

Часть I

Логический подход к автоматическому решению задач

Формализация понятия задачи при разработке систем автоматического решения задач часто приводит к необходимости использования логического языка для представления обрабатываемой информации, а также связанной с этим языком формальной дедуктивной системы, определяющей допустимые процессы логического вывода. Возникающие здесь математические модели мы проиллюстрируем на двух типичных примерах - языке и исчислении логики высказываний, а также языке и исчислении логики предикатов.

Часть II

1. Язык логики высказываний

При задании формального логического языка обычно следуют следующей схеме:

- a) Указывается конечный либо бесконечный набор символов, образующих алфавит языка; при описании алфавита могут вводиться те или иные характеристики его символов, в частности определяться разбиения их на подклассы.
- б) Вводится индуктивное описание множества правильных выражений языка - конечных последовательностей символов алфавита.
- в) Индуктивным образом определяется интерпретация языка (либо множеств допустимых интерпретаций), сопоставляющая каждому выражению языка обозначаемую им функцию, либо объект из некоторой "области интерпретации".

Пункты а) и б) задают синтаксис логического языка, пункт в) - его семантику

В случае языка логики высказываний эта общая схема реализуется следующим образом:

- а) Алфавит языка логики высказываний состоит из счётного списка переменных x_1, x_2, \dots ; двух логических констант И, Л; заданного набора логических связок (например, связок $\vee, \&, \neg, \rightarrow, \leftrightarrow$), а также скобок (,)
- б) Правильные выражения языка логики высказываний, называемые формулами логики высказываний, вводятся при помощи следующего определения:
 - 1) Однобуквенное слово, состоящее из переменной, есть формула логики высказываний.
 - 2) Если слова A и B суть формулы логики высказываний, то слова $(\neg A)$, $(A \vee B)$, $(A \& B)$, $(A \rightarrow B)$, $(A \leftrightarrow B)$ суть формулы логики высказываний.

в) При задании интерпретации формул логики высказываний возможны два различных подхода - эти формулы могут рассматриваться либо как обозначения функции алгебры логики, либо как обозначения единичных высказываний, имеющих определённое истинностное значение. В обоих случаях определяемый формулой объект (функция либо истинностное значение) вводится индуктивным образом, по шагам построения формулы, с использованием хорошо известных таблиц истинности для основных логических связок. В первом случае базис индукции сопоставляет каждой однобуквенной формуле x_i тождественную функцию; во втором случае - некоторое конкретное значение из множества И, Л.

Для большей определённости будем ниже рассматривать второй случай. Заметим, что при этом возникает множество различных интерпретаций языка логики высказываний, отличающиеся друг от друга лишь сопоставлением переменным алфавита x_1, X_2, \dots различных истинностных констант.

Введём для логики высказываний несколько общих понятий, связанных с интерпретацией формул и являющихся типичными для логических языков.

Моделью формулы F будем называть произвольную интерпретацию языка логики высказываний в которой F имеет значение И. Формулу имеющую хотя бы одну модель назовём **выполнимой**; формулу, для которой каждая интерпретация языка логики высказываний является моделью, назовём **общезначимой**. Описание класса всех общезначимых формул, является одной из наиболее важных задач, возникающих при изучении логических языков, и для него обычно используются следующие подходы:

1) Нахождение индуктивного описания класса всех общезначимых формул. Такое описание определяет в базисе индукции некоторое подмножество общезначимых формул (конечное либо бесконечное), называемых аксиомами, и указывает в индуктивном шаге перечень допустимых правил вывода, позволяющих получать новые общезначимые формулы исходя из ранее найденных. Логический язык вместе с индуктивным описанием такого рода образует **дедуктивную систему**.

2) Нахождение алгоритма, перечисляющего все общезначимые формулы - фактически, обеспечивается при построении дедуктивной системы.

3) Нахождение алгоритма, распознающего общезначимые формулы в классе всех правильных выражений языка. В отличие от пунктов 1 и 2, описание такого типа удаётся найти лишь для весьма немногих логических языков, к числу которых относится и язык логики высказываний.

Существует много различных дедуктивных систем для описания общезначимых формул логики высказываний, отличающихся множеством используемых логических связок и выбором аксиом; приведём здесь одну из простейших таких систем в которой рассматриваются лишь две логические связки - \neg , \rightarrow . Остальные логические связки легко могут быть выражены через них и трактоваться как своего рода "сокращённые обозначения" для соответствующих формул с \neg и \rightarrow . Аксиомы данной дедуктивной системы задаются при помощи так называемых "схем аксиом":

A1) Если f, g - формулы, то формула $(f \rightarrow (g \rightarrow f))$ есть аксиома.

A2) Если f, g, h - формулы, то формула $((f \rightarrow (g \rightarrow h)) \rightarrow ((f \rightarrow g) \rightarrow (f \rightarrow h)))$ есть аксиома.

A3) Если f, g - формулы, то формула $((\neg f \rightarrow \neg g) \rightarrow ((\neg f \rightarrow g) \rightarrow f))$ есть аксиома.

В действительности каждая из схем аксиом A1, A2, A3 определяет бесконечное множество аксиом, отличающихся выбором формул f, g, h .

Правило вывода в этой дедуктивной системе одно - если уже получены общезначимые формулы f и $(f \rightarrow g)$, то из них выводится общезначимая формула g (согласно классификации предложенной ещё Аристотелем, это правило называется *modus ponens*).

Чтобы проиллюстрировать технику, применяемую при изучении дедуктивных систем, приведём схематические наброски доказательств некоторых важных свойств только, что определённой дедуктивной системы.

Выводом формулы f из списка формул $\Gamma = g_1, \dots, g_n$ будем называть произвольную последовательность формул $f_1, f_2, \dots, f_m = f$, такую, что каждое $f_i (i = 1, \dots, m)$ есть либо аксиома, либо элемент списка Γ , либо получено по правилу *modus ponens* из некоторых f_j, f_k , при $j, k \in \{1, \dots, i-1\}$. Если существует вывод f из Γ , то обозначаем это обстоятельство посредством $\Gamma \vdash f$ (в частности, список Γ может быть пуст; те формулы, которые выводимы из пустого Γ , как легко можно видеть, образуют класс всех выводимых формул нашей дедуктивной системы). В качестве примера вывода из пустого Γ (или, короче, вывода) приведём следующую последовательность формул:

$$(f \rightarrow ((f \rightarrow f) \rightarrow f)) \rightarrow ((f \rightarrow (f \rightarrow f)) \rightarrow (f \rightarrow f));$$

$$(f \rightarrow ((f \rightarrow f) \rightarrow f)); (f \rightarrow (f \rightarrow f)) \rightarrow (f \rightarrow f);$$

$$f \rightarrow (f \rightarrow f); (f \rightarrow f).$$

Первый её элемент - аксиома, возникающая по схеме аксиом A2; второй и четвёртый - аксиомы по A1; третий и пятый - получены применением правила вывода.

Имеет место следующее важное утверждение, известное как теорема дедукции (Эрбран):

Утверждение 1. Если Γ - список формул и f, g - формулы, то из $\Gamma, f \vdash g$ вытекает $\Gamma \vdash (f \rightarrow g)$

Доказательство данного утверждения легко получается индукцией по длине n вывода $g_1, \dots, g_n = g$ формулы g из Γ, f . Если $n = 1$, то g - либо аксиома, либо элемент списка Γ, f . В первом случае вывод $(f \rightarrow g)$ из Γ имеет вид $(g \rightarrow (f \rightarrow g)), g, (f \rightarrow g)$. Во втором случае возможны два подслучаи - если $g \in \Gamma$, то снова берём указанный выше вывод; если же $g = f$, то используем приведённый выше вывод $(f \rightarrow f)$ из пустого списка. Если $n > 1$ и g - аксиома, либо элемент списка Γ, f , то поступаем так же как выше. Наконец если g получено по правилу *modus ponens* из формул g_i, g_j , ($i, j <$

n), то согласно предположению индукции имеем $\Gamma \vdash (f \rightarrow g_i)$, $\Gamma \vdash (f \rightarrow g_j)$. Заметим, что одна из формул g_i, g_j , например g_j , должна иметь вид $(g_i \rightarrow g)$. Для получения вывода $(f \rightarrow g)$ из Γ теперь достаточно рассмотреть последовательность образованную расположенным подряд выводами $(f \rightarrow g_i)$, $(f \rightarrow g_j)$ из Γ , и присоединить к ней в конце формулы $((f \rightarrow (g_i \rightarrow g)) \rightarrow ((f \rightarrow g_i) \rightarrow (f \rightarrow g)))$, $((f \rightarrow g_i) \rightarrow (f \rightarrow g))$, $(f \rightarrow g)$. Первая из этих формул есть аксиома полученная по схеме аксиом А2; две последние получены применением правила вывода.

Для доказательства того, что каждая общезначимая формула алгебры логики выводима в рассматриваемой нами дедуктивной системе, нам понадобится следующее вспомогательное утверждение:

Утверждение 2. Пусть f -формула логики высказываний; x_1, \dots, x_n - все переменные входящие в f , и I - некоторая интерпретация языка логики высказываний. Для произвольной формулы g обозначим посредством $(g)_I$ формулу, совпадающую с g , если значение g в интерпретации I есть И, и совпадающую с $(\neg g)$ в противном случае. Тогда выполняется $(x_1)_I, \dots, (x_n)_I \vdash (f)_I$.

Доказательство этого утверждения поведём индукцией по числу логических связок в f . Если $m = 0$, то f совпадает с x_1 , и утверждение сводится к очевидному факту $(x_1)_I \vdash (x_1)_I$. Пусть $m > 0$; рассмотрим следующие два случая:

1) f имеет вид $(\neg h)$. Если $(h)_I = h$, то $(f)_I = \neg \neg h$, и для установления истинности утверждения достаточно доказать выводимость формулы $h \rightarrow \neg \neg h$. Если же $(h)_I = (\neg h)$, то $(f)_I = f = (\neg h)$, и утверждение сводится к предположению индукции.

2) f имеет вид $(h_1 \rightarrow h_2)$. Если $(h_1)_I = \neg h_1$ либо $(h_2)_I = \neg h_2$, то $(f)_I = f$; в первом случае для извлечения $(x_1)_I, \dots, (x_n)_I \vdash f$ из $(x_1)_I, \dots, (x_n)_I \vdash \neg h_1$ достаточно установить выводимость формулы $\neg h_1 \rightarrow (h_1 \rightarrow h_2)$, а во втором воспользоваться аксиомой $h_2 \rightarrow (h_1 \rightarrow h_2)$ и предположением индукции $(x_1)_I, \dots, (x_n)_I \vdash h_2$. Наконец в случае $(h_1)_I = h_1$ и $(h_2)_I = \neg h_2$ имеем $(f)_I = \neg f$; здесь достаточно установить выводимость формулы $(h_1 \rightarrow (\neg h_2 \rightarrow \neg(h_1 \rightarrow h_2)))$ и воспользоваться индуктивными предположениями $(x_1)_I, \dots, (x_n)_I \vdash h_1$ и $(x_1)_I, \dots, (x_n)_I \vdash \neg h_2$.

Выводимость формул $h \rightarrow \neg \neg h$, $\neg h_1 \rightarrow (h_1 \rightarrow h_2)$ и $(h_1 \rightarrow (\neg h_2 \rightarrow \neg(h_1 \rightarrow h_2)))$, которыми мы пользовались при доказательстве утверждения, может быть достаточно легко установлена с помощью теоремы дедукции, это рекомендуется проделать в качестве самостоятельного упражнения.

Утверждение 3. (Теорема о полоноте для исчисления высказываний). Каждая общезначимая формула алгебры логики является выводимой в рассматриваемой дедуктивной системе.

Будем обозначать посредством f^σ , где $\sigma \in \{\text{И}, \text{Л}\}$, формулу f в случае $\sigma = \text{И}$, и формулу $(\neg f)$ в случае $\sigma = \text{Л}$. Пусть f произвольная общезначимая формула алгебры логики, и x_1, \dots, x_n - все входящие в неё переменные. Покажем, что для любого целого неотрицательного k , $k \leq n$, и любого набора $(\sigma_1, \dots, \sigma_k)$ логических констант И, Л выполняется $x_1^{\sigma_1}, \dots, x_k^{\sigma_k} \vdash f$. При $k = n$ это является очевидным следствием общезначимости f и утвержде-

ния 2. Если данное утверждение уже доказано для некоторого k , $k > 0$, то рассмотрим произвольный набор $(\sigma_1, \dots, \sigma_{k-1})$ констант И, Л. По предположению индукции, имеем $x_1^{\sigma_1}, \dots, x_{k-1}^{\sigma_{k-1}}, x_k \vdash f$ и $x_1^{\sigma_1}, \dots, x_{k-1}^{\sigma_{k-1}}, \neg x_k \vdash f$. По теореме дедукции находим отсюда $x_1^{\sigma_1}, \dots, x_{k-1}^{\sigma_{k-1}} \vdash x_k \rightarrow f$ и $x_1^{\sigma_1}, \dots, x_{k-1}^{\sigma_{k-1}} \vdash \neg x_k \rightarrow f$. Далее достаточно установить выводимость формулы $(x_k \rightarrow f) \rightarrow ((\neg x_k \rightarrow f) \rightarrow f)$ (что рекомендуется в качестве самостоятельного упражнения), и воспользоваться дважды правилом *modus ponens*, после чего будем иметь $x_1^{\sigma_1}, \dots, x_{k-1}^{\sigma_{k-1}} \vdash f$. Шаг индукции, таким образом, доказан, и при $k = 0$ окончательно получим $\vdash f$.

В случае логики высказываний проверка общезначимости формулы, имеющей n переменных, может быть осуществлена путём перебора всех 2^n возможных наборов логических констант, сопоставляемых этим переменным при различных интерпретациях, и установлении того, что все значения формулы на этих наборах равны И. Таким образом здесь имеется не только алгоритм перечисления всех общезначимых формул, но и алгоритм проверки общезначимости. Однако даже для такого простейшего случая остаётся проблема уменьшения трудоёмкости распознавания общезначимости по сравнению с практически малоприемлемым для сколь-нибудь значительных n перебором всех 2^n наборов значений переменных. К проблеме распознавания общезначимости формул логики высказываний тесно примыкает проблема решения систем логических уравнений сводящаяся к нахождению всех наборов значений переменных при которых заданная формула алгебры логики принимает значение И. Последняя проблема часто встречается в различных прикладных задачах дискретной оптимизации и диагностики, и поиску эффективных процедур решения её, учитывающих в своих эвристических решающих правилах статистические особенности рассматриваемого класса задач, посвящено большое количество исследований. Мы ограничимся здесь лишь несколькими простейшими процедурами подобного рода (применительно к задаче распознавания общезначимости), поскольку они позволяют нам развить технику доказательства теорем в логике высказываний, представляющую собой упрощённый аналог техники, развиваемой далее для автоматического доказательства теорем в логике предикатов.

Прежде всего, опишем алгоритм Куайна, осуществляющий проверку общезначимости формулы f логики высказываний при помощи построения так называемого семантического дерева. Корню этого дерева - исходной вершине - приписывается формула f . Пусть уже построено некоторое подмножество вершин семантического дерева, каждой из которых сопоставлена некоторая формула. Если каждой концевой вершине данного дерева оказалась сопоставлена константа И, то процесс завершается, и формула f является общезначимой. Если некоторой концевой вершине дерева сопоставлена константа Л, то формула f не общезначима, и процесс также завершается. В противном случае найдётся концевая вершина v , которой сопоставлена формула отличная от констант И, Л. если эта формула g не содержит переменных, то её можно преобразовать в логическую константу, используя тождества:

$\neg I = L; \neg L = I; I \vee x = I; L \vee x = x; I \& x = x; L \& x = L; I \rightarrow x = x;$
 $L \rightarrow x = I; x \rightarrow I = I; x \rightarrow L = \neg x; I \leftrightarrow x = x; L \leftrightarrow x = \neg x.$

Если же она содержит хотя бы одну переменную x , и выполняются следующие действия:

- a) Находятся результаты g_1 и g_2 подстановки в формулу g вместо переменной x , соответственно констант И и Л.
- б) Находятся результаты g'_1 и g'_2 упрощения формул g_1 и g_2 при помощи перечисленных выше тождеств, применяемых до тех пор пока это возможно.
- в) вводятся две новые вершины v_1 и v_2 семантического дерева, которым приписываются, соответственно, формулы g'_1 и g'_2 . К этим вершинам от вершины v проводятся ребра, отмеченные соответственно формулами x и $\neg x$.

Далее повторяется описанный выше процесс рассмотрения концевых вершин семантического дерева.

В данной процедуре остался недоопределённым выбор конкретной переменной x формулы g , по которой проводится "разбор случаев". Этот выбор в прикладных программах, решающих системы логических уравнений путём построения семантических деревьев, осуществляется на основании различных эвристических решающих правил. Простешим таким правилом является выбор переменной, имеющей наибольшее число вхождений в g , для получения наиболее сильного упрощения при переходе к g'_1 и g'_2 , другим полезным соображением является такой выбор переменных x , при котором сопоставленные концевым вершинам семантического дерева формулы g оказывались бы представимы, например, как $h_1 \& h_2$ либо $h_1 \vee h_2$, где формулы h_1 и h_2 не имеют общих переменных. В этом случае вместо построения ветви дерева для $h_1 \& h_2$ ($h_1 \vee h_2$, $h_1 \rightarrow h_2$ и т.п.) можно было бы рассмотреть независимо формируемые деревья для h_1 , h_2 и из анализа их концевых вершин сделать вывод относительно общезначимости g .

В прикладных ситуациях логические языки редко используются во всём многообразии своих синтаксических возможностей. Обычно их формулы приводятся эквивалентными преобразованиями к тому или иному "стандартному виду", который и сохраняется в процессе обработки информации. С точки зрения логики высказываний, наиболее типичной логической стандартной формой, отражающей тенденцию к описанию ситуаций в виде конъюнкции условий, является конъюнктивная нормальная форма. Формула, представленная, в этой форме, имеет вид $A_1 \& \dots \& A_n$, $n \geq 1$, где каждое A_i ($i = 1, \dots, n$) - формула вида $(B_{i1} \vee \dots \vee B_{im_i})$, где $m_i \geq 1$, причём $(B_{i1}, \dots, B_{im_i})$ - переменные или отрицания переменных. Более точно, для определения конъюнктивной нормальной формы введём сначала понятие **дизъюнкта**. Если x_1, \dots, x_m - различные переменные, $m \geq 1$, то формулу вида $(x_1^{\sigma_1} \vee \dots \vee x_m^{\sigma_m})$ назовём дизъюнктом. Логическую константу L по определению так же считаем дизъюнктом. Если A_1, \dots, A_n - различные дизъюнкты ($n \geq 1$), то формула $(A_1 \& \dots \& A_n)$ называется конъюнктивной нормальной формой.

Произвольную формулу логики высказываний можно преобразовать к

виду конъюнктивной нормальной формы, используя следующие эквивалентные преобразования:

- а) логические связки \rightarrow , \leftrightarrow устраняются при помощи тождеств $(a \rightarrow b) = (\neg a \vee b)$; $(a \leftrightarrow b) = (a \& b \vee \neg a \& \neg b)$.
- б) отрицания в формуле "опускаются до переменных" при помощи тождеств $\neg(\neg a) = a$; $\neg(a \vee b) = \neg a \& \neg b$; $\neg(a \& b) = \neg a \vee \neg b$.
- в) применяются преобразования дистрибутивности: $(a \vee (b \& c)) = (a \vee b) \& (a \vee c)$.
- г) устраняются повторные вхождения переменных в дизъюнктивных подформулах, а также константы И, Л (если сама формула не есть И или Л): $a \vee a = a$; $a \vee \neg a = \text{И}$; $a \vee \text{И} = \text{И}$; $a \& \text{И} = a$; $a \& \text{Л} = \text{Л}$.
- д) устраняются одинаковые дизъюнкты: $a \& a = a$.

В процедурах автоматического доказательства теорем часто применяется метод доказательства "от противного". В этом случае для доказательства общезначимости формулы f переходят к рассмотрению формулы $\neg f$ и пытаются доказать её невыполнимость. Опишем модифицированный алгоритм Куайна, предназначенный для установления невыполнимости формулы логики высказываний g , преобразованной к виду конъюнктивной нормальной формы. Вершинам семантического дерева в этом случае будут сопоставляться не формулы, а их множества дизъюнктов. Если некоторой концевой вершине v сопоставлено множество дизъюнктов S , $S \neq$ (пустое множество соответствует логической константе И), то либо S состоит из единственного дизъюнкта Л, либо в S входят дизъюнкты с переменными. В последнем случае выбираем одну из таких переменных x и разбиваем S на три подкласса: S_1 - все дизъюнкты, в которые x входит без внешнего отрицания, S_2 - все дизъюнкты, в которые x входит с отрицанием, S_3 - все дизъюнкты в которых x не встречается. Далее рассматриваем множество дизъюнктов $S_x = \{d | d \vee \neg x \in S_2\}$ (если $\neg x \in S_2$, то здесь берётся $d = \text{Л}$) и множество дизъюнктов и множество дизъюнктов $S_{\neg x} = \{d | d \vee x \in S_1\}$. Нетрудно видеть, что невыполнимость конъюнкции дизъюнктов списка S эквивалента одновременной невыполнимости конъюнкций дизъюнктов списков $S_x \cup S_3$ и $S_{\neg x} \cup S_3$. Поэтому для продолжения построения семантического дерева вводим две новых вершины v_1 и v_2 , которым сопоставляем, соответственно, списки дизъюнктов $S_x \cup S_3$ и $S_{\neg x} \cup S_3$, причём проводим к v_1 и v_2 рёбра от v , отмеченные формулами x и $\neg x$. Построение семантического дерева обрывается как только всем его концевым вершинам оказываются сопоставлены одноэлементные списки, состоящие из дизъюнкта Л. Это означает, что рассматриваемая конъюнктивная нормальная форма невыполнима. Наоборот, появление в некоторой концевой вершине пустого списка дизъюнктов означает её выполнимость.

Другой алгоритм распознавания невыполнимости конъюнктивной нормальной формы связан с рассмотрением специальной дедуктивной системы, использующей в качестве правила вывода так называемое **правило резолюции**. В этом случае конъюнктивная нормальная форма снова представляется как множество своих дизъюнктов, эти дизъюнкты и образуют перечень аксиом данной дедуктивной системы. Правило резолюции, будучи

применено к двум дизъюнктам $A \vee B$ и $\neg A \vee C$ создаёт в качестве следствия новый дизъюнкт $B \vee C$ (здесь возможны вырожденные случаи отсутствия B либо C , если они оба отсутствуют, то результатом служит дизъюнкт \top). Достаточность применения данного правила вывода для распознавания невыполнимости гарантируется следующим утверждением:

Утверждение 4. Конъюнкция конечного семейства дизъюнктов S невыполнима тогда и только тогда, когда за конечное применение правила резолюции из S выводится дизъюнкт \top .

Доказательство поведём индукцией по числу n переменных, встречающихся в дизъюнктах из S . Если $n = 0$, то утверждение очевидно, так как в этом случае непустое семейство дизъюнктов может содержать только логическую константу \top . Пусть утверждение уже доказано для некоторого n , рассмотрим семейство S с $n+1$ переменной: x_1, x_2, \dots, x_{n+1} . Разобьём S на три подкласса: S_1 - все дизъюнкты представимые в виде $x_{n+1} \vee a$, S_2 - все дизъюнкты представимые в виде $x_{n+1} \vee b$, S_3 - дизъюнкты не содержащие x_{n+1} . Дизъюнкты множества $R = \{a \vee b | x_{n+1} \vee a \in S_1, \neg x_{n+1} \vee b \in S_2\}$ получены из дизъюнктов семейства S применением правила резолюции (здесь допускаются вырожденные случаи, когда a либо b отсутствует, причём при одновременном отсутствии a и b в R включается константа \top). Рассмотрим семейство дизъюнктов $R \cup S_3$ и покажем, что его невыполнимость эквивалентна невыполнимости S . Если S выполнимо, то интерпретация, обращающая в И дизъюнкты из S даёт значения И для всех дизъюнктов из $R \cup S_3$, т.е. $R \cup S_3$ выполнимо. Пусть S невыполнимо, рассмотрим набор истинностных значений $\alpha_1, \dots, \alpha_n$ переменных x_1, \dots, x_n и покажем, что на этом наборе хотя бы один из дизъюнктов семейства $R \cup S_3$ имеет значение \top . Так как S невыполнимо, то на наборе $\alpha_1, \dots, \alpha_n, \top$ значений переменных x_1, \dots, x_n, x_{n+1} некоторый дизъюнкт d из S имеет значение \top . Если $d \in S_3$, то он и является искомым дизъюнктом в $R \cup S_3$. Так как x_{n+1} имеет значение И, то при $d \notin S_3$ остаётся единственная возможность $d \in S_2$, т.е. $d = \neg x_{n+1} \vee b$. Аналогично на наборе $\alpha_1, \dots, \alpha_n, \top$ значений переменных x_1, \dots, x_n, x_{n+1} некоторый дизъюнкт d' из S имеет значение \top . Снова, если $d' \in S_3$, то он и есть искомый, а в противном случае (так как x_{n+1} имеет значение \top) необходимо $d' \in S_1$, то есть $d' = x_{n+1} \vee a$. Дизъюнкты a и b не содержат переменной x_{n+1} , и поэтому оба принимают значение \top , если переменные x_1, \dots, x_n имеют значения $\alpha_1, \dots, \alpha_n$. Следовательно и дизъюнкт $d'' = a \vee b$, принадлежащий R , будет иметь на наборе $\alpha_1, \dots, \alpha_n$ значений переменных x_1, \dots, x_n значение \top . Это и означает доказательство эквивалентности невыполнимости S и $R \cup S_3$. Но число переменных в $R \cup S_3$ не более n , т.е. для него имеет место эквивалентность невыполнимости и существования вывода константы \top . Если константа \top выводима из S , то S невыполнимо, если S невыполнимо, то невыполнимо $R \cup S_3$ и из $R \cup S_3$, а значит S выводима константа \top , то есть имеем требуемое утверждение.

Часть III

2. Язык логики предикатов.

В случае языка логики предикатов наш алфавит будет состоять из следующих групп символов:

- 1) Счётный список предметных переменных x_1, x_2, \dots
- 2) Счётный список предметных констант a_1, a_2, \dots
- 3) Счётный список функциональных символов f_1, f_2, \dots
- 4) Счётный список предикатных символов P_1, P_2, \dots
- 5) Логические константы И, Л
- 6) Логические связки $\neg, \vee, \&, \rightarrow, \leftrightarrow$
- 7) Скобки (,)

Каждый функциональный либо предикатный символ характеризуется своей **арностью** - некоторым натуральным числом. При этом предполагается, что для каждого натурального n имеется бесконечно много как предикатных, так и функциональных символов арности n .

Правильные выражения языка логики предикатов разбиваются на два непересекающихся множества - множество термов и множество формул. Дадим сначала индуктивное определение множества термов:

- T1) Однобуквенное слово, состоящее из предметной переменной либо предметной константы, есть терм
- T2) Если t_1, \dots, t_n - термы и g - функциональный символ арности n , то слово $g(t_1, \dots, t_n)$ - есть терм

Индуктивное определение формул логики предикатов опирается на уже введённое понятие терма:

- Ф1) Однобуквенное слово состоящее из логических констант И и Л, суть формулы логики предикатов
- Ф2) Если t_1, \dots, t_n - термы и g - предикатный символ арности n , то слово $g(t_1, \dots, t_n)$ - есть формула логики предикатов.
- Ф3) Если f и g - формулы логики предикатов, то слова $(\neg f)$? $(f \vee g)$, $(f \& g)$, $(f \rightarrow g)$, $(f \leftrightarrow g)$ суть формулы логики предикатов.
- Ф4) Если f - формула логики предикатов и x - предметная переменная, то слова $(\forall x f)$ и $(\exists x f)$ суть формулы логики предикатов.

Интерпретация языка логики предикатов называем объектом (M, F_1, F_2, F_3) , такой что:

- а) M - непустое множество, называемое областью интерпретации;
- б) F_1 - функция, сопоставляющая каждой предметной константе некоторый элемент из M ;
- в) F_2 - функция, сопоставляющая каждому функциональному символу f арности n некоторую n -местную функцию, определённую на M и принимающую значения из M .
- г) F_3 - функция, сопоставляющая каждому предикатному символу P арности n некоторой n -местный предикат определённый на M (функцию со значениями И, Л).

Если задана интерпретация $I = (M, F_1, F_2, F_3)$, то каждый терм t языка логики предикатов определяет в этой интерпретации некоторую функцию $(t)_I$, определённую на M и принимающую значения из M , а каждая формула f - предикат $(f)_I$, определённый на M (в вырожденных случаях $(t)_I$ может отождествляться с элементом M , а $(f)_I$ с логической константой).

Здесь используется следующее индуктивной определение:

- 1) Если x - предметная переменная, то $(x)_I$ есть тождественная функция от переменной x определённая на M .
- 2) Если a - предметная константа, то $(a)_I$ есть элемент $F_1(a)$ из M .
- 3) Если для термов t_1, \dots, t_n уже определены $(t_1)_I, \dots, (t_n)_I$, f - функциональный символ арности n и $\phi = F_2(f)$ - функция от n переменных отображающая M^n в M , то $(f(t_1, \dots, t_n))_I = \phi((t_1)_I, \dots, (t_n)_I)$
- 4) $(\text{И})_I = \text{И}$, $(\text{Л})_I = \text{Л}$.
- 5) Если для термов t_1, \dots, t_n уже определены $(t_1)_I, \dots, (t_n)_I$, P - предикатный символ арности n и $\pi = F_3(P)$ - функция от n переменных отображающая M^n в M , то $(P(t_1, \dots, t_n))_I = \pi((t_1)_I, \dots, (t_n)_I)$
- 6) Если для f и g уже определены $(f)_I$ и $(g)_I$, то $(\neg f)_I, (f \vee g)_I, (f \& g)_I, (f \rightarrow g)_I, (f \leftrightarrow g)_I$ получаются применением истинностных функций для $\neg, \vee, \&, \rightarrow, \leftrightarrow$ к $(f)_I$ и $(g)_I$.
- 7) Если уже определён предикат $(f)_I = \phi(x_1, \dots, x_n)$, то $(\forall x_i f)_I = \psi(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$ - предикат, принимающий значение И на таких наборах $\alpha_1, \dots, \alpha_{i-1}, \alpha_{i+1}, \dots, \alpha_n$ элементов M , что для каждого α_i из M значение $\phi(\alpha_1, \dots, \alpha_n)$ есть И. Аналогично, $(\exists x_i f)_I = \xi(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$ - предикат, принимающий значение И на таких наборах $\alpha_1, \dots, \alpha_{i-1}, \alpha_{i+1}, \dots, \alpha_n$ элементов M , что существует α_i из M значение $\phi(\alpha_1, \dots, \alpha_n)$ есть И.

Вхождение переменной x в формулу f логики предикатов, расположено внутри подформул вида $(\forall x g)$ либо $(\exists x g)$, называется **связанным**, вхождения переменных не являющиеся связанными, называются **свободными**. Переменная имеющая хотя бы одно свободное вхождение в формулу либо терм f , называется **свободной переменной** f . Как нетрудно видеть, функция либо предикат $(f)_I$, определённая выше, зависит лишь от свободных переменных f . Формула без свободных переменных называется **замкнутой**.

Если формула f определяет в интерпретации I предикат, тождественно равный И, то она называется **истинной в I** , а I в этом случае называется **моделью для f** . Формулу логики предикатов f называем **общезначимой**, если она истинна во всех интерпретациях, и выполнимой, если хотя бы в одной интерпретации она определяет не тождественно ложный предикат. Если формула $f_1 \& \dots \& f_n \rightarrow f_0$ общезначима, то f_0 называется **логическим следствием** формул f_1, \dots, f_n , если формула $f_1 \leftrightarrow f_2$ общезначима, то формулы f_1 и f_2 называются **эквивалентными**.

Для указания дедуктивной системы, перечисляющей все общезначимые формулы логики предикатов, воспользуемся имеющимися у нас схемами аксиом А1)-А3), в которых теперь f, g, h - произвольные формулы логики предикатов, и добавим к ним следующие две схемы аксиом:

A4) Если f, g - формулы логики предикатов и x - предметная переменная, не являющаяся свободной переменной формулы f , то формула $(\forall x(f \rightarrow g)) \rightarrow (f \rightarrow (\forall xg))$ есть аксиома.

A5) Если f - есть формула логики предикатов, x - предметная переменная и t -терм, причём никакое свободное вхождение переменной x в f не расположено в области действия квантора по переменной, входящей в t , то формула $(\forall xf) \rightarrow S_t^x f$, где посредством $S_t^x f$ обозначен результат подстановки в f терма t вместо всех свободных вхождений переменной x - есть аксиома.

Правилами вывода в этой дедуктивной системе являются уже известное нам правило modus ponens, а также новое правило (известное как правило обобщения), позволяющее из формулы f выводить формулу $(\forall xf)$. Имеет место следующее утверждение, которое мы приводим без доказательства, так как в последующем не будем рассматривать дедуктивную систему для логики предикатов:

Утверждение 5. (Теорема Гёделя о полноте исчисления предикатов). Каждая общезначимая формула логики предикатов является выводимой в приведённой выше дедуктивной системе.

Излагаемый далее алгоритмический подход к установлению общезначимости формул логики предикатов не использует данной дедуктивной системы. В отличие от логики высказываний, для логики предикатов не существует алгоритма, распознающего общезначимые формулы (теорема Чёрча). С другой стороны, существует алгоритмическая процедура, перечисляющая все общезначимые формулы (например, основанная на дедуктивной системе для логики предикатов) и позволяющая для любой общезначимой формулы за конечное число шагов установить её общезначимость. С практической точки зрения, однако, данная процедура является чрезмерно трудоёмкой, и развитие работ по автоматическому доказательству теорем в логике предикатов было связано с поиском более приемлемых в "прикладном" отношении её модификаций. Мы опишем здесь одну из таких модификаций, связанную с использованием при поиске доказательства так называемого "правила резолюции" аналога уже известного нам правила для логики высказываний. Согласно данной процедуре, доказательство общезначимости замкнутой формулы F складывается из следующих этапов:

1) Рассматривается формула $F_0 = \neg F$, и далее предпринимаются действия, направленные на установление невыполнимости формулы F_0 (то есть формула F доказывается "от противного").

2) К формуле F_0 применяется цепочка эквивалентных (в смысле определённой выше эквивалентности формул) преобразований:

a) используются эквивалентные замены $(f \leftrightarrow g) \rightarrow (f \& g \vee (\neg f) \& (\neg g)), (f \rightarrow g) \rightarrow (\neg f \vee g)$ позволяющие устраниТЬ логические связки вида $\leftrightarrow, \rightarrow$.

b) используются замены вида $(A * QxB) \rightarrow Qy(A * S_y^x B)$, где $*$ - связка \vee либо $\&$, Q - квантор \forall либо \exists , A и B - формулы, y - предметная переменная не входящая ни в A , ни в B . Кроме того используются замены $\neg \forall x A \rightarrow \exists x(\neg A), \neg \exists x A \rightarrow \forall x(\neg A)$.

Замены указанные в пункте б), применяются до тех пор пока это воз-

можно, они осуществляют перемещение кванторов к "началу" формулы, в результате чего F_0 преобразуется в формулу F_1 вида $Q_1x_1 \dots Q_nx_n A$ где A - бескванторная формула, Q_1, \dots, Q_n - символы кванторов. Про F_1 говорят, что она находится в **предварённой нормальной форме**.

3) Опишем вспомогательное преобразование формулы f логики предикатов в формулу f' , такое что f выполнима тогда и только тогда, когда выполнима f' . Предполагаем, что f - замкнутая формула в предварённой нормальной форме, причём в её "кванторной приставке" есть квантор \exists : $f = \forall x_1 \dots \forall x_k \exists x_{k+1} g$, $k \geq 0$ g - формула в предварённой нормальной форме. Выберем функциональный символ ϕ ариности k (если $k = 0$, то - предметную константу ϕ) не встречающуюся в g , и положим $f' = \forall x_1 \dots \forall x_k g'$, где $g' = S_{\phi(x_1, \dots, x_k)}^{x_{k+1}} g$ - результат замены всех свободных вхождений переменной x_{k+1} в g на $\phi(x_1, \dots, x_k)$. Предположим, что формула f выполнима, пусть I - интерпретация, в которой её значение есть И. Рассмотрим предикат $(g)_I = h(x_1, \dots, x_{k+1})$. Так как $(\forall x_1 \dots \forall x_k \exists x_{k+1} g)_I = \text{И}$, то для любых элементов a_1, \dots, a_k области M интерпретации I существует элемент a_{k+1} этой области такой, что $h(a_1, \dots, a_{k+1}) = \text{И}$. Определим на M функцию $p(x_1, \dots, x_k)$, полагая в указанной ситуации $p(a_1, \dots, a_k) = a_{k+1}$. Тогда $h(x_1, \dots, x_k, p(x_1, \dots, x_k)) \equiv \text{И}$ на M .

Пусть I' - интерпретация, отличающаяся от I только тем, что в ней символу ϕ сопоставляется функция $p(x_1, \dots, x_k)$. Индукцией по построению формулы A нетрудно установить, что формула $S_{t_1, \dots, t_n}^{y_1, \dots, y_n} A$ - результат одновременной замены в A всех свободных переменных y_1, \dots, y_n на термы t_1, \dots, t_n - определяет в произвольной интерпретации J предикат, получающийся подстановкой в $(A)_J$ функций $(t_1)_J, \dots, (t_n)_J$ вместо переменных y_1, \dots, y_n (если только соблюдено условие отсутствия свободного вхождения переменной y_i , расположенного в области действия квантора по переменной из терма t_i , $i = 1, \dots, n$). Используя это утверждение (называемое далее "леммой об интерпретации"), получим, что $(g')_{I'} \equiv \text{И}$ на M . Таким образом, $(f')_{I'} \equiv \text{И}$, и формула f' выполнима. Обратно, если f' выполнима, то рассматриваем интерпретацию I , в которой $(f')_I \equiv \text{И}$? то есть $(g')_I \equiv \text{И}$ на M , снова используя лемму об интерпретациях, находим, что $(g')_I$ получено подстановкой в функцию $(g)_I = h(x_1, \dots, x_k, x_{k+1})$ функции $(\phi(x_1, \dots, x_k))_I = p(x_1, \dots, x_k)$ вместо переменной x_{k+1} , то есть для любых a_1, \dots, a_k из M выполнено $h(a_1, \dots, a_k, p(a_1, \dots, a_k)) = \text{И}$, и $(f)_I \equiv \text{И}$.

Описанное выше преобразование $f \rightarrow f'$ позволяет последовательно устранять кванторы \exists из кванторной приставки, применяя его необходимое число раз к формуле F_1 , получим в результате формулу F_2 вида $\forall x_1 \dots \forall x_n A$, где A - бескванторная формула. Говорят, что F_2 находится в Сколемовской нормальной форме. Заметим, что общезначимость исходной функции F эквивалентна невыполнимости формулы F_2 .

4) Для анализа выполнимости формул f вида $\forall x_1 \dots \forall x_n g$, где g - бескванторная формула, обычно используется теорема Эрбрана. Чтобы св-

формулировать эту теорему, введём понятие Эрбрановского универсума H_f формулы f указанного вида. H_f есть множество термов, задаваемое посредством следующего индуктивного определения:

- а) Для каждой предметной константы a , встречающейся в f , однобуквенный терм a входит в H_f . Если f не имеет предметных констант, то в H_f включается терм a_1 , где a_1 - первая по алфавитному списку предметная константа.
- б) Если термы t_1, \dots, t_n принадлежат H_f и функциональный символ ϕ арности n встречается в формуле f , то терм $\phi(t_1, \dots, t_n)$ входит в H_f .

Иными словами, H_f образовано всеми термами, которые можно построить при помощи предметных констант и функциональных символов, встречающихся в f (если f не имеет предметных констант, то разрешается использовать константу a_1). Имеет место следующее утверждение.

Теорема (Эрбран). Замкнутая формула f вида $\forall x_1 \dots \forall x_n g$, где g не содержит кванторов, выполнима тогда и только тогда, когда каждое конечное подмножество множества $\{S_{t_1, \dots, t_n}^{x_1, \dots, x_n} g \mid t_1, \dots, t_n \in H_f\}$ - выполнимо.

Доказательство. Если в некоторой интерпретации I формула $\forall x_1 \dots \forall x_n g$ имеет значение И, то в этой же интерпретации (согласно лемме об интерпретациях истинны и все формулы $S_{t_1, \dots, t_n}^{x_1, \dots, x_n} g$. Поэтому предположим, что выполнимо каждое конечное подмножество множества $M = \{S_{t_1, \dots, t_n}^{x_1, \dots, x_n} g \mid t_1, \dots, t_n \in H_f\}$, и будем доказывать выполнимость f .

Прежде всего, установим существование такой интерпретации I , в которой все формулы множества M имеют значение И. Если M - конечно, то это сразу вытекает из сделанного предположения. Пусть M бесконечно. Рассмотрим множество $A = \{A_1, A_2, \dots\}$ всех встречающихся в формулах из M подформул вида $P(\tau_1, \dots, \tau_m)$, где P -предикатный символ (такие формулы, получающиеся согласно пункту Ф2 определения формул логики предикатов принято называть **атомарными**, или, короче, **атомами**). Легко видеть, что A бесконечно. Пусть M_i - подмножество множества M , состоящее из всех таких формул, в которых встречаются только атомы из $\{A_1, A_2, \dots, A_i\}$. Очевидно, что $M = \bigcup_{i=1}^{\infty} M_i$, причём каждое M_i - конечно. Так как каждая формула множества M построена из своих атомов при помощи одних и только логических связок, то определение истинностных значений этих атомов в какой-либо интерпретации позволяет однозначно определить и значение самой формулы. Ввиду выполнимости каждого M_i , мы можем рассмотреть теперь непустые множества V_i наборов $\alpha_1, \dots, \alpha_i$ истинностных значений атомов A_1, \dots, A_i , при которых все формулы из M_i имеют значение И. Построим бесконечный граф - дерево T , вершинами i -го яруса которого будут служить элементы множества V_i (при $i = 0$ рассматриваем дополнительную вершину v_0 - корень дерева T). Нетрудно видеть, что если $(\alpha_1, \dots, \alpha_i, \alpha_{i+1}) \in V_{i+1}$, то $(\alpha_1, \dots, \alpha_i) \in V_i$, в этом случае вершины $(\alpha_1, \dots, \alpha_i, \alpha_{i+1})$ и $(\alpha_1, \dots, \alpha_i)$ соединяются в дереве T ребром, и других рёбер граф T не имеет (за исключением рёбер ведущих от корня v_0 ко всем вершинам (α_1) из V_1). По построению, граф T бесконечен и из каждой из его вершин выходит лишь конечное число рёбер. Он связан (вершина v_0 достижима из любой вершины) и не имеет циклов (так как из любой верши-

ны множества V_i в множество V_{i-1} ведёт лишь одно ребро, то при наличии цикла можно было бы усмотреть противоречие из наличия двух рёбер, выходящих из той его вершины, которая относится к V_i с наибольшим номером i), то есть является деревом. Поэтому, согласно известной лемме Кёнига из теории графов, в дереве T существует бесконечный путь $\pi = v_0, v_1, v_2, \dots, v_i \in V_i$ при $i = 1, 2, \dots$. Согласно построению дерева T , если $v_i = (\alpha_1, \dots, \alpha_i)$ и $v_{i+1} = (\beta_1, \dots, \beta_i + 1)$, то $\alpha_1 = \beta_1, \dots, \alpha_i = \beta_i$.

Сопоставим каждому натуральному i истинностное значение α_i , такое что $v_i = (\gamma_1, \dots, \gamma_{i-1}, \alpha_i)$. Тогда, очевидно, для каждого $i = 1, 2, \dots$ будет выполнено $v_i = (\alpha_1, \alpha_2, \dots, \alpha_i)$. Определим теперь в качестве области D интерпретации I множество H_f . Каждому символу предметной константы, встречающемуся в f , сопоставляем в качестве значения самого себя, функциональному символу ϕ , встречающемуся в f , сопоставляем функцию ϕ_{t_1} над термами из H_f , преобразующую набор (t_1, \dots, t_n) таких термов в $\phi(t_1, \dots, t_n)$. Если $A_i = P(\tau_1, \dots, \tau_m)$ (здесь $\tau_1, \dots, \tau_m \in H_f$), то значение предиката P_I на наборе (τ_1, \dots, τ_m) полагаем равным α_i , $i = 1, 2, \dots$. В остальном интерпретацию I доопределяем произвольно.

По определению I атомы A_1, A_2, \dots имеют в ней истинностные значения $\alpha_1, \alpha_2, \dots$. Если h - произвольная формула из M , то она входит в некоторое M_i , и принимает на наборе истинностных значений $(\alpha_1, \dots, \alpha_i)$ своих атомов A_1, \dots, A_i , значение И, иными словами, $(h)_I = \text{И}$, и существование требуемой интерпретации установлено. Пусть $(g)_I = G(x_1, \dots, x_n)$. Если (τ_1, \dots, τ_n) - произвольный набор термов из H_f , то согласно лемме об интерпретациях имеем $G((\tau_1)_I, \dots, (\tau_n)_I) = (S_{t_1, \dots, t_n}^{x_1, \dots, x_n} g)_I = \text{И}$. С другой стороны, согласно определению I имеем $(\tau_1)_i = \tau_1, \dots, (\tau_n)_i = \tau_n$. Отсюда $G(\tau_1, \dots, \tau_n) = \text{И}$, то есть G - тождественно истинный предикат, и $(\forall x_1 \dots \forall x_n g)_I = \text{И}$. Теорема доказана.

Согласно доказанной теореме, невыполнимость формулы F_2 , полученной в пункте 3) и имеющей вид $\forall x_1 \dots \forall x_n A$, эквивалентна существованию конечного подмножества множества $\{S_{t_1, \dots, t_n}^{x_1, \dots, x_n} A | t_1, \dots, t_n \in H_{F_2}\}$, являющегося невыполнимым, так как проверка выполнимости конечного подмножества указанного множества формул является, по существу проверкой выполнимости конечного множества формул логики высказываний и алгоритмически реализуема, то для установления невыполнимости F_2 можно применять перебор таких конечных подмножеств. Эта процедура, однако, является чрезмерно трудоёмкой, и наши дальнейшие действия будут использовать теорему Эрбрана лишь в неявном виде.

Используя уже описанную для логики высказываний процедуру, преобразуем формулу A к конъюнктивной нормальной форме получим формулу A' вида $\&_{i=1}^m B_i$, где каждое B_i - дизъюнкция атомов либо их отрицаний. (Атомы и их отрицания называются **литерами**, дизъюнкции литер - **дизъюнктами**, логическая константа Л по определению также считается дизъюнктом). Нетрудно видеть, что существование конечного невыполнимого подмножества множества $\{S_{t_1, \dots, t_n}^{x_1, \dots, x_n} A | t_1, \dots, t_n \in H_{F_2}\}$ эквивалентно существованию конечного невыполнимого подмножества множества $\{S_{t_1, \dots, t_n}^{x_1, \dots, x_n} B_i | t_1, \dots, t_n \in H_{F_2}, i = 1, \dots, m\}$. Последнее нам удобно будет впо-

следствие формулировать как существование таких формул f_1, \dots, f_l , не имеющих общих переменных и полученных из некоторых формул списка $\{B_1, \dots, B_m\}$ переобозначением без отождествления их переменных x_1, \dots, x_n , а также такой подстановки σ вместо переменных формул f_1, \dots, f_l термов множества H_{F_2} , что система $\{\sigma(f_1), \dots, \sigma(f_l)\}$ - невыполнима. (Здесь подстановка σ рассматривается как оператор на множестве формул).

5) Как и в случае логики высказываний, для установления невыполнимости формулы F_2 мы введём вспомогательную дедуктивную систему, аксиомами которой будут являться определённые выше дизъюнкты B_1, \dots, B_m , и докажем, что выводимость в этой системе константы \perp эквивалента невыполнимости F_2 . Для формулировки правил вывода данной дедуктивной системы нам понадобится следующая **лемма об унифицирующей подстановке** (подстановку термов t_1, \dots, t_n вместо переменных x_1, \dots, x_n рассматриваем как оператор на множестве бескванторных формул и термов):

Лемма. Если $f_1, g_1, \dots, f_k, g_k$ - бескванторные формулы либо термы и существует такая подстановка σ , что $\sigma(f_1) = \sigma(g_1), \dots, \sigma(f_k) = \sigma(g_k)$, то существует минимальная подстановка σ_1 , обладающая этим свойством, то есть $\sigma_1(f_1) = \sigma_1(g_1), \dots, \sigma_1(f_k) = \sigma_1(g_k)$ и для любой подстановки σ_2 , удовлетворяющей условиям $\sigma_2(f_1) = \sigma_2(g_1), \dots, \sigma_2(f_k) = \sigma_2(g_k)$, выполнено: $\sigma_2 = \sigma_3 \sigma_1$ при подходящей подстановке σ_3 (здесь умножение подстановок означает их последовательное применение).

Доказательство Если рассматривать каждый входящий в бескванторную формулу либо терм f символ ϕ (предикатный символ, функциональный симболов либо логическую связку) арностип как обозначение n -местной операции над термами (в случае логической связки - над бескванторными формулами), преобразующей набор (t_1, \dots, t_n) в слово $\phi(t_1, \dots, t_n)$, то f будет определять некоторый оператор f^* на наборе (τ_1, \dots, τ_m) значений переменных x_1, \dots, x_m будет равно $S_{\tau_1, \dots, \tau_m}^{x_1, \dots, x_m} f$. Пусть x_1, \dots, x_m - все переменные, встречающиеся в $f_1, g_1, \dots, f_k, g_k$ из условия леммы. Тогда выполнение условий $\sigma(f_1) = \sigma(g_1), \dots, \sigma(f_k) = \sigma(g_k)$ для подстановки σ термов τ_1, \dots, τ_m вместо переменных x_1, \dots, x_m является решением системы уравнений $f_1^* = g_1^*, \dots, f_k^* = g_k^*$. Для описания в явном виде всей совокупности её решений используем следующие эквивалентные преобразования данной системы:

- a) Если при некотором i слова f_i^*, g_i^* имеют, соответственно вид $\phi(t_1, \dots, t_p)$ и $\psi(t'_1, \dots, t'_p)$, то при $\phi \neq \psi$ делаем вывод о несовместности системы (согласно предположению леммы, это невозможно), а при $\phi = \psi$ (следовательно, при $p = q$) заменяем уравнение $f_i^* = g_i^*$ на группу уравнений $t_1 = t'_1, \dots, t_p = t'_p$.
- б) Если f_i^* совпадает с g_i^* , то уравнение $f_i^* = g_i^*$ удаляется.
- в) Если при некотором i одно из слов f_i^*, g_i^* есть переменная x , а другое терм t , $x \neq t$, то в случае, когда t содержит x , делаем вывод о несовместности системы (невозможно по предположению леммы), иначе - подставляем t вместо x во все оставшиеся уравнения системы. Данное преобразование выполняется только при условии, что существует отличное от $f_i^* = g_i^*$ уравнение содержащее x .

Преобразование в) уменьшает число переменных x , не имеющих явного выражения $x = t$ через остальные переменные системы либо имеющих более одного вхождения в систему, и оно применимо лишь конечное число раз.

Преобразование а), в случае применения его к уравнениям максимально возможной длины, уменьшает число уравнений данной либо большей длины, и таким образом его возможно применять лишь конечное число раз по завершении преобразований в)

Таким образом процесс применения к системе уравнений преобразований а), б), в) обрывается за конечное число шагов. Результирующая система к которой эти преобразования неприменимы может иметь лишь вид $x_{i_1} = \theta_1^*, \dots, x_{i_p} = \theta_p^*$, где термы $\theta_1, \dots, \theta_p$ не содержат переменных x_{i_1}, \dots, x_{i_p} . Выберем теперь в качестве σ_1 подстановку термов $\theta_1, \dots, \theta_p$ вместо переменных x_{i_1}, \dots, x_{i_p} . Если переменные x_{i_1}, \dots, x_{i_p} имеют своими значениями термы $\theta_1, \dots, \theta_p$, а каждая переменная $x_j, j \in \{1, \dots, m\} \setminus \{i_1, \dots, i_p\}$, имеет значением однобуквенный терм x_j , то уравнения $x_{i_1} = \theta_1^*, \dots, x_{i_p} = \theta_p^*$ выполняются, а следовательно, $\sigma_1(f_1) = \sigma_1(g_1), \dots, \sigma_1(f_k) = \sigma_1(g_k)$. Предположим, что σ_2 - подстановка такая, что $\sigma_2(f_1) = \sigma_2(g_1), \dots, \sigma_2(f_k) = \sigma_2(g_k)$. Без ограничения общности можно считать, что σ_2 подставляет термы τ_1, \dots, τ_q вместо переменных $x_1, \dots, x_q, q \geq m$. Набор значений (τ_1, \dots, τ_m) переменных x_1, \dots, x_m удовлетворяет системе $x_{i_1} = \theta_1^*, \dots, x_{i_p} = \theta_p^*$. Это означает, что выполнены соотношения:

$$\tau_{i_1} = S_{\tau_{j_1}, \dots, \tau_{j_r}}^{x_{j_1}, \dots, x_{j_r}} \theta_1, \dots, \tau_{i_p} = S_{\tau_{j_1}, \dots, \tau_{j_r}}^{x_{j_1}, \dots, x_{j_r}} \theta_p,$$

где $\{j_1, \dots, j_r\} = \{1, \dots, m\} \setminus \{i_1, \dots, i_p\}$. Если теперь выбрать в качестве σ_3 подстановку термов $\tau_{j_1}, \dots, \tau_{j_r}, \tau_{m+1}, \dots, \tau_q$ вместо переменных $x_{j_1}, \dots, x_{j_r}, x_{m+1}, \dots, x_q$, то получим $\sigma_2 = \sigma_3 \sigma_1$. Лемма доказана.

Подстановка σ_1 существование которой установлено в данной лемме, называется **унифицирующей подстановкой** для пар $(f_1, g_1), \dots, (f_k, g_k)$. **Замыканием** дизъюнкта D (обозначаемым $[D]$) будем называть формулу $\forall x_1 \forall x_k D$, где x_1, \dots, x_k - все входящие в D предметные переменные. Если определённое на множестве дизъюнктов "правило вывода" позволяет выводить из дизъюнктов D_1, \dots, D_r лишь такие следствия D_0 , что формула $[D_0]$ есть логическое следствие формул $[D_1], \dots, [D_r]$, то такое правило вывода назовём корректным. Как легко видеть, если при использовании некоторого набора корректных правил вывода из дизъюнктов B_1, \dots, B_m , введённых в пункте 4) удаётся за конечное число шагов извлечь логическую константу L , то формула F_2 невыполнима, а F - общезначима. Предполагаемая здесь процедура автоматического доказательства теорем использует при поиске такого вывода константы K следующие (как легко видеть, корректные) правила вывода:

R0 (переобозначения переменных). Из произвольного дизъюнкта D выводится дизъюнкт D' , полученный из D переобозначением без отождествлений переменных, входящих в D .

R1 (правило резолюции). Если $f_1 \vee \dots \vee f_s$ и $g_1 \vee \dots \vee g_r$ - дизъюнкты, множества переменных которых не перемекаются, причём f_i имеет вид

$P(t_1, \dots, t_k)$, а g_j - вид $\neg P(t'_1, \dots, t'_k)$ и σ - унифицированная подстановка для пары $(P(t_1, \dots, t_k), P(t'_1, \dots, t'_k))$, $i \in \{1, \dots, s\}$, $j \in \{1, \dots, r\}$, то выводится дизъюнкт

$$\sigma(f_1) \vee \dots \vee \sigma(f_{i-1}) \vee \sigma(f_{i+1}) \vee \dots \vee \sigma(f_s) \vee \sigma(g_1) \vee \dots \vee \sigma(g_{j-1}) \vee \sigma(g_{j+1}) \vee \dots \vee \sigma(g_r)$$

(если $s = r = 1$, то выводится константа \perp).

Дизъюнкт получаемый согласно правилу R1, называется **резольвентой** исходных дизъюнктов.

R2 (правило склеивания) Если $f_1 \vee \dots \vee f_s$ - дизъюнкт и σ - унифицирующая подстановка для (f_i, f_j) , $i \neq j$, $i, j \in \{1, \dots, s\}$, то выводится дизъюнкт $\sigma(f_1) \vee \dots \vee \sigma(f_{i-1}) \vee \sigma(f_{i+1}) \vee \dots \vee \sigma(f_s)$

Утверждение 6. Если формула F_2 невыполнима, то за конечное число шагов из дизъюнктов B_1, \dots, B_m при помощи правил вывода R0, R1, R2 может быть выведена логическая константа \perp .

Доказательство Как было замечено в конце пункта 4), невыполнимость F_2 влечёт существование таких формул f_1, \dots, f_l , не имеющих общих переменных и полученных из некоторых формул списка $\{B_1, \dots, B_m\}$ переобозначением без отождествления их переменных x_1, \dots, x_n , а также такой подстановки σ вместо переменных формул f_1, \dots, f_l термов множества H_{F_2} , что система $\{\sigma(f_1), \dots, \sigma(f_l)\}$ невыполнима.

Каждая формула $\sigma(f_i)$ имеет вид $A_1^{\alpha_1} \vee \dots \vee A_s^{\alpha_s}$, где A_1, \dots, A_s - атомы без переменных, $\alpha_1, \dots, \alpha_s$ - логические константы. Если некоторые $A_j^{\alpha_j}$ в этой формуле совпадают между собой, то после их отождествления формула $\sigma(f_i)$ преобразуется в некоторый дизъюнкт, который обозначим d_i . Пусть A_1, \dots, A_n - все различные атомы, встречающиеся в дизъюнктах d_1, \dots, d_l , $n \geq 1$. Множество $M = \{d_1, \dots, d_l\}$ представим в виде $M_1 \cup M_2 \cup M_3$, где M_1 - все дизъюнкты, в которые не входит A_1 , M_2 - все дизъюнкты вида $A_1 \vee B$, M_3 - все дизъюнкты вида $\neg A_1 \vee B$. Рассмотрим множество M_4 , образованное всеми дизъюнктами вида $B \vee C$, где $A_1 \vee B \in M_2$, $\neg A_1 \vee C \in M_3$ (если $A_1 \in M_2$, $\neg A_1 \in M_3$, то в M_4 включается \perp). Так как $\{d_1, \dots, d_l\}$ - невыполнимое множество дизъюнктов, то для любых истинностных значений a_1, \dots, a_n атомов A_1, \dots, A_n существует формула $d \in M$, имеющая значение \perp . Если $d \in M_1$, то $d \in M' = M_1 \cup M_4$. Если такой формулы d в M_1 нет, то при $a_1 = \text{И}$ существует формула $d' = \neg A_1 \vee C$ имеющая значением \perp , а при $a_1 = \perp$ - формула $d'' = A_1 \vee B$, имеющая значение \perp , $d' \in M_3$, $d'' \in M_2$. Таким образом, варьируя при фиксированных a_2, \dots, a_n , значение a_1 , находим указанные формулы d', d'' и получаем, что формула $d''' = B \vee C$, принадлежащая M_4 , а следовательно и M' , имеет на наборе значений a_2, \dots, a_n фтормов A_2, \dots, A_n значение \perp . Мы получили, что для любых истинностных значений атомов A_2, \dots, A_n в M' имеется формула, значение которой есть \perp , т.е. M' невыполнимо и имеет не более $n-1$ атома.

Пусть дизъюнкт $B \vee C$ из M_4 получен из дизъюнктов $A_1 \vee B \in M_2$ и $\neg A_1 \vee C \in M_3$. $A_1 \vee B$ возникло из некоторого $\sigma(f_i)$ отождествлением одинаковых литер. Следовательно, $\sigma(f_i) = \sigma'_1 \sigma''_1(f_i)$, где σ''_1 унифицирует те

литеры из f_i , которые совпадают после применения к ним σ . Аналогично, $\sigma(f_j) = \sigma'_2 \sigma''_2(f_j)$. Так как переменные в f_i и f_j различны, то можно считать $\sigma''_2 = \sigma''_1 = \sigma''$, $\sigma'_2 = \sigma'_1 = \sigma'$. Нетрудно видеть, что $\sigma''(f_i)$ и $\sigma''(f_j)$ получены из f_i , f_j несколькими применениями правила R2. Выделяя в $\sigma''(f_i)$ и $\sigma''(f_j)$ те атомы, применение подстановки σ' к которым даёт, соответственно, A_1 и $\neg A_1$, будем иметь для $\sigma''(f_i)$ и $\sigma''(f_j)$ следующие представления: $\sigma''(f_i) = P(\theta'_1, \dots, \theta'_s) \vee B'$, $\sigma''(f_j) = \neg P(\theta'_1, \dots, \theta'_s) \vee C'$, $\sigma'(P(\theta_1, \dots, \theta_s)) = A_1$, $\sigma'(B') = B$, $\sigma'(P(\theta'_1, \dots, \theta'_s)) = A_1$, $\sigma'(C') = C$. Рассматривая далее подстановку ρ , унифицирующую пару $(P(\theta_1, \dots, \theta_s), P(\theta'_1, \dots, \theta'_s))$, получим, что для некоторой подстановки ρ' выполняется $\sigma' = \rho' \rho$. При этом дизъюнкт $\rho(B') \vee \rho(C')$ получен из $\sigma''(f_i)$, $\sigma''(f_j)$ по правилу R1, причём $B \vee C = \rho'(\rho(B') \vee \rho(C'))$. Проведённые рассуждения показывают, что применяя к $\{f_1, \dots, f_l\}$ правила R0, R1, R2 можно получить такие формулы $\{g_1, \dots, g_r\}$, не имеющие общих переменных, что $M_1 \cup M_4 = \{\tilde{\sigma}(g_1), \dots, \tilde{\sigma}(g_r)\}$ для подходящей подстановки $\tilde{\sigma}$, то есть для g_1, \dots, g_r выполнены те же условия, что и для f_1, \dots, f_l , и число атомов в формулах $\{\tilde{\sigma}(g_1), \dots, \tilde{\sigma}(g_r)\}$, хотя бы на один меньше, чем в $\{\sigma(f_1), \dots, \sigma(f_l)\}$.

Продолжая описанный процесс логического вывода, за конечное число шагов получим систему $\{h_1, \dots, h_q\}$, удовлетворяющую указанным условиям, причём такую, что для неё число атомов в соответствующем семействе $\{\tilde{\sigma}(h_1), \dots, \tilde{\sigma}(h_q)\}$ окажется равно 0. Это означает, что $q = 1$ и $\sigma(h_1) = \Pi$, а следовательно, и $h_1 = \Pi$, и утверждение доказано.

Приведём простой пример применения описанной выше процедуры автоматического доказательства теорем логики предикатов. Формула F , обозначимость которой требуется установить имеет вид:

$$(\forall x \exists y (P(x, y) \vee Q(x, y)) \& \forall x \forall y (P(x, y) \rightarrow R(x, f(x, y))) \& \\ \& \forall x \forall y (Q(x, y) \rightarrow R(x, f(x, y))) \rightarrow \forall x \exists y R(x, y))$$

Отрицание F_0 формулы F представляет собой конъюнкцию следующих формул:

$$\forall x \exists y (P(x, y) \vee Q(x, y));$$

$$\forall x \forall y (P(x, y) \rightarrow R(x, f(x, y)));$$

$$\forall x \forall y (Q(x, y) \rightarrow R(x, f(x, y)));$$

$$\exists x \forall y \neg R(x, y).$$

нетрудно проверить, что описанные в пунктах 2)-4) преобразования, позволяющие определить исходную совокупность дизъюнктов B_1, \dots, B_m , можно применять к каждой из этих формул по отдельности. В первой формуле избавляемся от квантора существования, вводя в рассмотрение новый

функциональный символ g : $\forall x(P(x, g(x)) \vee Q(x, g(x)))$, в последней формуле для этой цели используем новую предметную переменную a : $\forall y \neg R(a, y)$. Окончательно, получаем следующую систему дизъюнктов:

- 1) $P(x, g(x)) \vee Q(x, g(x))$
- 2) $\neg P(x, y) \vee R(x, f(x, y))$
- 3) $\neg Q(x, y) \vee R(x, f(x, y))$
- 4) $\neg R(a, y)$

Далее переобозначаем переменную y в дизъюнкте 4 на y' и унифицируем $R(a, y')$ с $R(x, f(x, y))$ из дизъюнкта 2: $a = x$, $y' = f(x, y)$, $y' = f(a, y)$. Таким образом, унифицирующая подстановка имеет вид $S_{a, f(a, y)}^{x, y'}$, и с её помощью получаем по правилу R1 новый дизъюнкт:

- 5) $\neg P(a, y)$

Далее из дизъюнктов 3, 4 получаем аналогичным образом:

- 6) $\neg Q(a, y)$

Унифицируем $P(a, y)$ с $P(x, g(x))$ из дизъюнкта 1: $x = a$, $y = g(a)$, и получаем следствие дизъюнктов 5, 1:

- 7) $Q(a, g(a))$

Наконец, унифицируем $Q(a, g(a))$ и $\neg Q(a, y)$ (то есть дизъюнкты 6, 7) и выводим константу Λ .

Вообще говоря, процесс поиска вывода константы Λ из исходных дизъюнктов B_1, \dots, B_m при помощи правил вывода R0, R1, R2 может оказаться неприемлемо трудоёмким, даже в рассматриваемом выше простом примере можно было бы указать достаточное количество альтернативных способов применения правил вывода (например, получить резольвенты дизъюнктов 1 и 2, либо 1 и 3).

Трудоёмкость поиска вывода для описанной процедуры растёт экспоненциально с ростом глубины вывода, и это заставляет предпринимать исследования, направленные на нахождение принципов управления выводом следствий. Мы приведём здесь краткий обзор такого рода принципов, которые были найдены при практическом использовании указанной процедуры, а также при теоретическом её исследовании. Эти принципы можно разбить на две группы - принципы исключения из рассмотрения некоторых "избыточных" следствий, с доказательством сохранения полноты процедуры при их применении (они получили название "стратегии очищения"), и эвристические принципы упорядочения перебора резольвент ("стратегии упорядочения"). Начнём с простейших примеров таких стратегий:

1) Стратегия предпочтения одночленов Как нетрудно заметить, применение правила резолюции к дизъюнктам $A \vee K_1 \vee \dots \vee K_{m-1}$ и $\neg A' \vee K'_1 \vee \dots \vee K'_{n-1}$ длины m и n соответственно даёт резольвенту $K''_1 \vee \dots \vee K''_{m-1} \vee K'''_1 \vee \dots \vee K'''_{n-1}$ длины $m + n - 2$. Длина эта лишь в том случае окажется меньшей длины одного из исходных дизъюнктов, когда другой дизъюнкт имел длину 1, то есть являлся "одночленом". Исходя из такого, вообще говоря, эвристического соображения получения следствий, более простых чем исходные формулы, стратегия предпочтения одночленов рекомендует первоочередное применение правила резолюции к одночленам.

2) **Исключение тавтологий и уникальных литер** в процессе вывода следствий отбрасываются дизъюнкты вида $(A \vee \neg A \vee B_1 \vee \dots \vee B_k)$ ("тавтологии"). Если предикатный символ P встречается во всех дизъюнктах только с отрицанием, либо только без отрицания, то все содержащие P дизъюнкты отбрасываются.

Стратегия предпочтения одночленов даёт пример стратегии упорядочения, принцип исключения тавтологий и уникальных литер - пример стратегии очищения.

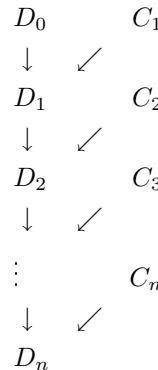
3) **Стратегия первоочерёдного применения правила склеивания:** правило R1 применяется лишь после того, как были получены все возможные в текущей ситуации следствия с применением правила R2.

4) **Стратегия использования подслушаев** Дизъюнкт D называется **подслушаем** дизъюнкта C , если существует такая подстановка σ , что C имеет вид $\sigma(D) \vee C'$. Стратегия заключается в отбрасывании всех возникающих при выводе дизъюнктов C , для которых уже найден был ранее некоторый их подслушач (данная стратегия является стратегией очищения и гарантирует получение за конечное число шагов константы " \perp ").

5) **Стратегия применения несущего множества дизъюнктов** Эта стратегия связана с выделением в исходном множестве дизъюнктов M такого, возможно б'ольшего подмножества M' , про которое известно, что оно выполнимо (например M' может быть образовано всеми аксиомами, из которых требуется извлечь заданное следствие). При выводе следствий допускается лишь такое применение правила резолюции, когда хотя бы один из несущих дизъюнктов принадлежит несущему **множеству**. Последнее определяется следующим индуктивным образом:

- a) Каждый элемент из $M \setminus M'$ относится к несущему множеству.
- б) Если резольвента получена при участии хотя бы одного элемента несущего множества, то она входит в несущее множество.
- в) Результаты применения правила R0 либо R2 к элементу несущего множества дают элемент несущего множества.

6) **стратегия использования линейных выводов** Линейный вывод представляет собой последовательность применений правила резолюции определяемую диаграммой следующего вида:



Здесь каждое C_i - либо элемент исходного множества дизъюнктов M , либо есть некоторое D_j при $j \in \{0, 1, \dots, i-1\}$. Имеет место утверждение (Андерсон-Бледсоу), согласно которому для получения константы Π достаточно использовать лишь линейные выводы (здесь и далее мы рассматриваем лишь встречающиеся в выводах правила R1, так как именно они ответственны за экспоненциально нарастающую с увеличением глубины вывода трудоёмкость поиска доказательства, не упоминая о возможных применениях "одноместных" правил R0 и R2). Заметим, что само по себе ограничение линейности вывода не устраняет древовидной схемы поиска линейного вывода при переборе различных возможностей для C_1, C_2, \dots, C_n .

7) Совместное применение стратегий использования подслучаев и слияния. Введём сначала вспомогательные понятия **слияния** и **литеры слияния**. Резольвента $\sigma(D'_1) \vee \dots \vee \sigma(D'_n) \vee \sigma(D''_1) \vee \dots \vee \sigma(D''_m)$ двух дизъюнктов $A \vee D'_1 \vee \dots \vee D'_n$ и $\neg A \vee D''_1 \vee \dots \vee D''_m$ называется **слиянием** этих дизъюнктов, если существуют такие $i \in \{1, \dots, n\}$ и $j \in \{1, \dots, m\}$, что $\sigma(D'_i) = \sigma(D''_j)$. Формула $\sigma(D'_i)$ при этом называется **литерой слияния**. Андерсону и Бледсоу принадлежит утверждение о том, что при поиске вывода константы Π достаточно ограничиться лишь такими линейными выводами, у которых каждое C_i (обозначения те же, что в пункте 6) либо принадлежит исходному множеству дизъюнктов M , либо является некоторым D_j , $j < i$, полученным при слиянии двух дизъюнктов, причём литературой (унифицируемой при применении правила резолюции формулой A либо $\neg A$) в C_i при получении D_i является некоторая литерой слияния, а само D_i является подслучаем дизъюнкта D_{i-1} . Несмотря на существенное ограничение в использовании формул C_i , не принадлежащих M (по сравнению с пунктом 6), данное утверждение практически не устраивает эффект экспоненциального нарастания трудоёмкости, так как достаточное количество альтернативных применений правил вывода на каждом шаге даёт уже исходное множество M .

8) Стратегия упорядочения литер. Возможно применение при поиске вывода константы Π другого ограничения на вид линейного вывода. В этом случае разрешающей литературой в каждом D_i является первая литерой, то есть D_{i+1} возникает из $D_i = A \vee D'_1 \vee \dots \vee D'_n$ и $C_i = D''_1 \vee \dots \vee D''_{j-1} \vee \neg A' \vee D''_{j+1} \vee \dots \vee D''_n$ как

$$\sigma(D''_1) \vee \dots \vee \sigma(D''_{j-1}) \vee \sigma(D''_{j+1}) \vee \dots \vee \sigma(D''_n) \vee \sigma(D'_1) \vee \dots \vee \sigma(D'_n),$$

причём указанное здесь упорядочение литер в дизъюнкте D_{i+1} сохраняется. Как и в пункте 7, предполагается, что каждое C_i - либо из исходного множества дизъюнктов M , либо совпадает с некоторым D_j , $j < i$, являющимся дизъюнктом слияния. Полнота данной стратегии также была установлена в работах Андерсона и Бледсоу.

Для иллюстрации применения перечисленных выше стратегий рассмотрим сравнительно простой пример доказательства теорем из теории групп. Будет доказываться утверждение о том, что в ассоциативной системе в которой уравнения вида $xa = b$ и $ay = b$ имеют левое и правое решение x и

y , существует правый единичный элемент. при формулировке этого утверждения на языке логики предикатов используем трёхместный предикатный символ $P(x, y, z)$, интерпретируемый как равенство $xy = z$. После преобразования к виду дизъюнктов аксиомы существования решений уравнений $xa = b$ и $ay = b$ принимают вид:

$$D_1 \quad - \quad P(g(x, y), x, y) \quad \text{существование решения для } xa = b$$

$$D_2 \quad - \quad P(x, h(x, y), y) \quad \text{существование решения для } ay = b$$

ассоциативность выражается двумя дизъюнктами:

$$D_3 \quad - \quad \neg P(x, y, z) \vee \neg P(y, v, w) \vee \neg P(x, w, u) \vee P(z, v, u);$$

$$D_4 \quad - \quad \neg P(x, y, z) \vee \neg P(y, v, w) \vee \neg P(z, v, u) \vee P(z, w, u);$$

Наконец, отрицание утверждения о существовании правого единичного элемента приводится к виду:

$$D_5 \quad - \quad \neg P(f(y), y, f(y)).$$

В исходной ситуации несущее множество состоит из единственного элемента D_5 , при применении правил вывода будем строить лишь линейные выводы. Стратегия предпочтения одночленов даёт лишь две возможности - применение правила R1 к D_5 и D_3 , либо к D_5 и D_4 . Выберем, например, первую из них, получим дизъюнкт:

$$D_6 \quad - \quad \neg P(x, y, f(z)) \vee \neg P(y, z, v) \vee \neg P(x, v, f(z))$$

(по мере надобности в новых дизъюнктах проводим переобозначение переменных). Теперь несущее множество имеет уже два элемента - D_5 и D_6 , и стратегия предпочтения одночленов вовлекает в рассмотрение также дизъюнкты D_1 и D_2 . Применяя правило R1 к D_1 и D_6 , получаем дизъюнкт:

$$D_7 \quad - \quad \neg P(x, y, z) \vee \neg P(g(x, f(y)), z, f(y))$$

Далее из D_7 и D_1 выводим:

$$D_8 \quad - \quad \neg P(x, y, z)$$

и, наконец, унифицируя D_8 с отрицанием D_2 , выводим константу \perp .

Разбирая пример, мы привели лишь ведущую к цели цепочку вывода. Вместе с тем, здесь имелось уже достаточно большое множество альтернативных выводов, не отсекаемых перечисленными выше стратегиями: применение правила резолюции к парам (D_5, D_4) , (D_2, D_6) , (D_2, D_7) , а также альтернативные применения этого правила к (D_1, D_6) и (D_1, D_7) дают

новые дизъюнкты, которые в свою очередь приводят к новым следствиям. Таким образом, в целом сохраняется ветвящийся характер рассматриваемого в описанной процедуре логического вывода, и это ограничивает область эффективной её применимости классом сравнительно простых задач, где проведение полного перебора является всё ещё реализуемым.

Попытки применения изложенной выше процедуры автоматического доказательства теорем в различных прикладных задачах, связанных с логическим выводом, привели к созданию языка ПРОЛОГ. Не останавливаясь на технических подробностях, изложенных в руководствах по программированию на этом языке, мы приведём здесь лишь "общелогическую" схему организации и функционирования программ на ПРОЛОГ'е. каждая такая программа представляет собой упорядоченный набор дизъюнктов (D_1, D_2, \dots, D_n), причём входной информацией для программ также служит некоторый дизъюнкт D_0 . Программа пытается найти такую подстановку σ , для которой совокупность дизъюнктов $\{\sigma(D_0), D_1, D_2, \dots, D_n\}$ становится невыполнимой, причём она не останавливается, найдя первую такую σ , а продолжает поиск и перечисляет все найденные σ вплоть до исчерпания всех возможностей, заложенных в её схеме перечисления. Сама эта схема исключительно проста - по существу, это полный перебор всех возможных линейных выводов, определяемых дизъюнктами D_1, \dots, D_n , реализуемый по принципу "сначала вглубь". Более подробно, функционирование программы происходит следующим образом. Текущая ситуация описывается при помощи набора троек $S = (S_1, \dots, S_m)$, где каждое S_i , $i = 1, \dots, m$, имеет вид (σ_i, D_i^*, k_i) , σ_i - подстановка, D_i^* - дизъюнкт, $k_i \in \{1, 2, \dots, n + 1\}$. В исходной ситуации $m = 1$, $\sigma_1 = e$ -тождественная подстановка, $D_1^* = D_0$, $k_1 = 1$. На очередном шаге преобразований происходит рассмотрение "текущей" тройки $S_m = (\sigma_m, D_m^*, k_m)$, $D_m^* = K_1 \vee \dots \vee K_s$, и перебор всех дизъюнктов номер которых не меньше k_m . Для каждого D_j , $D_j = Q_1 \vee \dots \vee Q_p$, предпринимается попытка унификации литер K_1 и $\neg Q_1$. Как только это удалось сделать, определяется резольвента D' дизъюнктов D_m^* и D_j , и к концу набора S присоединяется новая тройка $(\sigma_m \sigma', D', 1)$, где σ' - унифицирующая подстановка. Одновременно индекс k_m в тройке S_m заменяется на $j + 1$, и процесс возобновляется. Если просмотр всех дизъюнктов D_j закончился безрезультатно и новая тройка S_{m+1} не возникла, то либо $m = 1$, и тогда программа завершает работу (то есть завершает процесс перечисления результирующих подстановок), либо $m > 1$, и тогда тройка S_m отбрасывается, после чего - переход к следующему шагу. Исключительным является случай, когда дизъюнкт D' оказался константой L . В этом случае тройка S_{m+1} не вводится, а лишь предпринимается замена k_m на $j + 1$ и происходит выдача очередного результата - подстановки $\sigma_m \sigma'$. Затем - переход к очередному шагу.

Дизъюнкт $D_i = Q_1 \vee \dots \vee Q_p$ ПРОЛОГ-программы можно интерпретировать как последовательность "операторов" $\neg Q_2, \dots, \neg Q_p$, выполняемых в качестве подпрограммы для реализации условия Q_1 . Здесь возникают два принципиально разных свойства, отсутствующих у операторов программирования "обычных" алгоритмических языков. Во-первых, отсутствует чёт-

кое разделение программных переменных, встречающихся в операторе, на входные и выходные, и в различных ситуациях в одной и той же программе роли таких переменных могут меняться. Во-вторых, оператор реализуется как бы в режиме перечисления своих "выходных" переменных: сначала находится первая версия набора этих значений, реализуются последующие операторы, и если при их обработке возникает в некоторый момент ложное условие, то происходит возвращение к ранее рассмотренному оператору, который выдаёт очередную версию набора значений выходных переменных, и так далее.

Такой "режим перечисления" существенно упрощает программирование: исчезают не только затрудняющие отладку операторы "goto", но и становятся крайне редкими в явно выделенные в программе конструкции с циклами, так как эти циклы реализуются автоматически. По-видимому, указанное обстоятельство и предопределило в значительной степени дальнейшее развитие ПРОЛОГа - в первую очередь появление в нём большого числа встроенных предикатов, реализуемых интерпретатором (либо компилируемых) без привлечения механизма логического вывода, но с сохранением указанного принципа перечисления выходных значений. В ПРОЛОГ были введены также некоторые дополнительные средства управления ходом выполнения программы, и в конечном итоге программирование на нём значительно приблизилось к программированию на традиционных алгоритмических языках, хотя бы в том смысле, что написанию программы здесь также должна предшествовать разработка алгоритма, учитывающего существенным образом специфику конкретной задачи и включающая в себя оптимизацию применяемой схемы вычислений. Практическая значимость описанной выше процедуры автоматического доказательства теорем как универсального средства решения задач оказалась в результате весьма скромной, будучи явно отодвинутой на второй план предоставляемой ПРОЛОГ'ом возможностью программировать "почти без циклов".

Важный этап в развитии логических методов автоматического решения задач, связанный с попытками обнаружить универсальные, предметно-независимые способы борьбы с перебором, возникающим при поиске решения, привёл к пониманию невозможности таких способов и необходимости использования в каждой конкретной предметной области своих алгоритмических конструкций для решения задач, учитывающих статистические особенности как данной предметной области в целом так и рассматриваемых в прикладных ситуациях "потоков задач". По-видимому выработка решающих правил в предметной области является результатом достаточно трудоёмкой и сложной в логическом отношении адаптации, и лишь на уровне анализа общих принципов процессов такой адаптации можно надеяться на обнаружение предметно-независимых процедур, используемых интеллектуальными системами.