют словесным понятиям, а связи между узлами — отношениям между понятиями;
- Структура сети: узел-тип соответствует какому-либо понятию и связан с определенной комбинацией узлов-лексем, являющихся определением данного понятия, а смысл узла-лексемы определяется через ссылку на соответствующие узлы-типы (подобно толковому словарю).

Пример: три различных определения понятия «Коса»

На рисунке представлены три плоскости, в которых представлены сети для определения различных значений слова «Коса»:

- Коса 1 — женская прическа, состоящая из заплетенных волос.
- Коса 2 — сельскохозяйственное орудие, используемое людьми для срезания травы.
- Коса 3 — небольшой участок побережья, уходящий в море или реку.



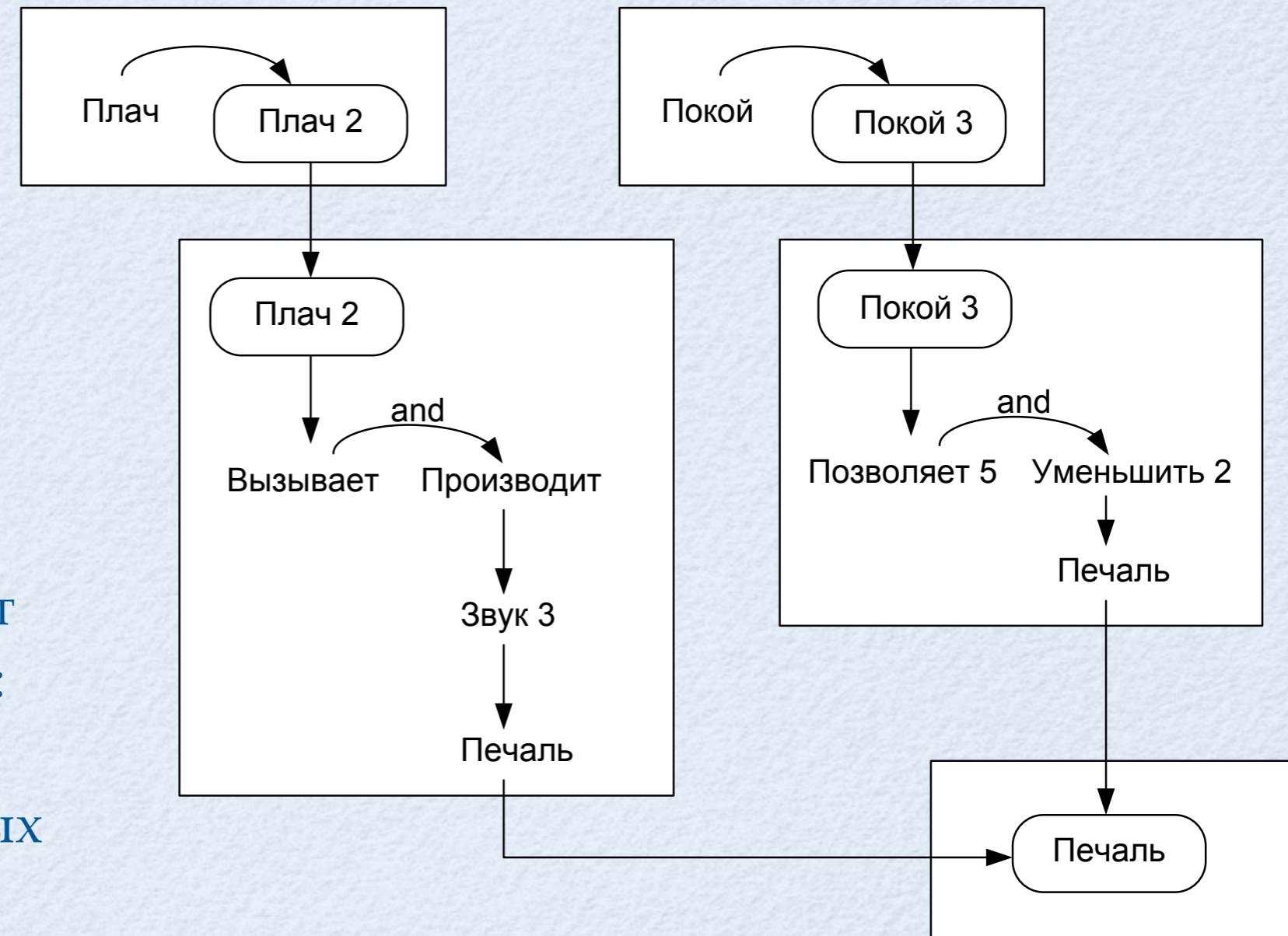
Когнитивная Экономия

- Передача свойств от определяющих понятий или типов к определяемому понятию (сегодня используется термин «наследование»);

Определим термин «машина» как конструкцию, состоящую из связанных компонентов, выполняющих некоторую работу. Это определение требует связать тип «машина» с лексемами «конструкция» и «компонент». Если теперь определить тип «компьютер», как разновидность «машины», то можно будет сказать, что компьютер является конструкцией из компонентов, выполняющих определенную работу.

Пример работы программы (1961)

- Программа осуществляла поиск в базе знаний отношений между парами слов, пытаясь определить общее определяющее понятие или узел пересечения.
- Эта программа смогла отыскать пересечение понятий плач и комфорт и заключила следующее: «Плач 2 связан с производством печальных звуков. Покой 3 может уменьшить печаль»



Психологические теории и Эксперименты

- Интеллектуальные функции человека подразумевают существование ассоциативной сети, в которой одни понятия соединяются с другими, Гордона Бауэра (Gordon H. Bower), 1979. Эта сеть своего рода «метауровень», отвечающий за отбор, организацию и преобразование информации.
- Теория распространения активации, Коллинз и Элизабет Лофтус (Loftus), предполагает, что в сети ассоциаций связи имеют различную «длину». Более короткие связи соответствуют более прочной связи между понятиями, а более длинные — менее сильной связи. Понятие становится более доступным после предъявления связанного с ним подготавливающего стимула или какого-либо другого слова. Например, при предъявлении зеленого цвета, вероятнее, что человек опознает слово «зеленый» быстрее. Более того, при предъявлении зеленого цвета, опознавание таких слов, как «трава» или более отдаленной ассоциации — «лужайка» происходит быстрее, чем при отсутствии подготавливающего стимула.

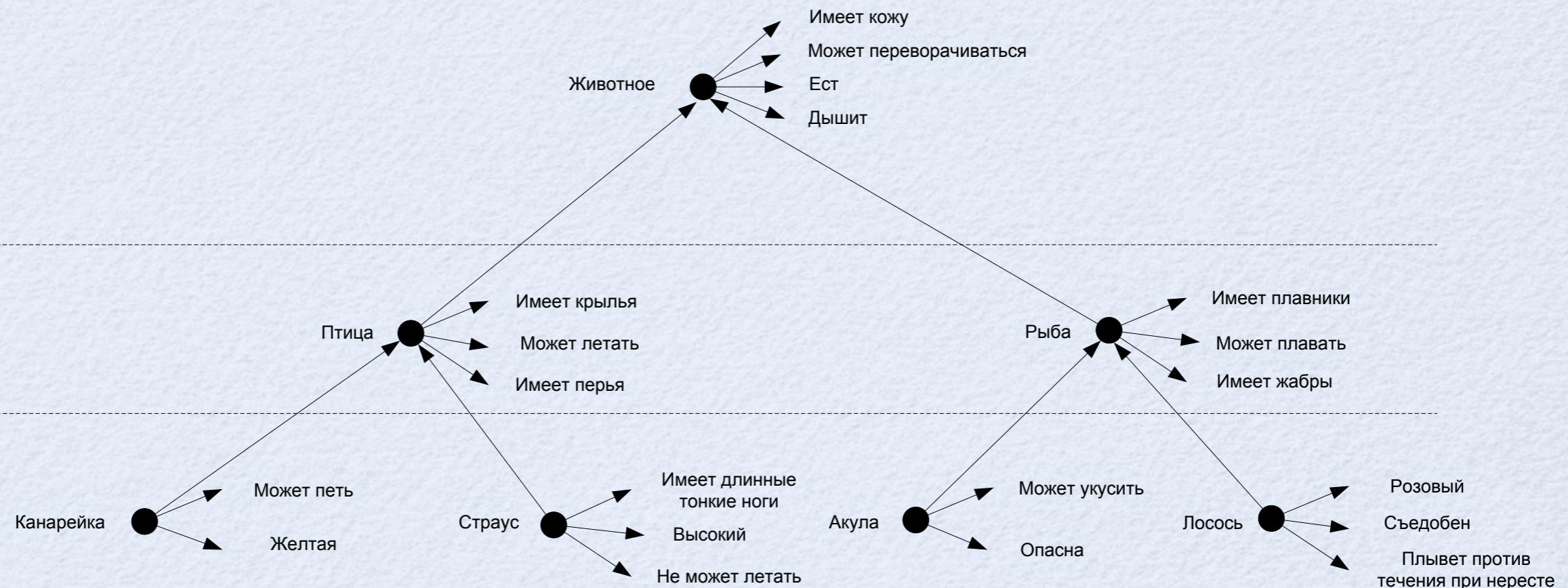
Психологические теории и эксперименты

Коллинзом и Квиллианом были исследованы вопросы хранения и скорости извлечения информации из памяти человека, 1968. Лабораторные эксперименты показали, что время реакции людей на простые вопросы типа «Канарайка – это птица?», «Канарайка может летать?» или «Канарайка может петь?» отличается.

Уровень 2

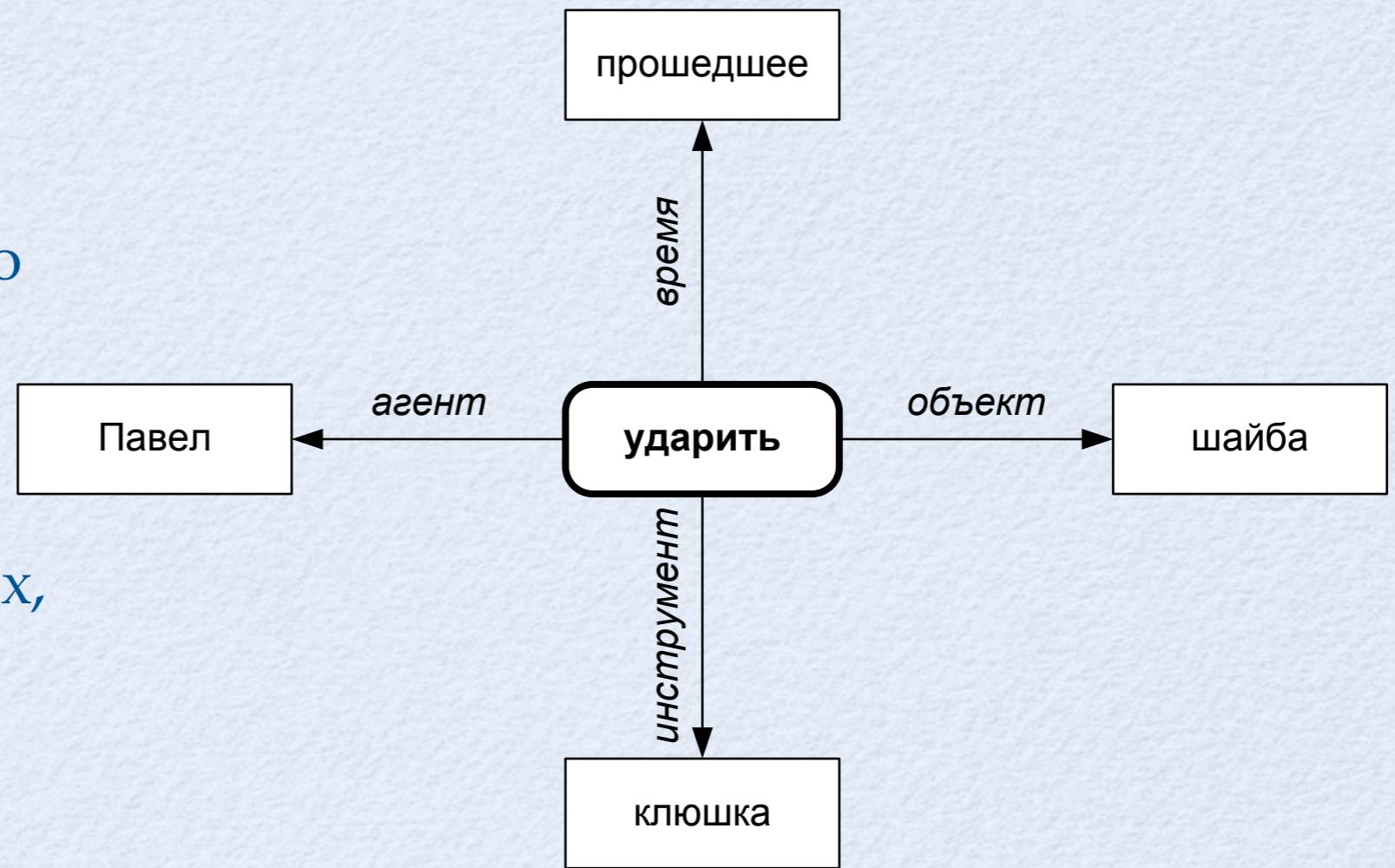
Уровень 1

Уровень 0



Пример падежного фрейма (case frame)

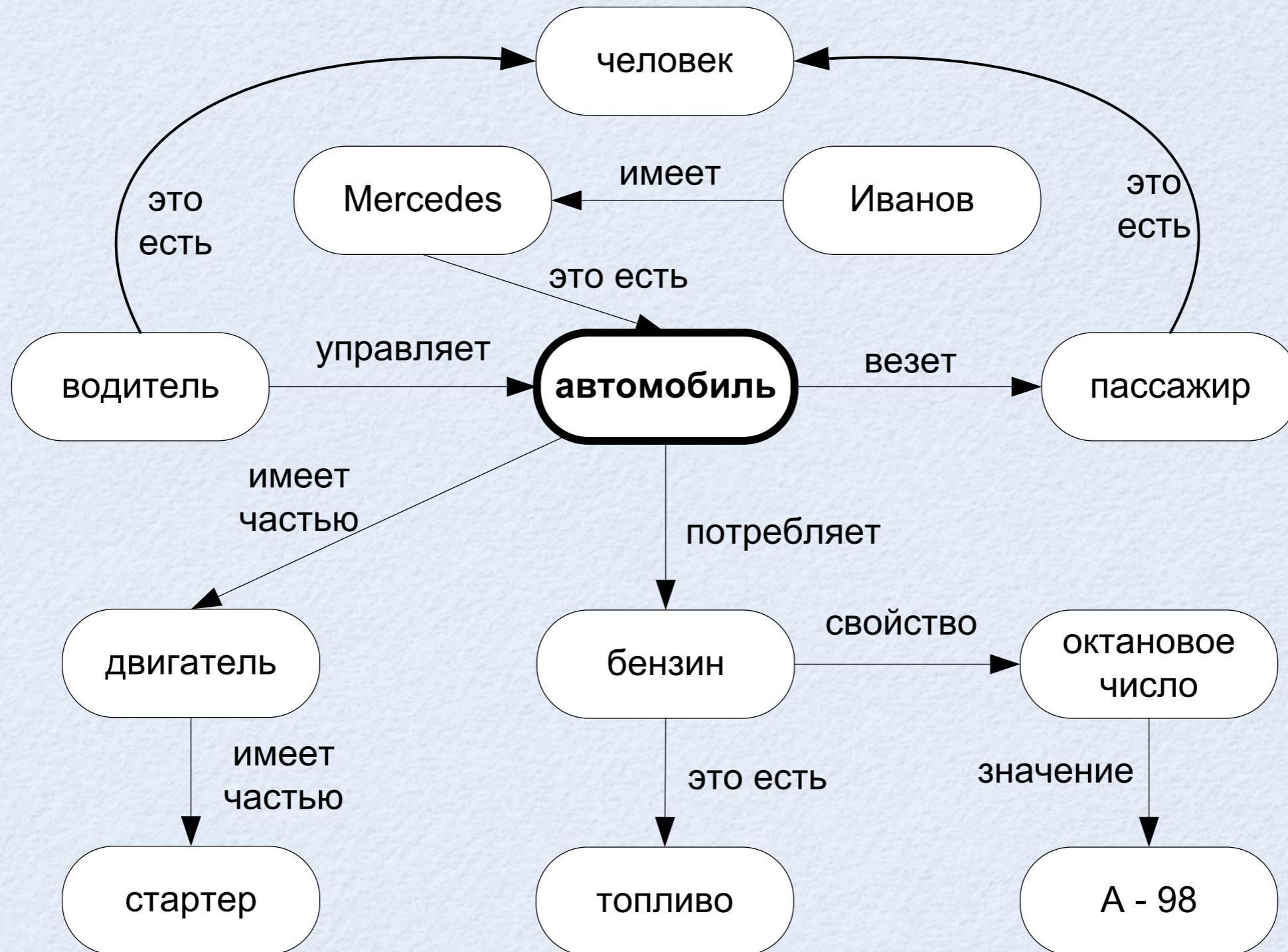
- Филмором (Fillmore), 1968, предложена сеть, в которой отношения определяются на основе грамматики английского языка. Связи соответствуют роли существительного или группы существительных, входящих в заданное предложение. К числу возможных ролей относятся агент, объект, инструмент, время и место.



Набор наиболее используемых отношений в семантической сети

- связи, определяющие тип объектов ("это есть" или "класс-подкласс", "иметь частью" или "часть-целое", "принадлежать" или "элемент-множество" и т.п.);
- функциональные связи (определяемые обычно глаголами "производит", "влияет" ...);
- количественные ("больше", "меньше", "равно" ...);
- пространственные ("далеко от", "близко от", "за", "под", "над" ...);
- временные ("раньше", "позже", "в течение" ...);
- атрибутивные связи (иметь свойство, иметь значение...);
- логические связи ("и", "или", "не") и др.

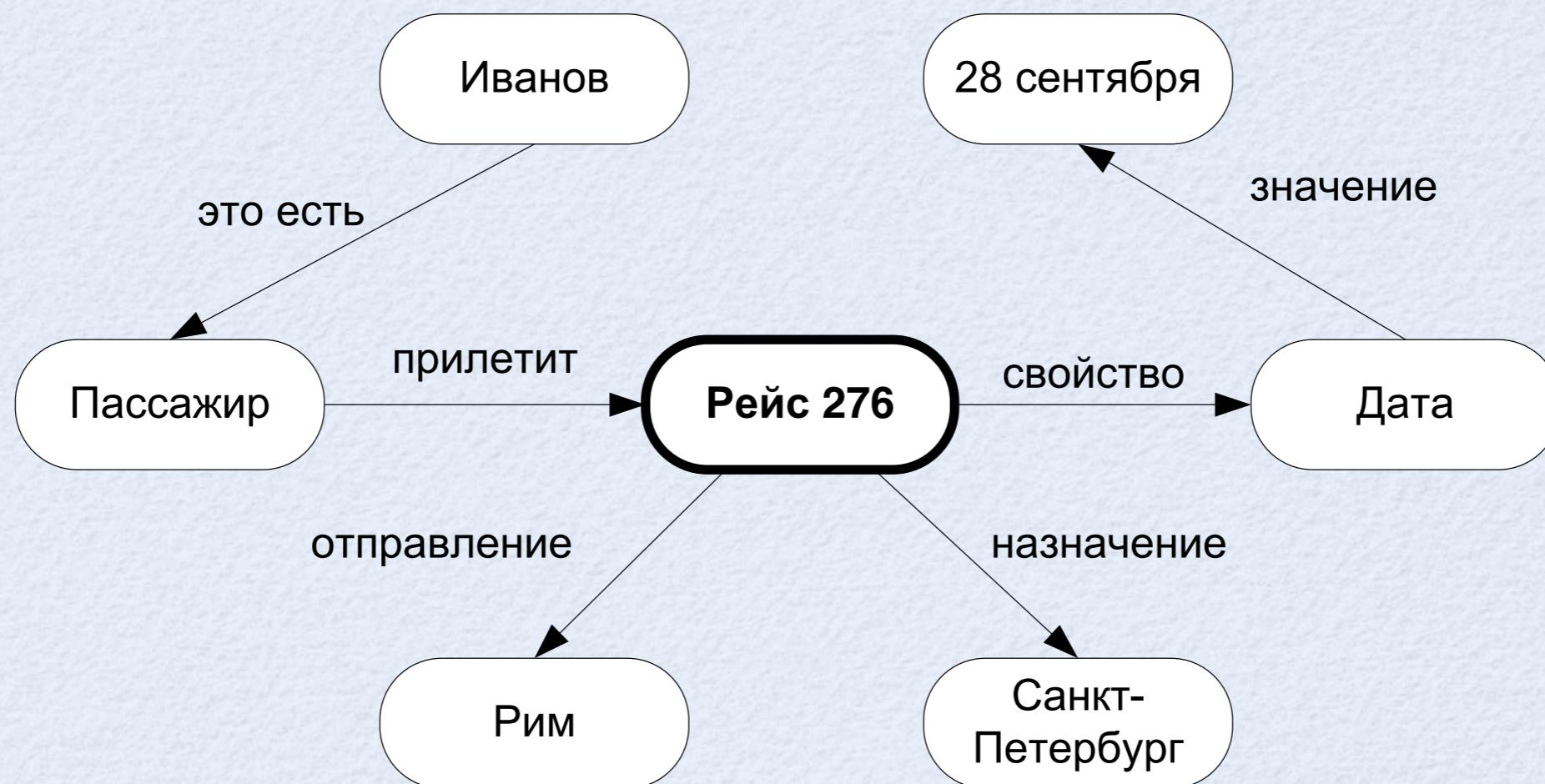
Пример простейшей семантической сети



Представление n -арных отношений на семантической сети

«Иванов прилетит из Рима в Санкт-Петербург 28 сентября»

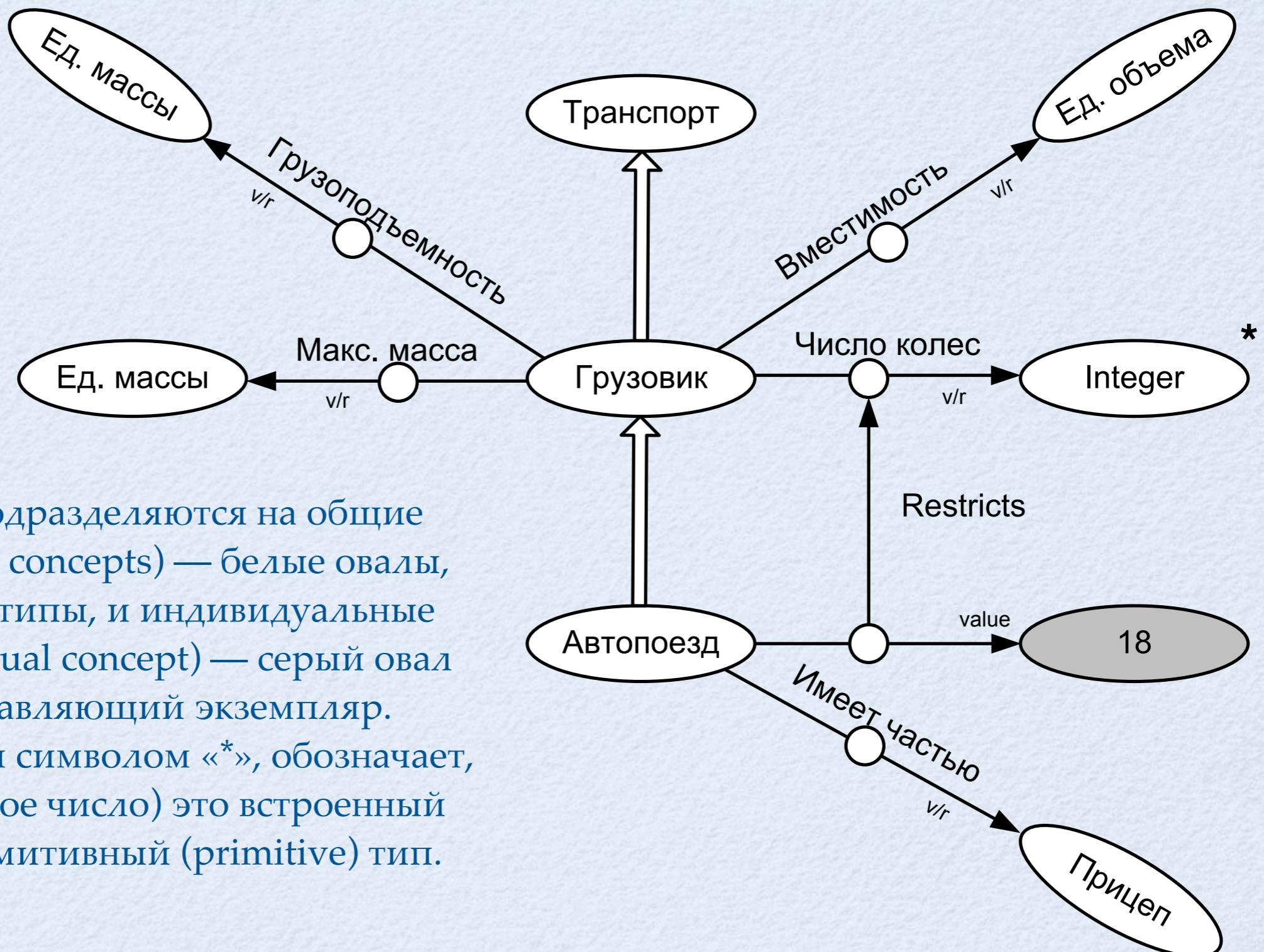
Запись с помощью 4-х местного предиката:
прилетит (Иванов, Рим, Санкт-Петербург, 28 сентября)



Сетевые языки представления смысла выражений

- *ассоциативные сети*, Г.С. Цейтин, 1985
- *релативные графы*, Parker-Rhodes, 1978
- SNOOP
- *декларативные сети*, Brachman, 1979
- *пропозициональные семантические сети*, Shapiro, 1971
- *казуальные сети*, Rieger, 1976
- *концептуальные графы*, Sowa, 84

Пример сети, определенной в KL-ONE (дефинитивные сети)

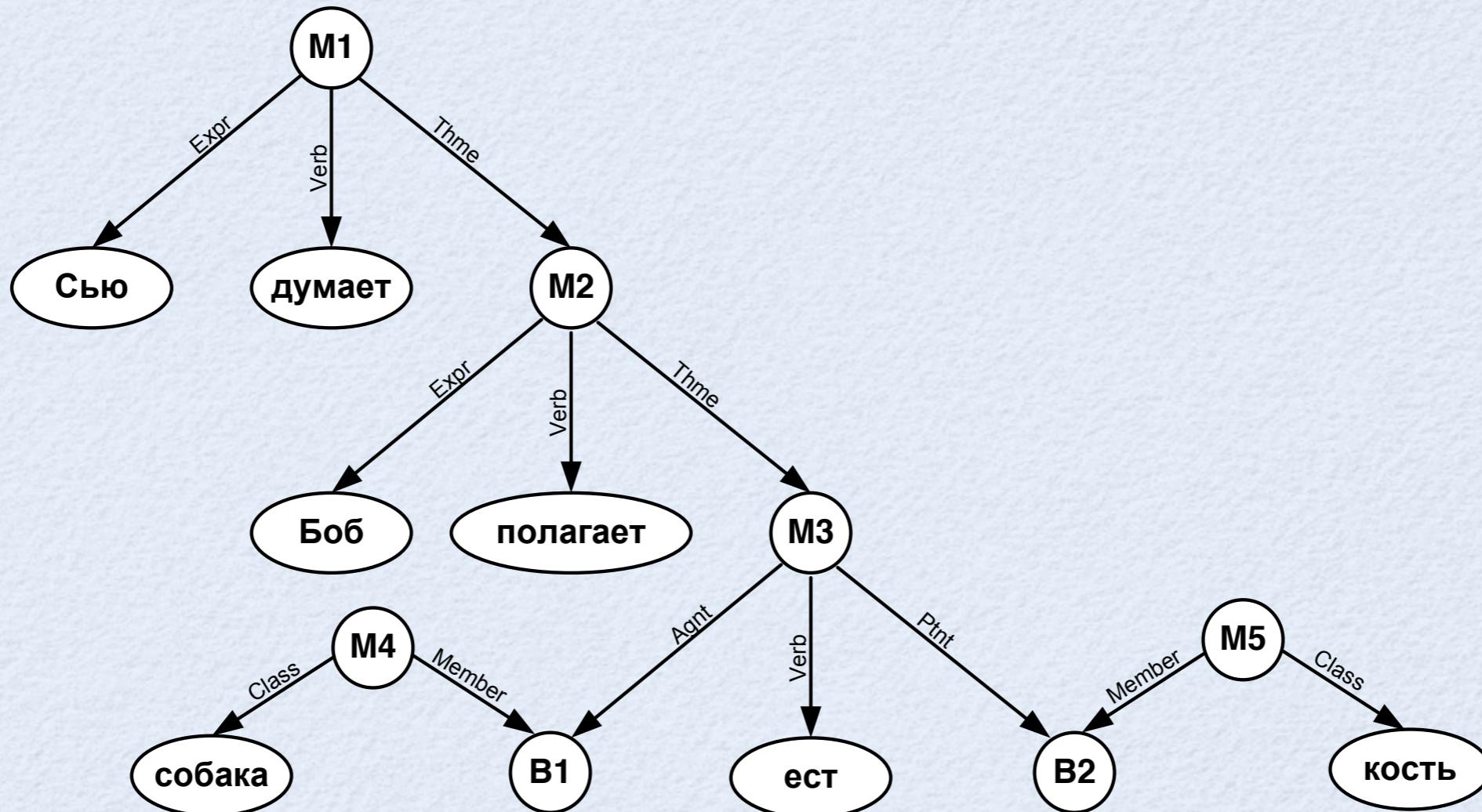


Узлы этой сети подразделяются на общие концепты (generic concepts) — белые овалы, представляющие типы, и индивидуальные концепты (individual concept) — серый овал (число 18), представляющий экземпляр.

Овал, отмеченный символом «*», обозначает, что «Integer» (Целое число) это встроенный (built-in) или примитивный (primitive) тип.

Пример сети в системе SNePS (пропозициональные семантические сети)

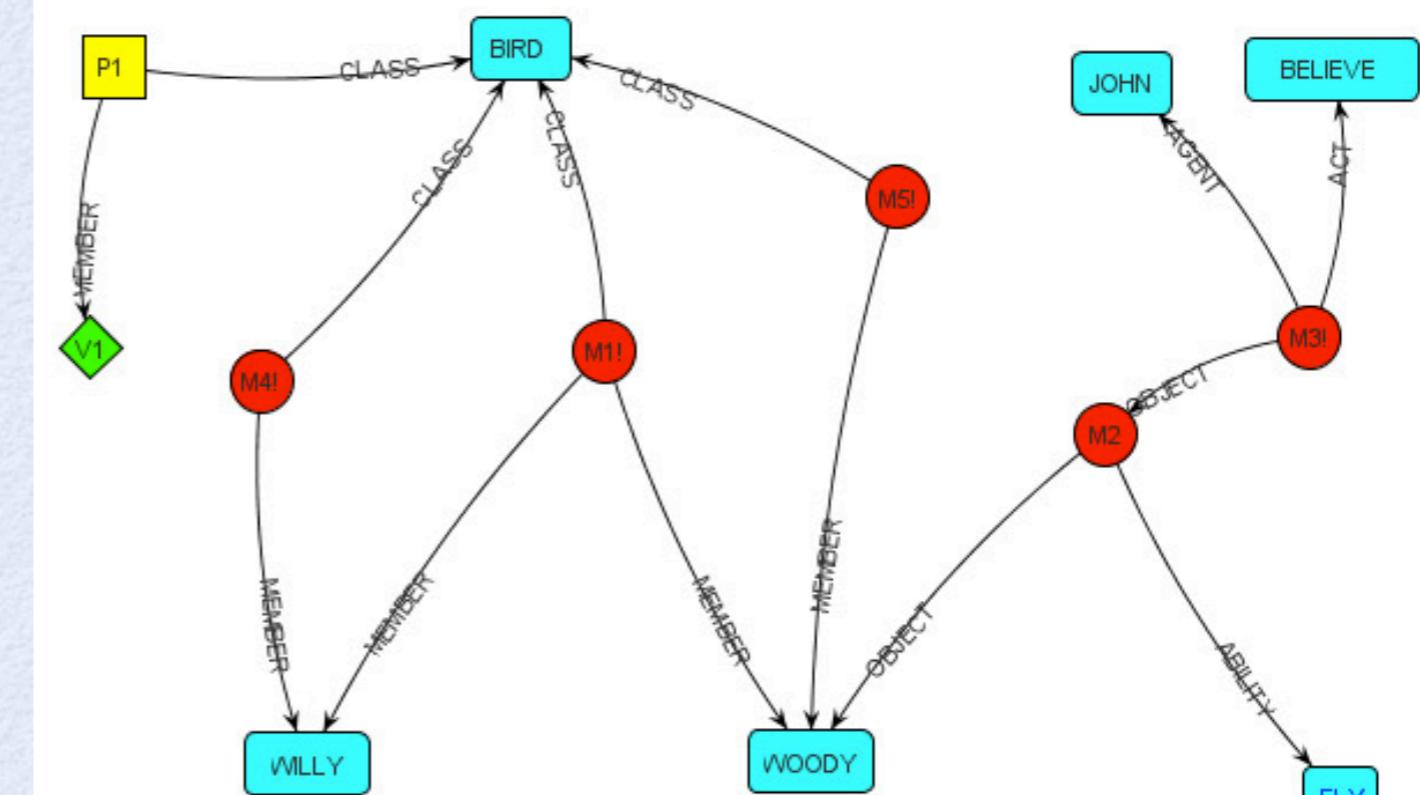
Система Semantic Network Processing System (SNePS) Стюарта Шапиро (Stuart C. Shapiro) предназначена для представления семантики естественного языка. Предложение «Сью думает, что Боб полагает, что собака ест кость» - каждый из узлов, помеченных от M1 до M5, представляет отдельное предложение, относительное содержание которого соотнесено к пропозициальному узлу (propositional node).



Пример сети в системе SNePS (продолжение)

Предложение М1 утверждает, что «Сью» — это потребитель (experiencer — Expr) глагола (verb) «думать», темой (theme — Thme) которого является другое предложение М2. Для М2 потребителем является «Боб», глаголом — «полагает», а темой — предложение М3. В предложении М3 присутствует агент (agent — Agnt) — некое существо В1, которое является экземпляром (member) класса (class) «собака», глагол «ест» и пациент (patient — Ptnt) — существо В2, которое является экземпляром класса «кость». Данный пример иллюстрирует, как предложения могут быть рассмотрены на метауровне с целью получения других утверждений: М1 утверждает, что М2 есть предмет размышлений Сью, а М2, в свою очередь, утверждает, что М3 — это то, что предположил Боб.

БЗ системы SNePS



```
(assert member Tom class cat)
(assert member Flounder class fish)
(assert member Tary class bird)
(assert member cat class animal)
(assert member fish class animal)
(assert member bird class animal)
```

Концептуальные графы (conceptual graph)

- Универсальный сетевой язык для представления смысла
- Впервые предложены в работе Джона Сова (John Sowa), 1984
- Ресурсы: <http://www.jfsowa.com/cg/> или <http://conceptualgraphs.org/>



Возникновение идеи концептуальных графов

- 1) Является развитием формализма
экзистенциальных графов и графической
логики (Ч. Пирс);
- 2) Построение механизма, обладающего
выразительностью естественного языка и
вычислительными возможностями
символической логики;
- 3) Предназначены для реализации различных
сетевых способов представления семантики.

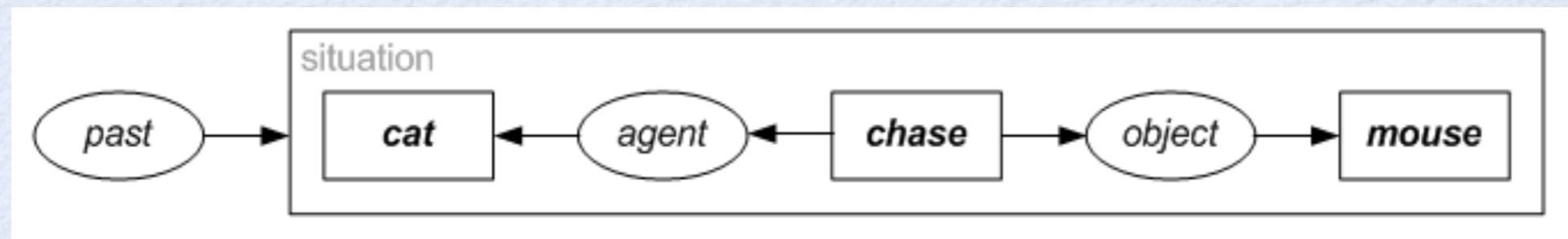
Определение концептуальных графов

- Концептуальный граф (conceptual graph) — это конечный связный двудольный граф;
- Используются узлы двух видов: первые представляют понятия, а вторые — концептуальные отношения (conceptual relation);
- Метки дуг на графах не используются;
- Для отражения взаимосвязи между понятиями используются вершины, представляющие концептуальные отношения;
- Чтобы различать вершины понятий и отношений используется графическое обозначение двух видов: прямоугольники — для вершин—понятий и эллипсы — для вершин отношений

Пример простого концептуального графа

Предложение «*A cat chased mouse*»:

- Четыре прямоугольника соответствуют концептам: кот, преследовать, мышь и факт преследования, имеющий неявный тип «ситуация» («situation»);
- Эллипсы соответствуют отношениям: «агент» («agent»), «объект» («object») и «время действия» («past»);
- Последнее отношение присоединяется к графу-контексту, т.е. графу, который содержит вложенный в него подграф;



Представление концептуального графа и в текстовой форме:

```
(PAST) -> [ [CAT] <- (AGENT) <- [CHASE] -> (OBJECT) -> [MOUSE] ]
```

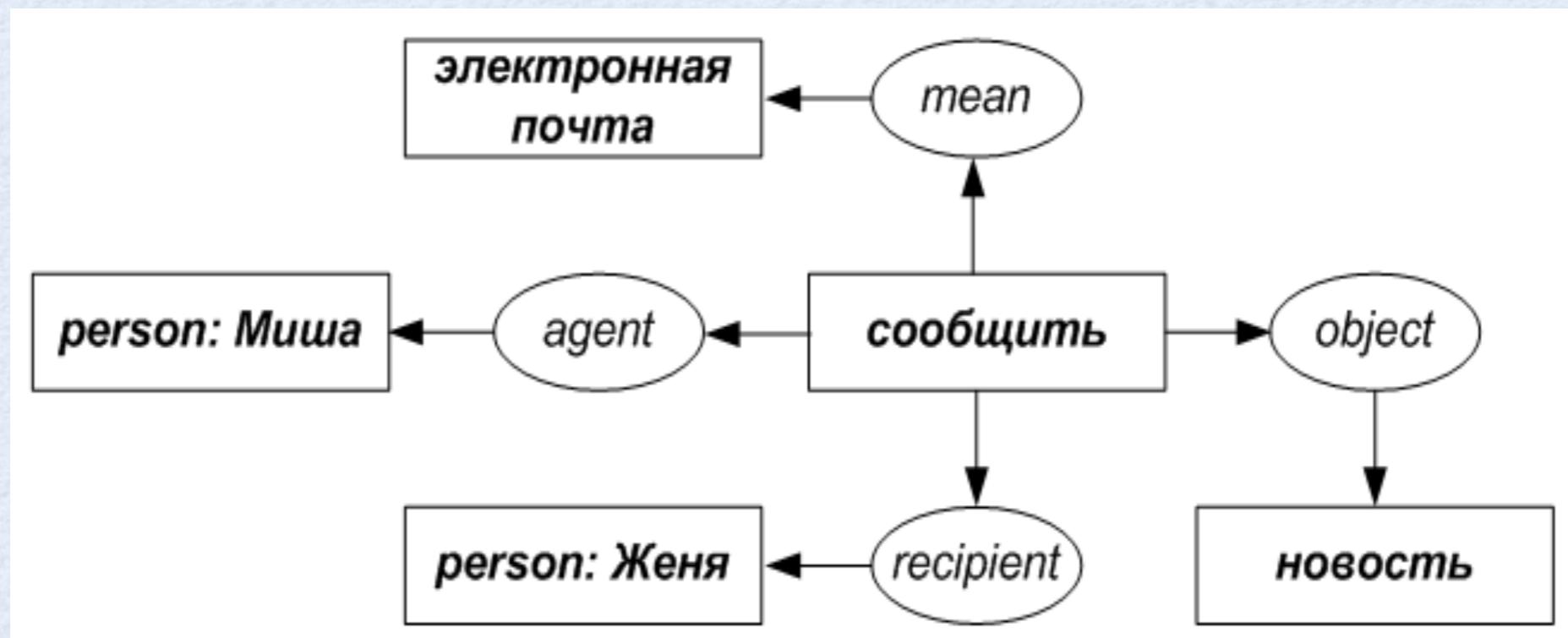
Представления отношений любой арности

- В отличие от дуг, соответствующих бинарным отношениям, вершины, представляющие концептуальные отношения, могут быть связаны с любым конечным количеством вершин–понятий.
- Аналогично какое-либо понятие может быть связано множеством отношений.

Пример n-арных отношений

Пример: «Миша сообщил Жене новость по электронной почте».

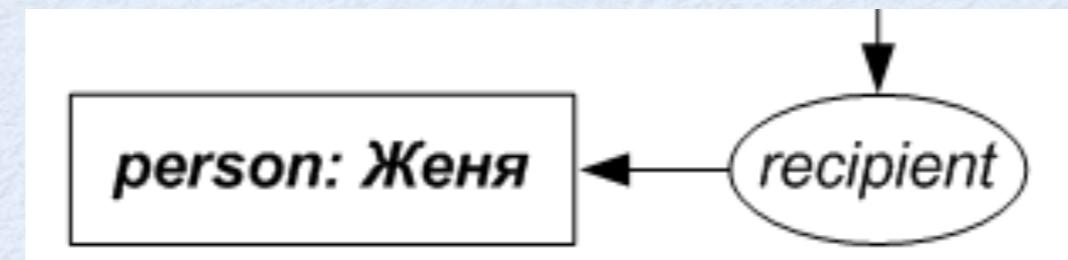
Глагол «сообщить» связан с другими понятиями с помощью четырех концептуальных отношений: «агент» («agent»), «объект» («object»), «получатель» («recipient») и «средство» («mean»).



Типы в концептуальных графах

В нотации концептуальных графов вводятся явные обозначения отношений класс–элемент, класс–подкласс. Любое понятие рассматривается как экземпляр конкретного типа.

Обозначение: Вершина «Женя» имеет тип «*person*».



Тот факт, что один тип является подклассом другого типа
принято обозначать символом \leq :

person \leq homo sapiens \leq being

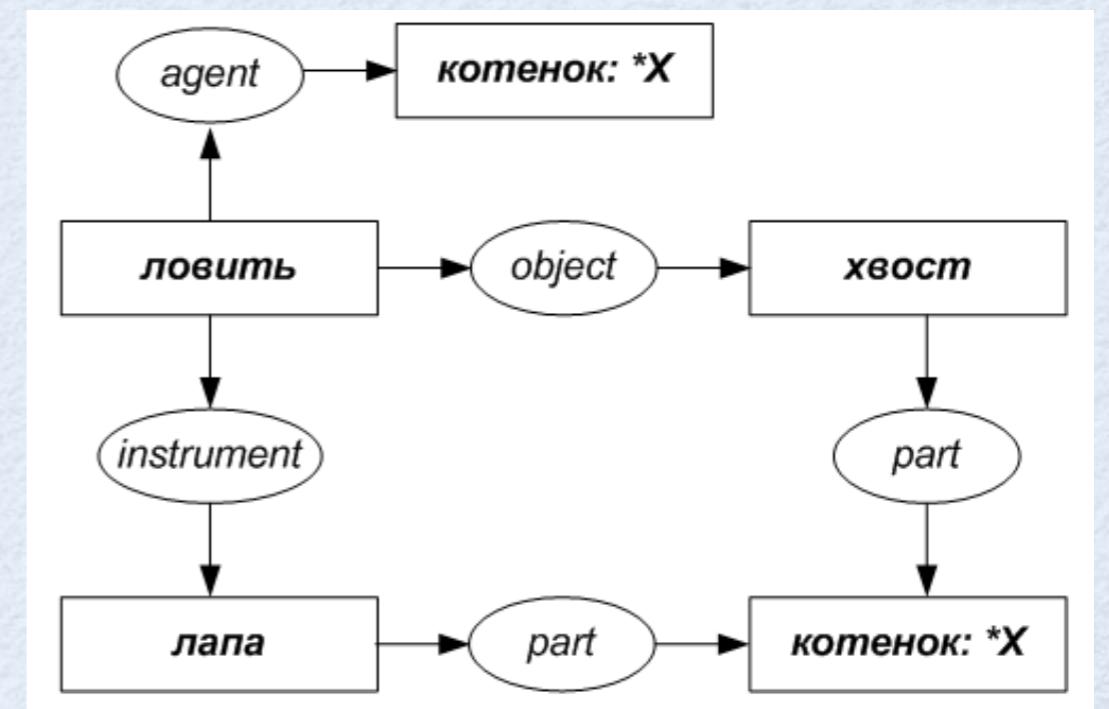
Решетка (иерархия) наследования

- 1)Совокупность всех типов образует решетку множественного наследования, в которой каждый тип может иметь множество родителей и детей.
- 2)Чтобы решетка типов была единой иерархией, в концептуальные графы включают два специальных типа: *универсальный тип* (universal type), являющийся суперклассом для всех классов, и *абсурдный тип* (absurd type), являющийся подклассом для всех типов.
- 3)Универсальный тип принято обозначать символом «Т», а абсурдный тип — символом « \perp ».

Обобщенный маркер «*»

- Для обозначения любого или неспецифицированного экземпляра используется обобщенный маркер «*».
- Метки понятий «*person: **» и «*person*» являются эквивалентными и указывают на произвольный объект типа *person*.
- В дополнение к обобщенному маркеру «*» допускается использование переменных, например «*person: *X*».
- Для более сложных манипуляций с неспецифицированными экземплярами классов используются лямбда выражения.

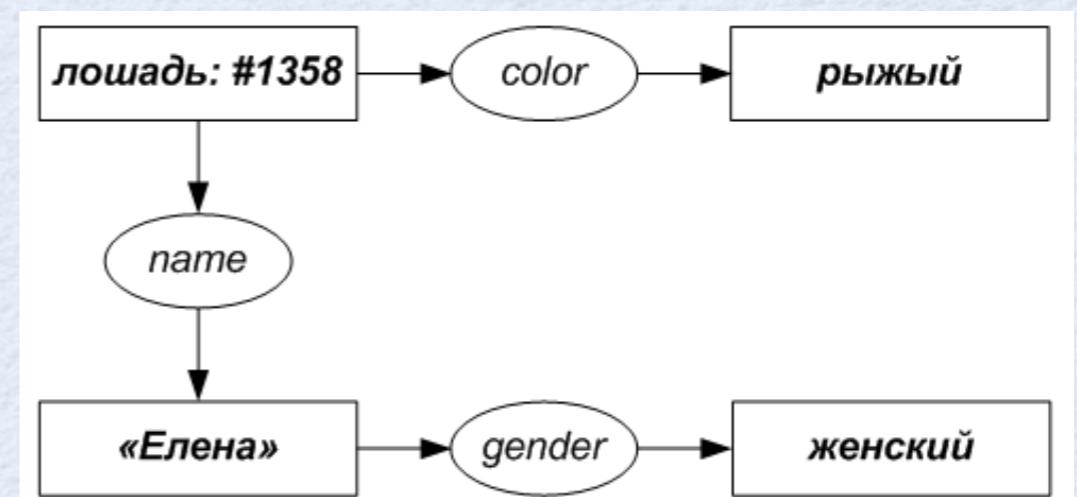
Пример: «Котенок ловит лапой свой хвост». Не известно, о каком конкретно котенке идет речь. Переменная **X* позволяет указать, что лапа и хвост принадлежат одному и тому же котенку.



Индексный маркер «#»

- Для указания конкретного экземпляра класса, не используя при этом имени объекта служат индексные маркеры «#»
- Позволяющие отделить экземпляры от своих имен.
- Для указания имени объекта можно использовать отношение «name». Это позволяет, с одной стороны, указать для одного объекта несколько имен, а с другой стороны — обозначать различные объекты одним и тем же именем.

Пример: «Рыжую лошадь звали женским именем Елена». Речь идет о свойстве имени какой-то конкретной лошади «быть женским». Отношение «gender» («род») указывает на свойство имени, а отношение «color» («цвет») — на свойство конкретной лошади.



Канонические правила формирования (canonical formation rules)

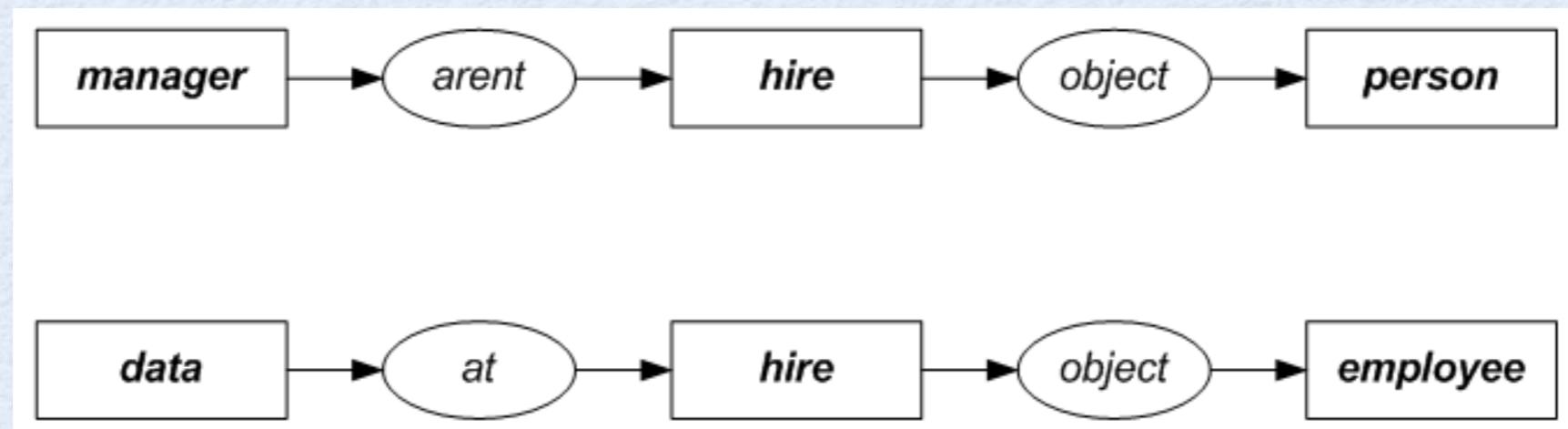
- *Копирование* (copy) — позволяет получить точную копию какого-либо графа.
- *Упрощение* (detach) — позволяет исключить дублирующиеся отношения. Дублирование отношений часто возникает в результате операции объединения.
- *Ограничение* (restrict) — позволяет заменить вершины понятий графа другими вершинами, представляющими их специализацию, или заменить метку типа на метку подтипа.
- *Объединение* (join) — позволяет интегрировать два графа в один, если одна из вершин первого графа, представляющая какое-либо понятие, идентична одной из вершин второго графа.

Не являясь правилами вывода, канонические правила формирования накладывают определенные ограничения на операции с графиками и не позволяют формировать бессмысленные графы на основе осмысленных.

Пример специализации и обобщения концептуальных графов

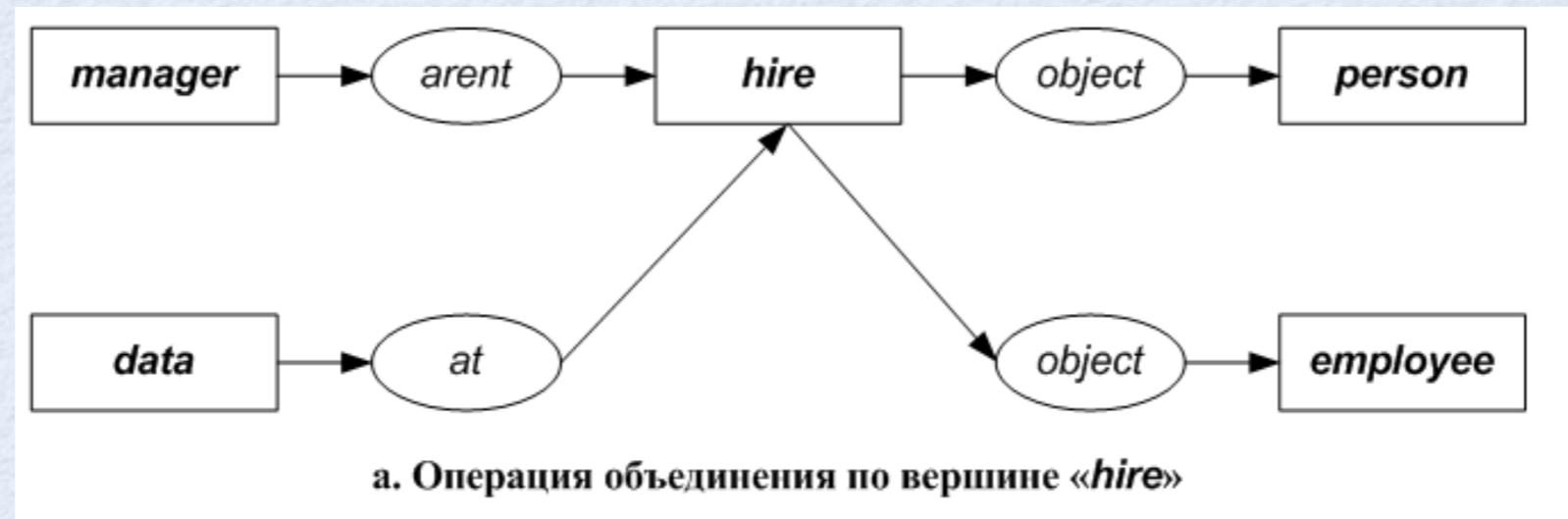
Два исходных предложения «*Manager hired a person*» («Менеджер нанял на работу человека») и «*Employee hired at data (13/12/06)*» («Работник принят на работу 13/12/06»).

Типы концептов в данном примере образуют следующую иерархию классов «*manager* \leq *employee* \leq *person*»: тип «*person*» является более общим (т.е. суперклассом) по отношению к «*employee*» и «*manager*», тип «*manager*» является подклассом типа «*employee*».



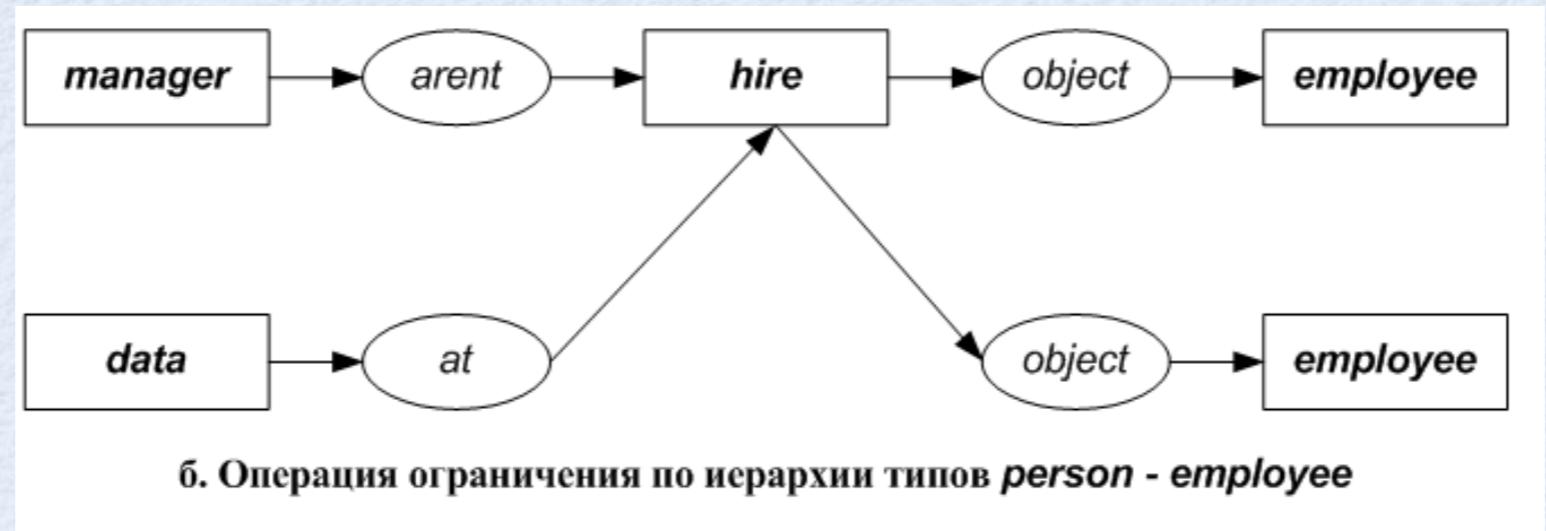
Продолжение примера

Оба исходных графа содержат вершину «*hire*». Применив правило объединения к этим двум графикам, по вершине «*hire*» получим новый граф:



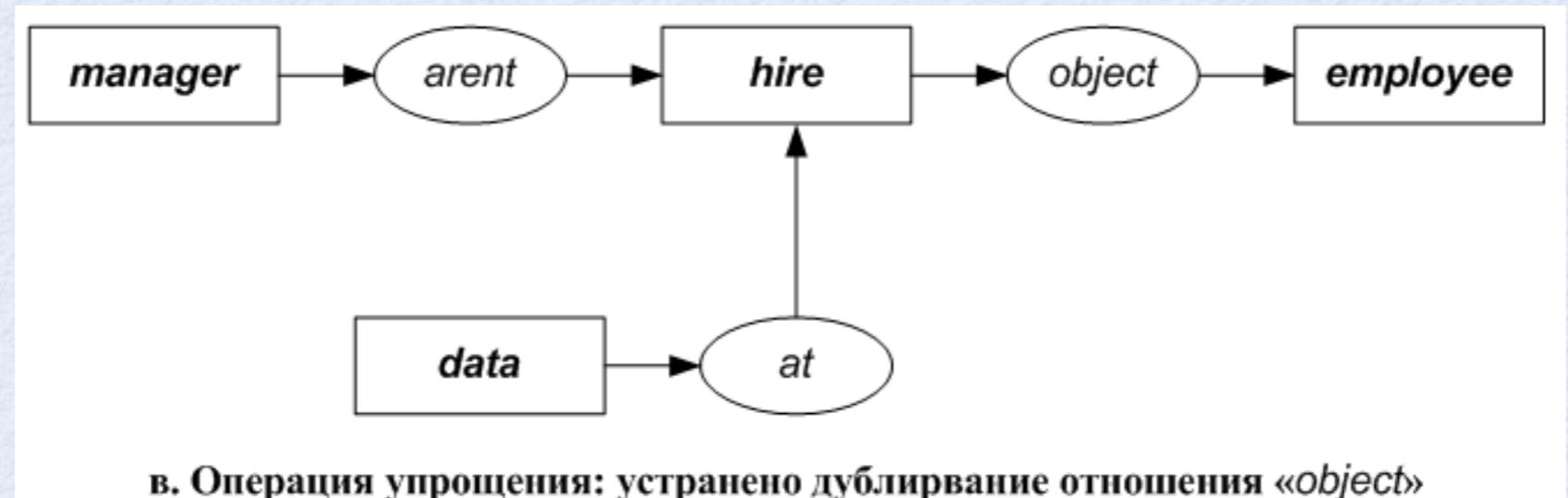
Продолжение примера

На основании отношения класс–подкласс между концептами *person* и *employee* можно применить правило ограничения и заменить метку вершины «*person*» на метку «*employee*»:



Продолжение примера

Полученный граф содержит две вершины «*employee*», которые могут быть объединены в одну, т.к. указывают на один и тот же концепт. Операция упрощения позволяет устраниТЬ дублирование отношения «*object*» между концептами «*hire*» и «*employee*».



Пропозициональные понятия

- Часто возникает необходимость для определения отношений между высказываниями или высказыванием и концептом.
- Например, предложение «Tom believes that Marry like him» («Том верит, что он нравится Мэри») устанавливает отношение между концептом «believe» и высказыванием «Marry like Tom».
- В концептуальных графах вводится специальный тип высказывание (proposition), объектом ссылки которого является множество концептуальных графов (являющихся подграфами данного графа), обозначаемое прямоугольниками, содержащими подграф.

