Экспертные системы и базы знаний

Тема 1. Тест Тьюринга, основные постулаты ИИ. Нейроинформатика

Постулат нейроинформатики: Единственный объект, способный мыслить — это человеческий мозг. Поэтому любое «мыслящее» устройство должно каким-либо образом воспроизводить структуру человеческого мозга

Постулат кибернетики*:* Для искусственного интеллекта не важно, как устроено «мыслящее» устройство. Главное, чтобы на заданные воздействия оно реагировало бы так же, как и мозг человека.

Научные школы в ИИ: две научные школы: конвенциональный и вычислительный интеллект.

1. ***Конвенциональный ИИ***.:

• Экспертные системы (программы основаны на базе знаний и механизмах вывода по определенным правилам, обрабатывают информацию и выдают заключение на её основе);

• Рассуждения на основе аналогий;

• Байесовские сети (статистические метод анализа данных);

• Поведенческие модели (включают программы-агенты, поведение которых зависит от изменений внешней среды).

2. ***Вычислительный ИИ.***:

• Нейросети;

• Нечеткие системы (рассуждения в условиях НЕ-факторов);

• Эволюционные вычисления (эволюционные алгоритмы, разновидности Монте-Карло, гармоничный поиск, роевые алгоритмы, меметика и т.д.);

• Гибридные системы (например, исходные экспертные правила генерируются нейросетью, а заключения - байесовской сетью).

Современное состояние:

- ***Нейросетевой подход*** привлекателен.

Трудности связаны с необходимостью переобучение сети, локальной оптимальностью решений, выбором адекватной архитектуры сети и, главное, нельзя выяснить, как было получено определенное решение задачи.

- ***Разработка гибридных моделей*** и алгоритмов вывода знаний в условиях неопределенности в реальном времени, онтологий знаний, многоагентных систем, роботов, игровых программ.

Значимые достижения в области теории:

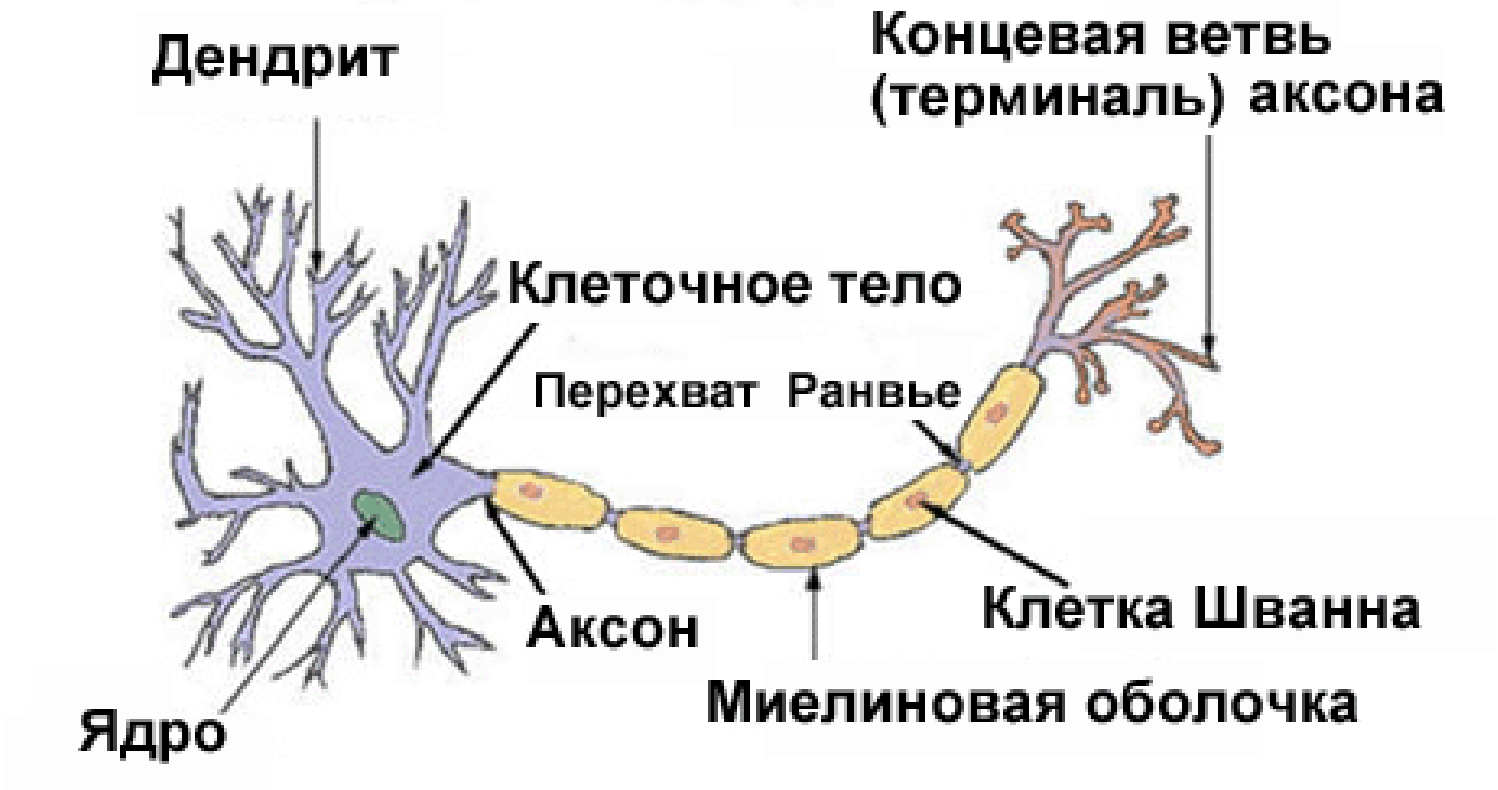
- Формирование нейроматематики, создание нейроалгоритмов и нейрокомпьютеров на их основе, разработку «мягких» вычислительных моделей мозга, связанных с восприятием, языком и знаниями;

Сферы практического применения:

- Системы распознавания; прогнозирование в бизнесе, экономике, медицине; управление в космосе, робототехнике, металлургии, машиностроении, нефте- и газовой промышленности; решение трудных задач моделирования в реальном времени;

- Экспертные системы в медицине, геологии, бизнесе, экономике, промышленности; системы управления знаниями и принятия решений, обучающие системы, системы интеллектуального анализа данных, семантический анализ и обработка информации на естественном языке; визуализация и интеллектуализация интерфейса.

Количество нейронов мозга составляет около 10^{21}.



Объем коры: толщина 2-3 мм, площадь около 2200 $см^2$.

Потребляемая мощность: 1 – 25 Вт.

Нейрон имеет в среднем 10000 связей.

100 триллионов синапсов.

Огромные объемы информации мозг обрабатывает за доли секунды при том,

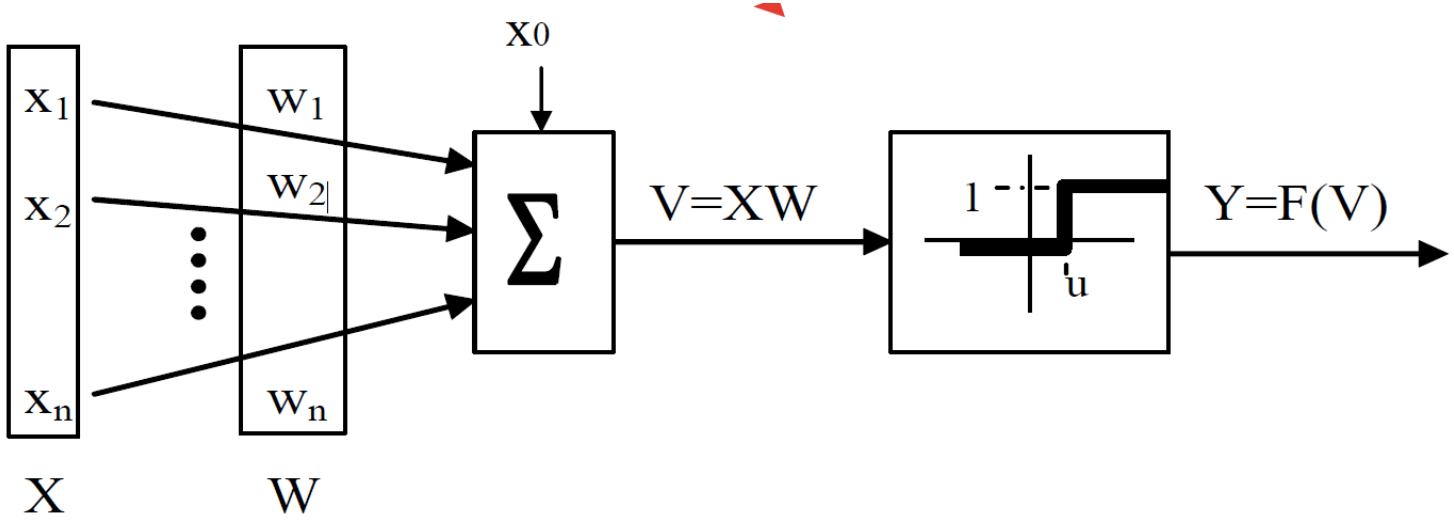
что мозг – тихоход (время реакции нейрона несколько миллисекунд).

Объем передаваемой мозгом информации равен:

1012 (волокон)×103 (импульсов) = 1015.

В лучших ЭВМ: 10^3 (каналов)×10^9 (импульсов в сек) = 10^{12}, т.е. 1000 лучших ЭВМ равнозначны мозгу по производительности.

Искусственный нейрон МакКаллока - Питтса



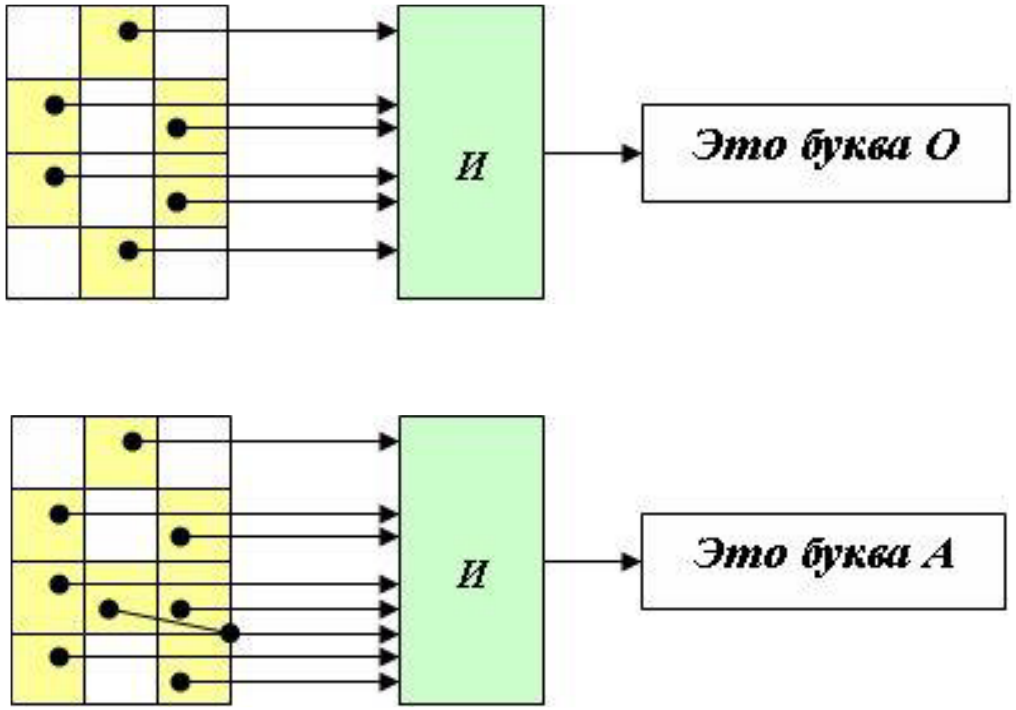
$$Y=F(V) = F(\Sigma x_j w_j>\theta),\ j = 1, 2,\ldots, n,$$

Y – выходной сигнал нейрона, ***F*** – функция активации выхода нейрона, w_j – «вес» входа x_j, **θ** - пороговое значение.

Добавив постоянный единичный вход $x_0=1$ и положив $w_0= -\theta$, получим:

$$Y= F(\Sigma x_j w_j + x_0),\ j=0, 1, 2,\ldots, n.$$

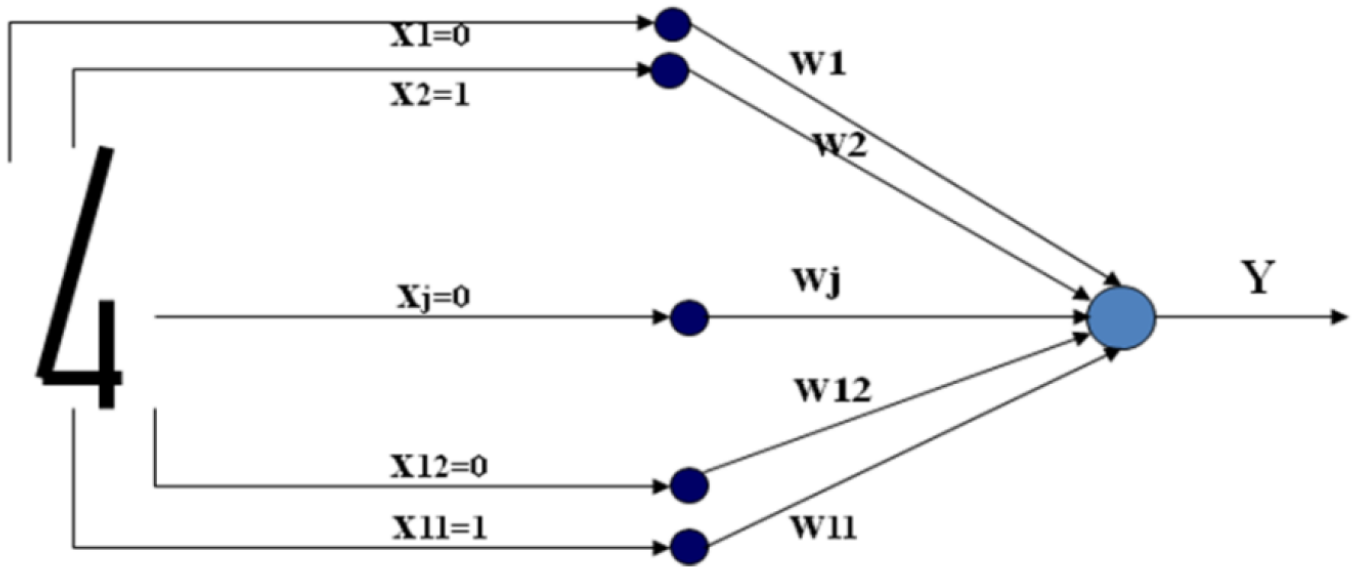
Нейронная сеть для распознавания символов



(1,2)& (2,1)& (2,3)& (3,1)& (3,3)& (4,2) → ***О***;

(1,2)& (2,1)& (2,3)& (3,1)& (3,2)& (3,3)& (4,1)& (4,3)→ ***А***.

Персептрон Розенблатта и правило Хебба



Фотоэлемент срабатывает ($x_j=1$), если на него попадает фрагмент цифры, иначе – $x_j=0$.

Цель: $y=1$, если на карточке четная цифра; $y = 0$ – в противном случае.

Корректировка весов w_j по правилам Хебба:

- если выход неправильный и равен 0, то увеличить веса для $x_j=1$;
- если выход неправильный и равен 1, то уменьшить веса для $x_j=1$.

Теорема (о сходимости персептрона): Если существует множество значений весов, обеспечивающих распознавание образов, то в конечном итоге алгоритм его обучения приведет к этому множеству.

Функции активации:

Пороговая функция:

$$Y(x) = \begin{cases} 1, & если \ \ x \geq \Theta, \\ 0, & иначе. \end{cases}$$

Линейная функция активации:

$$Y(x) = \begin{cases} 0, & если \ \ x \leq -0{,}5 \\ 1, & если \ \ x \geq 0{,}5 \\ x, & иначе. \end{cases}$$

Логистическая функция активации:

$$Y(x) = \frac{1}{1 + \exp(-ax)},$$

a - параметр функции, определяющий её крутизну.

Когда ***a*** стремится к бесконечности, функция вырождается в пороговую.

При ***a = 0*** сигмоида вырождается в постоянную функцию со значением ***0,5***.

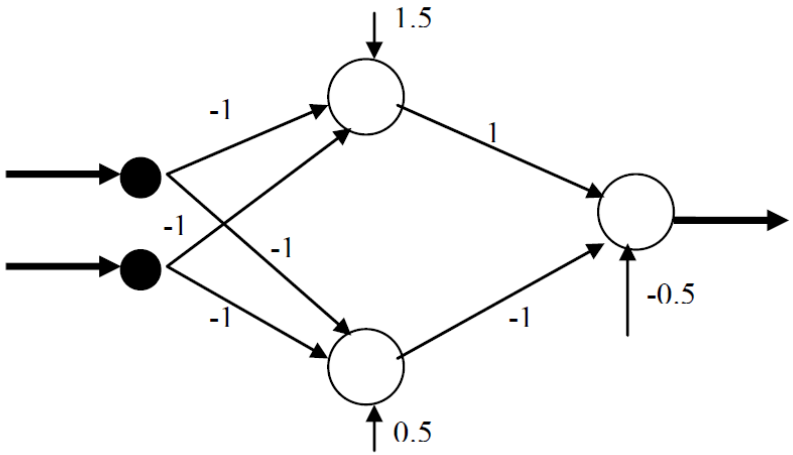
Достоинство логистической функции: простота её производной:

$$\frac{dY(x)}{dx} = tY(x)(1 - Y(x)).$$

Обучение персептронов: минимизировать среднеквадратичную ошибку

$$\varepsilon = \frac{1}{2}\sum_i (Y_{i\ эталон} - Y_{i\ факт})^2.$$

Пример:



Входы		Выход
x_1	x_2	XOR
1	1	0
1	0	1
0	1	1
0	0	0

Значение входов вычисляется по формуле:

$$V_j = \sum_{i=0}^{n} x_i w_{ij},$$

Выходное значение нейрона - результат применения пороговой функции:

$$Y(V_j) = \begin{cases} 1, & если\ V_j \geq 0, \\ 0, & если\ V_j < 0. \end{cases}$$

Вводимые данные - $x_1=1$, $x_2=1$.

Для первого скрытого элемента со смещением ***1.5*** получаем

$$V=(x_0\times 1.5)+(x_1\times(-1))+(x_2\times(-1)) = (1\times 1.5)+(1\times(-1))+(1\times(-1))= \mathbf{-0.5}.$$

Выходным значением элемента будет ***0***.

Для второго скрытого элемента со смещением ***0.5*** получаем

$$V=(x_0\times 0.5)+(x_1\times(-1))+(x_2\times(-1)) = (1\times 0.5)+(1\times(-1))+(1\times(-1))= \mathbf{-1.5}.$$

Выходным значением будет ***0***.

Общие ***признаки нейросетевой технологии***:

- Система начинает обнаруживать закономерности во входной информации.
- Система не знает, как она обучается - ей безразличен предмет рассуждений.
- Система легко доучивается и переучивается.

Нейросеть можно "недокормить" примерами, но можно и «перекормить» (переобучить).

Две парадигмы обучения нейронных сетей – с учителем и без учителя.

- В первом случае, на входной вектор имеется готовый ответ,
- Во втором случае нейронная сеть самообучается.

Однослойные искусственные нейронные сети

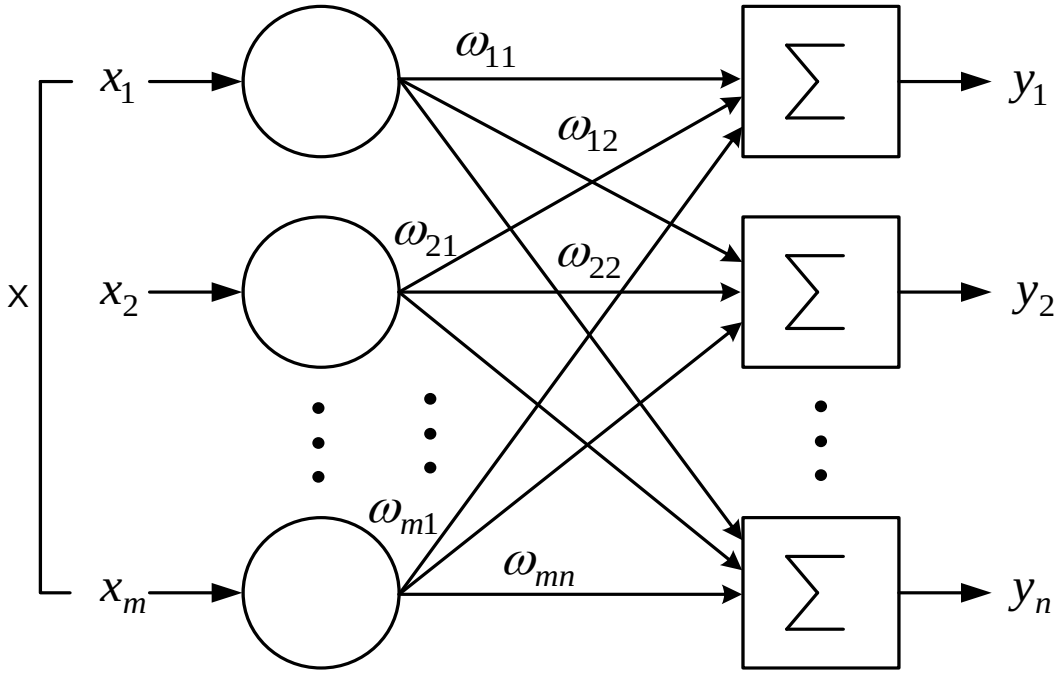


Рис. 1. Однослойная нейронная сеть

Вычисление выходного вектора ***Y*** сводится к матричному умножению: ***Y=X×W***.

Многослойные искусственные нейронные сети.

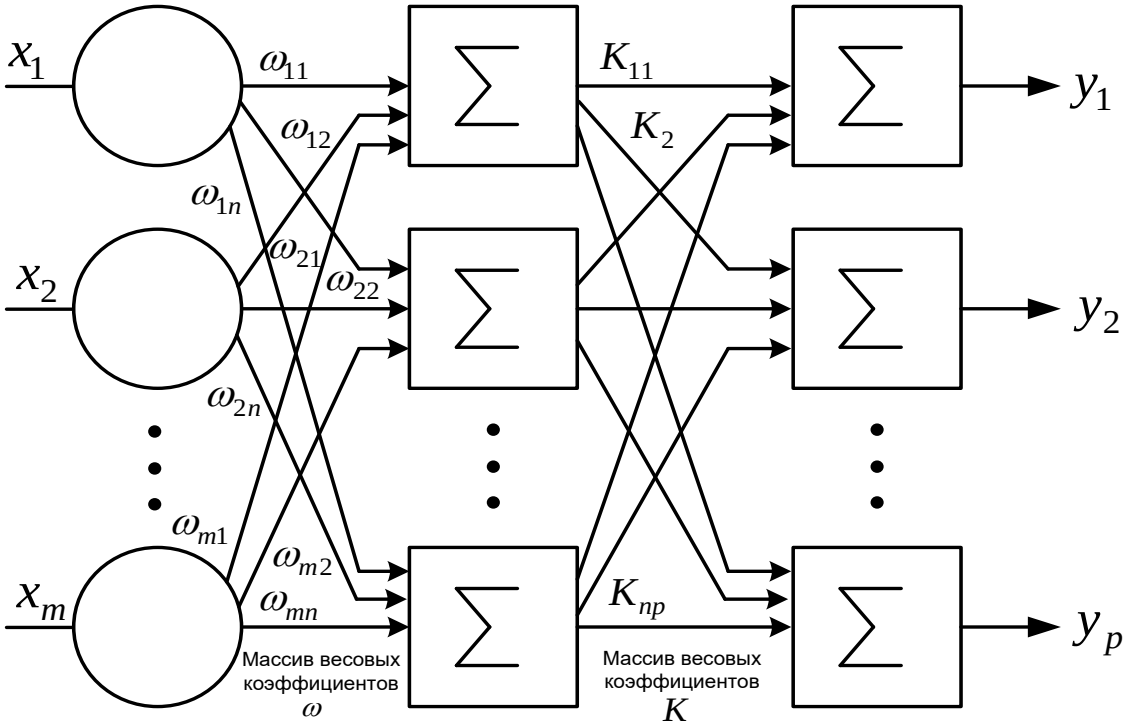


Рис. 2. Многослойная нейронная сеть

Обучение искусственных нейронных сетей

Обучение с учителем предполагает, что для каждого входного вектора существует целевой вектор, представляющий собой требуемый выход.

Обучение без учителя не нуждается в целевом векторе для выходов и, следовательно, не требует сравнения с предопределенными идеальными ответами.

3. Теорема Колмогорова

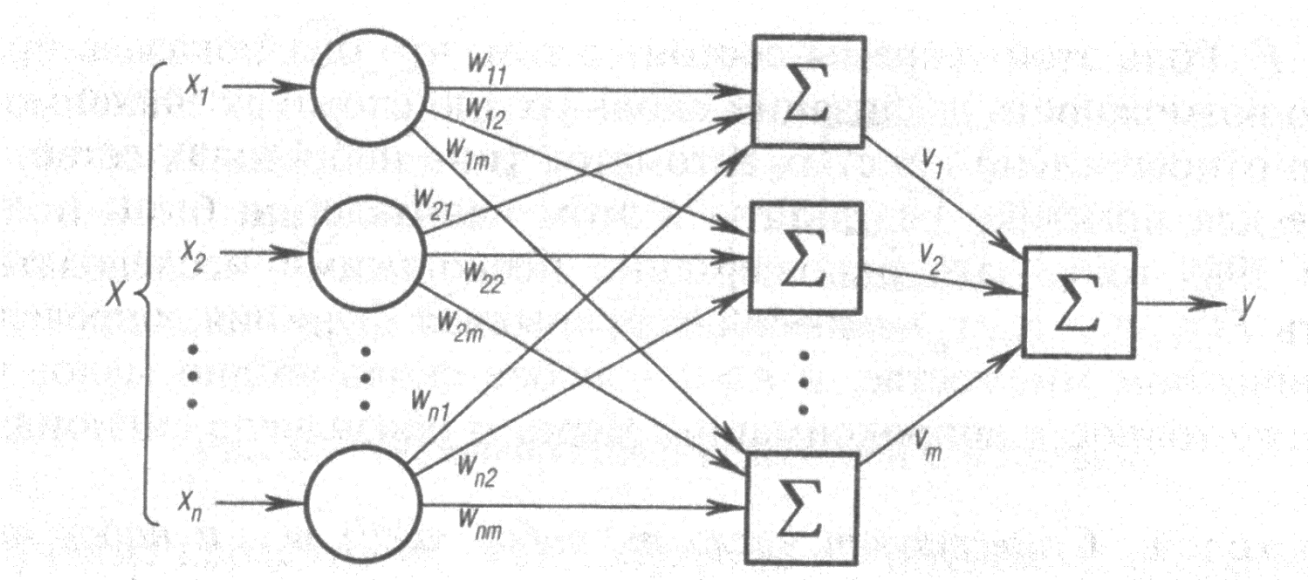


Рис. 3. Пример двухслойной нейронной сети с ***n*** входами и одним выходом

Каждый *i*-й нейрон первого слоя (*i*=1, 2, ..., *m)* имеет *n* входов, с весами w_{1i},., w_{2i}, ..., w_{ni}.

Подавая на входы любые числа x_1 x_2, ..., x_n, получим на выходе значение некоторой функции $y=F(x_1,x_2,..., x_n)$, которое является ответом (реакцией) сети.

Точный вид этой функции:

$$F(x_1, x_2, ..., x_n) = \sum_{i=1}^{m} v_i \sigma(\sum_{j=0}^{n} x_j w_{ji}), \quad \text{где} \quad \sigma(s) = \frac{1}{1+e^{-as}}.$$

Теорема Колмогорова. Любая непрерывная функция F, определенная на n-мерном единичном кубе, может быть представлена в виде суммы $2n+1$ суперпозиций непрерывных и монотонных отображений единичных отрезков:

$$F(x_1, x_2, ..., x_n) = \sum_{i=1}^{2n+1} g_i(\sum_{j=1}^{n} h_{ij}(x_j)),$$

$$x=(x_1,x_2,...,x_n), \quad 0 \le x_i \le 1,$$

где g_i и h_{ij} — непрерывные функции, h_{ij}. не зависят от функции F.

Пусть $\boldsymbol{F(x_1,x_2,\ldots,\ x_n)}$ - любая непрерывная функция, определенная на ограниченном множестве, и $\boldsymbol{\varepsilon>0}$ — любое сколь угодно малое число, означающее точность аппроксимации. Пусть σ - сигмоидальная функция.

Теорема. Существуют число m, набор чисел w_{ij}, и набор чисел v_i таких, что функция

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{m} v_i \sigma\left(\sum_{j=0}^{n} x_j w_{ji}\right),$$

приближает данную функцию $F(x_1,x_2,\ldots,\ x_n)$ с погрешностью не более ε на всей области определения.

Сеть обратного распространения

В 1986 году Румельхарт (D.E.Rumelhart) разработал алгоритм обратного распространения ошибок - **back propagation**.

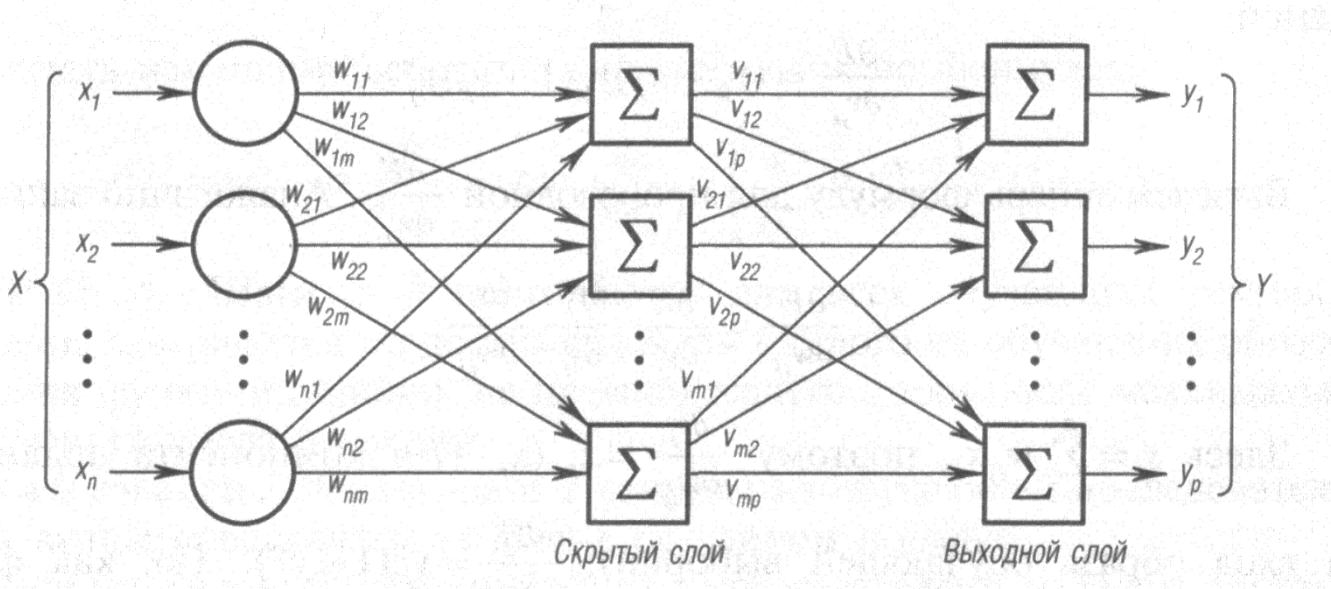


Рис. 4. Нейронная сеть обратного распространения

Матрица весовых коэффициентов от входов к скрытому слою - ***W***, а матрица весов, соединяющих скрытый и выходной слой — ***V***. Входы будем нумеровать индексом ***i***, элементы скрытого слоя — индексом ***j***, а выходы — индексом ***k***. Число входов сети - ***n***, число нейронов в скрытом слое— ***m***, число нейронов в выходном слое— ***p***.

Пусть сеть обучается на выборке (X^t, D^t), $t = 1, 2, ..., T$.

Ставится задача минимизации ***целевой функции ошибки*** по методу ***наименьших квадратов***:

$$E(W,V) = \frac{1}{2}\sum_{k=1}^{p}(y_k - d_k)^2 ,$$

где y_k - полученное реальное значение k-го выхода сети при подаче на нее одного из входных образов обучающей выборки; d_k — требуемое (целевое) значение k-го выхода для этого образа. На каждой итерации изменение веса производится по формулам:

$$w_{ij}^{N+1} = w_{ij}^{N} - \alpha \frac{\partial E}{\partial w_{ij}} , \quad (1)$$

$$v_{jk}^{N+1} = v_{jk}^{N} - \alpha \frac{\partial E}{\partial v_{jk}} , \quad (2)$$

где α — параметр, определяющий скорость обучения.

В качестве активационной функции обычно используется сигмоидальная функция $f(s) = \frac{1}{1+e^{-as}}$, где ***s*** — взвешенная сумма входов нейрона.

Производная: $f'(s) = \frac{e^{-as}}{(1+e^{-as})^2} = f(s)(1-f(s))$.

Можно показать, что $\frac{\partial E}{\partial w_{ij}} = (\sum_{k=1}^{p} \delta_k v_{jk}) y_j^c (1-y_j^c) x_i$ и $\frac{\partial E}{\partial v_{jk}} = \delta_k y_j^c$,

где $\delta_k = (y_k - d_k) y_k (1-y_k)$, а y_j^c - значение выхода ***j*** -го нейрона скрытого слоя.

Алгоритм обратного распространения:

Шаг 1. Инициализация сети.

Весовым коэффициентам присваиваются малые случайные значения (-0,3; 0,3); задаются ε — параметр точности обучения, α — параметр скорости обучения (α≈0,1), N_{max} — максимально допустимое число итераций.

Шаг 2. Вычисление текущего выходного сигнала.

На вход сети подается один из образов обучающей выборки, и определяются значения выходов всех нейронов сети.

Шаг 3. Настройка синаптических весов.

Рассчитать изменение весов для выходного слоя нейронной сети (формула (2)).

Рассчитать изменение весов для скрытого слоя (формула (1)).

Шаг 4. Шаги 2—3 повторяются для всех обучающих векторов. Обучение завершается когда для каждого из обучающих образов значения функции ошибки, не превосходит ε, или после максимально допустимого числа итераций.

Замечание 1. На шаге 2 векторы из обучающей последовательности лучше предъявлять на вход в случайном порядке.

Замечание 2. Количество входов и выходов сети, как правило, диктуется условиями задачи, а размер скрытого слоя находят экспериментально. Обычно число нейронов в скрытом слое составляет 30—50% от числа входов.

Замечание 3. Выходы каждого нейрона сети лежат в диапазоне (0, 1) —области значений логистической функции. Если необходимо получить от сети бинарный выход, то, как правило, вместо 0 используют 0,1, а вместо 1 — 0,9.

Сеть встречного распространения

В качестве меры близости двух векторов - евклидово расстояние $d(x,y)=\Sigma(x_i-y_i)^2$.

Сеть Кохонена (T.Kohonen) состоит из одного слоя нейронов. Число входов каждого нейрона ***n*** равно размерности вектора параметров объекта. Количество нейронов ***m*** совпадает с требуемым числом классов, на которые нужно разбить объекты (меняя число нейронов, можно динамически менять число классов).

- Для ***j***-го нейрона (***j=1, 2, .-.., m)*** определяется расстояние от него до входного вектора ***X***:

$$d_j = \sum_{i=1}^{n} (x_i - w_{ij})^2 .$$

- Выбирается нейрон с номером ***k, 1≤k≤m***, для которого это расстояние минимально.

- На текущем шаге обучения ***N*** будут модифицироваться только веса нейронов из окрестности нейрона ***k***:

$$w_{ij}^{N+1} = w_{ij}^{N} + \alpha_N (x_i - w_{ij}^{N}) .$$

Темп обучения α_N с течением времени уменьшается ***(α_0 =0,9, a_{N+1} = α_N -0,001)***.

Алгоритм обучения сети Кохонена

Шаг 1. Инициализация сети. Весовым коэффициентам сети w_{ij}, $i=1,2\ldots,n$, $j=1,2,\ldots,m$ присваиваются малые случайные значения. Задаются значения α_0 —начальный темп обучения и D_o — максимальное расстояние между весовыми векторами (столбцами матрицы W).

Шаг 2. Предъявление сети нового входного сигнала X.

Шаг 3. Вычисление расстояния от входа X до всех нейронов сети:

$$d_j = \sum_{i=1}^{n} (x_i - w_{ij}^N)^2,\ j = 1,2,\ldots,m.$$

Шаг 4. Выбор нейрона k, $1 \le k \le m$ с наименьшим расстоянием d_k.

Шаг 5. Настройка весов нейрона k и всех нейронов, находящихся от него на расстоянии, не превосходящем D_N: $w_{ij}^{N+1} = w_{ij}^N + \alpha_N (x_i - w_{ij}^N)$.

Шаг 6. Уменьшение значений α_N, D_N .

Шаг 7. Шаги 2—6 повторяются до тех пор, пока веса не перестанут меняться (или пока суммарное изменение всех весов станет очень мало).

Замечание. Если предварительно провести единичную нормировку входных векторов, т. е. подавать на вход сети образы X', компоненты которого связаны с компонентами вектора X как

$$x_i' = \frac{x_i}{\sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}},$$

а также если после каждой итерации процесса обучения осуществлять нормировку весов каждого нейрона (столбцов матрицы W), то ***в качестве меры близости входных векторов*** и весовых векторов нейронов сети можно рассматривать ***скалярное произведение*** между ними.

Каждый нейрон реализует тождественную активационную функцию:

$$f(s) = s, \quad s = \sum_{i=1}^{n} w_{ij} x_i.$$

2. Данные – информация – знания. Свойства знаний, сходство/различие понятий

Данные

В упрощённом представлении данными можно считать полученные факты (сведения, сообщения, сигналы), характеризующие объект. Данные не тождественны информации. Станут ли данные информацией, зависит от того, известен ли метод преобразования данных в известные понятия. Информация – это не статичный объект. Она динамически меняется и существует только в момент взаимодействия данных и методов их преобразования. Данные объективны, а методы субъективны, в их основе лежат алгоритмы, придуманные людьми (субъектами).

Информация

Информация – это обработанные данные независимо от формы их представления. Информация, в отличие от данных, имеет смысл. Однако трудно найти понятие, более общее для всех наук и более загадочное, чем информация. Существует несколько подходов к измерению информации. Рассмотрим наиболее популярные из них.

Мера Хартли. Пусть имеется система ***S***. Она может находиться в ***N*** равновероятных состояниях. Если каждое состояние системы закодировать, например, двоичными кодами определённой длины d, то эту длину необходимо выбрать так, чтобы число всех различных комбинаций было бы не меньше, чем ***N***. Наименьшее число, при котором это возможно или мера разнообразия множества состояний системы задаётся формулой Р. Хартли:

$$I = k \log_a N$$

где ***k*** - коэффициент пропорциональности (масштабирования, в зависимости от выбранной единицы измерения), а ***a*** - основание рассматриваемой системы.

Мера К. Шеннона. Формула Шеннона дает оценку информации независимо от ее смысла:

$$I = -\sum_{i=1}^{N} p_i \log_2 p_i,$$

где ***N*** - число состояний системы; p_i - вероятность (или относительная частота) перехода системы в ***i***-е состояние, причем

$$\sum_{i=1}^{N} p_i = 1.$$

Если все состояния равновероятны (т.е. $p_i=1/N$), то $I=log_2N$, т.е. мера Шеннона совпадает с мерой Хартли.

Увеличение (уменьшение) меры Шеннона свидетельствует об уменьшении (увеличении) энтропии (организованности, порядка) системы.

Энтропия может являться мерой дезорганизации систем от полного хаоса ($H=Hmax$) и полной информационной неопределённости ($I=Imin$) до полного порядка ($H=Hmin$) и полной информационной определённости ($I=Imax$) в системе.

Знания

- Знания – это обработанная информация, обеспечивающая увеличение вероятности достижения цели, «ноу-хау», технология. Знание - это информация, но не всякая информация - знание. Между информацией и знаниями имеется разрыв. Человек должен творчески перерабатывать информацию, чтобы получить новые знания. Приведем несколько любопытных определений понятия «знание»:

- «В моем текущем Знании нет ничего абсолютного, и оно ничтожно мало. Я знаю, что ничего не знаю. Чем больше Знание, тем больше и Незнание» (Сократ).

- «Знание есть особая форма Незнания. Незнание – неизменно, бесконечно, всеобще, изначально. Знание – эклектично, субъективно, ограниченно, локально и временно. Только для философии и искусственного интеллекта предметом изучения являются сами Знания» (А.С.Нариньяни).

- «Знание – обоснованное истинное убеждение» (В.Н. Вагин).

При изучении некоторого объекта понятия появляются в следующем порядке: ***данные-информация-знания***.

Прагматический подход: - знания – это формализованная информация, на которую ссылаются и/или которую используют в процессе логического вывода.

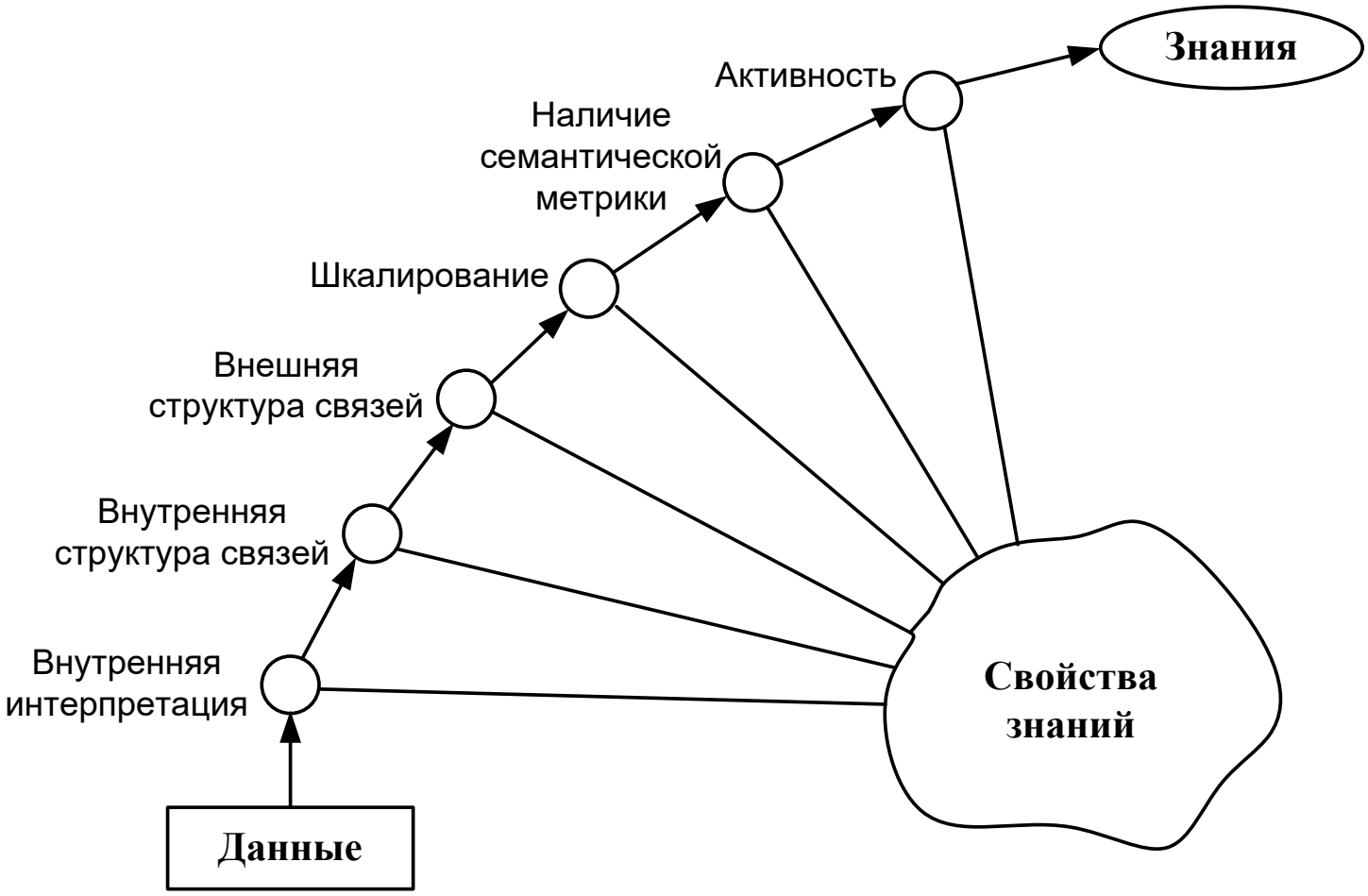
Однако такое определение ограничено: оно фиксирует сознание на уже существующих методах представления знаний и, соответственно, механизмах вывода, не давая возможности представить себе другие ("новые").

Возможен и другой подход: попытаться на основе определения понятия *"данные"*, выявить их свойства и особенности, сформировать дополнительные требования к ним и уже затем перейти к понятию "знания".

Данными называют формализованную информацию, пригодную для последующей обработки, хранения и передачи средствами автоматизации профессиональной деятельности.

Какие же свойства "превращают" данные в знания?

Свойства знаний



- *Внутренняя интерпретация* (интерпретируемость). Это свойство предполагает, что в ЭВМ хранятся не только "собственно (сами) данные", но и "данные о данных", что позволяет содержательно их интерпретировать.
- Наличие *внутренней структуры связей.* Предполагается, что в качестве информационных единиц используются не отдельные данные, а их упорядоченные определенными отношениями (родовидовыми, причинно-следственными и др.) структуры (эти отношения называют классифицирующими). Пример: факультет – курс – учебная группа – студент.

3. Наличие *внешней структуры связей.*

Внутренняя структура связей позволяет описывать отдельный объект (понятие). Однако объекты (понятия) способны находиться и в других отношениях (вступать в ситуативную связь). Пример: объекты "курс Государственного университета управления им. С. Орджоникидзе" и "урожай овощей в совхозе "Зареченский" могут находиться в ситуативной связи "принимает участие в уборке".

4. Возможность *шкалирования.*

Эта возможность предполагает введение соотношений между различными информационными единицами (т. е. их измерение в какой-либо шкале – порядковой, классификационной, метрической и т. п.) и упорядочение информационных единиц путем измерения интенсивности отношений и свойств. Пример: "97/ЭИ. 6-01 учебная группа занимает первое место на курсе по успеваемости".

5. Наличие *семантической метрики.* Шкалирование позволяет соотнести информационные единицы, но прежде всего для понятий, имеющих "количественное" толкование (характеристики). На практике довольно часто встречаются понятия, к которым не применимы количественные шкалы, но существует потребность в установлении их близости (например, понятия "искусственный интеллект" и "искусственный разум").

Семантики классифицируются следующим образом:

- *значение,* т. е. объективное содержание;
- *контекстуальный смысл,* определяемый связями данного понятия с другими, соседствующими в данной ситуации;
- *личностный смысл,* т. е. объективное значение, отраженное через систему взглядов эксперта;
- *прагматический смысл,* определяемый текущим знанием о конкретной ситуации (например, фраза "информация получена" может иметь как негативную, так и позитивную оценку – в зависимости от того, нужно это было или нет).

6. Наличие *активности.*

Данное свойство *принципиально отличает* понятие "знание" от понятия "данные". Например, знания человека, как правило, активны, поскольку ему свойственна познавательная активность (обнаружение противоречий в знаниях становится побудительной причиной их преодоления и появления новых знаний, стимулом активности является неполнота знаний, выражается в необходимости их пополнения).

Кроме перечисленных, знаниям присущи такие свойства, как *омонимия* (слово "коса" может иметь три смысла, связанных с определениями: девичья; песчаная; острая) и *синонимия* (знания "преподаватель читает лекцию" и "студенты слушают лекцию" во многих случаях являются синонимами) и др. Классифицировать знания можно по самым различным основаниям.

По способу существования различают *факты* (хорошо известные обстоятельства) и *эвристики* (знания из опыта экспертов).

По *способу использования* в экспертных системах – *фактические знания* (факты) – знания типа "А – это А"; *правила* – знания для принятия решений ("Если... – то..."); *метазнания* (знания о знаниях – указывают системе способы использования знаний и определяют их свойства). Классическими примерами метазнаний являются народные пословицы и поговорки, каждая из которых характеризует знания (рекомендации по деятельности) в широком классе конкретных ситуаций (например, пословица "Семь раз отмерь, один – отрежь" применима не только в среде хирургов или портных).

По *формам представления* знания подразделяются на *декларативные* (факты в виде наборов структурированных данных) и *процедуралъные* (алгоритмы в виде процедур обработки фактов).

По *способу приобретения* знания бывают научные (полученные в ходе систематического обучения и/или изучения) и *житейские, бытовые* (полученные в "ходе жизни").

Дадим еще ряд определений, часто встречающихся в литературе.

Интенсиональные знания – знания, характеризующие или относящиеся к некоторому классу объектов.

Экстенсиональные знания – знания, относящиеся к конкретному объекту из какого-либо класса (факты, сведения, утверждения и т. д.)

Заметим: отношения интенсиональных и экстенсиональных знаний – это родовидовые отношения. Например, понятие "технологическая операция" – это интенсионал, а понятие "пайка" – это экстенсионал, так как пайка – одна из технологических операций. Очевидно, что эти *понятия относительны.* Так понятие "пайка", в свою очередь, можно считать

интенсионалом по отношению к понятиям "пайка серебром" и "пайка оловом". Как правило, такого рода знания относятся к декларативным.

Физические знания – знания о реальном мире.

Ментальные знания – знания об отношениях объектов.

Мир задачи – совокупность знаний, используемых в задаче.

Мир пользователя – совокупность знаний пользователя.

Мир программы – совокупность знаний, используемых в программе.

Морфологические и синтаксические знания – знания о правилах построения структуры описываемого явления или объекта (например, правила написания букв, слов, предложений и др.).

Семантические знания – знания о смысле и значении описываемых явлений и объектов.

Прагматические знания – знания о практическом смысле описываемых объектов и явлений в конкретной ситуации.

Предметные знания – знания о предметной области, объектах из этой области, их отношениях, действиях над ними и др.

3. Модели представления знаний. Задача вывода в базе знаний

Для того *чтобы манипулировать всевозможными знаниями* из реального мира с помощью компьютера, необходимо осуществить их *моделирование.*

При проектировании модели представления знаний следует учесть *два требования:*

- *однородность представления;*
- простота *понимания.*

Выполнение этих требований позволяет упростить механизм логического вывода и процессы приобретения знаний и управления ими, однако, как правило, создателям интеллектуальной системы приходится идти на некоторый компромисс в стремлении обеспечить одинаковое понимание знаний и экспертами, и инженерами знаний, и пользователями.

Классификация методов моделирования знаний с точки зрения подхода к их представлению в ЭВМ показана на рисунке.

Дадим общую характеристику основных методов представления знаний с помощью моделей, основанных на эвристическом подходе.

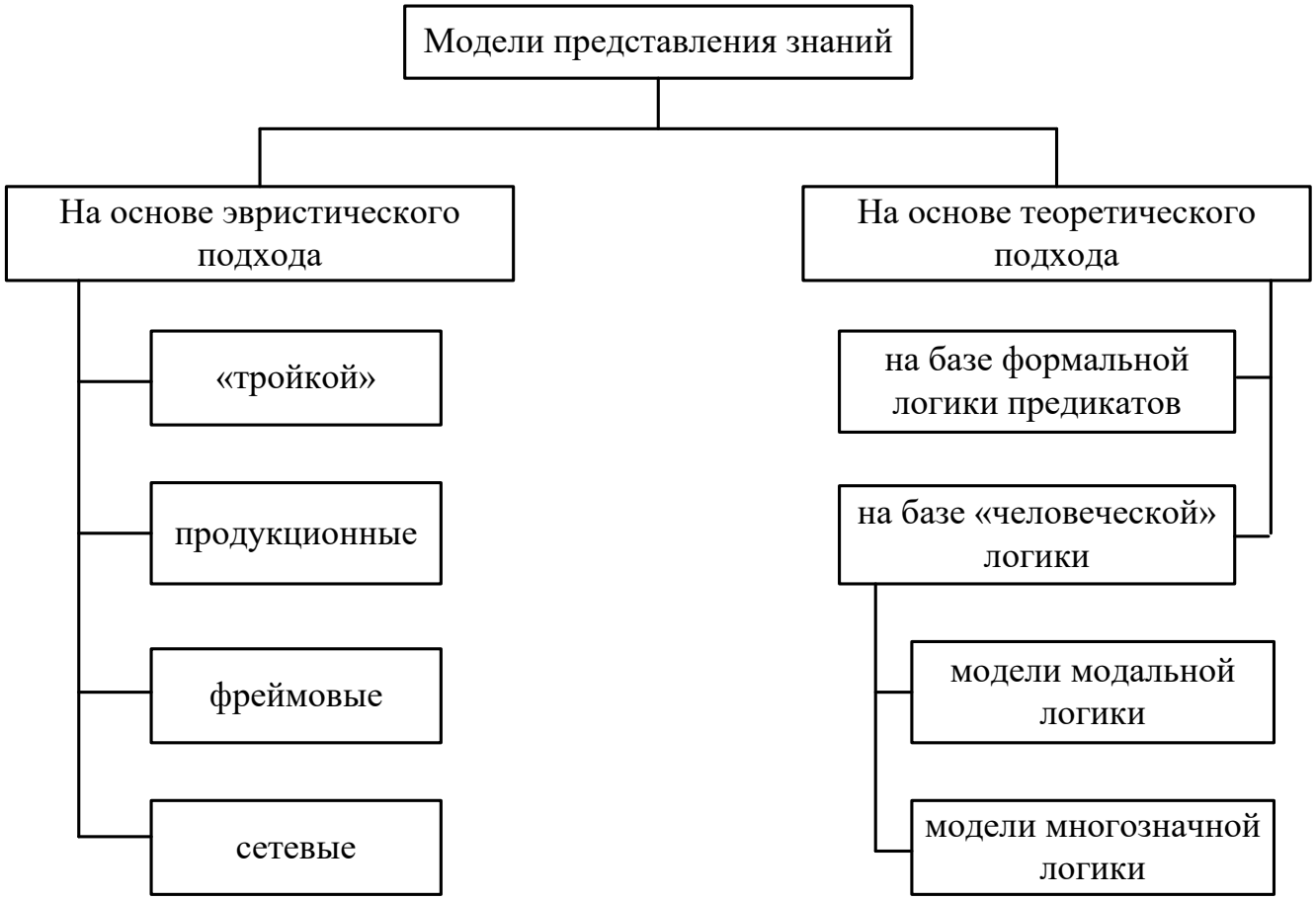


Рис. 3.1. Классификация моделей представления знаний

1. Представление знаний тройкой *"объект – атрибут – значение"* – один из первых методов моделирования знаний. Как правило, используется для представления *фактических знаний в простейших системах.*

Примеры:

Объект	Атрибут	Значение
Студент	Успеваемость	Отличник
Дом	Цвет	Белый
Пациент	Температура	Нормальная

Очевидно, что для моделирования знаний даже об одном объекте (например, о "студенте" или "доме") из предметной области *необходимо хранить значительное число "троек".*

2. Продукционная модель (модель правил; модель продукций – от англ, *production* – изготовление, выработка). В настоящее время наиболее проработанная и распространенная модель представления знаний, в частности – в экспертных системах.

Модель предусматривает разработку системы продукционных правил (правил продукций), имеющих вид:

ЕСЛИ A_1 И A_2 И ... A_n, ТО B1 ИЛИ B_2 ИЛИ...ИЛИ B_m, где Ai и Bj. – некоторые высказывания, к которым применены логические операции И и ИЛИ. Если высказывания в левой части правила (ее часто называют *антецедент* – условие, причина) истинно, истинно и высказывание в правой части *(консеквент* – следствие).

Полнота базы знаний (базы правил) определяет возможности системы по удовлетворению потребностей пользователей. Логический вывод в продукционных системах основан на построении прямой и обратной цепочек заключений, образуемых в результате последовательного просмотра левых и правых частей соответствующих правил, вплоть до получения окончательного заключения.

Пусть в некоторой области памяти хранятся следующие правила (суждения):

правило 1 – ЕСЛИ в стране происходит падение курса национальной валюты; ТО материальное положение населения ухудшается;

правило 2 – ЕСЛИ объемы производства в стране падают; ТО курс национальной валюты снижается;

правило 3 – ЕСЛИ материальное положение населения ухудшается; ТО уровень смертности в стране возрастает.

Если на вход системы поступит новый факт "В стране высокий уровень падения объемов производства", то из правил можно построить цепочку рассуждений и сформулировать два заключения:

факт 1 – правило 2 – правило 1 – заключение 1 – правило 3 – заключение 2,

где *заключение 1* (промежуточный вывод) – "Материальное положение населения ухудшается"; *заключение 2* (окончательный вывод) – "В стране возрастает уровень смертности".

В современных экспертных системах в базе знаний могут храниться тысячи правил, а коммерческая стоимость одного невыводимого (нового, дополнительного) правила очень высока.

Преимущества продукционных правил: простота создания и понимания отдельных правил; простота пополнения, модификации и аннулирования; простота механизма логического вывода, наличие программных средств (языки Prolog, LISP, CLIPS, ЭС).

Недостатки ПС: неясность взаимных отношений правил; сложность оценки целостного образа знаний; невысокая эффективность обработки; отсутствие гибкости в логическом выводе; сложность проверки правил на непротиворечивость, особенно если правил больше 1000, неоднозначность выбора правил.

Таким образом, если объектом является небольшая задача, выявляются только сильные стороны системы продукций. В случаях увеличения объема знаний, необходимости решения сложных задач, выполнения гибких выводов или повышения скорости вывода требуется структурирование базы данных.

3. *Фреймовая модель.* Сравнительно новая модель представления знаний. Само понятие "фрейм" (англ, *frame* – рама, рамка, скелет, сгусток, сруб и т. д.) было введено в 1975 г. Марком Минским (M. Minsky, США).

Фрейм – это минимальная структура информации, необходимая для представления знаний о стереотипных классах объектов, явлений, ситуаций, процессов и др. С помощью фреймов можно моделировать знания о самых разнообразных объектах интересующей исследователя предметной области – важно лишь, чтобы эти объекты составляли класс концептуальных (повторяющихся: стереотипных) объектов, процессов и т. п.

Примерами стереотипных жизненных ситуаций могут служить собрание, совещание; сдача экзамена или зачета; защита курсовой работы и др. Примеры стереотипных бытовых ситуаций: отъезд в отпуск; встреча гостей; выбор телевизора; ремонт и др. Примеры стереотипных понятий: алгоритм; действие; методика и др. На рисунке представлен фрейм технологической операции "соединять":

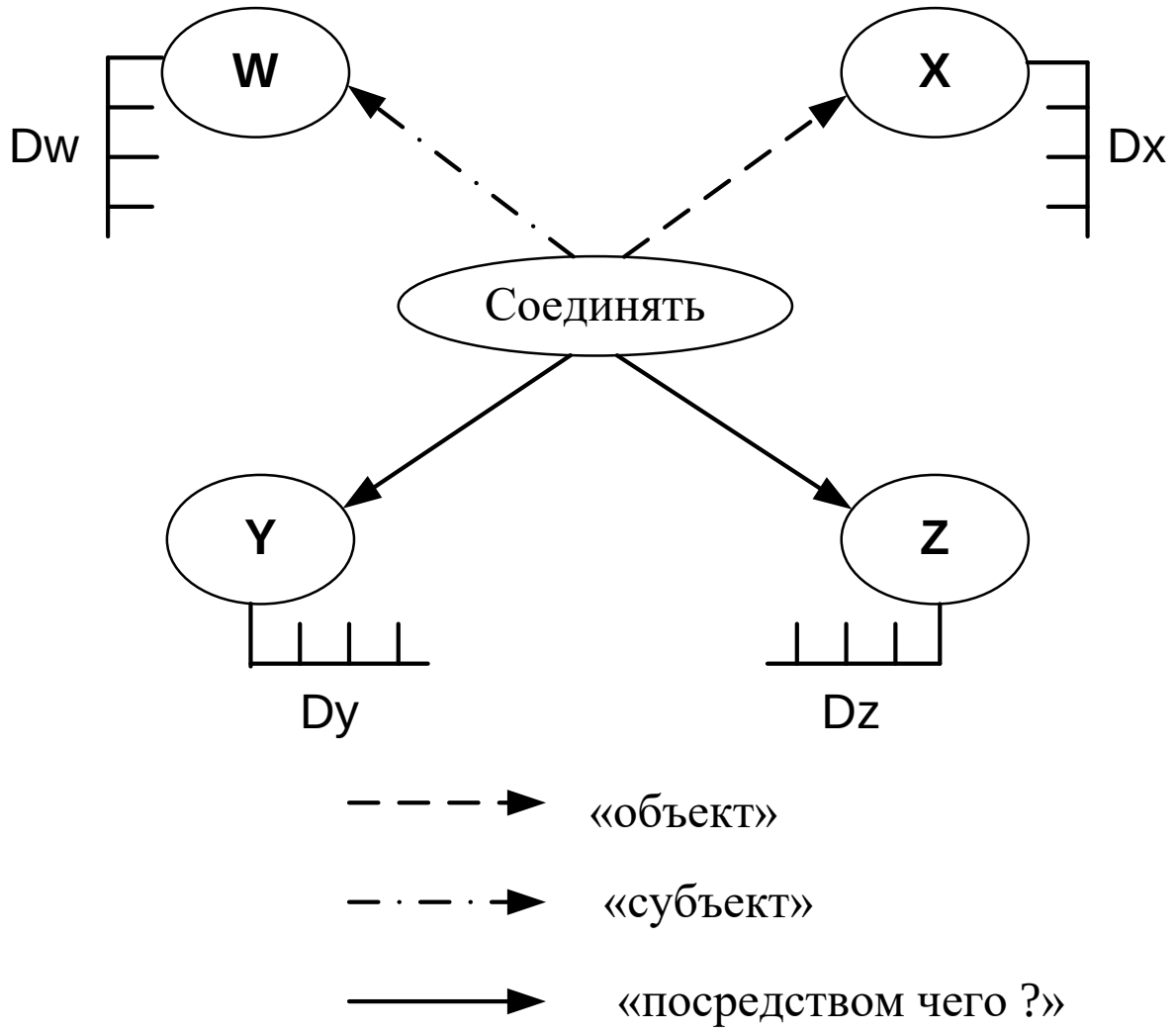


Рис. Фрейм ситуации «соединить»

Данный фрейм описывает ситуацию "Субъект *X* соединяет объект У с объектом Z способом W". На рисунке обозначены: вершины *X*, Y, Z, W – *слоты* (англ, *slot* – прорез; щель; пустота – составляющие фрейма); дуги – отношения; *Dx*, D_y, *D*z, D_w – так называемые *шанции* – области возможных значений соответствующих слотов.

Наполняя слоты конкретным содержанием, можно получить фрейм конкретной ситуации, например; "Радиомонтажник соединяет микросхему с конденсатором способом пайки". Заполнение слотов шанциями называют *активизацией* фрейма,

С помощью фреймов можно моделировать как процедурные, так и декларативные знания.

На рис. 3.2 представлен пример представления процедурных знаний.

На рис. 3.3 приведен пример фрейма "технологическая операция", иллюстрирующий представление декларативных знаний для решения задачи проектирования технологического процесса.

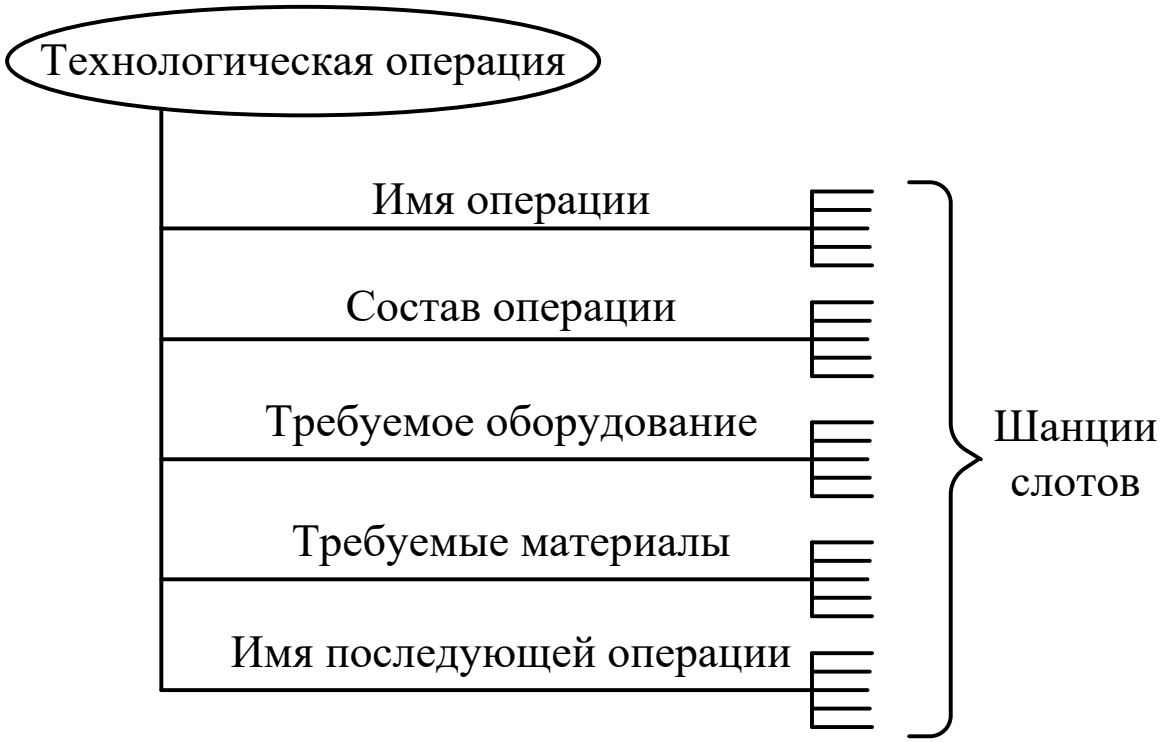


Рис. 3.3. Фрейм понятия "технологическая операция"

По содержательному смыслу фрейма выделяют:

- фреймы-понятия;
- фреймы-меню;
- фреймы с иерархически вложенной структурой.

Фрейм-понятие – это фрейм типа *И.* Например, фрейм "операция" содержит объединенные связкой *И* имена слотов "что делать", "что это дает", "как делать", "кто делает", "где делать" и т, д., а фрейм "предмет" – слоты с именами "назначение", "форма", "вес", "цвет" и т. д.

Фрейм-меню - это фрейм типа *ИЛИ.* Он служит для организации процедурных знаний с помощью оператора "выбрать". Например, фрейм "что делать" может состоять из объединенных связкой *ИЛИ* слотов "решить уравнение", "подставить данные", "уточнить задачу" и т, д., причем каждый из этих слотов может иметь несколько значений,

Фрейм с иерархически вложенной структурой предполагает, что в нем в качестве значений слотов можно использовать имена других фреймов, слотов и т, д., т, е, использовать иерархическую структуру, в которой комбинируются другие виды фреймов (в итоге получают так называемые фреймы-сценарии).

Значения слотов могут содержать ссылки на так называемые *присоединенные процедуры.*

Различают два вида присоединенных процедур:

- Процедуры – демомы.
- Процедуры – слуги.

Процедуры-демоны присоединяются к слоту и активизируются при изменении информации в этом слоте (выполняют вспомогательные операции – например, автоматически корректируют информацию во всех других структурах, где используется значение данного слота) – см. рис. 3.4.

Процедуры-слуги активизируются при выполнении некоторых условий относительно содержимого слотов (часто по запросу). Данные процедуры определяются пользователем при создании фрейма. Например, во фрейме "Учебная аудитория" можно предусмотреть слоты "длина" и "ширина", а по их значениям вычислять значение слота "площадь".

Процедуры-демоны:

1	Процеду добавлено"	Выполняется, когда новая
2	Процеду удалено"	Выполняется, когда
3	Процеду нужно"	Выполняется, когда

Рис. 3.4. Типы присоединенных процедур

Преимущества фреймов: наглядность, соответствие модели памяти человека, нет ограничений по размерам, поддерживается программно. Достоинство фрейма – представления во многом основываются на включении в него предположений и ожиданий.

Фреймовые модели обеспечивают требования структурированности и связанности. Это достигается за счет свойств наследования и вложенности, которыми обладают фреймы, т.е. в качестве слотов может выступать система имен слотов более низкого уровня, а также слоты могут быть использованы как вызовы каких-либо процедур для выполнения.

***Недостаток*:** трудность обмена большими объемами данных.

4. *Модель семантической сети* (модель Куилиана).

Одной из структурных моделей долговременной памяти является предложенная Куиллианом модель понимания смысла слов, получившая название TLC- модели (Teachable Language Comprehender: доступный механизм понимания языка).

Идея семантической сети (СС) – любое знание можно представить совокупностью понятий и отношений между ними. СС – это ориентированный граф, вершины которого понятия, а ребра – отношения между понятиями:

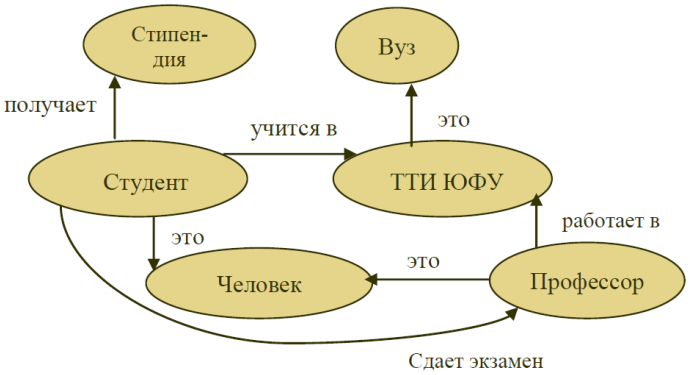


Рис.3.5.

Таким образом, с*емантическая сеть* – это *направленный граф с поименованными вершинами и дугами,* причем *узлы* обозначают конкретные *объекты,* а *дуги – отношения между ними*.

В семантических сетях используют три типа вершин:

1. вершины-понятия (обычно это существительные);
2. вершины-события (обычно это глаголы);
3. вершины-свойства (прилагательные, наречия, определения).

Дуги сети (семантические отношения) делят на четыре класса:

1. *лингвистические* (падежные, глагольные, атрибутивные);
2. *логические* (И, ИЛИ, НЕ);
3. *теоретико-множественные* (множество – подмножество, отношения целого и части, родовидовые отношения);
4. *квантифицированные* (определяемые кванторами общности $\forall$ и существования $\exists$).

Пример. На рис 3.6 представлена семантическая сеть для предложения (ситуации) "Студент Табуреткин добросовестно изучает новый план счетов на 2022 год перед сдачей экзамена по дисциплине "Бухгалтерский учет".

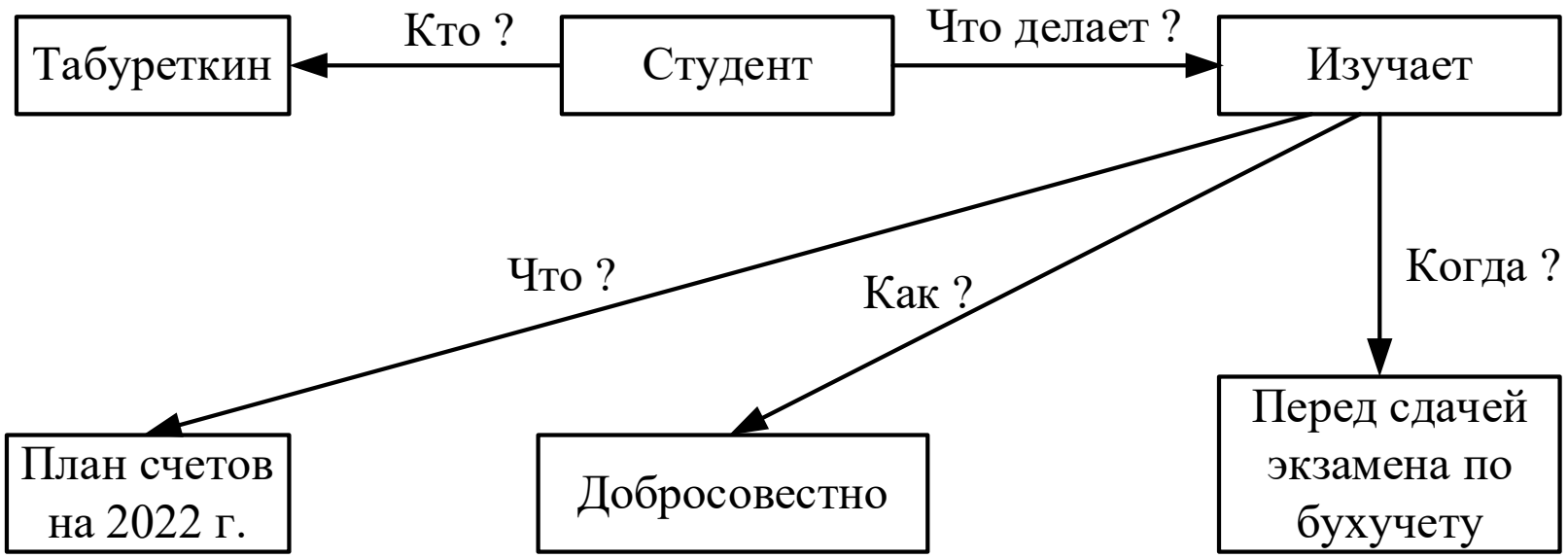


Рис.3.6. Семантическая сеть для предложения (ситуации)

Рисунок 3.7 содержит фрагмент семантической сети для понятия "автомобиль" (обозначения: IS-A – есть, является; HAS-PART – имеет часть).

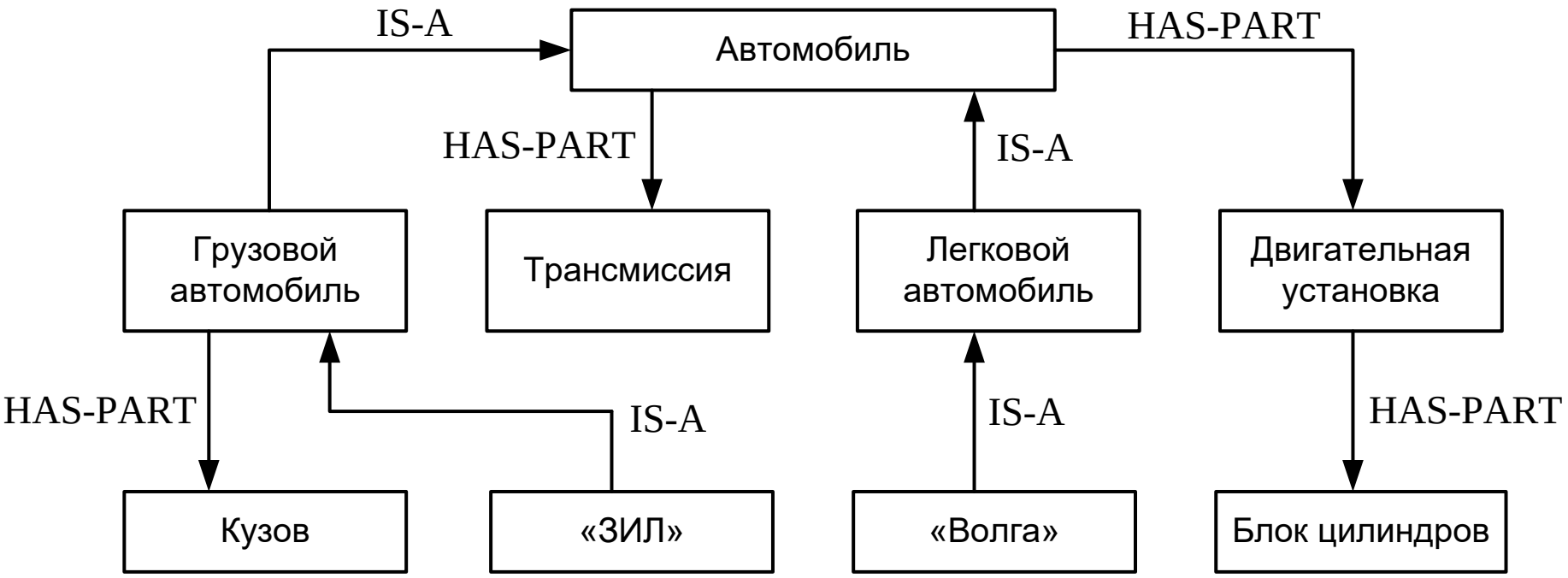


Рис. 3.7. Фрагмент семантической сети понятия "автомобиль"

Достоинство методов моделирования знаний с помощью семантических сетей и фреймов – универсальность, удобство представления как декларативных, так и процедуральных знаний. Имеют место и два ***недостатка***:

1. *громоздкость, сложность* построения и *изменения;*
2. *потребность* в разнообразных *процедурах обработки,* связанная с разнообразием типов дуг и вершин.

Достоинства семантической сети: описание объектов и событий производится на уровне очень близком к естественному языку; обеспечивается возможность соединения различных фрагментов сети; отношения между понятиями и событиями образуют небольшое, хорошо организованное множество; для каждой операции над данными или знаниями можно выделить некоторый участок сети, который охватывает необходимые в данном запросе характеристики; обеспечивается наглядность системы знаний, представленной графически; близость структуры сети, представляющей знания, семантической структуре фраз на естественном языке; соответствие сети современным представлениям об организации долговременной памяти человека.

Недостатки семантической сети: сетевая модель не дает ясного представления о структуре предметной области, поэтому формирование и модификация такой модели затруднительны; сетевые модели представляют собой пассивные структуры, для обработки которых необходим специальный аппарат формального вывода и планирования.

Семантические сети нашли применение в основном в системах обработки естественного языка, частично в вопросно-ответных системах, а также в системах искусственного видения. В последних семантические сети используются для хранения знаний о структуре, форме и свойствах физических объектов. В области обработки естественного языка с помощью семантических сетей представляют семантические знания, знания о мире, эпизодические знания (т.е. знания о пространственно-временных событиях и состояниях).

В рамках реализации *теоретического подхода* применяют *логические модели,* прежде всего использующие представления знаний в системе *логики предикатов.*

Преимущества такого подхода очевидны: единственность теоретического обоснования и возможность реализации системы путем введения формально точных определений и правил получения выводов. Однако в полной мере претворить в жизнь данный подход даже для "простых" задач оказалось весьма сложно. Поэтому появились *попытки перейти от формальной логики к так называемой человеческой логике* (модальной логике, многозначной логике и др.), модели которой в большей или меньшей степени учитывают "человеческий фактор", т. е. являются в определенном смысле компромиссными "в плане использования и теоретического, и эвристического подходов.

Очень коротко остановимся на предикатной модели представления знаний. Первые попытки использовать такую модель относятся к 50-м гг. прошлого века. Дадим несколько определений.

Пусть имеется некоторое *множество объектов,* называемое *предметной областью.* Выражение $P(x_1, x_2,...,x_n)$, где x_i ($i = 1,..,n$) – так называемая предметная переменная, а P принимает значения 0 или 1, называется *логической функцией* или *предикатом.*

Предикат $P(x_1,x_2,...,x_n)$ задает *отношение* между элементами x_1 $x_2,...,x_n$ и обозначает высказывание, что "x_1, $x_2,...,x_n$ находятся между собой в отношении P".

Из подобного рода элементарных высказываний с помощью *логических связок* образуют *более сложные высказывания,* которые могут принимать те же значения – "истина" и "ложь". В качестве связок используются *конъюнкция, дизъюнкция, импликация, отрицание, эквивалентность.*

Предикат от *n* переменных называют n-местным.

Одноместные (унарные) предикаты отражают свойства *определенного объекта* или класса *объектов.* Многоместные предикаты позволяют записывать *отношения,* которые существуют *между группой элементов.*

Если *a* – тоже предикат, то P(*a*) – предикат 2-го порядка, и т. д. до n~го порядка.

Приведем примеры различных предикатов.

1. Унарный предикат (высказывание) "Река впадает в Каспийское море" имеет значение 1, если "Река" = "Волга", и значение 0, если "Река" = "Днепр".

2. Двухместный предикат *"x_1* не меньше x_2" может иметь значение 1 или 0 в зависимости от значений *x_1* и x_2. Если значение предиката *тождественно равно* 1 при любых значениях предметных переменных, он называется *тавтологией.*

В аппарат исчисления предикатов входят также *символы функций* (обычно обозначаемые латинскими буквами /, *g*, *h* и т. д.), задаваемых на множестве предметных переменных, и *кванторы общности* ∀ и *существования* ∃.

3. Представление с помощью предиката знаний, заключенных в теореме Пифагора: P{g[f(x), f(y)], f(z)}, где предикат P – "быть равным", функция $g(x, y) = x + y;$ функция $f(x) = x^2$.

Иногда используется такая форма записи:

РАВНЫ [СУММА (КВАДРАТ *(x),* КВАДРАТ *(y)),* КВАДРАТ (z)].

Предикат P равен 1, если *x, y, z* – соответственно длины катетов и гипотенузы прямоугольного треугольника.

4. Достоверный вывод, метод резолюций. Правдоподобные методы вывода

Различают *правдоподобный* и *достоверный* механизмы вывода.

К правдоподобному выводу относят *вывод по аналогии* (от частного к частному), *индуктивный* (от частного к общему), *нечёткий* (основанный на логике Заде), *нейросетевой* и некоторые другие методы вывода.

Единственным достоверным методом вывода является *дедуктивный логический вывод* (рассуждения от общего к частному на основе аксиом).

Если поиск является целенаправленным, а не случайным, то используется модель графа в виде И/ИЛИ-дерева. Задаётся множество начальных вершин графа, с которых поиск может начинаться, и множество конечных (целевых) вершин, при достижении которых поиск прекращается. И/ИЛИ-граф обладает следующими свойствами: при движении по входным дугам к некоторой вершине реализуется либо конъюнкция, либо дизъюнкция.

Методы поиска по дереву различаются по способу обхода путей на графе: поиск *в глубину* или *в ширину* (относительно порядка обхода вершин), *прямой* (от корня к висячей вершине)

или *обратный* поиск в глубину; поиск *без возврата* или *с возвратом*; *безусловный* или *условный* поиск (следующий ход зависит от предыдущего); *полный* или *сокращённый* перебор ходов по дереву; поиск без прогнозной оценки (метод проб и ошибок) или поиск с прогнозной оценкой (метод ветвей и границ, симплекс-метод и др.).

Для реальных задач с неполиномиальной оценкой сложности (*NP-задачи*) дерево поиска может стать вычислительно необозримым, экспоненциально растущим при линейном увеличении размерности задачи (комбинаторный взрыв). Сокращение размерности достигается либо разбиением задачи на фрагменты, либо использованием методов эвристического программирования.

С позиции теории информации каждый ход при поиске решения должен приводить к уменьшению энтропии в системе от максимального значения до нуля. Согласно Шеннону, энтропия (неопределённость) – это вероятность $p(mi)$ получения определённого сообщения из множества $M=\{m1,\ldots, mn\}$. Объем информации, содержащейся в этом сообщении, будет равен

$$I(mi) = -\log p(mi).$$

Здесь объем информации в сообщении и вероятность получения этого сообщения связаны обратной монотонной зависимостью. Энтропия множества сообщений *M* является суммой вида

$$U(M) = -\Sigma i\, p(mi) \log p(mi),\ i=1,\ldots,\ n.$$

Чем более неожиданным является сообщение, тем оно информативнее. Отсюда следует эвристика выбора при построении вершин дерева поиска: выбирается то состояние (правила, действие), которое сулит наибольший прирост информации.

При достоверном логическом выводе используется такое понятие из математической логики как общезначимость (формула является *общезначимой*, если она истинна при любых значениях входящих в нее переменных). Пусть, например, необходимо проверить общезначимость формулы

$$F = ((A+B)\rightarrow C)\rightarrow(A\rightarrow(B\rightarrow C)),$$

где *A, B, C* – булевы переменные (1 или 0); операция импликация (→) является ложной тогда и только тогда, когда посылка истинна, а заключение ложно.

Проверим общезначимость формулы несколькими методами.

В *алгоритме Квайна* для проверки используется семантическое дерево из $\mathbf{2^k}$ висячих вершин (***k*** – число переменных). Надо пройти все маршруты от корня до этих вершин.

Например, пусть A = 1. Тогда F = ((1+B)→C)→(1→(B→C)) = =C→ (B→C). Если B = 1, то F = C→(1→C) = C→C = 1. Если B = 0, то F = C→(0→C) = C→1 = 1.

Пусть теперь A = 0. Тогда F = ((0+B)→C)→(0→(B→C)) = (B→C)→1. Здесь, если B = 1, то F = 1; если B = 0, то F = 1. Следовательно, F ≡ 1.

Алгоритм редукции методом от противного использует свойства операции импликации. Предположим, что формула F ложна на некотором наборе A, B, C. Тогда f = ((A+B)→C) = 1, g = (A→(B→C)) = 0, откуда следует, что A = 1, B = 1, C = 0, однако f = ((1+1)→0) = 0. Получили противоречие. Следовательно, F является общезначимой формулой.

Наконец, если переменных в проверяемой на общезначимость формуле немного, то доказательство можно провести с использованием булевых таблиц истинности.

Ньюэлл и Саймон обосновали две гипотезы, на которых базируется большинство исследований в области ИИ. В соответствии с этими гипотезами человеческое мышление представляет собой систему, оперирующую с символами, отображающими действительность.

Гипотеза 1. Необходимым и достаточным условием для осуществления интеллектуальных действий в символьных системах является универсальность формальных манипуляций над конкретными символами (достаточно прочесть строку символов, разделить её на компоненты и переупорядочить, добавив или удалив какие-то символы без учёта семантики символов).

Гипотеза 2. Символьные системы решают задачи при помощи поиска, т.е. они генерируют потенциальные решения и модифицируют их до тех пор, пока решения не будут удовлетворять заданным условиям поиска. Поиск - это движение по дереву решений от одних узлов этого пространства к другим.

Для описания символьных систем используются языки логики или исчисления. *Исчисление* – это знаковая система, состоящая из алфавита, аксиом и процедур вывода истинных (синтаксически правильных) формул из аксиом. Если символам языка приписать конкретные значения, то это формальный язык. Различают исчисление высказываний и исчисление предикатов.

В *исчислении высказываний* вопрос об истинности или ложности высказываний решается с помощью формул, которым подчиняются логические операций конъюнкции, дизъюнкции, отрицания, импликации и др. В естественном языке высказыванием может быть повествовательное предложение, о котором можно сказать, истинно оно или ложно.

В *исчислении предикатов* наряду с формулами исчисления высказываний, используются формулы, в которые могут входить отношения (предикаты), связывающие между собой группы элементов исчисления и *кванторы общности и существования*.

Логика предикатов в отличие от логики высказываний позволяет количественно охарактеризовать связи между понятиями, их свойства и отношения.

Язык логики предикатов — один из формальных языков, наиболее приближенных к человеческому языку.

Язык искусственного интеллекта Пролог основан на логике предикатов 1-го порядка (под знаком квантора не могут находиться символы предикатов).

В исчислении высказываний из простых высказываний можно составлять сложные высказывания посредством применения логических операций, например:

Высказывание	Название операции	Обозначение
X и Y	Конъюнкция	X & Y
X или Y	Дизъюнкция	X v Y
не X	Отрицание	¬X
если X, то Y	Импликация	X→Y

Истинность высказываний, получающихся в результате этих операций, определяется по следующей таблице:

X	Y	X & Y	X v Y	¬X	X→Y
0	0	0	0	1	1
0	1	0	1	1	1
1	0	0	1	0	0
1	1	1	1	0	1

Формула является символьным представлением высказывания. Формула без операций называется *атомарной*.

Чтобы судить об истинности формулы, необходимо связать её с содержанием, т.е. выполнить *интерпретацию* формулы.

Интерпретацию будем обозначать буквой ***j***, за которой в скобках указывается обозначение интерпретируемой формулы. Например, то, что формула ***X*** в интерпретации ***j*** истинна, можно записать в виде ***j(X) = 1***.

Формула ***G*** называется *логическим следствием* формул ***F1,F2,..., Fk***, если при любой интерпретации ***j*** из ***j(F1) = j(F2) = ... = j(Fk) = 1*** следует, что ***j(G) = 1***.

Понятие логического следствия тесно связано с понятием выполнимости.

Множество формул {***F1, F2,..., Fk***} называется *выполнимым*, если существует интерпретация ***j*** такая, что ***j(F1) = j(F2) = ... = j(Fk) = 1***.

Проверить выполнимость множества формул {***F1, F2,..., Fk***} можно, построив для них таблицы истинности. Если найдется хотя бы одна строка, в которой в столбцах формул ***F1, F2,..., Fk*** стоят единицы, то это множество формул выполнимо. Если такой строки нет, то множество формул невыполнимо.

Теорема. Формула ***G*** является логическим следствием формул ***F1, F2,..., Fk*** тогда и только тогда, когда множество формул ***L = {F1, F2,..., Fk, ¬G}*** невыполнимо.

Доказательство сводится к тому, что некоторая формула ***G*** (ее называют *гипотезой* теоремы), является логическим следствием множества формул ***F1,..., Fk*** (*допущения* теоремы). Текст теоремы может быть сформулирован следующим образом: если формулы ***F1,..., Fk*** истинны, то истинна и формула ***G***.

Метод резолюций:

Задача логического вывода сводится к задаче проверки выполнимости множества формул {***F1, F2,..., Fk,*** ¬***G***}. Чтобы установить невыполнимость такого множества формул, применяется определенное правило. В этом правиле используются следующие понятия:

- *литерал* - атомарная формула или её отрицание;

- *дизъюнкт* - дизъюнкция одного или нескольких литералов;

- *пустой дизъюнкт* - специальный дизъюнкт, не содержащий литералов. Для его обозначения используется специальный символ ■ (или #)**.** Считается, что пустой дизъюнкт ложен при любой интерпретации;

- *противоположные литералы* - литералы ***X*** и ¬***X***.

Правило резолюции. Из дизъюнктов (***X*** ∨ ***F***) и (¬***X*** ∨ ***G***) выводим дизъюнкт (***F*** ∨ ***G***). Другими словами, дизъюнкт (***F*** ∨ ***G***) является логическим следствием дизъюнктов (***X*** ∨ ***F***) и (¬***X*** ∨ ***G***).

Резолюция – это приём, используемый при достоверном логическом выводе. Этот приём заключается в нахождении двух дизъюнктов, один из которых содержит литеру, а другой - её отрицание. На основании их сравнения формируется новый дизъюнкт, называемый *резольвентой*. Именно порождение новых дизъюнктов, являясь основой метода резолюций, широко применяется в интеллектуальных системах.

Пусть ***S*** - множество дизъюнктов.

Выводом из **S** называется последовательность дизъюнктов ***D1***, ***D2***,..., ***Dn*** такая, что каждый дизъюнкт этой последовательности принадлежит **S** или следует из предыдущих по правилу резолюции.

Дизъюнкт **D** выводим из **S**, если существует вывод из **S**, последним дизъюнктом которого является **D**.

Применение метода резолюций основано на следующей теореме.

Теорема (о полноте метода резолюций). Множество дизъюнктов логики высказываний **S** невыполнимо тогда и только тогда, когда из **S** выводим пустой дизъюнкт ■ .

Алгоритм резолютивного вывода состоит в доказательстве того, что формула **G** является логическим следствием множества формул ***F1***,..., ***Fk*** и включает следующую последовательность шагов.

1. Составляется множество формул {***F1***,..., ***Fk***, ¬***G***}.

2. Каждая из этих формул приводится к конъюнктивной нормальной форме (КНФ)- конъюнкции элементарных дизъюнкций. В КНФ зачеркиваются знаки конъюнкции. Получается множество дизъюнктов ***S***.

3. Осуществляется поиск вывода пустого дизъюнкта ■ из **S**. Если пустой дизъюнкт выводим из **S**, то формула **G** является логическим следствием формул **F1**,..., **Fk**. Если из **S** нельзя вывести ■, то **G** не является логическим следствием формул **F1**,..., **Fk** .

Рассмотрим несколько примеров вывода по методу резолюций.

Пример 1. Пусть даны два утверждения:

1) «Яблоко красное и ароматное»,

2) «Если яблоко красное, то яблоко вкусное».

Докажем утверждение, что «Яблоко вкусное».

Для этого введем множество формул, описывающих простые высказывания, соответствующие приведённым выше утверждениям:

X1 – «Яблоко красное»,

X2 – «Яблоко ароматное»,

X3 – «Яблоко вкусное».

Исходные утверждения запишем в виде следующих формул:

X1 & **X2** — «Яблоко красное и ароматное»,

X1→X3 — «Если яблоко красное, то яблоко вкусное».

Утверждение, которое надо доказать, выражается формулой **X3**.

Докажем, что **X3** является логическим следствием формул (**X1** & **X2**) и (**X1→X3**). Для этого составляем множество формул с отрицанием доказываемого высказывания:

{(**X1**&**X2**), (**X1→X3**), ¬**X3**}.

Приводим все формулы к КНФ:

{(**X1**&**X2**), (¬**X1** v **X3**), ¬**X3**}.

Зачёркивая конъюнкции, получаем следующее множество дизъюнктов:

S = {**X1**, **X2**, (¬**X1** v **X3**), ¬**X3**}.

Ищем вывод пустого дизъюнкта:

X1, (¬**X1** v **X3**), **X3**, ¬**X3**, ■.

Получили пустой дизъюнкт. Следовательно, утверждение о том, что яблоко вкусное, верно.

Пример 2. Пусть даны два утверждения:

1) **X1→X2** ,

2) **X2→X3** .

Докажем, что утверждение **X1→X3** является логическим следствием из этих утверждений. Для этого составляем множество формул с отрицанием доказываемого высказывания:

{(**X1→X2**), (**X2→X3**), ¬(**X1→X3**)}.

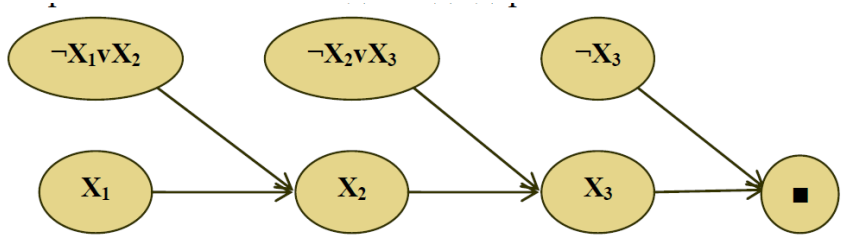
Устраняем импликации и приводим все формулы к КНФ:

{(¬**X1** v **X2**), (¬**X2** v **X3**), ¬(¬**X1** v **X3**)} = {(¬**X1** v **X2**), (¬**X2** v **X3**), (**X1**&¬**X3**)}.

Зачёркивая конъюнкции, получим множество из четырёх дизъюнктов:

S = {(¬**X1** v **X2**), (¬**X2** v **X3**), **X1**, ¬**X3**}.

Представим резолютивный вывод в виде дерева поиска:



Получен пустой дизъюнкт. Следовательно, утверждение ***X1→X3*** является логическим следствием утверждений ***X1→X2*** и ***X2→X3*** .

Отметим также, что известное правило логического вывода «*modus ponens*» (если ***X*** - истина и из ***X→G***, то ***G*** - истина) получается по правилу резолюции: из дизъюнктов (***X***) и (***¬X∨G***) выводим дизъюнкт ***G***. Кроме того, метод резолюций является обобщением метода доказательства от противного.

Метод резолюций является *частично корректной процедурой*. Потому что возможен следующий исход: процесс не заканчивается, правило резолюции применяется, множество предложений пополняется, среди них нет пустых, и есть резольвируемые. Исход нельзя назвать зацикливанием, поскольку нет никакого способа определить, почему метод резолюции продуцирует новые резольвенты, но при этом не получается пустого предложения. Означает ли это, что теорема верна, но мы просто не дождались завершения? Не известно. Означает ли это, что в результате так никогда и не получится пустой формулы? И этого мы тоже не знаем. Это логически неразрешимая задача. В принципе нельзя узнать, как долго нужно ждать для того, чтобы получить ответ на этот вопрос.

Метод резолюций применим также к логике предикатов 1-го порядка. Для этого требуется внести дополнения к версии метода для логики высказываний. В частности, в исчислении предикатов используется понятие *переменной*. Поэтому правило резолюций необходимо дополнить возможностью делать *подстановку* переменной.

В методе резолюций для логики предикатов используются еще два правила *склейки*.

В остальном всё сводится к проверке невыполнимости множества дизъюнктов: множество дизъюнктов **S** логики первого порядка невыполнимо тогда и только тогда, когда из **S** выводим пустой дизъюнкт ■ .

Например, пусть даны два утверждения:

1) «Существуют студенты, которые любят всех преподавателей»,

2) «Ни один из студентов не любит невежд».

Докажем утверждение «Ни один преподаватель не является невеждой».

Введем множество формул, описывающих следующие простые утверждения:

C(x) – **x** есть студент;

P(y) – **y** есть преподаватель;

H(y) – **y** есть невежда;

L(x, y) – **x** любит **y**.

Тогда два исходных утверждения можно записать в виде формул:

1) $\mathbf{\forall x(C(x)\&\forall y(P(y)\rightarrow L(x,y)))}$;

2) $\mathbf{\forall x(C(x)\rightarrow\forall y(H(y)\rightarrow\neg L(x,y)))}$.

Утверждение, которое надо доказать, выражается следующей формулой:

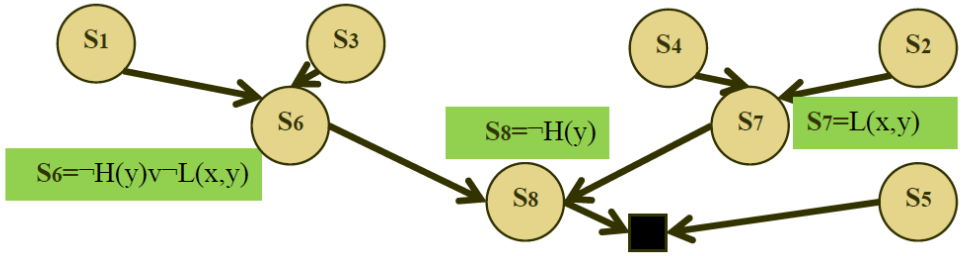
y(P(y)→¬H(y)).

Применяя специальный алгоритм, приводим утверждения к набору дизъюнктов **S** = {**S1, S2, S3, S4, S5**}:

S1 = C(x); **S2** = ¬P(y) v L(x, y); **S3** = ¬C(x) v ¬H(y) v ¬L(x, y));

S4 = P(y); **S5** = H(y).

Дерево вывода для данного примера, построенное согласно методу резолюций для логики предикатов 1-го порядка, имеет вид, представленный на рисунке:



Если проанализировать механизмы ПР человека или компьютерного алгоритма, использующих информацию о среде ПР, то можно заметить, что в их основе, в том или ином виде лежит одно из следующих правил вывода:

1. Дедуктивное правило modes ponens (или правило отделения):

$$\frac{\text{если Н то А;}\quad \text{Н - истинно;}}{\text{А – истинно.}} \qquad (1)$$

2. Правило вывода по индукции:

$$\frac{\text{если Н то А;}\quad \text{А - истинно;}}{\text{Н – более правдоподобно.}} \qquad (2)$$

3. Правило вывода по аналогии:

$$\frac{\text{если Н то А;}\quad \text{если Н то В;}\quad \text{А – истинно;}}{\text{В – истинно по аналогии.}} \qquad (3)$$

Здесь посылка **Н** и следствия **А** и **В** – некоторые высказывания (факты) о процессе ПР в конкретных ситуациях.

Пример 1. Согласно ОСТ для того, чтобы нарезать резьбу М6 шагом 1мм в стальном листе необходимо просверлить отверстие диаметром 4,95 мм. Отвечая на вопрос «каким должно быть отверстие в стальном листе, чтобы нарезать резьбу М6 шагом 1мм», человек по справочнику или информационно-справочная система по своей базе данных определяя ответ «диаметр 4,7мм» реализуют дедуктивное правило modus ponens. Здесь в качестве высказывания **Н** выступает предложение «резьба М6 и шаг 1мм», а в качестве высказывания **А** – предложение «диаметр 4,7мм». Здесь используется дедуктивный вывод.

Пример 2. Автомобиль плохо заводится (или не заводится вообще) если неисправен аккумулятор. Когда владелец автомобиля обнаруживает, что последний не заводится, он проверяет аккумулятор. В этой ситуации реализуется вывод по индукции. Здесь посылкой **Н** выступает предложение «неисправен аккумулятор», а следствием **А** - предложение «автомобиль плохо заводится». По наблюдаемому следствию **А** владелец автомобиля делает предположение о правдоподобности посылки **Н**. Здесь используется андуктивный вывод.

Пример 3. Написав письмо Вы решили переслать его по электронной почте с помощью компьютера, стоящего на Вашем столе. В это время настольная лампа, также стоящая на столе, начинает мигать. Вы делаете вывод, что компьютер включать не стоит. В этом примере в неявном виде реализуется вывод по аналогии. Здесь посылкой **H** является высказывание типа «сбой в электросети», а следствиями **A** – «лампа мигает» и **B** – «компьютер не включать». Наблюдая факт «лампа мигает» Вы без вольтметра и осциллографа делаете предположение о посылке **H** «сбой в электросети», а затем по факту **H** и правилу «если **H** то **B**» делаете вывод о следствии **B** – «компьютер не включать». Здесь используется вывод по аналогии.

Пример 4. Если на улице идет дождь, то тротуар будет влажным. Если на улице идет дождь, то выходя из квартиры вы берете зонт. Утром вы смотрите в окно и замечаете, что тротуар влажный и выходя из квартиры вы берете зонт. Здесь посылкой **H** является высказывание типа «идет дождь», а следствиями **A** – «влажный тротуар» и **B** – «взять зонт». Наблюдая факт «влажный тротуар» Вы делаете предположение о посылке **H** «идет дождь», а затем по факту **H** и правилу «если **H** то **B**» делаете вывод о следствии **B** – «взять зонт». Здесь в рассуждениях также был использован вывод по аналогии.

5. Технологии инженерии знаний

Знания являются основой любой интеллектуальной информационной системы (ИИС).

Научное направление, занимающееся вопросами формализации, представления и обработки знаний в информационных системах, называют ***инженерией знаний***.

В 1977 г. американский ученый Э. Фейгенбаум писал: «По опыту нам известно, что большая часть знаний в конкретной предметной области остается личной собственностью эксперта. И это происходит не потому, что он не хочет разглашать своих секретов, а потому, что он не в состоянии сделать этого, — ведь эксперт знает гораздо больше, чем сам осознает».

Данное высказывание констатирует существование так называемых неформальных, или неявных, знаний, связанное с рядом факторов, в частности:

- часть экспертных знаний носит неосознаваемый характер;
- эксперт не всегда способен оценить важность тех или иных знаний для принятия решения;
- опыт, накопленный экспертом, сложно вербализовать и представить в формализованном виде.

Инженерия знаний ***изучает:***

- извлечение знаний из экспертов и (или) литературы;
- формализация и обработка знаний;
- проектирование и разработка баз знаний.

Знания представляют собой более сложную информационную категорию по сравнению с данными.

Данные несут в себе фактическую информацию, связанную с состоянием объектов, процессов, явлений предметной области.

Знания описывают не только данные, но и взаимосвязи между ними, поэтому их иногда называют структурированными данными. Знания представляют собой результат мыслительной деятельности человека, направленной на обобщение его опыта.

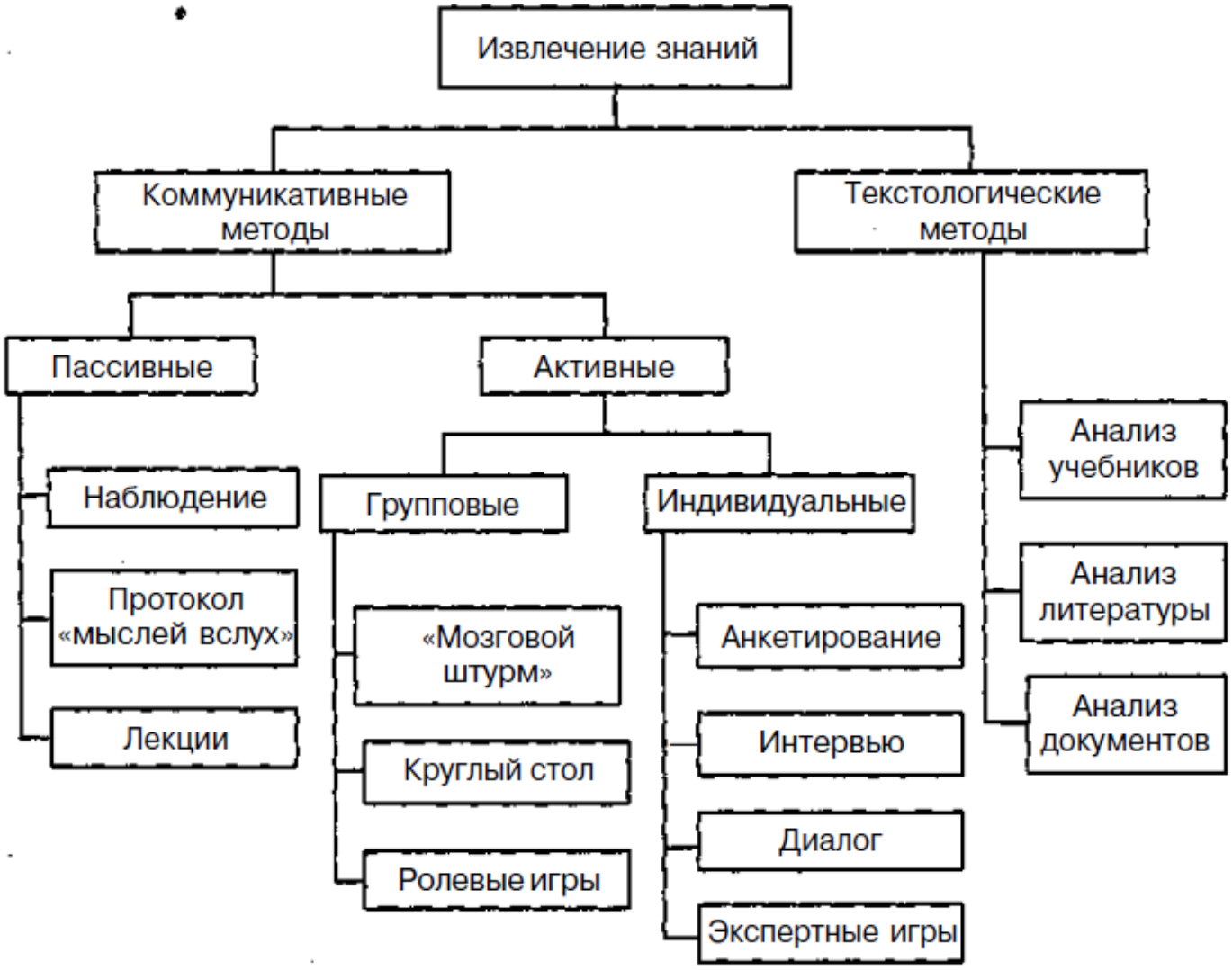


Рис.5.1. Классификация методов извлечения знаний

Основной принцип деления связан с источником знаний.

Коммуникативные методы можно также разделить на две группы: *активные и пассивные*.

Пассивные методы подразумевают, что ведущая роль в процедуре извлечения как бы передается эксперту, а инженер по знаниям только протоколирует рассуждения эксперта во время его реальной работы по принятию решений или записывает то, что эксперт считает нужным самостоятельно рассказать в форме лекции.

В ***активных*** методах, напротив, инициатива полностью в руках инженера по знаниям, который активно контактирует с экспертом различными способами — в играх, диалогах, беседах за круглым столом и т. д.

Пассивные методы достаточно просты, но требуют от инженера по знаниям умения четко анализировать поток сознания эксперта и выявлять в нем значимые фрагменты знаний. *Отсутствие обратной связи* ослабляет эффективность этих методов.

Активные методы можно разделить на две группы: в зависимости от числа экспертов, отдающих свои знания.

Если их число больше одного, то целесообразно помимо серии индивидуальных контактов с каждым применять и методы групповых обсуждений предметной области.
Индивидуальные методы на сегодняшний день остаются ведущими.

Игровые методы сейчас широко используются в социологии, экономике, менеджменте, педагогике для подготовки руководителей, учителей, врачей и других специалистов. Игра — это особая форма деятельности и творчества, где человек раскрепощается и чувствует себя намного свободнее, чем в обычной трудовой деятельности.
На выбор метода влияют три фактора: личностные особенности инженера по знаниям, личностные особенности эксперта и характеристика предметной области.

Классификация людей по психологическим характеристикам:

• мыслитель (познавательный тип);

• собеседник (эмоционально-коммуникативный тип);

• практик (практический тип).

Мыслители ориентированы на интеллектуальную работу, учебу, теоретические обобщения и обладают такими характеристиками когнитивного стиля, как поле независимость и рефлексивность.

Собеседники - это общительные, открытые люди, готовые к сотрудничеству.

Практики предпочитают действие разговорам, хорошо реализуют замыслы других, направлены на результативность работы.

Для характеристики предметных областей можно предложить следующую классификацию:

• хорошо документированные;

• средне документированные;

• слабо документированные.

Пусть Z_1 —экспертное «личное» знание, а Z_2 — материализованное в книгах «общее» знание в данной конкретной области.

На рисунке 5.2 показана предметных областей:

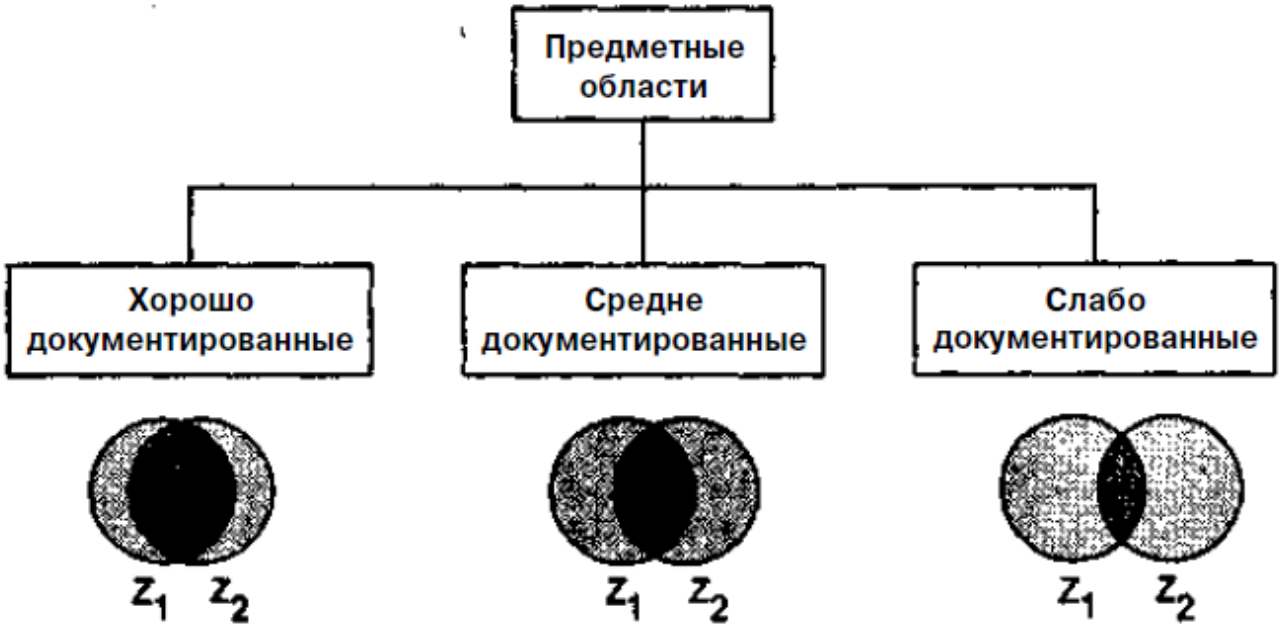


Рис.5.2. Классификация

Предметные области можно разделить по критерию структурированности знаний.

Под ***структурированностью*** понимается степень теоретического осмысления и выявленности основных закономерностей и принципов, действующих в данной предметной области.

По степени структурированности знаний предметные области могут быть:

• *хорошо структурированными* — с четкой аксиоматизацией, широким применением математического аппарата, устоявшейся терминологией;

• *средне структурированными* — с определившейся терминологией, развивающейся теорией, явными взаимосвязями между явлениями;

• *слабо структурированными* — с размытыми определениями, богатой эмпирикой, скрытыми взаимосвязями, с большим количеством «белых пятен».

Коммуникативные методы: ***пассивные и активные***.

К пассивным методам относятся: *наблюдения; анализ протоколов «мыслей вслух»; лекции.*

Наблюдения. В процессе наблюдений инженер по знаниям находится непосредственно рядом с экспертом во время его профессиональной деятельности или имитации этой деятельности. При подготовке к сеансу извлечения эксперту необходимо объяснить цель наблюдений и попросить максимально комментировать свои действия.

Во время сеанса аналитик записывает все действия эксперта, его реплики и объяснения. Может быть сделана и видеозапись в реальном масштабе времени. Непременное условие этого метода — невмешательство аналитика в работу эксперта хотя бы на первых порах. Именно метод наблюдений является единственно «чистым» методом, исключающим вмешательство инженера по знаниям и навязывание им каких-то своих структур представлений.

Существуют *две основные разновидности* проведения наблюдений:

- наблюдение за реальным процессом;
- наблюдение за имитацией процесса.

Протоколирование «***мыслей вслух***» отличается от наблюдений тем, что эксперта просят не просто прокомментировать свои действия и решения, но и объяснить, как это решение было найдено, то есть продемонстрировать всю цепочку своих рассуждений.

Во время рассуждений эксперта все его слова, весь «поток сознания» протоколируется инженером по знаниям, при этом полезно отметить даже паузы и междометия.

Трудность: при протоколировании «мыслей вслух» является принципиальная сложность для любого человека объяснить, как он думает. При этом существуют экспериментальные психологические доказательства того факта, что люди не всегда в состоянии достоверно описывать мыслительные процессы. Кроме того, часть знаний, хранящихся в невербальной форме вообще слабо коррелируют с их словесным описанием.

Лекция - самый старый способ передачи знаний.

В лекции эксперту также предоставлено много степеней свободы для самовыражения; при этом необходимо сформулировать эксперту тему и задачу лекции.

Метод извлечения знаний в форме лекций, как и все пассивные методы, используют в начале разработки как эффективный способ быстрого погружения инженера по знаниям в предметную область.

Активные индивидуальные методы извлечения знаний — наиболее распространенные. К основным активным методам можно отнести:

- анкетирование;
- интервью;
- свободный диалог;
- игры с экспертом.

Анкетирование — наиболее стандартизированный метод. В этом случае инженер по знаниям заранее составляет вопросник или анкету, размножает ее и использует для опроса нескольких экспертов. (основное преимущество анкетирования).

Процедура может проводиться двумя способами:

1. Аналитик вслух задает вопросы и сам заполняет анкету по ответам эксперта.
2. Эксперт самостоятельно заполняет анкету после предварительного инструктирования.

Под ***интервью*** будем понимать специфическую форму общения инженера по знаниям и эксперта, в которой инженер по знаниям задает эксперту серию заранее подготовленных вопросов с целью извлечения знаний о предметной области.

Три основные характеристики вопросов, которые влияют на качество интервью:

- язык вопроса (понятность, лаконичность, терминология);
- порядок вопросов (логическая последовательность и немонотонность);
- уместность вопросов (этика, вежливость).

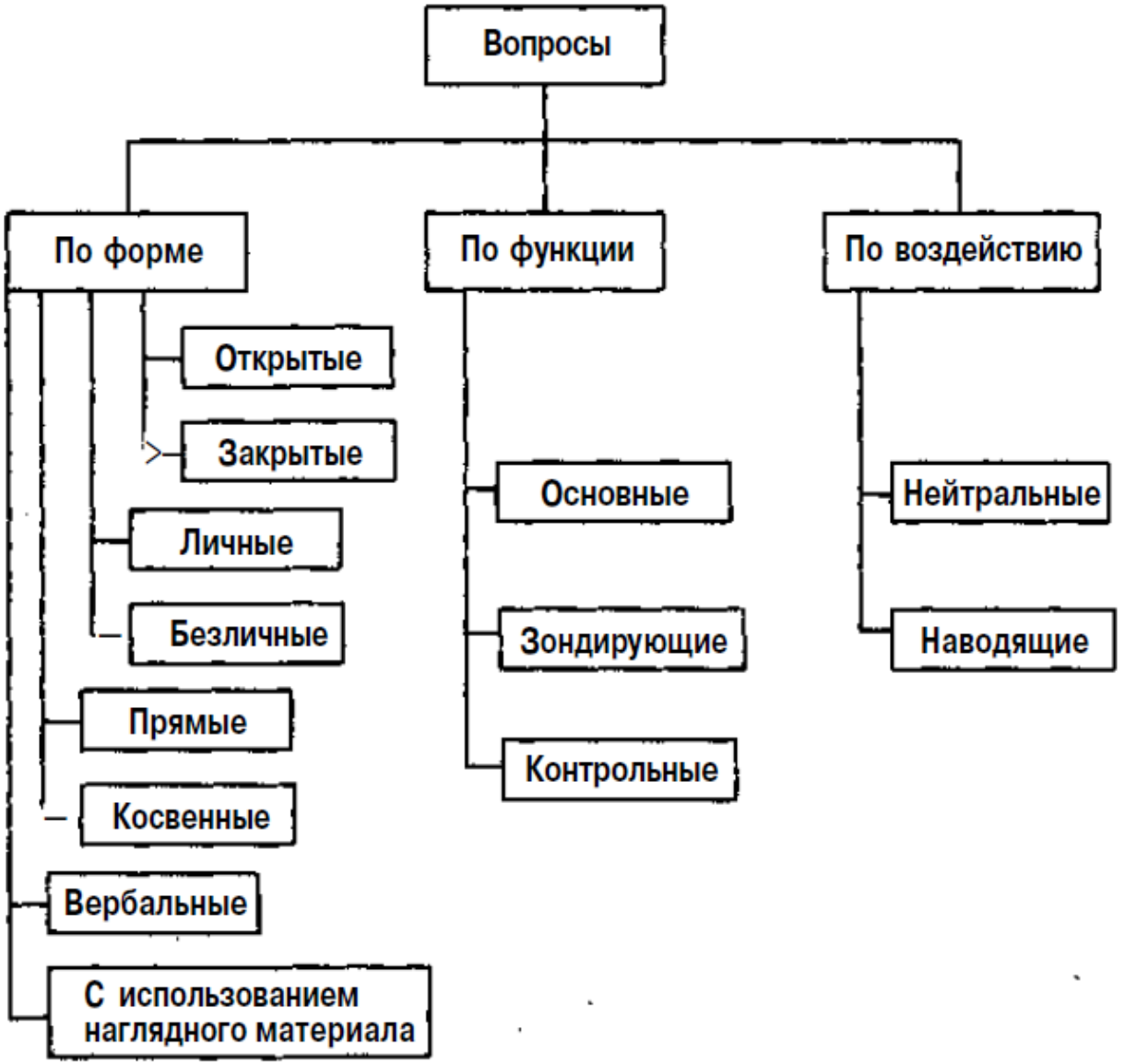


Рис.5.3. Классификация вопросов для интервью

Свободный ***диалог*** — это метод извлечения знаний в форме беседы инженера по знаниям и эксперта, в которой нет жесткого регламентированного плана и вопросника.

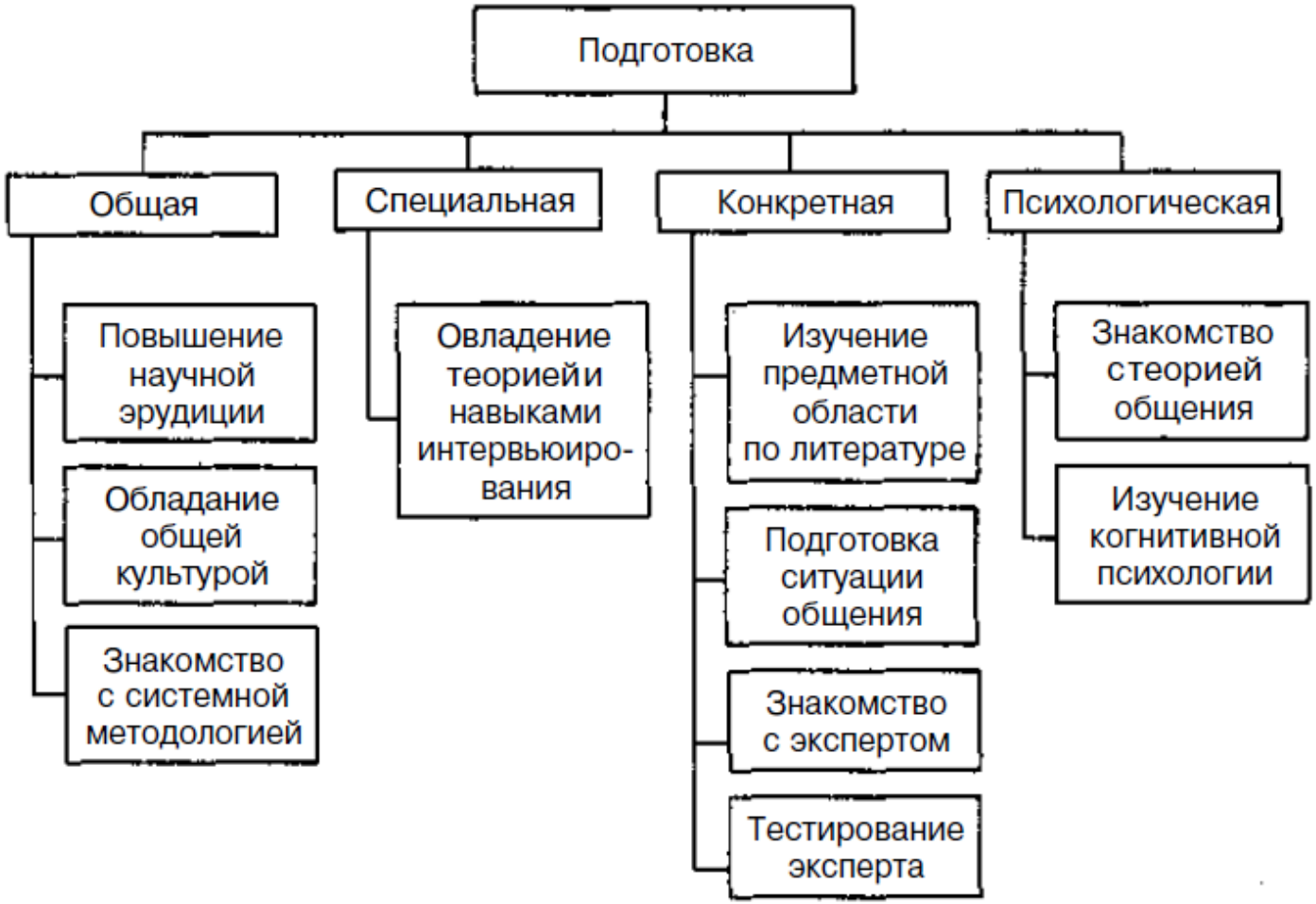


Рис.5.4. Подготовка к интервью

Активные групповые методы: *ролевые игры*, *дискуссии за «круглым столом»* с участием нескольких экспертов и «*мозговые штурмы*». Основное достоинство групповых методов — это возможность одновременного «поглощения» знаний от нескольких экспертов, взаимодействие которых вносит в этот процесс элемент принципиальной новизны от наложения разных взглядов и позиций.

Метод круглого стола предусматривает обсуждение какой-либо проблемы из выбранной предметной области, в котором принимают участие с равными правами несколько экспертов. Обычно вначале участники высказываются в определенном порядке, а затем переходят к живой свободной дискуссии. Число участников дискуссии колеблется от трех до пяти - семи.

«***Мозговой штурм***» или «мозговая атака» — один из наиболее распространенных методов раскрепощения и активизации творческого мышления.

Впервые этот метод был использован в 1939 г. в США как способ получения новых идей в условиях запрещения критики. Замечено, что боязнь критики мешает творческому мышлению, поэтому основная идея штурма - это отделение процедуры генерирования идей в замкнутой группе специалистов от процесса анализа и оценки высказанных идей.

Игрой называют такой вид человеческой деятельности, который отражает (воссоздает) другие ее виды. При этом для игры характерны одновременно условность и серьезность.

Понятие экспертной игры или игры с экспертами в целях извлечения знаний восходит к трем источникам — это деловые игры, широко используемые при подготовке специалистов и моделировании, и компьютерные игры, часто применяемые в обучении.

Под ***деловой игрой*** чаще всего понимают эксперимент, где участникам предлагается производственная ситуация, а они на основе своего жизненного опыта, своих общих и специальных знаний и представлений принимают решения.

Диагностическая игра — это та же деловая игра, но применяемая конкретно для диагностики методов принятия решения в медицине (диагностика методов диагностики). Эти игры возникли при исследовании способов передачи опыта от опытных врачей новичкам.

Плодотворность моделирования реальных ситуаций в играх подтверждается сегодня практически во всех областях науки и техники. Они развивают логическое мышление, умение быстро принимать решения, вызывают интерес у экспертов.

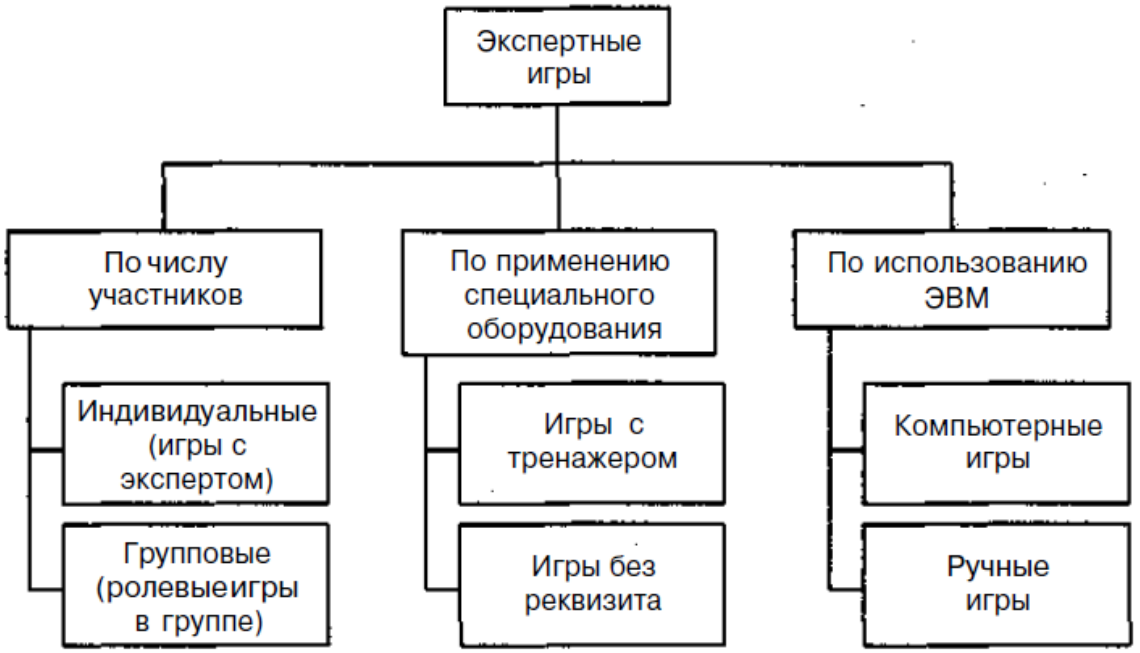


Рис.5.5. Классификация экспертных игр

6. Гипертекстовые, многоагентные и онтологические системы

Текст – универсальное средство представления, накопления и передачи знаний.

Гипертекст (ГТ) – это сеть текстов, в которой имеются ссылки на другие фрагменты текста.

ГТ – одна из фундаментальных моделей представления знаний. Если обычный текст – это строка символов, читаемая в одном направлении, то ГТ многомерен, его можно читать «нелинейно», двигаясь по разным траекториям сети.

Гипертекстовая информационная технология (ГИТ) – это технология поиска и обработки семантической ГТ-информации.

ГТ-документ может быть не только электронным, но и бумажным (энциклопедия), если в нем расставлены ссылки.

Д. Энгельбарт (изобретатель электронной мыши) предложил рассматривать отношения людей и программ как гетерогенное сообщество. Его проект NLS (60- годы) был нацелен на расширение познавательных возможностей группы людей: групповые переговоры по системе электронной почты, личные настройки пользователей и т.п., которые нашли широкое применение относительно недавно.

Т. Нельсон работал над созданием системы электронных публикаций и всеобщего архива. Предложил термин «***гипертекст***». Подчеркивал, что гипертекст в его понимании не является иерархической структурой. Информационных структур не могут быть верно представлены иерархией из-за их параллелизма, перекрестных связей, одновременного присутствия одного элемента в нескольких местах.

Гипертекст – это многоагентная структура, внутри которой могут существовать сложные неиерархические отношения между агентами.

Модели гипертекста:

Формализованная модель. В основе модели лежит понятие *информационно-справочной статьи (ИСС)* – это объект, имеющий смысловое единство и включающий локальные и глобальные *гиперссылки*. Графика и мультимедиа, содержащие гиперссылки, называют *гиперграфикой* и *гипермедиа.*

Формализованная модель ГТ состоит из 2-х слоев.

Первый слой – это отображение на экране содержимого документа с выделенными цветом гиперссылками.

Во втором скрытом слое находятся адреса переходов.

ИСС описывает кортеж <**x0, x1,…, x11>, где**

x0 –имя ИСС; x1 – заголовок;

x2 – аннотация;

x3 – точка входа в ИСС;

x4 – текст-фрагменты ИСС;

x5 – цифровые объекты ИСС;

x6 – программные объекты;

x7 – справка по ИСС;

x8 – признак ускоренного просмотра;

x9 – признак детального просмотра;

x10 – список локальных гиперссылок;

x11 – список глобальных гиперссылок.

Недостаток модели - отсутствие возможности явного определения типов гиперссылок.

Тезаурусная модель гипертекста. Имеет вид:

$$\Gamma_0 = (T, I, S, Q)$$

где **T** – тезаурус гипертекста, состоящий из тезаурусных статей $\mathbf{t_i}$; **I** – информационная составляющая гипертекста, состоящая из информационных статей I_i ($I = U\ I_i$); **S** – алфавитный словарь терминов; **Q** – список главных тем гипертекста.

Тезаурус Т – это семантически упорядоченный перечень терминов предметной области;

Тезаурусная статья - $t_i = \{m_i, A_{mi}\}$,

m_i - термин, A_{mi} - множество терминов, с которыми m_i связан семантическими отношениями R.

Выделено ***200*** *семантических типов отношений.*

Наиболее часто употребляются **8** типов: *синоним, род-вид, вид-род, часть-целое, целое-часть, причина-следствие, следствие-причина и ассоциация.*

Гипертекстовые информационные системы

Зипф предположил, что слова с большим количеством букв встречаются в тексте реже коротких слов.

Два универсальных закона.

• Первый закон Зипфа «ранг – частота»;

• Второй закон Зипфа «количество – частота».

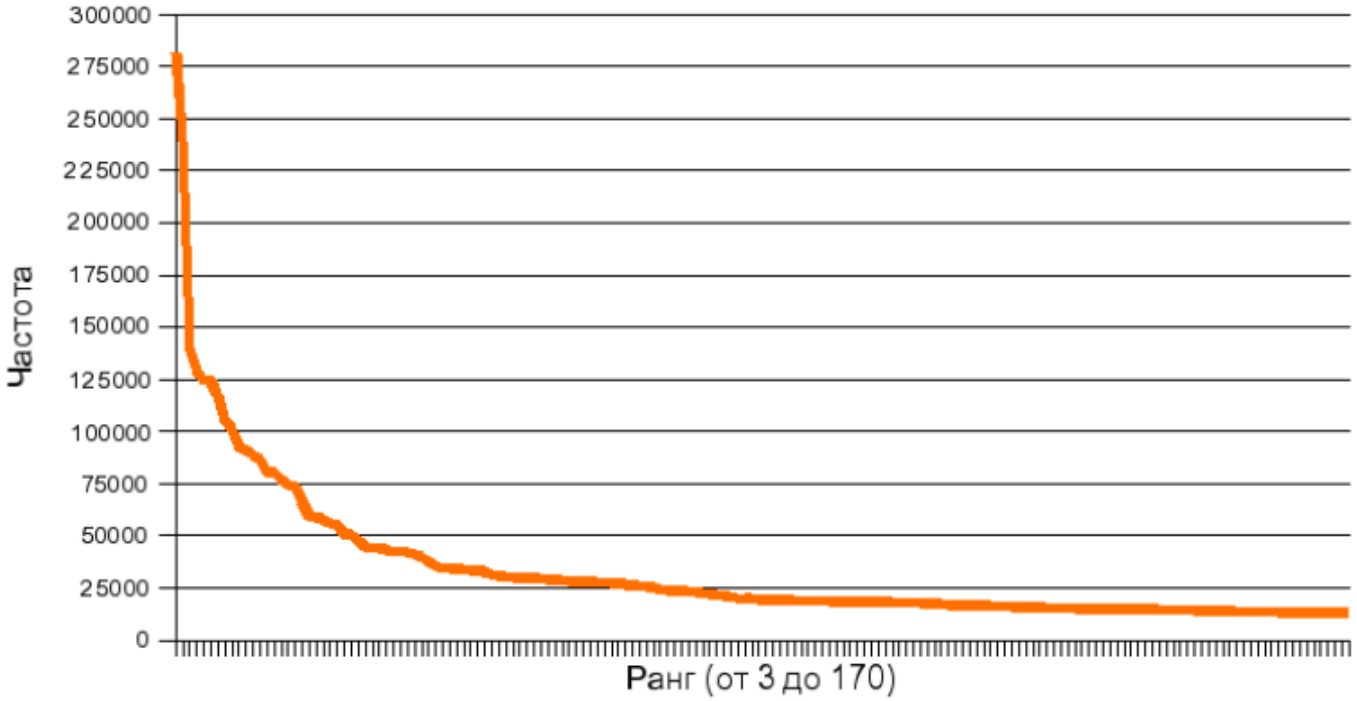
Первый закон Зипфа. Выбрав любое слово можно подсчитать, сколько раз оно встречается в тексте. Эта величина называется частотой вхождения слова. Далее, можно измерить частоту каждого слова текста. Некоторые слова будут иметь одинаковую частоту. Их нетрудно сгруппировать, пронумеровать и расположить группы по мере убывания их частоты. Порядковый номер частоты назовём рангом частоты. Наиболее часто встречающиеся слова будут иметь ранг 1, следующие за ними - 2 и т.д. Вероятность встретить в тексте наугад выбранное слово будет равна отношению частоты вхождения этого слова к общему числу слов в тексте:

Вероятность: = Частота вхождения слова / Число слов

Зипф обнаружил закономерность: Если умножить вероятность обнаружения слова в тексте на ранг частоты, то получившаяся величина (*C*) приблизительно постоянна!

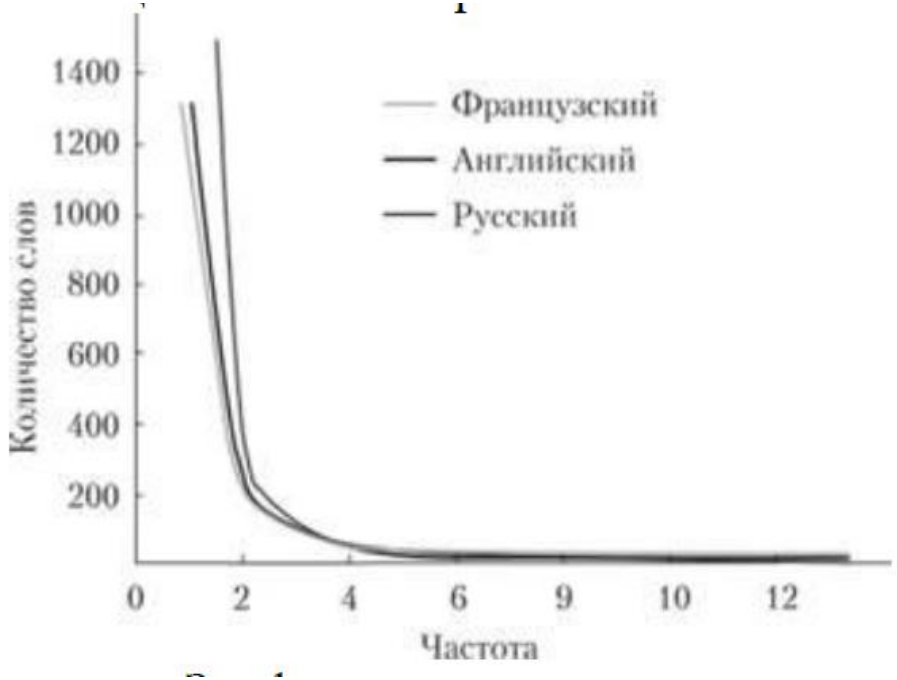
*C: = (Частота вхождения слова * Ранг частоты) / Число слов*

Таким образом, ранг (x) и частота (y) связаны формулой вида $y = k/x$:



Второй закон Зипфа. Разные слова входят в текст с одинаковой частотой. Зипф установил, что частота и количество слов, входящих в текст с этой частотой, тоже связаны между собой.

Если построить график, отложив по одной оси (оси ***X***) частоту вхождения слова, а по другой (оси ***Y***) -- количество слов в данной частоте, то получившаяся кривая будет сохранять свои параметры для всех без исключения созданных человеком текстов:



Наиболее значимые слова лежат в средней части гиперболы.

Часто встречающиеся слова - предлоги, местоимения, артикли и т.п.

Редко встречающиеся слова не имеют решающего смыслового значения.

Каждая поисковая система решает вопрос о выборе диапазона по-своему, руководствуясь общим объемом текста, словарями и т.п.

Матричное представление базы документов.

Запросы могут быть простыми (одно слово) и сложными (несколько слов, связанных логическими операторами).

Взаимодействие со сложными запросами требует изощренной организации базы документов.

Наиболее простой способ представить элементы базы - создать матрицу «Документ-термин».

База включает 8 документов (Д1, Д2, …, Д8), в которых содержатся 11 терминов.

Если термин входит в документ, в соответствующей клетке ставится 1, иначе – 0:

Документы / *Термины*	**Д1**	**Д2**	**Д3**	**Д4**	**Д5**	**Д6**	**Д7**	**Д8**
Алкоголизм	0	1	0	0	1	0	0	0
Бутылка	1	1	0	1	0	0	1	0
Вино	0	1	0	1	0	0	1	0
Корабль	1	0	0	0	0	0	0	1
Модель	1	0	0	0	0	1	0	1
Море	0	1	1	0	0	1	0	0
Парус	0	0	1	0	0	1	0	1
Пиво	0	0	0	1	1	0	0	0
Судомоделизм	0	0	1	0	0	0	0	0
Урожай	0	0	0	1	1	0	1	0
Хобби	0	0	1	0	0	0	0	1

Пространственно-векторное представление базы документов.

Пространственно-векторная модель позволяет получить результат, хорошо согласующийся с запросом.

Документ может оказаться полезным, даже не имея 100% соответствия.

В найденном документе может вовсе не оказаться одного или нескольких слов запроса, но при этом его смысл будет запросу соответствовать. Как достигается такой результат?

Все документы в базе размещаются в воображаемом многомерном пространстве. Координаты каждого документа зависят от структуры терминов, содержащихся в них.

В результате окажется, что документы с похожим набором терминов разместятся в пространстве ближе друг к другу.

Методика поиска информации в сети Интернет.

Алгоритм:

- Удаляем из текста стоп-слова.

- Вычисляем частоту вхождения каждого термина без учета морфологии слов и регистра.

- Сформируем лист терминов в порядке убывания их частоты вхождения.

- Выбираем примерно посередине диапазон частот, ориентируясь на конкретный смысл текста.

- Из выбранного диапазона выбираем 10-20 терминов.

- Составляем запрос и отправляем его поисковой системе.

В ответ вы можете получить несколько миллионов ссылок. Поисковая машина ранжирует результаты, на первых страницах окажутся практически стопроцентно релевантные документы.

Модели автоматизации поиска.

- поиск релевантной по составу и содержанию гипертекстовой статьи:

- поиск гипертекстовых статей с наиболее желательными свойствами, например наличие одинакового родового объекта;

- комбинированная модель.

Эффективность информационного поиска принято оценивать по *информационной полноте (kП)* и *информационному шуму (kШ)* на интервале [0,1].

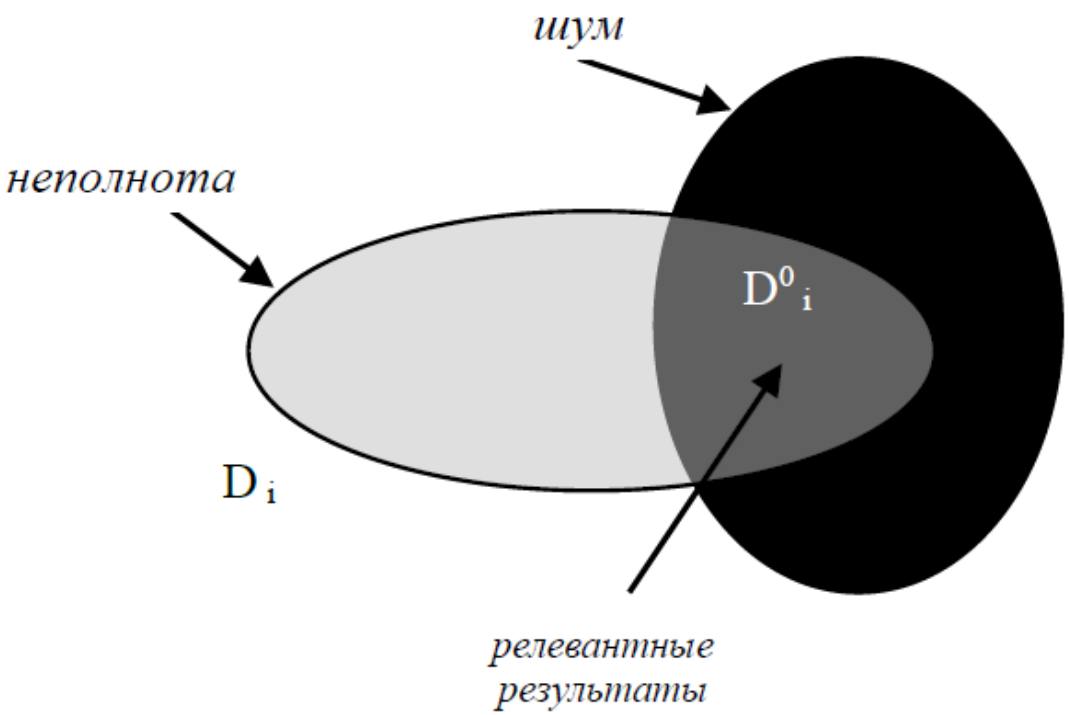
Идеальный вариант: полнота максимальна *(kП =1),* а шум нулевой *(kШ=0).*

Коэффициенты информационной *полноты* и *шума* определяются следующим образом:

$$k_{П} = \lim_{k \to m} \frac{1}{k} \sum_{i=1}^{k} \frac{| D_i \cap D_i^0 |}{| D_i |} \quad \text{и} \quad k_{Ш} = \lim_{k \to m} \frac{1}{k} \sum_{i=1}^{k} \frac{| D^0{}_i \setminus D_i |}{| D^0{}_i |}$$

m – достаточно большое число (по теореме о больших числах), $D^0{}_i$– подмножество релевантных документов из *Di*, полученных по *i*-запросу.

Смысл коэффициентов полноты и шума:



В идеале $D^0_i = D_i$. Это дает возможность ввести оценку:

$$E_1 = 2 \lim_{k \to m} \frac{1}{k} \sum_{i=1}^{k} \frac{| D_i \cap D_i^0 |}{| D_i | + | D_i^0 |}$$

Коэффициент точности: $k_T = 1 - k_Ш$.

Тогда оценка *эффективности информационного поиска* равна:

$$\boldsymbol{E_1 = 2k_Tk_П / (k_T + k_П)}$$

Обобщенный комплексный показатель эффективности E_β (*мера Ван Ризбергена*:

$$E_\beta = \frac{(\beta^2 + 1)k_T k_П}{\beta^2 k_T + k_П}$$

где β – параметр, отражающий предпочтение пользователя ИПС ($\beta \in [0, \infty]$).

При $\beta = 1$ точность и полнота одинаково важны.

На интервале [0, 1] приоритет имеет точность, а на интервале [1, ∞] – полнота.

Частные случаи:

- $\boldsymbol{E_{\beta=1} = E_1}$**;**
- $\boldsymbol{E_{\beta=0}} = \boldsymbol{k_T}$ **(**значима только точность**);**
- $\boldsymbol{E_{\beta\to\infty}} = \boldsymbol{k_П}$ **(**значима только полнота**).**

Сравнение гипертекстовых, фактографических и документальных информационно-поисковых систем

3 типа информационно-поисковых систем (ИПС):

• *документальные;*

• *фактографические* (хранят не документы, а факты о предметной области, например в виде реляционных БД). Поиск в них в отличие от документального является полным и точным,

• *гипертекстовые* (хранят документы и их семантическую структуру).

Для документальных ИПС: $\mathbf{k_{Пmax}} = 0{,}5$; $\mathbf{k_{Шmax}} = 1$;

Для фактографических ИПС: $\mathbf{k_{Пmax}} = 1$; $\mathbf{k_{Шmax}} = 0$;

Для гипертекстовых ИПС: $\mathbf{k_{Пmax}} = \mathbf{0{,}9 \div 1}$; $\mathbf{k_{Шmax}} = \mathbf{0{,}1 \div 0{,}2}$.

Поисковый агент – самый интеллектуальный компонент поисковой машины. Различают ***агенты:***

• *кроулеры* (просматривают заголовки страниц и возвращают машине только первую найденную ссылку),

• «*роботы*» (проходят по ссылкам с различной глубиной),

• «*пауки*» (сообщают о содержимом найденного документа, индексируют его и пересылают информацию в БД машины поиска).

Автоматическое реферирование и аннотирование

Реферат – это доклад, излагающий краткое содержание работы на определенную тему с обзором библиографических источников.

Аннотация – это краткий реферат с указанием категории читателей, для которых предназначено аннотируемое произведение.

Основные методы автоматического реферирования и аннотирования: поверхностные (извлекают из текста обороты типа «идея состоит…» и т.п.), объединяют их в реферат; глубинные (используют тезаурусы и синтаксический разбор текста).

Проблемы: составление реферата из разноязычных и разнотипных работ, сборника докладов конференции или газетных статей, семантика графики и т.п.

Системы машинного перевода

Машинный перевод - это процесс перевода текстов (письменных, а в идеале и устных) с одного естественного языка на другой с помощью специальной компьютерной программы. Так же называется направление научных исследований в ИИ, связанных с построением подобных систем.

Мысль использовать ЭВМ для перевода была высказана в 1947 году, сразу после появления первых ЭВМ. Первая публичная демонстрация машинного перевода состоялась в 1954 году. Несмотря на примитивность той системы (словарь в 250 слов, грамматика из 6 правил, перевод нескольких простых фраз), этот эксперимент получил широкий резонанс: начались исследования по всему миру.

Автоматическая классификация документов

Классификация позволяет сузить поиск, увеличить его скорость и точность в системах документооборота, каталогах Интернет, каналах вещания, службах электронной почты, электронных библиотеках, информагентствах, Интернет-порталах и др.

Цель классификации – автоматическое распределение поступающих в систему документов в зависимости от их типа и содержания по рубрикам (классам) по заданным критериям (библиотечные системы).

Наиболее эффективный метод группировка и поиск ближайшего соседа путем вычисления «расстояния» между парами документов и объединения ближайших соседей в кластеры.

Наиболее известными технологиями являются фильтрация в Microsoft Outlook, рубрицирование в продукте Inxight Categorizer и др.

Программные продукты, реализующие технологии обработки текстов на ЕЯ

• Смысловой анализатор текста Text Analyst;

• ИПС ERW;

• Пакет NeurOK Semantic Suite.

Анализатор Text Analyst – российский продукт, выполняющий «Смысловой портрет» документа и автоматическое реферирование текста. Формируемый реферат требует ручного контекстного «сглаживания».

Информационно-поисковая система ERW определяет количественно «семантическое расстояние» между понятиями, находит документы, в которых идея запроса выражена по-другому. Характеристики ERW: логарифмический рост времени поиска при увеличении объема информации; учет семантики, возможности нечеткого поиска, открытая архитектура.

Пакет *NeurOK Semantic Suite Analyst* реализует автоматическую рубрикацию документов; реферирование и аннотирование; мониторинг web-сайтов, персонализацию информации. Используется метод машинного обучения семантики ЕЯ, запатентованный компанией «НейрОК Интелсофт».

Интеллектуальный агент

По классификации ACM (*Association for Computing Machinery*) многоагентные системы (МАС) являются разделом распределенного искусственного интеллекта (Distributed Artificial Intelligence) и представляют собой системы с взаимодействующими интеллектуальными агентами.

Термин «интеллектуальный агент» (ИА) имеет два значения:

• в ИИ под интеллектуальным агентом понимается разумная сущность, наблюдающая за окружающей средой и действующая в ней, способная воспринимать среду посредством рецепторов и взаимодействовать с ней.

• в информатике и программной инженерии ИА - это программа, самостоятельно выполняющая некоторое задание. Например, задание по постоянному поиску и сбору необходимой информации в Интернете (компьютерные вирусы, боты, поисковые роботы).

Агенты в искусственном интеллекте

Различаются физические и временные агенты.

Физический агент воспринимает окружающий мир через сенсоры и действует с помощью манипуляторов.

Временной агент использует изменяющуюся с ходом времени информацию, предлагает действия или предоставляет данные компьютерной программе или человеку, а также может получать информацию через программный ввод.

Программа-агент может быть описана как агентская функция:

$$f: P \text{ (результат восприятия)} \rightarrow A \text{ (действия)}.$$

Принято различать агентов с простым поведением, агентов с поведением, основанным на моделях, целенаправленных, практичных и обучающихся агентов:

Агенты с простым поведением действуют только на основе текущих знаний.

Их агентская функция основана на схеме продукционных правил типа «условие-действие»:

IF (условие) THEN действие

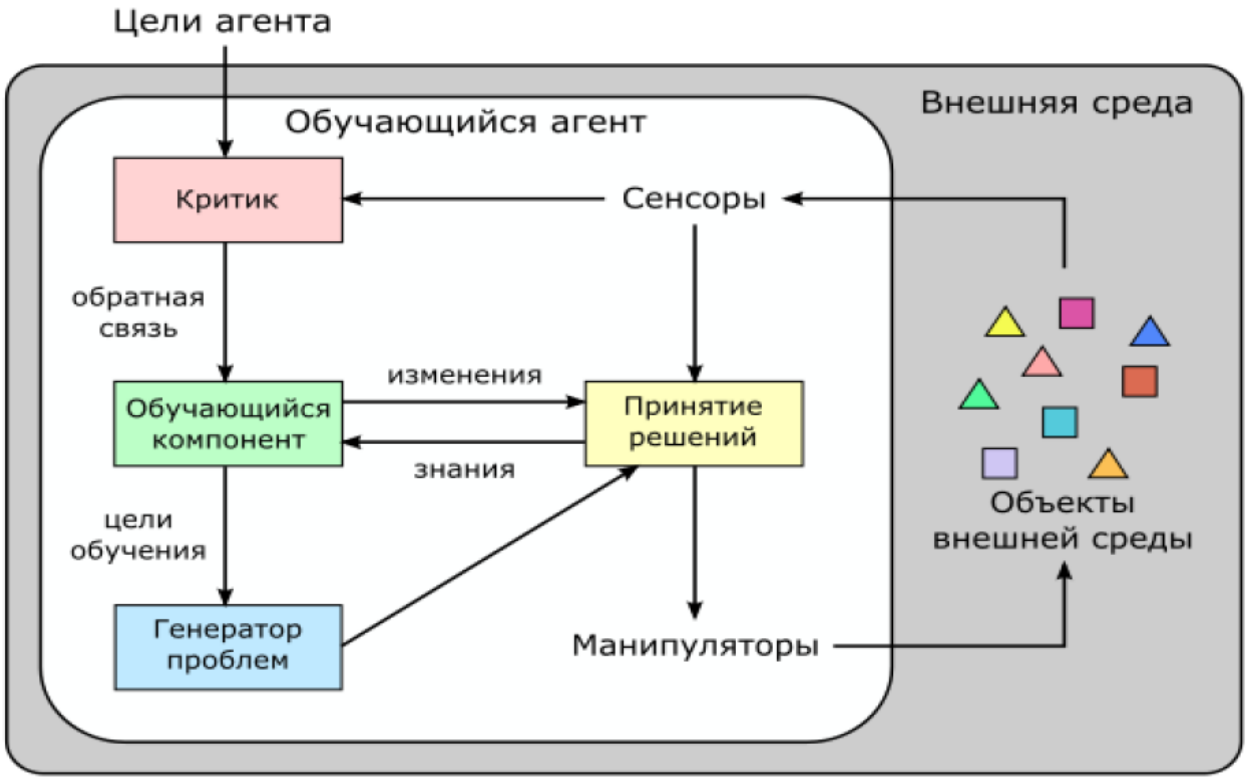
Агенты с поведением, основанным на модели, могут взаимодействовать со средой, лишь частично поддающейся наблюдению.

Целенаправленные агенты хранят информацию о тех ситуациях, которые для них желательны. Это дает агенту способ выбрать среди многих путей тот, что приведет к нужной цели.

Практичные агенты различают, когда цель достигнута, и когда не достигнута, а также насколько желательно для них текущее состояние с помощью «функции полезности».

Обучающиеся агенты (автономные интеллектуальные агенты) должны обладать некоторой независимостью, способность к обучению и приспособлению.

Архитектура ИА имеет вид:



Агенты в информатике и программной инженерии

Роботы по закупкам, просматривают сетевые ресурсы, собирают информацию о товарах и услугах. (*Amazon.com).*

Пользовательские или *персональные агенты* − это агенты, которые действуют в ваших интересах, от вашего имени. Например, проверяют почту, сортируют её по важности, оповещают о поступлении важных писем; играют в компьютерной игре как ваш оппонент; собирают новости; и т.п.

Управляющие и *наблюдающие агенты*, например, ведут наблюдение за компьютерными сетями и следят за конфигурацией каждого компьютера, подключенного к сети, или управляют ботами компьютерных игр (Quake, Robot soccer, Hoshimi).

Агенты, добывающие информацию, действуют в хранилищах данных, собирая информацию, и предупреждая Вас о наличии новой информации.

Характеристики агентов в многоагентной системе

Агенты в МАС должны обладать:

•• *Автономностью* – агенты работают без непосредственного вмешательства со стороны

•• *Интерактивностью* – взаимодействуют с другими агентами

•• *Реактивностью* – воспринимают окружающую среду и взаимодействуют с ней

•• *Проактивностью* – сами являются источником возмущения для окружающей среды, проявляя целеустремленное поведение

•• *Целеустремленностью* – агенты способны выполнять высокоуровневые задачи и проявлять интеллектуальное поведение при достижении цели.

Современные направления изучения МАС:

- знания, желания и намерения;
- координация
- самоорганизация;
- мультиагентное обучение;
- надежность и устойчивость к сбоям;

Общие принципы координации в МАС

Координация предназначена для согласования индивидуальных целей и вариантов поведения агентов, при которых каждый агент улучшает или не ухудшает значение своей функции полезности, а система в целом улучшает качество решения общей задачи.

Методы решения задачи координации базируются на результатах теории управления, *исследования операций, теории игр и планирования*.

Самоорганизация и эмерджентность

Самоорганизация – это динамические и адаптивные процессы, ведущие к поддержанию системы без внешнего управления.

В качестве прообразов часто используются примеры из области эволюционных вычислений (роевые алгоритмы). Например:

1. Взаимодействия между индивидуумами через среду;

2. Запаздывающая положительная обратная связь (например, увеличение количества феромона, оставляемого муравьями при обнаружении источника пищи);

3. Запаздывающая отрицательная обратная связь (испарение феромона по времени).

4. Изменение поведения с увеличением количества феромона на пути к источнику пищи).

Основными чертами самоорганизации являются:

1. *Автономность* – взаимодействие в внешним миром допустимо, но недопустимо управление из внешнего мира;

2. *Адаптивность и робастность* по отношению к изменениям – способность реагировать подходящим образом на изменения среды;

3. *Возрастание порядка* – в соответствии с возрастанием организации системы (уменьшение числа состояний системы, появление пространственной, временной и функциональной структур; появление состояний системы на метауровне - аттракторов);

4. *Динамика* – самоорганизация есть процесс, но не какое-либо конечное состояние.

Говорят, что система проявляет *эмерджентность*, если на макроуровне в системе возникают новые свойства, паттерны, и т.д., и эти процессы являются следствием взаимодействий на микроуровне.

Сложные взаимодействия агентов. Аукционы.

Аукцион (лат. «повышаю») − метод проведения торговли каким-либо товаром, ценными бумагами и т.д.

Для каждого участника аукциона ценности делятся на три вида: личная, общая и коррелированная ценность.

Существуют 4 основных вида аукционов:

•• прямой (английский),

•• голландский (оптовый),

•• янки (своей цены),

•• обратный.

Английский аукцион - самый распространенный вид. Он проводится с гласными торгами и поднятием цены, начинается с минимальной цены. Товар достается покупателю, давшему максимальную цену. Не всегда торги заканчиваются продажей.

На *обратном аукционе* покупатели выставляют запросы на требуемые товары, а продавцы соревнуются, предлагая лучшую цену и условия.

Главная особенность *аукционов янки* − закрытые от других участников торги. Каждый участник подает свою цену в конверте, продавец выбирает наибольшую, а покупатель покупает товар по той цене, которую он назвал.

Данный вид аукционов не имеет доминирующих стратегий, считается, что способствует шпионажу.

Название *голландского* аукциона происходит от рынка цветов в Голландии. Покупатели могут претендовать на покупку многих единиц товара. Все выигравшие покупатели платят только минимальную из выигравших цен.

Оригинальным видом аукционов считается также *аукцион Викри*. В нем побеждает вторая по величине цена.

Биржи

Биржи являются естественным выбором в случае, когда много продавцов и много покупателей. В отличие от аукционов здесь нет понятия аукционист. Все участники могут быть как продавцами, так и покупателями.

Аукционы – это частный случай биржи.

Администратор биржи определяет набор предметов $M = \{1, 2, \ldots, m\}$, которые доступны для торговли.

Ставка (*bid*) – это совокупность вида

$Bj=((\lambda_{1j},\lambda_{2j},\ldots\lambda_{mj}),p_j)$ где λ_{ij} – количество предметов вещи *i*, цена p_i.

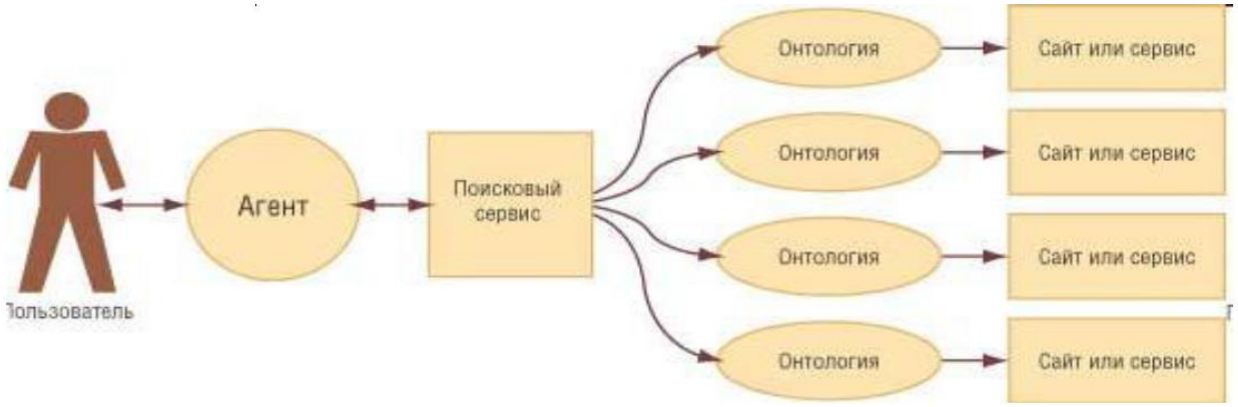
Если $p_j>0$, то ставка – это запрос на продажу, иначе – на покупку.

Должен быть максимизирован остаток при условии, что спрос не превышает предложение. Проблема состоит в разбиении всех биржевых ставок на выигравшие и проигравшие.

Онтология (от греч. *ontos* - сущее) – это описание на формальном языке понятий некоторой предметной области и семантических отношений между ними. Это знания о знаниях, т.е. метазнания.

В Интернете онтологии скоро станут обычным явлением. Сейчас в сети онтологии варьируются от больших таксономий по категориям веб-сайтов (Yahoo), до каталогов продаваемых товаров и их характеристик (Amazon.com).

В области медицины создан стандартный структурированный словарь *SNOMED* и семантическая сеть Системы Унифицированного Медицинского Языка (UMLS). Появляются общецелевые онтологии. Например, Программа ООН по развитию онтологии *UNSPSC*, которая предоставляет терминологию товаров и услуг.



Формальное определение онтологии - тройка множеств вида:

$$O = \langle X, R, F \rangle,$$

где ***X*** - множество понятий (концептов, терминов); ***R*** - множество отношений между понятиями; ***F*** - множество интерпретаций понятий.

Онтология представляет собой явную спецификацию предметной области. В неё входят:

-множество понятий;

- отношения между понятиями;
- атрибуты и свойства понятий;
- ограничения на свойства и атрибуты;
- экземпляры;
- знания из предметной области

Частными случаями являются:

- Простой словарь ($X \neq \varnothing$, $R = \varnothing$, $F \neq \varnothing$);
- Таксономия ($X \neq \varnothing$, $R = \{\text{is-a}\}$, $F \neq \varnothing$);
- Тезаурус ($X \neq \varnothing$, $R = \{\text{eqiv-to}\}$, $F \neq \varnothing$);
- Каталог-справочник с множеством понятий и ссылками на них.

Основные *свойства онтологий*:

- существенность (охват ПО),
- непротиворечивость,
- независимость от реализации,
- декларативность,
- расширяемость,
- ясность.

Пример. Пусть, например, речь идёт об онтологии издательства, выпускающего книги и журналы. Основные понятия этой предметной области: издательство, книга и журнал. Это классы онтологии:

Класс *«Издательство»* имеет следующие атрибуты:

- название (строка);
- город (строка).

(в скобках обычно указываются типы значений и разрешенные значения).

Класс *«Книга»* имеет следующие атрибуты:

• название (строка);

• автор (строка);

• ISBN (строка специального формата);

• число страниц (натуральное число);

• тип обложки (строка; возможные значения: мягкая, твердая, суперобложка);

• издательство (экземпляр класса «издатель»);

• год издания (натуральное число — четыре цифры);

• описание (текст);

• цена (число с плавающей точкой — два знака после запятой).

Класс *«Журнал»:*

• название;

• ISSN;

• число страниц;

• издательство;

• год выпуска;

• номер;

• описание;

• цена.

Классы *«Книга»* и *«Журнал»* имеют много общих атрибутов, вынесем их в отдельный класс *«Печатная продукция»*.

Фрагмент онтологии:

Процесс разработки онтологии

В общем виде процесс разработки онтологии включает следующие этапы:

• составляется глоссарий терминов (понятий);

• на ЕЯ создается список точных определений терминов;

• на основе таксономических отношений строятся деревья классификации понятий (иерархии классов);

• из понятий, не задействованных при составлении деревьев классификации, выделяются атрибуты классов и их возможные значения. Эти понятия устанавливают связи между классами;

• в зависимости от целей, для которых разрабатывается онтология, в нее могут добавляться экземпляры классов;

• эксперты по той предметной области, в которой разрабатывается онтология, создают правила логических выводов, позволяющие оперировать данными, представленными в онтологии, и извлекать из созданной онтологии новые знания.

Классификация онтологий по содержимому

Существуют различные варианты классификации онтологических систем. Наиболее удачной представляется классификация онтологий по их содержимому:

В общем случае онтологическая система представляет собой совокупность трех множеств:

$$\Sigma = <O_{meta}, O_{ПО}, O_{выв}>$$

где O_{meta} - онтология верхнего уровня (статична, не привязана к какой-либо предметной области), $O_{ПО}$ - предметная онтология, $O_{выв}$ - онтология, связанная с машиной вывода.

Основными вопросами инжиниринга онтологических систем является их создание, вопросы, на которые она будет способна отвечать, а также то, какую часть знаний она будет покрывать.

Основными онтологическими подходами к выделению понятий являются:

• подход сверху вниз, когда сначала определяются наиболее общие концепты, затем они специализируются,

• подход снизу вверх, когда сначала определяются конкретные концепты, затем они объединяются в классы,

• смешанный подход, когда сначала выделяются наиболее очевидные понятия.

7. Системы и методы распознавания образов.

Проблема обучения распознаванию образов интересна как с *прикладной*, так и с *принципиальной* точки зрения.

С ***прикладной*** точки зрения решение этой проблемы важно прежде всего потому, что оно открывает возможность автоматизировать многие процессы, которые до сих пор связывали лишь с деятельностью живого мозга.

Принципиальное значение проблемы тесно связано с вопросом: что может и что принципиально не может делать машина?

Классификация на основе Решающих Функций

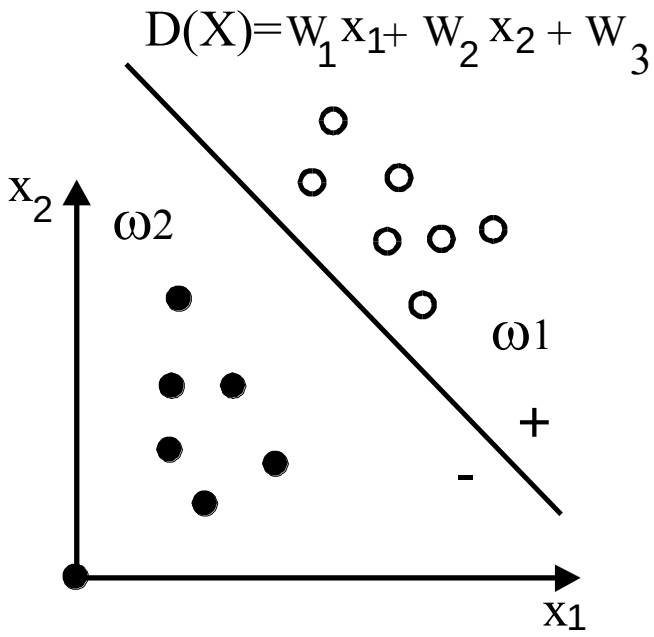


Рис.7.1

$d(X)=w_1x_1+w_2x_2+w_3=0$ - уравнение разделяющей прямой, где w_i - параметры, а x_1 и x_2 - переменные.

Образ X принадлежит классу ω_1, если $d(X)>0$, и классу ω_2, если $d(X)<0$.

Если образ лежит на разделяющей границе, то имеет место случай, соответствующий неопределенности $d(X)=0$.

Общий вид линейной решающей функции задается формулой:

$$d(X)=w_1x_1+w_2x_2+...+ w_nx_n+w_{n+1}=W'_0X+ w_{n+1} , \qquad (1)$$

где вектор $W_0 = (w_1, w_2,..., w_n)'$ называется весовым или параметрическим.

Выражение (1) перепишем в виде:

$$d(X)= W'X, \qquad (2)$$

где $X=(x_1,x_2,...,x_n,1)'$ и $W=(w_1,w_2,...,w_n,w_{n+1})'$ - пополненные векторы образов и весов соответственно.

Предполагается, что в случае разбиения на два класса решающая функция $d(X)$ обладает свойством:

$$d(X) = W'X \begin{cases} > 0, \textbf{if } \boldsymbol{X} \in \omega_1, \\ < \boldsymbol{0}, \textbf{if } \boldsymbol{X} \in \omega_2. \end{cases} \qquad (3)$$

Случаи разбиения на несколько классов $\omega_1, \omega_2, ..., \omega_M$:

Случай 1. Каждый класс отделяется от всех остальных одной разделяющей поверхностью.

В этом случае существует *M* решающих функций ($\boldsymbol{d_i}$), обладающих свойством:

$$d_i(X) = W_i^{\prime} X \begin{cases} > 0, \text{if } X \in \omega_i, \\ < 0, \text{if } X \notin \omega_i., \end{cases} \quad i=1, 2, \ldots, M, \qquad (4)$$

где $W_i=(w_{i1},w_{i2},...,w_{in},w_{i,n+1})'$ - весовой вектор, соответствующий *i*-й решающей функции.

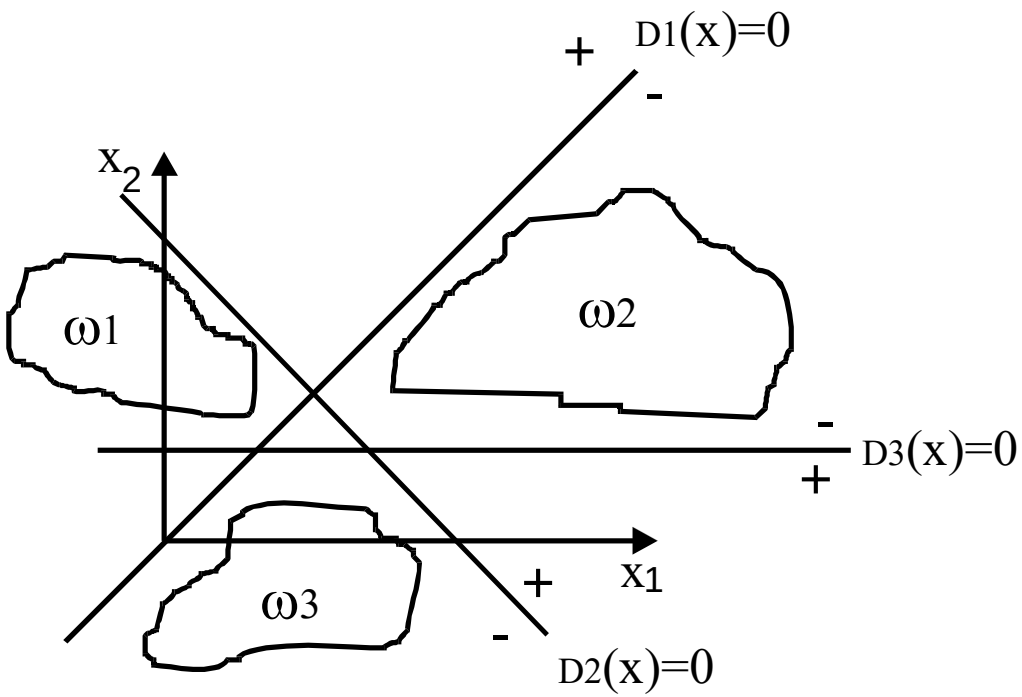


Рис.7.2а

Если некоторый образ X принадлежит классу ω_1, на основании геометрических соображений заключаем, что $d_1(X)>0$, а $d_2(X)<0$ и $d_3(X)<0$.

Граница, отделяющая класс ω_1 от остальных, определяется значениями X, при которых $d_1(X)=0$.

Пусть решающие функции, соответствующие Рис.2а, имеют вид:

$$d_1(X)= -x_1+x_2,\ \ d_2(X)=x_1+x_2-5,\ \ \text{и}\ d_3(X)=-x_2+1.$$

Следовательно, три разделяющие границы определяются уравнениями:

$$-x_1+x_2=0,\ \ \ x_1+x_2-5=0,\ \ \ \ -x_2+1=0.$$

Любой образ, для которого выполняются условия $d_1(X)>0$, а $d_2(X)<0$ и $d_3(X)<0$ автоматически зачисляется в класс ω_1. (Рис.2б).

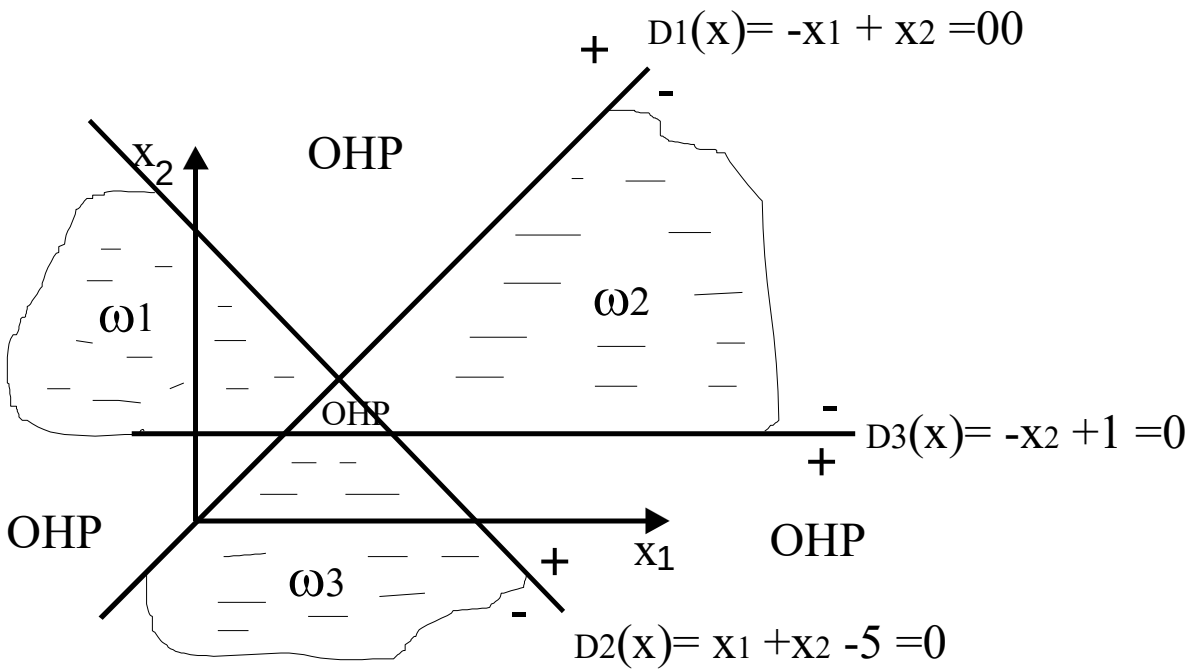


Рис.7.2б

Пример.

Пусть надо классифицировать образ $X=(6,5)'$.

Подставляем его параметры в наши три решающие функции, получаем:

$$d_1(X)=-1<0, \quad d_2(X)=6>0, \quad d_3(X)=-4<0.$$

Вывод: образ X зачисляется в класс ω_2.

Случай 2.

Каждый класс отделяется от любого другого взятого в отдельности класса «индивидуальной» разделяющей поверхностью, т.е. классы попарно разделимы.

В этом случае существует $C_M^2 = M \times (M-1)/2$ разделяющих поверхностей.

Решающие функции имеют вид: $d_{ij}(X) = W_{ij}'X$ и обладают тем свойством, что если образ X принадлежит классу ω_i, то $d_{ij}(X) > 0$ для всех $j \neq i$, и кроме того, $d_{ij}(X) = -d_{ji}(X)$.

Пример 2. На Рис. 3а представлены три класса образов, разделимых согласно случаю 2.

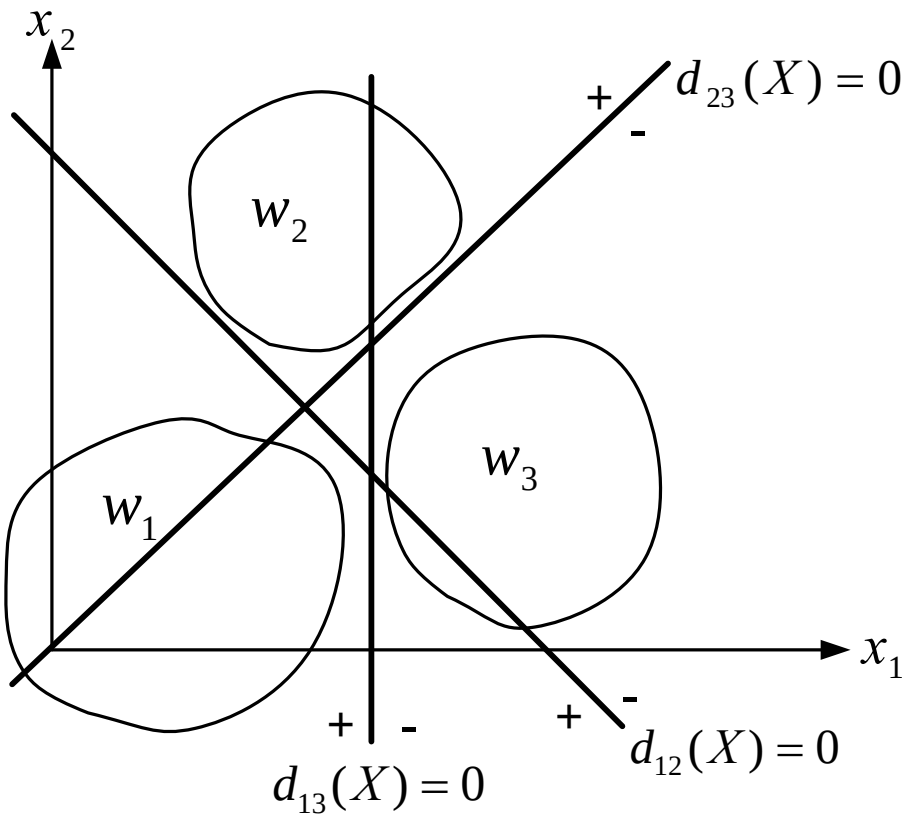


Рис.7.3а

Пусть решающие функции имеют вид:

$$d_{12}(X)= -x_1-x_2+5, \quad d_{13}(X)= -x_1+3, \quad d_{23}(X)= -x_1+x_2.$$

Разделяющие границы получим, приравнивая решающие функции к нулю.

Области решений, однако, теперь могут содержать несколько зон, где соответствующие функции положительны.

В частности, область, отвечающая классу ω_1, определяется значениями X, при которых $d_{12}(X)>0$ и $d_{13}(X)>0$.

Значение решающей функции $d_{23}(X)$ в этой области **не существенно**, поскольку эта функция никак не связана с классом ω_1.

Области, определяемые тремя указанными решающими функциями, представлены на рис.3б, причем $d_{ij}(X) = -d_{ji}(X)$.

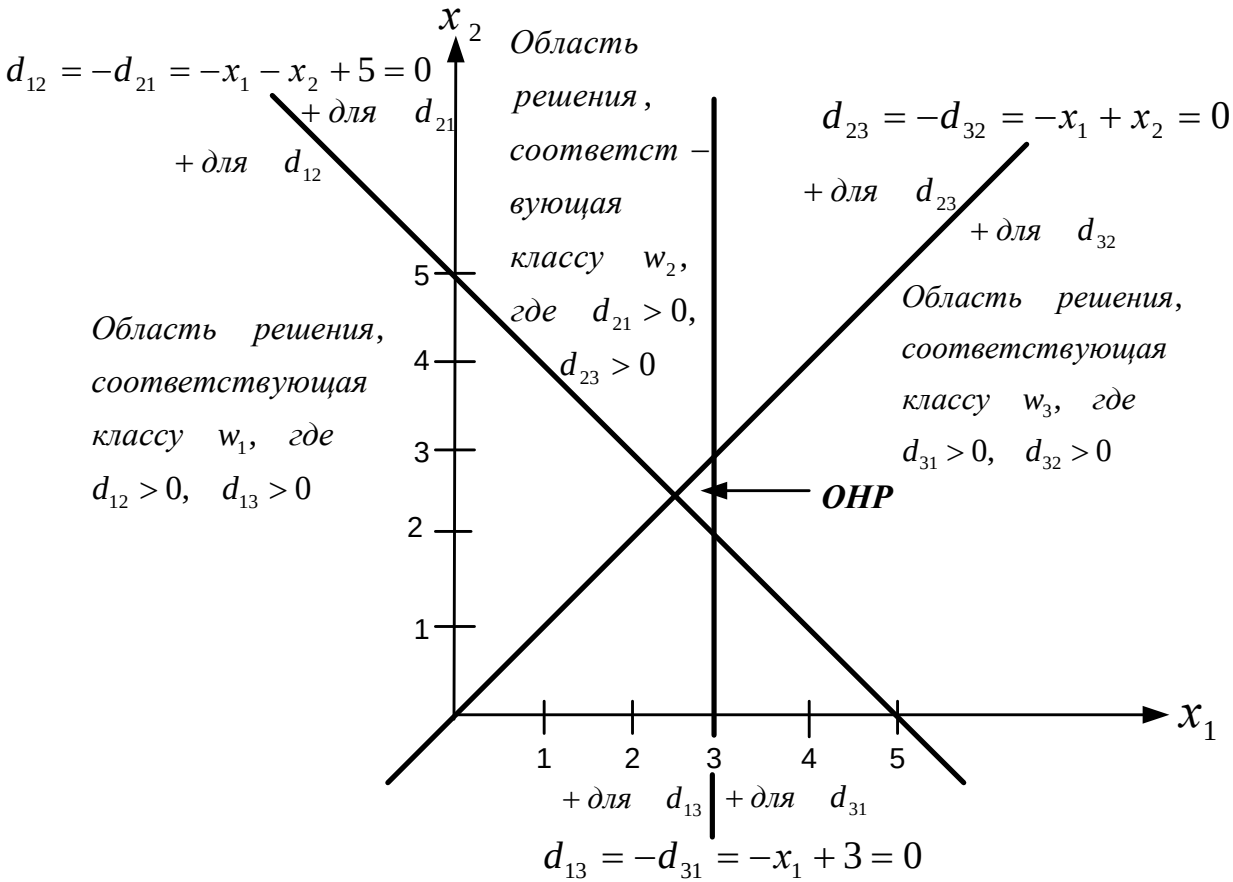


Рис.7.3б

Так, если $d_{12}(X)= -x_1-x_2+5$, то $d_{21}(X)= +x_1+x_2-5$ и зона положительности функции $d_{12}(X)$ совпадает с зоной отрицательности функции $d_{21}(X)$.

Пусть нам надо классифицировать образ $X=(4,3)'$.

Подставляем его параметры в решающие функции, получаем:

$d_{12}(X)=-2$, $d_{13}(X)=-1$, $d_{23}(X)=-1$.

Отсюда следует, что $d_{21}(X)=+2$, $d_{31}(X)=+1$, $d_{32}(X)=+1$.

Так как $d_{3j}(X)>0$ для всех $j=1,2$ и в область неопределенности мы не попали, то образ X зачисляется в класс ω_3.

Случай 3.

Существует M решающих функций: $d_k(X)= W_k'X$, $k=1,2,...,M$, таких, что если образ X принадлежит классу ω_i, то

$$d_i(X)> d_j(X) \text{ для всех } j\neq i. \qquad (6)$$

Эта ситуация является разновидностью случая 2, поскольку можно положить

$$d_{ij}(X)=d_i(X)-d_j(X) = (w_i-w_j)'X = w'_{ij}X, \qquad (7)$$

где $w_{ij}=w_i-w_j$.

Можно показать, что если $d_i(X)>d_j(X)$ для всех $j\neq i$, то $d_{ij}(X)>0$ для всех $j\neq i$, т.е. если классы разделимы, как в случае 3, то они автоматически разделимы и как в случае 2.

Пример 3. На Рис.3а представлены **три класса** образов, разделимых согласно **случаю 3**:

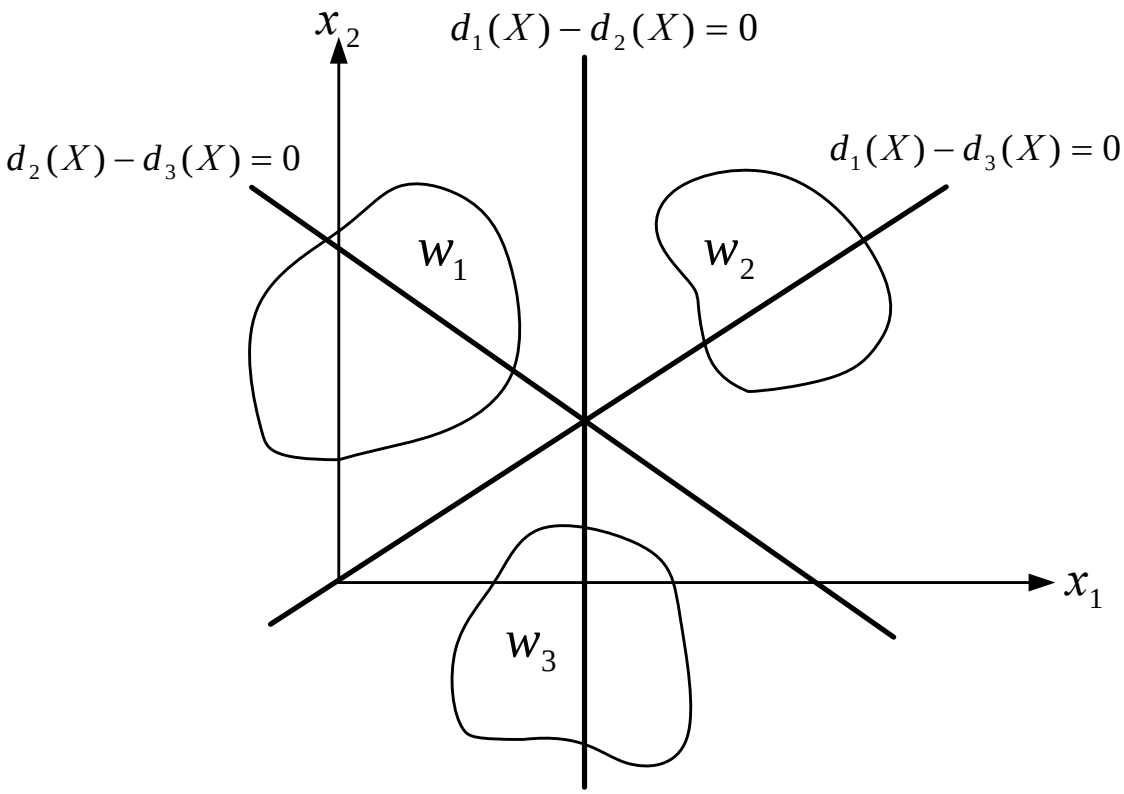


Рис.7.3а

Для образов, принадлежащих классу ω_1, должны выполняться условия

$$d_1(X)>d_2(X) \text{ и } d_1(X)>d_3(X).$$

Это эквивалентно требованию того, чтобы входящие в данный класс образы располагались в положительных зонах поверхностей $d_1(X)-d_2(X)=0$ и $d_1(X)-d_3(X)=0$.

Пусть в качестве решающих функций выбраны прямые:

$d_1(X) = -x_1+x_2$, $d_2(X)=x_1+x_2-1$, $d_3(X)= - x_2$.

Разделяющие границы для трех классов выглядят как:

$d_1(X)-d_2(X)=-2x_1+1=0$,

$d_1(X)-d_3(X)=-x_1+2x_2=0$,

$d_2(X)-d_3(X)=x_1+2x_2-1=0$.

Для того чтобы определить область решений, соответствующую классу ω_1, необходимо выделить область, в которой выполняются $d_1(X)>d_2(X)$ и $d_1(X)>d_3(X)$.

Эта область совпадает с положительными зонами для прямых $-2x_1+1=0$ и $-x_1+2x_2=0$. (Рис.3б):

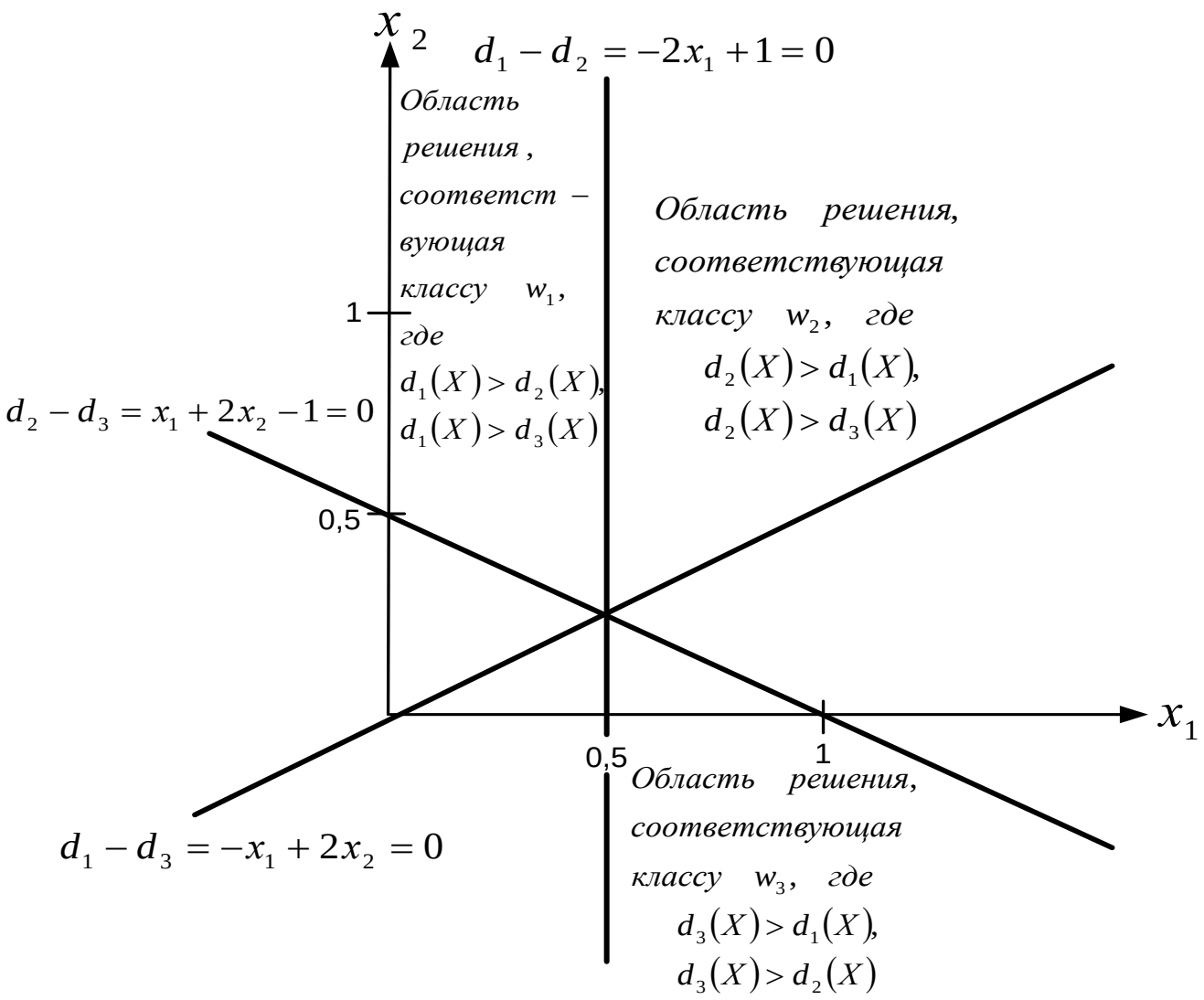

x_2
$d_1 - d_2 = -2x_1 + 1 = 0$
Область решения, соответст – вующая классу w_1, где $d_1(X) > d_2(X)$, $d_1(X) > d_3(X)$
Область решения, соответствующая классу w_2, где $d_2(X) > d_1(X)$, $d_2(X) > d_3(X)$
1
$d_2 - d_3 = x_1 + 2x_2 - 1 = 0$
0,5
x_1
0,5
1
Область решения, соответствующая классу w_3, где $d_3(X) > d_1(X)$, $d_3(X) > d_2(X)$
$d_1 - d_3 = -x_1 + 2x_2 = 0$

Рис.7.3б

Область принятия решения о принадлежности образа классу ω_2 совпадает с положительными зонами для прямых $2x_1-1=0$ и $x_1+2x_2-1=0$.

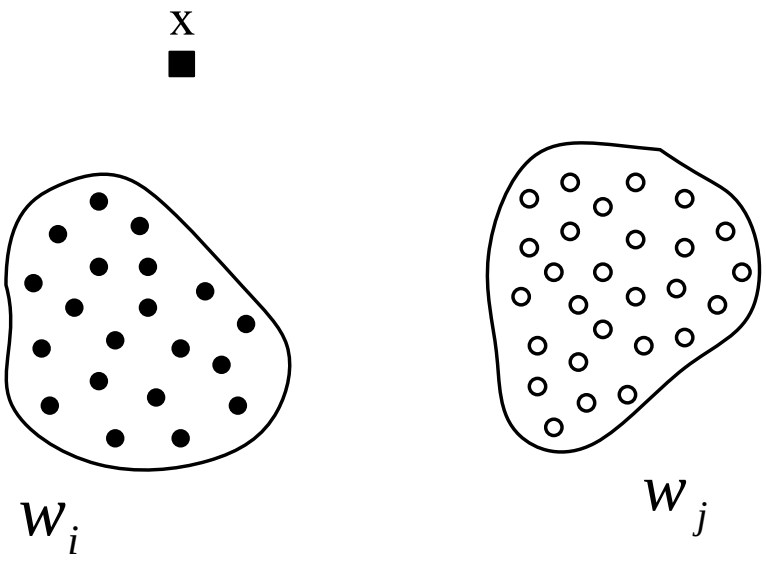
Область, отвечающая классу ω_3 совпадает с положительными зонами для прямых $x_1-2x_2=0$ и $-x_1-2x_2+1=0$.

Пусть нам надо классифицировать образ $X=(1,1)'$.

Подстановка компонент этого вектора в выбранные решающие функции дает: $d_1(X)=0$, $d_2(X)=1$, $d_3(X)= -1$.

Вывод: образ относится к классу ω_2.

Классификация образов с помощью функций расстояния



1.Случай единственности эталона.

Рассмотрим *M* классов. Пусть эти классы допускают представление с помощью эталонных образов $\mathbf{z_1, z_2, \ldots, z_M}$.

Евклидово расстояние между произвольным вектором образа *X* и *i*–м эталоном определяется выражением:

$$D_i = \|X - z_i\| = \sqrt{(X - z_i)^2} = \sqrt{(X - z_i)' \times (X - z_i)}\,. \qquad (1)$$

Классификатор, построенный по принципу минимума расстояния, вычисляет расстояние, отделяющее неклассифицированный образ X от эталона каждого класса, и зачисляет этот образ в класс, оказавшийся ближайшим к нему. Другими словами, образ X приписывается к классу ω_i, если условие $D_i<D_j$ выполняется для всех $j \neq i$. Случаи равенства расстояний разрешаются произвольным образом.

Формуле (1) можно придать более удобный вид.

Возведение D_i в квадрат дает:

$$D^2{}_i = \left\| X - z_i \right\|^2 = (X - z_i)' \times (X - z_i) =$$

$$X'X - 2X'z_i + z_i'z_i = X'X - 2(X'z_i - \frac{1}{2} z_i'z_i). \cdot \qquad (2)$$

Выбор минимального значения D_i^2 эквивалентен выбору минимального D_i.

Формула (2) показывает, что выбор минимального значения D_i^2 эквивалентен выбору максимального значения $(X'z_i - \frac{1}{2} z_i' z_i)$, поскольку при вычислении любых D_i^2, член $X'X$ не зависит от значения i.

Следовательно,

решающие функции можно определить как:

$$\mathbf{d_i(X) = X'z_i - (1/2)z_i'z_i, \quad i=1,2,\ldots,M,} \qquad (3)$$

где образ X относится к классу ω_i, если условие $d_i(X) > d_j(X)$ справедливо для всех $j \neq i$.

Примечание:

$d_i(X)$ - линейная решающая функция,

т.к, если z_{ij}, j=1,2,…,n, - компоненты вектора Z_i , то положим:

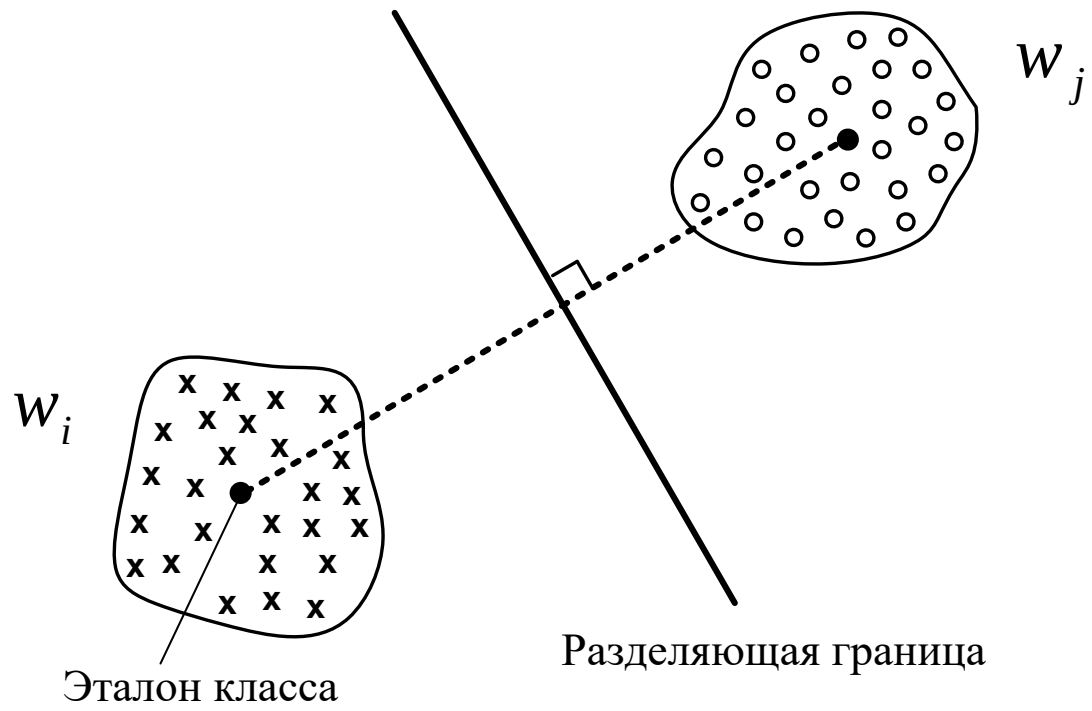
$$w_{ij}=z_{ij},\ \ j=1,2,\ldots,n,$$

$$w_{i,n+1}=-(1/2)z'_i z_i \qquad (4)$$

и вектор $X=(x_1,x_2,..,x_n,1)'$, тогда (3) можно представить в обычной линейной форме:

$$d_i(X) = w_i'X,\ \text{i=1,2,…,M,} \qquad (5)$$

где $w_i=(w_{i1},w_{i2},\ldots,w_{i,n+1})'$.


w_j
w_i
Разделяющая граница
Эталон класса

2.Множественность эталонов.

Любой образ, принадлежащий классу ω_i, проявляет тенденцию к группировке вокруг одного из эталонов z_i^1, z_i^2,…, z_i^{Ni}, где N_i– количество эталонных образов, определяющий i–й класс.

Расстояние между произвольным образом X и классом ω_i :

$$D_i = \min_l \| X - z_i^l \|, \quad l = 1,2,...,N_i . \tag{6}$$

D_i - наименьшее из расстояний от образа X до каждого эталона класса ω_i.

Вычисляются значения расстояний D_i, $i=1,2,\ldots,M$.

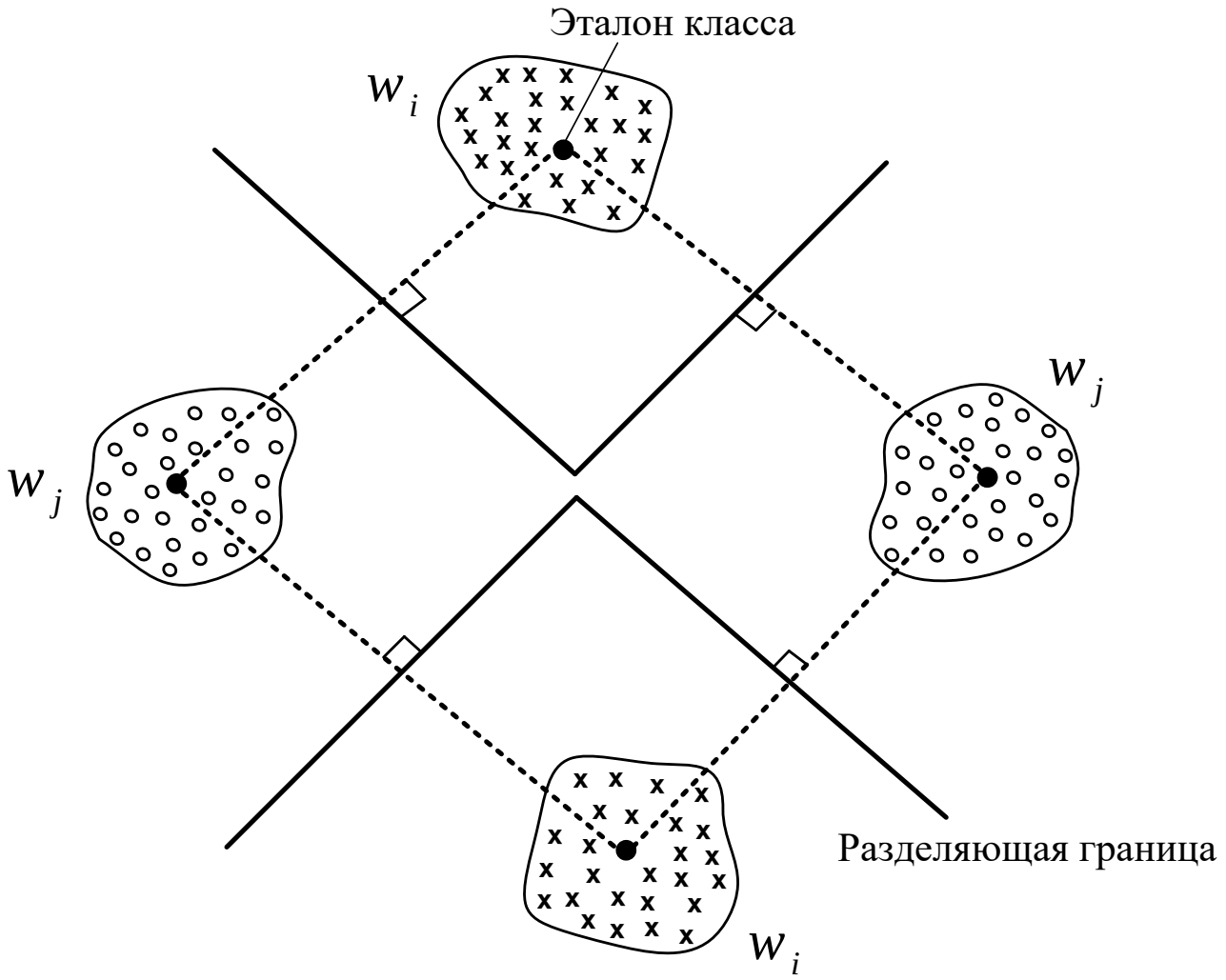
Классифицируемый образ зачисляется в класс ω_i, если $D_i < D_j$ справедливо для всех $j \neq i$.

В случае равенства расстояний решение принимается произвольным образом.

Учитывая выражение (3), получаем:

$$d_i(X) = \max_l \{(X' z_i^l) - (1/2)(z_i^l)' z_i^l\}, \quad l = 1,2,\ldots,N_i. \tag{7}$$

Образ X зачисляется в класс ω_i, если $d_i > d_j$ справедливо для всех $j \neq i$.


Эталон класса
w_i
w_j
w_j
Разделяющая граница
w_i

Обобщение принципов классификации по минимуму расстояния.

Рассмотрим выборку образов с известной классификацией $\{s_1, s_2, \ldots, s_N\}$.

Предполагается, что каждый образ выборки входит в один из классов $w_1, w_2, \ldots, w_M$.

Можно определить правило классификации, основанное на принципе

ближайшего соседа (БС - правило).

правило относит классифицируемый образ к классу, к которому принадлежит его ближайший сосед, причем образ s_i называется ближайшим соседом образа X, если

$$D(s_i, X) = \min_l \{D(s_l, X)\}, \; l = 1,2,...,M \,, \tag{10}$$

где D - любое (в частности евклидово) расстояние, определение которого допустимо на пространстве образов.

1-БС – правило;

q-БС – правило.

Выявление кластеров

Меры сходства:

Евклидово расстояние между образами X и z: $D=||X-z||$.

Если для каждого признака j можно установить его вес h_j, $j \in \{1,2,...,n\}$, то используется ***евклидово взвешенное*** расстояние: $D_{взв} = \sqrt{\sum h_j (x_j - z_j)^2}$.

Мера сходства ***Такимото***:

$$D_T = \frac{\sum x_j \times z_j}{\sum (x_j)^2 + \sum (z_j)^2 - \sum x_j z_j} .$$

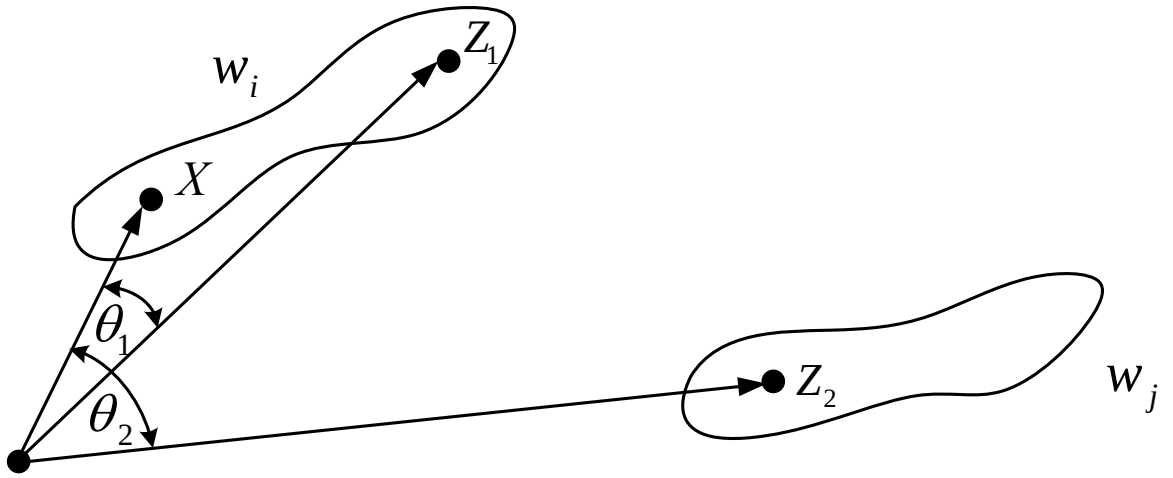
Коэффициент ***Дейка***:

$$D_D = \frac{2\sum x_j \times z_j}{\sum (x_j) + \sum (z_j)} .$$

Коэффициент ***Жаккарта*:**

$$D_Ж = \frac{\sum x_j \times z_j}{\sum (x_j) + \sum (z_j) - \sum (x_j z_j)} .$$

Непараметрическая функция сходства: $$s(X,z)=\frac{X'\times z}{\|X\|\times\|z\|}.$$ (1)



Начало координат

$$S(X,Z_1)=\cos\theta_1=\frac{X'Z_1}{\|X\|\|Z_1\|}$$

$$S(X,Z_2)=\cos\theta_2=\frac{X'Z_2}{\|X\|\|Z_2\|}$$

Перцептроны

Одним из методов решения задач обучения распознаванию образов основан на моделировании гипотетического механизма человеческого мозга. Структура модели заранее постулируется. При таком подходе уровень биологических знаний или гипотез о биологических механизмах является исходной предпосылкой, на которой базируются модели этих механизмов. Примером такого направления в теории и практике проблемы ОРО является класс устройств, называемых перцептронами. Нужно отметить, что перцептроны на заре своего возникновения рассматривались только как эвристические модели механизма мозга. Впоследствии они стали основополагающей схемой в построении кусочно-линейных моделей, обучающихся распознаванию образов.

В наиболее простом виде перцептрон (Рис.7.17) состоит из совокупности чувствительных (сенсорных) элементов (S-элементов), на которые поступают входные сигналы. S-элементы случайным образом связаны с совокупностью ассоциативных элементов (А-элементов), выход которых отличается от нуля только тогда, когда возбуждено достаточно большое число S-

элементов, воздействующих на один А-элемент. А-элементы соединены с реагирующими элементами (R-элементами) связями, коэффициенты усиления которых переменны и изменяются в процессе обучения. Взвешенные комбинации выходов R-элементов составляют реакцию системы, которая указывает на принадлежность распознаваемого объекта определенному образу. Если распознаются только два образа, то в перцептроне устанавливается только один R-элемент, который обладает двумя реакциями – положительной и отрицательной. Если образов больше двух, то для каждого образа устанавливают свой R-элемент, а выход каждого такого элемента представляет линейную комбинацию выходов А-элементов:

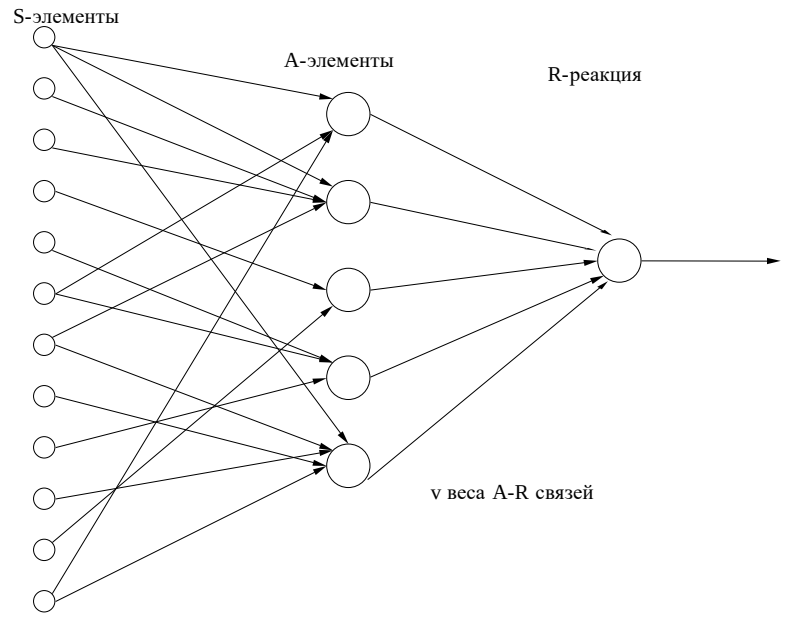


Рис. 7.17.

Реакция всей системы пропорциональна сумме взятых с определенными весами реакций элементов ассоциативной сетчатки. Обозначим через xi реакцию i -го ассоциативного элемента и через wi - соответствующий вес. Тогда реакцию системы можно записать как:

$$R = \sum_{i=1}^{n+1} w_i \times x_i = w'X \tag{7.22}$$

Если $R>0$, то предъявленный системе образ принадлежит классу w1. Если $R<0$, то образ относится к классу w2.

Данная перцептронная модель представляет собой не что иное, как реализацию линейной решающей функции.

Данный подход легко распространить на случай разделения на несколько (M) классов посредством увеличения числа реагирующих элементов в R-сетчатке. Классификация проводится аналогично: рассматриваются значения реакций R1, R2,…,RM и образ причисляется к классу ωi, если $Ri>Rj$ для всех $j\neq i$.

Если в перцептроне допускаются лишь связи, идущие от бинарных S-элементов к A-элементам и от A-элементов к единственному R-элементу, то такой перцептрон принято называть элементарным α-перцептроном.

О перцептронах было сформулировано и доказано несколько основополагающих теорем, две из которых, определяющие основные свойства перцептрона, приведены ниже.

Теорема 1. Класс элементарных α-перцептронов, для которых существует решение для любой задуманной классификации, не является пустым.

Эта теорема утверждает, что для любой классификации обучающей последовательности можно подобрать такой набор (из бесконечного набора) А-элементов, в котором будет осуществлено задуманное разделение обучающей последовательности при помощи линейного решающего правила.

Теорема 2. Если для некоторой классификации решение существует, то в процессе обучения α-перцептрона с коррекцией ошибок, начинающегося с произвольного исходного состояния, это решение будет достигнуто в течение конечного промежутка времени.

Смысл этой теоремы состоит в том, что если относительно задуманной классификации можно найти набор А-элементов, в котором существует решение, то в рамках этого набора оно будет достигнуто в конечный промежуток времени.

Обычно обсуждают свойства бесконечного перцептрона, т. е. перцептрона с бесконечным числом А-элементов со всевозможными связями с S-элементами (полный набор А-элементов). В таких перцептронах решение всегда существует, а раз оно существует, то оно и достижимо в α-перцептронах с коррекцией ошибок.

Очень интересную область исследований представляют собой многослойные перцептроны и перцептроны с перекрестными связями, но теория этих систем практически еще не разработана.

Принцип покрепления – наказания. Обучающий алгоритм для перцептрона, приведенного на рис.2.17, сводится к простой схеме итеративного определения весов *W*. Рассмотрим описание этой схемы, которую называют *алгоритмом перцептрона*.

Заданы два обучающих множества, представляющие классы ω_1 и ω_2 соответственно. Пусть *W(1)* – начальный вектор весов, который выбирается произвольно. В таком случае *k*-й шаг обучения выглядит следующим образом.

Если $X(k) \in \omega_1$ и $W'(k) \times X(k) \leq 0$, то вектор весов *W(k)* заменяется вектором

$$W(k+1) = W(k) + cX(k), \qquad (7.23)$$

где с - корректирующее приращение.

Если $X(k) \in \omega_2$ и $W'(k) \times X(k) \geq 0$, то вектор весов *W(k)* заменяется вектором

$$W(k+1) = W(k) - cX(k), \qquad (7.24)$$

В противном случае не изменяется, т.е.

$$W(k+1) = W(k). \qquad (7.25)$$

Иными словами, алгоритм вносит изменения в вектор весов W в том и только том случае, если образ, предъявленный на k-м шаге обучения был классифицирован неверно. Корректирующее приращение **c** должно быть положительным, и в данном случае предполагается, что оно постоянно.

Алгоритм перцептрона является процедурой типа «подкрепление-наказание», причем, подкреплением за правильную классификацию образа служит отсутствие наказания. Т.е., если образ классифицирован правильно, то система подкрепляется тем, что в вектор весов W не вносится никаких изменений. С другой стороны, если образ классифицируется неправильно и произведение W’(k)×X(k) оказывается меньше нуля, когда оно должно быть больше нуля, система «наказывается» увеличением значения вектора весов W(k) на величину, пропорциональную X(k). Точно также, если произведение W’(k)×X(k) оказывается больше нуля, когда оно должно быть меньше нуля, система наказывается противоположным образом. *Сходимость* алгоритма наступает при правильной классификации *всех образов* с помощью

некоторого вектора весов. Можно *доказать* (*мы этого делать не будем*), что алгоритм перцептрона сходится за *конечное число итераций*, если заданные классы *линейно разделимы*.

Пример. Рассмотрим образы, приведенные на рис.2.18. Следует применить к этим образам алгоритм перцептрона с тем, чтобы с его помощью определить весовой вектор решения.

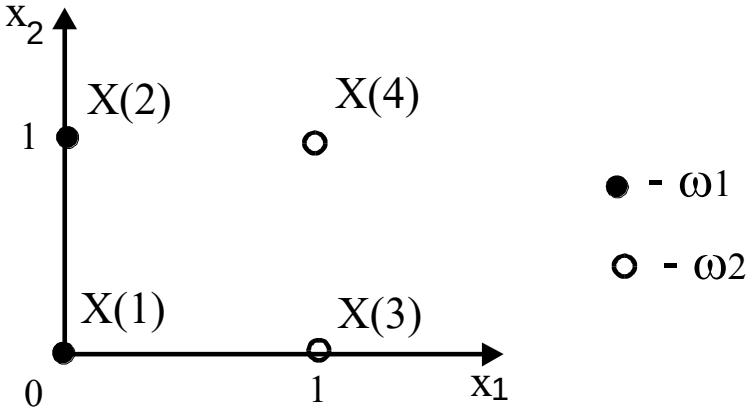


Рис.7.18.

Из рисунка видно, что два заданных класса линейно разделимы и, следовательно, применение алгоритма окажется успешным.

До начала применения алгоритма пополним все образы. При этом рассматриваемые классы обратятся в ω_1: {(0,0,1)', (0,1,1)'} и в ω_2: {(1,0,1)', (1,1,1)'} . Задав c=1 и W(1)=0 и предъявив образы в указанном порядке, получим (по шагам):

$$W'(1)\times X(1) = (0,0,0)\times\begin{pmatrix}0\\0\\1\end{pmatrix} = 0,$$ следовательно: $$W(2) = W(1) + X(1) = \begin{pmatrix}0\\0\\1\end{pmatrix},$$

$$W'(2)\times X(2) = (0,0,1)\times\begin{pmatrix}0\\1\\1\end{pmatrix} = 1,$$ следовательно: $$W(3) = W(2) = \begin{pmatrix}0\\0\\1\end{pmatrix},$$

$$W'(3)\times X(3) = (0,0,1)\times\begin{pmatrix}1\\0\\1\end{pmatrix} = 1,$$ следовательно: $$W(4) = W(3) - X(3) = \begin{pmatrix}-1\\0\\0\end{pmatrix},$$

$$W'(4)\times X(4) = (-1,0,0)\times\begin{pmatrix}1\\1\\1\end{pmatrix} = 1,$$ следовательно: $$W(5) = W(4) = \begin{pmatrix}-1\\0\\0\end{pmatrix}.$$

Коррекция вектора весов проводилась на 1 и 3 шагах. Т.к. полученный результат можно считать искомым решением только в том случае, когда алгоритм осуществит без ошибок полный цикл итерации по всем образам, обучающее множество следует предъявить еще раз.

Процесс обучения системы продолжается при X(5)=X(1), X(6)=X(2), X(7)=X(3) и X(8)=X(4). Второй цикл итерации приводит к следующим результатам:

$$W'(5)\times X(5)=(-1,0,0)\times\begin{pmatrix}0\\0\\1\end{pmatrix}=0,\ \text{следовательно:}\ W(6)=W(5)+X(5)=\begin{pmatrix}-1\\0\\1\end{pmatrix},$$

$$W'(6)\times X(6)=(-1,0,1)\times\begin{pmatrix}0\\1\\1\end{pmatrix}=1,\ \text{следовательно:}\ W(7)=W(6)=\begin{pmatrix}-1\\0\\1\end{pmatrix},$$

$$W'(7)\times X(7)=(-1,0,1)\times\begin{pmatrix}1\\0\\1\end{pmatrix}=0,\ \text{следовательно:}\ W(8)=W(7)-X(7)=\begin{pmatrix}-2\\0\\0\end{pmatrix},$$

$$W'(8)\times X(8)=(-2,0,0)\times\begin{pmatrix}1\\1\\1\end{pmatrix}=-2,\ \text{следовательно:}\ W(9)=W(8)=\begin{pmatrix}-2\\0\\0\end{pmatrix}.$$

Поскольку в данном цикле итерации опять совершено 2 ошибки, все образы предъявляются еще раз для значений X(9)=X(1), X(10)=X(2), X(11)=X(3) и X(12)=X(4):

$$W'(9)\times X(9)=(-2,0,0)\times\begin{pmatrix}0\\0\\1\end{pmatrix}=0,\ \text{следовательно:}\ W(10)=W(9)+X(9)=\begin{pmatrix}-2\\0\\1\end{pmatrix},$$

$$W'(10)\times X(10)=(-2,0,1)\times\begin{pmatrix}0\\1\\1\end{pmatrix}=1,$$ следовательно: $$W(11)=W(10)=\begin{pmatrix}-2\\0\\1\end{pmatrix},$$

$$W'(11)\times X(11)=(-2,0,1)\times\begin{pmatrix}1\\0\\1\end{pmatrix}=-1,$$ следовательно: $$W(12)=W(11)=\begin{pmatrix}-2\\0\\1\end{pmatrix},$$

$$W'(12)\times X(12)=(-2,0,1)\times\begin{pmatrix}1\\1\\1\end{pmatrix}=-1,$$ следовательно: $$W(13)=W(12)=\begin{pmatrix}-2\\0\\1\end{pmatrix}.$$

Можно убедиться, что в следующем итерактивном цикле все образы классифицируются правильно, т.е. вектор решений имеет вид: W=(-2,0,1)’ . Следовательно решающая функция будет: $d(X)=-2x_1+1$.

Алгоритм перцептрона можно представить в другой, эквивалентной форме, умножив пополненные образы одного из классов на –1. Таким образом, умножив все образы, например класса ω_2, на -1, алгоритм перцептрона можно переписать как:

$$W(k+1)=\begin{cases}W(k), & \text{если } W'(k)\times X(k)>0,\\ W(k)+cX(k), & \text{если } W'(k)\times X(k)\le 0,\end{cases} \qquad (7.26)$$

где с – положительное корректирующее приращение.

Разновидности перцептронного подхода. Варьируя способ выбора корректирующего приращения c, можно получить несколько модификаций алгоритма перцептрона.

В алгоритме *фиксированного приращения* корректирующее приращение **c** является константой, большей нуля (наш пример).

В алгоритме *коррекции абсолютной величины* ***c*** выбирается достаточно большим, для того чтобы гарантировать правильную классификацию образа после коррекции весов. Другими словами, если значение W’(k)×X(k)≤0, то коэффициент c выбирается так, чтобы выполнялось:

$$W'(k+1)\times X(k) = [W(k)+cX(k)]'\times X(k)>0. \qquad (7.27)$$

Один из способов, обеспечивающий неравенство состоит в выборе в качестве **c** *наименьшего целого числа*, превышающего значение |W’(k)×X(k)|/ [X’(k)×X(k)].

В алгоритме дробной коррекции c выбирается таким, чтобы величина |W(k)×X(k)-W’(k+1)×X(k)| составляла некоторую долю λ от величины |W(k)×X(k)|, т.е.

$$|W(k)\times X(k)-W'(k+1)\times X(k)| = \lambda\times|W(k)\times X(k)|, \qquad (7.28)$$

Подстановка W(k+1)= W(k)+c×X(k)| в выражение (7.26) дает: $c = \lambda \frac{|W'(k)\times X(k)|}{X'(k)\times X(k)}$.

Синтаксическое распознавание образов

Под строкой будем понимать конечное множество символов, каждый из которых является элементом некоторого заранее определенного произвольного непустого конечного множества символов, называемого *алфавитом.*

Если строка содержит *m* символов (подсчитываются все символы строки независимо от их повторений), то о такой строке говорят, что она имеет длину *m.* Например, строка 001110 имеет длину 6. Длина строки обозначается |x|. Пустой строкой называют строку длины 0, т.е. строку, не содержащую ни одного символа. Пустую строку принято обозначать буквой ε.

Пусть Σ - некоторый алфавит. Обозначим через Σ^* множество определенных над этим алфавитом строк. Например, если Σ = {0,1}, тогда Σ^*={ε,0,1,00,01,10,11,000,001,010,...}.

Если $x \in \Sigma^*$ - строка длины *m,* а $y \in \Sigma^*$- строка длины *n*, то их *объединение,* обозначаемое *xy,* может быть определено как строка длины *m* + *n*, причем первые *m* символов составляют строку, совпадающую со строкой *x,* а последние *n* символов составляют строку, совпадающую со строкой *y.*

Формальным языком L над алфавитом Σ называется произвольное подмножество множества Σ^*. Если L_1 и L_2 -два формальных языка, то их объединение $L_1L_2 = \{xy \mid x \in L_1, y \in L_2\}$ также является формальным языком.

Как и в случае объединения строк операция объединения формальных языков ассоциативна, но не коммутативна.. Для произвольного формального языка L справедливо следующее утверждение: $L\{\varepsilon\} = \{\varepsilon\}L = L$.

Объединение формального языка L с самим собой записывается как L^2 или в общем случае как $L^0=\{\varepsilon\}$, $L^1=L$, $L^i = LL^{i-1} = L^{i-1}L$ для $i \geq 2$. Замыкание Клини формального языка L, обозначаемое как L^*, может быть определено как $\bigcup_{i=0}^{\infty} L^i$. Если определить L^+ как $\bigcup_{i=1}^{\infty} L^i$, то определение замыкания Клини формального языка L можно записать как $L^*=L^+ \cup \{\varepsilon\}$ Обычно совокупность строк, принадлежащих множеству Σ^* и имеющих длину 2, обозначают Σ^2, и имеющих длину 3 – соответственно Σ^3, и т.д. Совокупность строк, принадлежащих множеству Σ^* и имеющих длину, большую или равную 1, обозначают Σ^+. Воспользуемся принятыми обозначениями и запишем следующее утверждение: $\Sigma^+ = \bigcup_{i=1}^{\infty} \Sigma^i$ и $\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i$.

Контекстная грамматика G может быть представлена совокупностью четырех объектов (N, T, P, S), где

N - конечное множество нетерминальных символов языка. Согласно принятому соглашению элементы множества N обозначаются прописными буквами (иногда с нижними индексами);

T - конечное множество терминальных символов {поэтому $N \cap T = \varnothing$}. Согласно принятому соглашению элементы множества T обозначаются строчными буквами (иногда с нижними индексами);

P - конечное множество *продукций* вида $\alpha \to \beta$, где α – строка в левой части продукции, такая, что $\alpha \in (N \cup T)^+$, а β - строка в правой части продукции, такая, что $\beta \in (N \cup T)^*$; элемент $S \in N$ - начальный символ грамматики.

Символ $\to$ используется для обозначения отношения «определяется как».

Пример грамматики G:

$$S \to A|B,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

Из первого правила следует, что S либо совпадает с A, либо совпадает с B. Из второго правила следует, что A представляет собой либо строку вида a, либо строку вида aa, либо строку вида $aa...a$. Аналогичные рассуждения справедливы для B.

Если строка β может быть получена из строки α с помощью одного или нескольких правил вывода, то будем записывать это следующим образом: $\alpha \Rightarrow \beta$. Если $\alpha_1 \Rightarrow \alpha_2$, $\alpha_2 \Rightarrow \alpha_3, \ldots, \alpha_{n-1} \Rightarrow \alpha_n$, $(n \geq 1)$, то эту последовательность выводов можно кратко записать так: $\alpha_1 \Rightarrow \alpha_2 \Rightarrow \alpha_3 \Rightarrow \ldots \Rightarrow \alpha_n$, или, еще более кратко, $\alpha_1 \overset{*}{\Rightarrow} \alpha_n$.

Воспользуемся теперь формальным определением грамматики для строгого определения выводимой строки. Пусть $G=(N,T,P,S)$. - контекстная грамматика и пусть $\gamma_1 \alpha \gamma_2 \in (N \cup T)^+$ - строка терминальных и нетерминальных символов длиной ≥ 1. Если $\alpha \to \beta$ - продукция из P, то строка

α в строке $\gamma_1\alpha\gamma_2$ может быть заменена строкой β, и в результате получим $\gamma_1\beta\gamma_2$ Это записывают следующим образом:

$$\gamma_1\alpha\gamma_2 \underset{G}{\Rightarrow} \gamma_1\beta\gamma_2.$$

В этом случае говорят, что строка $\gamma_1\alpha\gamma_2$ генерирует строку $\gamma_1\beta\gamma_2$, или что строка $\gamma_1\beta\gamma_2$ выводится из строки $\gamma_1\alpha\gamma_2$.

Если $\alpha_1, \alpha_2, \ldots, \alpha_n \in (N \cup T)^*$ и $\alpha_1 \underset{G}{\Rightarrow} \alpha_2$, $\alpha_2 \underset{G}{\Rightarrow} \alpha_3, \ldots, \alpha_{n-1} \underset{G}{\Rightarrow} \alpha_n$ (n≥1), то обычно пишут сокращенно $\alpha_1 \underset{G}{\Rightarrow} \alpha_2 \underset{G}{\Rightarrow} \alpha_3 \ldots \underset{G}{\Rightarrow} \alpha_{n-1} \underset{G}{\Rightarrow} \alpha_n$, или еще короче, $\alpha_1 \overset{+}{\underset{G}{\Rightarrow}} \alpha_n$, и при этом говорят, что строка α_n выводится из строки α_1 за один или более шагов. Далее, $\overset{+}{\underset{G}{\Rightarrow}}$ обозначает транзитивное замыкание отношения $\underset{G}{\Rightarrow}$, а $\overset{*}{\underset{G}{\Rightarrow}}$ обозначает рефлексивное замыкание отношения $\overset{+}{\underset{G}{\Rightarrow}}$. В таком случае $\alpha_1 \overset{*}{\underset{G}{\Rightarrow}} \alpha_n \Leftrightarrow \alpha_1 = \alpha_n$ или $\alpha_1 \overset{+}{\underset{G}{\Rightarrow}} \alpha_n$.

Если строка $\alpha \in (N \cup T)^*$ такая, что $S \overset{*}{\underset{G}{\Rightarrow}} \alpha$, то строку α называют *сентенциальной формой* контекстной грамматики *G:*

Сентенцией грамматики *G* называют произвольную сентенциальную форму из *T**, т.е. произвольную строку терминальных символов, которая может быть выведена из начального символа *S.* В этом случае множество всех сентенций грамматики *G* называют *языком, порождаемым грамматикой G,* и обозначают *L(G).*

Контекстно-свободные грамматики. Очень часто вид левой части каждой продукции грамматики может быть ограничен лишь единственным нетерминальным символом. Контекстные грамматики, продукции которых удовлетворяют такому ограничению, в теории формальных языков носят название *контекстно свободных грамматик.* Строгое определение контекстно-свободной грамматики может быть таким:

Контекстно-свободной грамматикой называют такую контекстную грамматику $G=(N,T,P,S)$, каждая продукция которой имеет вид $A \to \beta$, где $A \in N$, $\beta \in (N \cup T)^*$.

Любой язык, порожденный контекстно-свободной грамматикой, называют *контекстно свободным языком.* Поводом к возникновению термина "контекстно свободный" послужил тот

факт, что любой символ $A \in N$ в сентенциальной форме грамматики G может быть раскрыт согласно продукции $A \to \beta$ независимо от того, какими строками он окружен внутри самой сентенциальной формы.

Если $G = (N, T, P, S)$ – контекстно-свободная грамматика, то любой нетерминальный символ $x \in L(G)$ может быть выведен из начального символа S. Общепринятым методом представления вывода некоторой непустой строки считается *дерево вывода* (иногда его называют деревом грамматического разбора). Корнем этого дерева будет начальный символ S, остальные его узлы пометим слева направо следующим образом: $a_1, a_2, \ldots, a_n$, где $a_i \in T$, $1 \le i \le n$ и $x = a_1 a_2 \ldots a_n$; при этом внутренние узлы дерева будут соответствовать нетерминальным символам языка. Пусть A - нетерминальный символ, который может быть расширен с помощью продукции вида $A \to x_1 x_2 \ldots x_m$ ($x_i \in N \cup T$, $i = 1, 2, \ldots, m$). В таком случае узел дерева, соответствующий нетерминальному символу A, является родительским узлом для m узлов дерева, соответствующих m нетерминальным символам $x_1 x_2 \ldots x_m$ (рис.7.26).

Пример дерева вывода (рис. 7.27), иллюстрирующее вывод сентенции a^3b^2 контекстно-свободной грамматики G:

$$S \to AB,$$

$$A \to aA|a,$$

$$B \to bB|b.$$

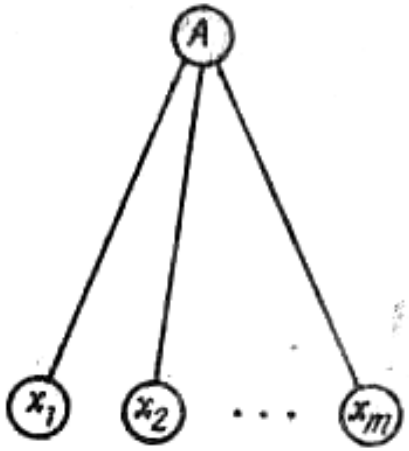


Рис.7.26.

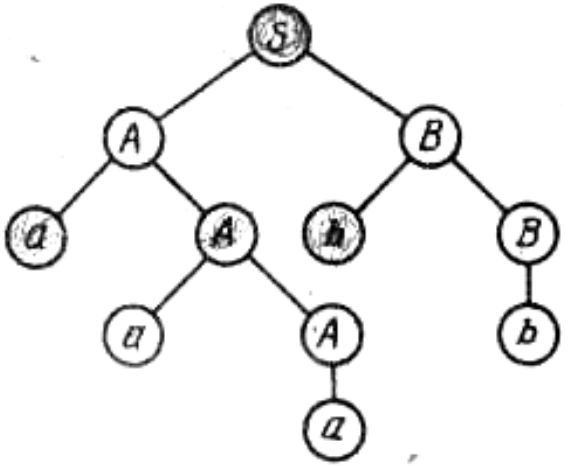


Рис.7.27.

Это дерево вывода в точности соответствует последовательности выводов:

$$S \Rightarrow AB,$$
$$\Rightarrow aAB,$$
$$\Rightarrow aaAB,$$
$$\Rightarrow aaaB,$$
$$\Rightarrow aaabB,$$
$$\Rightarrow aaabb.$$

Если $x \in L(G)$ то, как правило, можно найти несколько возможных вариантов вывода. Например, для сентенции a^3b^2:

$S \Rightarrow AB \Rightarrow AbB \Rightarrow aAbB \Rightarrow aaAbB \Rightarrow aaAbb \Rightarrow aaabb,$

$S \Rightarrow AB \Rightarrow aAB \Rightarrow aAbB \Rightarrow aAbb \Rightarrow aaAbb \Rightarrow aaabb,$

$S \Rightarrow AB \Rightarrow aAB \Rightarrow aaAB \Rightarrow aaAbB \Rightarrow aaabB \Rightarrow aaabb.$

Понятно, что все эти схемы вывода соответствуют одному и тому же дереву вывода.

Если для любого элемента $x \in L(G)$ все возможные схемы его вывода соответствуют одному и тому же дереву вывода, то такую контекстно свободную грамматику называют *однозначной*

грамматикой. Если же некоторому элементу $x \in L(G)$ соответствует несколько несовпадающих деревьев вывода, то контекстно свободная грамматика G называется *неоднозначной.*

Схема вывода сентенции $x \in L(G)$ называется *левосторонней,* если раскрывается всегда самый левый нетерминальный символ сентенциальной, формы. Поскольку теперь каждому дереву вывода соответствует единственная левосторонняя схема вывода, то можно переопределить неоднозначную грамматику следующим образом: контекстно свободная грамматика G называется неоднозначной, если существует такая сентенция $x \in L(G)$, которой можно поставить в соответствие две различные левосторонние схемы вывода.

Рассмотрим неоднозначную грамматику G', имеющую продукцию

$$S \to SbS|ScS|a.$$

Пусть сентенции $abaca \in L(G')$ соответствуют две различные левосторонние схемы вывода

$$S \Rightarrow SbS \Rightarrow abS \Rightarrow abScS \Rightarrow abacS \Rightarrow abaca,$$

$$S \Rightarrow ScS \Rightarrow SbScS \Rightarrow abScS \Rightarrow abacS \Rightarrow abaca.$$

Понятия формальных грамматик могут быть связаны с распознаванием следующим образом:

Пусть у нас имеются два класса образов ω_1 и ω_2 и пусть образы этих классов могут быть построены из признаков, принадлежащих некоторому конечному множеству T. Каждый образ может рассматриваться как цепочка или предложение некоторого языка.

Допустим, что существует грамматика G, такая, что порождаемый ею язык состоит из предложений (образов), принадлежащих только классу ω_1.

Очевидно, что эта грамматика может быть использована в целях классификации, т.к. заданный образ неизвестной природы может быть отнесен к ω_1, если он является предложением языка $L(G)$. В противном случае образ приписывается классу ω_2.

Образ попадает в класс ω_2 исключительно потому, что он не принадлежит классу ω_1. Тем не менее не исключено, что он не принадлежит и классу ω_2.

Он может представлять собой зашумленную или искаженную цепочку, которую лучше всего изъять из распознавания.

Чтобы обеспечить эту возможность, необходимо задать две грамматики G_1 и G_2, порождающие языки $L(G_1)$ и $L(G_2)$ соответственно.

Образ зачисляется в класс, язык которого считает этот образ правильным предложением. Если он не является предложением ни $L(G_1)$ ни $L(G_2)$, образ изымается.

В случае M классов мы рассматриваем M грамматик и связанных с ними языков $L(G_i)$, $i=1,2,\ldots,M$.

Распознаваемый объект относится к классу ω_i в том и только том случае, если он является предложением языка $L(G_i)$. Если объект является предложением более одного языка или не принадлежит ни одному из них, он может быть изъят из рассмотрения или произвольно отнесен к одному из классов.

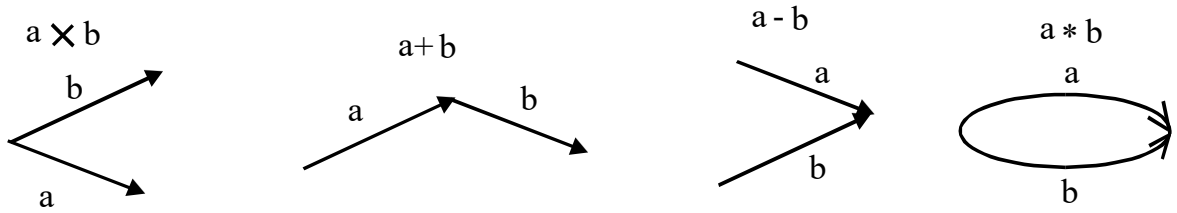
Чтобы получить реальную пользу от структурных свойств объекта в процессе синтаксического распознавания, понятие цепочки должно быть обобщено на двумерный случай. Правила подстановки в грамматиках цепочек заключаются в простом соединении цепочек с целью формирования новых.

Рассмотрим позиционный дескриптор ***НАД(a,b)***, обозначающий, что структура, представленная символом ***a***, расположена над структурой, представленной символом ***b***. И рассмотрим позиционный дескриптор ***СЛЕВА(a,b)***, обозначающий, что структура, представленная символом ***a***, расположена слева от ***b***. Квадратная структура □, составленная из непроизводных элементов | и -, описывается предложением: ***НАД(- , НАД(СЛЕВА(|, |), -)).***

Основная трудность при подобном подходе заключается в определении содержания дескрипторов ***НАД*** и ***СЛЕВА***. Так, предыдущему описанию «квадрат» удовлетворяет и структура:

В таких случаях накладываются ограничения. Так, для дескриптора ***НАД(a,b)*** таким ограничением является требование, чтобы хотя бы часть элемента ***a*** находилась над элементом ***b***. В этом случае данная структура не будет считаться допустимой, т.к. элемент – не находится над элементом | | и элемент | | не находится над элементом -.

В некоторых работах используется схема, заключающаяся в соединении структур только в особых точках. Например, чтобы каждая структура имела две выделенные точки. И соединение



структур должно происходить только в этих точках (рис.2.28.а).

Рис.7.28.а

Две выделенные точки интерпретируются как «головной» и «хвостовой» концы стрелы. Типичные правила соединения показаны на рис.2.28.б:

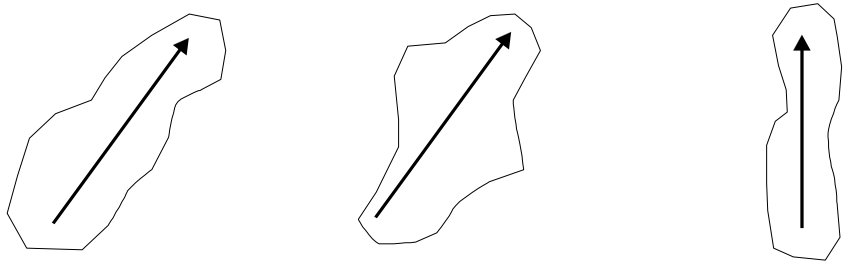


Рис.7.28.б

Пример 1. Структура типа квадрат. Непроизводными элементами служат вертикальный и горизонтальный отрезки, (Рис.7.29). Контекстно-свободная грамматика ***G***, способная порождать квадраты, задается как:

G=(N,T,P,S,) при **$T=\{a_1, a_2\}$, $V_N=\{S,O_1, O_2\}$;**

P: $S\rightarrow A(a_1,O_2)$; $O2\rightarrow A(O_1,a_1)$; $O1\rightarrow L(a_2,a_2)$,

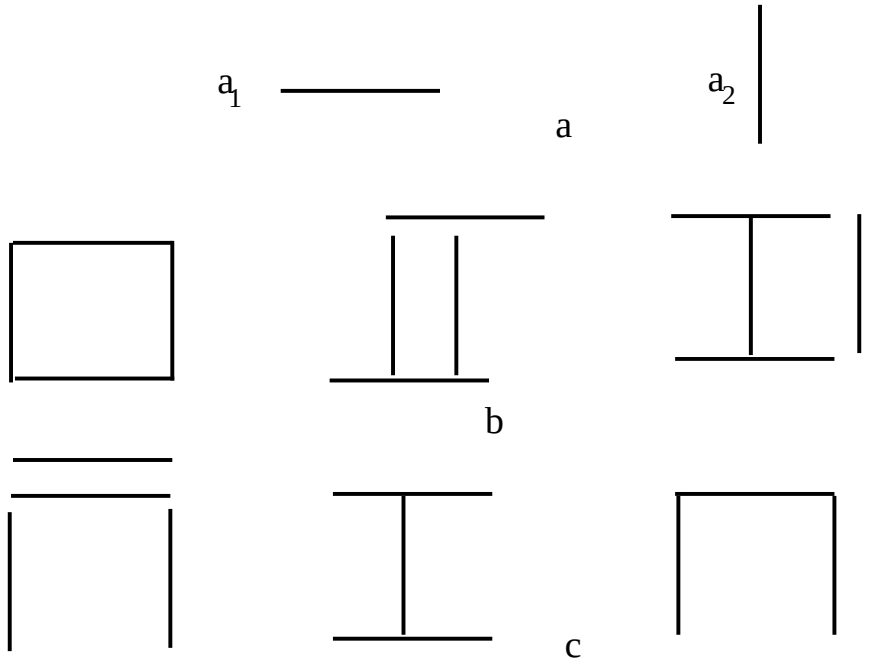


Рис.7.29

Здесь **A(x,y)** и **L(x,y)** читаются соответственно «x расположен над y» и «x расположен слева от y». При этом, позиционный дескриптор **A(x,y)** считается допустимым только в том случае, если часть **x** находится непосредственно над **y**, а дескриптор **L(x,y)** допустим тогда, когда часть **y** находится непосредственно справа от **x**.

Грамматический разбор представляет здесь простую процедуру, поскольку используется только одна последовательность правил подстановки. Предположим надо установить, принадлежит или нет данная структура к классу «квадрат». Синтаксический разбор «сверху-вниз» будет производиться следующим образом. Первое правило подстановки начинается с **S** и предполагает поиск некоторого объекта $\mathbf{O_2}$ ниже непроизводного элемента $\mathbf{a_1}$. Если ниже некоторого объекта $\mathbf{a_1}$ не найдено ни одного элемента, грамматический разбор прерывается и образ отклоняется. Если же правило применено успешно, на следующем шаге отыскивается некоторый объект $\mathbf{O_1}$ над другим непроизводным элементом $\mathbf{a_1}$. Если $\mathbf{O_1}$ обнаружен, грамматический разбор продолжается, в противном случае образ отклоняется. Наконец, объект $\mathbf{O_1}$, обнаруженный на предыдущем шаге, должен для принятия образа разделиться на два

непроизводных элемента $\mathbf{a_2}$ по условию $\mathbf{L(a_2,a_2)}$. Этой схеме грамматического разбора удовлетворяют структуры, показанные на Рис.7.28.(б) и не удовлетворяют на Рис.7.29.(в).

Пример 2. Одним из приложений лингвистических понятий в распознавании образов является язык PDL (Picture Description Language) – язык описания изображений. Здесь непроизводным элементом служит любая n-мерная структура с двумя выделенными точками – хвостовой и головной (Рис.2.28.а – для двумерных структур). Непроизводный элемент может примыкать к другим непроизводным элементам ***только*** в хвостовой и\или головной точке. Т.о. структуры языка PDL представляют собой ориентированные графы и для обработки этих структур можно использовать грамматики.

Основные правила соединения непроизводных элементов приведены ранее (Рис.7.28.б). Также могут быть использованы ***пустые*** непроизводные элементы для порождения внешне разъединенных структур, а также ***нулевые*** точки – непроизводный элемент с идентичными головной и хвостовой точками.

Рассмотрим простую грамматику языка PDL: **G=(N,T,P,S,)**,

где **N={S, A_1, A_2 A_3, A_4, A_5};**

T={a, b, c, d},

P: $S \rightarrow d + A_1$;

$A_1 \rightarrow c + A_2$;

$A_2 \rightarrow \sim d * A_3$;

$A_3 \rightarrow a + A_4$;

$A_4 \rightarrow b * A_5$;

$A_5 \rightarrow c$.

Причем, **~d** означает перемену мест головной и хвостовой точек непроизводного элемента **d**.

Непроизводные элементы показаны на Рис.2.30:

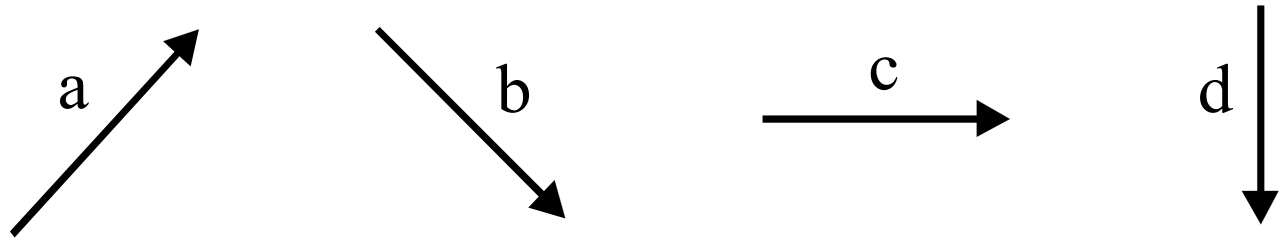


Рис.7.30.

Применение первого правила постановки приводит к получению непроизводного элемента ***d***, сопровождаемого еще не определенной переменной. На этом этапе мы знаем только, что хвостовая точка структуры, представленная символом A_1, будет связана с головной точкой элемента ***d***, потому что этот непроизводный элемент сопровождается оператором «+». Переменная A_1 разлагается на $c+A_2$, причем A_2 пока не определена. Аналогичным образом A_2 разлагается на $\sim d*A_3$.

Результаты применения этих трех правил показаны на Рис.7.31.(a,b,c). Из определения оператора «*» мы знаем, что при разложении элемента A_3 происходит его соединение с составной структурой ***8.7.c*** путем присоединения хвостовой точки к хвостовой, а головной точки к головной. Конечный результат показан на Рис.7.31.f.

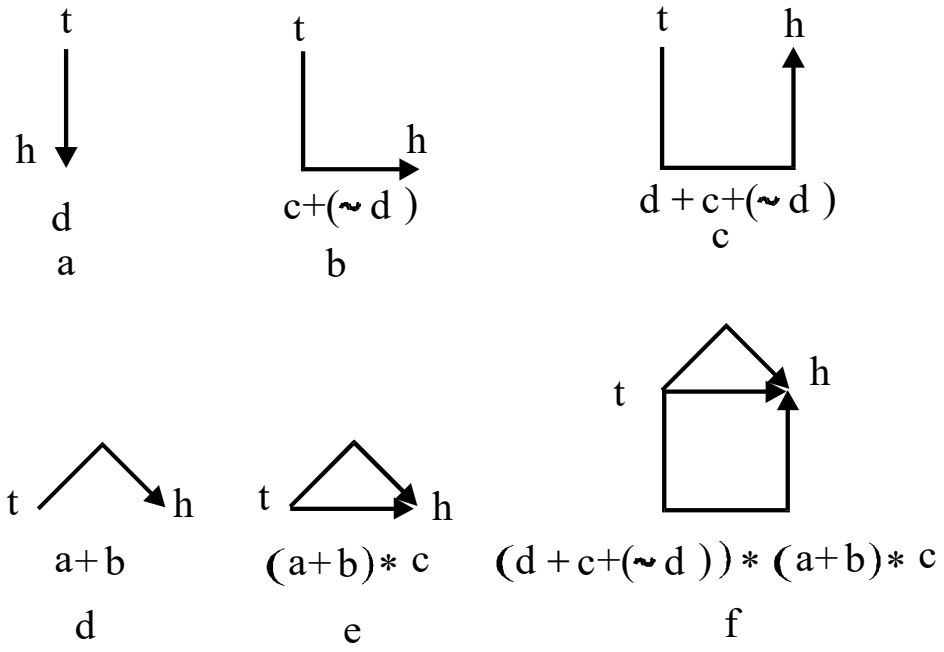


Рис.7.31.

Грамматика языка PDL, описанная выше, способна порождать только одну структуру. Однако, можно расширить число структур, введением рекурсивности – способности переменной замещаться этой переменной.

Пример:

P: $S \rightarrow d + A_1;$

$A_1 \rightarrow c + A_1;$

$A_1 \rightarrow \sim d * A_2;$

$A_2 \rightarrow a + A_2;$

$A_2 \rightarrow b * A_2;$

$A_2 \rightarrow c.$

Результатом применения этих правил в указанном порядке будет структура «домик» (Рис.7.31.f). Кроме этого, новое множество правил допускает возможность порождать и другие (бесконечные) структуры.

Грамматический разбор ***сверху вниз*** будет происходить в следующей последовательности. Предположим наш объект – «домик». Грамматический анализатор начинает с корневого символа ***S.*** В данном случае ***S*** заменяется на $d + A_1$ и т.д. приходим к выводу, что образ поддается грамматическому разбору. При наличии двух и более грамматик анализ

производится до тех пор, пока образ не идентифицируется либо пока не исчерпаются возможности грамматики; в последнем случае образ отклоняется.

Обучение и грамматический вывод

Используя лингвистическую терминологию, процедуру получения решений с помощью обучающей выборки легко интерпретировать как задачу получения грамматики из множества выборочных предложений. Эту процедуру обычно называют ***грамматическим выводом*** (или *восстановление грамматики*).

Модель вывода грамматики:

Рис.7.32.

Здесь задача заключается в том, что множество выборочных цепочек $\{x_i\}$ подвергается обработке с помощью адаптивного обучающего алгоритма. На выходе в конечном счете воспроизводится грамматика G, согласованная с данными цепочками,

Алгоритм Фельдмана. Для заданного множества терминальных цепочек выводит ***автоматную грамматику***. (Автоматная грамматика – грамматика вида **A→aA** или **A→a**,).

Алгоритм можно разделить на три части:

Первая часть формирует нерекурсивную грамматику.

Вторая часть преобразует ее в рекурсивную грамматику.

Затем в третьей части происходит упрощение этой грамматики.

Пример.

Пусть задано выборочное множество терминальных цепочек ***{caaab, bbaab, caab, bbab, cab, bbb, cb}***. Требуется построить автоматную грамматику, способную порождать эти цепочки. Алгоритм построения грамматики состоит из следующих этапов.

Часть1. Строится нерекурсивная грамматика, порождающая в точности заданное множество выборочных цепочек. Выборочные цепочки обрабатываются в порядке уменьшения длины. Правила подстановки строятся и прибавляются к грамматике по мере того, как становятся нужны для построения соответствующей цепочки из выборки. Заключительное правило постановки, используемое для порождения самой длинной выборочной цепочки, называется ***остаточным правилом***, а длина его правой части ***равна 2*** (т.е. имеет вид $\mathbf{A \rightarrow a_1 a_2}$).

В нашем примере первой цепочкой максимальной длины является ***caaab***.

Для ее порождения строятся следующие правила подстановки:

$S \rightarrow cA_1$

$A_1 \rightarrow aA_2$

$A_2 \rightarrow aA_3$

$A_3 \rightarrow ab$ где A_3 - правило остатка.

Вторая цепочка - ***bbaab***. Для порождения этой цепочки к грамматике добавляются следующие правила:

$S \rightarrow bA_4$

$A_4 \rightarrow bA_5$

$A_5 \rightarrow aA_6$

$A_6 \rightarrow ab$.

Для порождения третьей цепочки требуется добавление к грамматике одного правила

$$A_3 \rightarrow b.$$

Рассмотрев остальные цепочки, получаем, что множество правил подстановки, построенных для порождения обучающей выборки, имеет вид:

$S \rightarrow cA_1$	$A_3 \rightarrow ab$
$S \rightarrow bA_4$	$A_4 \rightarrow bA_5$
$A_1 \rightarrow b$	$A_5 \rightarrow b$
$A_1 \rightarrow aA_2$	$A_5 \rightarrow aA_6$
$A_2 \rightarrow b$	$A_6 \rightarrow b$
$A_2 \rightarrow aA_3$	$A_6 \rightarrow ab$
$A_3 \rightarrow b$	

Часть 2. В этой части, соединяя каждое правило остатка ***длины 2*** другим (неостаточным) правилом грамматики, получаем рекурсивную автоматную грамматику. Это происходит в результате слияния каждого нетерминального элемента правила остатка с нетерминальным элементом неостаточного правила, который может порождать остаток. Так, например, если A_r - остаточный нетерминал вида $A_r \rightarrow a_1 a_2$ и A_n - неостаточный нетерминал вида $A_n \rightarrow a_1 A_m$, где $A_m \rightarrow a_2$, все встречающиеся A_r заменяются на A_n, а правило подстановки $A_r \rightarrow a_1 a_2$ отбрасывается. Таким образом создается автоматная грамматика, способная порождать данную

обучающую выборку, а также обладающая общностью, достаточной для порождения бесконечного множества других цепочек.

В нашем случае $\mathbf{A_6}$ может слиться с $\mathbf{A_5}$, а $\mathbf{A_3}$ - с $\mathbf{A_2}$. В результате имеем:

$\mathbf{S \to cA_1}$	$\mathbf{A_2 \to b}$
$\mathbf{S \to bA_4}$	$\mathbf{A_4 \to bA_5}$
$\mathbf{A_1 \to b}$	$\mathbf{A_5 \to b}$
$\mathbf{A_1 \to aA_2}$	$\mathbf{A_5 \to aA_5}$
$\mathbf{A_2 \to b}$	$\mathbf{A_5 \to b}$
$\mathbf{A_2 \to aA_2}$	

Здесь рекурсивными являются правила: $\mathbf{A_2 \to aA_2}$ и $\mathbf{A_5 \to aA_5}$.

Часть 3. Полученная грамматика упрощается объединением эквивалентных правил постановки. Два правила с левыми частями $\mathbf{A_i}$ и $\mathbf{A_j}$ эквивалентны, если соблюдены следующие условия. Предположим, что, начиная с $\mathbf{A_i}$ можно породить множество цепочек $\boldsymbol{\{x\}_i}$, а начиная с $\mathbf{A_j}$ - множество цепочек $\boldsymbol{\{x\}_j}$. Если $\boldsymbol{\{x\}_i = \{x\}_j}$, то два правила постановки считаются

эквивалентными, и каждый символ $\mathbf{A_j}$ может быть заменен на $\mathbf{A_i}$ без ущерба для языка, порождаемого этой грамматикой.

В нашем примере эквивалентны правила с левыми частями $\mathbf{A_1}$ и $\mathbf{A_2}$. После слияния $\mathbf{A_1}$ и $\mathbf{A_2}$ получаем:

$$\mathbf{S \to cA_1} \qquad \mathbf{A_4 \to bA_5}$$

$$\mathbf{S \to bA_4} \qquad \mathbf{A_5 \to b}$$

$$\mathbf{A_1 \to b} \qquad \mathbf{A_5 \to b}$$

$$\mathbf{A_1 \to aA_1} \qquad \mathbf{A_5 \to aA_5}$$

Точно также эквивалентны правила с левыми частями $\mathbf{A_1}$ и $\mathbf{A_5}$. После слияния $\mathbf{A_1}$ и $\mathbf{A_5}$ получаем:

$$\mathbf{S \to cA_1\ ,\ S \to bA_4\ ,\ A_1 \to b\ ,\ A_1 \to aA_1\ ,\ A_4 \to bA_1.}$$

Дальнейшее слияние правил невозможно, поэтому алгоритм в процессе обучения строит следующую грамматику:

G=(N,T,P,S), **N={S,A,B},** **T={a,b,c}** где

P: S→cA

S→bB

A→aA

B→bA

A→b

Эта грамматика порождает обучающую выборку, использованную в процессе ее вывода.