МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ АГЕНТСВО ПО ОБРАЗОВАНИЮ
БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ ИМ В.Г.ШУХОВА

Математическая логика и теория алгоритмов

Учебное пособие

Терехов Д.В., Куценко Д.А.

Белгород 2007 (v 1.0.0b)

Учебное пособие посвящено изложению основ математической логике и теории алгоритмов. Учебное пособие предназначено для студентов, обучающихся по направлению «Информатика и вычислительная техника».

Высказывания, логика высказываний

Высказывание - это утверждение об изучаемых объектах имеющее однозначное и точно определенное значение. В русском языке высказывание представляет собой повествовательное предложение, о котором можно сказать, что оно сообщает нам нечто верное либо нечто совершенно неверное. Следовательно, высказывание может быть либо истинным, либо ложным. Иначе говоря, каждому высказыванию приписывается *истинностное значение*. В классической (двузначной) логике рассматривается всего два истинностных значения: T – «истина» и $\perp$ - «ложь».

Если даны два высказывания A и B, то логика, во-первых, говорит нам, как из них построить новое высказывание, а во-вторых, учит, как найти его истинностное значение, Для построения новых высказывании используются логические связки $\&, \vee, \Rightarrow, \Leftrightarrow, \neg$:

$$A \& B,\ A \vee B,\ A \Rightarrow B,\ A \Leftrightarrow B,\ \neg A,\ \neg B$$

Связка & интерпретируется как союз «и» (*конъюнкция*); связка $\vee$ - «или» (*дизъюнкция*); связка $\Rightarrow$ как глаголы «следует» (*импликация*); «вытекает», связка $\Leftrightarrow$ - «тогда и только тогда, когда»; и, наконец, $\neg$ - понимается как отрицание высказывания A («не A»).

Для установления истинностного значения нового высказывания применяют таблицы истинности:

A	B	$A \& B$	$A \vee B$	$A \Rightarrow B$	$A \Leftrightarrow B$	$\neg A$
T	T	T	T	T	T	$\perp$
T	$\perp$	$\perp$	T	$\perp$	$\perp$	$\perp$
$\perp$	T	$\perp$	T	T	$\perp$	T
$\perp$	$\perp$	$\perp$	$\perp$	T	T	T

Логическая наука, занимающаяся высказываниями, называется *алгеброй высказываний*, Название это становится понятным, если связку & назвать произведением, а $\vee$ - сложением. Мы как бы перемножаем и складываем высказывания.

Классическая логика как наука о правильных умозаключениях, основывается на следующих четырех законах.

Закон тождества: $(A \Leftrightarrow A) \equiv \mathrm{T}$

Закон *непротиворечия*: $(A \& \neg A) \equiv \perp$

Закон *исключенного третьего*: $(A \vee \neg A) \equiv \mathrm{T}$

Закон *достаточного основания*: никакое высказывание А не может быть принято, если оно не является следствием, полученным в ходе применения силлогизмов из ранее принятых утверждений или строго установленных фактов, выраженных также в форме высказываний.

В логике переменным параметрам, входящим в утверждения, естественно приписывать значение Т или $\perp$, получая при этом высказывания. Назовем такие переменные параметры, следуя традиции, *пропозициональными переменными*. Как правило, для краткости пропозициональные переменные именуют просто переменными. Будем обозначать их как $X_1, X_2, ..., X_n, ...$.

Поскольку логика под влиянием математики превратилась в математическую логику, то естественным было использование в логике терминологии, принятой в математике. В частности, утверждение А, содержащее переменные, стали называть *формулой*. Для того чтобы быть точным, логик обязан был дать строгое определение того утверждения, которое он хотел бы назвать формулой. Отсюда следующее индуктивное определение формулы:

1) пропозициональная переменная есть (атомарная) формула;
2) если A и B формулы, то $A \& B$, $A \vee B$, $A \Rightarrow B$ формулы;
3) если A формула, то $\neg A$ формула.

Как видим, при построении формул используются скобки (и). Однако, как часто делается, лишние скобки опускают. К примеру, формулу $((A \vee B) \& C)$ пишут как $(A \vee B) \& C$, а $(A \& B)$ как $A \& B$. Существуют и другие правила, позволяющие не ставить лишних скобок.

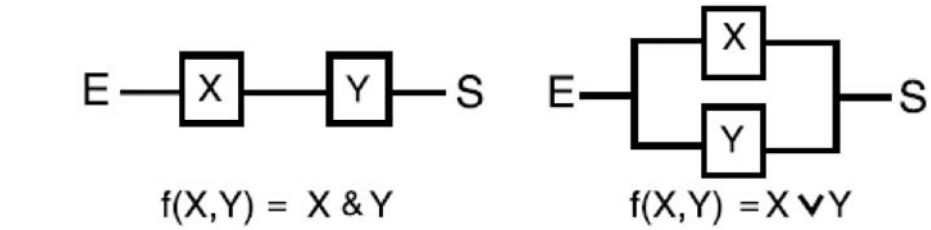
Формулу A, содержащую пропозициональные переменные $X_1, X_2, ..., X_n$, будем обозначать как $A(X_1, X_2, ..., X_n)$.

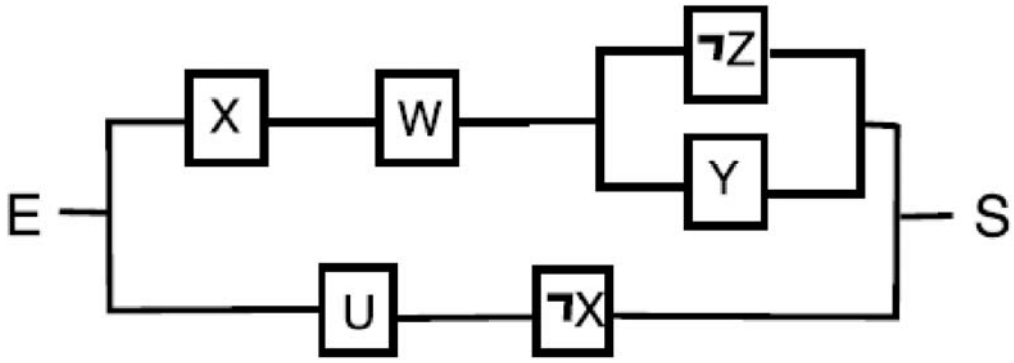
Теория, которая изучает формулы, определенные выше, называется *алгеброй высказываний*. В алгебре высказывании каждая пропозициональная переменная, каждая формула принимает одно из двух значений - Т (истина) или $\perp$ (ложь).

Каждой формуле алгебры высказываний можно поставить в соответствие так называемую *релейно-контактную схему*.

Релейно-контактная схема строится в предположении, что пропозициональная переменная X - это *замыкающий контакт* в электрической схеме, который замкнут при подаче (управляющего) тока X т.е. $X = \mathrm{T}$, и разомкнут при его отсутствии - $X = \perp$. Далее, формуле $\neg X$ отвечает *размыкающий контакт*, который замкнут, т.е. $\neg X = \mathrm{T}$, пока нет тока ($X = \perp$), и размыкается - $\neg X = \perp$, когда ток есть ($X = \mathrm{T}$).

Удобно вместо символа T писать символ 1 (ток проходит), а вместо $\perp$ - 0 (ток не проходит). Дизъюнкции $A \vee B$ соответствует параллельное соединение (замыкающих) контактов, а конъюнкции $A \& B$ - последовательное соединение.





f(X,Y,Z,U,W) = (X & W & (¬Z v Y)) v (U &¬X)

Две формулы $A(X_1, X_2, ..., X_n)$ и $B(X_1, X_2, ..., X_n)$ называются *равносильными*, если их значения совпадают при любых значениях входящих в них переменных $X_1, X_2, ..., X_n$. Для равносильных формул используется обозначение $A \equiv B$.

В логике выделяют следующие равносильные формулы:

$$A \& B \equiv B \& A$$
$$A \vee B \equiv B \vee A$$
$$(A \& B) \& C \equiv A \& (B \& C)$$
$$(A \vee B) \vee C \equiv A \vee (B \vee C)$$
$$A \& A \equiv A$$
$$A \vee A \equiv A$$
$$A \& (B \vee C) \equiv (A \& B) \vee (A \& C)$$
$$A \vee (B \& C) \equiv (A \vee B) \& (A \vee C)$$
$$A \& \perp \equiv \perp$$
$$A \& \mathrm{T} \equiv A$$
$$A \vee \perp \equiv A$$
$$A \vee \mathrm{T} \equiv \mathrm{T}$$
$$\neg(\neg A) \equiv A$$
$$\neg(A \& B) \equiv \neg A \vee \neg B$$
$$\neg(A \vee B) \equiv \neg A \& \neg B$$

$$A \,\&\, \neg A \equiv \bot$$
$$A \vee \neg A \equiv \mathrm{T}$$
$$A \vee (A \,\&\, B) \equiv A$$
$$A \,\&\, (A \vee B) \equiv A$$
$$A \Rightarrow B \equiv \neg A \vee B$$
$$A \Rightarrow B \equiv \neg(A \,\&\, \neg B)$$

Формула $A(X_1, X_2, ..., X_n)$ называется *истинной* (*выполнимой*), если существует набор значений переменных $X_{10}, X_{20}, ..., X_{n0}$ такой, что $A(X_{10}, X_{20}, ..., X_{n0}) = \mathrm{T}$.

Формула $A(X_1, X_2, ..., X_n)$ называется *общезначимой*, если при любом наборе переменных она истинна. Для общезначимой формулы используется символическая запись $\mapsto A$.

Пусть дана произвольная формула $A(X_1, X_2, ..., X_n)$. Можно ли как-то проверить, что она является общезначимой? Если существует такой способ (алгоритм), позволяющий в конечное число шагов убедиться в этом то говорят, что проблема проверки общезначимости формул алгебры высказываний разрешима. *Проблема разрешимости* в алгебре высказываний имеет, положительное решение.

Две формулы, не содержащие $\Rightarrow$ и $\Leftrightarrow$, называются *двойственными*, если каждую из них можно получить из другой заменой $\&, \vee, \mathrm{T}, \bot$ соответственно на $\vee, \&, \bot, \mathrm{T}$. *Принцип двойственности* утверждает следующее: если две формулы (не содержащие знаков $\Rightarrow$ и $\Leftrightarrow$) равносильны, то двойственные им формулы тоже равносильны.

Литерой (*литералом*) в логике высказываний называется любая пропозициональная переменная с отрицанием или без него. *Дизъюнкт* - дизъюнкция конечного числа литер. Следующие формулы являются примерами дизъюнктов: $(P \vee \neg Q \vee R)$, $(\neg T)$, $(\neg S \vee Q)$. Любой дизъюнкт, содержащий хотя бы одну литеру, является выполнимой формулой. Единственным невыполнимым дизъюнктом является *пустой дизъюнкт*, то есть дизъюнкт, не содержащий ни одной литеры.

Конъюнктивной нормальной формой (*КНФ*) называется конъюнкция конечного числа дизъюнктов. *Приведенной конъюнктивной нормальной формой* называется КНФ, в которой опущены дизъюнкты, содержащие контрарные пары (тавтологии) и повторения литер в пределах одного и того же дизъюнкта. *Совершенной конъюнктивной нормальной формой* (*СКНФ*) называется КНФ, в каждый дизъюнкт которой входят все переменные.

Любая формула может быть неоднозначно преобразована в логически эквивалентную ей КНФ с использованием следующего алгоритма:

•Исключить из формулы все связки импликации и эквивалентности.
•Сузить области действия отрицаний и исключить двойные отрицания.
•Применить необходимое число раз правило дистрибутивности дизъюнкции относительно конъюнкции:

Пусть даны формулы $A_1, A_2, ..., A_m, B$. Формула B является *логическим следствием* формул $A_1, A_2, ..., A_m$, если, придавая значения переменным $X_1, X_2, ..., X_n$ от которых зависят все рассматриваемые формулы, всякий раз, когда истинны одновременно все формулы $A_1, A_2, ..., A_m$, истинна и формула B.

Для логического следствия используется запись $A_1, A_2, ..., A_m \mapsto B$.

Теорема дедукции: если $A \mapsto B$, то $\mapsto (A \Rightarrow B)$. Следующее утверждение является важным следствием теоремы дедукции: $A_1, A_2, ..., A_m \mapsto B$ тогда и только тогда, когда $\mapsto (A_1 \,\&\, A_2 \,\&\, ... \,\&\, A_m) \mapsto B$.

Силлогизм - это правило, позволяющее из истинных высказываний получать новые истинные высказывания. Приведем неполный список силлогизмов:

$$\frac{A,\ A \Rightarrow B}{B}$$ (Modus ponens)

$$\frac{A \Rightarrow B,\ \neg B}{\neg A}$$ (Modus tollens)

$$\frac{A \Rightarrow B,\ B \Rightarrow C}{A \Rightarrow C}$$

$$\frac{A,\ B}{A \,\&\, B}$$

$$\frac{A \,\&\, B}{A,\ B}$$

$$\frac{A \vee B,\ \neg B}{A}$$

$$\frac{\neg (A \,\&\, B),\ A}{\neg B}$$

$$\frac{A \Rightarrow (B \Rightarrow C)}{B \Rightarrow (A \Rightarrow C)}$$

$$\frac{A \Rightarrow (B \Rightarrow C)}{A \,\&\, B \Rightarrow C}$$

$$\frac{A \,\&\, B \Rightarrow C}{A \Rightarrow (B \Rightarrow C)}$$

Предикаты, логика предикатов

Логика предикатов - это расширение возможностей логики высказываний, позволяющее строить высказывания с учетом свойств изучаемых объектов или отношений между ними.

Одноместный предикат $P(x)$ - это утверждение об объекте x, где x рассматривается как переменная. Иначе говоря, если в $P(x)$ вместо x подставить конкретный изучаемый объект a, то получаем высказывание, принадлежащее алгебре высказывании. В таком случае $P(a) = \mathrm{T}$ или $P(a) = \perp$.

Многоместный предикат $P(x_1,...,x_n)$ - это утверждение об объектах $x_1,...,x_n$, где $x_1,...,x_n$ рассматриваются как переменные. Следовательно, при подстановке $x_1 = a_1,...,x_n = a_n$ получим высказывание $P(a_1,...,a_n)$, являющееся истинным или ложным.

Помимо подстановки вместо переменных конкретных объектов, существует еще способ получения высказываний из предикатов высказываний. Для этого применяются *квантор всеобщности* $\forall$ и *квантор существования* $\exists$.

Символ $\forall x$ интерпретируется как фраза «для всех x». При добавлении к предикату $P(x)$ квантора всеобщности мы получим высказывание $\forall x P(x)$. Оно примет истинное значение, если предикат $P(x)$ выполняется для всех объектов $a_1,...,a_n,...$, которые можно подставить вместо x. Что можно записать $\forall x P(x) \equiv P(a_1) \& ... \& P(a_n) \& ...$.

Символ $\exists x$ представляет фразу «существует x». При добавлении данного квантора к предикату получаем также высказывание $\exists x P(x)$. Данное высказывание будет истинным, если предикат $P(x)$ выполняется хотя бы для одного из объектов $a_1,...,a_n,...$, которые можно подставить вместо x. Это можно записать $\exists x P(x) \equiv P(a_1) \vee ... \vee P(a_n) \vee ...$

Переменная x, входящая в формулу A называется *связанной*, если она находится под действием квантора $\forall x$ или $\exists x$ В противном случае, переменная x в формуле A является *свободной*, Например, в формулах $\exists x(x = y)$ и $\forall x B(x, y)$ переменная x связанная, а переменная y свободная. Ясно, что формула без свободных переменных является высказыванием.

За каждой формулой скрывается нечто, в связи с чем она была написана и о чем она повествует, то есть скрывается ее содержание Содержательная часть формул, их смысл, относится к специальному разделу логики, называемому *семантикой*. Выяснить содержание формулы можно обращаясь к реальному миру предметов. Делается это посредством интерпретации формулы.

Чтобы определить *интерпретацию* для формулы логики первого порядка, мы должны указать предметную область (область значений предметных переменных) и значения констант, функциональных и предикатных символов, встречающихся в формуле.

Определение. Интерпретация формулы F логики предикатов состоит из непустой (предметной) области D и указания значения всех констант, функциональных символов и предикатных символов, встречающихся в F.

1.Каждой константе мы ставим в соответствие некоторый элемент из D.
2.Каждому n-местному функциональному символу мы ставим в соответствие отображение из D^n в D.
3.Каждому n-местному предикатному символу мы ставим и соответствие отображение из D^n в $\{\mathrm{T}, \perp\}$.

Содержанием при интерпретации формулы наполняются благодаря тому, что элементы множества D уже привязаны к конкретной реальности, знакомы и понятны. Например, в случае, когда $D = \mathbb{R}$, т.е. является множеством действительных чисел, у нас не появляется чувства беспокойства, под формулами мы понимаем сведения из математического анализа, который прочно привязан к практической деятельности инженеров, физиков, химиков и т.д. Следовательно, нам становится ясным, когда формула при определенной фиксации своих переменных истинна, а когда ложна.

Формула A логики предикатов называется *выполнимой* в области M, если существуют значения переменных входящих в эту формулу и отнесенных к области M (иначе – *существует модель*), при которых формула A принимает истинные значения.

Формула A логики предикатов называется *выполнимой*, если существует область, на которой эта формула выполнима.

Формула A логики предикатов называется *тождественно-истинной* в области M, если она принимает истинные значения для всех значений переменных, входящих в эту формулу и отнесенных к этой области.

Формула A логики предикатов называется *общезначимой*, если она тождественно-истинна на всякой области (на любой модели). Для общезначимой формулы пишем $\mapsto A$. Если формула A общезначима, то формула $\neg A$ называется логически ложной, или противоречием.

Пусть даны две формулы $A(x_1,...,x_n)$ и $B(x_1,...,x_n)$. Формула B является *логическим следствием* формулы A, если во всякой интерпретации формула B выполнена на каждом наборе переменных $x_1 = a_1,...,x_n = a_n$ на котором

выполнена формула A. Символически для логического следствия используют обозначение $A \mapsto B$.

В логике предикатов выполняется *теорема дедукции*: $A \mapsto B$ тогда и только тогда, когда $\mapsto A \Rightarrow B$.

Формулы $A(x_1,...,x_n)$ и $B(x_1,...,x_n)$ называются *равносильными*, если $A \mapsto B$ и $B \mapsto A$. Для равносильных формул используется запись: $A \equiv B$.

Примеры равносильных формул:

$$\forall x A(x) \,\&\, B \equiv \forall x[A(x) \,\&\, B]$$
$$\exists x A(x) \,\&\, B \equiv \exists x[A(x) \,\&\, B]$$
$$\forall x A(x) \vee B \equiv \forall x[A(x) \vee B]$$
$$\exists x A(x) \vee B \equiv \exists x[A(x) \vee B]$$
$$[\forall x A(x) \Rightarrow B] \equiv \exists x[A(x) \Rightarrow B]$$
$$[\exists x A(x) \Rightarrow B] \equiv \forall x[A(x) \Rightarrow B]$$
$$[B \Rightarrow \forall x A(x)] \equiv \forall x[B \Rightarrow A(x)]$$
$$[B \Rightarrow \exists x A(x)] \equiv \exists x[B \Rightarrow A(x)]$$
$$\neg \forall x A(x) \equiv \exists x \neg A(x)$$
$$\neg \exists x A(x) \equiv \forall x \neg A(x)$$

Всякую формулу логики предикатов А с помощью приведенных тождеств можно привести к равносильной формуле в *предваренной форме*, в которой кванторы $\forall$, $\exists$ не перемешиваются, а встречаются группами и все «вынесены влево», т.е. либо к виду $\exists x_1...\exists x_{n_1} \forall y_1...\forall y_{n_2} \exists z_1...\exists z_{n_3}...B(x_1,...,x_{n_1},y_1,...,y_{n_2},z_1,...,z_{n_3},...)$, либо к виду $\forall x_1...\forall x_{n_1} \exists y_1...\exists y_{n_2} \forall z_1...\forall z_{n_3}...B(x_1,...,x_{n_1},y_1,...,y_{n_2},z_1,...,z_{n_3},...)$, где в формуле В нет кванторов. Легко добиться, чтобы последними стояли кванторы существования. Для этого используется тождество: $B(x_1,...,z_1,...) \equiv \exists u[B(x_1,...,z_1,...) \,\&\, I(u)]$, где $I(u)$ - произвольная общезначимая формула.

Будем теперь «снимать» в формуле А последовательно группы кванторов слева направо, заменяя их на функции. Путеводной здесь является идея, что пара кванторов $\forall u \exists v$ - это функция $v = f(u)$. Следовательно, набору кванторов $\forall y_1...\forall y_{n_2} \exists z_i$ отвечает функция $\exists z_i = g_i(y_1,...,y_{n_2})$. Если самой левой группой кванторов являются кванторы существования $\exists x_1...\exists x_{n_1}$, то им соответствуют постоянные функции, которые сводятся к заданию их значений $a_1...a_{n_1}$.

Сняв группу кванторов в формуле А, мы в Б оставляем переменные $y_1...y_{n_2}$, по которым стояли кванторы всеобщности а вместо следующих за ними переменных $z_1...$, связанных кванторами существования, подставляем полученные функции $g_i(y_1,...,y_{n_2})$.

В результате имеем набор *сколемовских функций* $a_1,...,a_{n_1}, g_1(y_1,...,y_{n_2}),...,g_{n_3}(y_1,...,y_{n_2}),...,h_1(w_1,...,w_{n_{k-1}}),...h_{n_k}(w_1,...,w_{n_{k-1}})$ и формулу $A_s = \forall y_1 ... \forall y_{n_2} ... \forall w_1 ... \forall w_{n_{k-1}} B(a_1,...,a_{n_1}, y_1,..., g_1(y_1,...,y_{n_2}),..., w_1,..., h_1(w_1,...,w_{n_{k-1}}),...h_{n_k}(w_1,...,w_{n_{k-1}})$

Формула A_s не содержит кванторов существования и является $\forall$-формулой, полученной в результате *сколемизации* исходной формулы А.

Метод резолюций

Предположим, что дана формула логики высказываний А. Как проверить общезначимость формулы А? Это задача решается с помощью *метода резолюций*. Очевидно, что $\mapsto A$ тогда и только тогда, когда является противоречием формула $\neg A$.

Формулу $\neg A$ приводим к конъюнктивной нормальной форме (КНФ). КНФ - это формула, равносильная данной формуле и записанная в виде конъюнкции элементарных дизъюнкций, построенных на пропозициональных переменных, т.е. в данном случае $\neg A = D_1 \& ... \& D_p$, где D_i есть дизъюнкция конечного числа пропозициональных переменных или их отрицаний. Тем самым мы формируем множество дизъюнктов $K = \{D_1, ..., D_p\}$.

Два дизъюнкта этого множества D_i и D_j. содержащие пропозициональные переменные с противоположными знаками, - *контрарные литералы* (к примеру Y и $\neg Y$), и, следовательно, $D_i = D_i' \vee Y$, $D_j = D_j' \vee \neg Y$ формируют третий дизъюнкт - *резольвенту* $D_i' \vee D_j'$, в которой исключены контрарные литералы:

$$R\frac{D_i' \vee Y, D_j' \vee \neg Y}{D_i' \vee D_j'}.$$

В частности, если $D_i = Y$ и $D_j = \neg Y$, то резольвента для них - это дизъюнкция, ничего не содержащая (отсутствуют D_i' и D_j'). Ее называют *пустой резольвентой* и обозначают знаком $\square$.

Резольвента $D_i' \vee D_j'$ - это логическое следствие дизъюнктов $D_i = D_i' \vee Y$ и $D_j = D_j' \vee \neg Y$ т.е. $D_i, D_j \mapsto (D_i' \vee D_j')$.

Неоднократно применяя правило получения резольвент к множеству дизъюнкт, стремятся получить пустой дизъюнкт $\square$. Наличие пустого дизъюнкта $\square$ свидетельствует о получении противоречия, поскольку пустая резольвента получается из двух противоречащих друг другу дизъюнктов Y и $\neg Y$, каждый из которых логическое следствие формулы $\neg A$ в соответствии с правилом $\frac{A \& B}{A, B}$.

Доказательство от противного: если $\Gamma, \neg A \mapsto \perp$, где Γ – множество формул, то $\Gamma \mapsto A$. Следовательно, получение противоречия с помощью формулы $\neg A$ означает общезначимость формулы A.

Изложим по шагам алгоритм метода резолюций.

Шаг 1. Принять отрицание формулы A, т.е. $\neg A$.

Шаг 2. Привести формулу $\neg A$ к конъюнктивной нормальной форме.

Шаг 3. Выписать множество ее дизъюнктов $K = \{D_1, ..., D_p\}$.

Шаг 4. Выполнить анализ пар множества K по правилу: если существуют дизъюнкты D_i и D_j, один из которых (D_i) содержит литерал X, а другой (D_j) - контрарный литерал $\neg X$, то нужно соединить эту пару логической связкой дизъюнкции ($D_i \vee D_j$) и сформировать новый дизъюнкт – резольвенту, исключив контрарные литералы X и $\neg X$.

Шаг 5. Если в результате соединения дизъюнктов, содержащих контрарные литералы, будет получена пустая резольвента $\square$, то результат достигнут (доказательство подтвердило противоречие), в противном случае включить резольвенту в множество дизъюнктов K и перейти к шагу 4.

При реализации указанного алгоритма возможны три случая:

1. Среди текущего множества дизъюнктов нет резольвируемых. Это означает, что формула A не является общезначимой.
2. На каком-то шаге получается пустая резольвента. Формула A общезначима.
3. Процесс не останавливается, т.е. множество дизъюнкт пополняется все новыми резольвентами, среди которых нет пустых. В таком случае нельзя ничего сказать об общезначимости формулы A.

Метод резолюций пригоден и для доказательства того, что формула B является логическим следствием формул $F_1, ..., F_n$, поскольку, как вытекает, из следствия $F_1, ..., F_n \mapsto B \Leftrightarrow\mapsto (F_1 \& ... \& F_n \Rightarrow B)$. Для того чтобы «запустить» алгоритм метода резолюций, нужно воспользоваться тождеством $(F_1 \& ... \& F_n \Rightarrow B) \equiv \neg(F_1 \& ... \& F_n \& \neg B)$.

Следовательно, формула $A \equiv \neg(F_1 \& ... \& F_n \& \neg B)$ общезначима, если формула $\neg A \equiv F_1 \& ... \& F_n \& \neg B$ является противоречием. Далее применяем описанный по шагам метод резолюций.

Предположим, что дана формула логики предикатов А. Как проверить общезначимость этой формулы? Применим ли метод резолюций? Применим, но с некоторыми дополнениями.

Если формула А не содержит предикатов и знаков операций (функций), то фактически имеем формулу логики высказываний, поскольку предикаты можно рассматривать как пропозициональные переменные. Следовательно, метод, резолюций не требует существенных изменений. В случае, если формула А содержит кванторы, то ее надо привести к предваренной форме, а затем устранить кванторы, вводя сколемовскне функции. Появятся, естественно, при этом знаки операций (функций). Затем полученную

формулу приводят к конъюнктивной нормальной форме и ищут резольвенты для дизъюнктов.

Что делать, если два контрарных литерала, в данном случае они имеют вид $P(...)$ и $\neg P(...)$, содержат два разных терма? Например, $D_i = D_i' \vee P(a,x)$ и $D_j = D_j' \vee \neg P(a, f(b))$. В этом случае проводят их унификацию $< x \mid f(b) >$, т.е. замену термов. Символическая запись $A(t_i) < t_1 \mid t_2 >$ означает замену в формуле A терма t_1 термом t_2.

$$D_i < x \mid f(b) >= [D_i' \vee P(a,x)] < x \mid f(b) >= D_i' < x \mid f(b) > \vee P(a, f(b))$$

$$D_j < x \mid f(b) >= [D_j' \vee \neg P(a, f(b))] < x \mid f(b) >= D_j' < x \mid f(b) > \vee \neg P(a, f(b))$$

Далее ищется резольвента: $R\dfrac{D_i, D_j}{D_i' < x \mid f(b) > \vee D_j' < x \mid f(b) >}$.

Сформулируем более строго правила замены термов и понятие унификации.

Замена - это пара вида $< x \mid t >$, где x - переменная, а t - терм. Применение замены $< x \mid t >$ к терму t_0, входящему в некоторую формулу, определяется индуктивно:

- $x < x \mid t >= t$, если x - переменная;
- $a < x \mid t >= t$, если a - константа (фиксированный объект);
- $f(t_1,...,t_n) < x \mid t >= f(t_1 < x \mid t >,...,t_n < x \mid t >)$

Таким образом, *унификация двух последовательностей термов* $t_1,...,t_n$ и $\tau_1,...,\tau_n$ - это замена $< x \mid t >$ акая, что $t_i < x \mid t >= \tau_i < x \mid t >, i = (1,...,n)$

Резольвента для дизъюнктов $D_i = D_i' \vee P(t_1)$ и $D_j = D_j' \vee \neg P(t_2)$ ищется с помощью унификации термов t_1, t_2: $R\dfrac{D_i, D_j}{D_i' < x \mid t > \vee D_j' < x \mid t >}$.

Формальные теории

Формальная теория (*исчисление*)- это способ изложения логики без приписывания пропозициональным переменным, предикатам и формулам какого-либо значения. По замыслу создателей формальных теорий таким образом можно избежать многих неприятностей, возникающих при использовании в логике человеческого языка, допускающего двусмысленность, недосказанность, переиначивание исходно заложенного значения, смысла и т.д.

Формальная теория - это игра со знаками, игра со словами и предложениями, составленными из знаков. При этом имеется в виду, что правила составления слов из знаков и правила игры со словами и предложениями заранее оговорены, точно и строго прописаны. В основе формальной теории – язык, на котором и разговаривает теория. На первое место выходит синтаксис этого языка, т.е. способ построения формул, в противопоставление семантике.

Имеет место следующее определение формальной теории, или исчисления. Формальная теория Т состоит из следующих компонент:

1. Множества знаков, образующих *алфавит* языка теории.
2. Множества слов, составленных из знаков алфавита, называемых *формулами*.
3. Подмножества формул всего множества формул, называемых *аксиомами*.
4. Множества *правил вывода*, с помощью которых из формул получают формулы.

В язык теории Т входит алфавит и формулы. Количество аксиом может быть конечным или бесконечным. В последнем случае для наглядного представления они задаются с помощью схем. По схемам легко выписываются сами аксиомы. Под логическими аксиомами, как правило, понимают аксиомы *базовой логики* над, которой надстраиваются конкретные теории за счет добавления новых аксиом, отражающих специфику конкретной теории. Такие аксиомы называют *нелогическими*.

Обычно формулы состоят из конечного числа знаков. Но бывают теории с бесконечно длинными формулами, т.е. с формулам, содержащими бесконечное число знаков.

Формальное *доказательство* формулы A в теории Т - это конечная последовательность формул $A_1, A_2, ..., A_n$, где каждая формула A_i является либо аксиомой, либо получена из предыдущих с помощью одного из правил

вывода, а A_n - это формула A. Формула A в этом случае называется *теоремой*, для теорем используется запись $\mapsto A$.

Формальная теория называется *полной*, если для всякого высказывания A имеем: $\mapsto A$ или $\mapsto \neg A$.

По замыслу создателя исчисления предикатов Фреге, любая правильно построенная формула, а точнее высказывание (высказывание в логике предикатов - это *закрытая формула*, т.е. формула, все переменные которой связаны кванторами), должна быть теоремой, т.е. должна быть доказываемым утверждением. Иначе говоря, исчисление высказываний и исчисление предикатов должны быть полными теориями. Ожидания Фреге оправдались. Но, как оказалось, более сильные теории, включающие арифметику, неполны. Это было доказано Гёделем (см. ниже).

Формальная теория называется *непротиворечивой*, если в ней не является доказуемой формула $A\&\neg A$, где A - произвольное высказывание теории.

Смысл условия непротиворечивости в том, что оно может быть доказано средствами и в рамках самой формальной теории. Увы, таким способом, как выяснилось, невозможно установить даже непротиворечивость (формальной) арифметики.

Исчисление высказываний - это следующая формальная теория:

1. Алфавит.

   • Знаки пропозициональных переменных
   • Логические связки $\&\vee\neg\Rightarrow$.
   • Вспомогательные знаки (,).

2. Формулы.

   • Пропозициональная переменная есть (атомарная) формула.
   • Если A и B формулы, то $A\&B$, $A\vee B$, $A\Rightarrow B$ формулы.
   • Если A формула, то $\neg A$ формула.

3. Аксиомы (схемы Клини). Под A и B понимается любая формула исчислениявысказываний.

I

а) $A\Rightarrow(B\Rightarrow A)$

б) $(A\Rightarrow(B\Rightarrow C))\Rightarrow((A\Rightarrow B)\Rightarrow(A\Rightarrow C))$

II

а) $(A\&B)\Rightarrow A$

б) $(A\&B)\Rightarrow B$

в) $A\Rightarrow(B\Rightarrow(A\&B))$

III

а) $A \Rightarrow (A \vee B)$

б) $B \Rightarrow (A \vee B)$

в) $(A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \vee B) \Rightarrow C))$

IV

а) $(A \Rightarrow B) \Rightarrow ((A \Rightarrow \neg B) \Rightarrow \neg A)$

б) $\neg\neg A \Rightarrow A$

4. Правила вывода.

$$\frac{A \Rightarrow B, A}{B} \text{ (modus ponens)}$$

5. Определение доказательства

Пусть Г - множество формул. Формула А выводима из множества гипотез Г, если существует конечная последовательность формул $A_1, A_2, ..., A_n$, где каждая формула A_i является либо аксиомой, либо гипотезой, либо получена из предыдущих с помощью правила вывода и A_n -это формула А.

Последовательность $A_1, A_2, ..., A_n$ - это *вывод* или *доказательство* формулы А из множества гипотез Г. Используется обозначение для вывода: $\Gamma \mapsto A$.

Если $\Gamma = \varnothing$, то вывод называется доказательством формулы А, а сама формула А в называется *теоремой*. Используется символическая запись для теорем: $\mapsto A$.

В исчислении высказываний имеет место *теорема дедукции*: если $\Gamma, A \mapsto B$, то $\Gamma \mapsto A \Rightarrow B$.

Исчисление высказываний - полная теория. Исчисление высказываний - непротиворечивая теория.

Исчисление предикатов - это формальная теория, получаемая за счет добавления к исчислению высказываний новых знаков, понятия терма новых типов формул и новых правил вывода.

1. Дополнения в алфавит.

- Знаки индивидуальных переменных $x_1, x_2, ..., x_n, ...$
- Знаки индивидуальных констант $c_1, c_2, ..., c_n, ...$
- Знаки предикатов $P_1^{(n_1)}\underbrace{(.,...,.)}_{n_1 мест}, P_2^{(n_2)}\underbrace{(.,...,.)}_{n_2 мест}, ...$
- Знаки операций (функции) $f_1^{(n_1)}\underbrace{(.,...,.)}_{n_1 мест}, f_2^{(n_2)}\underbrace{(.,...,.)}_{n_2 мест}, ...$
- Логические знаки (кванторы) $\forall$, $\exists$

2. Термы.

• переменные $x_1, x_2, ..., x_n, ...$ - это термы;

• константы $c_1, c_2, ..., c_n, ...$ - это термы;

• если $f_m^{(n)}\underbrace{(.,...,.)}_{n\text{мест}}$ - n-местный знак операции и $t_1, t_2, ..., t_n$ - термы, то $f_m^{(n)}(t_1, ..., t_n)$ - терм,

3. Формулы.

• если $P_m^{(n)}\underbrace{(.,...,.)}_{n\text{мест}}$ - n-местный знак предиката и $t_1, t_2, ..., t_n$ - термы, то $P_m^{(n)}(t_1, ..., t_n)$ - (атомарная) формула;

• если A и B формулы, то $A \& B$, $A \vee B$, $A \Rightarrow B$ формулы;

• если A формула, то $\neg A$ формула;

• если $A(x)$ формула, содержащая переменную x, то $\forall x A(x)$, $\exists x A(x)$ формулы.

4. Дополнительные аксиомы.

$$\forall x A(x) \Rightarrow A(t)$$
$$A(t) \Rightarrow \exists x A(x)$$

5. Дополнительные правила вывода.

$$\frac{B \Rightarrow A(x)}{B \Rightarrow \forall x A(x)}$$
$$\frac{A(x) \Rightarrow B}{\exists x A(x) \Rightarrow B}$$

Приведенный язык исчисления предикатов образует теорию, которую называют *узким исчислением предикатов*, или *чистым исчислением предикатов*.

Появление в теории аксиом, разрешающих навешивание кванторов по знакам операций или предикатов, приводит к *исчислениям высших порядков*, Исчисление предикатов - это *теория первого порядка*.

Расширение узкого исчисления предикатов за счет добавления новых знаков индивидуальных констант, операций (знаков функций, знаков операций), знаков предикатов, новых аксиом, связывающих новые знаки, и новых правил вывода называется часто *прикладным исчислением предикатов*.

Исчисление предикатов - полная теория. Исчисление предикатов - непротиворечивая теория.

Теория, содержащая исчисление предикатов, называется *эгалитарной*, если она имеет дополнительный двуместный предикат $=(.,.)$, для которого выполняются две нелогические аксиомы (A - произвольная формула):

$$\forall x = (x, x)$$
$$= (x, y) \Rightarrow (A(..., x, ..., y, ...) \Rightarrow A(..., y, ..., x, ...))$$

Вместо префиксной записи $=(x, y)$ пишут в инфиксной форме $x = y$. Таким образом, эгалитарная теория - это просто теория с равенством.

Формальная арифметика - это эгалитарное прикладное исчисление, в котором дополнительно имеются:

1. Предметная константа 0.

2. Двуместные операции $+$ и $\bullet$, а также одноместная операция $'$.

3. Знак равенства $.=.$

4. Нелогические аксиомы равенства (см. выше) и следующие нелогические аксиомы арифметики:

$$(A(0) \,\&\, \forall x(A(x) \Rightarrow A(x'))) \Rightarrow \forall x A(x)$$
$$(t_1' = t_2') \Rightarrow (t_1 = t_2)$$
$$\neg(t_1' = 0)$$
$$(t_1 = t_2) \Rightarrow ((t_2 = t_3) \Rightarrow (t_1 = t_3))$$
$$(t_1 = t_2) \Rightarrow (t_1' = t_2')$$
$$t + 0 = t$$
$$(t_1 + t_2') = (t_1 + t_2)'$$
$$t \bullet 0 = 0$$
$$t_1 \bullet t_2' \Rightarrow t_1 \bullet t_2 + t_1,$$

где A - любая формула, а. t, t_1 и t_2 - любые термы.

Первая аксиома - это известный способ доказательства посредством *математической индукции*. Если вместо t' писать $t+1$, то ясно, что t' - это следующее натуральное число, идущее за t. Другими словами, аксиомы арифметики определяют натуральные числа и правила оперирования с ними с помощью операций сложения и умножения.

Метод математической индукции, который входит в качестве аксиомы в формальную арифметику может быть усилен за счет расширения области его применения до так называемых *трансфинитных чисел*, которые, как видно из их названия, «идут следом» за финитными, т.е. натуральными числами. Получается более мощный способ доказательства теорем, названный *методом трансфинитной индукции*.

Непротиворечивость формальной арифметики доказывается в более широкой формальной теории, содержащей арифметику и принцип трансфинитной индукции.

Теорема Гёделя о неполноте дедуктивных систем: во всякой теории первого порядка, включающей формальную арифметику:

1) существует такая (истинная) формула A, что ни A, ни $\neg A$ не являются доказуемыми;
2) утверждение о непротиворечивости этой теории - это формула данной теории, которая не является доказуемой.

Теорема Гёделя говорит о том: что в любой достаточно богатой теории существуют высказывания, которые воспринимаются как истинные и разумные, но тем не менее они не могут быть обоснованы, доказаны теми средствами, которые теория предоставляет исследователю. По существу, теорема утверждает, что в любом мире есть вещи, познание которых требует выхода в более обширный, высший мир. Над каждой теорией нужно надстраивать более изощренную метатеорию, над метатеорией - метаметатеорию и т.д.. Собственно говоря, так вышло с доказательством непротиворечивости формальной арифметики (см. выше).

Кроме того, теорема Гёделя говорит о несостоятельности идеи полной, формализации процесса логического вывода. Иначе говоря, не все может быть формализовано, как бы этого ни хотелось математикам.

Машина Тьюринга-Поста

Алгоритм - это раз и навсегда составленная программа, в которой все заранее предусмотрено. Выражаясь популярно, алгоритм - это точное, воспроизводимое, поддающееся исполнению предписание, определяющее - шаг за шагом - каким путем надлежит решать данную задачу. Алгоритмом является любое формализованное доказательство математической теоремы, равно как и программа цифровой машины, переводящей с одного языка на другой.

Алгоритмом принято называть систему вычислений, которая для некоторого класса математических задач из записи А «условий» задачи позволяет при помощи однозначно определенной последовательности операций, совершаемых «механически», без вмешательства творческих способностей человека, получить запись В «решений» задали.

Таким образом, алгоритм - это процедура Ψ, которая:

1) Применяется к строго определенным исходным данным А. Ее нельзя применять к другому типу данных, она либо будет не способна с ними что-то сделать, либо может выдать бессмысленный результат, т.е. результат, не поддающийся анализу с точки зрения тех ожиданий, которые связывались с алгоритмом при его создании.

2) Расчленена на отдельные простые шаги; каждый шаг состоит в непосредственной обработке возникшего к этому шагу состояния S в состояние $S^* = \Omega_\Psi(S)$.

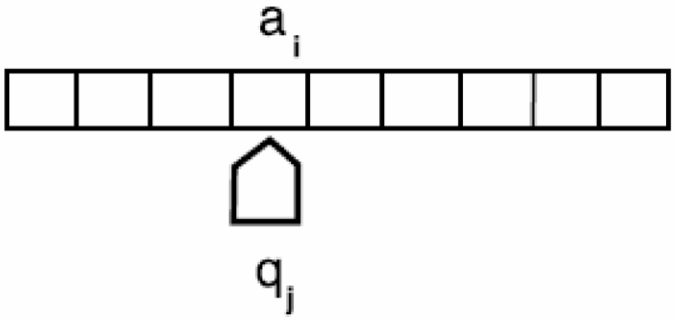
3) Непосредственная переработка S в $S^* = \Omega_\Psi(S)$ производится однозначно заданным способом лишь на основании информации о виде заранее ограниченной «активной» части состояния S и затрагивает лишь эту активную часть.

4) Процесс переработки $A_0 = A$ в $A_1 = \Omega_\Psi(A_0)$, A_1 в $A_2 = \Omega_\Psi(A_1)$ и т.д. продолжается до тех пор, пока либо не произойдет безрезультатная остановка (или оператор Ω_Ψ не определен для возникшего состояния), либо не появится сигнал о получении «решения». При этом не исключается возможность непрекращающегося процесса переработки.

Каждый алгоритм задает функцию, поскольку по набору исходных данных выдает результат применения алгоритма к этим данным. Естественно назвать функцию, значения которой могут находиться с помощью некоторого алгоритма, - *вычислимой функцией*. Таким образом, вычислимая функция - это такая функция, для которой существует вычисляющий ее значения алгоритм.

Машины Тьюринга-Поста - это пример алгоритма. Придуман этот алгоритм независимо Аланом Тьюрингом и Эмилем Постом. Машина Тьюринга-Поста Т состоит из следующих «частей»:

1. Ленты 1 разбитой на конечное число ячеек.
2. Внешней памяти, принимающей одно из состояний входящих в множество $A = \{a_0, a_1, ..., a_m\}$. Ячейки ленты находятся в одном и только в одном из состояний из множества А. Состояние a_0 называется пустым.
3. Внутренней памяти, принимающей одно из состояний; входящих в множество $Q = \{q_0, q_1, ..., q_n\}$. Состояние q_0 называется «стоп»
4. Головки, двигающейся вдоль ленты и считывающей содержимое ячейки, напротив которой она останавливается.
5. Механического устройства, передвигающего головку и меняющего состояния внешней и внутренней памяти. Если головка, в состоянии q стоит напротив ячейки с номером k, то изменения состояния внутренней памяти и состояния ячейки происходят одновременно.



Работа машины Тьюринга-Поста Т осуществляется посредством команд, которые выполняет механическое устройство, Команда имеет один из следующих трех возможных видов: $q_s a_i \to q_t a_j$, $q_s a_i \to q_t L$, $q_s a_i \to q_t R$, где L - это движение головки влево на одну ячейку, а R - вправо. При этом всегда самый левый символ в записи команды $q_s \neq q_0$.

Смысл команд таков: если головка в состоянии q_s обозревает ячейку в состоянии a_i, то в первом случае она меняет свое состояние на q_t, а ячейки на a_j, во втором случае - свое состояние на q_t и сдвигается влево и, наконец, в третьем - вправо.

Конечный набор команд образует *программу*.

Состояние машины Т - это последовательность состояний $a_{i_1}, ..., a_{i_r}$ ячеек ленты, состояние внутренней памяти q_s головки и номер k воспринимаемой (читаемой) ячейки в состоянии a_{i_k}.

Состояние машины Т записываем в виде $a_{i_1} ... a_{i_{k-1}} q_s a_{i_k} ... a_{i_r}$ и называем *машинным словом* М в алфавите $A \cup Q$. Символ q_s может быть самым левым,

но не может быть самым правым в машинном слове, так как справа от него должно быть считываемое состояние ячейки.

Под воздействием программы происходит изменение состояния машины, сопровождающееся переделкой исходного машинного слова $\text{М} \to \text{М}^{(1)} \to ... \to \text{М}^{(p)}$. В этом случае будем говорить, что машина Т перерабатывает слово М в слово $\text{М}^{(p)}$, и обозначать этот факт $\text{М}^{(p)} = T(\text{М})$. Запись $T(\text{М})$ можно употреблять и в другом смысле – просто как обозначение машины Т с исходными данными М.

В теории алгоритмов имеет место *тезис Тьюринга*: все вычислимые функции вычисляются на машинах Тъюринга-Поста,

Машина Тьюринга-Поста называется *универсальной*, если она может при определенных начальных входных данных вычислить любую функцию, которая вычислима на какой-либо машине Тьюринга-Поста.

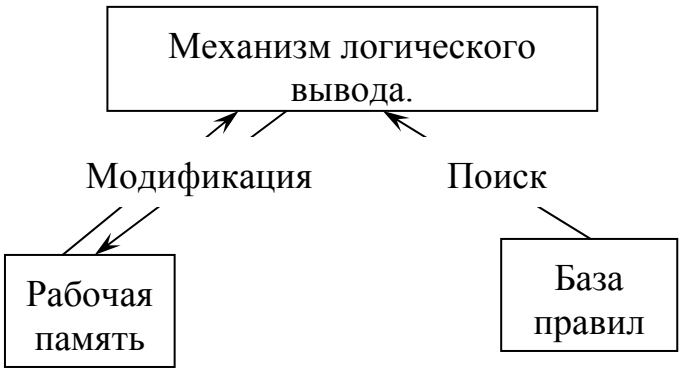
Иначе говоря, с учетом тезиса Тьюринга можно сказать, что универсальная машина Тьюринга-Поста способна вычислить любую вычислимую функцию. Доказано, что универсальная машина Тьюринга-Поста существует.

Существование универсальной машины Тьюринга означает, что систему команд любой машины Т можно интерпретировать двояким образом: либо как описание работы конкретного устройства машины Т, либо как программу для универсальной машины U. Для современного инженера, проектирующего систему управления, это обстоятельство вполне естественно. Он хорошо знает, что любой алгоритм управления может быть реализован либо аппаратно — построением соответствующей схемы, либо программно — написанием программы для универсального управляющего компьютера.

Однако важно сознавать, что сама идея универсального алгоритмического устройства совершенно не связана с развитием современных технических средств его реализации (электроники, физики твердого тела и т. д.) и является не техническим, а математическим фактом, описываемым в абстрактных математических терминах, не зависящих от технических средств, и к тому же опирающимся на крайне малое количество весьма элементарных исходных понятий. Характерно, что основополагающие работы по теории алгоритмов (и, в частности, работы Тьюринга) появились в 30-х гг. XX в, до создания современных компьютеров.

Продукционная экспертная система

Продукционная система состоит из трех основных компонентов, схематично изображенных на рисунке ниже. Первый из них — это база правил типа ЕСЛИ (условие), ТО (действие): «ЕСЛИ холодно, ТО надеть шубу»; «ЕСЛИ идет дождь, ТО взять зонтик», и т.п. Вторым компонентом является *рабочая память*, в которой хранятся исходные данные к задаче и выводы, полученные в ходе работы системы. Третий компонент — *механизм логического вывода*, использующий правила в соответствии с содержимым рабочей памяти.



Блок-схема продукционной системы

Рассмотрим конкретный пример работы подобной продукционной системы. В базе правил экспертной системы имеются два правила.

Правило 1: ЕСЛИ «намерение — отдых» И «дорога ухабистая», ТО «использовать джип».

Правило 2: ЕСЛИ «место отдыха — горы», ТО «дорога ухабистая».

Допустим, что в рабочую память поступили исходные данные: «намерение — отдых»; «место отдыха — горы».

Механизм вывода построен на основе правила «modus ponens»: $\frac{A, A \Rightarrow B}{B}$. Он начинает сопоставлять *образцы* (факты) A из условных частей правил $A \Rightarrow B$ с образцами (фактами) A', хранимыми в рабочей памяти. Если образцы из условной части имеются в рабочей памяти (A=A'), то условная часть считается истинной (правило срабатывает), в противном случае — ложной. Если условная часть истинна, то образец B из заключительной части правила помещается в рабочую память и это правило больше не рассматривается.

В данном примере при рассмотрении правила 1 оказывается, что образец «намерение — отдых» имеется в рабочей памяти, а образец «дорога ухабистая» отсутствует, поэтому условная часть правила 1 считается ложной. При рассмотрении правила 2 выясняется, что его условная часть истинна. Механизм вывода выполняет заключительную часть этого правила, и образец

«дорога ухабистая» заносится в рабочую память. Правило 2 при этом выбывает из числа кандидатов на рассмотрение.

Снова рассматривается правило 1, условная часть которого теперь становится истинной, и содержимое рабочей памяти пополняется образцом «использовать джип». В итоге правил, которые можно было бы применять, не остается и система останавливается.

В рассмотренном примере приведен *прямой вывод* — от данных к поиску цели. Однако применяют и *обратный вывод* — от цели для ее подтверждения к данным. Продемонстрируем этот способ на нашем примере. Допустим, что наряду с исходными данными «намерения — отдых»; «место отдыха — горы» имеется цель «использовать джип».

Согласно правилу 1 для достижения этой цели требуется выполнение условия «дорога ухабистая», поэтому условие становится новой целью. При рассмотрении правила 2 оказывается, что условная часть этого правила в данный момент истинна, поэтому рабочая память пополняется образцом «дорога ухабистая». При повторном рассмотрении правила 1 подтверждается цель «использовать джип».

При обратном выводе система останавливается в двух случаях: либо достигается первоначальная цель, либо кончаются правила (ни одно из правил нельзя применить). При прямом выводе система останавливается только тогда, когда кончаются правила, либо при появлении в рабочей памяти специально предусмотренного образца, например, «использовать джип».

В приведенном примере на каждом этапе прямого вывода можно было использовать только одно правило. В общем же случае на каждом этапе вывода таких правил несколько, и тут возникает *проблема выбора*. Например, введем в рассмотрение еще одно правило.

Правило 3: ЕСЛИ «намерение — отдых», ТО «нужна скорость».

Кроме того, введем условие останова системы — появление в рабочей памяти образца «использовать джип».

Теперь на первом этапе прямого вывода появляется возможность применять либо правило 2, либо правило 3. Если сначала применить правило 2, то на следующем этапе можно будет применять правило 1 и правило 3. Если на этом этапе применить правило 1, то выполнится условие останова системы, но если прежде применить правило 3, то потребуется еще один этап вывода.

Этот пример показывает, что выбор применяемого правила оказывает прямое влияние на эффективность вывода. В реальной системе, где имеется множество правил, появляется проблема их оптимального выбора.

Если на каждом этапе логического вывода существует множество применимых правил, то это множество носит название *конфликтного набора*, а выбор одного из них называется *разрешением конфликта*.

Аналогичная ситуация возникает и при обратном выводе. Например, дополним предыдущий пример еще одним правилом.

Правило 4: ЕСЛИ «место отдыха — пляж», ТО «дорога ухабистая».

Если на основании этого условия подтверждается цель «использовать джип», то для достижения первоначальной цели достаточно применить только одно правило 1, однако, чтобы подтвердить новую цель «дорога ухабистая», открывается возможность применения правила 1, нужно использовать либо правило 2, либо правило 4. Если сначала применить правило 2, то это будет самый удачный выбор, поскольку сразу же можно применить и правило 1. С другой стороны, если попытаться применить правило 2, то, поскольку образца «место отдыха — пляж», который является условием правила 4, в рабочей памяти не существует и, кроме того, не существует правила, подтверждающего его, данный выбор является неудачным. И лишь со второго захода, применяя правило 2, можно подтвердить цель «дорога ухабистая».

Следует обратить внимание на то, что при обратном выводе правило 3, которое не оказывает прямого влияния на достижение цели, не принималось в расчет с самого начала. Таким образом, для обратных выводов характерна тенденция исключения из рассмотрения правил, не имеющих прямого отношения к заданной цели, что позволяет повысить эффективность вывода.

Продукционная модель — это наиболее часто используемый способ представления знаний в современных экспертных системах. Основными преимуществами продукционной модели являются наглядность, высокая модульность, легкость внесения изменений и дополнений, простота механизма логического вывода. Экспертные системы и продукционные модели более подробно рассматриваются в рамках дисциплины «Системы искусственного интеллекта».