**КАЗАНСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**
**им. А. Н. Туполева**

**Ш. И. ГАЛИЕВ**

# МАТЕМАТИЧЕСКАЯ ЛОГИКА И ТЕОРИЯ АЛГОРИТМОВ

*УЧЕБНОЕ ПОСОБИЕ*

Казань 2002

УДК 6

Галиев Ш. И. Математическая логика и теория алгоритмов. – Казань: Издательство КГТУ им. А. Н. Туполева. 2002. - 270 с.

ISBN 5-93629-031-X

Пособие содержит следующие разделы. Логику высказываний и предикатов с приложениями, в том числе метод резолюций и элементы его реализации в языке ПРОЛОГ. Классические исчисления (высказываний и предикатов) и элементы неклассических логик: трёхзначные и многозначные логики, модальную, временную и нечеткую логики. Теорию алгоритмов: нормальные алгоритмы, машины Тьюринга, рекурсивные функции и их взаимосвязи. Понятие о сложности вычислений, различные (по сложности) классы задач и примеры таких задач.

Все главы снабжены контрольными вопросами и упражнениями, приведены варианты типовых заданий и тесты для самоконтроля усвоения материала.

Пособие предназначено студентам технических вузов по специальности 2201 направления «Информатика и вычислительная техника» и может быть использовано для специальности 2202 и других специальностей данного направления.

# ОГЛАВЛЕНИЕ

# ВВЕДЕНИЕ

Логика обычно понимается как наука о способах доказательств и опровержений. Математическая логика – это логика, развиваемая с помощью математических методов.

Изучая методы доказательств и опровержений, логика интересуется в первую очередь формой получения истинных выводов, а не содержанием посылок и заключений в том или ином рассуждении. Рассмотрим, например, следующие два вывода:

1. Все люди смертны. Сократ – человек. Следовательно, Сократ – смертен.
2. Все котята любят играть. Мура – котенок. Следовательно, Мура любит играть.

Оба эти вывода имеют одну и ту же форму: Все *A* суть *B*; *C* есть *A*; следовательно, *C* есть *B*. Эти выводы верны в силу своей формы, независимо от содержания, независимо от того истинны или ложны взятые по себе посылки и заключения. Систематическая формализация и каталогизация правильных способов рассуждений – одна из основных задач логики. Если при этом применяется математический аппарат и исследования посвящены в первую очередь изучению математических рассуждений, то эта логика является математической логикой (формальной логикой). Данное определение не является строгим (точным) определением. Чтобы понять предмет и метод математической логики лучше всего приняться за ее изучение.

Математическая логика начала формироваться давно. Зарождение ее идей и методов происходило в Древней Греции, Древней Индии и Древнем Китае примерно с VI в. до н. э. Уже в этот период ученые пытались расположить цепь математических доказательств в такую цепочку, чтобы переход от одного звена к другому не оставлял сомнений и завоевал всеобщее признание. Уже в самых ранних дошедших до нас рукописях «канон» математического стиля изложения прочно установлен. Впоследствии он получает окончательное завершение у великих классиков: Аристотеля, Евклида, Архимеда. Понятие доказательства у этих авторов уже ничем не отличается от нашего.

Логика как самостоятельная наука берет свое начало в исследованиях Аристотеля (384 – 322 г. до н. э.). Великий философ древности Аристотель осуществляет энциклопедическую систематизацию античных знаний во всех областях существовавшей тогда науки. Логические исследования Аристотеля изложены, в основном, в двух его трудах «Первая аналитика» и «Вторая аналитика», объединенных под общим названием «Органон» (Орудие познания).

Следует особо отметить большое значение для становления и развития математической логики одного из самых блестящих достижений в истории человечества, а именно, превращение геометрии в точную дедуктивную систему в работе Евклида (330 – 275 г. до н. э.) «Начала». Именно этот дедуктивный подход с ясным осознанием целей и методов был положен в основу развития философской и математической мысли последующих столетий.

Также большое значение для становления и развития логики сыграли достижения в алгебре (алгебра Буля) и в других математических дисциплинах, в том числе и вновь в геометрии (создание неевклидовой геометрии - геометрии Лобачевского – Гаусса – Бойяи). Краткий обзор становления математической логики можно найти в [6].

В формировании и становлении математической логики участвовали многие и многие ученые, как древних времен, так средневековья и последующих времен.

### *Принципиальное и прикладное значение математической логики*

Принципиальное значение математической логики – обоснование математики (анализ основ математики).

Прикладное значение математической логики в настоящее время очень велико. Математическая логика применяется для следующих целей:

- анализа и синтеза (построения) цифровых вычислительных машин и других дискретных автоматов, в том числе и интеллектуальных систем;
- анализа и синтеза формальных и машинных языков, для анализа естественного языка;
- анализа и формализации интуитивного понятия вычислимости;
- выяснения существования механических процедур для решения задач определённого типа;
- анализа проблем сложности вычислений.

Также математическая логика оказалась тесно связанной и с рядом вопросов лингвистики, экономики, психологии и философии.

В данном пособии излагаются основные понятия математической логики и теории алгоритмов. Материал, изложенный в пособии,

соответствует государственному образовательному стандарту для направления «Информатика и вычислительная техника» и может быть использован для студентов обучающихся по разным специальностям этого направления.

При написании пособия использовались литература [1-31], и, конечно, использованы и другие источники. В перечень литературы включены книги, которые желательно просмотреть любознательному и требовательному студенту.

В пособии в каждой главе приведены вопросы для самопроверки теоретического материала и упражнения, предназначенные для выработки навыков решения задач и углубления знаний по излагаемой теме. Кроме того, в пособии приведены варианты типовых заданий и тесты для самоконтроля усвоения материала.

Автор выражает искреннюю благодарность рецензентам д.ф.м.-н., профессорам И. З. Батыршину и Ф. Г. Мухлисову за полезные предложения и замечания, которые позволили улучшить работу.

Автор признателен своим студентам за помощь по набору текста.

*Но не всякая речь есть высказывающая речь, а лишь та, в которой содержится истинность или ложность чего-либо; мольба, например, есть речь, но она не истинна и не ложна.*

*Аристотель*

# Глава 1. ЛОГИКА ВЫСКАЗЫВАНИЙ

*И не облекайте истину ложью,*
*чтобы скрыть истину,*
*в то время как вы знаете.*

*Коран*

*Ибо мы не сильны против истины,*
*но **сильны** за истину.*

*Библия*[1]

## §1. Высказывание. Логические операции

*Высказыванием* называется любое повествовательное предложение, которое *истинно* либо *ложно*.

Примерами высказываний в математической логике являются следующие предложения:

Сократ - человек.

$2 + 2=4$.

$5 > 7$.

Не являются высказываниями в математической логике предложения:

$x > 5$ (здесь $x \in (-\infty , \infty )$ и считается переменной).

Закройте книгу!

Данное предложение ложно.

Какое же у меня есть дело на земле?

Высказывания будем обозначать заглавными печатными латинскими буквами (*A, B, C*,...) и этими же буквами с числовыми индексами.

В логике высказываний отвлекаются от содержания высказывания и интересуются истинностью либо ложностью (*истинностными значениями*) высказывания.

Таким образом, высказывание рассматривается как величина, которая может принимать два значения: «истина» либо «ложь». В дальнейшем для краткости будем обозначать значение «истина» через *И*, а «ложь» - *Л*. Если

---

[1] *По Синодальному изданию; по Восстановительному переводу эта фраза имеет вид: «Ибо мы не можем **делать** что либо против истины, а **можем** ради истины». Выделение слов здесь и в эпиграфе сделано согласно указанным источникам.*

высказывание *А* истинное, то будем говорить, что *А* принимает значение *И* (истина), и писать: *А = И*. Если высказывание *А* ложное, то будем говорить, что *А* принимает значение *Л* (ложь), и писать: *А = Л*. Заметим, что мы не будем определять, что такое истина и что такое ложь, но считаем себя способными охарактеризовать некоторые высказывания как истинные, другие - как ложные.

Из высказываний можно образовывать другие высказывания, соединяя их различными способами, т.е. производя операции над высказываниями. Эти логические операции над высказываниями таковы, что истинностные значения составных высказываний определяются только истинностными значениями составляющих высказываний, а не их содержательным смыслом.

**1**. Первой из операций над высказываниями введем отрицание. Прежде отметим, что в разговорном языке высказывание может быть отрицаемо многими способами, например: отрицанием для высказывания «2 - четное число» может служить: «2 не является четным числом», «неверно, что 2 - четное число», «не имеет места, что 2 - четное число» и т.п. Мы будем строить отрицание для данного высказывания одним способом, помещая знак отрицания $\neg$ перед всем высказыванием.

*Отрицание* - логическая операция, с помощью которой из данного высказывания *А* образуется новое высказывание, обозначаемое $\neg A$, которое истинно тогда и только тогда, когда *А* ложно. Следовательно, имеем следующую таблицу, которая называется *таблицей истинности*.

| *А* | $\neg A$ |
|---|---|
| *Л* | *И* |
| *И* | *Л* |

Высказывание $\neg A$ читается «не *А*» и логическая операция отрицания соответствует образованию нового высказывания из высказывания *А* с помощью частицы «не». В литературе встречаются и другие обозначения для $\neg A$: $\overline{A}$ или $\sim A$. Отрицание является одноместной логической операцией. Другие логические операции, которые введем ниже, - двуместные операции, сложное высказывание строится из двух данных высказываний *А* и *В*.

**2**. *Конъюнкция* - логическая операция, с помощью которой из двух данных высказываний *А* и *В* образуется новое высказывание, обозначаемое *А&В*, которое истинно тогда и только тогда, когда *А* и *В* оба истинны.

Высказывание *А&В* читается «*А* и *В*» и называется конъюнкцией *А* и *В*, а *А* и *В* называются *конъюнктивными членами*. По определению конъюнкции имеем следующую таблицу истинности.

| *А* | *В* | *А&В* |
|---|---|---|
| *Л* | *Л* | *Л* |
| *Л* | *И* | *Л* |
| *И* | *Л* | *Л* |
| *И* | *И* | *И* |

Для конъюнкции высказываний *А* и *В* в литературе встречаются и другие обозначения, например: $A \wedge B$, $A \cdot B$ или просто *АВ*.

Из определения следует, что операция конъюнкция соответствует образованию нового высказывания из двух данных соединением их союзом "и". Выражение *А&В* может служить обозначением не только для

высказывания $A$ и $B$, но и для высказываний: «как $A$, так и $B$»; «$A$ вместе с $B$»; «$A$, в то время как $B$»; «не только $A$, но и $B$», «$A$, хотя и $B$». Очевидно, что этот список можно продолжить.

Однако, $A\&B$ не является моделью для каждого случая употребления союза «и». Поясним это. Согласно определения конъюнкции истинностные значения $A\&B$ и $B\&A$ одинаковы, т.е. $A\&B$ и $B\&A$ понимаются как равносильные (равнозначные) высказывания. В то же время высказывания «Таня проснулась и солнце взошло над горизонтом», «Солнце взошло над горизонтом и Таня проснулась» понимаются как различные. Также различны высказывания «Иванову стало жарко и он пошёл искупаться», «Иванов пошёл искупаться и ему стало жарко».

**3**. *Дизъюнкция* - логическая операция, с помощью которой из двух данных высказываний $A$ и $B$ образуется новое высказывание, обозначаемое $A \vee B$, которое ложно тогда и только тогда, когда ложны оба высказывания $A$ и $B$.

Высказывание «$A \vee B$» читается «$A$ или $B$», а $A$ и $B$ называются *дизъюнктивными членами*. Из определения видно, что операция дизъюнкции соответствует образованию нового высказывания из данных $A$ и $B$ соединением их связкой «или», где «или» понимается в соединительном (хотя бы одно), а не в разделительном (либо - либо) смысле.

Согласно определению дизъюнкции получим таблицу истинности:

| $A$ | $B$ | $A \vee B$ |
|---|---|---|
| *Л* | *Л* | *Л* |
| *Л* | *И* | *И* |
| *И* | *Л* | *И* |
| *И* | *И* | *И* |

Другие, кроме $A \vee B$, обозначения дизъюнкции в литературе встречаются очень редко, например, дизъюнкцию $A \vee B$ обозначают иногда $A \cup B$ или $A + B$.

**4**. Прежде чем ввести следующую операцию, отметим, что в разговорной речи, а также в литературном языке мы привыкли к тому, что в высказываниях «если $A$, то $B$» между посылкой $A$ и заключением $B$ имеется определенная (обычно причинная) связь. Если же такого рода связи между $A$ и $B$ нет, то не всегда ясно, истинно либо ложно высказывание «если $A$, то $B$». Неясно, например, истинными или ложными будут высказывания:

1) если 2х2=4, то Л. Н. Толстой - автор романа «Война и мир»,

2) если 2х2≠4, то Л. Н. Толстой - автор романа «Война и мир»,

3) если 2х2≠4, то А. П. Чехов - автор романа «Война и мир».

Введем правила, по которым можно будет определять истинность высказывания «если $A$, то $B$», зная только истинностные значения $A$ и $B$ вне зависимости от того, существует ли между $A$ и $B$ какая-нибудь содержательная связь или нет.

*Импликация (следование)*- логическая операция, с помощью которой из двух данных высказываний $A$ и $B$ образуется новое высказывание, обозначаемое $A \Rightarrow B$, которое ложно тогда и только тогда, когда *посылка* $A$ истинна, а *заключение* $B$ ложно.

Высказывание $A \Rightarrow B$ читается «если $A$, то $B$» или «из $A$ следует $B$» и называется импликацией $A$ и $B$.

Согласно определению импликации получим таблицу истинности:

| $A$ | $B$ | $A \Rightarrow B$ |
|---|---|---|
| *Л* | *Л* | *И* |
| *Л* | *И* | *И* |
| *И* | *Л* | *Л* |
| *И* | *И* | *И* |

Из определения следует, что если посылка $A$ ложна, то вне зависимости, истинно или ложно $B$, высказывание $A \Rightarrow B$ считается истинным, т.е. из **лжи следует что угодно**. Таким образом, высказывания 1) - 3) будут считаться истинными.

Такое определение истинности высказывания «если $A$, то $B$» не противоречит обычной практике. Иногда встречается некоторое не истинностно-функциональное употребление связки «если..., то...», связанное с законами причинности и в так называемых условных контрафактических предложениях. Но мы будем использовать, введенную импликацию, только там где не используются законы причинности и условные контрафактические предложения.

Другие обозначения импликации (следования): $A \to B$, $A \supset B$.

**5**. *Эквивалентность* - логическая операция, при помощи которой из двух данных высказываний $A$ и $B$ образуется новое высказывание, обозначаемое $A \equiv B$, которое истинно тогда и только тогда, когда $A$ и $B$ принимают одинаковые истинностные значения.

Высказывание $A \equiv B$ читается «$A$ тогда и только тогда, когда $B$» и называется эквивалентностью $A$ и $B$. Другие обозначения для $A \equiv B$: $A \Leftrightarrow B$, $A \leftrightarrow B$, $A \sim B$.

Из определения эквивалентности получаем следующую таблицу истинности.

| $A$ | $B$ | $A \equiv B$ |
|---|---|---|
| *Л* | *Л* | *И* |
| *Л* | *И* | *Л* |
| *И* | *Л* | *Л* |
| *И* | *И* | *И* |

Таким образом, для произвольных данных высказываний, введены пять операций. С помощью этих операций из данных высказываний можно образовать новые, более сложные высказывания, истинность или ложность которых можно выяснить по таблицам истинности. Можно ввести и другие операции, но доказывается, что этих операций достаточно, более того, сложное высказывание выражается с использованием только некоторых из введенных выше операций.

Отметим еще раз, что мы ввели операции над произвольными высказываниями без учета их смыслового содержания. При этом можно выяснить истинностное значение полученных высказываний, не обращая внимания на смысловое содержание высказывания. Так, например, можно образовать высказывание: «Если Иванов - студент, то 1 сентября 2002 года в Воронеже шел дождь», и это высказывание будет ложно, когда высказывание «Иванов – студент» истинно, а 1 сентября 2002 года в Воронеже дождя не было, в остальных случаях будем считать, что рассматриваемое условное предложение истинно.

Также отметим, что приведенные контрпримеры (для конъюнкции и импликации) показывают, что логика высказываний позволяет моделировать не все возможные предложения языка, а только часть. Но эта часть достаточно объемна и важна. В других главах данной работы рассмотрены логики, которые расширяют возможности логики высказываний, учитывая, например, многозначность высказываний или их возможный нечеткий характер, зависимость от времени и т. п.

*Математические символы – те письмена, которыми Бог начертил великую книгу Природы. Не знающий их не в состоянии понять в ней ни одного слова и обречен вечно блуждать по лабиринту в кромешной тьме….*
*Г. Галилей*

## § 2. Пропозициональные буквы, связки и формы (формулы логики высказываний). Построение таблиц истинности

Символы $\neg, \&, \vee, \Rightarrow, \equiv$ называются *пропозициональными связками*.

Заглавные буквы алфавита ($A,B,C,...$) и те же буквы с числовыми индексами ($A_1,A_2,...,B_1,B_2,...,C_1,C_2,...$) называются *пропозициональными буквами*. Считается, что каждая пропозициональная буква может принимать значение *И* либо *Л*.

*Выражением* называется конечная последовательность определенных символов. Например, $\vee A\&\vee B$ - выражение, построенное из символов $\vee$, &, $A$ и $B$, а ?§!! - выражение, построенное из символов ?, § и !.

Пропозициональная форма представляет собой выражение, полученное по некоторым правилам из пропозициональных букв с помощью пропозициональных связок.

Индуктивное определение *пропозициональной формы*:

1) все пропозициональные буквы суть пропозициональные формы,

2) если $\boldsymbol{A}$ и $\boldsymbol{B}$ пропозициональные формы, то $(\neg\boldsymbol{A})$, $(\boldsymbol{A}\&\boldsymbol{B})$, $(\boldsymbol{A}\vee\boldsymbol{B})$, $(\boldsymbol{A}\Rightarrow\boldsymbol{B})$, $(\boldsymbol{A}\equiv\boldsymbol{B})$ тоже пропозициональные формы,

3) только те выражения являются пропозициональными формами, для которых это следует из пп.1,2.

Примеры пропозициональных форм: $A$, $(\neg B)$, $((A\&B)\Rightarrow(\neg C))$, $(((\neg A)\vee B)\equiv C)$.

Пропозициональные формы часто называют *формулами логики высказываний*.

Жирные заглавные буквы латинского алфавита ($\boldsymbol{A,B,C}$ **,...**) или те же буквы с числовыми индексами ($\boldsymbol{A}_1,\boldsymbol{A}_2,...,\boldsymbol{B}_1,\boldsymbol{B}_2,...,\boldsymbol{C}_1,\boldsymbol{C}_2,...$) употребляются для обозначения произвольных пропозициональных форм, тогда как обычное написание этих букв применяется лишь для пропозициональных букв.

*Истинностной функцией* от $n$ аргументов называется $n$-аргументная функция, принимающая одно из двух значений: *И* либо *Л*, когда ее аргументы пробегают те же значения.

Составное (сложное) высказывание, образованное с помощью введенных операций $\neg$, $\&$, $\vee$, $\Rightarrow$, $\equiv$ будет истинным либо ложным в зависимости от значений исходных высказываний. Следовательно, полученное составное высказывание порождает некоторую истинностную функцию.

Как определено выше, каждая пропозициональная буква может принимать значения *И* либо *Л*. Будем считать, что пропозициональные формы $(\neg A)$, $(A\&B)$, $(A\vee B)$, $(A\Rightarrow B)$ и $(A\equiv B)$ имеют те же таблицы истинности, что и обозначаемые таким образом высказывания (см.§1). Тогда каждому распределению (истинностных) значений *И* и *Л* пропозициональных букв, входящих в пропозициональную форму, соответствуют согласно таблицам истинности для пропозициональных связок некоторые истинностные значения этой пропозициональной формы.

Таким образом, каждая пропозициональная форма порождает некоторую функцию, принимающую значение *Л* или *И* в зависимости от истинностных значений пропозициональных букв в нее входящих, следовательно, каждая пропозициональная форма порождает некоторую истинностную функцию.

Заметим, что пропозициональная форма не является высказыванием. По определению пропозициональная форма - это выражение, построенное из пропозициональных букв, т.е. букв $A, B, C,..., A_1, A_2, ..., B_1, B_2, ..., C_1, C_2,...$ с помощью пропозициональных связок согласно правилам 1), 2), 3) и ничего более. В частном случае пропозициональные буквы могут обозначать высказывания, пропозициональные связки - логические операции, тогда пропозициональная форма будет обозначать некоторое высказывание. Истинностное значение полученного высказывания можно определить с помощью таблиц истинности.

Так как добавление каждой новой пропозициональной буквы увеличивает количество строк в таблице истинности вдвое, то пропозициональная форма, содержащая $n$ различных пропозициональных букв, имеет таблицу истинности с $2^n$ строками. Например, для формы $(((A\&B)\vee C)\Rightarrow A)$ имеем следующую таблицу истинности.

| $A$ | $B$ | $C$ | $(A\&B)$ | $((A\&B)\vee C)$ | $(((A\&B)\vee C)\Rightarrow A)$ |
|---|---|---|---|---|---|
| *Л* | *Л* | *Л* | *Л* | *Л* | *И* |
| *Л* | *Л* | *И* | *Л* | *И* | *Л* |
| *Л* | *И* | *Л* | *Л* | *Л* | *И* |
| *Л* | *И* | *И* | *Л* | *И* | *Л* |
| *И* | *Л* | *Л* | *Л* | *Л* | *И* |
| *И* | *Л* | *И* | *Л* | *И* | *И* |
| *И* | *И* | *Л* | *И* | *И* | *И* |
| *И* | *И* | *И* | *И* | *И* | *И* |

Составление таблицы истинности можно сократить, выписывая шаг за шагом под каждой пропозициональной связкой истинностные значения той составляющей пропозициональной формы, для которой применяется эта связка. Например, для той же формы *(((A&B)∨C)⇒A)* получаем таблицу:

| *A* | *B* | *C* | *((( A & B) ∨ C) ⇒ A)* |
|---|---|---|---|
| *Л* | *Л* | *Л* | *Л Л И* |
| *Л* | *Л* | *И* | *Л И Л* |
| *Л* | *И* | *Л* | *Л Л И* |
| *Л* | *И* | *И* | *Л И Л* |
| *И* | *Л* | *Л* | *Л Л И* |
| *И* | *Л* | *И* | *Л И И* |
| *И* | *И* | *Л* | *И И И* |
| *И* | *И* | *И* | *И И И* |

Следующий метод построения таблиц истинности, называют *алгоритмом Квайна*. В форме выбирается некоторая буква, например, та буква, которая чаще всего встречается в рассматриваемой форме. Выбранной букве (для формы ***D***=*(((A&B)∨C)⇒A)* это будет буква *A*) приписывается значение *И* либо *Л*. Далее проводят вычисления, где возможно, при выбранном значении этой буквы. Если *A=И*, то для формы ***D***=*(((A&B)∨C)⇒A)*, вне зависимости от значении букв *B* и *C*, легко получить, что ***D***=*И*. При *A=Л* и *C=Л* получим снова, что ***D***=*И*. Наконец, если *A=Л* и *C=И*, то ***D***=*Л*. В результате получим сокращенную запись таблицы истинности содержащую всего три строки (в данном случае результат не зависит от значений буквы *B,* а при *A=И* не зависит и от значений *C*):

| *A* | *B* | *C* | *(((A&B)∨C)⇒A)* |
|---|---|---|---|
| *Л* | | *Л* | *И* |
| *Л* | | *И* | *Л* |
| *И* | | | *И* |

## § 3. Упрощения в записях пропозициональных форм

Введем некоторые соглашения о более экономном употреблении скобок в записях форм. Эти соглашения облегчат нам чтение сложных выражений.

Во-первых, будем опускать в пропозициональной форме внешнюю пару скобок. (В случае пропозициональной буквы этой внешней пары скобок нет по определению).

Во-вторых, если форма содержит вхождения только одной бинарной связки (т.е. *&, ∨, ⇒* или *≡*), то для каждого вхождения этой связки опускаются внешние скобки у той из двух форм, соединяемых этим вхождением, которая стоит слева.

Пример. $A \vee B \vee C \vee A$ пишется вместо $(((A \vee B) \vee C) \vee A)$, а $B \Rightarrow B \Rightarrow A \Rightarrow (C \Rightarrow A)$ пишется вместо $(((B \Rightarrow B) \Rightarrow A) \Rightarrow (C \Rightarrow A))$.

В-третьих, договоримся считать связки упорядоченными следующим образом: $\neg$, $\&$, $\vee$, $\Rightarrow$, $\equiv$ и будем опускать во всякой пропозициональной форме все те пары скобок, без которых возможно восстановление этой формы на основе следующего правила.

Каждое вхождение знака $\neg$ относится к наименьшей пропозициональной форме, следующей за ним; после расстановки всех скобок, относящихся ко всем вхождениям знака $\neg$, каждое вхождение знака $\&$ связывает наименьшие формы, окружающие это вхождение; затем (т.е. после расстановки всех скобок, относящихся ко всем вхождениям знаков $\neg$ и $\&$) каждое вхождение знака $\vee$ связывает наименьшие формы, окружающие это вхождение, и аналогично для $\Rightarrow$ и $\equiv$. При применении этого правила к одной и той же связке мы продвигаемся слева направо.

Пример. В форме $A \equiv \neg A \& B \Rightarrow C \vee \neg D$ скобки восстанавливаются следующими шагами:

$A \equiv (\neg A) \& B \Rightarrow C \vee (\neg D)$,
$A \equiv ((\neg A) \& B) \Rightarrow C \vee (\neg D)$,
$A \equiv ((\neg A) \& B) \Rightarrow (C \vee (\neg D))$,
$A \equiv (((\neg A) \& B) \Rightarrow (C \vee (\neg D)))$,
$(A \equiv (((\neg A) \& B) \Rightarrow (C \vee (\neg D))))$.

Однако не всякая форма может быть записана без скобок. Например, нельзя опустить оставшиеся скобки в формах: $A \& (B \Rightarrow C)$, $A \Rightarrow (B \Rightarrow C)$, $\neg (A \vee B)$.

*Приходится порой простые мысли доказывать всерьез, как теоремы.*
*О. Сулейменов*[1]

## § 4. Тавтологии (общезначимые формулы). Противоречия

*Тавтологией* (*тождественно истинной пропозициональной формой или общезначимой формулой*) называется пропозициональная форма, которая принимает значение *И* при любой совокупности истинностных значений пропозициональных букв, входящих в нее.

Таблица истинности тавтологии имеет результирующий столбец, состоящий только из *И*.

Примером тавтологии является пропозициональная форма $(A \vee \neg A)$, в чем легко убедиться, составив таблицу истинности. Другие примеры тавтологий: $A \Rightarrow A$, $(A \equiv B) \equiv (B \equiv A)$.

*Противоречием* (*тождественно ложной пропозициональной формой*) называется пропозициональная форма, принимающая значение *Л* при любой

[1] *Казахстанский поэт.*

совокупности истинностных значений пропозициональных букв, входящих в нее.

Примеры противоречий: $A\&\overline{A}$, $\overline{(A\Rightarrow A)}$, $(A\equiv\overline{A})$.

Истинностная таблица для противоречия, очевидно, имеет в результирующем столбце только значения *Л*.

Очевидно, что пропозициональная форма ***A*** является тавтологией тогда и только тогда, когда $\overline{A}$ есть противоречие.

Тавтологию будем обозначать через ***T***, а противоречие - через ***П***.

Сформулируем и докажем две несложные теоремы.

**Теорема 1.1.** Если ***A*** и (***A*⇒*B***) - тавтологии, то ***B*** - тавтология.

**Доказательство**. Пусть ***A*** и (***A*⇒*B***) - тавтологии. Допустим, что при некотором распределении истинностных значений для пропозициональных букв, входящих в ***A*** и ***B***, ***B*** принимает значение *Л*. Поскольку ***A*** есть тавтология, то при этом распределении истинностных значений ***A*** принимает значение *И*. Тогда (***A*⇒*B***) получит значение *Л*. Это противоречит предположению о том, что (***A*⇒*B***) есть тавтология. Теорема доказана.

**Теорема 1.2.** Если ***A*** есть тавтология, содержащая пропозициональные буквы $A_1, A_2,..., A_n$, и ***B*** получается из ***A*** подстановкой в ***A*** пропозициональных форм $\boldsymbol{A_1}, \boldsymbol{A_2}, \ldots \boldsymbol{A_n}$ вместо букв $A_1, A_2,..., A_n$ соответственно, то ***B*** есть тавтология, т.е. подстановка в тавтологию приводит к тавтологии.

**Доказательство**. Пусть задано произвольное распределение истинностных значений для пропозициональных букв, входящих в ***B***. Формы $\boldsymbol{A_1}, \boldsymbol{A_2}, \ldots, \boldsymbol{A_n}$ примут тогда некоторые значения $x_1, x_2,..., x_n$ (каждое $x_i$ есть *И* или *Л*). Если мы придадим значения $x_1, x_2,..., x_n$ соответственно буквам $A_1, A_2,..., A_n$ , то так как ***A*** есть тавтология, то ***A*** будет истинно, и это же значение принимает ***B***. Таким образом, при произвольных значениях пропозициональных букв форма ***B*** принимает значение *И*, что и требовалось доказать.

Ясно, что при подстановке в пропозициональную форму вместо пропозициональных букв высказываний, мы получим некоторое высказывание.

Высказывание, которое получается из какой-либо тавтологии посредством подстановки высказываний вместо пропозициональных букв, при условии, что вхождение одной и той же буквы замещается одним и тем же высказыванием, называется *логически истинным высказыванием*. Это высказывание истинно в силу своей формы, а не в силу своего содержания. Например, высказывание: "Если Иванов - студент, то Иванов - студент"

всегда истинно (логически истинно), в то время как высказывание: "Иванов - студент", если и истинно, то в силу уже других причин.

Высказывание, которое можно получить с помощью подстановки в противоречие, называется *логически ложным высказыванием*. Примером логически ложного высказывания может служить высказывание: "2x2=4 и 2x2≠4", где имеет место одновременно какое-то высказывание и отрицание этого же высказывания.

Пропозициональная форма называется *выполнимой* если она принимает значения *И* хотя бы для одной совокупности значений пропозициональных букв, в нее входящих.

Например, *A&B* является выполнимой пропозициональной формой, так как принимает значения *И*, когда *A=И* и *B=И*, а форма *A&* $\overline{A}$ не будет выполнимой, так как всегда ложна.

Очевидно, что пропозициональная форма ***A*** выполнима тогда и только тогда, когда ***A*** не является противоречием.

*Проблема разрешимости* (алгебры высказываний) состоит в следующем. Существует ли правило, позволяющее для каждой пропозициональной формы ***A*** конечным числом действий выяснить, является ***A*** выполнимой или нет.

Ясно, что выяснение выполнимости ***A*** равносильно выяснению, является ли ***A*** противоречием или нет, что, в свою очередь, равносильно выяснению, является ли $\overline{A}$ тавтологией или нет. Таким образом, если есть метод, позволяющий для произвольной формы конечным числом действий выяснить, тавтология это или нет, то можно решить вопрос выполнима или нет произвольная форма ***A***. Для этого достаточно выяснить $\overline{A}$ - тавтология или нет. Если $\overline{A}$ - тавтология, то ***A*** - противоречие, следовательно, ***A*** невыполнимо, если же $\overline{A}$ - не тавтология, то ***A*** выполнимо.

Проблема разрешимости (алгебры высказываний) полностью решается, например, с помощью таблиц истинности. Если пропозициональная форма содержит *n* различных букв, то, как известно, ее таблица истинности имеет $2^n$ строк. При больших значениях *n* составление таблиц истинности и выяснение выполнимости по ним является громоздкой операцией. Для решения проблемы разрешимости существуют и другие способы, основанные на приведении к так называемым нормальным формам. Эти формы и способы решения с их помощью проблемы разрешимости будут рассмотрены ниже.

*Шел я садом однажды и вдруг увидал,*
*Как делили коврижку Сова и Шакал.*
*И коврижку Шакал проглотил целиком,*
*А Сове только блюдечко дал с ободком.*

*А потом предложил ей: «Закончим дележ –*
*Ты возьми себе ложку, я – вилки и нож».*
*И наевшись, улегся Шакал на траву,*
*Но сперва на десерт проглотил он…*

*Л. Кэрролл (Приключения Алисы в стране чудес)*[1]

## § 5. Равносильность пропозициональных форм

Пропозициональные формы ***A*** и ***B*** называются *равносильными*, если при каждой совокупности значений всех пропозициональных букв, входящих в ***A*** и ***B***, эти формы принимают одинаковые истинностные значения.

Например, форма $\neg A \vee B$ равносильна форме $A \Rightarrow B$, в чем легко убедиться с помощью таблиц истинности:

| $\neg$ | $A$ | $\vee$ | $B$ |
|---|---|---|---|
| *И* | *Л* | ***И*** | *Л* |
| *И* | *Л* | ***И*** | *И* |
| *Л* | *И* | ***Л*** | *Л* |
| *Л* | *И* | ***И*** | *И* |

| $A$ | $\Rightarrow$ | $B$ |
|---|---|---|
| *Л* | ***И*** | *Л* |
| *Л* | ***И*** | *И* |
| *И* | ***Л*** | *Л* |
| *И* | ***И*** | *И* |

В этих таблицах результирующие столбцы совпадают, т.е. при одинаковых значениях букв $A$ и $B$ значения форм $\neg A \vee B$ и $A \Rightarrow B$ равны, следовательно, эти формы равносильны. Далее, форма $A \vee A\&B$ равносильна $A$. Действительно, имеем следующую таблицу:

| $A$ | $\vee$ | $A$ | $\&$ | $B$ | | $A$ |
|---|---|---|---|---|---|---|
| *Л* | ***Л*** | *Л* | *Л* | *Л* | | ***Л*** |
| *Л* | ***Л*** | *Л* | *Л* | *И* | | ***Л*** |
| *И* | ***И*** | *И* | *Л* | *Л* | | ***И*** |
| *И* | ***И*** | *И* | *И* | *И* | | ***И*** |

Также, очевидно, $A \vee \neg A$ равносильно $B \vee \neg B$.

При определении равносильности двух форм не обязательно предполагать, что они содержат одни и те же пропозициональные буквы. Так, в последних примерах имеем случаи, когда в равносильные формы входят разные буквы. При этом, если какая-нибудь пропозициональная буква входит только в одну из двух равносильных форм, то эта форма при всех значениях этой буквы принимает одно и то же значение, если значения других фиксированы. Следовательно, хотя эта буква и входит в форму, но истинностная функция, определенная формой, от этой буквы не зависит.

Высказывание "***A*** равносильно ***B***" будем обозначать следующим образом: $\boldsymbol{A \sim B}$.

---

[1] *Здесь и далее, цитаты из произведений Л. Кэрролла приводятся по переводу с английского, выполненного Н. Димуровой, в котором приведены стихи в переводах С. Маршака, Д. Орловской и О. Седаковой.*

Пусть ***A, B, C*** - произвольные пропозициональные формы. Отношение равносильности пропозициональных форм, как легко видеть, обладает следующими свойствами:

1) $A \sim A$ - рефлексивность;

2) если $A \sim B$, то $B \sim A$ - симметричность;

3) если $A \sim B$ и $B \sim C$ , то $A \sim C$ - транзитивность.

Следовательно, отношение равносильности является отношением эквивалентности и порождает разбиение множества пропозициональных форм на непересекающиеся классы. В каждый класс попадают равносильные между собой пропозициональные формы. Докажем теорему.

**Теорема 1.3.** Пропозициональные формы ***A*** и ***B*** равносильны тогда и только тогда, когда $A \equiv B$ является тавтологией.

**Доказательство.** Необходимость. Пусть ***A*** и ***B*** равносильны, следовательно, они при каждой совокупности значений всех пропозициональных букв, входящих в них, принимают одинаковые истинностные значения, тогда по определению связки ≡ форма $A \equiv B$ всегда принимает значение *И*, т.е. является тавтологией.

Достаточность. Пусть $A \equiv B$ тавтология, т.е. принимает всегда значение *И*. Это означает, что ***A*** и ***B*** имеют всегда одинаковые истинностные значения, т.е. они равносильны. Теорема доказана.

*В природе существует внутренне присущая ей скрытая гармония, отражающаяся в наших умах в виде простых математических знаков.*

*Г. Вейль*

## § 6. Важнейшие пары равносильных пропозициональных форм

Пусть *A, B, C* - пропозициональные буквы, ***T***- тавтология и ***П*** - противоречие. Используя таблицу истинности, легко показать, что

*1)* $\neg(\neg A) \sim A$ равносильно *A*.

Если под *A* понимать обозначение некоторого высказывания, то получаем, что двойное отрицание высказывания *A* означает то же, что и высказывание *A*. Полученное соотношение между $\neg(\neg A)$ и *A* называют *законом двойного отрицания*.

Аналогичным образом можно показать, что имеют место следующие законы.

$$\left.\begin{array}{l} 2)\ A \,\&\, B \sim B \,\&\, A; \\ 3)\ A \vee B \sim B \vee A; \end{array}\right\}$$ *законы коммутативности;*

4) $(A \& B) \& C \sim A \& (B \& C);$
5) $(A \vee B) \vee C \sim A \vee (B \vee C);$
} *законы ассоциативности;*

6) $A\&(B \vee C) \sim A\&B \vee A\&C$ - *первый закон дистрибутивности;*
7) $A \vee B\&C \sim (A \vee B)\&(A \vee C)$ - *второй закон дистрибутивности;*

8) $\neg(A\&B) \sim \neg A \vee \neg B,$
9) $\neg(A \vee B) \sim \neg A\& \neg B,$
} *законы де Моргана;*

10) $A\&A \sim A,$
11) $A \vee A \sim A,$
} *законы идемпотентности;*

12) $A \vee \neg A \sim \boldsymbol{T}$ - *закон исключенного третьего;*
13) $A\& \neg A \sim \boldsymbol{\Pi}$ - *закон противоречия;*

14) $A\&\boldsymbol{T} \sim A;$
15) $A \vee \boldsymbol{T} \sim \boldsymbol{T};$
16) $A\&\boldsymbol{\Pi} \sim \boldsymbol{\Pi};$
17) $A \vee \boldsymbol{\Pi} \sim A;$
} *свойство операций с* ***T*** *и с* $\boldsymbol{\Pi}$;

18) $A \vee A\&B \sim A;$
19) $A\&(A \vee B) \sim A;$
} *законы поглощения;*

20) $A \Rightarrow B \sim \neg B \Rightarrow \neg A$ - *закон контропозиции.*

Как уже замечено выше, соотношения *1) - 20)* доказываются с помощью таблиц истинности.

Можно показать, что соотношения *1) - 20)* будут иметь место и тогда, когда вместо пропозициональных букв *A*, *B* и *C* будут подставлены произвольные пропозициональные формы. Соотношения *1) - 20)* позволяют находить для заданных пропозициональных форм равносильные упрощенные формы или равносильные формы, имеющие более удобный с некоторых позиций вид. Из этих же соотношений видно, что над пропозициональными формами можно производить преобразования: раскрытие скобок, заключение в скобки, вынесение за скобки общего множителя.

Из соотношений *2) - 6)* видно, что операция *&* напоминает умножение (обладает некоторыми свойствами умножения), а $\vee$ - сложение, поэтому часто конъюнкцию двух высказываний называют (логическим) произведением их, а дизъюнкцию - (логической) суммой.

*Битый – правду говорит*
*Молвь людей простых –*
*Стоит двух, кто не был бит,*
*Грамотей – троих.*
*П.-Ж. Беранже*

## § 7. Зависимости между пропозициональными связками

Связки $\neg$, $\&$, $\vee$, $=>$, $\equiv$ не являются независимыми друг от друга в том смысле, что одни из них можно выражать через другие так, что при этом получаются равносильные пропозициональные формы.

Например, связка $\equiv$ может быть выражена через связки $\Rightarrow$ и $\&$ на основании соотношения

$$A \equiv B \sim (A \Rightarrow B) \,\&\, (B \Rightarrow A). \qquad (1.1)$$

Для доказательства (1.1) достаточно составить таблицы истинности и убедиться, что результирующие столбцы этих таблиц совпадают.

Для импликации имеем:

$$A \Rightarrow B \sim \neg A \vee B. \qquad (1.2)$$

Таким образом, связку $\equiv$ можно выразить через $\neg$, $\&$ и $\vee$:

$$A \equiv B \sim (\neg A \vee B) \,\&\, (\neg B \vee A). \qquad (1.3)$$

Так как $A$ равносильно $\neg(\neg A)$, то $A\&B$ равносильно $\neg(\neg A)\&\neg(\neg B)$, а последнее согласно закону де Моргана равносильно $\neg(\neg A \vee \neg B)$, следовательно,

$$A\&B \sim \neg(\neg A \vee \neg B). \qquad (1.4)$$

Из (1.4) видно, что $\&$ можно выразить через $\neg$ и $\vee$. Покажем, что $\vee$ можно выразить через $\neg$ и $\&$. Действительно, так как $A$ равносильно $\neg(\neg A)$, то $A \vee B$ равносильно $\neg(\neg A) \vee \neg(\neg B)$, последнее по закону де Моргана равносильно $\neg(\neg A \,\&\, \neg B)$. Итак,

$$A \vee B \sim \neg(\neg A \,\&\, \neg B). \qquad (1.5)$$

Из соотношения (1.2) заменой $\neg A$ на $A$ получаем, что

$$A \vee B \sim \neg A \Rightarrow B. \qquad (1.6)$$

Т.е. $\vee$ можно выразить через $\neg$ и $\Rightarrow$.

Изложенное показывает, что одни связки могут быть выражены через другие. Имеют место следующие теоремы.

**Теорема 1.4**. Для каждой пропозициональной формы ***A*** существует равносильная ей форма, содержащая только связки $\neg$, $\&$, $\vee$, причем связка $\neg$ относится только к пропозициональным буквам.

**Доказательство**. Связки $\Rightarrow$ и $\equiv$ можно исключить согласно соотношениям (1.2) и (1.3). При этом останутся только связки $\neg$, $\&$, $\vee$. Если пропозициональная связка $\neg$ стоит перед некоторой скобкой, например, $\neg(A\&B \vee C \vee \neg B)$, то на основании законов де Моргана можно внести $\neg$ под скобки, при этом связка $\&$ меняется на $\vee$, а $\vee$ на $\&$, а связки $\Rightarrow$ и $\equiv$ нигде не появляются. Внеся $\neg$ под скобки, перед которыми они стоят, добьемся, чтобы $\neg$ относилась только к пропозициональным буквам. Теорема доказана.

**Теорема 1.5**. Для каждой пропозициональной формы ***A*** существует равносильная ей форма, содержащая либо только связки $\neg$, $\&$, либо только $\neg$, $\vee$, либо только $\neg$, $\Rightarrow$.

**Доказательство.** Покажем, что можно оставить только связки $\neg$ и $\&$. Связки $\Rightarrow$ и $\equiv$ можно исключить (если они есть) на основании соотношений (1.2) и (1.3), а затем по (1.5) исключим $\vee$. В результате останутся только $\neg$ и $\&$. Остальные случаи доказываются аналогичным образом на основании соотношений (1.1) - (1.6).

Будем рассматривать пропозициональные формы, содержащие только связки $\neg$, $\&$, $\vee$. Как уже установлено выше, всякая пропозициональная форма может быть приведена преобразованиями равносильности к такому виду.

Будем говорить, что связка $\&$ *двойственна связке* $\vee$, и наоборот.

Пропозициональные формы $\boldsymbol{A}$ и $\boldsymbol{A}^*$ называются *двойственными*, если одна получается из другой заменой каждой связки $\&$ и $\vee$ на двойственную.

Например, если $\boldsymbol{A} = (A \vee \neg B) \& C$, то $\boldsymbol{A}^* = (A \,\&\, \neg B) \vee C$.

Отношение двойственности взаимно: если $\boldsymbol{A}$ двойственно $\boldsymbol{A}^*$, то $\boldsymbol{A}^*$ двойственно $\boldsymbol{A}$. Следующую теорему считают *законом двойственности*.

**Теорема 1.6.** Если пропозициональные формы $\boldsymbol{A}$ и $\boldsymbol{B}$ равносильны, то и двойственные им формы $\boldsymbol{A}^*$ и $\boldsymbol{B}^*$ также равносильны.

**Доказательство.** Пусть $\boldsymbol{A}$ и $\boldsymbol{B}$ равносильны, а $A_1, A_2, \ldots, A_n$ - буквы, входящие в $\boldsymbol{A}$ или $\boldsymbol{B}$. Будем считать, что $A_1, A_2, \ldots, A_n$ входят и в $\boldsymbol{A}$, и в $\boldsymbol{B}$. Если бы это было не так, например, $\boldsymbol{B}$ не содержала бы $A_k (1 \le k \le n)$, входящего в $\boldsymbol{A}$, то $\boldsymbol{B}$ можно заменить равносильной формой $\boldsymbol{B} \vee A_k \& \neg A_k$, содержащей эту букву. Таким образом, всегда можем добиться, чтобы $\boldsymbol{A}$ и $\boldsymbol{B}$ содержали все буквы $A_1, A_2, \ldots, A_n$.

По условию

$$\boldsymbol{A}(A_1, A_2, \ldots, A_n) \sim \boldsymbol{B}(A_1, A_2, \ldots, A_n). \qquad (1.7)$$

Если формы $\boldsymbol{A}$ и $\boldsymbol{B}$ равносильны, то, очевидно, равносильны и их отрицания, поэтому из (1.7) получим, что

$$\neg\boldsymbol{A}(A_1, A_2, \ldots, A_n) \sim \neg\boldsymbol{B}(A_1, A_2, \ldots, A_n). \qquad (1.8)$$

В пропозициональных формах соотношения (1.8) добьемся, чтобы $\neg$ относилась только к буквам. При этом согласно законам де Моргана связки $\&$ и $\vee$ поменяются на двойственные. Следовательно, получим

$$\boldsymbol{A}^*(\neg A_1, \neg A_2, \ldots, \neg A_n) \sim \boldsymbol{B}^*(\neg A_1, \neg A_2, \ldots, \neg A_n) \qquad (1.9)$$

По определению равносильности форм равносильность $\boldsymbol{A}^*(\neg A_1, \neg A_2, \ldots, \neg A_n)$ и $\boldsymbol{B}^*(\neg A_1, \neg A_2, \ldots, \neg A_n)$ означает, что они принимают одинаковые значения при любых совокупностях значений букв $A_1, A_2, \ldots, A_n$. Поэтому, если вместо букв $A_1, A_2, \ldots, A_n$ подставить $\neg A_1, \neg A_2, \ldots, \neg A_n$, то формы останутся равносильными. Учитывая, что $\neg\neg\boldsymbol{A}$ равносильно $\boldsymbol{A}$, из (1.9) получим $\boldsymbol{A}^*(A_1, A_2, \ldots, A_n) \sim \boldsymbol{B}^*(A_1, A_2, \ldots, A_n)$, что и требовалось доказать.

*Каких цветов в саду весеннем нет!*
*В. Шекспир*

## § 8. Нормальные формы

Известно, что в математическом анализе среди всевозможных функций выделяют элементарные (основные) функции: степенные, показательные, тригонометрические и т.п. Среди пропозициональных форм, выделим некоторые как элементарные, а далее из них можно получать более сложные.

*Элементарной суммой (элементарным произведением)* называют дизъюнкцию (конъюнкцию) пропозициональных букв либо их отрицаний. Одну букву тоже будем рассматривать как элементарную сумму или как элементарное произведение. Элементарную сумму часто называют *дизъюнктом,* а слагаемые этой суммы называются *литералами* (*литерами*).

Примеры элементарных сумм: $A,\ A \vee B,\ A \vee B \vee \overline{A} \vee C$.

Примеры элементарных произведений: $\overline{A},\ A\&B,\ \overline{A}\&C\&A\&B$.

**Теорема 1.7.** Чтобы элементарная сумма была тавтологией (общезначимой формулой), необходимо и достаточно, чтобы в ней содержалась хотя бы одна пара слагаемых, из которых одно есть некоторая буква, а другое - отрицание этой буквы.

**Доказательство.** Достаточность. Пусть такая пара букв существует, т.е. сумма имеет вид $A \vee \overline{A} \vee \ldots$ . Для нас важно, что имеются слагаемые $A$ и $\overline{A}$, остальные слагаемые могут быть, но могут и отсутствовать. Форма $A$ и $\overline{A}$ всегда принимает значение *И*, поэтому и вся элементарная сумма будет *И* при любых значениях букв, в нее входящих. Следовательно, наша элементарная сумма - тавтология.

Необходимость. Пусть данная элементарная сумма - тавтология. Допустим, что такой пары слагаемых нет. В таком случае каждой букве, стоящей без знака отрицания, можно дать значение *Л*, а буквам, стоящим со знаком отрицания, - значение *И*. Это возможно осуществить, ибо одна и та же буква не входит одновременно без отрицания и с отрицанием. После указанной подстановки каждое слагаемое примет значение *Л*, следовательно, и вся элементарная сумма примет значение *Л* и, значит, элементарная сумма не является тавтологией, что противоречит условию. Полученное противоречие и доказывает необходимость.

Также легко доказать следующую теорему.

**Теорема 1.8.** Для того, чтобы элементарное произведение $A$ было противоречием, необходимо и достаточно, чтобы в нем содержалась хотя бы одна пара множителей, из которых один множитель является отрицанием другого.

*Дизъюнктивной нормальной формой (д.н.ф.)* называется дизъюнкция элементарных произведений. Одно элементарное произведение тоже будем считать д.н.ф. Примеры д.н.ф.: $A \vee \neg B, A \vee B\&C, \neg A \vee A\&C \vee A\&B\& \neg C$.

**Теорема 1.9.** Для каждой пропозициональной формы существует равносильная ей д.н.ф. (не единственная).

**Доказательство.** Ранее было показано, что для любой формы существует равносильная ей форма, содержащая только связки $\neg$, $\&$, $\vee$, *причем* связка $\neg$ относится только к отдельным буквам. Еще раньше было замечено, что с формой, содержащей связки $\&$, $\vee$, можно проводить операции раскрытия скобок (см. законы дистрибутивности). В результате получаем дизъюнкцию элементарных произведений, т.е. д.н.ф.

Пример. Пусть задана формула $\neg(A \equiv B)\&C$. Исключим связку $\equiv$, затем $\Rightarrow$:

$\neg((A \Rightarrow B)\&(B \Rightarrow A))\&C$,

$\neg((\neg A \vee B)\&(\neg B \vee A))\&C$.

Теперь добьемся, чтобы $\neg$ относилась только к пропозициональным буквам:

$(A\& \neg B \vee B\& \neg A)\&C$.

Раскрыв скобки, получим $A\& \neg B\&C \vee B\& \neg A\&C$. Последнее и есть д.н.ф. Полученная д.н.ф., очевидно, равносильна следующей д.н.ф.: $A\& \neg B\&C \vee \vee \neg A\&B\&C \vee A\& \neg A$. Из примера видно, что д.н.ф., равносильная заданной форме, определяется не единственным образом.

**Теорема 1.10**. Для того, чтобы форма $\boldsymbol{A}$ была противоречием, необходимо и достаточно, чтобы равносильная ей д.н.ф. содержала в каждом слагаемом хотя бы одну пару множителей, из которых один - некоторая пропозициональная буква, а второй - отрицание этой буквы.

**Доказательство**. Пусть для $\boldsymbol{A}$ равносильной ей д.н.ф. является форма

$$\boldsymbol{B_1 \vee B_2 \vee \ldots \vee B_k} \ (k \geq 1), \qquad (1.10)$$

где $B_i$ $(1 \leq i \leq k)$ есть элементарное произведение. Дизъюнкция (1.10) будет противоречием тогда и только тогда, когда будет противоречием каждая $\boldsymbol{B_i}$ $(1 \leq i \leq k)$. $\boldsymbol{B_i}$ - элементарное произведение и по теореме 1.8 будет противоречием тогда и только тогда, когда содержит хотя бы одну пару множителей, из которых один - некоторая буква, а второй - отрицание этой буквы. Теорема доказана.

**Следствие 1.1**. Форма $\boldsymbol{A}$ будет выполнимой, если равносильная ей д.н.ф. содержит хотя бы одно слагаемое, в котором нет таких множителей, что один

из них - некоторая пропозициональная буква, а другой множитель - отрицание этой буквы.

Пропозициональная форма называется *конъюнктивной нормальной формой (к.н.ф.)*, если она представляет собой конъюнкцию элементарных сумм. Легко доказать следующую теорему.

**Теорема 1.11**. Для каждой пропозициональной формы существует равносильная ей к.н.ф. (не единственная).

Можно сформулировать правила для нахождения д.н.ф. и к.н.ф., равносильных заданной форме. Пусть задана форма ***A***:

1) исключить из ***A*** все связки, кроме $\neg, \&, \vee$,

2) добиться, чтобы $\neg$ относилась только к пропозициональным буквам,

3) если раскрыть скобки по первому дистрибутивному закону, то получится д.н.ф.,

4) если сгруппировать в скобки, пользуясь вторым дистрибутивным законом, то получится к.н.ф.

Легко доказать следующую теорему.

**Теорема 1.12**. Для того, чтобы пропозициональная форма ***A*** была тавтологией, необходимо и достаточно, чтобы равносильная ей к.н.ф. содержала в каждом множителе хотя бы одну букву вместе с отрицанием этой же буквы.

Приведенная теорема позволяет очень просто выяснить является ли форма тавтологией или нет. Для этого достаточно построить к.н.ф.

Также просто по теореме 1.12 выяснить выполнима форма ***A*** или нет. Для этого находим к.н.ф. для $\neg A$ и если найденная к.н.ф. тавтология, то ***A*** не выполнима, если же найденная к.н.ф. не тавтология, то форма ***A*** выполнима.

## § 9. Совершенные нормальные формы

*Совершенной дизъюнктивной нормальной формой (с.д.н.ф.)* пропозициональной формы $A(A_1,A_2,\ldots,A_n)$ называется д.н.ф. этой формы удовлетворяющая следующим условиям:

-нет одинаковых слагаемых;

-в каждое слагаемое входят все буквы $A_1,A_2,\ldots,A_n$ один и только один раз (и только они) с отрицанием либо без отрицания.

С.д.н.ф. для ***A*** можно построить по таблице истинности этой формы. Для этого выбираем строки, где ***A*** равна *И*; пусть это будут строки $k_1,k_2,\ldots,k_m$. Для каждой выбранной строки $k_i$, $1 \le i \le m$, строим элементарное произведение

(*конституенту единицы*) $K_i$ следующим образом. Если в выбранной строке $k_i$ буква $A_j$ принимает значение *И*, то в $K_i$ она входит без отрицания, если же $A_j$=*Л*, то в $K_i$ она входит с отрицанием. Дизъюнкция построенных произведений и будет с.д.н.ф. и можно доказать, что эта с.д.н.ф. равносильна ***A.***

| *A* | *B* | *C* | ***A(A,B,C)*** |
|---|---|---|---|
| *Л* | *Л* | *Л* | *И* |
| *Л* | *Л* | *И* | *И* |
| *Л* | *И* | *Л* | *И* |
| *Л* | *И* | *И* | *Л* |
| *И* | *Л* | *Л* | *И* |
| *И* | *Л* | *И* | *Л* |
| *И* | *И* | *Л* | *Л* |
| *И* | *И* | *И* | *И* |

Рассмотрим пример на построение *с.д.н.ф.* Пусть форма ***A(A,B,C)*** ложна тогда и только тогда, когда ложен один и только один из аргументов. Найти с.д.н.ф. для ***A(A,B,C)***.

Решение. Составим таблицу истинности.

Выберем строки, где ***A*** принимает значение *И*, т.е. строки с номерами: *1, 2, 3, 5* и *8*. Для первой строки элементарное произведение представляется в виде конъюнкции $\neg A \& \neg B \& \neg C$, для второй - $\neg A \& \neg B \& C$. Построив, таким образом элементарные произведения для этих строк, получим с.д.н.ф.:

$$\neg A \& \neg B \& \neg C \vee \neg A \& \neg B \& C \vee \neg A \& B \& \neg C \vee A \& \neg B \& \neg C \vee A \& B \& C.$$

Второй метод нахождения с.д.н.ф. – метод равносильных преобразований, который состоит в следующем. Для заданной формы ***A*** находим д.н.ф. (которая всегда существует) и пусть д.н.ф. равна:

$\boldsymbol{D}_1 \vee \boldsymbol{D}_2 \vee \ldots \vee \boldsymbol{D}_m \ (m \geq 1)$,

где $\boldsymbol{D}_i (1 \leq i \leq m)$ – элементарное произведение. Построенная д.н.ф. может удовлетворять требуемым условиям, тогда это с.д.н.ф.

Если в некоторое $\boldsymbol{D}_i$ входит некоторая буква вместе с ее отрицанием, то $\boldsymbol{D}_i$ – противоречие и из д.н.ф. $\boldsymbol{D}_i$ можно исключить. Если при такой процедуре нужно отбросить все слагаемые из д.н.ф., то ***A*** – противоречие и с.д.н.ф. не существует.

Если в некоторое $\boldsymbol{D}_i$ не входит одна из букв $A_j$ формы ***A***, то заменяем $\boldsymbol{D}_i$ на равносильную:

$(\boldsymbol{D}_i \& A_j) \vee (\boldsymbol{D}_i \& \neg A_j)$.

Таким образом, добиваемся, чтобы каждое слагаемое содержало все аргументы формы ***A***.

Если в полученной форме окажутся одинаковые слагаемые, то оставляем только одно из них. В результате получим *с.д.н.ф*

*Совершенной конъюнктивной нормальной формой (с.к.н.ф.)* пропозициональной формы $\boldsymbol{A}(A_1, A_2, \ldots, A_n)$ называется к.н.ф. этой формы удовлетворяющая следующим условиям:

-нет одинаковых множителей;

-в каждый множитель входят все буквы из пропозициональной формы ***A*** один и только один раз (и только они) с отрицанием, либо без отрицания.

С.к.н.ф. для пропозициональной формы ***A*** можно построить по таблице истинности этой формы. Для этого выбираем строки, где ***A***=*Л*. Для каждой строки, где ***A***=*Л* строим элементарную сумму (*конституенту нуля*) $K^0$

следующим образом. Если в выбранной строке буква $A_j$ принимает значение *И*, то в $K^0$ она входит с отрицанием, если $A_j$=*Л*, то $A_j$ входит в $K^0$ без отрицания. Конъюнкция построенных конституент нуля и будет с.к.н.ф.

Рассмотрим пример на построение с.к.н.ф. для пропозициональной формы, рассмотренной выше. Решение. Таблица истинности уже построена. Выберем строки, где ***A*** принимает значение *Л*, т. е. строки с номерами: *4, 6* и *7*. Для четвертой строки элементарная сумма (конституента нуля) представляется в виде $A \vee \neg B \vee \neg C$; для шестой в виде: $\neg A \vee B \vee \neg C$, а для седьмой - $\neg A \vee \neg B \vee C$. В результате получим с.к.н.ф.:

$(A \vee \neg B \vee \neg C) \& (\neg A \vee B \vee \neg C) \& (\neg A \vee \neg B \vee C)$.

Второй метод построения с.к.н.ф. – метод равносильных преобразований – заключается в следующем.

Для заданной формы ***A*** находим к.н.ф., которая имеет вид $\boldsymbol{K}_1 \& \boldsymbol{K}_2 \& \ldots \& \boldsymbol{K}_m$, затем добиваемся выполнения указанных выше условий .

Если в некоторое $\boldsymbol{K}_i$ входит некоторая буква вместе с ее отрицанием, то $\boldsymbol{K}_i$ – тавтология и из к.н.ф. множитель $\boldsymbol{K}_i$ можно исключить. Если при такой процедуре нужно отбросить все множители из к.н.ф., то ***A*** – тавтология и с.к.н.ф. не существует.

Если некоторый множитель к.н.ф., например $\boldsymbol{K}_i$, не содержит букву *A*, то вводим ее согласно равносильности

$\boldsymbol{K}_i \sim (\boldsymbol{K}_i \vee A) \& (\boldsymbol{K}_i \vee \neg A)$.

Таким образом, добиваемся, чтобы каждый множитель к.н.ф. содержал все аргументы исходной формулы ***A***.

Если в некотором множителе окажутся одинаковые слагаемые, то оставляем только одно из них. Если в полученной форме окажутся одинаковые множители, то оставляем только один из них. В результате получим с.к.н.ф.

## § 10. Булева (переключательная) функция

*Булевой переменной* назовем переменную, которая принимает одно из двух возможных значений. Ясно, что высказывание можно рассматривать как частный случай булевой переменной, ибо высказывание принимает только одно из двух значений: *Л* или *И*.

Функция $f(A_1,A_2,\ldots,A_n)$ называется *булевой (переключательной) функцией*, если она может принимать одно из двух возможных значений, когда ее аргументы принимают тоже одно из этих двух значений.

Очевидно, что истинностная функция является частным случаем булевой функции.

Значения булевой функции и булевых переменных можно обозначать любыми символами, например, + и -, либо *1* и *0*. В дальнейшем значения булевых переменных и функций будем обозначать через *1* и *0*.

При исследовании высказываний, истинностных функций и пропозициональных форм нигде не использовалась природа значений "истина" (*И*) и "ложь" (*Л*). Тогда всюду *И* и *Л* можно рассматривать как символы, служащие для различения двух значений. Поэтому *И* можно переобозначить через *1*, а *Л* - через *0*, и все результаты, полученные для истинностных функций, будут справедливы для булевых функций. Более того, булевы функции можно отождествить с истинностными функциями.

Выражения $(\overline{A})$, $(A\&B)$, $(A\vee B)$, $(A\Rightarrow B)$, $(A\equiv B)$ можно рассматривать как булевы функции, значения которых определяются по таблицам построенным для пропозициональных форм $(\overline{A})$, $(A\&B)$, $(A\vee B)$, $(A\Rightarrow B)$, $(A\equiv B)$ соответственно, если в них всюду заменить *И* на *1*, а *Л* на *0*. Эти таблицы будем тоже называть таблицами истинности.

Очевидно, что пропозициональную форму можно рассматривать как булеву функцию, значения которой определяются по таблицам истинности.

Можно показать, что число различных булевых функций от $n$ переменных равно $N=2^{2^n}$. Например: для $n=1$ $N=4$, для $n=2$ $N=16$, для $n=3$ $N=256$, для $n=5$ $N$ - более четырех миллиардов.

Булеву функцию можно задать с помощью таблиц, аналитически, графически и т.п.

## § 11. Приложение алгебры высказываний к анализу и синтезу контактных (переключательных) схем

Алгебра высказываний находит весьма широкое практическое применение. Рассмотрим приложение алгебры высказываний к анализу и синтезу контактных (переключательных) схем.

*Контакты (переключатели)* можно рассматривать как переменные и обозначать буквами *A, B, C,...* или теми же буквами с числовыми индексами: $A_1, A_2, ..., B_1, B_2, ..., C_1, C_2, ...$ . Каждая из переменных может принимать одно и только одно из двух возможных значений: если контакт *A* разомкнут, то по определению $A=0$, если контакт *A* замкнут, то по определению $A=1$.

Под *контактной (переключательной) схемой* понимается схема, состоящая из замкнутых и разомкнутых контактов, соединенных параллельно, или последовательно, или смешанным образом.

Отрицанием контакта *A* называется контакт, равный *1*, если $A=0$, и равный *0*, если $A=1$. Отрицание контакта *A* обозначается через $\overline{A}$.

Очевидно, что последовательное соединение двух контактов *A* и *B* моделируется конъюнкцией переменных *A* и *B*, а параллельное соединение - их дизъюнкцией, см. Рис. 1.1.

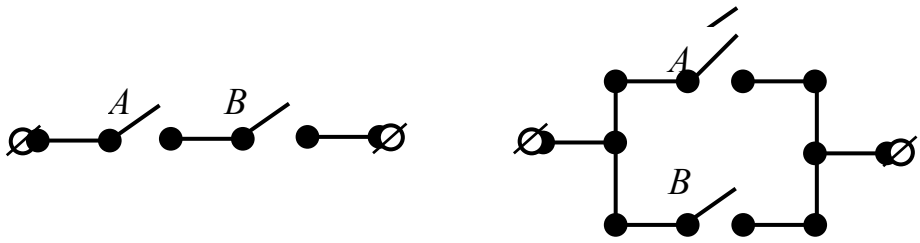



Рис. 1.1.

Известно, что любую булеву функцию можно представить, используя только $\neg$, $\&$, $\vee$, причем $\neg$ относится только к буквам. Следовательно, любую булеву функцию можно представить в виде контактной схемы.

Используя булевы функции, можно строить контактные схемы, удовлетворяющие заданным требованиям (производить синтез контактных схем), а также преобразовывать, упрощать схемы (производить анализ контактных схем). Покажем это на примере.

Требуется построить контактную схему для голосования комитета из трех человек. При голосовании "за" - нажатием кнопки свет должен загораться тогда и только тогда, когда "за" проголосовало большинство.

Решение. В нашем случае имеем три переменные. Обозначим их через *A, B*, и *C*. По условию свет должен загораться тогда и только тогда, когда большинство этих переменных принимает значение *1*, т.е. имеем таблицу:

| *A* | *B* | *C* | |
|---|---|---|---|
| *1* | *1* | *1* | *1* |
| *0* | *1* | *1* | *1* |
| *1* | *0* | *1* | *1* |
| *0* | *0* | *1* | *0* |
| *1* | *1* | *0* | *1* |
| *0* | *1* | *0* | *0* |
| *1* | *0* | *0* | *0* |
| *0* | *0* | *0* | *0* |

Используя эту таблицу, находим булеву функцию, выбирая строки, оканчивающиеся на *1*:

$$A\&B\&C \vee \neg A\&B\&C \vee A\& \neg B\&C \vee A\&B\& \neg C. \quad (1.11)$$

По этой функции строим схему, которая имеет четыре параллельные ветви в каждой из которых по три контакта, см. Рис. 1.2.

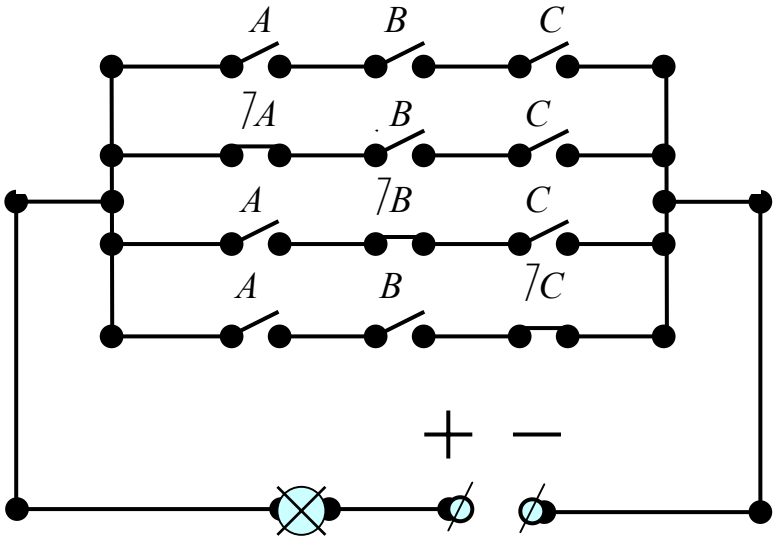



Рис. 1.2.

Эта схема выполняет поставленную задачу. Полученная схема содержит 12 контактов, и естественно попытаться проанализировать данную схему: нельзя ли, например, уменьшить количество контактов в переключательной схеме.

Можно получить, что форма (1.11) равносильна

форме $A\&B \vee A\&C \vee B\&C$, которую преобразуем к виду: $A\&B \vee C\&(A \vee B)$. Тогда схема будет содержащей всего пять контактов, см. Рис. 1.3.

Очевидно, что анализ позволил сильно упростить контактную схему.

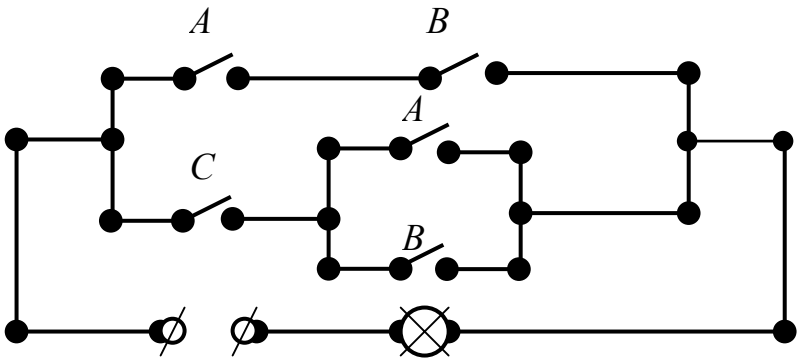
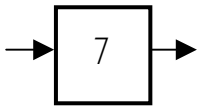



Рис. 1.3.

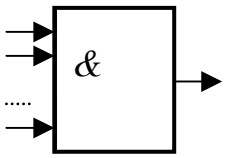
## § 12. Приложение алгебры высказываний к анализу и синтезу схем из функциональных элементов

Огромные скорости работы современных ЭВМ достигнуты за счет применения бесконтактных схем, работающих значительно быстрее, чем контактные схемы. В ЭВМ применяются электронные приборы, реализующие основные логические операции (отрицание, конъюнкцию, дизъюнкцию и др.).

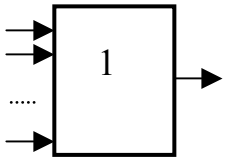
Не касаясь структуры и физических основ этих устройств, называемых *функциональными элементами*, обозначим их условно следующим образом:




устройство, реализующее отрицание;



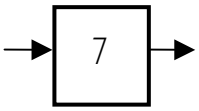

устройство, реализующее конъюнкцию;



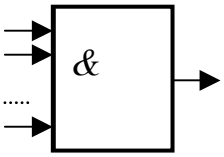

устройство, реализующее дизъюнкцию.

Ограничимся только этими устройствами, хотя на практике существуют функциональные элементы, реализующие и другие операции, например, отрицание конъюнкции и т. п. Но можно обойтись только перечисленными тремя устройствами, так как любую булеву функцию можно выразить, используя только $\neg, \&, \vee$.

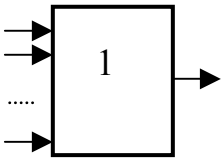
Об этих устройствах (функциональных элементах) мы знаем лишь следующее:




устройство, реализующее отрицание имеет один вход и один выход. Сигнал появляется на выходе, когда на входе нет сигнала, и не появляется сигнал, когда на вход подан сигнал.




устройство, реализующее конъюнкцию, имеет два и более входов и один выход. Сигнал появляется на выходе тогда и только тогда, когда на все входы поданы сигналы.




устройство, реализующее дизъюнкцию, имеет два и более входов и один выход. Сигнал появляется на выходе тогда и только тогда, когда подан сигнал хотя бы на один вход.

Этих свойств элементов достаточно для решения задач синтеза и анализа схем из этих элементов.

Рассмотрим пример построения одноразрядного сумматора двоичных чисел. Заданы двоичные числа $a_1a_2...a_k...a_n$ и $b_1b_2...b_k...b_n$. Требуется построить сумматор для $k$ –го разряда. Задача состоит в конструировании схемы (Рис. 1.4) с тремя входами *A, B, C* и двумя выходами *S* и *P*, чтобы при подаче на входы *A* и *B* сигналов, изображающих двоичные цифры - слагаемые данного разряда, а на вход *C* - сигнала, изображающего значение переноса из соседнего младшего разряда, получить на выходе *S* значение суммы в данном разряде, а на выходе *P* - значение переноса в соседний старший разряд.

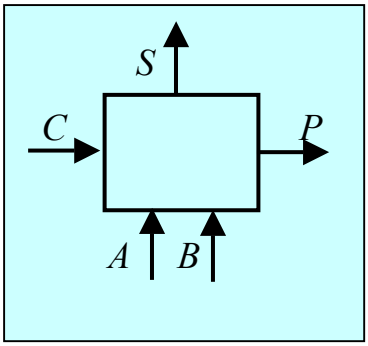



Рис. 1.4.

Напомним, что сложение чисел в двоичной системе производится следующим образом: *0+0=0, 0+1=1+0=1, 1+1=10, 1+1+1=11* и т.д. Воспользовавшись этой таблицей сложения чисел в двоичной системе, получим таблицу:

| *A* | *B* | *C* | *S* | *P* |
|---|---|---|---|---|
| *1*<br>*0* | *1*<br>*1* | *1*<br>*1* | *1*<br>*0* | *1*<br>*1* |
| *1*<br>*0* | *0*<br>*0* | *1*<br>*1* | *0*<br>*1* | *1*<br>*0* |
| *1*<br>*0* | *1*<br>*1* | *0*<br>*0* | *0*<br>*1* | *1*<br>*0* |

| | | | | |
|---|---|---|---|---|
| *1*<br>*0* | *0*<br>*0* | *0*<br>*0* | *1*<br>*0* | *0*<br>*0* |

Считая, что *0* и *1* есть значения булевой функции, и выбирая строки, оканчивающиеся на *1*, получим:

$$S=A\&B\&C\vee\neg A\&\neg B\&C\vee\neg A\&B\&\neg C\vee A\&\neg B\&\neg C,$$

$$P=A\&B\&C\vee\neg A\&B\&C\vee A\&\neg B\&C\vee A\&B\&\neg C \qquad (1.12)$$

Имея выражение (1.12), уже можно построить схему из функциональных элементов, выполняющих поставленную задачу. Но поскольку схема построена из функциональных элементов, выполняющих некоторые операции, то возникает задача такого упрощения, чтобы она содержала как можно меньше знаков операций. Можно показать, что

$$P=A\&B\vee A\&C\vee B\&C, \qquad S=A\&B\&C\vee(A\vee B\vee C)\&\neg P.$$

Тогда получим схему (Рис. 1.5), которая выполняет поставленную задачу, причем схема содержит значительно меньше функциональных элементов по сравнению со схемой, которая получилась при ее построении по (1.12) без проведения преобразований.

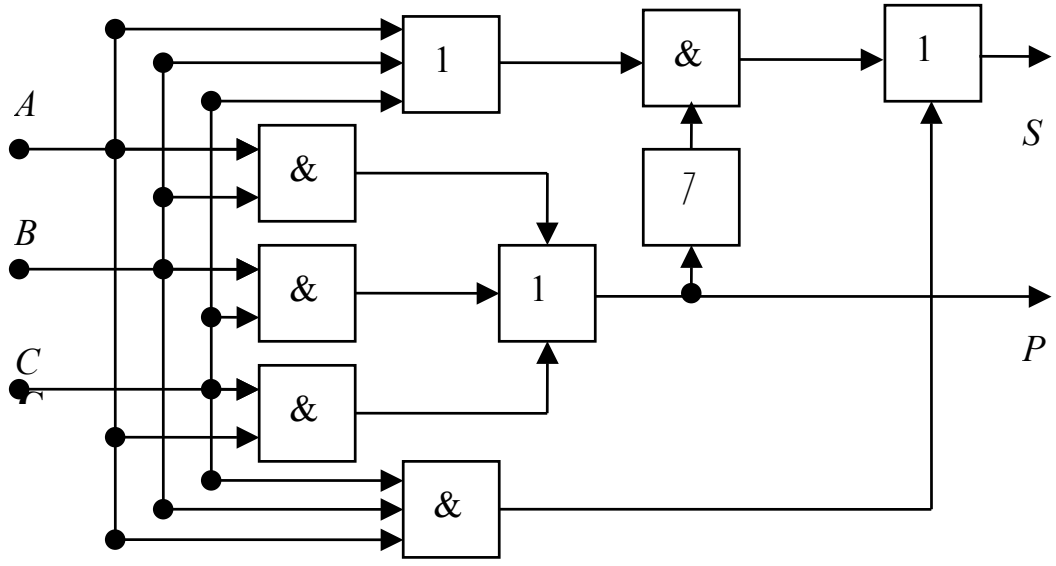



Рис. 1.5.

*Лишь только, подведя итог,*
*Ты свой дневной закончишь труд, …*
*Д. Чосер*

## § 13. Вопросы и темы для самопроверки

1. Высказывание, логические операции $\neg, \&, \vee, \Rightarrow, \equiv$; их определения и таблицы истинности.
2. Пропозициональные формы или формулы логики высказываний, определение, примеры. Упрощение записей пропозициональных форм.
3. Методы составления таблиц истинности.
4. Тавтологии (общезначимые формулы), противоречия. Две теоремы о тавтологиях.
5. Равносильность пропозициональных форм (формул логики высказываний), свойства отношения равносильности.
6. Важнейшие пары равносильных пропозициональных форм (запишите).
7. Зависимости между пропозициональными связками. Две теоремы о выражении пропозициональных форм с помощью форм, содержащих связки 3-х видов, 2-х видов.
8. Закон двойственности.
9. Выполнимые пропозициональные формы. Как можно выяснить выполнимость пропозициональной формы?
10. Элементарные суммы и произведения, их свойства.
11. Нормальные формы (д.н.ф. и к.н.ф.). Алгоритмы нахождения к.н.ф.; единственна ли к.н.ф. для заданной формы?
12. Выяснение общезначимости пропозициональной формы по к.н.ф.
13. Совершенная конъюнктивная нормальная форма (с.к.н.ф.). Алгоритмы нахождения с.к.н.ф.
14. Для каждой ли пропозициональной формы существует равносильная ей с.к.н.ф.? Единственна ли с.к.н.ф. для заданной формы?
15. Булева (переключательная) функция, определение, сколько существует различных булевых функций от $n$ переменных?
16. Можно ли любую булеву функцию представить в виде переключательной схемы?
17. Можно ли любую переключательную схему представить в виде булевой функции, содержащей только связки $\neg, \&, \vee$?
18. Можно ли любую булеву функцию представить в виде схемы из функциональных элементов?

*Если вы хотите научиться плавать, то смело входите в воду, а если хотите научиться решать задачи, то решайте их.*

*Д. Пойа (Математическое открытие)*

## § 14. Упражнения

*Логические операции. Символизация языка. Таблицы истинности*

1. Какие из следующих предложений являются высказываниями?

а) $2\times2=4$;

б) $2$ – простое число;

в) город Париж находится в Азии;

г) $3>5$;

д) $3+5$;

е) $3x=2$;

ж) числа вида $2^p-1$, где $p$ – простое число, назовем числами Мерсенна;

з) остаток от деления на $7$ числа $3$, возведенного в степень $123\ 456\ 789$ равен $6$;

и) является ли число $12345678998765432111111111111$ простым?

2. Следующие предложения являются составными высказываниями. Найдите их простые компоненты, т. е. такие высказывания, которые уже не построены из каких-либо других высказываний:

а) $7$ – простое число и не делится на $5$;

б) число $2^{11213}-1$ простое и его запись содержит более $3376$ цифр;

в) города Самара и Казань расположены на берегу Волги;

г) слово «алгоритм», которое иногда пишут «алгорифм», происходит от имени арабского математика аль-Хорезми (полное имя которого: Аль-Хорезми Абу Абдала Мохаммед бен Муса аль-Маджуси), который в IX столетии внес значительный вклад в распространение существовавших тогда методов вычислений;

д) число вида $aba\ bab$ делится на $7$, так как такое число является произведением чисел $ab$ и $10101$, а сомножитель $10101$ кратен $7$.

3. Пусть $A$ и $B$ обозначают соответственно «Андрей студент» и «Борис студент». Запишите приведенные ниже высказывания в символической форме, т. е. используя только обозначения для высказываний $(A,B)$, символы $\neg,\&,\vee,\Rightarrow,\equiv$ и скобки:

а) Андрей студент и Борис не студент;

б) Борис студент, а Андрей не студент;

в) Андрей и Борис оба не студенты;

г) Андрей или Борис студент;

д) либо Андрей студент, либо Борис студент;

е) ни Андрей, ни Борис не студенты;

ж) Андрей не студент и Борис не студент;

з) неверно, что Андрей и Борис оба студенты;

к) Андрей студент тогда и только тогда, когда Борис студент.

4. Пусть $C$ обозначает «Снег белый», а $D$ обозначает «Дважды два четыре». Сформулируйте словесно каждое из следующих высказываний:

а) $C\&D$; б) $C\&(\neg D)$;

в) $(\neg C)\&(\neg D)$; г) $C\vee(\neg D)$;

д) $\neg(C\&D)$; е) $\neg(C\vee D)$;

ж) $\neg((\neg C)\vee(\neg D))$; з) $(C\&(\neg D))\vee((\neg C)\&D)$.

5. Составьте таблицы истинности для высказываний:

а) $(A\&(\neg B))\vee B$; б) $((\neg A)\vee B)\&A$;

в) $(A\&(\neg B))\vee C$; г) $A\vee(B\&C)$.

6. Запишите следующие высказывания в символической форме употребляя заглавные латинские буквы для обозначения атомарных высказываний, т. е. таких высказываний, которые уже не построены из каких-либо других высказываний:

а) для того, чтобы число *a* было нечетным, достаточно, чтобы *a* было простым;

б) если *a* простое число, то $a^2$ – нечетное;

в) необходимым условием делимости числа на *4* является делимость его на *2*;

г) если *a* положительно, то $a^2$положительно;

д) Игорь или школьник, или студент, но он не студент, значит, он школьник;

е) $(n-1)!+1$ не делится на *n*, если *n* – не простое число;

ж) необходимым и достаточным условием делимости числа *a* на *6* является делимость его на *2* и *3*;

з) для того, чтобы число *a* делилось на *6*, достаточно чтобы *a* делилось на *36*;

и) Фиорелло ходит в кино только в том случае, когда там показывают комедию.

7. Пусть *A*, *B* обозначают некоторые высказывания. Запишите следующие высказывания в символической форме:

а) *A* достаточно для *B*;

б) *B* необходимо для *A*;

в) *B* тогда, когда *A*;

г) *B* только тогда, когда *A*;

д) без *B* нет и *A*;

е) *A* лишь тогда, когда *B*;

ж) *A* тогда и только тогда, когда *B*;

з) *A* тогда, когда *B*;

и) *A* необходимо и достаточно для *B*;

к) *A* если *B*;

л) *A* необходимое следствие из *B*;

м) *A* при условии, что *B*;

н) *A* влечет *B*;

о) в случае *A* имеет место *B*;

п) не только *A*, но и *B*;

р) как *A*, так и *B*;

с) *A* вместе с *B*;

т) если *A*, то *B*, и обратно.

8. Составьте списки выражений, которые могут быть заменены символами: а) $\neg A$; б) $A\&B$; в) $A\vee B$; г) $A\Rightarrow B$; д) $A\equiv B$.

*Пропозициональные связки и формы (формулы логики высказываний)*

9. Являются ли следующие выражения пропозициональными формами?

а) $\&A$; б) $(A\&(\vee B))$;

в) $((\vee C)\vee B)$; г) $(((\neg C)\Rightarrow B)\equiv B)$;

д) $(A\Rightarrow(\neg A))$; е) $(\neg(\neg(\neg(\neg A))))$.

10. а). Пусть значение пропозициональной формы $(A\equiv B)$ есть $И$. Что можно сказать о значениях пропозициональных форм $(A\equiv(\neg B))$ и $((\neg A)\equiv B)$?

б). Пусть значение пропозициональной формы $(A\equiv B)$ есть $Л$. Что можно сказать о значениях $(A\equiv(\neg B))$ и $((\neg A)\equiv B)$?

11. Найти значения $A, B, C$, если:

а) $(\neg(A\&B))=Л$; б) $(\neg(A\Rightarrow(\neg B)))=И$;

в) $((\neg(A\vee(A\equiv B)))\Rightarrow C)=Л$; г) $(A\vee(A\&B))=Л$;

д) $((A\&B)\equiv(B\vee C))=И$; е) $A\equiv B\&C\vee A=Л$;

ж) $\begin{cases}((\neg(A\&B))\equiv C)=И;\\(C\vee(\neg A))=Л;\end{cases}$ з) $\begin{cases}(((A\&B)\vee C)\Rightarrow A)=И;\\(A\vee(\neg C))=Л;\end{cases}$

и) $\begin{cases}(A\equiv C)=Л;\\(A\vee C)=Л;\end{cases}$ к) $\begin{cases}(((A\&B)\vee C)\equiv A)=И;\\(A\vee(\neg B))=Л;\end{cases}$

л) $\begin{cases}(A\Rightarrow(\neg B))=Л;\\((A\&B)\equiv C)=И.\end{cases}$ м) $\begin{cases}(A\Rightarrow C)=Л;\\(A\vee B)=И\end{cases}$

12. Составьте таблицы истинности для следующих пропозициональных форм:

а) $(A\Rightarrow(B\Rightarrow A))$;

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

в) $(((\neg B)\Rightarrow(\neg A))\Rightarrow(((\neg B)\Rightarrow A)\Rightarrow B))$;

г) $((A\Rightarrow B)\Rightarrow C)$;

д) $(A\Rightarrow(B\Rightarrow C))$;

е) $((A\Rightarrow B)\equiv((\neg B)\Rightarrow(\neg A)))$;

ж) $((A\equiv B)\vee((\neg C)\&B))$;

з) $((A\Rightarrow B)\equiv((\neg A)\Rightarrow(\neg B)))$.

Укажите, какие из этих пропозициональных форм являются тавтологиями и какие противоречиями.

14. Доказать, что если $\boldsymbol{A}$ – тавтология, то тавтологиями являются $(\boldsymbol{A}\vee\boldsymbol{B})$ и $(\boldsymbol{B}\Rightarrow\boldsymbol{A})$, где $\boldsymbol{B}$ – произвольная пропозициональная форма.

15. Для данных пропозициональных форм составить таблицы истинности. Определить, для которых из них истинностные значения всей формы можно записать без промежуточных выкладок (используйте результаты задачи 14):

а) $((A\Rightarrow A)\vee B)$; б) $(C\Rightarrow(A\equiv A))$;

в) $(A\Rightarrow(\neg A))$; г) $((A\Rightarrow A)\Rightarrow A)$;

д) $(A\Rightarrow(A\equiv A))$; е) $(((A\&B)\vee(\neg C))\Rightarrow(A\Rightarrow A))$;

ж) $(((C\&D)\vee(A\&B))\Rightarrow(B\vee(\neg B)))$; з) $((A\vee(B\vee(\neg A)))\&(C\vee(A\vee(\neg C))))$;

и) $((B\equiv B)\&(C\Rightarrow C)\&(A\vee(\neg A)))$; к) $(((A\&B)\&(A\&A))\vee(\neg(A\&B)))$.

16. Составить таблицу истинности для истинностной функции, зависящей от трех переменных, если известно, что функция истинна тогда и только тогда, когда

а) все переменные принимают одинаковые значения;

б) истинны значения большинства переменных этой функции;

в) истинно значение одного и только одного из ее переменных;

г) каждая переменная принимает значение, отличное от значения соседней переменной.

*Упрощение в записях пропозициональных форм.*

17. 1). Записать в сокращенном виде, т. е. по возможности опустив скобки:

а) $(((\neg A)\Rightarrow(B\vee(\neg C)))\equiv((B\&A)\vee(\neg B)))$;

б) $(((A\vee(\neg B))\vee C)\Rightarrow((\neg A)\&B))$;

в) $((A\Rightarrow B)\Rightarrow(C\Rightarrow(\neg C)))$;

г) $((((\neg A)\equiv B)\&(B\vee(\neg C)))\equiv A)$;

д) $((((A\Rightarrow(\neg A))\vee(\neg B))\vee C)\&(A\Rightarrow B))$;

е) $(A\Rightarrow(B\Rightarrow(C\Rightarrow D)))$.

2). В приведенных ниже сокращенных записях пропозициональных форм восстановить опущенные скобки:

а) $A\equiv B\vee\neg C\&A\Rightarrow\neg A$;

б) $\neg B\Rightarrow B\Rightarrow C\equiv C\&D$;

в) $A\equiv B\equiv\neg A\vee\neg B\&A$;

г) $A\Rightarrow B\Rightarrow\neg A\&B\vee C$;

д) $A\equiv B\Rightarrow C\vee\neg A\&\neg B\vee A$;

е) $A\&\neg B\&C\Rightarrow A\Rightarrow\neg A\vee A\equiv B$.

*Равносильные пропозициональные формы*

18. Найти простейшие (содержащие минимально возможное число вхождений пропозициональных букв) равносильные пропозициональные формы для заданных пропозициональных форм:

а) $A\&(A\vee B)$;

б) $A\vee\neg A\&B$;

в) $\neg A\vee A\&B$;

г) $A\&(\neg A\vee B)$;

д) $\neg A\&(A\vee B)$;

е) $A\vee A\vee A\&A\&B\&C$;

ж) $A\&A\&B\vee B\Rightarrow A\&B\vee B\vee A\&A\&C$;

з) $(A\vee(B\&C))\vee(\neg A\&(\neg B\vee C))$;

и) $(A\&B)\vee C\&D\vee(\neg A\vee\neg B)$;

к) $A\&B\vee(C\Rightarrow C)$;

л) $A\vee\neg C\&B\Rightarrow C\vee\neg C$;

м) $A\Rightarrow A\equiv A\Rightarrow A\Rightarrow A$.

19. Законы Де Моргана для $n$ переменных можно записать в виде

$\neg(\&_{i=1}^{n} A_i)$ равносильно $\vee_{i=1}^{n}(\neg A_i)$;

$\neg(\vee_{i=1}^{n} A_i)$ равносильно $\&_{i=1}^{n}(\neg A_i)$.

Для $n$=2 эти равносильности доказаны (например, с помощью таблицы истинности). Доказать их для любого $n$ индукцией по числу переменных.

20. Доказать, что $A \vee B \& C \& D$ равносильно $(A \vee B) \& (A \vee C) \& (A \vee D)$.

21. Доказать, что $A \vee (\mathop{\&}_{i=1}^{n} B_i)$ равносильно $\mathop{\&}_{i=1}^{n} (A \vee B_i)$.

22. Доказать, что $A \& (\mathop{\vee}_{i=1}^{n} B_i)$ равносильно $\mathop{\vee}_{i=1}^{n} (A \& B_i)$.

*Связь между логическими операциями*

23. Запишите $A \Rightarrow (B \Rightarrow C)$ без связки $\Rightarrow$. Запишите ответ так, чтобы связка $\neg$ не стояла перед скобками.

24. Запишите $(B \Rightarrow C) \Rightarrow \neg A$ без связки $\Rightarrow$. Запишите результат в форме, не содержащей $\neg$ перед скобками.

25. Запишите каждую из следующих пропозициональных форм без пропозициональной связки $\Rightarrow$, а окончательный результат без пропозициональной связки $\neg$, стоящей перед скобками:

а) $(A \vee \neg B) \Rightarrow C$; б) $(A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A)$;

в) $(A \Rightarrow B) \Rightarrow (A \& B \Rightarrow A)$.

26. Выразите $A \Rightarrow (B \equiv C)$ без $\Rightarrow$, $\equiv$, внутренних скобок и отрицаний, стоящих перед скобками.

27. Для пропозициональной формы $\neg A \vee B \vee \neg C$ найдите равносильную ей форму, содержащую только связку $\Rightarrow$.

28. Для пропозициональной формы $A \vee B \vee C$ найдите шесть равносильных ей форм, содержащих только связки $\neg$, $\Rightarrow$.

29. Для пропозициональной формы $(A \equiv B) \Rightarrow C$ найти равносильную, содержащую: а) только связки $\neg$, $\vee$; б) только связки $\neg$, $\&$; в) только связки $\neg$, $\Rightarrow$.

30. Для пропозициональной формы $A \Rightarrow B \Rightarrow C$ найти равносильную, содержащую: а) только связки $\neg$, $\vee$; б) только связки $\neg$, $\&$.

31. Найти простейшие равносильные пропозициональные формы для заданных пропозициональных форм:

а) $A \vee A \& B \& C \& D$; б) $A \& \neg B \vee \neg A \vee A$;

в) $(\neg A \vee B \equiv C) \& B \vee B$; г) $(A \Rightarrow B \equiv C) \Rightarrow B \vee \neg B$;

д) $A \vee A \& B \& B \& (D \vee A \vee A)$; е) $A \vee A \& B \& B \& B \Rightarrow B \& C \& A \vee A$;

ж) $A \equiv A \equiv A$; з) $A \equiv A \equiv A \equiv A$;

и) $A \Rightarrow A \Rightarrow A \Rightarrow A$; к) $A \Rightarrow (A \Rightarrow A) \Rightarrow A$;

л) $A \Rightarrow (A \Rightarrow (A \Rightarrow A))$.

*32.* Для пропозициональной формы $A \Rightarrow B \& C$ найти равносильную, содержащую а) только связки $\neg$, $\&$; б) только связки $\neg$, $\vee$; в) только связки $\neg$, $\Rightarrow$.

33. Упростить, насколько это возможно:

а) $(A \vee B \vee C) \& (A \vee B \vee \neg C)$; б) $(C \vee D \vee \neg E) \& C \& (A \vee \neg D \vee \neg E)$;

в) $D \vee (E \& D \& C)$; г) $(E \vee C \vee B) \& (D \vee \neg D)$;

д) $B\&(A\vee B)\&A$; е) $A\vee(A\vee\neg A)\vee(C\vee\neg C)\vee D$;

ж) $C\&(\neg A\vee C\vee D)\&E\&(E\vee\neg B)$; з) $C\&\neg C\vee D\vee A\vee B$;

и) $C\&\neg(C\&B)$; к) $C\&D\&\neg D\&(E\vee\neg A\vee B)$;

л) $A\&B\&C\vee B\&C\&A\vee A\Rightarrow B\vee\neg B$; м) $B\vee\neg B\vee C\&B\equiv A\&\neg A$.

34. Доказать, что связки $\vee$ недостаточно для выражения любой истинностной функции.

35. Доказать, что каждая из пар связок $(\Rightarrow, \vee)$, $(\&, \equiv)$ не является достаточной для выражения любой истинностной функции.

36. Показать, что для выражения любой истинностной функции недостаточно

а) связки $\&$; б) связки $\Rightarrow$;

в) связки $\equiv$; г) связок $\vee$, $\equiv$;

д) связок $\&$, $\Rightarrow$.

37. Следующие пропозициональные формы привести к д.н.ф. и к.н.ф.

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

б) $((A\Rightarrow B)\&(C\Rightarrow D))\&((A\&C)\Rightarrow(B\&D))$;

в) $(A\Rightarrow B)\Rightarrow((A\&C)\Rightarrow(A\&C))$;

г) $(A\Rightarrow(B\&C))\&((A\Rightarrow B)\&(A\Rightarrow C))$;

д) $((A\Rightarrow B)\vee(A\Rightarrow C))\Rightarrow(A\Rightarrow(A\vee C))$.

38. Для заданных пропозициональных форм:

1) найдите д.н.ф., к.н.ф.; 2) выясните, является ли выполнимой или тавтологией; 3)найдите с.д.н.ф и с.к.н.ф.:

а) $A\equiv B\vee C$; б) $(A\Rightarrow\neg B)\&C$;

в) $(A\equiv A\equiv A\equiv B)\&C$; г) $(A\Rightarrow A\Rightarrow A\Rightarrow B)\vee C$;

д) $A\equiv A\equiv B\vee C$; е) $(A\Rightarrow B)\Rightarrow C\Rightarrow\neg A$.

39. Сколько существует различных способов возможного заполнения последнего столбца таблицы истинности для истинностной функции от $n$ аргументов? Сколько существует различных истинностных (булевых) функций от $n$ аргументов?

*Приложение алгебры высказываний. Контактные схемы*

40. Записать пропозициональную форму, соответствующую схеме Рис. 1.6.

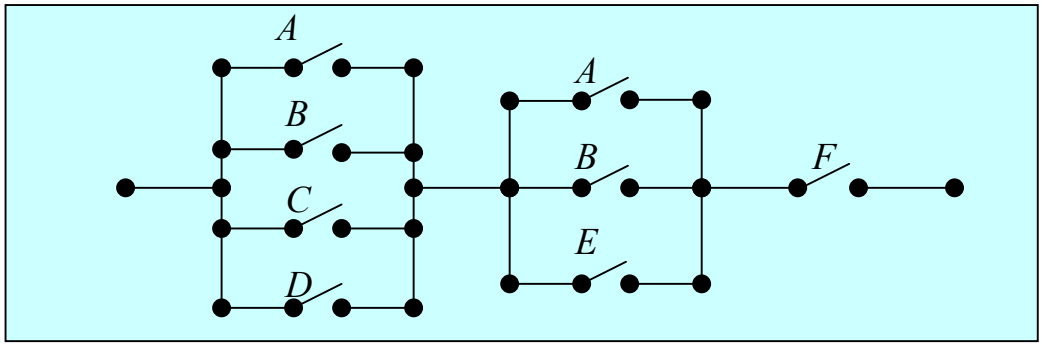



Рис. 1.6.

41. Построить контактные схемы для пропозициональных форм:

а) $(A \Rightarrow B) \& C \Rightarrow \neg A$; б) $A \& B \vee \neg A \Rightarrow C$;

в) $(A \vee \neg B) \Rightarrow C$; г) $(A \Rightarrow B) \equiv C$;

д) $(A \Rightarrow A \Rightarrow A) \& B$; е) $A \Rightarrow A \Rightarrow A \Rightarrow A$.

42. Комитет состоит из пяти членов. Решения выносятся большинством голосов; однако, если председатель против, решение не может быть принято. Построить схему, чтобы при голосовании «за» - нажатием кнопки – свет загорался только в том случае, если решение принято.

43. Требуется, чтобы в большом зале можно было включать и выключать свет при помощи любого из четырех переключателей, расположенных на четырех стенках. Построить схему. (Это осуществимо путем конструирования схемы, в которой свет включается, когда замкнуто четное число выключателей, и выключается, когда замкнуто нечетное число переключателей. Почему?)

44. В большом, совершенно темном зале стоит круглый стол, вокруг которого стоит *8* стульев. Около каждого стула имеется переключатель. В комнату сходят *4* мальчика и *4* девочки и садятся за стол. Каждая девочка замыкает свой переключатель, а каждый мальчик размыкает свой. Начертите схему, которая замыкается тогда и только тогда, когда мальчики и девочки сядут через одного.

*Схемы из функциональных элементов*

45. С помощью функциональных элементов составить схему с тремя входами и одним выходом так, чтобы на выходе появился сигнал тогда и только тогда, когда по крайней мере на двух входах поступают сигналы.

46. С помощью функциональных элементов построить схему с тремя входами *A, B, C* и двумя выходами *S,P,* такую, что на выходе *S* сигнал появляется тогда и только тогда, когда по крайней мере на одном из входов есть сигнал, а на выходе *P* сигнал появляется тогда и только тогда, когда на всех входах имеется сигнал.

47. С помощью функциональных элементов составить схему с двумя входами и двумя выходами так, чтобы на одном выходе появлялся сигнал тогда и только тогда, когда хотя бы на одном из входов поступает сигнал, а на другом выходе – когда только на одном из входов поступает сигнал.

*Логика непобедима, потому что одолеть ее можно только с помощью логики.*

*О. Хевисайд*

# Глава 2 ЛОГИКА ПРЕДИКАТОВ

*В то время как заурядный наблюдатель видит*
*Лишь ряд разрозненных, отдельных сцен*
*И бродит ощупью средь них всю жизнь,*
*Способны вы сводить их воедино.*

*Г. Ибсен (Пер Гюнт)*

## § 1. Понятие предиката

Логика предикатов представляет собой развитие логики высказываний. Она содержит в себе всю логику высказываний, т.е. элементарные высказывания, рассматриваемые как величины, которые принимают значения *И* либо *Л*. Но помимо этого, язык логики предикатов вводит в рассмотрение утверждения, отнесенные к предметам, т.е. производится более детальный анализ предложений. Рассмотрение логики предикатов вызвано тем, что логика высказываний не позволяет моделировать рассуждения всех видов, в частности рассуждении с использованием понятии «каждый», «некоторый». Отметим, что логика предикатов тоже не охватывает всевозможных случаев рассуждений, например, когда нужно исследовать рассуждения, истинность которых зависит от времени или вводятся понятия «должно быть» и «может быть» и т.п.; подробнее см. главу 5.

Пусть $\mathcal{M}$- некоторое множество предметов, $a_1,a_2,\ldots$- какие-то определенные предметы (элементы) из этого множества. Тогда через $A(a_1)$ будем обозначать некоторое высказывание о предмете $a_1$, а через $A(a_2)$ - то же высказывание о предмете $a_2$. Например, если $\mathcal{M}$ есть множество всех натуральных чисел и $a_1$=3, $a_2$=8, то $A(a_1)$ может обозначать высказывание "3-простое число", тогда $A(a_2)$ будет обозначать "8-простое число".

Как и в логике высказываний, будем рассматривать эти высказывания только с той точки зрения, что они представляют либо истину, либо ложь, обозначаемые соответственно *И* и *Л*. При этом значения высказывания $A(a_1)$ и $A(a_2)$ могут быть разными или нет в зависимости от выбранных предметов $a_1$ и $a_2$. Следовательно, в отличие от алгебры высказываний будем считать, что значения *И* и *Л* ставятся в соответствие определенным предметам или группам предметов.

Если же не будем фиксировать предмет, например, рассмотрим $A(x)$, где $x$ - любой предмет из $\mathcal{M}$, то получим некоторое предложение, которое становится высказыванием, когда $x$ замещено определенным предметом из $\mathcal{M}$. Например, если $\mathcal{M}$ является множеством всех натуральных чисел, то $A(x)$ может обозначать "$x$ - простое число". Это предложение становится высказыванием, если $x$ заменить числом, например, "3 -простое число", "4-простое число". При этом получаем высказывания, которые истинны, либо ложны. Следовательно, $A(x)$ порождает функцию, область определения которой есть множество $\mathcal{M}$, а область значений – множество $\{И,Л\}$. Отметим (еще раз), что $A(x)$ становится высказыванием при замене $x$ фиксированным (определенным) предметом из $\mathcal{M}$.

Предложения, в которых имеются две и более переменных, будем обозначать, например, $A(x,y)$, $B(x,y,z)$ и т. п. При этом $x, y, z$ пробегают все множество $\mathcal{M}$, а $A(x,y)$, $B(x,y,z)$ при фиксированных $x, y, z$ становятся высказываниями, следовательно, принимают значение $И$ либо $Л$. Например, пусть $\mathcal{M}$ есть множество всех действительных чисел Рассмотрим предложение: "$x$ делится нацело на $y$". Если вместо $x$ и $y$ подставить конкретные числа из $\mathcal{M}$, получится высказывание истинное либо ложное, так, при $x$=6, $y$=3 высказывание "6 делится нацело на 3" - истинное, а при $x$=5, $y$=7 высказывание "5 делится нацело на 7" - ложно. Рассмотренное предложение "$x$ делится нацело на $y$" можно обозначить, например, через $C(x,y)$. Такого типа предложения, порождающие функции одного или нескольких переменных, будем считать предикатами.

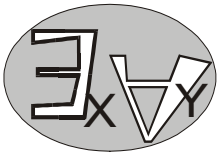
Точнее: *предикатом* называется повествовательное предложение об элементах некоторого заданного множества $\mathcal{M}$, которое (предложение) становится высказыванием, если все переменные в нем заменить фиксированными элементами из $\mathcal{M}$; высказывание тоже будем считать предикатом - *нульместным предикатом*. Часто вместо "предикат от $n$ переменных " говорят "*n-местный предикат*".

**Упражнение.** Пусть на множестве $\mathcal{M}$, состоящем из $m$ элементов, задан 3-х местный предикат $A(x,y,z)$. Сколько высказываний об элементах $\mathcal{M}$ можно получить, фиксируя переменные предиката $A(x,y,z)$?

Введем операции над предикатами. Пусть $A(x)$, $B(x)$ - заданные на $\mathcal{M}$ предикаты. Будем считать, что $\neg A(x)$ тоже определяет предикат на $\mathcal{M}$, причем при каждом фиксированном $x=b$ значение высказывания $\neg A(b)$ противоположно значению высказывания $A(b)$. Так же будем образовывать из предикатов $A(x)$, $B(x)$ новые предикаты с помощью операций $\&$, $\vee$, $\Rightarrow$, $\equiv$. Так, например, $A(x)\&B(x)$ обозначает предикат, который при фиксированном $x=b$ превращается в сложное высказывание $A(b)\&B(b)$, образованное из высказываний $A(b)$ и $B(b)$ соединением их связкой $\&$. Точно так же будем образовывать новые предикаты из произвольных многоместных предикатов. Например, $A(x,y)\Rightarrow C(x,y,z)$ обозначает предикат, который при фиксированных переменных: $x=a$, $y=b$, $z=c$ $(a,b,c\in \mathcal{M})$ превращается в

высказывание $A(a,b) \Rightarrow C(a,b,c)$, образованное из двух высказываний $A(a,b)$ и $C(a,b,c)$ соединением их связкой $\Rightarrow$.

## § 2. Кванторы




Введем специальные обозначения. Пусть $\mathcal{M}$ - множество, $P(x)$ - определенный на $\mathcal{M}$ одноместный предикат. Тогда выражение

$\forall x P(x)$

читается: "для всех $x$ $P(x)$" или "для всех $x$ выполняется $P(x)$", или "для любого $x$ $P(x)$", или "для каждого $x$ $P(x)$". Под выражением "$\forall x P(x)$" будем подразумевать высказывание истинное, когда $P(x)$ истинно для каждого $x$ из $\mathcal{M}$ и ложное - в противном случае. Символ $\forall x$ называется *квантором всеобщности*. Выражение

$\exists x P(x)$

читается "существует $x$ такое, что $P(x)$" или "хотя бы для одного $x$ $P(x)$", или "для некоторого (некоторых) $x$ $P(x)$". Под выражением "$\exists x P(x)$" будем подразумевать высказывание, которое истинно, если $P(x)$ принимает значение *И* хотя бы для одного значения переменной $x \in \mathcal{M}$, и ложно, если $P(x)$ для всех значений переменной $x$ принимает значение *Л*. Символ $\exists x$ называется *квантором существования*. Квантор $\exists x$ будем называть *двойственным* к квантору $\forall x$, и наоборот.

В литературе применяются и другие обозначения. Так, вместо $\forall x P(x)$ пишут $\Lambda x P(x)$ или $\Lambda_x P(x)$, а вместо $\exists x P(x)$ пишут $V x P(x)$ или $V_x P(x)$, или $E x P(x)$

Введенные обозначения позволяют записывать предложения в символической форме, которая оказывается более удобной для анализа и логических действий над этими предложениями. При символизации языка требуется определенная аккуратность и правильное понимание контекста. В естественном языке часто слово "все" опускается. Так, например, предложение "рыбы дышат жабрами" означает, что все рыбы дышат жабрами или что каждая рыба дышит жабрами. Поэтому при символизации необходимо ввести квантор общности. Таким образом, если положить для множества живых существ, что $R(x)$ означает "$x$ - рыба", а $G(x)$ - "$x$ дышит жабрами", то имеем

$\forall x(R(x) \Rightarrow G(x))$.

Но в то же время не в каждом случае встречающиеся в предложениях слова "все" понимаются как "каждый". Например, предложение "все песчинки образуют кучи пуска" не означает, что каждая песчинка образует кучи песка, следовательно, при символизации нельзя употреблять квантор $\forall x$, как это сделано в предыдущем примере.

В языке слово "все" имеет два значения: "любой, каждый" и "все вместе". Квантор $\forall x$ применяется для первого значения.

Из изложенного следует, что "$\forall xP(x)$" служит обозначением для следующих высказываний:

для всех $x$ выполняется (имеет место) $P(x)$;

для каждого $x$ выполняется (имеет место) $P(x)$;

для любого $x$ выполняется (имеет место) $P(x)$;

для произвольного $x$ выполняется (имеет место) $P(x)$;

каково бы ни было $x$ выполняется (имеет место) $P(x)$.

В языке слово "некоторый", так же как и "все", часто опускается. Например, предложение "люди побывали на Луне" означает, что некоторые люди побывали на Луне.

Символическая запись $\exists xP(x)$, как мы знаем, означает, что для некоторых $x$ имеет место $P(x)$, но не исключено, что и для всех $x$ имеет место $P(x)$. В естественном же языке слово "некоторый" иногда употребляют в смысле "не все". Когда говорят "некоторые студенты отличники", подразумевают, что некоторые, но не все студенты отличники. Следовательно, имеется в виду: "неверно, что все студенты отличники, но некоторые - отличники". Тогда, если $C(x)$ означает "$x$ - студент", $O(x)$ означает "$x$ - отличник", то получим:

$(\neg\forall x(C(x) \Rightarrow O(x))) \& \exists x(C(x)\&O(x))$.

Итак, слово "некоторый" имеет два значения: первое - "некоторый, но может быть и все", второе - "некоторый, но не все". Символ $\exists x$ обозначает первое. Следовательно, запись $\exists xP(x)$ служит обозначением для следующих высказываний:

для некоторых $x$ (имеет место) $P(x)$;

существует $x$, для которого $P(x)$;

найдется $x$, для которого $P(x)$;

хотя бы для одного $x$ (верно) $P(x)$;

имеется $x$, для которого $P(x)$.

Рассмотрим следующие часто встречающиеся предложения и справа от них приведем их символическую запись:

($A$) все $S$ суть $P$ - $\forall x(S(x) \Rightarrow P(x))$;

($E$) ни одно $S$ не есть $P$ - $\forall x(S(x) \Rightarrow \neg P(x))$;

($I$) некоторые $S$ суть $P$ - $\exists x(S(x)\&P(x))$;

($O$) некоторые $S$ не есть $P$ - $\exists x(S(x)\&\neg P(x))$.

Символизация приведенных предложений позволяет записывать в символическом виде довольно сложные выводы, использующие всевозможные комбинации предложений ($A$)-($O$).

До сих пор мы рассматривали приписывание кванторов к одноместным предикатам.

Далее рассмотрим приписывание кванторов к многоместным предикатам. Пусть $P(x_1,x_2,...,x_n)$ -$n$-местный $(n\geq 2)$ предикат, заданный на множестве $\mathcal{M}$. Выражение

$\forall x_i P(x_1,x_2,...,x_n),\ 1 \le i \le n,$ (2.1)

является *(n-1)*-местным предикатом, зависящим от (свободных) переменных $x_1,x_2,...,x_{i-1},x_{i+1},...,x_n$, причем высказывание

$\forall x_i P(a_1,a_2,...,a_{i-1},x_i,a_{i+1},...,a_n)$

истинно тогда и только тогда, когда для любого значения $a_i \in \mathcal{M}$ истинно высказывание

$P(a_1,a_2,...,a_{i-1},a_i,a_{i+1},...,a_n).$ (2.2)

Выражение

$\exists x_i P(x_1,x_2,...,x_n),\ 1 \le i \le n,$ (2.3)

является *(n-1)*-местным предикатом, зависящим от (свободных) переменных $x_1,x_2,...,x_{i-1},x_{i+1},...,x_n$, причем высказывание

$\exists x_i A(a_1,a_2,...,a_{i-1},x_i,a_{i+1},...,a_n)$

истинно тогда и только тогда, когда существует такое значение $a_i$, ($a_i \in \mathcal{M}$) переменной $x_i$, для которого высказывание (2.2) истинно.

Также положим, что если $P$ - нульместный предикат (высказывание), то записи $\forall xP$ и $\exists xP$ означают то же, что и $P$.

Приписывание (навешивание) квантора по переменной $x$ связывает переменную $x$. Приписывая к *(n-1)*-местным предикатам (2.1) и (2.3) любой квантор по любой из свободных переменных, получим *(n-2)*-местные предикаты (если $n=2$, то просто высказывание). Ясно, что к полученным предикатам можно снова приписать произвольные кванторы и т.д. Очевидно, что, приписав кванторы по всем переменным, получим высказывание. Например, пусть на множестве действительных чисел задан трехместный предикат $x^2 + y^2 \ge z^2$, который можно превратить в двуместный предикат, записав квантор: $\forall z(x^2 + y^2 \ge z^2)$ или превратить в одноместный предикат $\forall y \forall z(x^2 + y^2 \ge z^2)$, или же превратить в высказывание:

$\forall x \forall y \forall z(x^2 + y^2 \ge z^2).$ (2.4)

можно получить и другие высказывания, например:

$\exists x \forall y \forall z(x^2 + y^2 \ge z^2),$ (2.5)

$\forall x \forall y \exists z(x^2 + y^2 \ge z^2)$ (2.6)

и т.д. Ясно, что высказывание (2.6) истинно, а (2.4) и (2.5) - ложные.

**Упражнение**. Пусть на множестве $\mathcal{M}$ задан трехместный предикат $P(x,y,z)$. Определить, какое число одноместных предикатов можно получить из $P(x,y,z)$, приписывая к нему различные кванторы.

Пусть множество $\mathcal{M}$ состоит из конечного числа элементов. Для определенности положим $\mathcal{M} = \{a_1,a_2,...,a_n\}$ и пусть $P(x)$ заданный на $\mathcal{M}$ одноместный предикат. Тогда, очевидно, имеем:

$\forall xP(x)$ равносильно $P(a_1)\&P(a_2)\&...\&P(a_n)$, (2.7)

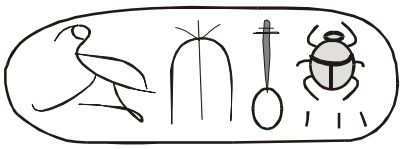
$\exists xP(x)$ равносильно $P(a_1) \vee P(a_2) \vee ... \vee P(a_n)$. (2.8)

Следовательно, квантор всеобщности является обобщением (аналогом) конъюнкции, а квантор существования - обобщением (аналогом) дизъюнкции на произвольное, не обязательно конечное, множество.

*Удачные обозначения обладают*
*утонченностью и будят мысль.*
*Б. Рассел*

## § 3. Формулы логики предикатов

[1]



В предыдущих параграфах записывались некоторые предложения содержательного языка в виде символов. В этом параграфе рассмотрим правила образования из определенных символов различных выражений (термов, формул) без каких-либо ссылок на их содержательный смысл. Только в дальнейшем (§4) будем придавать содержательный смысл этим наборам символов, т.е. рассматривать, какие предложения содержательного языка они могут обозначать.

Буквы начала латинского алфавита ($a,b,c,...$) и они же с числовыми индексами ($a_1,a_2,...,b_1,b_2,...,c_1,c_2,...$) называются *предметными постоянными*.

Буквы конца латинского алфавита ($x,y,z,...$) и они же с числовыми индексами ($x_1,x_2,...,y_1,y_2,...,z_1,z_2,...$) называются *предметными переменными*.

Буквы $A_i^n$ с числовыми индексами $i \geq 1,\ n \geq 0$, называются *предикатными буквами*, а $f_i^n$, $i \geq 1,\ n \geq 1$, - *функциональными буквами*. Верхний индекс предикатной или функциональной буквы указывает число аргументов, а нижний индекс служит для различения букв с одним и тем же числом аргументов.

Будем по возможности опускать числовые индексы у предикатных и функциональных букв, считая, что их легко можно восстановить. Например, вместо $A_1^2(x,y)$ будем писать $A_1(x,y)$, и если нет других двухаргументных (двуместных) предикатных букв $A$, то вместо $A_1(x,y)$ будем писать просто $A(x,y)$, кроме того иногда будем использовать буквы $P, Q, R, S$ для обозначения предикатных букв $A_i^n$.

Определение *терма*:

а) всякая предметная переменная или предметная постоянная есть терм;

б) если $f_i^n$ - функциональная буква и $t_1,t_2,...,t_n$ - термы, то $f_i^n(t_1,t_2,...,t_n)$ есть терм;

---

[1] *Здесь приведена некоторая копия древнеегипетских записей – иероглифов. Она перерисована из книги С. Коваль, От развлечения к знаниям. Варшава, 1972.*

в) выражение является термом только в том случае, если это следует из правил а) и б).

Примеры термов: $a, x_1, f_1^2(x,a), f_3^2(a, f_2^1(y))$.

Предикатные буквы, примененные к термам, порождают элементарные формулы, или точнее: если $A_i^n$ - предикатная буква, а $t_1,t_2,...,t_n$ - термы, то $A_i^n(t_1,t_2,...,t_n)$ - *элементарная формула*.

Будем считать, что нульместная предикатная буква тоже является элементарной формулой.

Примеры элементарных формул: $A_1^0$, $A_1^2(x_1,a_1)$, $A_1^1(f_1^1(x))$, $A_2^3(x,y,f_1^2(a,x))$.

*Формулы логики предикатов* определяются следующим образом:

а) всякая элементарная формула есть формула;

б) если $\boldsymbol{A}$ и $\boldsymbol{B}$ - формулы и $x$ - предметная переменная, то каждое из выражений $(\neg\boldsymbol{A})$, $(\boldsymbol{A}\Rightarrow\boldsymbol{B})$, $(\forall x\boldsymbol{A})$ и $(\exists x\boldsymbol{A})$ есть формула;

в) выражение является формулой только в том случае, если это следует из правил а) и б).

В выражениях $(\forall x\boldsymbol{A})$ и $(\exists x\boldsymbol{A})$ формула $\boldsymbol{A}$ называется областью действия кванторов $\forall x$ и $\exists x$ соответственно.

Примеры формул: $(A_1^0\Rightarrow A_1^2(x,a))$, $(\forall x\, A_1^2(x, f_1^1(f_1^1(x))))$, $(\forall x(\exists x\, A_1^1(a)))$.

Если $\boldsymbol{A}$ и $\boldsymbol{B}$ - формулы, то договоримся, что $\boldsymbol{A}\&\boldsymbol{B}$ является сокращенной записью формулы $\neg(\boldsymbol{A}\Rightarrow\neg\boldsymbol{B})$,

$\boldsymbol{A}\vee\boldsymbol{B}$ является сокращенной записью формулы $((\neg\boldsymbol{A})\Rightarrow\boldsymbol{B})$,

$\boldsymbol{A}\equiv\boldsymbol{B}$ является сокращенной записи формулы $\neg((\boldsymbol{A}\Rightarrow\boldsymbol{B})\Rightarrow\neg(\boldsymbol{B}\Rightarrow\boldsymbol{A}))$.

Будем придерживаться тех же правил опускания скобок, которые приняты в §3 главы 1, введя дополнительно, что кванторы $\forall x$, $\exists x$ располагаются по силе между связками $\neg,\&,\vee$ и связками $\Rightarrow,\equiv$. Например, вместо $((\forall x\boldsymbol{A})\Rightarrow\boldsymbol{B})$ пишем $\forall x\boldsymbol{A}\Rightarrow\boldsymbol{B}$.

Договоримся также опускать скобки, в которые заключается формула $\boldsymbol{A}$ в формулах вида $(Q_1(Q(\boldsymbol{A})))$, где $Q_1$ и $Q$ - любые кванторы. Например, вместо $(\forall x(\forall y(\exists z(A_1^3(x,y,z)))))$ пишем $\forall x\,\forall y\exists z\, A_1^3(x,y,z)$.

Вхождение переменной $x$ в данную формулу называется *связанным*, если $x$ является переменной входящих в эту формулу кванторов $\forall x$, $\exists x$ или находится в области действия входящих в эту формулу кванторов $\forall x$, $\exists x$; в противном случае вхождение переменной $x$ в данную формулу называется *свободным*. Например, вхождение переменной $x$ в формулу $\forall y\, A_1^2(x,y)\Rightarrow A_2^2(y,y)$ является свободным, первое и второе вхождение переменной $y$ - связанное, а третье и четвертое вхождение $y$ в эту формулу - свободное. Таким образом, одна и та же переменная в одной и той же формуле может иметь как связанные, так и свободные вхождения.

Переменная называется *свободной* (*связанной*) переменной в данной формуле, если существуют свободные (связанные) ее вхождения в эту формулу.

Формула называется *замкнутой*, если она не содержит никаких свободных переменных, т.е. либо все ее переменные связанные, либо переменных нет совсем. Например, формула $\forall x(\forall y A_1^2(x,y) \Rightarrow A_2^2(x,x))$ - замкнутая, а формула $\forall x A_1^2(x,y)$ - не замкнутая, ибо содержит свободную переменную $y$.

*Замыканием* формулы $\boldsymbol{A}$ называется формула, полученная из $\boldsymbol{A}$ приписыванием перед нею кванторов всеобщности по всем ее свободным переменным, при этом кванторы приписываются следующим образом: сначала записываем кванторы общности по всем свободным переменным $x$ (если они есть) в порядке возрастания их индексов, затем по свободным переменным $y$ (если они есть) в порядке возрастания их индексов и т.д. Так, замыканием формулы $A_1^3(x_3, x_1, y_2)$ будет формула:

$$\forall x_1 \forall x_3 \forall y_2 A_1^3(x_3,x_1,y_2),$$

а замыканием для формулы $\forall y_1 \exists x_3 A_1^3(x_1,y_1,x_3) \Rightarrow A_1^2(x_3,x_2)$ будет формула:

$$\forall x_1 \forall x_2 \forall x_3(\forall y_1 \exists x_3 A_1^3(x_1,y_1,x_3) \Rightarrow A_1^2(x_3,x_2)).$$

*Сначала узнайте факты, а потом интерпретируйте их как хотите.*

*М. Твен*

*Не существует фактов, есть лишь их интерпретации.*

*Ф. Ницше*

*Когда я беру слово, оно означает то, что я хочу, не больше и не меньше, - сказал Шалтай презрительно.*

*Вопрос в том, подчиняется ли оно вам, - сказала Алиса.*

*Л. Кэрролл*

## § 4. Интерпретация. Модель

*Интерпретацию* будем считать заданной, если:

1. Задано непустое множество $\mathcal{M}$, называемое областью интерпретации.
2. Заданы следующие соответствия:
   a) предикатным буквам $A_i^n$ поставлены в соответствие некоторые $n$ - местные предикаты (отношения) в $\mathcal{M}$;
   b) функциональным буквам $f_i^n$ поставлены в соответствие некоторые $n$ - аргументные функции, отображающие $\mathcal{M}^n$ в $\mathcal{M}$;

c) каждой предметной постоянной поставлен в соответствие некоторый (фиксированный) элемент из $\mathcal{M}$;

d) символам $\neg, \Rightarrow, \forall x, \exists x$ поставлен в соответствие их обычный смысл.

3. Считается, что предметные переменные пробегают всё множество $\mathcal{M}$.

Например, чтобы задать интерпретацию для формулы $\forall x\, A_1^3(x,y,b_1)$, нужно задать множество $\mathcal{M}$ - область интерпретации (область изменения переменных $x,y$). Из этой области $\mathcal{M}$ будем брать некоторый элемент, соответствующий предметной постоянной $b_1$. Далее нужно задать $3$-х местный предикат на $\mathcal{M}$, соответствующей предикатной букве $A_1^3$. Так, можно положить, что $\mathcal{M}=[0, \infty)$; предметной постоянной $b_1$ поставить в соответствие $1$, а $A_1^3(x,y,z)$ поставить в соответствие предикат $x + y \geq z$. Тогда формула $\forall x\, A_1^3(x,y,b_1)$ в заданной интерпретации запишется:

$$\forall x(x + y \geq 1)$$

и означает, что для любого $x$ $(x \in [0, \infty))$ сумма $x + y$ больше или равна $1$. Очевидно, что это отношение истинно при некоторых $y$ $(y \geq 1)$ и ложно при других значениях $y$ $(0 \leq y < 1)$. Если предметной постоянной $b_1$ поставить в соответствие $0$, а не $1$, то утверждение $\forall x(x + y \geq 0)$ будет истинно при любом значении свободной переменной $y$.

Легко видеть, что для той же формулы $\forall x\, A_1^3(x,y,b_1)$ можно построить бесчисленное множество других интерпретаций, выбирая различные множества $\mathcal{M}$, выбирая из $\mathcal{M}$, каким-то образом элемент, соответствующий $b_1$, и задавая различным образом на $\mathcal{M}$ трехместный предикат. Так, можно за $\mathcal{M}$ взять множество всех студентов Казани, за $b_1$ студента Иванова, а $A_1^3(x,y,z)$ поставить в соответствии предикат: «$x$ и $y$ учатся в той же группе, что и $z$». Тогда исходная формула $\forall x\, A_1^3(x,y,b_1)$ в этой интерпретации означает утверждение: $\forall x$($x$ и $y$ учатся в той же группе, что и Иванов). Это утверждение является ложным при каждом $y$, ибо не может быть, чтобы любой $x$ и некоторый $y$ учились в той же группе, что и Иванов.

При данной интерпретации всякая формула без свободных переменных (замкнутая формула) представляет собой высказывание, которое истинно либо ложно, а всякая формула со свободными переменными выражает некоторое отношение на $\mathcal{M}$, которое может быть истинно для одних значений из $\mathcal{M}$ и ложно для других.

Формула называется *выполнимой в данной интерпретации*, если она принимает значение *И* хотя бы для одной совокупности возможных значений её свободных переменных (если они есть). Если формула не содержит свободных переменных, то она называется выполнимой в том случае, если принимает значение *И* в этой интерпретации.

Формула называется *истинной в данной интерпретации*, если она принимает значение *И* для всех возможных значений её свободных

переменных (если они есть). Если же свободных переменных нет, то формула называется истинной, когда она принимает значение *И* в этой интерпретации.

Формула называется *ложной в данной интерпретации*, если она принимает значение *Л* для всех возможных значений её свободных переменных (если они есть). Если же свободных переменных нет, то формула называется ложной, когда она принимает значение *Л* в этой интерпретации. Очевидно, что формула ложна в данной интерпретации тогда и только тогда, когда она не выполнима в этой интерпретации. Так же ясно, что формула $A$ выполнима в данной интерпретации тогда и только тогда, когда она не является ложной в этой интерпретации.

Данная интерпретация называется *моделью* для множества формул $G$, если каждая формула из $G$ истинна в этой интерпретации.

## § 5. Свойства формул в данной интерпретации

Можно доказать следующие свойства формул в данной интерпретации (некоторые из них очевидны, другие читатель легко докажет самостоятельно).

1. Формула $A$ ложна в данной интерпретации тогда и только тогда, когда $\neg A$ истинно в этой же интерпретации. Формула $A$ истинна в данной интерпретации тогда и только тогда, когда $\neg A$ ложна в этой же интерпретации.

2. Никакая формула не может быть одновременно истинной и ложной в одной и той же интерпретации.

3. Если в данной интерпретации истинны $A$ и $A \Rightarrow B$, то истинно и $B$. Это утверждение легко доказать методом от противного (сравни с теоремой 1.1).

4. Формула $A \Rightarrow B$ ложна в данной интерпретации тогда и только тогда, когда $A$ истинно в этой интерпретации, а $B$ ложно. Доказательство этого утверждения следует из определения импликации.

5. Формула $A \& B$ выполнима в данной интерпретации тогда и только тогда, когда $A$ и $B$ принимают значение *И* одновременно хотя бы для одной совокупности значений своих свободных переменных. Если же свободных переменных нет, то формула $A \& B$ выполнима в данной интерпретации тогда и только тогда, когда обе формулы $A$ и $B$ истинны в этой интерпретации.

6. Формула $A \vee B$ выполнима в данной интерпретации, если хотя бы одна из них выполнима в этой интерпретации.

7. Формула $A \equiv B$ выполнима в данной интерпретации тогда и только тогда, когда $A$ и $B$ принимают значение *И* одновременно или значение *Л* (тоже одновременно) хотя бы для одной совокупности значений своих свободных переменных. Если же свободных переменных нет, то формула $A \equiv B$ выполнима в данной интерпретации тогда и только тогда, когда $A$ и $B$ принимают одинаковые истинностные значения в этой интерпретации.

8. Формула $\exists x A$ выполнима в данной интерпретации тогда и только тогда, когда $A$ принимает значение *И* хотя бы для одной совокупности значений её свободных переменных и хотя бы одного значения переменной $x$.

9. Формула $\forall x A$ истинна в данной интерпретации тогда и только тогда, когда в этой интерпретации истинно $A$.

Как следствие из этого утверждения получаем, что формула $A$ истинна в данной интерпретации тогда и только тогда, когда в этой интерпретации истинно замыкание формулы $A$.

Если формула $A$ замкнута, то в данной интерпретации $A$ означает некоторое высказывание (нет свободных переменных), следовательно, $A$ истинно либо ложно. Иначе, если $A$ замкнуто, то в любой данной интерпретации либо истинно $A$, либо истинно $\neg A$.

10. Рассмотрим некоторую пропозициональную форму. Если в пропозициональную форму вместо пропозициональных букв подставить формулы логики предикатов (вместо всех вхождений одной и той же пропозициональной буквы подставлять одну и ту же формулу), получим некоторую формулу, которая называется *частным случаем пропозициональной формы*. Тогда легко доказать следующее утверждение.

Всякий частный случай любой тавтологии истинен в каждой интерпретации.

*То что имеет основанием истину, следует напоминать, не боясь показаться надоедливым.*
*Н.И. Пирогов*

## § 6. Логически общезначимые формулы. Выполнимые и равносильные формулы

Формула логики предикатов называется *логически общезначимой*, если она истинна в любой интерпретации.

Логическая общезначимость формулы означает, что какую бы ни выбирали область интерпретации и какие бы соответствия, указанные в начале параграфа 4 данной главы, ни задавали, мы всегда будем получать истинные отношения или высказывания.

Примером логически общезначимых формул, очевидно, являются следующие формулы: $A \Rightarrow A$, $\forall x A \Rightarrow \exists x A$, $A \equiv A$.

Если формула $A$ является логически общезначимой, то будем записывать, иногда в сокращенном виде: «$\models A$» и эта запись читается: «$A$ является логически общезначимой формулой (логики предикатов)».

Формула логики предикатов $A$ называется *выполнимой*, если существует интерпретация, в которой выполнима $A$.

Примером выполнимой формулы является формула $\forall x A(x,y,b_1)$. Действительно, если взять $\mathfrak{M}=[0,\infty)$; $A(x,y,b_1)$ поставить в соответствие

предикат (отношение) $x + y \geq z$; $b_1$ поставить в соответствие число *2*, то наша формула в этой интерпретации означает, что $\forall x(x + y \geq 2)$. Последнее будет истинно, например, при любых значениях $y \geq 2$. Следовательно, заданная формула выполнима в этой интерпретации. Таким образом, для нашей формулы существует интерпретация, в которой она выполнима, поэтому эта формула выполнима.

Имеют место следующие очевидные утверждения. Формула ***A*** логически общезначима тогда и только тогда, когда формула $\neg A$ не является выполнимой. Формула ***A*** выполнима тогда и только тогда, когда формула $\neg A$ не является логически общезначимой.

Будем называть формулу логики предикатов ***A*** *противоречием*, если формула $\neg A$ является логически общезначимой, или, что то же самое, если формула ***A*** ложна во всякой интерпретации.

Говорят, что формула логики предикатов ***A*** *логически влечет* формулу логики предикатов ***B*** , если в любой интерпретации формула ***B*** принимает значение *И* при каждой совокупности значений свободных переменных (входящих в ***A*** и ***B***), при которых формула ***A*** приняла значение *И*. Иначе говорят, что ***B*** является *логическим следствием* формулы ***A***. В этом случае записываем

$$A \models B$$

и читаем: «из ***A*** логически следует ***B***» или «***B*** является логическим следствием из ***A***». Отметим, что «$A \models B$» не является формулой, а является метаутверждением относительно формул ***A*** и ***B*** (логики предикатов).

Формулы ***A*** и ***B*** логики предикатов называют *равносильными* (логически эквивалентными), если каждая из них логически влечет другую.

Если формулы ***A*** и ***B*** равносильны, то, как и в логике высказываний, записываем: $A \sim B$.

Имеют место следующие теоремы.

**Теорема 2.1**.Формула ***A*** логически влечет формулу ***B*** тогда и только тогда, когда формула $A \Rightarrow B$ логически общезначима.

(В сокращенном виде эту теорему можно записать: $A \models B$ тогда и только тогда, когда $\models A \Rightarrow B$).

**Доказательство.** Пусть $A \Rightarrow B$ логически общезначима, т.е. истинна в каждой интерпретации. Если ***B*** не является логическим следствием ***A***, то при некоторой интерпретации формула ***B*** принимает значение *Л* для некоторой совокупности значений свободных переменных, при которой ***A*** принимает значение *И*. Но тогда при этой совокупности значений свободных переменных $A \Rightarrow B$ будет ложным, что противоречит условию логически

общезначимости $A \Rightarrow B$. Итак, если $A \Rightarrow B$ логически общезначима, то $A$ логически влечёт $B$ .

Обратное тоже доказывается легко. Действительно, если $A$ логически влечёт $B$, то по определению импликации, очевидно, $A \Rightarrow B$ истинно в каждой интерпретации, следовательно, логически общезначима. Теорема доказана.

**Теорема 2.2.** Формулы $A$ и $B$ равносильны (логически эквивалентны) тогда и только тогда, когда формула $A \equiv B$ логически общезначима.

**Доказательство.** Согласно определению $A$ и $B$ равносильны тогда и только тогда, когда каждая из них влечёт другую, т.е. по предыдущей теореме тогда и только тогда, когда $A \Rightarrow B$ и $B \Rightarrow A$ логически общезначимы, т.е. тогда и только тогда, когда $A \equiv B$ логически общезначима.

**Теорема 2.3**. Если формула $A$ логически влечёт формулу $B$ и $A$ истинно в данной интерпретации, то в этой же интерпретации истинно и $B$.

Эту теорему легко доказать от противного.

Если формула $A$ является замкнутой формулой, то очевидно при приписывании к $A$ любых кванторов получим формулу равносильную $A$, т.е.:

$A \sim Q_1 Q_2 \ldots Q_n A$,

здесь $Q_1 Q_2 \ldots Q_n$ любая совокупность кванторов по любым переменным.

Также введем понятие логического следствия из заданного множества формул логики предикатов.

Формула $B$ называется *логическим следствием* формул $A_1, A_2, \ldots, A_n$, если в любой интерпретации формула $B$ принимает значение *И* при каждой совокупности значений свободных переменных (входящих в $B$ и $A_1, A_2, \ldots, A_n$), при которых одновременно все формулы $A_1, A_2, \ldots, A_n$ приняли значение *И*. Иными словами говорят, что $B$ является *логическим следствием* формул $A_1, A_2, \ldots, A_m$, $m \geq 1$. В этом случае записывается $A_1, A_2, \ldots, A_m \models B$.

$$\neg \forall x A \sim \exists x \neg A$$

$$\neg \exists x A \sim \forall x \neg A$$

## § 7. Правила перенесения отрицания через кванторы

Прежде чем рассматривать общий случай произвольной формулы, исследуем формулы частного вида $\neg \forall x P(x)$ и $\neg \exists x P(x)$, где $P$ - одноместная предикатная буква; более того, будем рассматривать интерпретации этих формул на конечных $n$-элементных множествах (областях интерпретации).

Пусть заданы формулы $\neg \forall x P(x)$ и $\neg \exists x P(x)$ и задана произвольная интерпретация этих формул на $n$-элементном множестве $M = \{a_1, a_2, \ldots, a_n\}$. В §

2 установлено, что квантор общности является обобщением конъюнкции, а квантор существования - дизъюнкции и для *n*- элементных областей интерпретации имеют место соотношения (2.7) и (2.8):

$\forall x P(x)$ равносильно $P(a_1)\&P(a_2)\&...\&P(a_n)$,

$\exists x P(x)$ равносильно $P(a_1)\vee P(a_2)\vee...\vee P(a_n)$.

Очевидно, что высказывания равносильны тогда и только тогда, когда равносильны отрицания этих высказываний, поэтому первое из этих соотношений эквивалентно следующему:

$$\neg\forall x P(x) \sim \neg(P(a_1)\&P(a_2)\&...\&P(a_n)) \sim \neg P(a_1)\vee\neg P(a_2)\vee...\vee\neg P(a_n).$$

Правая часть полученного соотношения есть не что иное, как запись высказывания $\exists x \neg P(x)$. Таким образом, для *n*- элементных областей интерпретации имеем:

$$\neg\forall x P(x) \sim \exists x \neg P(x). \qquad (2.9)$$

Аналогично получим:

$$\neg\exists x P(x) \sim \neg(P(a_1)\vee P(a_2)\vee...\vee P(a_n)) \sim \neg P(a_1)\&\neg P(a_2)\&...\&\neg P(a_n),$$

следовательно, для *n*-элементных областей интерпретации имеет место:

$$\neg\exists x P(x) \sim \forall x \neg P(x) \qquad (2.10)$$

Соотношения (2.9) и (2.10) показывают, что при перенесении отрицания через кванторы последние меняются на двойственные. Покажем, что эти правила имеют место для указанных формул, но уже без ограничения конечности областей интерпретации.

Рассмотрим формулу $\neg\forall x P(x)$. Возьмем произвольную (но фиксированную) интерпретацию. В каждой интерпретации эта формула означает некоторое высказывание (так как не имеет свободных переменных).

Пусть высказывание $\neg\forall x P(x)$ истинно. Тогда высказывание $\forall x P(x)$ - ложно, следовательно, существует значение переменной *x*, для которого $P(x)$ ложно. Обозначим одно из таких значений через *b*. Итак, $b\in \mathcal{M}$ и $P(b)$ - ложно. В таком случае $\neg P(b)$ истинно, т.е. существует такое *x* (равное *b*), что $\neg P(x)$ истинно, поэтому $\exists x \neg P(x)$ истинно.

Обратно, пусть теперь высказывание $\exists x \neg P(x)$ истинно. Тогда по определению квантора существования найдется такое значение переменной *x*, что $\neg P(x)$ истинно. Обозначим одно из таких значений через *b*. Итак, $b\in \mathcal{M}$ и $\neg P(b)$ истинно. Следовательно $P(b)$ ложно. Но тогда по определению квантора общности $\forall x P(x)=Л$, а $\neg\forall x P(x)$ истинно. В результате мы доказали,

что в каждой интерпретации $\neg\forall x P(x)$ истинно тогда и только тогда, когда истинно $\exists x \neg P(x)$, поэтому:

$\neg\forall x P(x) \sim \exists x \neg P(x)$.

Аналогичным образом можно установить, что:

$\neg\exists x P(x) \sim \forall x \neg P(x)$.

Далее рассмотрим формулу $\neg\forall x A^2(x,y)$ с одной двуместной предикатной буквой $A^2$. Возьмем для этой формулы произвольную (но фиксированную) интерпретацию. В выбранной интерпретации возьмем произвольное значение свободной переменной $y$, скажем, $y=c$ $(c \in \mathfrak{M})$. Выражение $\neg\forall x A^2(x,c)$ представляет собой отрицание результата навешивания квантора общности на одноместный предикат $A^2(x,c)$ и по доказанному истинностные значения $\neg\forall x A^2(x,c)$ и $\exists x \neg A^2(x,c)$ совпадают, следовательно:

$$\neg\forall x A^2(x,y) \sim \exists x \neg A^2(x,y). \qquad (2.11)$$

Ясно, что аналогично соотношению (2.11) можно доказать следующие равносильности:

$$\neg\forall x_i A_k^n(x_1,x_2,\ldots,x_n) \sim \exists x_i \neg A_k^n(x_1,x_2,\ldots,x_n),$$

$$\neg\exists x_i A_k^n(x_1,x_2,\ldots,x_n) \sim \forall x_i \neg A_k^n(x_1,x_2,\ldots,x_n).$$

Можно доказать, что и для произвольной формулы $\boldsymbol{A}$ имеют место:

$$\neg\forall x \boldsymbol{A} \sim \exists x \neg \boldsymbol{A}, \qquad (2.12)$$

$$\neg\exists x \boldsymbol{A} \sim \forall x \neg \boldsymbol{A}. \qquad (2.13)$$

Если отрицание стоит перед несколькими кванторами, то, используя (2.12) и (2.13), отрицание можно переносить последовательно через каждый из кванторов, изменяя его на двойственный.

В результате доказана следующая теорема.

**Теорема 2.4**. Отрицание формулы, начинающейся с кванторов, равносильно формуле, полученной заменой каждого квантора на двойственный и перенесением знака отрицания за кванторы.

Высказывания (2.12) и (2.13) являются аналогами законов де Моргана. Используя их, легко выразить один из кванторов через другой. Для этого применим операцию отрицания к левым и правым частям соотношений (2.12) и (2.13). Получим соответственно:

$$\forall x \boldsymbol{A} \sim \neg\exists x \neg \boldsymbol{A}, \qquad (2.14)$$

$$\exists x \boldsymbol{A} \sim \neg\forall x \neg \boldsymbol{A}. \qquad (2.15)$$

Равносильности (2.14) и (2.15) показывают, что при определении формул логики предикатов можно было ввести только один из кванторов. Например, можно считать по определению, что для произвольной формулы $\boldsymbol{A}$

выражение $\forall x A$ есть формула, а выражение $\exists x A$ уже является обозначением для формулы $\neg\forall x \neg A$.

*Кто хочет обрести счастье или мудрость, тот должен искать перемен.*

*Конфуций*

## § 8. Правила перестановки кванторов

Пусть $A$ - произвольная формула логики предикатов. Рассмотрим для $A$ произвольную, но фиксированную интерпретацию. Сразу же из определения кванторов получаем, что там, где истинно $\forall x \forall y A$, истинно и $\forall y \forall x A$, и наоборот . В силу произвольности интерпретации следует, что

$$\forall x \forall y A \sim \forall y \forall x A. \qquad (2.16)$$

Точно также получаем что

$$\exists x \exists y A \sim \exists x \exists y A. \qquad (2.17)$$

Таким образом, при перемене мест стоящих рядом одноименных кванторов получаем равносильные формулы. Итак, одноименные кванторы, стоящие рядом, можно переставлять местами.

Известно, что формулы $A$ и $B$ равносильны тогда и только тогда, когда $A \equiv B$ является логически общезначимой формулой (теорема 2.2). Тогда из (2.16) и (2.17) получаем, что формулы

$\forall x \forall y A \equiv \forall y \forall x A$ и $\exists x \exists y A \equiv \exists y \exists x A$

являются логически общезначимыми.

Разноименные кванторы, оказывается, можно переставлять не всегда. Докажем теорему.

**Теорема 2.5.** Для каждой формулы $A$ и любых предметных переменных $x$ и $y$ формула

$$\exists x \forall y A \Rightarrow \forall y \exists x A \qquad (2.18)$$

логически общезначима, а обратная импликация, т.е. формула

$$\forall y \exists x A \Rightarrow \exists x \forall y A \qquad (2.19)$$

не всегда является логически общезначимой.

**Доказательство**. Для доказательства логической общезначимости формулы (2.18) фиксируем произвольную интерпретацию формулы $A$, и из определений кванторов сразу получаем, что формула (2.18) истинна. Таким образом, в любой интерпретации формула (2.18) истинна, следовательно, она логически общезначима.

Чтобы доказать, что формула (2.19) не всегда является логически общезначимой, достаточно привести пример формулы $\boldsymbol{A}$ и интерпретации для нее, где формула (2.19) не истинна. Пусть область интерпретации $\mathcal{M}$ - множество действительных чисел и формула $\boldsymbol{A}$ означает предикат $x>y$, тогда высказывание

$$\forall y \exists x(x>y) \qquad (2.20)$$

означает, что для любого числа $y$ существует число $x$ большее, чем $y$. Это высказывание истинно. Высказывание, полученное из (2.20) перестановкой кванторов $\exists x \forall y(x>y)$, означает, что существует число $x$ больше любого другого числа и, очевидно, является ложным. Тогда ложна импликация: $\forall y \exists x(x>y) \Rightarrow \exists x \forall y(x>y)$. Следовательно, формула (2.19) не истинна в приведенной интерпретации, т.е. не является логически общезначимой. Теорема доказана.

Заметим, что в частном случае формула (2.19) может оказаться и логически общезначимой, например, когда $\boldsymbol{A}$ является замкнутой формулой. В этом случае, как известно (см. **§** 6), формула $\boldsymbol{A}$ равносильна формуле $Q_1Q_2...Q_n\boldsymbol{A}$, где $Q_1,Q_2,...,Q_n$ любая совокупность кванторов. Поэтому формула (2.19) будет логически общезначимой.

Однако подчеркнем еще раз, что для произвольной формулы перестановка разноименных кванторов не всегда приводит к равносильным формулам.

## § 9. Правила переименования связанных переменных

Рассмотрим формулы $\forall xA(x)$ и $\forall yA(y)$. Очевидно, что в каждой интерпретации из истинности первой следует истинность второй, и наоборот, поэтому эти формулы равносильны: $\forall xA(x) \sim \forall yA(y)$.

Таким образом, переименование переменной $x$ на $y$ в кванторе и в области действия этого квантора привело к формуле равносильной исходной.

В математическом анализе имеется аналогичное правило. Например, замена переменной в подинтегральном выражении определенного интеграла не меняет его величины:

$$\int_0^1 \cos x\,dx = \int_0^1 \cos y\,dy$$

Точно так же в сумме индекс суммирования можно переименовывать:

$$\sum_{n=1}^{m} x^n/n^2 = \sum_{k=1}^{m} x^k/k^2.$$

В последнем примере переименовывается $n$ на $k$, но нельзя переименовывать $m$, ибо это свободная переменная.

В формуле $\forall xA(x)$ можно заменить переменную $x$ на $y$. Ясно, что если $x$ заменить любой другой переменной, то полученная формула будет равносильна исходной. Можно показать, что и в произвольной формуле

переименование (замена) связанных переменных на любые другие приводит к формуле равносильной исходной. Имея в виду дальнейшие приложения, будем придерживаться следующего правила переименования связанных переменных.

Пусть $\boldsymbol{A}$ - произвольная формула логики предикатов. Формулу $\boldsymbol{A}_0$ получим из $\boldsymbol{A}$ заменой связанных переменных другими переменными, отличными от всех свободных переменных формулы $\boldsymbol{A}$, причем заменяемая переменная в формуле $\boldsymbol{A}$ должна меняться одинаковым образом всюду в области действия квантора, связывающего данную переменную и в самом кванторе. Тогда $\boldsymbol{A}_0$ равносильна $\boldsymbol{A}$.

При переименовании связанных переменных мы не обязаны переименовывать их всюду, где они входят в формулу $\boldsymbol{A}$, а лишь только переменную выбранного нами квантора и в области действия этого квантора. Это значит, что одинаковые переменные, для которых связывающие их кванторы имеют различные области действия, могут переименовываться разным образом или одна из них может переименовываться, а другая нет.

Рассмотрим действие приведенного правила на примерах. Пусть имеем формулу

$$\forall x A(x) \Rightarrow \exists x B(x) \Rightarrow C(x) \qquad (2.21)$$

Переименовав связанную переменную $x$ на $y$, в первой посылке получим формулу, равносильную исходной:

$\forall y A(y) \Rightarrow \exists x B(x) \Rightarrow C(x)$.

Из формулы (3.21) переименованием можно получить формулу

$\forall x A(x) \Rightarrow \exists z B(z) \Rightarrow C(x)$

или

$\forall x A(x) \Rightarrow \exists y B(y) \Rightarrow C(x)$.

При этом каждая полученная формула будет равносильна исходной формуле (2.21).

Отметим еще раз, что переименовываются только связанные переменные, а свободные не трогаются. Так, в формуле (2.21) последнее вхождение $x$ свободно, поэтому при любых переименованиях она остается неизменной.

Рассмотрим еще один пример. Пусть задана формула

$$\forall x(\exists y P(x,y) \Rightarrow \forall y Q(x,y) \Rightarrow R(x)). \qquad (2.22)$$

Переименовывая в этой формуле переменную $x$, мы должны заменить ее одинаковым образом всюду, где она входит, ибо $x$ в формуле (2.22) является либо переменной квантора, либо находится в области действия квантора. Например, переименовав $x$ на $z$ , получим следующую формулу, равносильную исходной:

$\forall z(\exists y P(z,y) \Rightarrow \forall y Q(z,y) \Rightarrow R(z))$

В формуле (2.22) переменную $y$ можно переименовать в первой посылке, например, на переменную $v$, а во второй оставить без изменения, либо заменить, например, переменной $u$. В последнем случае получим формулу: $\forall x(\exists v P(x,v) \Rightarrow \forall u Q(x,u) \Rightarrow R(x))$. Если учесть еще переименование

переменной $x$, то имеем: $\forall z(\exists v P(z,v) \Rightarrow \forall u Q(z,u) \Rightarrow R(z))$. Полученные формулы тоже равносильны исходной формуле (2.22).

## § 10. Правила вынесения кванторов за скобки. Предваренная нормальная форма

Выясним, каким образом выносятся кванторы за скобки, при этом получим и правила внесения кванторов под скобки.

**Теорема 2.6.** Пусть $\boldsymbol{A}$ обозначает формулу, не имеющую свободных вхождений переменной $x$, $\boldsymbol{B}(x)$ и $\boldsymbol{C}(x)$ -произвольные формулы , возможно, и содержащие свободные вхождения $x$. Тогда:

$$\boldsymbol{A} \& \forall x \boldsymbol{B}(x) \sim \forall x(\boldsymbol{A} \& \boldsymbol{B}(x)), \tag{2.23}$$

$$\boldsymbol{A} \vee \forall x \boldsymbol{B}(x) \sim \forall x(\boldsymbol{A} \vee \boldsymbol{B}(x)), \tag{2.24}$$

$$\boldsymbol{A} \& \exists x \boldsymbol{B}(x) \sim \exists x(\boldsymbol{A} \& \boldsymbol{B}(x)), \tag{2.25}$$

$$\boldsymbol{A} \vee \exists x \boldsymbol{B}(x) \sim \exists x(\boldsymbol{A} \vee \boldsymbol{B}(x)), \tag{2.26}$$

$$(\forall x \boldsymbol{B}(x)) \& \forall x \boldsymbol{C}(x) \sim \forall x(\boldsymbol{B}(x) \& \boldsymbol{C}(x)), \tag{2.27}$$

$$(\exists x \boldsymbol{B}(x)) \vee \exists x \boldsymbol{C}(x) \sim \exists x(\boldsymbol{B}(x) \vee \boldsymbol{C}(x)). \tag{2.28}$$

Кроме того формулы

$$(\forall x \boldsymbol{B}(x)) \vee \forall x \boldsymbol{C}(x) \Rightarrow \forall x(\boldsymbol{B}(x) \vee \boldsymbol{C}(x)), \tag{2.29}$$

$$\exists x(\boldsymbol{B}(x) \& \boldsymbol{C}(x)) \Rightarrow (\exists x \boldsymbol{B}(x)) \& \exists x \boldsymbol{C}(x) \tag{2.30}$$

логически общезначимы, а импликации, обратные к (2.29) и (2.30), уже не всегда логически общезначимы.

**Доказательство.** Докажем соотношение (2.23). Для этого возьмем произвольную, но фиксированную интерпретацию формулы $\boldsymbol{A} \& \forall x \boldsymbol{B}(x)$.

Если свободных переменных в формуле $\boldsymbol{A} \& \forall x \boldsymbol{B}(x)$ нет, то в интерпретации получим высказывание. Пусть это высказывание истинно, тогда истинны $\boldsymbol{A}$ и $\boldsymbol{B}(x)$ при любом значении $x$ из области интерпретации $\mathcal{M}$. Поэтому будет истинно высказывание $\forall x(\boldsymbol{A} \& \boldsymbol{B}(x))$. Аналогичным образом из истинности $\forall x(\boldsymbol{A} \& \& \boldsymbol{B}(x))$ следует истинность $\boldsymbol{A} \& \forall x \boldsymbol{B}(x)$.

Пусть формула $\boldsymbol{A} \& \forall x \boldsymbol{B}(x)$ имеет свободные переменные, для определенности пусть это $y_1, y_2, \ldots, y_n$. Придадим им произвольные значения из $\mathcal{M}$, например, $b_1, b_2, \ldots, b_n$ соответственно. Положим, что при указанных

значениях $y_1, y_2, ...,y_n$ формула $A\&\forall xB(x)$ принимает значение истина. Последнее означает, что при $y_1=b_1, y_2=b_2,...,y_n= b_n$ истинны $A$ и $B(x)$ при любом $x$ из $\mathfrak{M}$, следовательно, при выбранных значениях свободных переменных истинно высказывание $\forall x(A\&B(x))$. Очевидно, верно и обратное. Соотношение (2.23) доказано.

Доказательство равносильностей (2.24) - (2.28) и логической общезначимости формул (2.29) и (2.30) можно провести аналогично доказательству равносильности (2.23), т.е. фиксированием интерпретации.

Для завершения доказательства теоремы осталось доказать, что импликации, обратные к (2.29) и (2.30), точнее, формулы

$$\forall x(B(x)\vee C(x))\Rightarrow(\forall xB(x))\vee\forall xC(x), \qquad (2.31)$$

$$(\exists xB(x))\&\exists xC(x)\Rightarrow\exists x(B(x)\&C(x)) \qquad (2.32)$$

не всегда являются логически общезначимыми. Чтобы доказать это, достаточно для каждой из импликаций (2.31) и (2.32) указать формулы $B(x)$ и $C(x)$ и для них привести хотя бы одну интерпретацию, где формулы (2.31) и (2.32) не истинны.

Пусть для формулы (2.31) область интерпретации есть множество целых чисел, формуле $B(x)$ соответствует предикат "$x$ - четное число", а формуле $C(x)$ -предикат "$x$ - нечетное число". Тогда, считая, что нуль четно, получим истинное высказывание $\forall x$($x$ - четное число или $x$ - нечетное число ), в то время, как высказывание [ $\forall x$($x$-нечетное число ) или $\forall x$($x$ - четное число)] ложно. Поэтому формула (2.31) в приведенной интерпретации ложна, следовательно, не логически общезначима.

Для формулы (2.32) можно взять ту же интерпретацию, что и для формулы (2.31). При этом легко видеть, что формула (2.32) ложна, следовательно, не логически общезначима. Теорема доказана.

Правила вынесения кванторов за скобки, когда задана импликация некоторых формул, следуют из теоремы.

**Теорема 2.7**. Пусть $A$ - произвольная формула, не содержащая свободных вхождений переменой $x$, а $B(x)$- произвольная формула, возможно, и содержащая свободные вхождения $x$. Тогда

$$\exists xB(x)\Rightarrow A \sim \forall x(B(x)\Rightarrow A); \qquad (2.33)$$

$$A\Rightarrow\forall xB(x) \sim \forall x(A\Rightarrow B(x)), \qquad (2.34)$$

$$\forall xB(x)\Rightarrow A \sim \exists x(B(x)\Rightarrow A), \qquad (2.35)$$

$$A\Rightarrow\exists xB(x) \sim \exists x(A\Rightarrow B(x)). \qquad (2.36)$$

**Доказательство**. Докажем соотношение (2.33). Рассмотрим формулу:

$\forall x(\boldsymbol{B}(x) \Rightarrow \boldsymbol{A})$. (2.37)

Вместо формулы (2.37) введем равносильную формулу $\forall x(\neg \boldsymbol{B}(x) \vee \boldsymbol{A})$, которая по (2.24) равносильна формуле $(\forall x \neg \boldsymbol{B}(x)) \vee \boldsymbol{A}$, которая равносильна формуле $\neg \forall x \boldsymbol{B}(x) \Rightarrow \boldsymbol{A}$. Используя соотношение (2.15) между кванторами, получим формулу $(\exists x \boldsymbol{B}(x)) \Rightarrow \boldsymbol{A}$. Таким образом, формула $\forall x(\boldsymbol{B}(x) \Rightarrow \boldsymbol{A})$ равносильна формуле $(\exists x \boldsymbol{B}(x)) \Rightarrow \boldsymbol{A}$, что и требовалось доказать.

Доказательство равносильностей (2.34)-(2.36) можно провести примерно так же, как доказывалось соотношение (2.33).

Из изложенного выше видно, что при вынесении кванторов за скобки для указанных формул кванторы могут выноситься без изменения, либо меняться на двойственные. В общем случае, оказывается, нужно проводить еще и переименование переменных, прежде чем вынести квантор за скобки. Например, в формуле $\forall x \boldsymbol{B}(x) \vee \forall x \boldsymbol{C}(x)$ попытка вынести квантор $\forall x$ без изменения за скобки приводит к неравносильной формуле. Если при вынесении квантора за скобки его сменить на двойственный, то полученная формула тоже не всегда равносильна исходной. Именно поэтому необходимо переименование связанных переменных.

**Теорема 2.8**. Пусть $\boldsymbol{B}(x)$ и $\boldsymbol{C}(x)$ - произвольные формулы логики предикатов, которые, может быть, содержат свободные вхождения переменной $x$, тогда имеют место следующие равносильности:

$\forall x \boldsymbol{B}(x) \Rightarrow \forall x \boldsymbol{C}(x) \sim \exists y \forall z(\boldsymbol{B}(y) \Rightarrow \boldsymbol{C}(z))$, (2.38)

$\forall x \boldsymbol{B}(x) \Rightarrow \exists x \boldsymbol{C}(x) \sim \exists x(\boldsymbol{B}(x) \Rightarrow \boldsymbol{C}(x))$, (2.39)

$\exists x \boldsymbol{B}(x) \Rightarrow \exists x \boldsymbol{C}(x) \sim \forall y \exists z(\boldsymbol{B}(y) \Rightarrow \boldsymbol{C}(z))$, (2.40)

$\exists x \boldsymbol{B}(x) \Rightarrow \forall x \boldsymbol{C}(x) \sim \forall y \forall z(\boldsymbol{B}(y) \Rightarrow \boldsymbol{C}(z))$, (2.41)

где $z$ и $y$ - переменные, отличные от всех свободных переменных формул $\boldsymbol{B}(x)$ и $\boldsymbol{C}(x)$, а $\boldsymbol{B}(y)$ и $\boldsymbol{C}(z)$ -формулы полученные из $\boldsymbol{B}(x)$ и $\boldsymbol{C}(x)$ соответственно при переименовании связанной переменной $x$ на $y$ и $z$.

**Доказательство.** Рассмотрим формулу

$\forall x \boldsymbol{B}(x) \Rightarrow \forall x \boldsymbol{C}(x)$. (2.42)

Пусть $y$ и $z$ - переменные отличные от всех свободных переменных формулы (2.42). Проведем переименование связанной переменной $x$ в посылке этой формулы на переменную $y$, а в заключении - на $z$ . Получим равносильную формулу:

$\forall y \boldsymbol{B}(y) \Rightarrow \forall z \boldsymbol{C}(z)$. (2.43)

По построению в формуле (2.43) выражение $\forall z \boldsymbol{C}(z)$ не содержит свободных вхождений переменной $y$, тогда по равносильности (2.35) можно

вынести за скобки квантор $\forall y$, причем он сменится на двойственный, т.е. получим $\exists y(\boldsymbol{B}(y) \Rightarrow \forall z \boldsymbol{C}(z))$.

Так как формула $\boldsymbol{B}(y)$ не содержит свободных вхождений $z$, то, используя равносильность (2.34), получим требуемое соотношение (2.38).

Для доказательства соотношения (2.39) выразим импликацию в левой части соотношения (2.39) через $\neg$ и $\vee$:

$\forall x \boldsymbol{B}(x) \Rightarrow \exists x \boldsymbol{C}(x) \sim \neg(\forall x \boldsymbol{B}(x)) \vee \exists x \boldsymbol{C}(x) \sim (\exists x \neg \boldsymbol{B}(x)) \vee \exists x \boldsymbol{C}(x)$.

Далее по доказанной равносильности (2.28) получаем: $\exists x(\neg \boldsymbol{B}(x) \vee \boldsymbol{C}(x))$ откуда и следует равносильность (2.39).

Равносильности (2.40) и (2.41) доказываются аналогично как для соотношения (2.38).

Замечание 1. При доказательстве равносильности (2.38) мы вынесли за скобки квантор из посылки, а затем, из заключения, хотя, как легко видеть, можно сделать это и в обратной последовательности: сначала вынести квантор из заключения, а затем из посылки. В этом случае вместо (2.38) получим

$$\forall x \boldsymbol{B}(x) \Rightarrow \forall x \boldsymbol{C}(x) \sim \forall z \exists y(\boldsymbol{B}(y) \Rightarrow \boldsymbol{C}(z)) \qquad (2.44)$$

Так как левые части равносильностей (2.38) и (2.44) совпадают, то имеем:

$$\exists y \forall z(\boldsymbol{B}(y) \Rightarrow \boldsymbol{C}(z)) \sim \forall z \exists y(\boldsymbol{B}(y) \Rightarrow \boldsymbol{C}(z)). \qquad (2.45)$$

Известно, что в общем случае разноименные кванторы не перестановочны, а в частном случае (2.45) разноименные кванторы оказались перестановочными. Таким образом, в равносильности (2.45) в правой части порядок кванторов несущественен. Можно показать, что и в правых частях равносильностей (2.40) и (2.41) порядок кванторов не существенен.

Замечание 2. Равносильности (2.38), (2.40) и (2.41) показывают, что при вынесении кванторов за скобками получили не один квантор, как это было ранее, а уже два квантора.

Для рассмотренного выше примера, проведя переименования переменных, а затем используя равносильность (2.41) легко получить, что

$$(\forall x \boldsymbol{B}(x)) \vee \forall x \boldsymbol{C}(x) \sim \forall z \forall y(\boldsymbol{B}(y) \vee \boldsymbol{C}(z)). \qquad (2.46)$$

Также нетрудно получить, что

$$(\exists x \boldsymbol{B}(x)) \& \exists x \boldsymbol{C}(x) \sim \exists z \exists y(\boldsymbol{B}(y) \& \boldsymbol{C}(z)). \qquad (2.47)$$

Таким образом, из формул $(\forall x \boldsymbol{B}(x)) \vee \forall x \boldsymbol{C}(x)$ и $(\exists x \boldsymbol{B}(x)) \& \exists x \boldsymbol{C}(x)$ мы все же вынесли кванторы за скобки, но за скобками оказались уже два квантора с различными переменными. Сравнивая равносильности (2.46) и (2.47) с равносильностями (2.27) и (2.28), видим, что в последних кванторы вынесены без всякого изменения и удвоения их.

Из равносильностей (2.33) - (2.36) и (2.38)-(2.41) очевидным образом следует, что для любой формулы можно добиться, чтобы сначала были записаны кванторы, а затем формула, не имеющая кванторов, т.е. "вынести" кванторы за скобки. Здесь применены кавычки, т.к. для получения равносильных формул кванторы выносятся за скобки, возможно, оставаясь неизмененными, либо меняясь на двойственные, либо выносятся за скобки

только после переименования связанных переменных (в самом кванторе и области его действия). При этом переименование переменных осуществляется по правилам, описанным в предыдущем параграфе.

Формула вида: $Q_1Q_2...Q_n\boldsymbol{B}$, где $Q_1,Q_2,...,Q_n$ любая совокупность кванторов, а формула $\boldsymbol{B}$ не содержит кванторов называется формулой в *предваренной нормальной форме* или в *пренексной нормальной форме.*

Для формулы $\boldsymbol{A} \sim Q_1Q_2...Q_n\boldsymbol{B}$ совокупность кванторов $Q_1,Q_2,...,Q_n$ называется *префиксом* формулы $\boldsymbol{A}$, а формула $\boldsymbol{B}$ – *матрицей* формулы $\boldsymbol{A}$. Будем дополнительно считать, что матрица приведена к конъюнктивной нормальной форме

Из доказанных выше теорем легко следует следующая теорема.

**Теорема 2.9**. Для каждой формулы логики предикатов существует равносильная ей формула в предваренной нормальной форме.

*Усердие все превозмогает.*
*Козьма Прутков*

## § 11. Вопросы и темы для самопроверки

1. Понятие предиката.
2. Кванторы. Использование кванторов и предикатов для символизации языка.
3. Термы, элементарные формулы и формулы логики предикатов.
4. Свободные и связанные переменные. Замкнутые формулы. Замыкание формулы.
5. Интерпретация, выполнимые, истинные и ложные в данной интерпретации формулы.
6. Модель.
7. Свойства формул в данной интерпретации.
8. Логически общезначимые формулы. Выполнимые формулы.
9. Логическое следствие в логике предикатов. Равносильные формулы.
10. Правила перенесения отрицания через кванторы.
11. Можно ли переставлять рядом стоящие одноименные кванторы?
12. Можно ли переставлять рядом стоящие разноименные кванторы?
13. Определение предваренных нормальных форм. Для каждой ли формулы логики предикатов существует предваренная нормальная форма?
14. Алгоритмы нахождения предваренных нормальных форм.

*Если тебя не совсем Одиссеева кинула сметка,*
*Дело исполнить свое вполне ты надеяться можешь.*

*Гомер (Одиссея)*

## § 12. Упражнения

*Символизация языка. Предикаты, кванторы*

1. Какие из следующих выражений являются предикатами:

а) число $x$ – простое число;

б) $x=y+z;$
в) $x=2y+3;$
г) $2x+y;$
} здесь $x$, $y$ – действительные числа;

д) все подобные треугольники равны;

е) $x^2+y^2<0$ ($x$, $y$ – действительные числа);

ж) все четные числа делятся на число $y$;

з) все четные числа делятся на $2$;

и) $8$ – нечетное число;

к) имеется бесчисленное множество различных простых чисел;

л) число $2^{67}-1$ не является простым;

м) представьте число $2^{67}-1$ в виде произведения двух чисел, отличных от единицы и самого числа.

Выделите среди предикатов высказывания.

2. Запишите символически следующие предложения:

а) для всякого числа $x$ существует такое число $y$, что $x+y=5$;

б) для любого числа $y$ найдется хотя бы одно число $x$, что $y-x<0$;

в) при любом $x$, не равном нулю, существует $y$ такое, что $x/y=2$;

г) для любых чисел $x$ и $y$ имеет место равенство $x+y=y+x$;

д) все рациональные числа действительные;

е) ни одно рациональное число не является действительным;

ж) некоторые рациональные числа действительные;

з) некоторые рациональные числа не являются действительными.

3. Введем следующие обозначения:

$Z(x,t)$: я вижу предмет $x$ в момент времени $t$,

$P(x,t)$: я беру предмет $x$ в момент времени $t$,

$Q(t^*,t)$: момент времени $t^*$ предшествует моменту $t$ ($t^*<t$).

Напишите, используя эти обозначения, символические выражения для следующих предложений.

1). Я всегда что-то вижу.

2). Иногда я ничего не вижу.

3). Существуют предметы, которые я никогда не вижу.

4). Я вижу каждую вещь в некоторый момент времени.

5). Если я вижу предмет, то я тут же его беру.

6). Если я вижу предмет, то я беру его спустя некоторое время.

7). Перед тем, как я беру предмет, я вижу его.

8). Если я беру предмет, не видя его до этого, то через некоторое время я вижу его, но не беру.

9). Не существует предметов, которые я никогда не беру.

10). Я никогда не беру того, что я всегда вижу.

11). Всегда существуют вещи, которые я не вижу и не беру.

12). Я беру всякую вещь, которую я никогда не вижу.

13). Я беру всякий предмет, который я еще не взял до этого.

14). Я всегда вижу либо все, либо ничего.

15). Некоторые вещи, которые я видел ранее, я всегда вижу вновь спустя определенное время.

16). Если я когда-либо видел две вещи одновременно, то в будущем я тоже увижу их одновременно.

4. Пусть переменные в нижеследующих выражениях пробегают множество действительных чисел, а алгебраические знаки имеют свои обычные значения, прочтите эти выражения и определите, истинны ли они:

1) $\forall x\,\forall y(x+y=y+x)$;
2) $\forall x\exists y(x+y=3)$;
3) $\exists y\,\forall x(x+y=3)$;
4) $\exists x\exists y(x+y=3)$;
5) $\forall x\,\forall y(x+y=3)$;
6) $(\forall x\,\forall y(x+y=3))\Rightarrow(2=3)$;
7) $\exists x\exists y((x>y>0)\&(x+y=0))$;
8) $\forall x((x^2>x)\equiv((x>1)\vee(x<0)))$;
9) $\forall x(((x>2)\&\neg(x>3))\equiv(2<x\leq 3))$;
10) $\forall x(((x>2)\&(x<1))\equiv(x\neq x))$;
11) $\forall x(((x>1)\vee(x<2))\equiv(x=x))$.

5. Рассмотрите предложения, получающиеся в результате приписывания к предикату $x=y$, определенному на множестве всех действительных чисел, всевозможных комбинаций кванторов существования и всеобщности. Какие из полученных предложений истинны?

6. Пусть $P(x)$ обозначает: $x$ – смертен. Сформулируйте словесно следующие выражения:

а) $\forall xP(x)$; б) $\exists xP(x)$;
в) $\neg\forall xP(x)$; г) $\forall x\neg P(x)$;
д) $\exists x\neg P(x)$; е) $\neg\exists xP(x)$;
ж) $(\exists xP(x))\&\exists y\neg P(y)$; з) $(\forall x\neg P(x))\Rightarrow\neg\exists xP(x)$.

7. Для действительных чисел запишите символически, т.е. используя обозначения $\forall x$, $\exists x$, $x=y$ и т.п., предложения, выражающие:

а) коммутативность сложения;
б) коммутативность умножения;
в) ассоциативность сложения;
г) ассоциативность умножения;
д) дистрибутивность умножения относительно сложения.

8. Выразить область истинности предиката $P(x,y)$ через область истинности предикатов $A(x,y)$ и $B(x,y)$, если:

а) $P(x,y)=\neg A(x,y)$; б) $P(x,y)=A(x,y)\&B(x,y)$;
в) $P(x,y)=A(x,y)\vee B(x,y)$; г) $P(x,y)=A(x,y)\Rightarrow B(x,y)$;

д) $P(x,y)=A(x,y)\equiv B(x,y)$.

9. Пусть $M$ – множество действительных чисел, а $A(x)$ обозначает, что $x$ обладает некоторым свойством $A$. Запишите символически следующие предложения:

1) существует хотя бы одно $x\in M$ такое, что $A(x)$;
2) существует ровно одно $x\in M$ такое, что $A(x)$;
3) существует не более одного $x\in M$ такого, что $A(x)$;
4) существует в точности два $x\in M$ таких, что $A(x)$;
5) существует не менее двух $x\in M$ таких, что $A(x)$;
6) существует не более двух $x\in M$ таких, что $A(x)$.

10. Пусть $A(x,y)$ двухместный предикат на множестве всех вещественных чисел. Через $M_A$ обозначим область истинности предиката $A(x,y)$, т.е. множество тех точек $(x,y)$ плоскости $R^2$, для которых $A(x,y)=И$. Рассмотрите предикаты $\exists xA(x,y)$ и $\exists x\neg A(x,y)$ и выясните, как связаны области истинности этих предикатов с множеством $M_A$.

11. Запишите символически следующие предложения:

а) $A(x)$ при произвольном $x$; б) $A(x)$ каково бы ни было $x$;
в) всегда имеет место $A(x)$; г) найдется $x$, для которого $A(x)$;
д) не при всяком $x$ $A(x)$; е) $A(x)$ не для всех $x$;
ж) для всякого $x$ не $A(x)$; з) нет $x$ такого, что $A(x)$;
и) нет никакого $x$, такого, что $A(x)$; к) для некоторого $x$ не $A(x)$;
л) ни для какого $x$ не верно $A(x)$.

12. Составьте списки предложений, которые могут быть заменены символами:

а) $\forall xA(x)$; б) $\exists xA(x)$;
в) $\neg\forall xA(x)$; г) $\forall x\neg A(x)$;
д) $\exists x\neg A(x)$; е) $\neg\exists xA(x)$.

*Формулы логики предикатов. Свободные и связанные переменные*

13. Выяснить, какие из следующих выражений являются формулами логики предикатов:

а) $\neg f_1^1(a)$; б) $f_1^1(A_1^1(x))$;
в) $A_1^2(f_1^1(x),A_1^1(y))$; г) $\forall x\, f_1^1(x)\Rightarrow A_1^1(b)$;
д) $(\forall x\, A_1^3(a,b,c))$; е) $\forall a\, A_1^1(a)$;
ж) $(\forall x\, A_1^2(a))$; з) $f_1^1(x)\Rightarrow f_1^1(a)$.

14. Выяснить, какие из следующих выражений являются формулами логики предикатов:

1) $\forall x\,\forall x\,\left(A_1^1(x)\Rightarrow\forall yA_1^2(x,y)\right)$;
2) $\forall x\,\forall y\,\left(A_1^1(x)\Rightarrow\forall yA_1^2(x,y)\right)$;
3) $((\exists x(\forall y(A_1^1(x))))\Rightarrow(\exists x(\forall x(A_2^1(y)))))$;
4) $(\exists x(\forall x(\exists x(A_3^1(y)))))$;

5) $A_1^1(x) \Rightarrow \neg\forall x$;

6) $\forall x \Rightarrow \forall y A_3^1(x)$;

7) $\neg\exists x A_1^1(x) \vee A_2^1(y)$;

8) $\neg A_1^1(y) \& A_3^3(x,y,a)$;

9) $(A_1^2(f_1^1(a),a) \Rightarrow A_1^3(x,a,b))$;

10) $(A_1^2(\forall x,y) \Rightarrow A_1^1(f_1^1(y)))$.

15. Восстановить скобки и указать свободные и связанные переменные:

а) $\forall x \neg A(x) \Rightarrow B(x,y,z) \vee \forall x C(x)$;

б) $\exists x \forall y \exists z A(x) \vee \exists y \neg\forall z \neg A(z,y)$;

в) $\neg\forall x A(x) \Rightarrow \exists y B(y) \Rightarrow A(x,y) \vee A(y)$;

г) $\forall x \forall z \forall y A(x) \Rightarrow B(z) \& \neg A(x)$;

д) $\forall z \forall x A(x,y) \Rightarrow \exists y A(z,x)$;

е) $\forall y A(x,y) \Rightarrow \forall z A(z,y)$.

16. Указать свободные и связанные переменные:

а) $\exists x A(x) \& B(x)$; б) $P(x) \Rightarrow \exists x Q(x)$;

в) $\exists x \forall y P(x) \& Q(y) \Rightarrow \forall x R(x)$; г) $\exists x \exists y P(x,y) \& Q(z)$;

д) $\forall z P(z) \& \exists x Q(x,z) \Rightarrow \exists y R(z,y) \vee Q(z,x)$; е) $\exists y P(x,y) \& \forall x(x,z) \Rightarrow P(y,y)$.

17. Каждый из следующих предикатов, определенных на множестве всех действительных чисел, превратите в высказывание:

1) путем подстановки вместо свободной переменной какого-либо ее значения;

2) связывая свободную переменную каким-либо квантором.

Определите истинностное значение полученных высказываний.

а) $\forall x(x+y=y+x)$; б) $x>y$;

в) $x \cdot y=5$; г) $\exists x(x \cdot y=5)$;

д) $x^2+y^2 \geq 0$; е) $\forall x \exists y(x+y=z)$;

ж) $\forall x(\sin x>1) \Rightarrow (x^2<0)$; з) $\exists x(x>y)$.

18. Пусть все приведенные ниже предикаты определены на множестве всех действительных чисел. Изобразить графически области изменения свободных переменных, при которых следующие предикаты принимают значение *И*:

а) $y \geq x^2$; б) $\forall x(y<\sin x)$;

в) $\exists x(x^2+y^2 \leq 4)$; г) $\forall y \exists y(\sin x=1)$;

д) $\forall x(x^2+4y^2 \leq 4)$; е) $\forall y(x+\sin y \leq 2)$;

*Выполнимые, ложные и истинные в данной интерпретации формулы*

19. 1). Истинна, ложна или выполнима формула $A(f_1(f_2(x,y)),y)$ в каждой из следующих интерпретаций:

а) $M=(-\infty,\infty)$, $A(x,y)$: $x=y$, $f_2(x,y)$: $x+y$, $f_1(z)$: $\ln z$;

б) $M=(0,2\pi]$, $A(x,y)$: $x=-y$, $f_2(x,y)$: $xy$, $f_1(x)$: $x^2$;

в) $M=(0,1]$, $A(x,y)$: $x>-y$, $f_2(x,y)$: $1/x+y$, $f_1(x)$: $x^3$;

г) $M=[0,2\pi]$, $A(x,y)$: $x^2=-y$, $f_2(x,y)$: $x+y$, $f_1(x)$: $\sin x$.

2). Истинна, ложна или выполнима формула $\forall x A(f_1(x,y),f_2(f_1(x,y)))$ в интерпретации: $M=(-\infty,\infty)$; $f_2(z)$: $z^2$; $f_1(x,y)$: $x+y$; $A(x,y)$: $x=y$.

20. Предикат $A(x,y)$ задан на множестве $M=\{1,2,3\}$ таблицей

| $x \setminus y$ | 1 | 2 | 3 |
|---|---|---|---|
| 1 | И | И | И |
| 2 | Л | Л | И |
| 3 | И | Л | И |

Определить истинностное значение приведенных ниже формул при каждом значении свободной переменной:

а) $\forall x A(x,y)$; б) $\exists x A(x,y)$;

в) $\forall y A(x,y)$; г) $\exists y A(x,y)$.

21. Пусть $M=\{1,2,3\}$ и на этом множестве $M$ заданы предикаты $A(x,y)$ и $B(x)$ таблицами:

$A(x,y)$:

| $x \setminus y$ | 1 | 2 | 3 |
|---|---|---|---|
| 1 | Л | И | Л |
| 2 | И | Л | И |
| 3 | Л | И | Л |

$B(x)$:

| $x$ | $B(x)$ |
|---|---|
| 1 | Л |
| 2 | И |
| 3 | Л |

Определить истинностное значение формул:

а) $\exists x A(x,x)$; б) $\forall x A(x,x) \Rightarrow \exists x \forall y A(x,y)$;

в) $\exists x \exists y (B(x) \& A(x,y))$; г) $(\exists x B(x)) \& \forall x \neg A(y,y)$;

д) $\exists x \forall y (B(y) \Rightarrow A(x,y))$.

22. Пусть формула $\boldsymbol{B}$ не содержит свободных переменных, а $P(x)$ – одноместный предикат. Для области $M$, состоящей из двух элементов, построить таблицы истинностных значений формул:

а) $\forall x \boldsymbol{B}$; б) $\exists x \boldsymbol{B}$;

в) $\forall x P(x)$; г) $\exists x P(x)$;

д) $\forall x (P(x) \Rightarrow \boldsymbol{B})$; е) $\exists x (P(x) \Rightarrow \boldsymbol{B})$;

ж) $\forall x (P(x) \Rightarrow \boldsymbol{B}) \equiv \exists x P(x) \Rightarrow \boldsymbol{B}$; з) $\exists x (P(x) \Rightarrow \boldsymbol{B}) \equiv \forall x P(x) \Rightarrow \boldsymbol{B}$.

23. Предикат $P(x,y)$ задан на множестве целых положительных чисел бесконечной таблицей, в которой значения $И$ стоят в первой строке,

| $x/y$ | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | И | И | И | И | И |
| 2 | И | И | Л | Л | Л |
| 3 | И | Л | И | Л | Л |
| 4 | И | Л | Л | И | Л |
| 5 | И | Л | Л | Л | И |

первом столбце и по диагонали. Выяснить, при каких значениях $x$, $y$, $z$ следующие формулы принимают значения $И$:

а) $P(x,4)$; б) $P(x,2)\&P(z,5)$;

в) $P(x,y)\&P(z,5)$; г) $P(x,y)\&P(x,z)$.

24. Пусть предикат $P(x,y)$ тот же, что и в предыдущей задаче. Выяснить, принимают ли значение *И* следующие формулы:

а) $\exists x\,\forall y P(x,y)$; б) $\exists x P(x,x)$;

в) $\forall x \exists y P(x,y)$; г) $\forall x\,\forall y P(x,y)$;

д) $\forall x\,\forall y P(x,y) \Rightarrow \forall y P(5,y)$; е) $\exists x(\forall y P(y,y) \Rightarrow P(x,x))$;

ж) $\forall x P(x,2)$; з) $\exists y P(3,y)$.

25. Истинна ли формула $\exists x A_1^1(x) \Rightarrow \forall y A_1^1(y)$:

а) для произвольной одноэлементной области; б) для произвольной двухэлементной области.

26. Перед следующими предикатами, определенными на множестве всех действительных чисел, поставьте соответствующие кванторы так, чтобы получить истинное высказывание;

а) $x^3=27$; б) $x+1\geq 1$;

в) $x\cdot y=4$; г) $x-y\neq 0$;

д) $x+y=z$; е) $(x>y\geq 0)\Rightarrow(x>0)$.

27. Выяснить, выполнима ли формула $\forall x \exists y P(x,y,z)$ в интерпретации:

$M=(-\infty,\infty)$; $P(x,y,z)$: $x+y<z$. Является ли эта формула истинной для данной интерпретации?

28. Для формулы $\forall x P(x,y) \Rightarrow P(y,y)$ найдите интерпретацию, в которой эта формула выполнима.

29. Истинна ли формула $\forall x P(x,y) \Rightarrow P(y,y)$ на произвольной двухэлементной области.

*Выполнимые формулы. Логически общезначимые формулы. Равносильные формулы*

30. Показать, что формула $\forall x \exists y A(x,y) \Rightarrow \forall y \exists x A(x,y)$ не является логически общезначимой. Выполнима ли эта формула?

31. Является ли выполнимой формула

$$\forall y\,\forall x(P(x,y,z) \Rightarrow P(y,x,z)).$$

32. Доказать, что, если формула логики предикатов $\boldsymbol{A}$, содержащая свободно только переменную $x$, является логически общезначимой, то формула $\forall x\boldsymbol{A}$ также является логически общезначимой, и обратно.

Обобщить сформулированное утверждение на формулы, содержащие любое конечное число свободных переменных.

33. Если формула логики предикатов $\boldsymbol{A}$, содержащая только свободную переменную $x$, является логически общезначимой, то $\exists x\boldsymbol{A}$ также является логически общезначимой. Верно ли обратное?

34. Доказать, что

а) если формула логики предикатов $\boldsymbol{A} \Rightarrow \boldsymbol{B}$ является логически общезначимой, то формулы $\forall x\boldsymbol{A} \Rightarrow \forall x\boldsymbol{B}$ и $\exists x\boldsymbol{A} \Rightarrow \exists x\boldsymbol{B}$ также являются логически общезначимыми;

б) если формула логики предикатов $\boldsymbol{A \equiv B}$ является логически общезначимой, то формулы $\forall x\boldsymbol{A} \equiv \forall x\boldsymbol{B}$ и $\exists x\boldsymbol{A} \equiv \exists x\boldsymbol{B}$ также являются логически общезначимыми.

35. Показать, что формула $\forall x\, \forall y(P(x) \vee \neg P(y))$ является истинной для одноэлементной области и только для нее, здесь $P$ является одноместной предикатной буквой.

36. Докажите, что формула $\exists x\, \forall y\boldsymbol{A} \Rightarrow \forall y\, \exists x\boldsymbol{A}$ является логически общезначимой.

37. Является ли выполнимой формула $\exists x\, \forall yP(x,y) \Rightarrow \forall y\, \exists xP(f(x),y)$? Будет ли эта формула логически общезначимой?

38. 1). Являются ли истинными или выполнимыми для произвольной двухэлементной области следующие формулы ($\boldsymbol{A}$ и $\boldsymbol{B}$ не содержат свободных переменных):

а) $\exists x(\boldsymbol{A} \Rightarrow \boldsymbol{B}(x)) \equiv (\boldsymbol{A} \Rightarrow \forall x\boldsymbol{B}(x))$; б) $\exists x(\boldsymbol{A}(x) \Rightarrow \boldsymbol{B}) \equiv (\forall x\boldsymbol{A}(x) \Rightarrow \boldsymbol{B})$;

в) $\forall x(\boldsymbol{A}(x) \Rightarrow \boldsymbol{B}) \equiv (\exists x\boldsymbol{A}(x) \Rightarrow \boldsymbol{B})$; г) $\forall x(\boldsymbol{A} \Rightarrow \boldsymbol{B}(x)) \equiv (\boldsymbol{A} \Rightarrow \forall x\boldsymbol{B}(x))$.

2). Являются ли формулы а) – г) логически общезначимыми или нет?

39. Являются ли выполнимыми следующие формулы:

а) $\forall x\, \forall y\, \forall z(\boldsymbol{A}(x,x) \& (\boldsymbol{A}(x,z) \Rightarrow \boldsymbol{A}(x,y) \vee \boldsymbol{A}(y,z))) \Rightarrow \exists x\, \forall y\boldsymbol{A}(x,y)$;

б) $\forall x\, \exists y\, \forall z(\boldsymbol{A}(x,x) \& \neg \boldsymbol{A}(y,x) \& (\boldsymbol{A}(y,z)) \Rightarrow \boldsymbol{A}(x,y)))$.

40. Пусть $\boldsymbol{A}$ не содержит свободных переменных, $P$, $Q$ – одноместные предикатные буквы. Выяснить, являются ли логически общезначимыми следующие формулы:

1) $\boldsymbol{A} \& \forall xQ(x) \equiv \forall x(\boldsymbol{A} \& Q(x))$;

2) $\boldsymbol{A} \& \exists xQ(x) \equiv \exists x(\boldsymbol{A} \& Q(x))$;

3) $\forall xP(x) \& \forall xQ(x) \equiv \forall x(P(x) \& Q(x))$;

4) $\forall xP(x) \vee \forall xQ(x) \equiv \forall x(P(x) \vee Q(x))$;

5) $\forall xP(x) \vee \forall xQ(x) \Rightarrow \forall x(P(x) \vee Q(x))$;

6) $\exists xP(x) \& \exists xQ(x) \equiv \exists x(P(x) \& Q(x))$;

7) $\exists xP(x) \vee \exists xQ(x) \equiv \exists x(P(x) \vee Q(x))$;

8) $\exists x(P(x) \& Q(x)) \Rightarrow \exists\, xP(x) \& \exists xQ(x)$.

41. Пусть $\boldsymbol{A}$ не содержит свободных переменных $P$, $Q$ - одноместные предикатные буквы. Какие из следующих формул являются логически общезначимыми:

1) $\forall x(P(x) \Rightarrow \boldsymbol{A}) \equiv (\exists xP(x) \Rightarrow \boldsymbol{A})$; 2) $\exists x(P(x) \Rightarrow \boldsymbol{A}) \equiv (\exists xP(x) \Rightarrow \boldsymbol{A})$;

3) $\exists x(P(x) \Rightarrow \boldsymbol{A}) \equiv (\forall xP(x) \Rightarrow \boldsymbol{A})$; 4) $\exists x(\boldsymbol{A} \Rightarrow P(x)) \equiv (\boldsymbol{A} \Rightarrow \exists xP(x))$;

42. Какие из приведенных ниже формул являются выполнимыми, а какие из них логически общезначимыми ($P$, $Q$ – одноместные предикатные буквы):

1) $\exists x(P(x) \Rightarrow Q(x)) \equiv \forall xP(x) \Rightarrow \exists xQ(x)$;

2) $\exists x(P(x) \Rightarrow Q(x)) \equiv \forall x(P(x) \& \neg \exists xQ(x))$;

3) $(\forall xP(x) \Rightarrow \forall xQ(x)) \Rightarrow \forall x(P(x) \Rightarrow Q(x))$;

4) $\forall x(P(x) \vee Q(x)) \Rightarrow (\forall xP(x)) \vee \forall xQ(x))$.

43. Выяснить, являются ли равносильными следующие пары формул ($P$, $Q$ – одноместные предикатные буквы, $\boldsymbol{A}$ – произвольная формула, имеющая указанные аргументы):

а) $\forall xP(x) \Rightarrow \forall xQ(x)$ и $\forall xP(x) \Rightarrow \forall yQ(y)$;

б) $\forall x\boldsymbol{A}(x,y)$ и $\forall x\boldsymbol{A}(a,y)$;

в) $\neg\forall x(\neg P(x) \Rightarrow Q(x))$ и $\neg\forall x(\neg P(x) \Rightarrow Q(y))$;

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

д) $\forall xP(x) \& Q(x)$ и $\forall xP(x)) \& Q(x)$;

е) $\boldsymbol{A}(x,a)$ и $\boldsymbol{A}(y,a)$.

44. 1). Для следующих формул найти равносильные формулы, не содержащие кванторов вне скобок (внести кванторы под скобки):

а) $\exists x\, \forall y(\boldsymbol{A}(x) \Rightarrow \boldsymbol{B}(x,y))$; б) $\forall x\, \forall y(\neg\boldsymbol{A}(x) \Rightarrow \boldsymbol{B}(y))$;

в) $\exists x \exists y(\neg\boldsymbol{A}(x) \Rightarrow \boldsymbol{B}(y))$; г) $\forall x \exists y(\boldsymbol{B}(x,y) \Rightarrow \boldsymbol{A}(x))$.

2). Для следующих формул найти равносильные формулы, в которых $\neg$ относится только к элементарным формулам:

а) $\forall x\, \neg(\exists y(\boldsymbol{A}(x) \Rightarrow \boldsymbol{B}(y)))$;

б) $\neg\exists x(\forall y\boldsymbol{A}(x,y,z) \Rightarrow \exists u\boldsymbol{B}(x,u)) \& \forall t\, \neg\forall v(\boldsymbol{C}(t) \vee \boldsymbol{D}(v))$;

в) $\neg\forall x \exists z(\boldsymbol{A}(x) \Rightarrow \neg\boldsymbol{B}(z))$;

г) $\exists y\, \neg\forall x(\boldsymbol{A}(x) \& \boldsymbol{B}(y)) \Rightarrow \neg\exists z\boldsymbol{C}(x,y,z)$.

45. Пусть $A(x,y)$ двухместный предикат на множестве всех вещественных чисел. Через $M_A$ обозначим область истинности предиката $A(x,y)$, т.е. множество тех точек $(x,y)$ плоскости $R^2$, для которых $A(x,y)=И$. Рассмотрите предикаты $\forall xA(x,y)$ и $\forall x\, \neg A(x,y)$ и выясните, как связаны области истинности этих предикатов с множеством $M_A$

Формула $Q_1x_1Q_2x_2\ldots Q_nx_n\boldsymbol{B}$, где $Q_ix_i$ квантор всеобщности или существования, $x_i$ и $x_j$ различны, если $i \neq j$ и $\boldsymbol{B}$ не содержат кванторов, называется формулой в предваренной нормальной форме (иногда – пренексной нормальной формой). Сюда относится и случай $n=0$, когда вообще нет кванторов.

46. 1). Привести к предваренным нормальным формам формулы из предыдущей задачи.

2). Привести к предваренным нормальным формам следующие формулы:

а) $\forall x(\boldsymbol{A}(x) \Rightarrow \boldsymbol{B}(x,y)) \Rightarrow ((\exists y\boldsymbol{A}(y)) \Rightarrow \exists z\boldsymbol{B}(y,z))$;

б) $\exists x\boldsymbol{B}(x,y) \Rightarrow (\boldsymbol{A}(x) \Rightarrow \neg\exists z\boldsymbol{B}(x,z))$;

в) $\forall x(\forall y \exists z\boldsymbol{C}(x,y,z) \Rightarrow \boldsymbol{A}(x)) \Rightarrow \forall x\boldsymbol{A}(x)$;

г) $\neg\exists x\boldsymbol{A}(x) \Rightarrow \forall z \exists y\, \forall x\boldsymbol{C}(x,y,z)$.

47. Привести к предваренным нормальным формам следующие формулы:

а) $\exists x\boldsymbol{A}(x) \Rightarrow \forall x\boldsymbol{B}(x)$;

б) $\forall x\boldsymbol{A}(x) \Rightarrow \exists x\boldsymbol{B}(x)$;

в) $\forall x\, \forall y(\exists z(\boldsymbol{A}(x,y) \& \boldsymbol{B}(y,z)) \Rightarrow \exists v\boldsymbol{C}(x,y,v))$;

г) $\forall x(\boldsymbol{A}(x) \Rightarrow \exists y\boldsymbol{B}(x,y))$;

д) $\exists x(\boldsymbol{A}(x) \Rightarrow \forall y\boldsymbol{B}(x,y))$;

е) $\exists x(\neg(\exists y\boldsymbol{A}(x,y)) \Rightarrow(\exists z\boldsymbol{B}(z) \Rightarrow\boldsymbol{C}(x)))$;

ж) $\forall x\exists y(\forall z\boldsymbol{A}(x,y,z)\&(\exists u\boldsymbol{B}(x,u) \Rightarrow\exists v\boldsymbol{C}(y,v)))$.

*Знание действия зависит от знания причины и заключает в себе последнее.*
*Б. Спиноза*

# Глава 3. ЛОГИЧЕСКОЕ СЛЕДСТВИЕ И МЕТОД РЕЗОЛЮЦИЙ

*Врубайся в суть, не боясь затупить клинок,*
*Ибо острота не вечна.*
*Лао Цзы (Дао Дэ Цзин)*

*Это конечно, Сова.*
*Или я не Винни-Пух.*
*А я – он…*
*А. М. Мили*

## § 1. Логическое следствие и проблема дедукции в логике высказываний

Пусть $A$ и $B$ пропозициональные формы (формулы логики высказываний). Считаем, что $B$ *логически следует* из $A$, если для каждой совокупности значений пропозициональных букв, при которых $A=И$ форма $B$ тоже принимает значение $И$. В этом случае записываем

$$A \models B$$

и читаем: «из $A$ логически следует $B$» или «$B$ является логическим следствием из $A$».

Легко доказать следующую теорему.

**Теорема 3.1**. Если $A \models B$ и $B \models C$, то $A \models C$.

Запись $\models A$ в логике высказываний означает, что $A$ является тавтологией (общезначимой). Докажем следующую теорему.

**Теорема 3.2**. $A \models B$ тогда и только тогда, когда $\models A \Rightarrow B$.

**Доказательство**. Построим таблицу истинности для $A \Rightarrow B$. Пусть имеем, что $A \models B$, тогда в каждой строке таблице, где $A=И$ будет $B=И$, следовательно, $A \Rightarrow B$ тавтология, то есть $\models A \Rightarrow B$. Если же имеем $\models A \Rightarrow B$, то в

таблице истинности для $A \Rightarrow B$ всюду, где $A=И$ должно быть $B=И$, следовательно получим $A \models B$.

Пропозициональная форма $B$ называется *логическим следствием* пропозициональных форм $A_1,A_2,\ldots,A_m$, $m\geq 1$, если для каждой совокупности значений пропозициональных букв при которых формы $A_1, A_2,\ldots,A_m$ одновременно равны *И* форма $B$ тоже принимает значение *И*. В этом случае записываем:

$$A_1, A_2,\ldots,A_m \models B \qquad (3.1)$$

Выяснение будет ли $B$ логическим следствием из $A_1, A_2,\ldots,A_m$, $m\geq 1$, называют *проблемой дедукции*.

Очевидно, что имеет место следующая теорема.

**Теорема 3.3**. Для произвольных формул логики высказываний $A_1,A_2,\ldots,A_m$, $m\geq 1$, имеют место соотношения:

$$A_1,A_2,\ldots,A_m \models A_1\&A_2\&\ldots\&A_m\,, \qquad (3.2)$$

$$A_1,A_2,\ldots,A_m \models A_i \text{ для любого } i,\ 1\leq i\leq m. \qquad (3.3)$$

**Теорема 3.4**. Если формула

$$A_1\&A_2\&\ldots\&A_m\&\neg B \qquad (3.4)$$

является противоречием, тогда $B$ является логическим следствием из $A_1,A_2,\ldots,A_m$, т.е.:

$$A_1,A_2,\ldots,A_m \models B. \qquad (3.5)$$

**Доказательство**. Пусть формула (3.4) является противоречием, тогда ее отрицание является тавтологией, т.е. имеем:

$$\models \neg (A_1\&A_2\&\ldots\&A_m\&\neg B). \qquad (3.6)$$

Очевидно, что имеем

$$\neg(A_1\&A_2\&\ldots\&A_m\&\neg B) \sim \neg(A_1\&A_2\&\ldots\&A_m)\vee B \sim$$
$$\sim (A_1\&A_2\&\ldots\&A_m)\Rightarrow B.$$

Следовательно, утверждение (3.6) можно записать в виде:

$$\models (A_1\&A_2\&\ldots\&A_m)\Rightarrow B. \qquad (3.7)$$

Из (3.7) по теореме 3.2 получаем, что

$$(A_1\&A_2\&\ldots\&A_m) \models B. \qquad (3.8)$$

Теперь используя утверждение (3.2) теоремы 3.3 и утверждение (3.8) получим требуемое утверждение (3.5). Теорема доказана.

Используя утверждения (3.2) и (3.3) теоремы 3.3, можно получать следствия из заданного множества формул следующим образом. Для заданного множества формул $\boldsymbol{A_1, A_2,\ldots,A_m}$, $m\geq 1$, строим их конъюнкцию: $\boldsymbol{C=A_1\&A_2\&\ldots\&A_m}$. Для $\boldsymbol{C}$ находим с.к.н.ф.: $\boldsymbol{C= D_1\&D_2\&\ldots\&D_k}$, здесь $\boldsymbol{D_i}$, $1\leq i\leq k$, элементарные суммы (дизъюнкты). Теперь по указанной теореме 3.3 получаем, что каждый дизъюнкт $\boldsymbol{D_i}$, $1\leq i\leq k$, а также их конъюнкции являются следствиями из $\boldsymbol{A_1, A_2,\ldots,A_m}$, $m\geq 1$, т. е. имеем:

$\boldsymbol{A_1, A_2,\ldots,A_m \models D_i}$ для любого $i$, $1\leq i\leq k$;

$\boldsymbol{A_1, A_2,\ldots,A_m} \models D_{s_1} \& D_{s_2} \& \ldots \& D_{s_r}$, для любого $r$, $1\leq r\leq k$ и любых $s_1, s_2,\ldots,s_r$, $1\leq s_1, s_2,\ldots,s_r \leq k$.

Заметим, что для формул логики предикатов понятие логического следствия из данной формулы (данных формул) введено в 6 – ом параграфе второй главы. Нетрудно убедиться, что теоремы 3.1 - 3.4 остаются в силе и для формул логики предикатов, в частности, теорема 3.2 для формул логики предикатов уже доказана, см. теорему 2.1.

*Хочешь взять-*
*Умей отдать.*
*Вот в чем глубокая истина.*
*Лао Цзы*

## § 2. Резольвента дизъюнктов логики высказываний

Пропозициональные буквы с отрицанием либо без отрицания, входящую в элементарную сумму (дизъюнкт), называют *литералами (литерами)* логики высказываний.

Литеры $L$ и $\neg L$ называются *контрарными*. Так, например, в дизъюнктах $\boldsymbol{D_1}=P\vee\neg Q$ и $\boldsymbol{D_2}=\neg P\vee Q\vee S$ литеры $P$ и $\neg P$ контрарные. Также контрарные литеры $Q$ и $\neg Q$.

Пусть для двух дизъюнктов $\boldsymbol{D_1}$ и $\boldsymbol{D_2}$ существует литера $L_1$ в $\boldsymbol{D_1}$, которая контрарна литере $L_2$ в $\boldsymbol{D_2}$. Вычеркнув $L_1$ и $L_2$ из $\boldsymbol{D_1}$ и $\boldsymbol{D_2}$ соответственно, построим дизъюнкцию оставшихся частей $\boldsymbol{D_1}$ и $\boldsymbol{D_2}$. Полученный таким образом дизъюнкт называется *(бинарной) резольвентой* $\boldsymbol{D_1}$ и $\boldsymbol{D_2}$, который часто обозначают через $\boldsymbol{R}$.

Примеры. 1. Пусть $\boldsymbol{D_1}=P\vee Q$, $\boldsymbol{D_2}=\neg P\vee T$, тогда $\boldsymbol{R}=Q\vee T$.

2. Пусть $\boldsymbol{D_1}=P$, $\boldsymbol{D_2}=\neg P\vee Q$ $(D_2\sim P\Rightarrow Q)$, тогда $\boldsymbol{R}=Q$, иначе из $P$ и $P\Rightarrow Q$ получаем $Q$.

3. Пусть $\boldsymbol{D_1}=\neg P\vee Q$ $(\boldsymbol{D_1}=P\Rightarrow Q)$, $\boldsymbol{D_2}=\neg Q\vee T$ $(\boldsymbol{D_2}=Q\Rightarrow T)$, тогда $\boldsymbol{R}=\neg P\vee T$ $(\boldsymbol{R}=P\Rightarrow T)$. Иначе из $P\Rightarrow Q$ и $Q\Rightarrow T$ получаем $P\Rightarrow T$.

**Теорема 3.5**. Пусть для дизъюнктов $D_1$ и $D_2$ существует резольвента $R$. Тогда $R$ есть логическое следствие из $D_1$ и $D_2$.

**Доказательство**. Пусть $D_1=P\vee D_1^*$, $D_2=\neg P\vee D_2^*$, где $D_i^*$ оставшаяся часть дизъюнкта $D_i$, $i=1,2$. Докажем, что $D_1,D_2\models D_1^*\vee D_2^*$. Выпишем всевозможные наборы истинностных значений букв входящих в $D_1$ и $D_2$. Выберем набор, положим $k$-ый, на котором $D_1=И$ и $D_2=И$. Допустим, что на этом $k$-ом наборе буква $P$ принимает значение $И$, тогда $\neg P=Л$, поэтому должно быть $D_2^*=И$, следовательно $D_1^*\vee D_2^*=И$. Таким образом из истинности $D_1$ и $D_2$ получили истинность $D_1^*\vee D_2^*$. В случае если на $k$-ом наборе $P=Л$, то $D_1^*=И$ и вновь получаем, что из истинности $D_1$ и $D_2$ следует истинность $D_1^*\vee D_2^*$. В силу произвольности выбранного набора получаем, что из истинности $D_1$ и $D_2$ следует истинность для $D_1^*\vee D_2^*$, что и требовалось.

*Следует поставить перед собой цель изыскать способ решения всех задач ... одним и притом простым способом.*

*Даламбер*

## § 3. Метод резолюций в логике высказываний

Рассмотрим задачу выяснения будет ли $B$ логическим следствием из $A_1,A_2,\ldots,A_m$, то есть истинна ли следующая запись:

$A_1,A_2,\ldots,A_m\models B$.

В § 1 данной главы показано, что эта задача сводится к выяснению невыполнимости формы

$C=A_1\&A_2\&\ldots\&A_m\&\neg B$.

Найдем для формулы $C$ ее к.н.ф., то есть получим конъюнкцию дизъюнктов: $C=D_1\&D_2\&\ldots\&D_k$.

Каждое слагаемое дизъюнкта является литералом.

Множество дизъюнктов $\{D_1,D_2,\ldots,D_k\}$ считается *невыполнимым* тогда и только тогда, когда формула $C$ невыполнима.

*Методом резолюций* называется последовательное получение бинарных резольвент из данных дизъюнктов и вновь получаемых дизъюнктов. Пусть, например, даны дизъюнкты

$D_1=P\vee T$, $D_2=\neg P\vee T$, $D_3=\neg T$.

Используя $D_1$ и $D_2$ затем $D_1$ и $D_3$ , получим резольвенты

$D_4=T$, $D_5=P$.

Затем из $D_3$ и $D_4$ получим *пустой дизъюнкт*. Пустой дизъюнкт будем обозначать через $\square$.

Можно доказать следующую теорему.

**Теорема 3.6** (полнота метода резолюций). Множество $S$ дизъюнктов невыполнимо тогда и только тогда, когда существует вывод пустого дизъюнкта $\square$ из $S$ (имеется в виду, что выводом является применение метода резолюций).

Существует много различных подходов к построению вывода с помощью метода резолюций. Рассмотрим некоторые из них: метод насыщения уровня, стратегию вычеркивания, лок-резолюцию и один метод для дизъюнктов специального вида.

*Коль с головой ты в воду погрузился,*
*Не все ли равно, какая глубина?*
*Дикаики (Услада Души, или Бахтияр-наме)*

## § 4. Метод насыщения уровня

Ранее был введен метод резолюций и приведена теорема, утверждающая полноту метода резолюций. Пусть имеем множество дизъюнктов $S=\{\boldsymbol{D_1},\boldsymbol{D_2},\ldots,\boldsymbol{D_k}\}$. Процедура получения бинарных резольвент неоднозначна, ибо можно сравнивать $\boldsymbol{D_1}$ и $\boldsymbol{D_2}$, затем $\boldsymbol{D_1}$ и $\boldsymbol{D_3}$ или как-то иначе. Рассмотрим метод насыщения уровня. Он состоит в вычислении всех резольвент всех пар дизъюнктов из $S$, добавлении этих резольвент к множеству $S$, вычислении всех следующих резольвент и повторении этого процесса, до тех пор пока не найдется пустой дизъюнкт $\square$. Это значит, мы порождаем $S^0,S^1,S^2,S^3,\ldots$, где $S^0=S$, а

$S^n=\{$резольвенты $\boldsymbol{D_j}$ и $\boldsymbol{D_k}$: $\boldsymbol{D_j}\in(S^0\cup S^1\cup\ldots\cup S^{n-1})$, $\boldsymbol{D_k}\in S^{n-1}$, $j<k\}$, $n=1,2,\ldots$

Чтобы запрограммировать этот метод на ЭВМ, мы можем сперва записать дизъюнкты $S^0\cup S^1\cup\ldots\cup S^{n-1}$ затем вычислять резольвенты, сравнивая последовательно каждый дизъюнкт $\boldsymbol{D_j}\in(S^0\cup S^1\cup\ldots\cup S^{n-1})$ с каждым дизъюнктом $\boldsymbol{D_k}\in S^{n-1}$, который расположен после $\boldsymbol{D_j}$. Когда резольвента вычислена, она присоединяется к концу списка, порожденного к этому времени. Перейдем к реализации этого процесса для случая когда $S=\{P\vee Q, \neg P\vee Q, P\vee\neg Q, \neg P\vee\neg Q\}$.

$S^0$: (1) $P\vee Q$;
(2) $\neg P\vee Q$;
(3) $P\vee\neg Q$;
(4) $\neg P\vee\neg Q$;

_____ используемые дизъюнкты

$S^1$: (5) $Q$ из (1) и (2);
(6) $P$ из (1) и (3);
(7) $Q\vee\neg Q$ из (1) и (4);
(8) $P\vee\neg P$ из (1) и (4);
(9) $Q\vee\neg Q$ из (2) и (3);

(10) $P \vee \neg P$ из (2) и (3);
(11) $\neg P$ из (2) и (4);
(12) $\neg Q$ из (3) и (4);

---

$S^2$: (13) $P \vee Q$ из (1) и (7);
(14) $P \vee Q$ из (1) и (8);
(15) $P \vee Q$ из (1) и (9);
(16) $P \vee Q$ из (1) и (10);
(17) $Q$ из (1) и (11);
(18) $P$ из (1) и (12);
(19) $Q$ из (2) и (6);
(20) $\neg P \vee Q$ из (2) и (7);

(21) $\neg P \vee Q$ из (2) и (8);
(22) $\neg P \vee Q$ из (2) и (9);
(23) $\neg P \vee Q$ из (2) и (10);
(24) $\neg P$ из (2) и (12);
(25) $P$ из (3) и (5);
(26) $P \vee \neg Q$ из (3) и (7);
(27) $P \vee \neg Q$ из (3) и (8);
(28) $P \vee \neg Q$ из (3) и (9);
(29) $P \vee \neg Q$ из (3) и (10);
(30) $\neg Q$ из (3) и (11);

(31) $\neg P$ из (4) и (5);
(32) $\neg Q$ из (4) и (6);
(33) $\neg P \vee \neg Q$ из (4) и (7);
(34) $\neg P \vee \neg Q$ из (4) и (8);
(35) $\neg P \vee \neg Q$ из (4) и (9);
(36) $\neg P \vee \neg Q$ из (4) и (10);
(37) $Q$ из (5) и (7);
(38) $Q$ из (5) и (9);
(39) $\square$ из (5) и (12).

Было порождено много не относящихся к делу и лишних дизъюнктов. Например, (7), (8), (9) и (10) – тавтологии. Так как тавтология всегда истинна, то если мы вычеркиваем тавтологию из невыполнимого множества дизъюнктов, оставшееся множество все еще должно быть невыполнимо. Следовательно, тавтология есть не относящийся к делу дизъюнкт и не должна порождаться. Если же она порождается, то (за исключением очень немногих случаев) ее следует вычеркнуть. Далее дизъюнкты $P$, $Q$, $\neg P$, $\neg Q$ порождаются неоднократно. Также имеются другие повторяющиеся дизъюнкты, см. (13) – (16), (20) – (23), (26) – (29) и (33) – (36). На самом деле, чтобы получить доказательство для $S$, нам нужно породить дизъюнкты (5),

(12) и (39). Для сокращения избыточности рассмотрим стратегию вычеркивания.

*Всякое ограничение осчастливливает.*
*А. Шопенгауэр*

## § 5. Стратегия вычеркивания

Дизъюнкт ***D*** называется *поддизъюнктом* ***D**** (или ***D*** поглощает ***D****) если ***D*** является некоторой частью дизъюнкта ***D****. При этом ***D**** называется *наддизъюнктом* для ***D***.

Пример. Пусть ***D***=*P*, ***D****=*P*∨*Q*, Ясно, что ***D*** поддизъюнкт для дизъюнкта ***D****, а ***D**** - наддизъюнкт для ***D***.

Стратегия вычеркивания зависит от того, как вычеркиваются тавтологии и наддизъюнкты. Стратегия вычеркивания будет полной, если ее использовать вместе с методом насыщения уровней следующим образом: сперва выписываются дизъюнкты $(S^0 \vee S^1 \vee \ldots \vee S^{n-1})$ по порядку; затем вычисляются резольвенты путем сравнения каждого дизъюнкта $\boldsymbol{D_i} \in (S^0 \vee S^1 \vee \ldots \vee S^{n-1})$ с дизъюнктом $\boldsymbol{D_k} \in S^{n-1}$, который стоит после $\boldsymbol{D_i}$. Когда резольвента вычислена, она записывается в конце списка, как только она порождается, если она не тавтология и не поглощается каким-нибудь дизъюнктом из списка. В противном случае она вычеркивается. Очевидно, что при этом не выписывается повторно один и тот же дизъюнкт. Применим эту стратегию вычеркивания к примеру из § 4 и получим следующий список:

$S^0$: (1) $P \vee Q$,
(2) $\neg P \vee Q$,
(3) $P \vee \neg Q$,
(4) $\neg P \vee \neg Q$,

---

$S^1$: (5) $Q$ из (1) и (2),
(6) $P$ из (1) и (3),
(7) $\neg P$ из (2) и (4),
(8) $\neg Q$ из (3) и (4),

---

$S^2$: □ из (5) и (8).

Получили существенно более короткий список. Следовательно, стратегия вычеркивания может уменьшить потребную память для реализации метода резолюций.

Ясно, что необходимые вычисления не уменьшаются, а увеличиваются. Чтобы использовать стратегию вычеркивания, мы должны уметь решать, является ли полученный дизъюнкт тавтологией или является ли один из дизъюнктов поддизъюнктом другого.

Метод резолюций, как уже указано, позволяет автоматизировать доказательство теорем. Мы видели, что неограниченное применение

резолюции может порождать много лишних и ненужных дизъюнктов наряду с полезными. Хотя можно использовать стратегию вычеркивания, чтобы выбросить некоторые из этих ненужных и бесполезных дизъюнктов после того, как они порождены, на их порождение уже потеряно время. Далее, если порождены бесполезные дизъюнкты, то нужны большие ресурсы машинного времени для того, чтобы определить, что эти дизъюнкты действительно лишние и ненужные. Поэтому для получения эффективных процедур доказательства теорем мы должны не допускать порождения большого числа бесполезных дизъюнктов. Имеются различные подходы к уменьшению вычислений, среди них: метод семантической резолюции; лок-резолюция; линейная резолюция и др. методы. Мы рассмотрим лок-резолюцию.

*Хорошее вдвойне хорошо, если кратко.*
*Бальтисар Грасман*

## § 6. Лок-резолюция

Идея лок-резолюции состоит, по существу, в использовании индексов для упорядочения литер в дизъюнктах из данного множества $S$ дизъюнктов. Иными словами она включает введение индексов для каждого вхождения литеры в $S$ некоторым целым числом; разные вхождения одной и той же литеры могут быть индексированы по-разному. После этого отрезать (удалять) разрешается только литеры с наименьшим индексом в каждом из дизъюнктов. Литеры в резольвентах наследуют свои индексы из посылок. Если литера в резольвенте может унаследовать более одного индекса, то ей ставится в соответствие наименьший индекс.

Рассмотрим следующие два дизъюнкта

$$P \vee Q, \quad \neg P \vee Q.$$

Введем индексы, которые будем писать слева снизу от литеры:

(1) ${}_1P \vee {}_2Q$,

(2) ${}_3\neg P \vee {}_4Q$.

Так как индекс *1* в ${}_1P$ меньше чем индекс 2 в ${}_2Q$, то разрешается отрезать ${}_1P$. В ${}_3\neg P \vee {}_4Q$ разрешается отрезать ${}_3\neg P$, так как 3<4. Таким образом, применяя правило резолюции к дизъюнктам (1) и (2) по ${}_1P$ и ${}_3\neg P$ мы получаем следующий дизъюнкт:

(3) ${}_2Q \vee {}_4Q$.

Литера ${}_2Q$ и ${}_4Q$ одна и та же. Так как 2<4, то $Q$ получает индекс 2, поэтому получаем

(4) ${}_2Q$.

Дизъюнкт (4) и является *лок-резольвентной* дизъюнктов (1) и (2). Отметим, что если бы литеры в дизъюнкте (2) были индексированы иначе, например, так:

(2*) ${}_4\neg P \vee {}_3Q$ ,

то литерой в дизъюнкте ($2^*$), которою разрешено отрезать, была бы $_3Q$. Однако к $_1P$ и $_3Q$ нельзя применить правило резолюции. Поэтому не существует лок-резольвенты дизъюнктов (1) и ($2^*$). Под *лок-резолюцией* понимается последовательное получение лок-резольвент из данного множества дизъюнктов и вновь получаемых дизъюнктов.

Рассмотрим множество ***S*** дизъюнктов, которое рассматривалось в §4:

$P \vee Q,$
$P \vee \neg Q,$
$\neg P \vee Q,$
$\neg P \vee \neg Q.$

Проведём следующую индексацию:

(1) $_1P \vee {}_2Q,$
(2) $_3P \vee {}_4 \neg Q,$
(3) $_6 \neg P \vee {}_5Q,$
(4) $_8 \neg P \vee {}_7 \neg Q.$

Из дизъюнктов (1) – (4) можно получить только одну лок-резольвенту

(5) $_6 \neg P$ из (3) и (4).

Из дизъюнктов (1) – (5) получаются только две лок-резольвенты:

(6) $_2Q$ из (1) и (5),
(7) $_4 \neg Q$ из (2) и (5).

Применяя правила резолюции к дизъюнктам (6) и (7), мы получаем

(8) □.

Таким образом, мы получаем вывод пустого дизъюнкта □.

Отметим, что были порождены всего три лок-реольвенты. При использовании обычной (неограниченной) резолюции для получения □ были порождены 38 резольвент. Результативность лок-резолюции не зависит от того, как проиндексировать литеры в ***S***. Введём в рассматриваемом примере индексы иначе, например, так:

(1) $_1P \vee {}_2Q,$
(2) $_3P \vee {}_4 \neg Q,$
(3) $_5 \neg P \vee {}_6Q,$
(4) $_7 \neg P \vee {}_8 \neg Q.$

Из (1) и (3) получим:

(5) $_2Q.$

Аналогично получим:

(6) $_2Q \vee {}_8 \neg Q$ из (1) и (4),
(7) $_6Q \vee {}_4 \neg Q$ из (2) и (3),
(8) $_4 \neg Q$ из (2) и (4),
(9) □ из (5) и (8).

Можно доказать следующую теорему о полноте лок-резолюции.

**Теорема 3.7**. Пусть $S$ множество дизъюнктов, в котором каждая литера индексирована целым числом. Если $S$ противоречиво (неудовлетворимо), то имеется лок-вывод пустого дизъюнкта $\square$ из $S$.

*Всё в жизни метод.*
*Андрей Белый*

## § 7. Метод резолюций для хорновских дизъюнктов

В общем случае метод резолюций требует больших вычислений. Если дизъюнкты имеют специальный вид, являются, так называемыми хорновскими дизъюнктами, то вычисления упрощаются.

Литера называется *позитивной*, если она не содержит отрицания и *негативной*, если содержит отрицание.

Дизъюнкт D называется *хорновским*, если он содержит не более одной позитивной литеры. Примеры хорновских дизъюнктов: $A$, $B$, $\neg A$, $\neg B$, $\neg A \vee \neg C \vee B$, $\neg A \vee \neg B$, $\neg A \vee \neg C \vee \neg C \vee D$. В общем случае хорновский дизъюнкт можно представить в виде $(A_1 \& A_2 \& \ldots \& A_n) \Rightarrow B$ или $\neg A_1 \vee \neg A_2 \vee \ldots \vee \neg A_n$, $n \geq 1$, или $A$. При этом дизъюнкт $\neg A_1 \vee \neg A_2 \vee \ldots \vee \neg A_n \Rightarrow B$ называют *точным*, дизъюнкт $\neg A_1 \vee \neg A_2 \vee \ldots \vee \neg A_n$ называют *негативным*, а дизъюнкт $A$ - *унитарным позитивным дизъюнктом*.

Рассмотрим множество $S$ хорновских дизъюнктов без тавтологий. Невыполнимость можно проверить с помощью следующего алгоритма.

1. Полагаем, что $S^0 = S$.
2. Пусть $S^{n-1}$, $n \geq 1$, построено. Для построения $S^n$ выбираем из $S^{n-1}$ дизъюнкты $D_1$ и $D_2$ такие, что:
   $D_1$ - унитарный позитивный дизъюнкт, пусть, например, $D_1 = P$; $D_2$ - дизъюнкт, содержащий литеру $\neg P$. Вычисляем резольвенту $R$ для дизъюнктов $D_1$ и $D_2$ и полагаем, что $S^n = (S^{n-1} \setminus \{D_2\}) \cup \{R\}$. Эту процедуру повторяем до тех пор пока не получим пустой дизъюнкт $\square$ либо пока не окажется, что в $S^{n-1}$ не существует дизъюнктов $D_1$ и $D_2$ указанных видов.

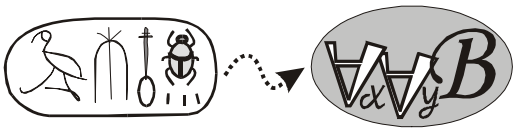
Можно доказать, что для приведенного алгоритма появление пустого дизъюнкта означает, что множество $S$ хорновских дизъюнктов невыполнимо. Если же окажется, что $S^{n-1}$ не содержит дизъюнктов $D_1$ и $D_2$ указанных видов, то исходное множество $S$ хорновских дизъюнктов выполнимо.

Реализацию этого алгоритма проще проводить с помощью таблицы. Продемонстрируем это на примере. Пусть имеем множество хорновских дизъюнктов: $S = \{P \vee \neg R \vee \neg T,\ Q,\ R,\ T \vee \neg P \vee \neg R,\ T \vee \neg Q,\ \neg P \vee \neg Q \vee \neg R\}$. Выпишем дизъюнкты из $S^0 = S$ в ячейки нулевой строки нижеследующей таблицы. Каждая $n$ - ая строка содержит дизъюнкты из $S^n$, $n \geq 0$. Дизъюнкты, из которых строятся резольвенты, помечаем снизу звёздочками.

| № итерации | $S^n$ | Дизъюнкты | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | $S^0$ | $P \vee \neg R \vee \neg T$ | $Q$ * | $R$ | $T \vee \neg P \vee \neg R$ | $T \vee \neg Q$ * | $\neg P \vee \neg Q \vee \neg R$ * |
| 1 | $S^1$ | $P \vee \neg R \vee \neg T$ * | $Q$ | $R$ * | $T \vee \neg P \vee \neg R$ * | $T$ | $\neg P \vee \neg R$ * |
| 2 | $S^2$ | $P \vee \neg T$ * | $Q$ | $R$ | $T \vee \neg P$ | $T$ * | $\neg P$ |
| 3 | $S^3$ | $P$ * | $Q$ | $R$ | $T \vee \neg P$ | $T$ | $\neg P$ * |
| 4 | $S^4$ | □ | | | | | |

На четвертом шаге получаем пустой дизъюнкт □, следовательно, множество $S$ хорновских дизъюнктов невыполнимо.

## § 8. Преобразование формул логики предикатов. Сколемовская стандартная форма



Из предыдущей главы известно, что любую формулу логики предикатов можно представить в предваренной нормальной форме, т. е. в виде: $Q_1Q_2...Q_n\boldsymbol{B}$, где $Q_1,Q_2,...,Q_n$ некоторая совокупность кванторов, а формула $\boldsymbol{B}$ не содержит кванторов.

Для формулы $\boldsymbol{A} \sim Q_1Q_2...Q_n\boldsymbol{B}$ совокупность кванторов $Q_1,Q_2,...,Q_n$ называется префиксом формулы $\boldsymbol{A}$, а формула $\boldsymbol{B}$ – матрицей формулы $\boldsymbol{A}$. Будем дополнительно считать, что матрица приведена к конъюнктивной нормальной форме

Очевидно, что формула $\boldsymbol{A}$ является противоречием тогда и только тогда, когда $\neg\boldsymbol{A}$ – является логически общезначимой. Из свойств формул (см. § 5 второй главы) следует, что формула $\boldsymbol{B}$ является логически общезначимой тогда и только тогда, когда логически общезначимо замыкание $\boldsymbol{B}^*$ формулы $\boldsymbol{B}$. Как известно, замыкание $\boldsymbol{B}^*$ формулы $\boldsymbol{B}$ получается приписыванием к $\boldsymbol{B}$ кванторов всеобщности по всем её свободным переменным.

Таким образом, при выяснении логически общезначимости либо противоречивости мы будем считать, что имеем дело только с замкнутыми формулами, ибо если это не так, то можно добиться этого.

Осуществим следующие преобразования формул логики предикатов (формулы записаны с использованием связок $\neg, \Rightarrow$):

1) исключим знаки импликации, выразив их через $\neg$, $\vee$;

2) добьемся того, чтобы $\neg$ относилась только к элементарным формулам это можно сделать, используя правила перенесения отрицания через кванторы и законы де Моргана;

3) проведём стандартизацию (переименование) переменных для вынесения кванторов за скобки;

4) вынесем кванторы за скобки, т.е. получим предваренную нормальную форму:

$\boldsymbol{A} \sim Q_1x_1\ Q_2x_2 \ldots Q_nx_n\boldsymbol{B}$, здесь $\boldsymbol{B}$ – матрица формулы, а $Q_1x_1Q_2x_2 \ldots Q_nx_n$ – префикс (совокупность кванторов). Будем считать, что матрица приведена к конъюнктивной нормальной форме;

5) проведём исключение кванторов существования, введением сколемовских функций (Skolem T).

Исключение кванторов существования проводится следующим образом. Пусть $\boldsymbol{A}= Q_1x_1Q_2x_2 \ldots Q_nx_n\boldsymbol{B}$, где $Q_1x_1Q_2x_2 \ldots Q_nx_n$ – кванторы всеобщности или существования. Положим, что $Q_rx_r$ – квантор существования в префиксе $Q_1x_1Q_2x_2 \ldots Q_nx_n$, $1 \le r \le n$. Если никакой квантор всеобщности не стоит в префиксе левее $Q_rx_r$, то выбираем новую предметную постоянную $c$, отличную от всех предметных постоянных в $\boldsymbol{B}$, и заменяем все $x_r$ встречающиеся в $\boldsymbol{B}$ на $c$ и вычеркиваем $Q_rx_r$ из префикса. Покажем это на примере. Пусть имеем формулу $\exists x\, \forall y(P_1^2(x,y) \Rightarrow Q_1^2(x,a))$. Тогда для исключения квантора $\exists x$ введем постоянную $c$. В результате получим формулу: $\forall y(P_1^2(c,y) \Rightarrow Q_1^2(c,a))$.

Рассмотрим другой пример. Пусть имеем формулу $\exists x \exists y\, \forall z(P_1^3(x,y,z) \Rightarrow Q_1^4(a,b,x,y))$. Тогда, исключая кванторы существования, получим: $\forall z(P_1^3(c,d,z) \Rightarrow Q_1^4(a,b,c,d))$.

Если $Qx_{S_1}Qx_{S_2} \ldots Qx_{S_m}$ – список всех кванторов всеобщности, встречающихся левее квантора существования $Q_rx_r$, $1 \le s_1 < s_2 < \ldots < s_m \le n$, то выберем новую $m$–местную функциональную букву $f^m$, отличную от других функциональных букв из $\boldsymbol{B}$, и заменим все $x_r$ в $\boldsymbol{B}$ на $f^m(x_{S_1}, x_{S_2}, \ldots, x_{S_m})$ и вычеркнем $Q_rx_r$ из префикса.

Пример. Пусть имеем формулу $\forall x \exists y(P(x,y) \Rightarrow Q(f_1(a),y))$. Введя функцию $f_2$, с аргументом $x$ и исключая импликацию получим:

$\forall x(\neg P(x, f_2(x)) \vee Q(f_1(a), f_2(x)))$.

Пример. Пусть имеем другую формулу: $\forall x\, \forall y \exists z(P(x,y) \Rightarrow R(x,z,f_1(x),f_2(f_3(z))))$. Тоже, вводя необходимую функцию и исключая импликацию, получим: $\forall x\, \forall y(\neg P(x,y) \vee R(x,f_4(x,y),f_1(x),f_2(f_3(f_4(x,y)))))$.

Проводим описанную процедуру до тех пор, пока не исключим все кванторы существования. Полученная в результате формула есть *сколемовская стандартная форма*, для краткости *стандартная форма* формулы $\boldsymbol{A}$**.** Константы и функции, используемые для замены переменных квантора существования, называются *сколемовскими функциями*.

Пример. Пусть имеем формулу $\forall x \exists y \exists z(\neg P(x,y) \& R(x,z) \vee S(x,y,z))$.
Приведём матрицу формулы к к.н.ф.: $\forall x \exists y \exists z(\neg P(x,y) \vee S(x,y,z)) \& (R(x,z) \vee S(x,y,z))$. Затем введём функции $f(x)$, $g(x)$:

$\forall x(\neg P(x,f(x)) \vee S(x,f(x),g(x)) \& (R(x,g(x)) \vee S(x,f(x),g(x)))$.

Полученная формула является стандартной формой исходной формулы.

Элементарную формулу или её отрицание называют *литералом (литерой) в логике предикатов*.

*Дизъюнктом в логике предикатов* называют дизъюнкцию литералов.

Иногда литералы или дизъюнкты называют *клаузами* (clause – предложение, являющееся частью сложного предложения).

Пусть формула $\boldsymbol{A}$ приведена в предваренную нормальную форму, а ее матрица представлена в к.н.ф., т. е. $\boldsymbol{A}=(Q_1x_1Q_2x_2\ldots Q_nx_n)\boldsymbol{B}=(Q_1x_1Q_2x_2\ldots Q_nx_n)\boldsymbol{D_1\&D_2\&\ldots\&D_m}$ где $(Q_1x_1Q_2x_2\ldots Q_nx_n)$ префикс формулы $\boldsymbol{A}$, а $\boldsymbol{D_1,D_2,\ldots,D_m}$ – дизъюнкты. Положим, что стандартная форма для $\boldsymbol{A}$ равна $\boldsymbol{A_s}=(Q_1x_1Q_2x_2\ldots Q_nx_n)'\boldsymbol{C_1\&C_2\&\ldots\&C_m}$, где в префиксе опущены кванторы существования, а $\boldsymbol{C_i}$ получены из $\boldsymbol{D_i}$ введением сколемовских функций вместо переменных кванторов существования.

Отметим, что стандартная форма $\boldsymbol{A_s}$ формулы $\boldsymbol{A}$ определяется не единственным образом, ибо сколемовские функции можно вводить неоднозначно.

Имеет место следующая теорема.

**Теорема 3.8**. Формула $\boldsymbol{A}$ является противоречием тогда и только тогда, когда её стандартная форма $\boldsymbol{A_s}$ является противоречием.

**Доказательство**. Приведём $\boldsymbol{A}$ в предваренную нормальную форму

$\boldsymbol{A}= Q_1x_1\ Q_2x_2\ \ldots\ Q_nx_n\boldsymbol{B}.$

Пусть имеется только один квантор существования $Q_rx_r$:

$\boldsymbol{A}= \forall x_1\ldots\ \forall x_{r-1}\exists x_r\ \forall x_{r+1}\ldots\ \forall x_n\boldsymbol{B}(x_1,x_2,\ldots,x_n)$.

Положим $\boldsymbol{A_s}= \forall x_1\ldots\ \forall x_{r-1}\ \forall x_{r+1}\ldots\ \forall x_n\boldsymbol{B}(x_1,x_2,\ldots,x_{r-1},f(x_1,x_2,\ldots,x_{r-1}),\ x_{r+1},\ldots,x_n)$, где $f(x_1, x_2,\ldots,x_{r-1})$ – сколемовская функция.

Покажем, что $\boldsymbol{A}$ противоречива тогда и только тогда, когда противоречива $\boldsymbol{A_s}$.

Пусть $\boldsymbol{A}$ противоречие. Допустим, что $\boldsymbol{A_s}$ не противоречие, следовательно, существует интерпретация, в которой $\boldsymbol{A_s}$ выполнима, т.е. для $\forall x_1\ \forall x_2\ldots\ \forall x_{r-1}\ \exists x_r = f(x_1,x_2,\ldots,x_{r-1})$, что при $\forall x_{r+1}\ldots\ \forall x_n$ формула $\boldsymbol{B}$ принимает значение $''И''$, что противоречит тому, что $\boldsymbol{A}$ – противоречие. Следовательно, $\boldsymbol{A_s}$ – противоречие.

Обратно, пусть $\boldsymbol{A_s}$ – противоречие. Допустим, что $\boldsymbol{A}$ непротиворечиво, т.е. существует интерпретация, в которой $\boldsymbol{A}$ – выполнимо. Следовательно, для $\forall x_1,\ \forall x_2,\ldots,\ \forall x_{r-1}$ найдется $x_r$ такое, что при $\forall x_{r+1},\ldots,\ \forall x_n$ формула $\boldsymbol{B}=И$. Введем функцию $f(x_1,\ldots,x_{r-1})=x_r$. Тогда ясно, что $\boldsymbol{A_s}=И$, что противоречит условию $\boldsymbol{A_s}$ - противоречие.

Если в префиксе имеется $m$ кванторов существования, то доказательство проводится аналогично.

**Следствие 3.1**. Если $\boldsymbol{A}$ противоречие, и $\boldsymbol{A_s}=(Q_1,\ldots,Q_n)\boldsymbol{C_1}\&\boldsymbol{C_2}\&\ldots\&\boldsymbol{C_m}$ то $\boldsymbol{A}\sim\boldsymbol{C_1}\&\boldsymbol{C_2}\&,\ldots,\&\boldsymbol{C_m}$.

**Следствие 3.2**. Пусть $\boldsymbol{A_s}$ – стандартная форма формулы $\boldsymbol{A}$ и пусть $\boldsymbol{A}$ противоречие. Тогда $\boldsymbol{A_s}\sim\boldsymbol{A}$.

Отметим, что если $\boldsymbol{A}$ не является противоречием, то может быть, что $\boldsymbol{A}$ не равносильна $\boldsymbol{A_s}$. Например, пусть $\boldsymbol{A}=\exists xP(x)$. Тогда $\boldsymbol{A_s}=P(a)$. Построим интерпретацию. Пусть область интерпретации $\mathfrak{M}=\{1,2\}$ . Положим, что $a=1$, а $P(x)$ обозначает предикат «$x$ - четное число». Тогда $A_s=P(1)$ обозначает: «$1$ - четное число». Следовательно, формула $\boldsymbol{A_s}$ ложна в этой интерпретации. Формула $\boldsymbol{A}$ в этой интерпретации представляет истинное высказывание: $\exists x$(«$x$ - четное число»). Таким образом, для данной формулы $\boldsymbol{A}$ формулы $\boldsymbol{A}$ и $\boldsymbol{A_s}$ не равносильны.

## § 9. Унификация

Процесс унификации является основным в формальных преобразованиях, выполняемых при нахождении резольвент (для метода резолюций).

Пусть задано множество дизъюнктов. Каждый дизъюнкт составлен из литералов. Пример. Пусть имеем следующее множество дизъюнктов:

$$S = \{\overbrace{\underbrace{P(x, f(y), b)}_{\text{литерал}} \vee \underbrace{P(x, f(a), b)}_{\text{литерал}}}^{\text{дизъюнкт}},\ \overbrace{\underbrace{\neg P(c, f(a), b)}_{\text{литерал}}}^{\text{дизъюнкт}}\}$$

Термы литерала могут быть переменными, постоянными или выражениями, состоящим из функциональных букв и термов. Например, для литерала $P(x, f(y), b)$ имеем, что $x$ – переменная, $f(y)$ – сложный терм, $b$ - постоянная.

*Подстановочный частный случай литерала* получается при подстановке в литералы термов вместо переменных. Пусть имеем литерал $P(x, f(y), b)$. Его частными случаями будут:

$P_1 = P(z, f(w), b)$,
$P_2 = P(x, f(a), b)$,
$P_3 = P(g(z), f(a), b)$,

$P_4 = P(c, f(a), b)$ – *константный частный случай литерала* или *атом*, т.к. нет переменных.

Подстановки, примененные в рассматриваемом примере, можно обозначить следующим образом:

$\theta_1 = \{z/x, w/y\}$, здесь $z$ подставляется вместо $x$, а $w$ вместо $y$;

$\theta_2 = \{a/y\}$;

$\theta_3 = \{g(z)/x, a/y\}$;

$\theta_4 = \{c/x, a/y\}$.

Применение подстановки $\theta$ к литералу $P$ обозначаем $P_\theta$. Тогда имеем

$$P_{\theta_1} = P_1, \qquad P_{\theta_3} = P_3,$$

$$P_{\theta_2} = P_2, \qquad P_{\theta_4} = P_4.$$

Если $\theta$ – подстановка и она применяется к каждому из литералов $L_i$, то полученные частные случаи обозначаются через $\{L_i\}_\theta$.

Последовательное применение двух подстановок $\theta_1$ и $\theta_2$ дает новую подстановку $\theta_3$, которую обозначаем $\theta_3 = \theta_1 \circ \theta_2$.

Множество $\{L_i\}$ литералов называется *унифицируемым*, если существует такая подстановка $\theta$, что

$$(L_1)_\theta = (L_2)_\theta = \ldots = (L_n)_\theta.$$

В этом случае подстановку $\theta$ называют *унификатором* для $\{L_i\}$.

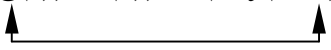
Пусть имеем множество литералов $\{P(x, f(y), b), P(x, f(b), b)\}$, где $L_1 = P(x, f(y), b)$, $L_2 = P(x, f(b), b)$. Подстановка $\theta = \{a/x, b/y\}$ является, очевидно, унификатором для этого множества литералов.

Унификатор $\sigma$ для множества выражений $\{E_1, E_2, \ldots, E_k\}$ называется наиболее общим унификатором тогда и только тогда, когда для каждого унификатора $\theta$ для этого множества существует такая подстановка $\lambda$, что $\theta = \sigma \circ \lambda$..

Существует алгоритм, называемый *алгоритмом унификации*, который приводит к наиболее общему унификатору для унифицируемого множества литералов $\{L_i\}$ и сообщает о неудаче, если это множество не унифицируемо.

**Алгоритм унификации**: Алгоритм начинает работу с пустой подстановки и шаг за шагом строит наиболее общий унификатор, если таковой существует. Предположим, что на $k$-ом шаге получена подстановка $\theta_k$. Если все литералы из $\{L_i\}$ в результате становятся идентичными, то $\theta = \theta_k$ и есть наиболее общий унификатор. В противном случае каждый из литералов в $\{L_i\}\theta_k$ рассматривается как цепочка символов и выделяется позиция первого символа, в которой не все из литералов имеют одинаковый символ. Рассмотрим пример двух литералов. Стрелками пометим позиции, где появились различные символы (при просмотре слева направо).

$\{P(a, f(a, g(z)), h(x)), P(a, f(a, u), g(w))\}$



Затем конструируется множество рассогласования, содержащее правильно построенные выражения из каждого литерала, который начинается с этой позиции (правильно построенное выражение представляет собой либо терм, либо литерал). Так для рассмотренного примера множеством рассогласования будет

$\{g(z), u\}$.

Далее модифицируем (если можно) подстановку $\theta_k$, чтобы сделать равным два элемента из множества рассогласования. Это можно сделать только тогда, когда множество рассогласования содержит переменную, которую можно положить равной одному из его термов. Если множество рассогласования не содержит переменных, то множество $\{L_i\}$ унифицировать нельзя.

Можно доказать, следующую теорему.

**Теорема 3.9** (Теорема Робинсона). Описанный выше алгоритм находит наиболее общий унификатор для множества унифицируемых литералов и сообщает о неудаче, если литералы неунифицируемы.

Рассмотрим пример. Пусть $S = \{P(a, x, f(g(y))), P(z, f(z), f(u))\}$. Найдем общий унификатор.

1. Пустая подстановка $\varepsilon$.

$S_0 = S$ – не единичный дизъюнкт, следовательно, не получили наиболее общий унификатор.

2. Множество рассогласования равно:

$W_0 = \{a, z\}$,

следовательно, $\theta_1 = \varepsilon \circ \{a/z\}$, тогда $S_1 = S_{\theta_1} = \{P(a, x, f(g(y))), P(a, f(a), f(u))\}$, $S_1$ – не единичный дизъюнкт. Множество рассогласований для $S_1$ равно:

$W_1 = \{x, f(a)\}$,

тогда $\theta_2 = \theta_1 \circ \{f(a)/x\} = \{a/z, f(a)/x\}$, и $S_2 = S_{\theta_2} = \{P(a, f(a)), f(g(y)), P(a, f(a), f(u))\}$. $S_2$ – не единичный дизъюнкт. Множество рассогласований для $S_2$ равно:

$W_2 = \{g(y), u\}$,

тогда вновь строим: $\theta_3 = \theta_2 \circ \{g(y)/u\} = \{a/z, f(a)/x, g(y)/u\}$ и

$S_3 = S_{\theta_2} = \{P(a, f(a)), f(g(y)), P(a, f(a), f(g(y)))\} = \{P(a, f(a), f(g(y)))\}$. $S_3$– единичный дизъюнкт, следовательно, $\theta_3$ наиболее общий унификатор.

Рассмотрим еще пример. Пусть $S = \{Q(f(a), g(x)), Q(y, y)\}$.

Пустая подстановка $\varepsilon$.

$S_0 = S$ – не единичный дизъюнкт и

$W_0 = \{f(a), y\}$, $\theta_1 = \varepsilon \circ \{f(a)/y\} = \{f(a)/y\}$,

$S_1 = S_{\theta_1} = \{Q(f(a), g(x)), Q(f(a), f(a))\}$, $S_1$ – не единичный дизъюнкт.

$W_1 = \{g(x), f(a)\}$.

В множестве $W_1$ нет переменной как элемента этого множества. Следовательно, алгоритм унификации завершается, и мы заключаем, что $S$ не унифицируемо.

*Насколько я могу судить, это один из тех несложных случаев, которые чрезвычайно трудны (Шерлок Холмс).*

*А. К. Дойль*

## § 10. Метод резолюций в логике предикатов

В логике высказываний метод резолюций применялся к множеству дизъюнктов, где дизъюнкты были формулами логики высказываний.

Теперь мы имеем формулы логики предикатов, в которых присутствуют термы (в частности переменные). Отличие метода резолюций в логике предикатов состоит в дополнительной процедуре работы с термами для унификации формул.

Пусть имеем множество литер $\{L_i\}$. Если для этого множества существует общий унификатор $\theta_1$, например, $(L_1)\theta_1 = (L_2)\theta_1 = (L_5)\theta_1$, то из этих трех литер достаточно оставить только одну. Если для некоторых оставшихся существует общий унификатор $\theta_2$, например, $(L_3)\theta_2 = (L_4)\theta_2$, то тоже оставляем только одну из них и т.д.

Пример. Пусть $\boldsymbol{D} = P(f(x)) \vee P(x) \vee Q(a,f(u)) \vee Q(x,f(b)) \vee Q(z,w)$, тогда:

$\theta = \{a/x, b/u, a/z, f(b)/w\}$, $\boldsymbol{D}_\theta = P(f(a)) \vee P(a) \vee Q(a, f(b))$, где

$P(f(a))$, $P(a)$, $Q(a, f(a))$ – факторы.

Если две или более литер (с одинаковым знаком) дизъюнкта $\boldsymbol{D}$ имеют наиболее общий унификатор $\theta$:

$(L_i)_\theta = (L_j)_\theta = (L_k)_\theta$ ,

то оставление одного из этих литералов вместо всех них называют *склейкой*.

Пусть имеем два дизъюнкта $\boldsymbol{D}_1$ и $\boldsymbol{D}_2$ и переменные входящие в $\boldsymbol{D}_1$ не входят в $\boldsymbol{D}_2$ и обратно. Если это не так, то переименованием переменных этого можно добиться. Пусть в $\boldsymbol{D}_1$ есть литера, например, $L_1$, а в $\boldsymbol{D}_2$ - литера $L_2$. Если $L_1$ и $\neg L_2$ имеют наиболее общий унификатор, т. е.

$$(L_1)_\theta = (\neg L_2)_\theta ,$$

то новый дизъюнкт $\boldsymbol{R}$:

$$\boldsymbol{R} = ((\boldsymbol{D}_1)_\theta - (L_1)_\theta) \vee ((\boldsymbol{D}_2)_\theta - (L_2)_\theta)$$

называется *бинарной резольвентой* $\boldsymbol{D}_1$ и $\boldsymbol{D}_2$ (в логике предикатов). Литеры $L_1$ и $L_2$ называются *отрезаемыми литерами*.

Рассмотрим пример. Пусть $\boldsymbol{D}_1 = P(x) \vee Q(x)$, $\boldsymbol{D}_2 = \neg P(a) \vee T(x)$. Переименуем $x$ в $\boldsymbol{D}_2$: $\boldsymbol{D}_2 = \neg P(a) \vee T(y)$. Положим, что $\theta = \{x/a\}$. Тогда имеем

$$(\boldsymbol{D}_1)_\theta = P(a) \vee Q(a) , (\boldsymbol{D}_2)_\theta = \neg P(a) \vee T(y) ,$$

Отбрасываем литеры $P(a)$, $\neg P(a)$ и получим бинарную резольвенту:

$$\boldsymbol{R} = Q(a) \vee T(y).$$

Пусть имеем $\boldsymbol{D_1} = \neg P(x) \vee Q(x)$, $\boldsymbol{D_2} = \neg Q(x) \vee T(x)$. Тогда $\boldsymbol{D_2} = \neg Q(y) \vee T(y)$ и $\boldsymbol{R} = \neg P(y) \vee T(y)$.

*Резольвентой дизъюнктов* – посылок $\boldsymbol{D_1}$ и $\boldsymbol{D_2}$ является одна из следующих резольвент:

1) бинарная резольвента $\boldsymbol{D_1}$ и $\boldsymbol{D_2}$,

2) бинарная резольвента $\boldsymbol{D_1}$ и склейки $\boldsymbol{D_2}$,

3) бинарная резольвента $\boldsymbol{D_2}$ и склейки $\boldsymbol{D_1}$,

4) бинарная резольвента склейки $\boldsymbol{D_1}$ и склейки $\boldsymbol{D_2}$.

Применение описанных резольвент к множеству дизъюнктов и называется методом резолюций.

Можно доказать следующую важную теорему.

**Теорема 3.10** (полнота метода резолюций). Множество $S$ дизъюнктов невыполнимо тогда и только тогда, когда существует вывод пустого дизъюнкта из $S$.

Проблема дедукции логики предикатов состоит, как и в логике высказываний, в выяснении будет ли формула $\boldsymbol{B}$ логическим следствием формул $\boldsymbol{A_1}, \boldsymbol{A_2}, \ldots, \boldsymbol{A_n}$.

Теоремы 3.1-3.4, доказанные для логики высказываний, можно распространить и для логики предикатов, записывая всюду вместо слова «тавтология» слова «логически общезначимая формула».

Из вышеизложенного можно получить следующую последовательность действий для выяснения будет ли формула $\boldsymbol{B}$ логическим следствием формул $\boldsymbol{A_1}, \boldsymbol{A_2}, \ldots, \boldsymbol{A_n}$.

1. Строим конъюнкцию $\boldsymbol{C} = \boldsymbol{A_1}\&\boldsymbol{A_2}\& \ldots \& \boldsymbol{A_n}.\& \neg \boldsymbol{B}$. Отметим, что требуемое следствие (заключение) взято с отрицанием.

2. Находим (сколемовскую) стандартную форму $\boldsymbol{C_s}$ для формулы $\boldsymbol{C}$. Положим, что форма $\boldsymbol{C_s} = (Q_1x_1Q_2x_2\ldots Q_nx_n)'\boldsymbol{C_1}\&\boldsymbol{C_2}\&\ldots\&\boldsymbol{C_m}$, где $=(Q_1x_1Q_2x_2\ldots Q_nx_n)'$ префикс формулы без кванторов существования, а $\boldsymbol{C_1}, \boldsymbol{C_2}, \ldots, \boldsymbol{C_m}$ – дизъюнкты, в которых по необходимости введены сколемовские функции.

3. Для множества дизъюнктов $S=\{\boldsymbol{C_1}, \boldsymbol{C_2}, \ldots, \boldsymbol{C_m}\}$ применяем метод резолюций.

Формула $\boldsymbol{B}$ будет логическим следствием формул $\boldsymbol{A_1}, \boldsymbol{A_2}, \ldots, \boldsymbol{A_n}$ тогда и только тогда когда существует вывод пустого дизъюнкта из $S$.

*В мозгах как на мануфактуре, есть ниточки и узелки,*
*Посылка не по той фигуре грозит запутать узелки.*
*И. Гете (Фауст)*

## § 11. Приложение метода резолюций для анализа силлогизмов Аристотеля.

По Аристотелю «*силлогизм же есть речь, в которой при предположении чего-нибудь из него с необходимостью вытекает нечто отличное от утверждённого и,* [именно] *в силу того, что это есть*».

Известно, что каждый силлогизм Аристотеля можно представить как некоторую комбинацию предложений следующих видов:

*A:* Всякий *M* есть *P* $\quad \forall x(M(x) \Rightarrow P(x));$

*E:* Всякий *M* не есть *P* $\quad \forall x(M(x) \Rightarrow \neg P(x));$

*I:* Некоторый *M* есть *P* $\quad \exists x(M(x) \,\&\, P(x));$

*O:* Некоторый *M* не есть *P* $\quad \exists x(M(x) \,\&\, \neg P(x)).$

Силлогизм Аристотеля это вывод, который можно получить на основании истинности двух посылок указанного вида. Силлогизм обозначают тремя буквами по виду посылки и виду заключения. Так, например, силлогизм *AAA* можно представить в виде:

всякий *M* есть *P,*
всякий *P* есть Q,
следовательно, всякий *M* есть *Q*.

В зависимости от положения среднего термина в посылках, различают четыре фигуры силлогизма. Модусами силлогизма называются разновидности силлогизма отличающиеся качественной и количественной характеристикой посылок и заключения. Силлогизмам присваивают собственные имена, например, силлогизм *AAA* называется Barbara (в слове “ Barbara” гласными являются три буквы а).

Можно подсчитать, что количество различных модусов силлогизмов Аристотеля равно 256 (по 64 в каждой фигуре). Сколько же среди этих модусов правильных, когда из истинности посылок следует истинность заключения? Известно, что правильных, модусов 24. Рассмотрим применение метода резолюций для выяснения правильности модусов.

Рассмотрим силлогизм Барбара построенный по первой фигуре. В символьной записи он означает, что из истинности формул

$$\forall x\,(S(x) \Rightarrow P(x)),$$
$$\forall x\,(P(x) \Rightarrow Q(x))$$

нужно получить истинность формулы

$$\forall x(S(x) \Rightarrow Q(x)),$$

т. е. доказать, что из формул $\forall x(S(x) \Rightarrow P(x))$ и $\forall x(P(x) \Rightarrow Q(x))$ логически следует формула $\forall x(S(x) \Rightarrow Q(x))$. Известно, что если формула

$$\boldsymbol{C} = (\forall x(S(x) \Rightarrow P(x)) \,\&\, (\forall x(P(x) \Rightarrow Q(x)) \,\&\, \neg \forall x(S(x) \Rightarrow Q(x)) \qquad (3.9)$$

является противоречием, то указанное логическое следствие будет доказано. Так как формула $\boldsymbol{C}$ содержит только одноместные предикатные буквы *(S,P,Q),* то выяснение противоречие $\boldsymbol{C}$ или нет можно было бы свести к

проблеме разрешимости алгебры высказываний. При этом получили бы нетривиальную задачу в силу следующего. В формуле $\boldsymbol{C}$ три предикатные буквы, следовательно, нужно было бы взять множество, содержащее $2^3 = 8$ элементов, положим $M = \{1,2,\ldots,8\}$. Тогда каждый предикат порождает на $M$ по 8 высказываний: $S(1),\ldots,S(8);\ P(1),\ldots,P(8)\ ;\ Q(1),\ldots,Q(8)$. Следовательно, в пропозициональной форме, построенной для $\boldsymbol{C}$, будет 24 переменных и таблица истинности будет содержать $2^{24}$ строк. Ясно, что составление такой таблицы сложно даже с помощью ЭВМ. При применении метода резолюций вычисления оказываются совсем незначительными.

Следуя указанной в параграфе 10 процедуре, нужно получить сколемовскую стандартную форму $\boldsymbol{C}_s$ для формулы $\boldsymbol{C}$, которая имеет вид (3.9). Сначала переименуем вхождения $x$ во второй посылке на $y$, а в заключении силлогизма на $z$ (в первой посылке переменную $x$ не переименовываем). Легко видеть, что первые два дизъюнкта в $\boldsymbol{C}_s$ будут следующими:

1) $\neg S(x) \vee P(x)$,
2) $\neg P(y) \vee Q(y)$

которые, очевидно, получены из посылок силлогизма. Заключение силлогизма в (3.9) взято с отрицанием: $\neg\forall z(S(z) \Rightarrow Q(z))$. Преобразуя последнюю формулу, получим: $\exists z(S(z) \& \neg Q(z))$. При нахождении сколемовской стандартной формы квантор существования можно вынести первым и затем, удаляя его, вводим постоянную $a$ вместо переменной удаляемого квантора (переменной $z$). В результате в стандартной форме $\boldsymbol{C}_s$ будут еще два дизъюнкта:

3) $S(a)$,
4) $\neg Q(a)$,

которые получены из заключения силлогизма. Теперь к множеству дизъюнктов 1)-4) применяем метод резолюций. Тогда имеем последовательно:

5) $P(a)$ из 3) и 1),
6) $\neg P(a)$ из 4) и 2),
7) $\square$ из 5) и 6).

Таким образом, получили пустой дизъюнкт, следовательно, силлогизм *AAA* верен.

Рассмотрим силлогизм Darii (*AII* по 3-ей фигуре). В символической записи имеем:

$$\forall x\ (P(x) \Rightarrow S(x))$$
$$\exists x\ (P(x)\ \&\ Q(x))$$
$$\text{----------}$$
$$\exists x\ (S(x)\ \&\ Q(x)).$$

Преобразуя, как выше, получим:

1) $\neg P(x) \vee S(x)$
2) $P(a)$
3) $\neg Q(a)$

*4)* $\neg S(y) \vee \neg Q(y)$

Далее получаем следующие бинарные резольвенты:

*5) S(a)* из *1)* и *2)*,

*6)* $\neg S(a)$ из *3)* и *4),*

*7)* $\square$ из *5)* и *6).*

Следовательно, силлогизм *AII* верен.

Рассмотрим ещё силлогизм, который в символьной записи имеет вид:

$\forall x\ (P(x) \Rightarrow S(x)),$

$\forall x\ (Q(x) \Rightarrow \neg P(x)),$

---

$\forall x\ (Q(x) \Rightarrow \neg S(x))$ .

Преобразовав формулы, получим:

*1)* $\neg P(x) \vee S(x),$

*2)* $\neg Q(y) \vee S(y),$

*3) Q(a),*

*4) S(a),*

Далее получаем бинарную резольвенту:

*5) 7P(a)* из *2)* и *3).*

Никаких других резольвент получить нельзя, в том числе нельзя получить пустой дизъюнкт, тогда из теоремы полноты метода резолюций следует, что из посылок не будет (не всегда будет) следовать заключение. Невыполнимость этого силлогизма можно пояснить на Рис. 3.1. Возможны различные варианты расположения областей истинности предикатов, см. Рис. 3.1, на котором области истинности обозначены теми же буквами, что и сами предикаты. В варианте изображённом на Рис. 3.1 а) видно, что из истинности посылок следует истинность заключения. В вариантах изображённых на Рис. 3.1 б), в) из истинности посылок не следует истинность заключения.

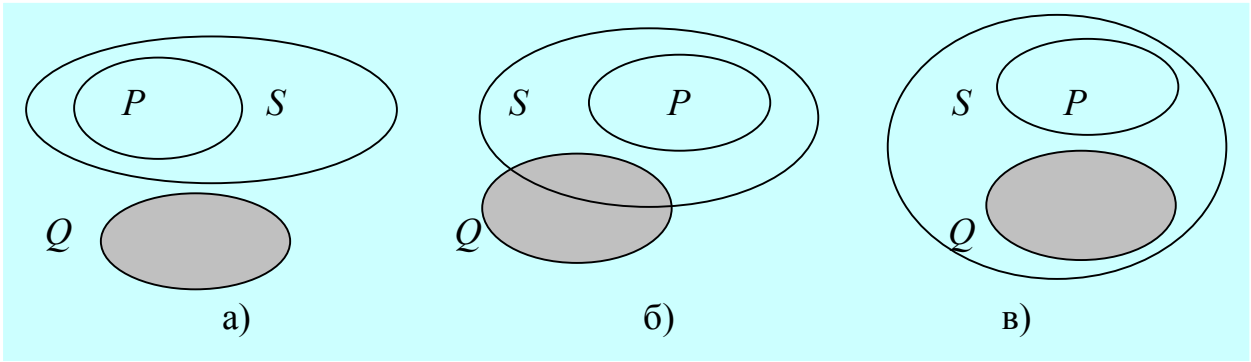



Рис. 3.1

## § 12. Использование метода резолюций в языке ПРОЛОГ

Язык программирования ПРОЛОГ существенно опирается на метод

резолюций. Само название *ПРОЛОГ* есть сокращение, означающее программирование в терминах логики. Вычисления на ПРОЛОГе можно рассматривать как доказательство теорем с использованием метода резолюций и применением дизъюнктов специального вида – хорновских дизъюнктов.

Отметим следующую принципиальную отличительную черту языка ПРОЛОГ. В языках программирования таких как АЛГОЛ, ФОРТРАН, С++ описывается как решать ту или иную задачу. Программирование на ПРОЛОГе указывает только что нужно (необходимо) сделать, т.е. указывается цель, а не процедура решения. Программист должен дать определение ситуации и формулировать задачу, а система сама решает задачу. При решении задачи используется метод резолюций.

Рассмотрим простейший пример. Пусть задано следующее дерево родственных отношений, изображенное на следующем рисунке 3.2.

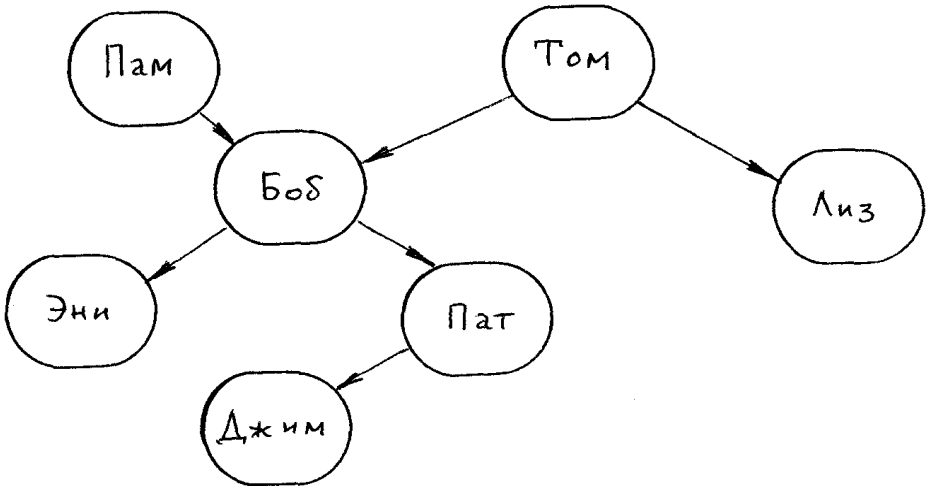



Рис.3.2. Дерево родственных отношений.

То, что Том является родителем Боба можно записать на Прологе так:

**родитель (том, боб).**

Все приведенное дерево родственных отношений описывается следующей *пролог-программой*:

**родитель (пам, боб).** (3.10)

**родитель (том, боб).** (3.11)

**родитель (том, лиз).** (3.12)

**родитель (боб, эни).** (3.13)

**родитель (боб, пат).** (3.14)

**родитель (пат, джим).** (3.15)

Отметим, что каждое предложение пролога заканчивается точкой. После ввода такой программы в пролог-систему этой системе можно задавать вопросы, касающиеся отношения родитель. Например, является ли Боб родителем Пата? Этот вопрос ставится следующим образом:

**? - родитель (боб, пат).**

Система ответит

**YES** (да).

Ответ ищется следующим образом. Вопрос преобразуется в дизъюнкт:

$\overline{\text{родитель (боб, пат)}}$. (3.16)

Далее из (3.14) и (3.16) бинарной резолюцией получаем пустой дизъюнкт □, что и означает положительный ответ.

Другим вопросом мог бы быть такой

**? - родитель (пам, лиз).**

Система ответит

**No** (нет) .

Этот ответ в данном случае требует особого пояснения. Для получения указанного ответа образуется дизъюнкт:

$\overline{\text{родитель (пам, лиз)}}$. (3.17)

Этот дизъюнкт ни с одним из дизъюнктов (3.10)-(3.15) не дает пустого дизъюнкта. По идеологии вопросно-ответных систем система должна была бы организовать отрицание вопроса, т.е. дизъюнкт:

родитель (пам, лиз). (3.18)

Но и этот дизъюнкт вместе с (3.10)-(3.15) не порождает пустого дизъюнкта. Следовательно, система должна была бы ответить: "недостаточно информации". Но в реализации ПРОЛОГа этого не будет. Система рассмотрит только дизъюнкты (3.10)-(3.15) и (3.17) и так как они не дают пустой дизъюнкт, то получим ответ: **No** (нет). Это происходит оттого, что считается, что в программу включены все возможные знания об объектах. Тогда, поскольку нет никакой информации является ли Пам родителем Лиз, то система говорит: **No** (нет), а не будет строить дизъюнкт (3.18).

Можно задавать и такие вопросы: "Кто является родителем Лиз?" в виде:

**? - родитель (X,лиз).**

Система ответит: **X = том**

Покажем, как находится ответ на этот вопрос.

Вопрос, как и выше, можно было бы преобразовать в дизъюнкт:

$$\neg\text{родитель}(X, \text{лиз}).$$

Здесь имеется переменная X. Если подставлять вместо X постоянные (пам, том, боб, пат) и сравнивать с (3.10)-(3.15), можно получить пустой дизъюнкт при X равном «том». Получится пустой дизъюнкт, означающий, что X=том истинно, но полученный ответ не сохранится. Для сохранения ответа вводится специальная функция, зависящая от X, называемая ANS-предикатом (от answer – ответ).

Следовательно, к исходным дизъюнктам (3.10)-(3.15) нужно добавить дизъюнкт соответствующий вопросу:

$$\neg\text{родитель}(X,\text{лиз}) \vee \text{ANS}(X). \qquad (3.19)$$

Применяя метод резолюций из (3.12) и (3.19) система получит: ANS(том) и выдаст результат (на печать): **X=том**

Вопрос: "Кто дети Боба?" можно передать системе в виде:

**? - родитель (боб,X).**

В данном случае ответ не единственен.

В зависимости от вариантов ПРОЛОГа система может выдавать один из ответов, например:

**X = пат**

и ждать введения точки с запятой, после чего выдает второй ответ:

**X = эни**

В других реализациях ПРОЛОГа могут выдаваться последовательно все возможные ответы без дополнительного запроса.

Системе можно задавать и другие вопросы, например,

1) "Кто родитель Боба?":

**? - родитель (X,боб).**

Система ответит: **X = пам; X=том**

2) "Кто чей родитель?":

**? - родитель (X,Y).**

Система распечатает возможные значения **X** и **Y**.

3) "Кто внуки Тома? Так как в системе не введено понятие внуков, то строим сложный вопрос.

**?-родитель (том, X ), родитель (X, Y).**

Запятая между предложениями воспринимается как конъюнкция. Составной вопрос интерпретируется: найти X,Y, удовлетворяющие двум требованиям

родитель(том,X) и родитель (X,Y).

Конъюнкция коммутативна, поэтому можно вопрос сформулировать и иначе

**?-родитель(X,Y), родитель(том,X)**

Ответ не зависит от порядка в силу того, что система получит для вопроса дизъюнкт:

$\neg$родитель(X,Y)$\vee \neg$родитель(том,X),

который равносилен дизъюнкту:

$\neg$родитель(том,X)$\vee \neg$ родитель(X,Y).

На третий вопрос система ответит

**X = боб Y = эни;**

**X = боб Y = пат**

Пользователь ПРОЛОГа может совершенно не знать метод резолюций. Система сама находит ответы.

## § 13. Введение и использование правил в ПРОЛОГе

В качестве расширения рассмотренной в § 12 программы на ПРОЛОГе введем отношение дети, которое обратно отношению родитель. Можно было бы определить дети тем же способом, что и родитель, т.е. представив описок:

**дети (лиз, том).**

**дети (боб, пам).**

и так далее. Однако это отношение можно определить проще, использовав, тот факт, что оно обратно отношению родитель, которое уже определено. Такой способ основан на следующем логическом утверждении:

Для всех X и Y

Y дети X, если

X является родителем Y.

Соответствующее прологовское предложение с тем же смыслом имеет вид:

**дети (Y,X): - родитель (X,Y).** (3.20)

Приведенное прологовское предложение (3.20) называется правилом.

Предложение

**родитель (том, лиз).**

считающееся фактом, безусловно, истинно. Правила описывают утверждения, которые могут быть истинными, если выполнены условия.

К рассмотренной в параграфе 12 программе добавим еще предложение (3.20). Спросим полученную программу, является ли Лиз отпрыском Тома?

**? - дети (лиз, том).**

Предложение (3.20) на языке логики имеет вид:

родитель(X,Y) $\Rightarrow$ дети(Y,X).

Преобразуем последнее выражение в дизъюнкт:

$\neg$родитель(X,Y)$\vee$дети(Y,X). (3.21)

Вопрос к системе преобразуется в дизъюнкт:

$\neg$дети (лиз, том). (3.22)

Из (3.21) и (3.22) получим:

$\neg$родитель (том, лиз).

Далее, используя предложение (дизъюнкт) (3.12) получим пустой дизъюнкт, следовательно, система ответит

**YES** (да).

## § 14. Рекурсивное задание правил в ПРОЛОГе

Вновь рассмотрим программу из параграфа 12. Добавим к этой программе, кроме отношений родитель и дети, еще одно отношение - предок. Определим его через отношение родитель. Все отношение можно выразить с помощью двух правил. Первое правило будет определять непосредственных (ближайших) предков, а второе - отдаленных. Будем считать, что некоторый X является отдаленным предком некоторого Z, если между X и Z существует цепочка людей, связанных между собой отношением родитель - родитель.

Первое правило простое и его можно сформулировать так:

Для всех X и Z

X - предок Z, если

X - родитель Z.

Это переводится на Пролог в виде предложения

```
предок(X,Z): -
    родитель(X,Z).
```

Второе правило сложнее. Один из способов определения отдаленных родственников задать их следующим множеством предложений

```
предок(X,Z):-
        родитель(X,Z).

_____________________________

предок(X,Z): -
        родитель(X,Y),
        родитель(Y,Z).
_____________________________
предок(X,Z):-
        родитель(X,Y1),
        родитель(Y1,Y2),
        родитель(Y2,Z).
 -------------------------
```

```
предок(X,Z):-
        родитель(X,Y1),
        родитель(Y1,Y2),
        родитель(Y2,Y3),
        родитель(Y3,Z).
```

Эта программа длинна и, что важнее, работает только в определенных пределах. Она будет обнаруживать предков лишь до определенной глубины фамильного дерева.

Существует компактная и корректная формулировка отношения предок, которая работает на любую глубину. Ключевая идея для этого: определить отношение предок через него самого, т.е. ввести рекурсивное определение. Это определение будет следующим:

Для всех X и Z
X- предок Z, если
существует Y, такой, что
(1) X -родитель Y и
(2) Y -предок Z.

Предложение ПРОЛОГа, имеющее тот же смысл, записывается в виде:

```
предок(X,Z):-
        родитель(X,Y),
        предок(Y,Z).
```

Полная программа для отношения предок содержит оба правила: одно для ближайших предков и другое для отдаленных предков. В результате имеем:

```
предок(X,Z): -
        родитель(X,Z).

предок(X,Z):-
        родитель(X,Y),                    (3.23)
        предок(Y,Z).
```

Отметим, что рекурсивное задание правил проводится не единственным образом. Так предложение (3.23) можно заменить, например, на следующие

```
предок(X,Y): -
        родитель(X,Y).
предок(X,Z): -
        предок(X,Y),
        предок(Y,Z).
```

Если к рассмотренной в параграфе 12 программе добавить предложение (3.23), то полученную программу можно, например, спросить:

«Кто потомки Пам?»

На Прологе этот вопрос запишем в виде:

**?-предок(пам,X).**

Система ответит

**X=боб;**

**X=эни;**

**X=пат;**

**X=джим**

Выясним как получает программа ответ. Предложения (3.23) записываются в виде:

родитель(X,Z)⇒предок(X,Z),

родитель(X,Y)&предок(Y,Z)⇒предок(X,Z).

Преобразовав эти формулы получим:

$$\neg\text{родитель}(X,Z)\vee\text{предок}(X,Z), \qquad (3.24)$$

$$\neg\text{родитель}(X,Y)\vee\neg\text{предок}(Y,Z)\vee\text{предок}(X,Z). \qquad (3.25)$$

Вопрос к системе преобразуется к виду

$$\neg\text{предок}(\text{пам},X)\vee\text{ANS}(X). \qquad (3.26)$$

Для получения ответа используется предложения (дизьюнкты) (3.10)-(3.15) и дизъюнкты (3.24)-(3.26).

Непосредственный потомок для Пам выявится, если из (3.25) и (3.26) получить бинарную резольвенту:

$$\neg\text{родитель}(\text{пам},Z)\vee\text{ANS}(Z). \qquad (3.27)$$

Затем получим ANS(боб) как бинарную резольвенту из (3.27) и (3.10).

Потомки второго уровня (внуки Пам) выявятся в результате получения следующих резольвент

$$\neg\text{родитель}(\text{пам},Y)\vee\neg\text{предок}(Y,X)\vee\text{ANS}(X), \qquad (3.28)$$

которое получено из (3.25) и (3.26),

$$\neg\text{родитель}(X,Y)\vee\neg\text{родитель}(\text{пам},X)\vee\text{ANS}(Y), \qquad (3.29)$$

которое получено из (3.28) и (3.24),

$$\neg\text{родитель}(\text{боб},Y)\vee\text{ANS}(Y), \qquad (3.30)$$

получено из (3.29) и (3.10). Далее имеем:

ANS(эни**)** -получено из (3.30) и (3.13),

ANS(пат) -получено из (3.30) и (3.14).

Потомки третьего уровня (правнуки Пам) выявятся в результате следующих преобразований

$$\neg\text{родитель}(\text{пам},Z)\vee\text{ANS}(Z), \qquad (3.31)$$

получено из (3.26) и (3.24),

$$\neg\text{предок}(\text{боб},X)\vee\text{ANS}(X), \qquad (3.32)$$

получено из (3.31) и (3.10),

$$\neg\text{родитель}(\text{боб},Y)\vee\neg\text{предок}(Y,X)\vee\text{ANS}(X), \qquad (3.33)$$

получено из (3.32) и (3.25),

$$\neg\text{родитель}(X,Y)\vee\neg\text{родитель}(\text{боб},X)\vee\text{ANS}(Y), \qquad (3.34)$$

получено из (3.33) и (3.24),

$$\neg\text{родитель}(\text{пат},Y)\vee\text{ANS}(Y), \qquad (3.35)$$

получено из (3.34) и (3.14). Затем из (3.35) и (3.15) получим: ANS(джим).

Отметим, что в каждом из рассмотренных случаев выполняется следующее:

1) в первой выполняемой резолюции используется дизъюнкт, построенный для вопроса;

2) в каждой последующей резолюции должна участвовать резольвента предыдущей резолюции.

## § 15. Особенности ПРОЛОГа

Целью данного пособия не является изложение языка ПРОЛОГ. Эти параграфы только иллюстрируют, как работает логика в многообещающем языке ПРОЛОГ. Этот язык некоторые исследователи считали языком будущего.

Язык ПРОЛОГ находит существенное использование:

- при описании и решении задач на графах;
- в экспертных системах (системы испытаний, медицинская диагностика, нахождение неисправностей в технических системах);
- в системах искусственного интеллекта (решение задач, доказательство теорем, различные игры, такие, как шахматы, кубики);
- при создании динамических реляционных баз данных;
- при создании систем перевода с одного языка на другой;
- в системах управления производственными процессами;
- при создании пакетов символьных вычислений для решения уравнений, дифференцирования и интегрирования;
- при создании специализированных и общих вопросно-ответных систем.

Метод резолюций используется не только в языке ПРОЛОГ, но и в некоторых системах управления базами данных, и некоторых

специализированных экспертных системах. Программа на ПРОЛОГе состоит из:

*фактов* типа: **родитель(том,боб).**
**родитель(боб,лиз).**
*правил*: **предок(X,Y):**
**родитель(X,Z)**
**предок(Z,Y).**
*целей*: **?-предок(X,том).**

Механизм ПРОЛОГа состоит в том ,чтобы доказать истинность цели и (или) найти значение переменного цели при котором цель истинна.

Вычисления всегда начинаются с цели и рассматриваются возможные варианты нахождения резольвент. Вычисления в ПРОЛОГе можно интерпретировать как нисходящие вычисления, см. Рис. 3.3.

Вычисления продолжаются насколько возможно вглубь. При выяснении невыполнимости (на этом пути) осуществляется возврат до основания ветви дерева и продолжается обход слева от просмотренной ветви. Этот процесс называется бэктрекингом (backtracking). Бэктрекинг позволяет обойти все ветви возможного дерева выводов. Ясно, что если цель достигается в самой левой ветви, то тратится много времени на обход ненужных участков. Для управления бэктрекингом (изменения порядка обхода) в Прологе вводятся нелогические примитивы: **cut, retract, assert** и т. д. Они позволяют управлять процедурой обхода.

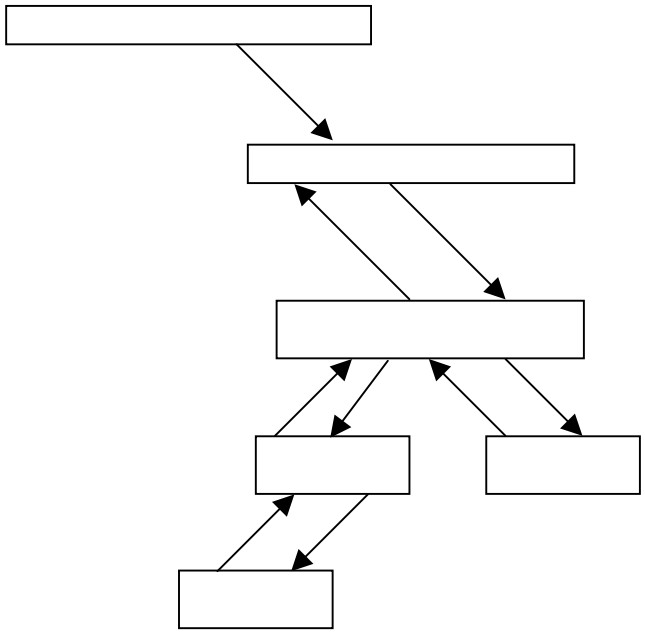


Рис. 3.3. Схема нисходящих вычислений в Прологе

В настоящее время разработаны различные модификации Пролога, например: Coustraint ПРОЛОГ, Мпролог, PRO-SQL, EDUCE, Deitalog и др.

Существует много работ по связыванию ПРОЛОГа с базами данных. В принципе программу на ПРОЛОГе можно уже рассматривать как некоторую базу данных.

*Соберись и действуй.*
*Билл Гейтс (Бизнес со скоростью мысли)*

## § 16. Вопросы и темы для самопроверки

1. Определение логического следствия из данной пропозициональной формы (формулы логики высказываний), из множества данных форм.
2. Свойства логического следования.
3. Критерий логического следствия из множества формул логики высказываний.
4. Проблема дедукции логики высказываний. Решение проблемы дедукции с использованием противоречивости специальной формулы.
5. Литералы, контрарные литералы, дизъюнкты.
6. Бинарные резольвенты дизъюнктов логики высказываний.
7. Теорема о полноте метода резолюций.
8. Метод резолюций в логике высказываний.
9. Метод насыщения уровня.
10. Стратегии вычёркивания.
11. Лок-резолюция. Теорема о полноте метода лок-резолюций. Зависит ли результат лок-резолюции от изменения индексов литералов?
12. Метод резолюций для хорновских дизъюнктов.
13. Определение логического следствия для формул логики предикатов.
14. Определение сколемовских стандартных форм. Для каждой ли формулы логики предикатов существует сколемовская стандартная форма?
15. Единственна ли сколемовская стандартная форма для заданной формулы логики предикатов?
16. Алгоритмы нахождения сколемовских стандартных форм.
17. Когда формула равносильна своей сколемовской стандартной форме?
18. Унификация. Алгоритмы унификации.
19. Метод резолюции в логике предикатов, отличия от метода резолюций в логике высказываний.
20. Приложение метода резолюций для анализа силлогизмов Аристотеля.
21. Использование метода резолюций в ПРОЛОГе при работе с фактами.
22. Для каких задач вводится ANS-предикат?
23. Приведите примеры введения правил в ПРОЛОГе.
24. Рекурсивное задание правил в ПРОЛОГе.

## § 17. Упражнения

1. Доказать, что $A \models B\&C$ тогда и только тогда, когда $\models A \Rightarrow B\&C$[1].
2. Доказать, что
   а) $A_1,A_2,\ldots,A_n \models B$ тогда и только тогда, когда $A_1,A_2,\ldots,A_{n-1} \models (A_n \Rightarrow B)$;
   б) $A_1,A_2,\ldots,A_n \models B$ тогда и только тогда, когда
   $\models (A_1 \Rightarrow (A_2 \Rightarrow (\ldots (A_n \Rightarrow B)\ldots)))$;
   в) $A_1,A_2,\ldots,A_n \models B$ тогда и только тогда, когда $A_1\&A_2\&\ldots\&A_n \models B$.
3. Докажите, что если $A \models C$ и $B \models C$, то $A \vee B \models C$ (доказательство разбором случаев).
4. Докажите, что если $A \models B$, то для произвольной пропозициональной формы $C$ имеет место $A\&C \models B$.
5. Докажите, что
   а) $A, A \Rightarrow B \models B$ (правило заключения); б) $A,B \models A\&B$;
   в) $A,B \models A \vee B$; г) $A \models A \vee B$;
   д) $A \Rightarrow B,\ \neg B \models \neg A$ (правило отрицания);
   е) $A \Rightarrow B,\ B \Rightarrow C \models A \Rightarrow C$ (правило силлогизма);
   ж) $A \Rightarrow B \models \neg B \Rightarrow \neg A$ (правило контрапозиции);
   з) $A \Rightarrow (B \Rightarrow C) \models B \Rightarrow (A \Rightarrow C)$ (правило перестановки посылок);
   и) $A \Rightarrow (B \Rightarrow C) \models A\&B \Rightarrow C$ (правило соединения посылок);
   к) $A\&B \Rightarrow C \models A \Rightarrow (B \Rightarrow C)$ (правило разъединения посылок);
   л) $A\&B \models B$;
   м) $B_1\&B_2\&\ldots\&B_k \models B_i$ для каждого $i(1 \le i \le k)$.

Из задачи 5 (м) следует, что если пропозициональная форма $C$ представлена в к.н.ф. или с.к.н.ф., то каждый ее конъюнктивный член, а также конъюнкция любого числа конъюнктивных членов является логическим следствием из $C$. Это позволяет находить следствия из данных пропозициональных форм. Пусть даны пропозициональные формы $A$ и $A \Rightarrow B$. требуется найти все логические следствия в с.к.н.ф. этих форм. Для этого составляем конъюнкцию данных форм: $A\&(A \Rightarrow B)$ и находим равносильную ей с.к.н.ф.:

$$(A \vee B)\&(A \vee \neg B)\&(\neg A \vee B).$$

Искомые следствия суть:

$A \vee B$; $A \vee \neg B$;
$\neg A \vee B$; $(A \vee B)\&(A \vee \neg B) = A$;
$(A \vee B)\&(\neg A \vee B) = B$; $(A \vee \neg B)\&(\neg A \vee B) = A \equiv B$;
$(A \vee B)\&(A \vee \neg B)\&(\neg A \vee B) = A\&B$.

6. Найти все следствия в с.к.н.ф. для следующих форм:
   а) $A\&B,\ \neg B$; б) $A \Rightarrow B,\ \neg A$;
   в) $A \Rightarrow B,\ B \Rightarrow \neg A,\ A$; г) $A \Rightarrow B,\ \neg B \Rightarrow \neg A$;
   д) $B \vee C,\ B \Rightarrow \neg A,\ B \Rightarrow C$; е) $\neg A \Rightarrow B,\ B \Rightarrow C,\ \neg C \Rightarrow A$.

---

[1] *В упражнениях 1-14 рассматриваются формулы логики высказываний (пропозициональные формы).*

Логическое следствие называется простым следствием, если оно является элементарной суммой, не содержащей повторяющихся слагаемых, а также не является тавтологией и является минимальной, т.е. после отбрасывания какого-нибудь из ее членов перестает быть логическим следствием из данных форм. Так, в разобранном выше примере простыми следствиями из $A$ и $A \Rightarrow B$ являются следствие $A$ и следствие $B$.

Конъюнкция всех простых следствий данной форты $A$ оказывается сокращенной к.н.ф.

7. Найти все простые следствия для пропозициональных форм задачи 6.

8. Для заданных посылок найти все их простые следствия. (Указание: найти сначала сокращенные к.н.ф. для конъюнкции посылок):

а) $A \Rightarrow \neg B,\ A \vee C,\ \neg(B \& C)$; б) $(A \& B) \Rightarrow \neg C,\ B,\ C$;

в) $(A \vee B) \& C,\ B \Rightarrow \neg C$; г) $A \equiv B,\ A \Rightarrow C \vee \neg B$;

д) $A \Rightarrow B,\ \neg B \Rightarrow C,\ B$.

Рассмотрим пример использования сокращенных к.н.ф.

В совершении некоторого поступка подозревается только одно из четырех лиц: *Л1, Л2, Л3* и *Л4*. *Л1* утверждает, что поступок совершил *Л2; Л2* утверждает, что поступок совершил *Л4; Л3* говорит, что он не совершал этого поступка, и *Л4* тоже говорит, что он этого поступка не совершал. Кто же совершил поступок, если известно, что только одно из этих утверждений истинно?

Через *A, B, C* и *D* обозначим соответственно высказывания «поступок совершил *Л1, Л2, Л3* и *Л4*». Тогда условие, что поступок мог совершить только один из четырех, запишется в виде пропозициональной формы:

$$\boldsymbol{A} = \neg(A \& B) \& \neg(A \& C) \& \neg(A \& D) \& \neg(B \& C) \& \neg(B \& D) \& \neg(C \& D).$$

которая означает, что никакие два из четырех высказываний не могут быть оба истинными. Заявление каждого из четырех означает: $B,\ D,\ \neg C$ и $\neg D$. Но так как истинно только одно из них, то, значит, никакие два из этих заявлений не являются одновременно истинными. Это условие запишется в следующем виде:

$$\boldsymbol{B} = \neg(B \& D) \& \neg(D \& \neg C) \& \neg(B \& \neg D) \& \neg(\neg D \& \neg C) \& \neg(D \& \neg D) \& \neg(\neg C \& \neg D).$$

Берем конъюнкцию посылок $\boldsymbol{A}$ и $\boldsymbol{B}$, находим для $\boldsymbol{A \& B}$ сокращенную к.н.ф.: $\neg B \& \neg D \& C \& \neg A$, которая означает, что высказывания *A, B, D* ложны, а высказывание *C*: « Поступок совершил Л3» истинно.

9. Используя условия из рассмотренного примера, узнать, кто совершил поступок, если известно, что только один из них говорит ложь.

10. С помощью метода резолюций доказать, что следующее множество дизъюнктов невыполнимо: $\neg P \vee \neg Q \vee R,\ P \vee R,\ Q \vee R,\ \neg R$.

11. Методом резолюций, используя метод исчерпания уровней, доказать, что следующее множество дизъюнктов невыполнимо:

а) $P \vee Q \vee R,\ \neg P \vee R,\ \neg Q,\ \neg R$;

б) $P \vee Q,\ \neg Q \vee R,\ \neg P \vee Q,\ \neg R$.

12. Пусть $S = \{P,Q,R,W, \neg P \vee \neg Q \vee \neg R \vee \neg W\}$. Сколько резольвент будет порождено из $S$ методом насыщения уровня до того как будет получен пустой дизъюнкт?

13. Для $S = \{P,Q,R,W, \neg P \vee \neg Q \vee \neg R \vee \neg W\}$ получить пустой дизъюнкт, используя лок-резолюцию.

14. Для $S = \{ P \vee Q \vee R, \neg P \vee R, \neg Q, \neg R\}$ получить пустой дизъюнкт, используя лок-резолюцию.

15. Найти сколемовские стандартные формы для следующих формул:

а) $\exists x \forall y \forall z(P(x,y) \Rightarrow Q(x,z))$;

б) $\exists x \exists y \forall z(P(x,y) \Rightarrow Q(x,z))$;

в) $\forall x \exists y \forall z(P(x,y) \Rightarrow Q(x,z))$;

г) $\forall x \forall y \exists z(P(x,y) \Rightarrow Q(x,z))$;

16. Найти сколемовские стандартные формы для следующих формул:

а) $\exists x \exists y( \forall z P(x,y) \Rightarrow Q(x,z))$;

б) $\forall x Q(x) \Rightarrow \exists x P(x)$;

в) $\forall x \forall y(\exists z(P(x,y) \& P(y,z)) \Rightarrow \exists v Q(x,y,v))$;

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

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

е) $\exists x( \neg(\exists y P(x,y)) \Rightarrow (\exists z Q(z) \Rightarrow R(x)))$;

ж) $\forall x \exists y( \forall z P(x,y,z) \& (\exists u Q(x,u) \Rightarrow \exists v Q(y,v)))$.

17. Определить унифицируемо ли каждое ли из следующих множеств:

а) $\{P(a), P(b)\}$; б) $\{P(x), P(f(x))\}$;

в) $\{P(a), P(f(x))\}$; г) $\{P(a,x), P(a,a)\}$;

д) $\{P(a,f(x)), P(x,y)\}$; е) $\{Q(f(a),g(x)), Q(y,y)\}$;

ж) $\{P(x,z,y), P(w,u,w), P(a,u,u)\}$.

18. Найти общий унификатор для следующего множества формул:

$\{P(a,x,f(g(y))), P(z,f(z),f(u))\}$.

19. Дано множество формул:

$\{ \forall x(P(x) \Rightarrow (Q(x) \& R(x))), \exists x(P(x) \& S(x))\}$.

Методом резолюций выяснить будет ли формула $\exists x(S(x) \& R(x))$ логическим следствием из заданного множества формул.

20. Дано множество дизъюнктов

$\{P(a), E(a), \neg S(a,y) \vee P(y), \neg P(x) \vee \neg R(x), \neg P(x) \vee \neg C(x), \neg E(x) \vee R(x) \vee S(x,f(x)),$
$\neg E(x) \vee R(x) \vee C(f(x))\}$.

Лок-резолюцией показать, что данное множество дизъюнктов не выполнимо.

21. Дано множество формул:

$\{\exists x(P(x) \& \forall y (Q(x) \Rightarrow R(x,y))), \forall x(P(x) \Rightarrow \forall y(S(y) \Rightarrow \neg R(x,y)))\}$.

Методом резолюций выяснить будет ли формула $\forall x(Q(x) \Rightarrow \neg S(x))$ логическим следствием из заданного множества формул.

22. Методом резолюций показать, что формула

$\forall x(\exists y S(y) \& V(x,y)) \Rightarrow \exists z(C(z) \& V(x,z)))$

является логическим следствием формулы $\forall y(S(y) \Rightarrow C(y))$.

23. С помощью метода резолюций установить правильность следующих силлогизмов (модусов) Аристотеля: Barbari, Ferio, Baroco. Напомним, что гласные участвующие в приведенных названиях силлогизмов определяют вид силлогизма, см. параграф 11.

24. Вводя подходящие обозначения, записать предложения, участвующие в нижеследующих выводах, в виде формул и выяснить, в каких случаях конъюнкция посылок логически влечет заключение. Результаты проиллюстрировать диаграммами Эйлера-Венна:

1). Некоторые *A* суть *B*. Ни одно *B* не есть не *C*. Следовательно, некоторые *A* суть *C*.

2). Все *A* суть *B*. Ни один *C* не есть не *B*. Следовательно, ни один *C* не есть *A*.

3). Некоторые *A* суть не *B*. Все не *C* суть *B*. Следовательно, некоторые *A* суть *C*.

4). Все *A* суть *B*. Ни один *C* не есть *B*. Следовательно, все *A* суть не *C*.

5). Некоторые не *B* суть не *A*. Ни одно не *B* не есть *C*. Следовательно, некоторые не *A* суть не *B*.

*Благо везде и повсюду зависит от соблюдения двух условий: 1) правильного установления конечной цели и 2) отыскания соответственных средств, ведущих к цели.*

*Аристотель*

# Глава 4. ДЕДУКТИВНЫЕ ТЕОРИИ

## § 1. Понятие об эффективных и полуэффективных процессах (методах)

Пусть имеем элементы некоторого класса $\mathcal{M}$, часть из которых может обладать некоторым свойством $U$. Будем считать, что задан *эффективный процесс* (метод) если: 1) есть предписание, определяющее последовательность преобразований которые надо применять одно за другим к элементу из $\mathcal{M}$; 2) если элемент $x$ из $\mathcal{M}$ задан, предписание однозначно определяет такую последовательность преобразований, что за конечное число шагов выясняем, обладает $x$ свойством $U$ или нет.

Таким образом, если задан эффективный процесс, то для любого элемента из $\mathcal{M}$ за конечное число шагов выясняем, обладает заданный элемент свойством $U$ или нет.

В отличие от эффективного процесса *полуэффективным процессом* считается некоторая процедура, которая не всегда позволяет для произвольного элемента $x$ из $\mathcal{M}$ за конечное число шагов выяснить, обладает $x$ свойством $U$ или нет. Точнее будем считать, что задан полуэффективный процесс (метод) если выполняется вышеуказанное условие 1), а вместо 2) следующее условие: если элемент $x$ из $\mathcal{M}$ задан, предписание однозначно определяет такую последовательность преобразований, что если $x$ обладает свойством $U$, то за конечное число шагов это выясняем, если же $x$ не обладает свойством $U$, то, возможно, мы не сможем это выяснить за конечное число шагов.

Пример эффективной процедуры. Пусть $\mathcal{M}=\{0,1,2,3,...\}$ . Число $n$ может быть квадратом какого-либо числа из $\mathcal{M}$, назовем это свойством $U$. Эффективной процедурой для выяснения обладает ли элемент $x$ из $\mathcal{M}$ свойством $U$ может быть следующая. Берем числа $1,...,x-1$ и возводя их в квадраты смотрим равны они $x$ либо нет. Очевидно, что таким образом мы всегда сможем выяснить для любого $x$ за конечное число шагов обладает $x$ свойством $U$ либо нет.

Теперь рассмотрим пример полуэффективной процедуры.

Рассмотрим известный процесс извлечения квадратного корня из положительных действительных чисел:

$\sqrt{328} = 18{,}1\ldots$

$1$

$28 \mid 228$

$8 \mid 224$

$361 \mid 400$

$1 \mid 361$

.....

Ясно, что если для заданного числа $a$ число $\sqrt{a}$ является рациональным, то рано или поздно заметим период. Если же $\sqrt{a}$ не является рациональным числом, то, сколько бы долго ни продолжались эти вычисления, мы не сможем на основе этих вычислений сказать, будет когда-нибудь период или нет, т.е. является $\sqrt{a}$ рациональным или нет. Таким образом, эта процедура будет полуэффективной процедурой для выяснения рациональности $\sqrt{a}$.

*Отыщи всему начало, и ты много поймешь.*
*Козьма Прутков*

## § 2. Дедуктивные теории

Дедукция (от латинского deductic - выведение) - форма мышления, когда заключение выводится чисто логическим путем (т.е. по правилам логики) из некоторых данных посылок.

Индукция (от латинского inductio - наведение) - форма мышления, посредством которой от некоторых фактов или истинных высказываний переходят к некоторой гипотезе (общему утверждению).

Примеры дедуктивных рассуждений.

1. Все люди смертны. Сократ – человек. Следовательно, Сократ смертен.

2. 5> 3, 3 >1, следовательно, 5 >1.

Примеры индуктивных рассуждений.

1. График функции $y=2x+3$ прямая, график функции $y=3x+1$ прямая, следовательно, функции вида $y=kx+b$ имеют графиком прямую. Полученная здесь гипотеза оказывается истинной.

2. Рассмотрим предположение Ферма, что число $p= 2^{2^n}+1$ является простым для всех $n$. При $n=0,1,2,3,4$ получим, что $p$ равно $3, 5, 17, 257, 65537$ и все они простые числа. Но Эйлер показал, что для $n=5$ $p=$ 4 294 967 297 и это число является составным (делится на 641). Следовательно, это предположение Ферма не верно.

2. Возьмем формулу Эйлера $N=x^2+x+41$. При каждом $x=1,2,3,...,39$ число $N$ является простым, следовательно, числа $N$, указанного вида, являются простыми. Сформулированная здесь гипотеза неверна, например, при $x=40$ число $N=41^2$, следовательно, оно не является простым.

Таким образом, заключение, полученное дедуктивным способом, уже не нуждается в доказательстве. Заключение, полученное индуктивным способом, требует доказательства его истинности. Будем рассматривать дедуктивные методы. Введем понятие дедуктивной теории.

*Дедуктивная теория* считается заданной, если задан язык этой теории и из множества правильно построенных выражений (предложений, называемых формулами) языка выделено дедуктивным образом множество теорем. Подробнее, дедуктивная теория считается заданной, если:

1). Задан алфавит и правила образования выражений (слов) в этом алфавите.

2). Заданы правила образования формул (правильно построенных выражений) языка.

3). Из множества всех формул языка выделено некоторым дедуктивным способом (который будет описан ниже) подмножество $T$, элементы которого будем называть теоремами. В зависимости от того, как задано это подмножество $T$, будем различать получающиеся при этом дедуктивные теории.

Подмножество $T$ может задаваться одним из следующих способов.

I. Задаются аксиомы и конечное число правил выводов, т.е.

а) из множества формул (правильно построенных выражений) выделяется подмножество $A$, элементы которого называются аксиомами (аксиом может быть как конечное число, так и бесконечное),

б) задается конечное число правил выводов, используя которые, и только их, из аксиом можно некоторым образом получать теоремы (подробнее этот вопрос будет изучаться в следующих параграфах).

Если теоремы заданы указанным образом, т.е. заданием аксиом и конечного числа правил вывода, то эта дедуктивная теория называется *формальной аксиоматической теорией* или *формальным (логическим) исчислением.*

Особо отметим, что аксиомы лишь задаются, поэтому их часто называют скрытыми определениями. Бытующее мнение, согласно которому аксиомы принимаются без доказательств, не совсем точно передает суть дела.

***Аксиомы не доказываются не потому, что могут не доказываться, а потому, что не могут быть доказаны.***

II. Задаются только аксиомы, а правила вывода считаются известными, т.е.:

а) из множества формул (правильно построенных выражений) выделяется подмножество $A$, элементы которого называются аксиомами (аксиом может быть как конечное число, так и бесконечное),

б) правила вывода (методы доказательства) теорем считаются известными из опыта изучения математики.

При таком задании теорем дедуктивной теории, говорим, что задана *полуформальная аксиоматическая теория*.

III. Аксиом нет, а задается только конечное число правил выводов, с помощью которых и получают теоремы. Такую дедуктивную теорию называют *теорией естественного вывода*.

Случай, когда нет аксиом и нет правил вывода, не рассматривается в логике.

Применяя один из указанных выше способов задания теорем, будем получать множества теорем $T_1$, $T_2$ и $T_3$ соответственно. Сразу возникают вопросы: когда эти множества $T_1$, $T_2$, и $T_3$ совпадают? Когда некоторые из $T_1$, или $T_2$, или $T_3$ дедуктивной теории $\boldsymbol{B}$ совпадают или покрывают класс "истинных" формул теории $\boldsymbol{B}_1$, при условии совпадения для $\boldsymbol{B}$ и $\boldsymbol{B}_1$, алфавитов и формул. Эти вопросы оказываются не всегда простыми и в рамках этого курса затрагиваются незначительно .

*Разве может леопард избавиться от пятен.*
*Английская пословица*

## § 3. Свойства дедуктивных теорий

Выберем один из трех способов задания теорем дедуктивной теории. Изменяя аксиомы или правила вывода (в случае, когда они задаются), можно получать различные множества теорем $T$. Это множество $T$ - множество теорем (множество доказуемых формул) является существенной характеристикой дедуктивной теории.

Может оказаться, что множество теорем $T$ покрывает все множество формул (правильно построенных выражений) теории. Иначе, это означает, что доказуема любая формула (правильно построенное выражение) и если в теории есть отрицание, то из доказуемости какой-то формулы тут же следует доказуемость ее отрицания. Следовательно, в этом случае доказуем какой-то факт и его отрицание. Ясно, что теории, в которых можно доказать, что угодно, не представляют интерес с многих позиций.

Дедуктивные теории, в которых множество теорем покрывает все множество формул (правильно построенных выражений) называются *противоречивыми*, в противном случае – *непротиворечивыми*. Выяснение непротиворечивости дедуктивной теории является одной из важнейших проблем. К сожалению, эта проблема оказывается и одной из очень сложных.

Пусть множество теорем $T$ является частью, не совпадающей со всем множеством формул (правильно построенных выражений) $\Phi$, т.е. наша дедуктивная теория непротиворечива. Тогда можно уже интересоваться, а какую часть $\Phi$ занимают теоремы. Для этого вводят свойство полноты теории. Свойство полноты дедуктивной теории характеризует достаточность теорем для каких-то целей. В зависимости от того, для каких целей должно быть достаточно теорем, будем в дальнейшем вводить различные понятия полноты.

Рассмотренные свойства - непротиворечивость и полнота, являются важнейшими свойствами дедуктивной теории. Кроме этих свойств, имеется и ряд других свойств. Рассмотрим еще два свойства дедуктивной теории.

*Независимость аксиом* теории. Отдельная аксиома дедуктивной теории, называется независимой, если эту аксиому нельзя вывести в этой теории из остальных аксиом. Система аксиом называется независимой, если каждую из них нельзя вывести из остальных.

*Разрешимость теории*. Дедуктивная теория называется разрешимой, если в этой теории понятие теоремы эффективно, т.е. существует правило (метод), позволяющее для произвольной формулы за конечное число действий выяснить, является она теоремой или нет.

Пусть заданы две дедуктивные теории $\boldsymbol{B}_1$ и $\boldsymbol{B}_2$ такие, что:

1) алфавит теории $\boldsymbol{B}_1$ содержится в алфавите теории $\boldsymbol{B}_2$ или эти алфавиты совпадают,

2) каждая формула из $\boldsymbol{B}_1$ является формулой из $\boldsymbol{B}_2$

3) каждая теорема из $\boldsymbol{B}_1$ является теоремой в $\boldsymbol{B}_2$

При выполнении этих условий говорят, что теория $\boldsymbol{B}_2$ является расширением теории $\boldsymbol{B}_1$.

В следующих разделах изучим более подробно каждую из дедуктивных теорий (полуформальную и формальную аксиоматическую теорию, естественный вывод), их свойства, а также примеры таких теорий.

*Очень трудно в каждой науке отобрать и расположить в надлежащем порядке элементы, из которых все дальнейшее следует. И во всем этом система элементов Евклида превосходит все остальное, … .*

*Прокл (около 410-485 гг.)*

*А открытие неевклидовой геометрии было величайшей революцией в области человеческой мысли, какую только знает история науки. Inde irae*[1].

*В.Ф. Коган*

## § 4. Пример полуформальной аксиоматической теории - геометрия

Согласно **§ 2** полуформальная аксиоматическая теория считается заданной, если задан язык этой теории и из множества предложений (формул) этого языка выделено подмножество - множество аксиом. Таким образом, полуформальная аксиоматическая теория (теория $\boldsymbol{B}$ ) считается заданной, если:

1) задан алфавит $A$ - алфавит теории $\boldsymbol{B}$ и заданы правила образования выражений (слов) теории $\boldsymbol{B}$;

2) заданы правила образования правильно построенных выражений (формул) теории $\boldsymbol{B}$;

---

[1] *«Отсюда гнев» (Ювенал)*

3) из множества правильно построенных выражений выделяется некоторое подмножество - множество аксиом теории ***B***.

В настоящее время многие математические теории строятся (задаются) в виде полуформальных аксиоматических теорий. Однако при этом построение их не доводится до того вида, какого требуют указанные выше п.п.1) - 3). Во многих случаях, алфавит не перечисляется, кроме того, не задаются правила образования слов и правильно построенных выражений, а считается, что мы в состоянии отличить, является ли произвольное предложение правильно построенным выражением теории. Например, предложение "в равнобедренном треугольнике углы при основании равны" отнесем к правильно построенным выражениям геометрии, а "треугольник - зеленый" не отнесем к таковым, хотя правила построения правильно построенных выражений геометрии и не сформулированы. Рассмотрим именно такой пример полуформальной аксиоматической теории, когда ее "строгость" не доведена до требований 1) - 3). Однако, как будет видно из построения, эту теорию можно построить и согласно требованиям 1) - 3).

Прежде чем задать геометрию в виде полуформальной аксиоматической теории, нужно отметить следующее.

Геометрия вначале развивалась как эмпирическая наука и в ранний период достигла особо высокого уровня развития в Египте. В первом тысячелетии до н. э. греческие геометры не только обогатили геометрию многочисленными новыми фактами, но и предприняли также серьезные шаги к строгому ее логическому обоснованию.

Многовековая работа греческих геометров за этот период была подытожена и систематизирована Евклидом (330 - 275 гг. до н. э.) в его знаменательном труде "Начала". На протяжении более чем 20 веков "Начала" Евклида служили образцом ясного и строгого изложения. Кризис основ математики, в частности, и создание неевклидовых геометрий, вынудил пересмотреть основы геометрии, и отказаться от понятия аксиомы как самоочевидной истины. Выяснилось, что интуиция во многих случаях может "подводить", приводить к неприятностям (противоречиям). Поэтому пришлось, по возможности, отказаться от попыток обращения к интуиции при построении геометрии и ввести как аксиомы все свойства и "очевидные" положения, которые Евклид использовал в геометрии, но не вводил их в постулаты и аксиомы. Доведение строгости геометрии до некоторых современных понятий строгости было завершено в работах Паша (1882г.) и Гильберта (1899г.).

В настоящее время имеются различные редакции задания геометрии. Введем задание геометрии (аксиоматику Гильберта) согласно работе [11]. Геометрия считается заданной, если:

1. Задано три различных множества:

а) элементы первого множества называются точками и обозначаются через $A,B,C,...,A_1, A_2,..., B_1, B_2,..., C_1, C_2,...;$

б) элементы второго множества называются прямыми и обозначаются через $a, b,c,..., a_1, a_2,..., b_1, b_2,..., c_1, c_2,...;$

в) элементы третьего множества называются плоскостями и обозначаются через $\alpha, \beta, \gamma, \ldots, \alpha_1, \alpha_2, \ldots, \beta_1, \beta_2, \ldots, \gamma_1, \gamma_2, \ldots$ .

Множество всех точек, прямых и плоскостей называется пространством. На множестве точек, прямых и плоскостей введены отношения, обозначаемые словом "лежат", "между" и "конгруэнтно".

2. Считается, что мы в состоянии различить, является ли данная последовательность выражений правильно построенным выражением геометрии или нет.

3. "Точки", "прямые", "плоскости" и отношения между ними, обозначаемые словами "лежат", "между" и "конгруэнтно", подчиняются перечисляемым ниже аксиомам, во всем остальном природа их произвольна. Подчеркнем еще раз, что под "точками", "прямыми" и "плоскостями" можно понимать любые объекты, а под словами "лежат", "между" и "конгруэнтно" любые отношения между объектами, лишь бы для них выполнялись перечисленные ниже аксиомы.

Будем употреблять такие термины как "отрезок", "прямая", "проходить через точку" и т.п. не вводя их определений. Считаем, что читатель это сделает сам, либо обратится к курсу геометрии.

Теперь зададим аксиомы. Все аксиомы подразделяются на 5 групп.

Первая группа - *аксиомы связи*.

1). Каковы бы ни были две точки *A, B,* существует прямая *a*, проходящая через каждую из точек *A, B*.

2). Каковы бы ни были две различные точки *A, B*, существует не более одной прямой, которая проходит через каждую из точек *A, B*.

3). На каждой прямой лежат по крайней мере две точки. Существуют по крайней мере три точки, не лежащие на одной прямой.

4). Каковы бы ни были три точки *A, B, C*, не лежащие на одной прямой, существует плоскость $\alpha$ , проходящая через каждую из трех точек *A, B, C*. На каждой плоскости лежит хотя бы одна точка.

5). Каковы бы ни были три точки *A, B, C*, не лежащие на одной прямой, существует не более одной плоскости, которая проходит через каждую из трех точек *A, B, C*.

6). Если две точки *A, B* прямой *a* лежат на плоскости $\alpha$ , то каждая точка прямой *a* лежит на плоскости $\alpha$ .

7). Если две плоскости $\alpha, \beta$ имеют общую точку *A*, то они имеют еще по крайней мере одну общую точку *B*.

8). Существуют, по крайней мере, четыре точки, не лежащие в одной плоскости.

Вторая группа - *аксиомы порядка*.

1). Если точка *B* лежит между точкой *A* и точкой *C*, то *A, B, C* - различные точки одной прямой *b* и точка *B* лежит также между *C* и *A*.

2). Каковы бы ни были точки *A* и *C*, существует по крайней мере одна точка *B* по прямой *AC* такая, что *C* лежит между *A* и *B*.

3). Среди любых трех точек прямой существует не более одной точки, лежащей между двумя другими.

4). Аксиома Паша. Пусть *A, B, C* - три точки, не лежащие на одной прямой, и *a* - некоторая прямая в плоскости *ABC*, не содержащая ни одной из точек *A, B, C*. Тогда, если прямая *a* проходит через точку отрезка *AB*, то она проходит также либо через точку отрезка *AC*, либо через точку отрезка *BC*.

Кроме этих групп аксиом, задается еще 3-я группа - *аксиомы конгруэнтности* (5 аксиом), 4-я группа - *аксиомы непрерывности* (2 аксиомы) и 5-я группа - *аксиома параллельности*. Здесь не приводятся все эти аксиомы, ибо в наши цели входит не изучение геометрии, а только рассмотрение, каким образом (методом) задается геометрия.

Однако все же приведем аксиому параллельности как для евклидовой геометрии (аксиома Евклида), так и для неевклидовой геометрии (аксиома Лобачевского).

*Аксиома Евклида*. Пусть *a* - произвольная прямая и *A* - точка, лежащая вне прямой *a*, тогда в плоскости, определенной точкой *A* и прямой *a*, можно провести не более одной прямой, проходящей через *A* и не пересекающей *a*.

*Аксиома Лобачевского*. Пусть *a* - произвольная прямая и *A* - точка, лежащая вне прямой *a*, тогда в плоскости, определенной точкой *A* и прямой *a*, можно провести не менее двух прямых, не пересекающихся с заданной прямой *a*.

Если примем первые четыре группы аксиом и аксиому Евклида, то получим евклидову геометрию.

Если примем первые четыре группы аксиом и аксиому Лобачевского, то получим неевклидову геометрию (геометрию Лобачевского - Бойяи - Гаусса).

Из аксиом логическими методами уже получаются теоремы геометрии, однако этим заниматься не будем. Еще раз отметим, что в полуформальной аксиоматической теории система логических правил, т.е. методы доказательств, считаются известными из опыта изучения математики.

В математической практике аксиоматические теории обычно описываются, как и геометрия, в виде полуформальных теорий и предполагается, что логика, используемая в этой теории, есть та интуитивная логика, которая усваивается в ходе изучения математики.

*Для самой науки надобно было всегда желать, чтобы она стала на твёрдом основании, чтобы строгость и ясность сохранялись в самих её началах.*

*Н. Лобачевский*

## § 5. Формальные аксиоматические теории

Как уже известно, формальная аксиоматическая теория ***B*** считается заданной, если:

1. Задано некоторое множество символов - алфавит теории $\boldsymbol{B}$. Конечная последовательность букв алфавита называется выражением теории. Алфавит, следовательно, и выражения теории, задаются эффективным образом.

2. Заданы формулы теории $\boldsymbol{B}$ как некоторое подмножество выражений теории. Формулы тоже обычно задаются эффективным образом.

3. Заданы аксиомы теории $\boldsymbol{B}$ как подмножество множества формул. Если аксиом конечное число, то их можно задать перечислением. Если же их бесконечное множество, то задают с помощью схем, т.е. правил построения аксиом. Если аксиомы заданы эффективным образом, то теория $\boldsymbol{B}$ называется *эффективно аксиоматизированной*.

4. Задано конечное число правил вывода $R_1$, $R_2$,..., $R_n$, согласно каждому из которых некоторая формула, именуемая *непосредственным следствием* (заключением), непосредственно выводима из некоторого конечного множества формул, называемых *посылками*. При этом для каждого $R_i$ существует целое положительное $k$ такое, что для каждого множества, состоящего из $k$ формул и для каждой формулы $\boldsymbol{A}$ эффективно решается вопрос о том, является ли $\boldsymbol{A}$ непосредственным следствием данных $k$ формул по правилу $R_i$.

*Выводом* в $\boldsymbol{B}$ называется всякая последовательность $A_1$, $A_2$,...,$A_n$ формул, такая, что для каждого $i$ $(1 \le i \le n)$ формула $A_i$ есть либо аксиома теории $\boldsymbol{B}$, либо непосредственное следствие каких-либо предыдущих формул этой последовательности по одному из правил вывода.

Формула $A$ теории $\boldsymbol{B}$ называется *теоремой* теории $\boldsymbol{B}$ если существует вывод в $\boldsymbol{B}$, в котором последней формулой является $A$, такой вывод называется выводом формулы $A$.

Формула $A$ называется *следствием* в $\boldsymbol{B}$ множества формул $G$ тогда и только тогда, когда существует такая последовательность формул $A_1, A_2,..., A_n$, что $A_n=A$ и для любого $i$ $A_i$ есть либо аксиома, либо элемент $G$, либо непосредственное следствие некоторых предыдущих формул этой последовательности по одному из правил вывода. Такая последовательность называется *выводом $A$ из $G$*. Элементы $G$ называются *гипотезами* или *посылками вывода*. Для сокращения утверждения «$A$ есть следствие $G$» будем употреблять запись: $G \vdash A$ . Например, если $G=\{B_1,B_2,...B_m\}$, то будем писать

$$B_1,B_2,...,B_m \vdash A.$$

Нетрудно видеть, что если $G$ есть пустое множество, т. е. $G = \varnothing$ то $\varnothing \vdash A$ имеет место тогда и только тогда, когда $A$ является теоремой. Вместо $\varnothing \vdash A$ принято писать просто

$$\vdash A,$$

что читается: «формула $A$ является теоремой».

Чтобы избежать путаницы там, где будут рассматриваться не одна, а несколько теорий, употребляют запись

$$G \underset{\boldsymbol{B}}{\vdash} A \qquad \text{и} \qquad \underset{\boldsymbol{B}}{\vdash} A$$

указывая индексом $\boldsymbol{B}$ на то, о какой теории идет речь.

## § 6. Свойства выводимости

Пусть $G$ - некоторое множество формул данной теории, $A$, $B$ и $C$ - произвольные формулы той же теории. Рассмотрим некоторые свойства выводимости в формальных аксиоматических теориях.

1. Если $G$ содержится в некотором множестве формул $F$ и если $G \vdash A$, то $F \vdash A$.

Доказательство**.** Пусть $A$ имеет вывод

$$A_1, A_2, \ldots, A_n \qquad (1)$$

из гипотез $G$. Если некоторая формула $A_i$ принадлежит $G$, то, очевидно, $A_i \in F$. Следовательно, вывод (1) формулы $A$ является выводом формулы $A$ из гипотез $F$. Что и требовалось доказать.

2. $G \vdash A$ тогда и только тогда, когда в $G$ существует конечное подмножество $H$ такое, что $H \vdash A$.

Доказательство следует из определения вывода.

3. Пусть $G \vdash A$ и каждая формула $B$, принадлежащая $G$, выводима из некоторого множества формул $F$, тогда $F \vdash A$.

Доказательство. Пусть $A$ имеет вывод

$$A_1, A_2, \ldots, A_n \qquad (2)$$

из гипотез $G$. По определению вывода некоторые $A_i$ из (2) могут принадлежать $G$, но каждая формула из $G$, имеет вывод из $F$. Заменим в (2) все $A_i$, принадлежащие $G$, выводом $A_i$ из $F$. В результате получим последовательность формул:

$$B_1, B_2, \ldots, B_m,$$

которая уже является выводом $A$ из $F$. Что и требовалось доказать.

Как частный случай п.3 имеем:

3` . Если $A \vdash B$ и $B \vdash C$, то $A \vdash C$.

4. Если $G, A \vdash B$ и $G \vdash A$, то $G \vdash B$.

Доказательство. Пусть $B$ имеет вывод

$$B_1, B_2, \ldots, B_n \qquad (3)$$

из гипотез $G$ и $A$, а формула $A$ имеет вывод

$$A_1, A_2, \ldots, A_m \qquad (4)$$

из гипотез $G$. В выводе (3) формулы $B$ некоторые из $B_i$ могут быть равны $A$. Заменим такие $B_i$ последовательностью (4). В результате получим последовательность формул

$$C_1, C_2, \ldots, C_r,$$

которая является выводом для $B$ из гипотез $G$. Что и требовалось доказать.

*Первые понятия, с которых начинается какая-нибудь наука, должны быть ясны и приведены к самому меньшему числу. Тогда только они могут служить прочным и достаточным основанием учения.*

*Н. Лобачевский*

## § 7. Исчисление высказываний (теория *L*)

В качестве первого примера формальной аксиоматической теории рассмотрим исчисление высказываний - теорию *L*.

1. *Символами теории L* являются $\neg, \Rightarrow, ), ($ и буквы $A_i$ с целыми положительными числами в качестве индексов: $A_1, A_2, A_3, \ldots$ Символы $\neg, \Rightarrow$ будем называть *примитивными связками*, а $A_1, A_2, A_3, \ldots$ - пропозициональными буквами.

2. *Формулы теории L* определим индуктивным образом:

1) все буквы $A_1, A_2, A_3, \ldots$ суть формулы;

2) если $\boldsymbol{A}$ и $\boldsymbol{B}$ формулы, то $(\neg \boldsymbol{A})$ и $(\boldsymbol{A} \Rightarrow \boldsymbol{B})$ тоже формулы;

3) выражение теории *L* является формулой только тогда, когда это следует из 1) и 2).

Будем считать, что

$A \& B$ служит обозначением для формулы $(\neg (A \Rightarrow (\neg B)))$,

$A \vee B$ служит обозначением для формулы $((\neg A) \Rightarrow B)$,

$A \equiv B$ служит обозначением для формулы $(\neg ((A \Rightarrow B) \Rightarrow (\neg (B \Rightarrow A))))$.

Из определения формул видно, что всякая формула из *L* есть пропозициональная форма, построенная из пропозициональных букв $A_1, A_2 \ldots$ с помощью связок $\neg$ и $\Rightarrow$.

Будем придерживаться тех же правил опускания скобок в формулах, что и раньше для пропозициональных форм.

3. Аксиомы теории *L*. Каковы бы ни были формулы *A, B* и *C* теории *L*, следующие формулы суть *аксиомы теории L*:

*A1:* $A \Rightarrow (B \Rightarrow A)$;

*A2:* $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$;

*A3:* $(\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)$.

Заметим, что *A1 - A3* являются схемами аксиом, т.е. указывают, как строятся аксиомы для произвольных формул *A, B* и *C*. В силу произвольности формул *A, B* и *C* схемы аксиом *A1-A3* порождают бесчисленное множество аксиом. Легко убедиться, например, с помощью таблиц истинности, что каждая аксиома полученная по схемам *A1-A3* является тавтологией.

4. Единственным *правилом вывода теории L* служит правило *modus ponens*: *B* есть непосредственное следствие *A* и $A \Rightarrow B$. Это правило сокращенно обозначают *MP*.

*Modus ponens* в переводе означает «правило отделения». Отметим, что теорема 1.1 утверждает, что если *A* и $A \Rightarrow B$ тавтологии, то и *B* тоже тавтология. Следовательно, правило *modus ponens* из тавтологий получает тавтологию.

Очевидно, правило *MP* означает, что $A, A \Rightarrow B \vdash B$.

Таким образом, задали некоторую формальную аксиоматическую теорию, которая и называется исчислением высказываний. Рассмотрим некоторые доказательства в этой теории.

*То, что начала существуют, необходимо принять, прочее следует доказать.*
*Аристотель*

## § 8. Некоторые теоремы исчисления высказываний

Проведем доказательство некоторых теорем исчисления высказываний (теории *L*).

**Лемма 4.1**. $\vdash A \Rightarrow A$ для любой формулы *A* теории *L*. Иначе, формула $A \Rightarrow A$ является теоремой теории *L* для любой формулы *A* из *L*.

**Доказательство.**

1) $(A \Rightarrow ((A \Rightarrow A) \Rightarrow A)) \Rightarrow ((A \Rightarrow (A \Rightarrow A)) \Rightarrow (A \Rightarrow A))$ - является аксиомой, так как получается по схеме *A2*, если положить $B = A \Rightarrow A$ и $C = A$,

2) $A \Rightarrow ((A \Rightarrow A) \Rightarrow A)$ является аксиомой, которая получается по схеме *A1*, если положить $B = A \Rightarrow A$,

3) $(A \Rightarrow (A \Rightarrow A)) \Rightarrow (A \Rightarrow A)$ является непосредственным следствием из 1) и 2) по *MP*,

4) $A \Rightarrow (A \Rightarrow A)$ является аксиомой, полученной по схеме *A1* при $B = A$,

5) $A \Rightarrow A$ является непосредственным следствием из 3) и 4) по *MP*.

Таким образом, получили последовательность формул 1)-5), каждая из которых аксиома либо непосредственное следствие из некоторых предыдущих формул 1)-4) по правилу *MP*, причем последняя формула есть $A \Rightarrow A$. Следовательно, формулы 1)- 5) - есть вывод формулы $A \Rightarrow A$ и $A \Rightarrow A$ является теоремой в *L*.

**Теорема 4.1 (теорема дедукции)**. Если $\Gamma$ - множество формул, *A,B* - формулы и $\Gamma, A \vdash B$, то $\Gamma \vdash A \Rightarrow B$. В частности, если $A \vdash B$, то $\vdash A \Rightarrow B$.

**Доказательство**. Пусть $B_1, B_2, \ldots, B_n$ есть вывод формулы *B* из $\Gamma \cup \{A\}$, где $B_n = B$. Индукцией по $i$ $(1 \le i \le n)$ докажем, что $\Gamma \vdash A \Rightarrow B_i$.

Пусть $i=1$. Покажем, что $\Gamma \vdash A \Rightarrow B_1$. Так как $B_1$ является первой из формул в выводе *B* из $\Gamma \cup \{A\}$, то имеем следующие возможности:

$B_1 \in \Gamma$,

$B_1$ - является аксиомой,

$B_1=A$.

По схеме аксиом *A1* формула $B_1 \Rightarrow (A \Rightarrow B_1)$ есть аксиома. Поэтому в первых двух случаях (когда $B_1$ аксиома или формула из $\Gamma$) по *MP* получим

$\Gamma \vdash A \Rightarrow B_1$.

В третьем случае, т.е. когда $B_1$ совпадает с $A$ ($B_1=A$), по лемме 4.1 имеем

$\vdash A \Rightarrow B_1$,

следовательно, $\Gamma \vdash A \Rightarrow B_1$. Тем самым случай $i=1$ исчерпан.

Допустим теперь, что $\Gamma \vdash A \Rightarrow B_k$ для любого $k<i$. Для $B_i$ имеем четыре возможности:

$B_i$ есть аксиома,

$B_i \in \Gamma$,

$B_i = A$,

$B_i$ - следствие по *MP* из некоторых $B_j$ и $B_m$, где $j<i$, $m<i$ и $B_m$ имеет вид $B_j \Rightarrow B_i$.

В первых трех случаях то, что $\Gamma \vdash A \Rightarrow B_i$ доказывается так же, как для $i=1$. В последнем случае применим индуктивное предположение, согласно которому

1) $\Gamma \vdash A \Rightarrow B_j$,

2) $\Gamma \vdash A \Rightarrow (B_j \Rightarrow B_i)$.

По схеме аксиом *A2*:

3) $\vdash (A \Rightarrow (B_j \Rightarrow B_i)) \Rightarrow ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i))$;

далее применяя правило *MP*, из 3) и 2) получим:

4) $\Gamma \vdash (A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i)$;

снова по *MP* из 4) и 1) имеем:

$\Gamma \vdash A \Rightarrow B_i$.

Таким образом, доказательство по индукции завершено и для $i=n$ получено требуемое утверждение. Теорема доказана.

***Следствие 4.1***. $A \Rightarrow B, B \Rightarrow C \vdash A \Rightarrow C$.

**Доказательство.**

1) $A \Rightarrow B$ - гипотеза,

2) $B \Rightarrow C$ - гипотеза,

3) $A$ - гипотеза,

4) $B$ - по *MP* из 1) и 3),

5) $C$ - по *MP* из 2) и 4).

Таким образом, $A \Rightarrow B, B \Rightarrow C, A \vdash C$. Отсюда по теореме дедукции $A \Rightarrow B, B \Rightarrow C \vdash A \Rightarrow C$. Что и требовалось доказать.

Отметим, что по доказанному следствию имеем:

$\neg A \Rightarrow B, B \Rightarrow C \vdash \neg A \Rightarrow C$

исключив импликацию, получим:

$A \vee B, \neg B \vee C \vdash A \vee C$,

т.е. **бинарная резольвента дизъюнктов $A \vee B$ и $\neg B \vee C$ выводима из этих дизъюнктов в теории *L*.**

.

***Следствие 4.2.*** $A \Rightarrow (B \Rightarrow C), B \vdash A \Rightarrow C$.

**Доказательство**.

1) $A \Rightarrow (B \Rightarrow C)$ - гипотеза,

2) $A$ - гипотеза,

3) $B \Rightarrow C$ - по *MP* из 1) и 2),

4) $B$ - гипотеза,

5) $C$ - по *MP* из 3) и 4).

Таким образом, $A \Rightarrow (B \Rightarrow C), B, A \vdash C$, тогда по теореме дедукции $A \Rightarrow (B \Rightarrow C), B \vdash A \Rightarrow C$. Что и требовалось доказать.

**Лемма 4.2.** Для любых формул $A,B$ следующие формулы являются теоремами в $L$:

а) $\neg\neg B \Rightarrow B$; б) $B \Rightarrow \neg\neg B$;

в) $\neg A \Rightarrow (A \Rightarrow B)$; г) $(\neg B \Rightarrow \neg A) \Rightarrow (A \Rightarrow B)$;

д) $(A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A)$; е) $A \Rightarrow (\neg B \Rightarrow \neg(A \Rightarrow B))$;

ж) $(A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)$.

**Доказательство.**

а) $\vdash \neg\neg B \Rightarrow B$.

1) $(\neg B \Rightarrow \neg\neg B) \Rightarrow ((\neg B \Rightarrow \neg B) \Rightarrow B)$ - является аксиомой, полученной по схеме *A3* при $A = \neg B$,

2) $\neg B \Rightarrow \neg B$ - является теоремой в силу леммы 4.1, т.е. эта формула выводима и при желании можно выписать весь вывод этой формулы,

3) $(\neg B \Rightarrow \neg\neg B) \Rightarrow B$ - по следствию 4.2 из 1) и 2),

4) $\neg\neg B \Rightarrow (\neg B \Rightarrow \neg\neg B)$ - является аксиомой, полученной по схеме *A1*, когда вместо $A$ и $B$ взяты формулы $\neg\neg B$ и $\neg B$ соответственно,

5) $\neg\neg B \Rightarrow B$ - по следствию 4.1 из 3) и 4). Утверждение а) доказано.

б) $\vdash B \Rightarrow \neg\neg B$.

1) $(\neg\neg\neg B \Rightarrow \neg B) \Rightarrow ((\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B$ - является аксиомой, полученной по *A3*, когда вместо $A$ и $B$ взяты формулы $B$ и $\neg\neg B$ соответственно,

2) $\neg\neg\neg B \Rightarrow \neg B$ - теорема согласно п. а),

3) $(\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B$ - по *MP* из 1) и 2),

4) $B \Rightarrow (\neg\neg\neg B \Rightarrow B)$ - аксиома, полученная по схеме *A1*, когда вместо $A$ и $B$ взяты формулы $B$ и $\neg\neg\neg B$ соответственно,

5) $B \Rightarrow \neg\neg B$ - согласно следствию 4.1 из 3) и 4).

Утверждение б) доказано.

в) $\vdash \neg A \Rightarrow (A \Rightarrow B)$

1) $\neg A$ - гипотеза,

2) $A$ - гипотеза,

3) $A \Rightarrow (\neg B \Rightarrow A)$ - схема аксиом $A1$,

4) $\neg A \Rightarrow (\neg B \Rightarrow \neg A)$ - схема аксиом $A1$,

5) $\neg B \Rightarrow A$ - по $MP$ из 2) и 3),

6) $\neg B \Rightarrow \neg A$ - по $MP$ из 1) и 2),

7) $(\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)$ - схема аксиом $A3$,

8) $(\neg B \Rightarrow A) \Rightarrow B$ - по $MP$ из 6) и 7),

9) $B$ - по $MP$ из 5) и 8).

Итак, в силу 1) - 9) $\neg A, A \vdash B$. Тогда по теореме дедукции $\neg A \vdash A \Rightarrow B$ и, снова по той же теореме, получим, что $\vdash \neg A \Rightarrow (A \Rightarrow B)$.

Доказательство пунктов г) - ж) оставляем читателю в качестве нетривиальных упражнений.

## § 9. Эквивалентность двух определений непротиворечивости

Напомним, что дедуктивная теория называется противоречивой, если ее множество теорем $T$ совпадает со всем множеством ее формул $\Phi$ ($T=\Phi$), и непротиворечивой в противном случае $((T \subset \Phi) \ \& \ (T \neq \Phi))$. Будем считать это первым определением (не) противоречивости.

Второе определение (не) противоречивости. Дедуктивная теория называется противоречивой, если существует формула $A$ такая, что в этой теории выводимо $A$ и выводимо $\neg A$; если такой формулы не существует, то теория называется непротиворечивой.

**Теорема 4.2**. Для теорий, содержащих исчисление высказываний, приведенные определения (не) противоречивости эквивалентны.

**Доказательство**. Покажем, что из второго определения следует первое. Пусть существует формула $A$, такая, что выводима $A$ и $\neg A$, т.е.

$$\vdash A \qquad (1)$$

$$\vdash \neg A. \qquad (2)$$

В лемме 2 (в) доказано, что для любых формул $A$ и $B$ имеет место:

$$\vdash \neg A \Rightarrow (A \Rightarrow B). \qquad (3)$$

Из (2) и (3) по $MP$ получаем

$$\vdash A \Rightarrow B, \qquad (4)$$

а из (4) и (1) снова по *MP* имеем: $\vdash B$. Последнее и означает, что доказуема любая формула $B$, т.е. множество теорем совпадает с множество формул.

Если примем первое определение, то в противоречивой теории доказуема любая формула, т. е. для любой формулы $A$ получим, что $A$ и $\neg A$ являются теоремами, следовательно, теория противоречива согласно второго определения.

## § 10. Производные (доказуемые) правила вывода в исчислении высказываний

В исчислении высказываний (теории $L$) имеется только одно исходное правило вывода: *modus ponens* (*MP*). При этом доказана теорема дедукции: если $G, A \vdash B$, то $G \vdash A \Rightarrow B$. Последнее представляет тоже некоторое правило вывода, но уже производное (доказуемое) правило, получающееся, если принять правило *MP*. Кроме этого производного правила вывода, оказывается, есть и другие. Рассмотрим некоторые из них.

Пусть $G$ - произвольное множество формул из $L$; $A,B,C$ - произвольные формулы из $L$. Имеют место следующие *производные правила вывода*.

1. ***Правило перевертывания (контрапозиции)***:

если $G,A \vdash B$, то $G,\neg B \vdash \neg A$.

**Доказательство.**

1) $G,A \vdash B$ - по условию,
2) $G \vdash A \Rightarrow B$ - по теореме дедукции,
3) $(A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A)$ - теорема согласно п. д) леммы 4.2,
4) $(A \Rightarrow B) \vdash (\neg B \Rightarrow \neg A)$ - из 3) по *MP*,
5) $G \vdash \neg B \Rightarrow \neg A$ - из 2) и 4) по свойству выводимости (3 ),
6) $G, \neg B \vdash \neg B$ - по определению вывода,
7) $G, \neg B \vdash \neg B \Rightarrow \neg A$ - из 5),
8) $\neg B \Rightarrow \neg A, \neg B \vdash \neg A$ - по *MP*,
9) $G, \neg B \vdash \neg A$ - из 7) и 8) по 3-му свойству выводимости.

Что и требовалось доказать.

2. ***Правило удаления &***: $A\&B \vdash A$.

**Доказательство.**

1) $A, B \vdash A$ - по определению вывода,
2) $A, \neg A \vdash \neg B$ - из 1) по правилу перевертывания,
3) $\neg A \vdash A \Rightarrow \neg B$ - по теореме дедукции,
4) $\neg(A \Rightarrow \neg B) \vdash \neg\neg A$ - из 3) по правилу перевертывания,
5) $\neg\neg A \Rightarrow A$ - теорема согласно п. а) леммы 4.2,
6) $\neg\neg A \vdash A$ - из 5) по МР,
7) $\neg(A \Rightarrow \neg B) \vdash A$ - из 4) и 6) по свойству (3 ) выводимости.

Получили $A\&B \vdash A$. Что и требовалось.

3. ***Правило введения &***: $A,B \vdash A\&B$.

**Доказательство.**

1) $A, A \Rightarrow \neg B \vdash \neg B$ - по МР,

2) $A, \neg\neg B \vdash \neg(A \Rightarrow \neg B)$ - из 1) по правилу перевертывания,

3) $A, B \vdash A$, по определению вывода,

4) $B \Rightarrow \neg\neg B$ - *теорема* согласно п.б) леммы 4.2,

5) $B \vdash \neg\neg B$ - из 4) по МР,

6) $A, B \vdash \neg\neg B$ - из 5),

7) $A, B \vdash \neg(A \Rightarrow \neg B)$ - из 6) и 2) по 3-му свойству выводимости.

В результате получили, $A, B \vdash A\&B$. Что и требовалось. Аналогично доказывается, что $A, B \vdash B\&A$.

Так же можно доказать следующие правила.

4. ***Правила введения $\vee$:***

$A \vdash A \vee B$,

$B \vdash A \vee B$.

5. ***Правило доказательства разбором случаев***:

Если $A \vdash C$ и $B \vdash C$, то $A \vee B \vdash C$.

6. ***Правило сведения к нелепости (доказательство от противного):***

Если $A \vdash B$ и $A \vdash \neg B$, то $\vdash \neg A$.

Иногда правило вывода *modus ponens* записывают следующим образом:

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

Здесь в числителе перечислены гипотезы (посылки), в знаменателе- следствие из этих посылок, а символ $\Rightarrow$, стоящий справа, указывает, что данное правило исключает или вводит этот символ. Тогда рассмотренные выше правила можно записать следующим образом.

Теорема дедукции: $\dfrac{G, A \vdash B; G}{A \Rightarrow B} \Rightarrow$.

Правило перевертывания: $\dfrac{G, A \vdash B;\ G, \neg B}{\neg A} \neg$.

Правило удаления $\&$: $\dfrac{A \& B}{A} \ \&$ ; $\dfrac{A \& B}{B} \ \&$.

Правило введения $\&$: $\dfrac{A, B}{A \& B} \ \&$; $\dfrac{A, B}{B \& A} \ \&$.

Правила введения $\vee$: $\dfrac{A}{A \vee B} \ \vee$; $\dfrac{A}{B \vee A} \ \vee$.

Правило доказательства разбором случаев: $\dfrac{A \vdash C,\ B \vdash C,\ A \vee B}{C} \ \vee$.

Правило сведения к нелепости: $\dfrac{A \vdash B,\ A \vdash \neg B}{\neg} \ \neg$.

$\neg A$

Отметим, что перечисленные производные правила вывода позволяют намного упростить выкладки в теории $L$. Подчеркнем, что в принципе можно и не пользоваться этими производными правилами вывода, а пользоваться только правилом $MP$. Однако при этом может быть придется проводить более громоздкие выкладки. Так, например, если при выяснении, теорема или нет формула $A$, пользовались теоремой дедукции, то при отказе от этой теоремы дедукции фактически придется ее доказывать для этого частного случая. В других подобных случаях снова придется ее доказывать.

## § 11. Свойства исчисления высказываний

***Непротиворечивость исчисления высказываний.*** Исчисление высказываний, как и любую формальную аксиоматическую теорию, содержащую символ $\neg$ будем считать непротиворечивой, если ни для какой формулы $A$ не имеет места $\vdash A$ и $\vdash \neg A$, т.е. не может быть, чтобы одновременно были выводимы $A$ и $\neg A$. Для доказательства непротиворечивости исчисления высказываний (теории $L$) предварительно докажем теорему.

**Теорема 4.3.** Всякая теорема теории $L$ есть тавтология.

**Доказательство**. Легко убедиться, например, с помощью таблиц истинности, что каждая аксиома теории $L$ есть тавтология. Известно, что если $A$ и $A \Rightarrow B$ тавтологии, то и $B$ тавтология (см. теорему 1.1), т.е. правило *modus ponens*, примененное к тавтологиям, приводит к тавтологии. И так как всякую теорему можно доказать применением только правила *modus ponens* к аксиомам, то теорема есть тавтология. Что и требовалось доказать.

**Теорема 4.4**. Исчисление высказываний непротиворечиво, т.е. не существует формулы $A$ такой, чтобы $A$ и $\neg A$ были ее теоремами.

**Доказательство**. Согласно только что доказанной теореме 4.3, каждая теорема теории $L$ является тавтологией. Отрицание тавтологии не является тавтологией. Следовательно, ни для какой формулы $A$ невозможно, чтобы $A$ и $\neg A$ были теоремами исчисления высказываний.

Заметим, что все остальные свойства теории $L$ рассматриваем после выяснения ее непротиворечивости.

***Полнота исчисления высказываний.*** При описании формальных аксиоматических систем отмечалось, что свойство полноты характеризует достаточность теорем этой теории для некоторых целей. Исчисление высказываний бу-

дем считать *полным в широком смысле*, если в ней доказуема каждая формула, являющаяся тавтологией. Иначе можно сказать, что исчисление высказываний называется полной в широком смысле теорией, если множество теорем покрывает множество формул, являющихся тавтологиями.

Вводят, кроме того, понятие полноты в узком смысле. Исчисление высказываний называется *полным в узком смысле*, если присоединение к ее аксиомам какой-нибудь, не выводимой в ней формулы, приводит к противоречивой теории. Полнота в узком смысле означает, что аксиомы теории уже нельзя пополнить независимой аксиомой так, чтобы получить непротиворечивую теорию. Иначе можно сказать, что множество теорем $T$ такой теории еще не покрывает всего множества формул $\Phi$ (теория непротиворечива), но всякие попытки расширить множество $T$ приводят к тому, что $T$ покрывает все $\Phi$, т.е. теория становится противоречивой.

Покажем, что исчисление высказываний полно в широком смысле. Докажем сначала вспомогательную лемму.

**Лемма 4.3.** Пусть $A$ есть формула, а $B_1,B_2,...,B_k$ - пропозициональные буквы, входящие в $A$, и пусть задано некоторое распределение истинностных значений для $B_1,B_2,...,B_k$. Пусть тогда $B'_i$ есть $B_i$ , если $B_i$ принимает значение *И*, и $\neg B_i$, если $B_i$ принимает значение *Л*, и пусть, наконец, $A'$ есть $A$, если при этом распределении $A$ принимает значение *И*, и $\neg A$, если $A$ принимает значение *Л*. Тогда $B'_1,B'_2,...,B'_k \vdash A'$.

**Доказательство.** Будем считать, что формула $A$ записана без сокращений, т.е. без использования символов $\&$, $\vee$, $\equiv$. Доказательство проведем индукцией по числу ($n$) вхождений в $A$ примитивных связок (связок $\neg$ и $\Rightarrow$).

Если $n$=0, то $A$ представляет собой просто пропозициональную букву $B_1$ и утверждение леммы сводится к очевидным утверждениям: $B_1 \vdash B_1$ и $\neg B_1 \vdash \neg B_1$.

Допустим теперь, что лемма верна при любом $j<n$.

Случай 1. $A$ имеет вид отрицания: $\neg B$. Число вхождений примитивных связок в $B$, очевидно, меньше $n$.

Случай 1а. Пусть при заданном (выбранном) распределении истинностных значений букв $B_1,B_2,...,B_k$ форма $B$ принимает значение *И*. Тогда $A$ принимает значение *Л* и $A'=\neg A$, а $B'=B$. По индуктивному предположению, имеем

$$B'_1,B'_2,...,B'_k \vdash B. \qquad (1)$$

Согласно лемме 4.2(б) имеем:

$$\vdash B \Rightarrow \neg\neg B. \qquad (2)$$

Из (1) и (2) по *MP* получим:

$B'_1,B'_2,...,B'_k \vdash \neg\neg B.$ (3)

В рассматриваемом случае $A=\neg B$, $A' = \neg A = \neg\neg B$ тогда из (3) следует требуемое: $B'_1,B'_2,...,B'_k \vdash A'$.

Случай 1б. Пусть $B$ принимает значение *Л*, тогда $B'$ есть $\neg B$, а $A'$ совпадает с $A$. По индуктивному предположению $B'_1,B'_2,...,B'_k \vdash \neg B$, что и требовалось получить, ибо $\neg B$ есть $A'$.

Случай 2. $A$ имеет вид $B \Rightarrow C$. Тогда число вхождений примитивных связок в $B$ и $C$ меньше, чем в $A$. Поэтому, в силу индуктивного предположения

$B'_1,B'_2,...,B'_k \vdash B',$ (4)

$B'_1,B'_2,...,B'_k \vdash C'.$ (5)

Случай 2а. $B$ принимает значение *Л*, тогда, вне зависимости от значения формулы $C$, формула $A=B \Rightarrow C$ принимает значение *И*. Так как значение $B$ есть *Л*, то $B' = \neg B$ , а из истинности $A$ следует, что $A' = A$. Из соотношения (4) получаем

$B'_1,B'_2,...,B'_k \vdash \neg B.$ (6)

Согласно лемме 4.2(в) имеем:

$\vdash \neg B \Rightarrow (B \Rightarrow C).$ (7)

Из (6) и (7) по *MP* следует:

$B'_1,B'_2,...,B'_k \vdash B \Rightarrow C.$

Последнее и есть требуемое, так как $B \Rightarrow C$ есть $A'$.

Случай 2б. $C$ принимает значение *И*. Следовательно, $A$ принимает значение *И* и $C'$ есть $C$, а $A'$ есть $A$. Тогда из (5) получим

$B'_1,B'_2,...,B'_k \vdash C.$ (8)

Согласно схеме аксиом *A1* имеем

$\vdash C \Rightarrow (B \Rightarrow C).$ ( 9)

Из (8) и (9) по *MP* следует

$B'_1,B'_2,...,B'_k \vdash B \Rightarrow C$ (10)

и так как $B \Rightarrow C$ совпадает с $A'$, то (10) является требуемым.

Случай 2в. $B$ принимает значение $И$, $C$ принимает значение $Л$. Тогда $A$ принимает значение $Л$, следовательно, $A' = \neg A$, $B'$ есть $B$, а $C'$ есть $\neg C$. Таким образом, из (4) и (5) получим соответственно

$$B'_1,B'_2,...,B'_k \vdash B, \qquad (11)$$

$$B'_1,B'_2,...,B'_k \vdash \neg C. \qquad (12)$$

По лемме 2(е) имеем

$$\vdash B \Rightarrow (\neg C \Rightarrow \neg(B \Rightarrow C)). \qquad (13)$$

Из (11) и (13) по МР следует

$$B'_1,B'_2,...,B'_k \vdash \neg C \Rightarrow \neg(B \Rightarrow C), \qquad (14)$$

а из (12) и (14) по МР следует

$B'_1,B'_2,...,B'_k \vdash \neg(B \Rightarrow C)$.

Последнее является требуемым, ибо $\neg(B \Rightarrow C)$ есть $A'$. Лемма доказана.

**Теорема 4.5 (о полноте)**. Если формула $A$ теории $L$ является тавтологией, то она является теоремой теории $L$.

**Доказательство**. Предположим, что $A$ есть тавтология и $B_1,B_2,...,B_k$ - пропозициональные буквы, входящие в $A$. При каждом распределении истинностных значений для букв $B_1,B_2,...,B_k$ имеем, в силу леммы 4.3, $B'_1,B'_2,...,B'_k \vdash A$ ($A'$ совпадает с $A$, так как $A$ всегда $И$). Поэтому в случае, когда $B_k$ принимает значение $И$, получим

$B'_1,B'_2,...,B'_{k-1},B_k \vdash A$ ,

а когда $B_k$ принимает значение $Л$ имеем

$B'_1,B'_2,...,B'_{k-1}, \neg B_k \vdash A$.

Отсюда по теореме дедукции получим соответственно

$$B'_1,B'_2,...,B'_{k-1} \vdash B_k \Rightarrow A, \qquad (1')$$

$$B'_1,B'_2,...,B'_{k-1} \vdash \neg B_k \Rightarrow A. \qquad (2')$$

По лемме 4.2(ж) имеем:

$$\vdash (B_k \Rightarrow A) \Rightarrow ((\neg B_k \Rightarrow A) \Rightarrow A). \qquad (3')$$

Теперь по *МР* из (1') и (3') получим

$B'_1,B'_2,...,B'_{k-1} \vdash (\neg B_k \Rightarrow A) \Rightarrow A.$ (4')

Далее снова по МР из (4') и (2') следует
$B'_1,B'_2,...,B'_{k-1} \vdash A.$
Таким образом, из $B'_1,B'_2,...,B'_k \vdash A$ получили $B'_1,B'_2,...,B'_{k-1} \vdash A$, т.е. исключили $B'_k$. Точно так же исключим $B'_{k-1}$ и так далее, после $k$ таких шагов придем к
$\vdash A$,
т.е. $A$ является теоремой, что и требовалось доказать.

Можно доказать и полноту в узком смысле, см., например, [7]. Примем без доказательства.

***Независимость аксиом***. Отдельная аксиома дедуктивной теории, в том числе и исчисления высказываний, является независимой, если эту аксиому нельзя вывести в этой теории из остальных аксиом. Система аксиом является независимой, если каждую из них нельзя вывести из остальных.

**Теорема 4.6**. Каждая из схем *A1, A2* и *A3* независимы от остальных.

**Доказательство.** Независимость *A1*. Рассмотрим следующую таблицу.

| $A$ | $B$ | $\neg A$ | $A \Rightarrow B$ |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 2 |
| 2 | 0 | 0 | 0 |
| 0 | 1 | | 2 |
| 1 | 1 | | 2 |
| 2 | 1 | | 0 |
| 0 | 2 | | 2 |
| 1 | 2 | | 0 |
| 2 | 2 | | 0 |

При всяком распределении значений *0, 1, 2* для букв, входящих в формулу $A$, эта таблица позволяет найти соответствующее значение формулы $A$. Если формула $A$ всегда принимает значение *0*, то она называется выделенной.

Правило *modus ponens* сохраняет свойство формулы быть выделенной, так как, если $A$ и $A \Rightarrow B$ принимает значение *0*, т.е. выделенные, то согласно таблице и $B$ принимает значение *0*, следовательно тоже выделенная.

Покажем, что всякая аксиома, получающаяся по схеме *A2* и *A3*, тоже выделенные. Для *A3* имеем следующую таблицу:

| $(\neg$ | $B$ | $\Rightarrow$ | $\neg$ | $A)$ | $\Rightarrow$ | $((\neg$ | $B$ | $\Rightarrow$ | $A)$ | $\Rightarrow$ | $B)$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 2 | 1 | 0 | 0 | 1 | 0 | 2 | 0 | 0 | 0 |
| 1 | 0 | 2 | 1 | 1 | 0 | 1 | 0 | 2 | 1 | 0 | 0 |
| 1 | 0 | 2 | 0 | 2 | 0 | 1 | 0 | 0 | 2 | 0 | 0 |
| 1 | 1 | 2 | 1 | 0 | 0 | 1 | 1 | 2 | 0 | 0 | 1 |
| 1 | 1 | 2 | 1 | 1 | 0 | 1 | 1 | 2 | 1 | 0 | 1 |
| 1 | 1 | 2 | 0 | 2 | 0 | 1 | 1 | 0 | 2 | 2 | 1 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *0* | *2* | *2* | *1* | *0* | *0* | *0* | *2* | *0* | *0* | *2* | *2* |
| *0* | *2* | *2* | *1* | *1* | *0* | *0* | *2* | *2* | *1* | *0* | *2* |
| *0* | *2* | *0* | *0* | *2* | *0* | *0* | *2* | *2* | *2* | *0* | *2* |

Из приведенной таблицы видно, что аксиома, полученная по *A3,* является выделенной. Аналогично можно показать выделенность *A2*.

Если бы *A1* была выводима в этой теории из аксиом *A2* и *A3*, то она должна быть выделенной, так как применение *MP* к выделенным формулам приводит к выделенным. Но *A1* не является выделенной, так как формула

| $A_1$ | $\Rightarrow$ | $(A_2$ | $\Rightarrow$ | $A_1)$ |
|---|---|---|---|---|
| *1* | *2* | *2* | *0* | *1* |

$A_1 \Rightarrow (A_2 \Rightarrow A_1)$ принимает значение *2,* когда $A_1=1$, $A_2=2$. Таким образом, аксиома *A1* не может быть выведена из аксиом *A2* и *A3*, следовательно, она независима от них.

Независимость *A2*. Рассмотрим следующую таблицу:

| *A* | *B* | $\neg A$ | $A \Rightarrow B$ |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 |
| 0 | 1 | | 2 |
| 1 | 1 | | 2 |
| 2 | 1 | | 0 |
| 0 | 2 | | 1 |
| 1 | 2 | | 0 |
| 2 | 2 | | 0 |

Всякую формулу, принимающую согласно этой таблице всегда значение *0*, назовем гротескной. Правило *MP* сохраняет гротескность, ибо если *A* и $A \Rightarrow B$ гротескны, то по приведенной таблице формула *B* тоже гротескна.

Всякая аксиома, получаемая по схеме *A1* и *A3*, гротескна. Действительно, для *A1* имеем

| *A* | $\Rightarrow$ | (*B* | $\Rightarrow$ | *A*) |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 2 | 1 |
| 2 | 0 | 0 | 1 | 2 |
| 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 2 | 1 |
| 2 | 0 | 1 | 0 | 2 |
| 0 | 0 | 2 | 0 | 0 |
| 1 | 0 | 2 | 0 | 1 |
| 2 | 0 | 2 | 0 | 2 |

Аналогичным образом доказывается гротескность *A3* (проделать самостоятельно).

Если *A2* выводима из *A1* и *A3*, то она должна быть гротескной, ибо *MP*, примененное к гротескным формулам, дает гротескную. Но частный случай *A2* не является гротескным:

$(A_1 \Rightarrow (A_2 \Rightarrow A_3)) \Rightarrow ((A_1 \Rightarrow A_2) \Rightarrow (A_1 \Rightarrow A_3))$

*0 1 0 2 1* ***2*** *0 0 0 1 0 2 1*

ибо принимает значение *2*. Следовательно, *A2* независима от *A1* и *A3*.

Независимость *A3*. Пусть *A* - произвольная формула и $h(A)$ - формула, полученная из *A* стиранием всех вхождений знака отрицания в *A*. Нетрудно убедиться, что для всякой аксиомы *A*, полученной по схемам *A1* и *A2,* $h(A)$ будет

тавтологией. Очевидно, что $h(A \Rightarrow B)$ есть $h(A) \Rightarrow h(B)$. Если $h(A \Rightarrow B) = h(A) \Rightarrow h(B)$ и $h(A)$ - тавтологии, то и $h(B)$ - тавтология, следовательно, правило *MP* сохраняет свойство $A$ иметь в качестве $h(A)$ тавтологию. Таким образом, всякая формула $A$, выводимая из *A1, A2* с помощью *MP* имеет в качестве $h(A)$ тавтологию. Но

$$h((\neg A_1 \Rightarrow \neg A_1) \Rightarrow ((\neg A_1 \Rightarrow A_1) \Rightarrow A_1)) = (A_1 \Rightarrow A_1) \Rightarrow ((A_1 \Rightarrow A_1) = A_1),$$

и эта последняя формула не является тавтологией. Следовательно, частный случай *A3* - формула $(\neg A_1 \Rightarrow \neg A_1) \Rightarrow ((\neg A_1 \Rightarrow A_1) = A_1)$, не выводима из *A1* и *A2* с помощью *MP*. Теорема доказана.

***Разрешимость***. Дедуктивная теория, в том числе и исчисление высказываний, является разрешимой, если в этой теории понятие теоремы эффективно, т.е. существует правило (метод), позволяющее для произвольной формулы за конечное число действий выяснить, является она теоремой или нет.

**Теорема 4.7.** Исчисление высказываний (теория $L$) является разрешимой теорией.

**Доказательство**. Было доказано, что каждая теорема теории $L$ является тавтологией и (обратно) каждая формула, являющаяся тавтологией, есть теорема. Таким образом, формула $A$ теории $L$ является теоремой тогда и только тогда, когда она является тавтологией. Следовательно, для того, чтобы выяснить, $A$ теорема или нет, достаточно выяснить, $A$ тавтология или нет. Последнее легко определить, например, с помощью таблиц истинности или приведением к к.н.ф. В результате имеем требуемое правило, позволяющее для каждой формулы $A$ за конечное число шагов выяснить, $A$ теорема или нет.

Итак, исчисление высказываний является непротиворечивой, полной в широком и узком смысле, разрешимой и эффективно аксиоматизированной формальной теорией с независимой системой аксиом.

## § 12. Другие аксиоматизации исчисления высказываний

В предыдущих параграфах рассматривался пример формальной аксиоматической теории - исчисление высказываний (теория $L$), которая была задана множеством символов, формул, аксиом и правил вывода. Теперь рассмотрим некоторую *формализованную теорию C*. Для задания формализованной теории необходимо задать ее символы, формулы (правильно построенные выражения) и из множества формул выделить подмножество, элементы которого считаются «истинными» формулами. Пусть формализованная теория $C$ задана так, что

1) алфавиты теорий $L$ и $C$ совпадают, т.е. алфавитом для $C$ является следующее множество символов: $\neg, \Rightarrow, ), (, A_1, A_2, \ldots$;

2) множества формул $L$ и $C$ совпадают ($\Phi_L=\Phi_C$ , где $\Phi_L(\Phi_C)$ - множество формул теории $L(C)$). Таким образом, формулой в $C$ является всякая пропозициональная форма, построенная из букв $A_1, A_2, \ldots$ с помощью связок $\neg$ и $\Rightarrow$;

3) "истинными" формулами в теории $C$ считаются те и только те формулы теории $C$, которые являются тавтологиями. Пусть $V_C$ обозначает множество "истинных" формул теории $C$.

В исчислении высказываний было задано $T_L$ - множество теорем (с помощью аксиом *A1-A3* и правила вывода *MP*). При этом теоремами в $L$ оказались те, и только те формулы из $L$, которые являются тавтологиями. Следовательно, множество теорем формальной теории $L$ совпадает с множеством "истинных" формул формализованной теории $C$: $T_L = V_C$ .

Итак, формальная теория $L$ оказалась построенной таким образом, что ее множество теорем ($T_L$) в точности совпадает с множеством "истинных" формул ($V_C$ ) формализованной теории $C$.

Из полноты в узком смысле теории $L$ следует, что всякая попытка расширить множество $T_L$ приводит к противоречивой теории, поэтому нельзя к *A1-A3* добавить недоказуемую из них формулу, не приходя при этом к противоречию. Естественно выяснить, можно ли, взяв другие аксиомы и, может быть, другие правила вывода, получить $T_L = V_C$.

Оказывается, что можно. Например, построим формальную аксиоматическую (дедуктивную) теорию *L1*, которая отличается от $L$ только тем, что вместо схем аксиом *A1-A3* здесь имеются лишь три конкретные аксиомы:

1) $A_1 \Rightarrow (A_2 \Rightarrow A_1)$,

2) $(A_1 \Rightarrow (A_2 \Rightarrow A_3)) \Rightarrow ((A_1 \Rightarrow A_2) \Rightarrow (A_1 \Rightarrow A_3))$,

3) $(\neg A_2 \Rightarrow \neg A_1) \Rightarrow ((\neg A_2 \Rightarrow A_1) \Rightarrow A_2)$,

но имеется, кроме правила вывода MP (*modus ponens*), еще одно правило вывода - правило подстановки, разрешающее подстановку любой формулы на места всех вхождений данной пропозициональной буквы в данную формулу. При этом можно показать, что $T_{L1} = V_C = T_L$.

Оказывается, можно даже построить формальную аксиоматическую (дедуктивную) теорию *L2* с тем же алфавитом и множеством формул, что и $L$, но всего с одной схемой аксиом

$$(((A \Rightarrow B) \Rightarrow (\neg C \Rightarrow \neg D)) \Rightarrow E) \Rightarrow ((E \Rightarrow A) \Rightarrow (D \Rightarrow A))$$

и единственным правилом вывода – MP (*modus ponens*). Здесь тоже $T_{L2} = V_C = T_L$. Возможны и другие задания формальных аксиоматических (дедуктивных) теорий так, чтобы ее множество теорем совпадало с множеством $V_C$.

*... абстракции отражают природу глубже, вернее, полнее.*

*В. И. Ленин (Философские тетради)*

## § 13. Теории первого порядка

Теория первого порядка (теория $K$) представляет собой формальную аксиоматическую теорию. Следовательно, для ее задания необходимо определить символы, формулы, аксиомы и конечное число правил вывода.

1. Символами всякой теории первого порядка служат:

$\neg, \Rightarrow$ - пропозициональные связки,

(,), ' - знаки пунктуации,

$x_1, x_2, \ldots$ - счетное множество предметных переменных,

$a_1, a_2, \ldots$ -конечное (возможно и пустое) или счетное множество предметных констант,

$A^k_i$ $(i,k \geq 0)$ - непустое, конечное или счетное множество предикатных букв,

$f^k_i$ $(i,k \geq 1)$ - конечное (возможно и пустое) или счетное множество функциональных букв,

$\forall x_i, \exists x_i$ $(i \geq 1)$ - кванторы всеобщности и кванторы существования.

Различные теории первого порядка могут отличаться друг от друга составом символов, так, например, в некоторых теориях могут совсем отсутствовать функциональные буквы. Из перечня символов видно, что некоторые из них обязательно принадлежат всем теориям первого порядка.

2. Формулы теории первого порядка определяются точно так же, как определяются формулы в логике предикатов.

3. Аксиомы теории $K$ разбиваются на два класса: логические аксиомы и собственные аксиомы.

*Логические аксиомы*: каковы бы ни были формулы $A$, $B$, $C$ теории $K$, следующие формулы являются логическими аксиомами теории $K$:

*A1:* $A \Rightarrow (B \Rightarrow A)$;

*A2:* $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$;

*A3:* $(\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)$;

*A4:* $\forall x_i \; A(x_i) \Rightarrow A(t)$,

здесь $A(x_i)$ есть формула теории $K$ и $t$ есть терм теории $K$, свободный для $x_i$ в $A(x_i)$;

*A5:* $\forall x_i (A \Rightarrow B) \Rightarrow (A \Rightarrow \forall x_i B)$, если формула $A$ не содержит свободных вхождений $x_i$.

*Собственные аксиомы*. Собственные аксиомы не могут быть сформулированы в общем случае, ибо меняются от теории к теории. В дальнейшем рассмотрим конкретную теорию первого порядка (формальную арифметику), для которой будут сформулированы (перечислены) собственные аксиомы.

4. Правилами вывода во всякой теории первого порядка являются:

1) ***modus ponens***: $B$ является непосредственным следствием $A$ и $A \Rightarrow B$;

2) *правило обобщения* (***Gen***): из $A$ следует $\forall x_i A$, точнее $\forall x_i A$ является непосредственным следствием из $A$.

Теория первого порядка, не содержащая собственных аксиом, называется ***исчислением предикатов первого порядка*** и обозначается через $K_1$.

*Единица есть корень всякого числа, и она находится вне чисел. Корень числа она потому, что через неё определяют всякое число. Вне чисел она потому, что определяется сама по себе, т.е. без какого-либо другого числа. Остальные же числа не могут быть найдены без единицы.*

*Аль-Хорезми.*

## § 14. Формальная арифметика (теория *S*)

Первое полуаксиоматическое построение арифметики было предложено Дедекиндом (1901), далее усовершенствовано Пеано и известно под названием "система аксиом Пеано". Эта система формулируется следующим образом:

***P1***: *0* - есть натуральное число;

***P2***: для любого натурального числа *x* существует другое натуральное число, обозначаемое *x'* и называемое непосредственно следующим за *x*;

***P3***: $0 \neq x'$ для любого натурального числа *x*;

***P4***: если $x' = y'$, то $x=y$;

***P5***: если *U* есть свойство, которым, быть может, обладают одни и не обладают другие натуральные числа, и если:

1) натуральное число *0* обладает свойством *U* и

2) для всякого натурального числа *x* из того, что *x* обладает свойством *U* следует, что и натуральное число *x'* обладает свойством *U*,

то свойством *U* обладают все натуральные числа (принцип математической индукции).

Заметим, что иногда аксиома ***P1*** формулируется иначе: *1* - есть натуральное число. В этом случае легко видеть, что аксиоматика Пеано развивает идею о том, что единица есть корень всякого числа, см эпиграф.

Этих аксиом вместе с некоторыми фактами теории множеств достаточно для построения арифметики натуральных чисел, а также теории рациональных, вещественных и комплексных чисел. Однако это построение недостаточно строго с наших позиций - не является формальной аксиоматической теорией. Поэтому зададим некоторую формальную аксиоматическую теорию, а именно: некоторую теорию первого порядка, которая оказывается достаточной для вывода всех основных результатов элементарной арифметики.

*Формальной арифметикой (теорией S)* называем теорию первого порядка, имеющую:

единственную предикатную букву $A^2_1$,

единственную предметную постоянную $a_1$,

три функциональные буквы $f^1_1, f^2_1, f^2_2$,

кроме того, очевидно, ее символами являются $\neg, \Rightarrow, ), (, ', x_1, x_2, \ldots, \forall x_1, \forall x_2, \ldots, \exists x_1, \exists x_2, \ldots$ .

Чтобы использовать привычные для нас записи, обозначим

$A^2_1(t,s)$ через $t=s$,

$a_1$ - через *0*,
$f^1_1(t)$ - через *t'*,
$f^2_1(t,s)$ - через $t+s$,
$f^2_2(t,s)$- через $t \bullet s$ ($\bullet$ - понимается как знак умножения).

Как известно, аксиомы каждой теории первого порядка, следовательно, и формальной арифметики, делятся на две группы аксиом: логические аксиомы - аксиомы *A1-A5* и собственные аксиомы. Собственными аксиомами формальной арифметики являются следующие формулы:

*S1:* $x_1 = x_2 \Rightarrow ((x_1 = x_3) \Rightarrow (x_2 = x_3))$;
*S2:* $x_1 = x_2 \Rightarrow x_1' = x_2'$;
*S3:* $0 \neq x_1'$;
*S4:* $x_1' = x_2' \Rightarrow x_1 = x_2$;
*S5:* $x_1 + 0 = x_1$;
*S6:* $x_1 + x_2' = (x_1 + x_2)'$;
*S7:* $x_1 \bullet 0 = 0$;
*S8:* $x_1 \bullet x_2' = x_1 \bullet x_2 + x_1$;
*S9:* $A(0) \Rightarrow (\forall x(A(x) \Rightarrow A(x')) \Rightarrow \forall x A(x))$,

где $A(x)$ - произвольная формула теории *S*.

Напомним, что правилами вывода любой теории первого порядка являются правила *MP* и *Gen*.

Рассмотрим собственные аксиомы теории *S* и сравним их с аксиомами Пеано.

Заметим, что аксиомы *S1 - S8* являются конкретными формулами, в то время как *S9* представляет собой схему аксиом, порождающую бесконечное множество аксиом.

Аксиомы *S1* и *S2* обеспечивают некоторые необходимые свойства равенства, которые Дедекиндом и Пеано предполагались как интуитивно очевидные.

Аксиомы *S3* и *S4* соответствуют аксиомам ***P3*** и ***P5*** системы аксиом Пеано. Аксиомы ***P1*** и ***P2*** пеановской системы обеспечивают существование *0* (нуля) и операции "непосредственно следующий", которым в теории *S* соответствует предметная константа $a_1$ и функциональная буква $f_1^1$.

Аксиомы *S5-S8* служат для определения операций сложения и умножения. В аксиоматике Пеано нет аксиом соответствующих *S5-S8*, потому что Дедекинд и Пеано допускали использование интуитивной теории множеств, в рамках которой существование операций + и $\bullet$ (сложения и умножения) удовлетворяющих аксиомам *S5-S8* - выводимо.

Доказывается, что множество формул любой теории первого порядка счетно, поэтому по схеме *S9* можем получить лишь счетное множество аксиом. В аксиоме ***P5*** рассматривается некоторое свойство *U,* которым, быть может, обладают одни и не обладают другие натуральные числа, т.е. свойством *U* обладают элементы некоторого подмножества множества натуральных чисел. Известно, что мощность множества всех подмножеств множества натуральных

чисел больше, чем мощность счетного множества. Поэтому схема аксиом *S9*, которая называется принципом математической индукции, не соответствует аксиоме ***P5***, поскольку схема аксиом *S*9 может иметь дело лишь со счетным множеством свойств, определяемых формулами теории *S*, а в аксиоме ***P5*** интуитивно предполагается более, чем счетное свойств натуральных чисел.

В заданной таким образом теории *S* (формальной арифметике) можно получить вывод всех известных теорем арифметики, например, доказать, что для любых термов *t, s* и *z* имеют место

$\vdash t+s=s+t$ - коммутативность сложения;

$\vdash (t+s)+z=t+(s+z)$ - ассоциативность сложения;

$\vdash t \bullet s=s \bullet t$ - коммутативность умножения;

$\vdash t \bullet (s+z)=t \bullet s+t \bullet z$ - дистрибутивность и т.п.

Не будем доказывать приведенные утверждения, а также другие арифметические теоремы. Нас прежде всего будет интересовать не арифметические теоремы, а свойства теории *S* (формальной арифметики) и других теорий, содержащих в себе *S*.

*Ничто не совершенно во всех отношениях.*
*Гораций*[2]

## § 15. Свойства теорий первого порядка

Как было уже отмечено, одним из важнейших свойств дедуктивной теории является ее непротиворечивость. Для исчисления высказываний непротиворечивость была доказана сравнительно просто. Для произвольной теории первого порядка оказывается установить непротиворечивость в рамках этой же теории не удается. Но в некоторых частных случаях, например, для исчисления предикатов первого порядка, удается доказать непротиворечивость.

***Непротиворечивость исчисления предикатов первого порядка (теории $K_1$)***.

**Теорема 4.8**. Всякое исчисление предикатов первого порядка непротиворечиво.

**Доказательство**. Для произвольной формулы *A* обозначим через *h*(*A*) выражение, получающееся в результате следующего преобразования формулы *A*: в *A* спускаются все кванторы и термы (вместе с соответствующими скобками и запятыми).

Например, $h(\forall x_1 A^2_1 (x_1, x_2) \Rightarrow A^1_1 (x_3))$ есть $A^2_1 \Rightarrow A^1_1$ ,

$h(\neg\forall x_1 A^3_2(x_2, a_1, x_1) \Rightarrow A^1_3(a_2) \Rightarrow A^1_2(x_3)$ есть $\neg A^3_2 \Rightarrow A^1_3 \Rightarrow A^1_2$ .

[2] *Римский поэт (65-8 гг. до н. э.)*

По существу $h(A)$ всегда является пропозициональной формой, в которой роль пропозициональных букв играют символы $A^k_j$ .

Ясно, что имеют место

$h(\neg A)=\neg h(A)$, $h(A\Rightarrow B)=h(A)\Rightarrow h(B)$.

Также покажем, что для всякой аксиомы $A$, получаемой по какой-нибудь из схем *A1-A5*, $h(A)$ является тавтологией.

Для *A1,A2,A3* это очевидно.

Всякий частный случай *A4* $\forall x_i A(x_i)\Rightarrow A(t)$ преобразуется операцией $h$ в тавтологию вида $B\Rightarrow B$, а всякий частный случай *A5* $\forall x_i(A\Rightarrow B)\Rightarrow(A\Rightarrow \forall x_i B)$ преобразуется в тавтологию вида $(C\Rightarrow D)\Rightarrow(C\Rightarrow D)$.

Далее: 1) пусть $h(A)$ и $h(A\Rightarrow B)=h(A)\Rightarrow h(B)$ тавтологии, тогда $h(B)$ – тоже тавтология;

2) если $h(A)$ тавтология, то $h(\forall x_i A)$ тавтология, ибо результаты применения операции $h$ к $A$ и к $\forall x_i A$ совпадают.

Итак, из случаев 1) и 2) следует, что применение правил вывода *MP* и *Gen* сохраняет то свойство формул, что применение к ним функции $h$ снова приводит к тавтологии.

Теперь докажем, что если $A$ теорема, то $h(A)$ - тавтология. Пусть формула $A$ есть теорема теории $K_1$, т.е. существует ее вывод в $K_1$:

$B_1,B_2...,B_n$,

где $B_n=A$ и каждое из $B_i$ $(1\le i\le n)$ является либо аксиомой, либо непосредственным следствием по *MP* либо *Gen* из предыдущих формул этой последовательности. Тогда $h(B_i)$ является тавтологией для любого $i$, в частности, тавтология $h(B_n)=h(B)$.

Если бы существовала формула $B$ в $K_1$ такая, что $\vdash B$ и $\vdash \neg B$ (выводимо $B$ и $\neg B$), то по доказанному $h(B)$ и $h(\neg B)=\neg h(B)$ были бы тавтологиями, что невозможно. Значит, не существует формулы $B$ такой, что $\vdash B$ и $\vdash \neg B$, следовательно, всякое исчисление предикатов первого порядка непротиворечиво. Что и требовалось доказать.

***О полноте исчисления предикатов первого порядка (теории $K_1$)***.

Как было ранее замечено, в дедуктивных теориях вводят различные понятия полноты. *Исчисление предикатов 1-го порядка называется полным в широком смысле*, если каждая логическая общезначимая формула является теоремой.

Можно доказать, что во всяком исчислении предикатов первого порядка каждая теорема является логически общезначимой формулой.

Кроме того можно доказать, что каждая логически общезначимая формула из $K_1$ является теоремой в $K_1$.

Следовательно, во всяком исчислении предикатов первого порядка теоремами являются те, и только те формулы, которые логически общезначимы (теорема Гёделя о полноте). **Таким образом, исчисление предикатов первого порядка (теория $K_1$) является полным в широком смысле теорией.**

Теория $K$ считается *полной в узком смысле*, если для любой замкнутой формулы $A$ доказуемо $A$ либо $\neg A$.

В отличие от исчисления высказываний, исчисление предикатов оказывается неполным в узком смысле. Это означает, что множество теорем теории $K_1$ (множество $T_{K1}$) можно некоторым образом расширить и при этом $T_{K1}$ не покроет все множество формул теории $K_1$.

***О разрешимости исчисления предикатов***. Имеет место следующая теорема.

**Теорема 4.9** (теорема Чёрча). Исчисление предикатов является неразрешимой теорией.

Следовательно, не существует метода позволяющего для произвольной формулы $A$ логики предикатов за конечное число действий выяснить $A$ теорема или нет.

Можно привести примеры гораздо более богатых теорий, чем исчисление предикатов 1-го порядка, для которых оказалось возможным дать строгое доказательство непротиворечивости и полноты в широком смысле. Примером может служить арифметическая система натуральных чисел с одной операцией сложения (без операции умножения). Но для теорий, содержащих уже всю арифметику натуральных чисел, картина качественно меняется.

Знаменитые и важнейшие теоремы Геделя посвящены исследованию возможностей формальных аксиоматических теорий и выяснению их непротиворечивости.

**Первая теорема Геделя**. Первая теорема Геделя утверждает, что каждая формальная аксиоматическая теория $K$, настолько богатая, чтобы содержать формальную арифметику, такова, что если $K$ непротиворечива, то $K$ существенно не полна, т.е. содержит некоторую формулу, что в $K$ ее нельзя ни доказать, ни опровергнуть, хотя с помощью дополнительных средств, выходящих за рамки этой теории $K$, можно показать ее истинность. Более того, даже если дополнить аксиомы теории $K$ так, чтобы известные истинные формулы были доказуемы, все равно и для такой расширенной системы всегда существует истинная, но не разрешимая (нельзя ни доказать, и не опровергнуть) формула. Теорема Геделя утверждает, что такое расширение теории не может сделать ее полной.

Теорема Геделя показывает, что множество теорем теории (множество $T_s$) содержится во множестве всех истинных арифметических формул $V_s$ ($T_s \subset V_s$) и в то же время нельзя построить непротиворечивое эффективно аксиоматизированное расширение для $S$ так, чтобы полученное множество теорем покрыло все $V_s$.

Отсюда следует, что формальный аксиоматический подход к арифметике натуральных чисел не в состоянии охватить всю область истинных арифметических суждений (формул). Таким образом, результат Геделя указывает на не-

которую принципиальную ограниченность возможностей аксиоматического метода.

**Вторая теорема Геделя**. В этой теореме Гедель показал, что никакое предложение, которое можно точным образом интерпретировать как выражающее непротиворечивость какой-либо непротиворечивой формальной аксиоматической теории *К*, содержащей арифметику, не может быть доказано в этой теории *К*.

Более вольно излагая эту теорему, можно сказать, что «если теория *К*, содержащая арифметику, непротиворечива, то непротиворечивость ее нельзя доказать средствами самой теории *К*».

Конечно, неплохо, если получим доказательство на основе более богатой теории. Но нам нужно знать, непротиворечива ли эта более богатая теория. Теорема Геделя утверждает, что если она непротиворечива, то непротиворечивость ее нельзя доказать средствами самой теории, т.е. придется привлекать еще более богатую теорию, непротиворечивость которой тоже нельзя доказать в ней самой, и т.д.

Доказательство непротиворечивости арифметики натуральных чисел с привлечением новых правил вывода было впервые получено Генценом в 1936г., а впоследствии были получены еще несколько доказательств. Эти доказательства имеют относительную ценность. Они сводят доказательство непротиворечивости арифметики к доказательству непротиворечивости более богатых теорий. В то же время эти доказательства демонстрируют, какие новые правила вывода следует допустить (принять), если нужно установить непротиворечивость арифметики.

## § 16. Значение аксиоматического метода

Говоря об ограниченности аксиоматических подходов, нельзя умолчать о достоинствах и большом значении аксиоматического подхода.

Аксиоматизация теории позволяет: 1) систематизировать научный материал, 2) обеспечить определенную организацию научного знания, 3) исследовать структуру различных теорий и их взаимоотношение, 4) обеспечить необходимую строгость рассуждений.

Формализация теории, опирающаяся на аксиоматический метод, имеет существенное значение для объяснения и уточнения понятий теории и выявления используемых в ней методов доказательств. С первого взгляда может показаться, что научная терминология, во многих случаях значительно отличающаяся от обычного словоупотребления, является менее ясной и более искусственной, чем повседневная речь. Однако именно благодаря известной искусственности достигается большая точность и определенность понятий, их ясность. В ряде случаев нельзя правильно поставить вопрос, ни тем более ответить на него, пока не уточним соответствующее понятие.

Так же как уточнение понятий, уточнение логических средств вывода, достигаемое посредством формализации, имеет, как видим, решающее значение для того, чтобы сделать доказательство необходимо строгим и свободным от ссылок на очевидность и интуицию.

С проблемой формализации неразрывно связано создание различных научных или формализованных языков. Основная цель, которая при этом преследуется, состоит в том, чтобы построить язык, свободный от недостатков обычного языка. Благодаря точности и однозначности формализованные языки находят применение там, где предъявляются повышенные требования к строгости.

Д. Гильберт писал: "***Под знаком аксиоматического метода математика проявляет свою руководящую роль в науке вообще***".

## § 17. Теория естественного вывода

Естественный вывод является интересным подходом к заданию дедуктивной теории. Здесь, как и в любой дедуктивной теории, задаются алфавит, формулы (правильно построенные выражения), но нет аксиом, есть только правила вывода. Оказывается, что задание только правил выводов позволяет выделить из множества всех формул некоторое подмножество формул, элементы которого и называются теоремами. Следовательно, не имея ни одной аксиомы (!) можно получить теоремы.

Рассмотрим задание исчисления высказываний в виде теории естественного вывода.

Пусть символами являются $\neg, \&, \vee, \Rightarrow, (,)$, $A_1$, $A_2$,..., а формулами - пропозициональные формы, образованные из $A_1$, $A_2$,... с помощью $\neg, \&, \vee, \Rightarrow$.

Пусть $A$, $B$, $C$ - произвольные формулы. Правила вывода задаются следующим образом. В каждом правиле вывода справа стоит символ в качестве имени этого правила. Например, одно из правил записывается в виде:

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

Это правило называется правилом введения конъюнкции и означает, что из $A$ и $B$ можно вывести $A\&B$. Правило введения дизъюнкции

$$\frac{A}{A \vee B} \quad \vee$$

означает, что из $A$ можно вывести $A \vee B$, где $B$ - произвольная формула. Точно так же в каждом правиле вывода в "числителе" записываются гипотезы (посылки) а в "знаменателе" то, что выводится из этих гипотез.

Все правила вывода подразделяются на правила введения и правила исключения.

Правилами введения являются:

$$\frac{A, B}{A \& B} \ \&; \quad \frac{A, B}{B \& A} \ \&; \quad \frac{A}{A \vee B} \ \vee; \quad \frac{A}{B \vee A} \ \vee.$$

$\dfrac{A \vdash B}{A \Rightarrow B} \Rightarrow;$ $\qquad$ $\dfrac{A \vdash B,\ \neg B}{\neg A} \neg.$

Правилами исключения (удаления) являются:

$\dfrac{A \& B}{A}\ \&;$ $\quad \dfrac{A \& B}{B}\ \&;$

$\dfrac{A \vdash C,\ B \vdash C,\ A \vee B}{C} \vee;$ $\quad \dfrac{\neg\neg A}{A} \neg;$ $\quad \dfrac{A,\ A \Rightarrow B}{B} \Rightarrow$

Так как в естественном выводе нет аксиом, то доказательство должно основываться на правиле введения импликации, которое говорит, что если $B$ может быть выведено из $A$ по правилам вывода, то $A \Rightarrow B$ доказано. В формальной аксиоматической теории всякая формула вывода является либо аксиомой, либо непосредственным следствием по правилам вывода из предыдущих формул этого вывода. В доказательстве естественного вывода начинаем с гипотез (т.е. с недоказанных предложений), исследуем следствия, получающиеся по правилам вывода, и затем используем информацию вида $A \vdash B$ для того, чтобы заключить о доказуемости предложения $A \Rightarrow B$.

Например, докажем, что для любой формулы $A$ формула $\neg\neg A \Rightarrow A$ является теоремой. Это моментально следует, если применить правило исключения отрицания

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

т.е. $\neg\neg A \vdash A$, а затем правило введения импликации дает $\vdash \neg\neg A \Rightarrow A$, что и требовалось. Заметим, что при формальном аксиоматическом подходе (в теории $L$) доказательство того, что $\vdash \neg\neg A \Rightarrow A$ было получено, не столь просто.

Рассмотрим еще один пример на доказательство в естественном выводе. Докажем, что имеет место

$\vdash (A \vee (B \& C)) \Rightarrow (A \vee B) \& (A \vee C)$ (1)

Вид самой формулы для нас является подсказкой, как получить ее вывод (если он существует). Нам известно, что $\Rightarrow$ вводится с помощью правила введения импликации. Следовательно, прежде чем получить (1), нужно доказать, что

$(A \vee (B \& C)) \vdash (A \vee B) \& (A \vee C)$ (2)

Выражение (2) получим по правилу исключения дизъюнкции, если будет доказано

$A \vdash (A \vee B) \& (A \vee C)$ (3)

и

$B \& C \vdash (A \vee B) \& (A \vee C)$ (4)

Высказывание (3) будет доказано правилом введения конъюнкции, если будет доказано, что

$A \vdash (A \vee B)$ и $A \vdash (A \vee C)$, (5)

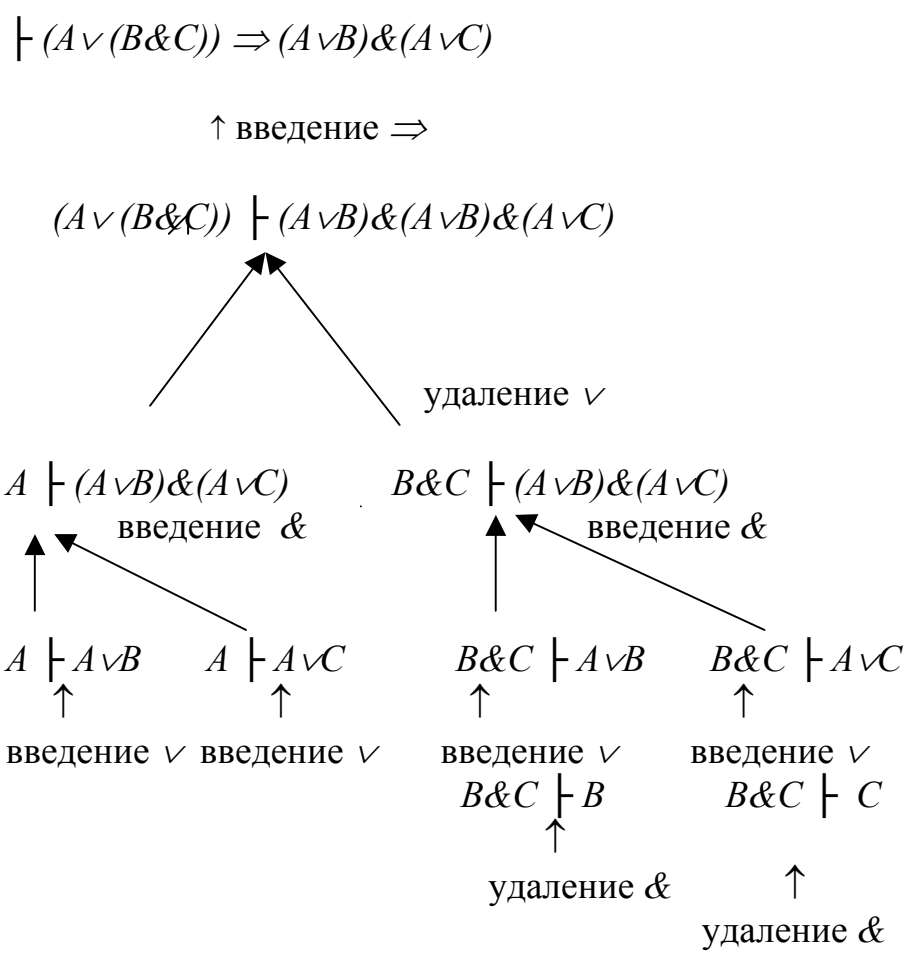
а (4) получим правилом введения конъюнкции из

$B\&C \vdash (A \vee B)$ и $B\&C \vdash (A \vee C)$ (6)

Ясно, что оба предложения в (5) доказуемы по правилу введения дизъюнкции.

Предложения (6) можно получить введением дизъюнкции из предложений $(B\&C) \vdash B$ и $(B\&C) \vdash C$, а последнее по правилу исключения конъюнкции.

Описанную выше процедуру можно свести в следующую схему:




Теперь доказательство формулы (1) восстанавливается, если проделать все рассуждения на приведенной схеме, начиная снизу.

Отметим, что доказательство предложения (1) в формальной аксиоматической теории при обычных аксиомах будет значительно более длинным и громоздким.

Для того, чтобы задать исчисление предикатов в виде естественного вывода, на заданные выше правила вывода накладывают некоторые ограничения и добавляют еще примерно столько же новых правил вывода.

## § 18. Вопросы и темы для самопроверки

1. Эффективные и полуэффективные методы.
2. Дедуктивные теории, их классификация.

3. Свойства дедуктивных теорий: непротиворечивость, полнота, независимость аксиом, разрешимость.
4. Полуформальные аксиоматические теории. Пример такой теории - геометрия. В чем отличие геометрии Евклида от геометрии Лобачевского-Бойяи-Гауса?
5. Формальные аксиоматические теории. Их задание, понятие вывода, теоремы, следствия.
6. Какие вы знаете свойства выводимости?
7. Исчисление высказываний. Задание. Конечно или бесконечно множество аксиом этой теории?
8. Является ли формула $A \Rightarrow A$ теоремой исчисления высказываний?
9. Укажите, какие из следующих формул являются теоремами исчисления высказываний:

а) $\neg\neg B \Rightarrow B$; б) $B \Rightarrow \neg\neg B$;
в) $\neg A \Rightarrow (\neg A \Rightarrow \neg B)$; г) $\neg B \Rightarrow B$;
д) $\neg A \Rightarrow (A \Rightarrow B)$; е) $(\neg B \Rightarrow \neg A) \Rightarrow (A \Rightarrow B)$;
ж) $(A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A)$; з) $A \Rightarrow (\neg B \Rightarrow \neg (A \Rightarrow B))$;
и) $(A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)$; к) $(A \Rightarrow B) \Rightarrow \neg B$.

10. Производные (доказуемые) правила вывода в исчислении высказываний.
11. Эквивалентность двух определений непротиворечивости для теорий содержащих исчисление высказываний.
12. Непротиворечивость исчисления высказываний.
13. Полнота исчислений высказываний.
14. Независимость схем аксиом исчисления высказываний.
15. Разрешимость исчисления высказываний.
16. Другие аксиоматизации исчисления высказываний.
17. Задание теорий первого порядка.
18. Исчисление предикатов первого порядка, его непротиворечивость.
19. Формальная арифметика.
20. Понятие о теоремах Геделя.
21. Значение аксиоматического метода.
22. Теория естественного вывода.

*Можно много видеть, читать, можно кое-что вообразить, но чтобы сделать, необходимо уметь, а умение дается только изучением техники.*

*М. Горький*

## § 19. Упражнения

1. Является ли выводом в исчислении высказываний следующая последовательность формул:
   1) $(\neg A \Rightarrow ((A \Rightarrow A) \Rightarrow \neg A)) \Rightarrow ((\neg A \Rightarrow (A \Rightarrow \neg A)) \Rightarrow (\neg A \Rightarrow \neg A))$,

2) $\neg A \Rightarrow ((A \Rightarrow \neg A) \Rightarrow \neg A)$,
3) $(\neg A \Rightarrow (A \Rightarrow \neg A)) \Rightarrow (\neg A \Rightarrow \neg A)$,
4) $\neg A \Rightarrow (A \Rightarrow \neg A)$,
5) $\neg A \Rightarrow \neg A$.

2. Доказать, что для любых формул $A$, $B$ исчисления высказываний следующие формулы являются теоремами исчисления высказываний:
   1) $(\neg B \Rightarrow \neg B)$; 2) $A \Rightarrow (\neg A \Rightarrow B)$;
   3) $(\neg B \Rightarrow \neg A) \Rightarrow (A \Rightarrow B)$; 4) $(A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A)$;
   5) $A \Rightarrow (\neg B \Rightarrow \neg (A \Rightarrow B))$; 6) $(A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)$.
3. Доказать, что в исчислении высказываний имеет место:
   1) $A \vdash A \vee B$; 2) $A \vdash B \vee A$;
   3) $A, B \vdash A \& B$; 4) $A, B \vdash B \& A$.
4. При доказательстве независимости *A1* от *A2* и *A3* в исчислении высказываний введены *выделенные формулы*, см. § 11 (свойства исчисления высказываний). Составить программу на одном из языков программирования для выяснения что формула исчисления высказываний, содержащая три пропозициональные буквы, является выделенной. Используя эту программу получить, что аксиома, полученная по *A2*, является выделенной.
5. При доказательстве независимости *A2* от *A1* и *A3* в исчислении высказываний введены гротескные формулы, см. § 11 (свойства исчисления высказываний). Составить программу на одном из языков программирования для выяснения, что формула исчисления высказываний содержащая две пропозициональные буквы является гротескной. Используя эту программу получить, что аксиомы, полученные по *A1* либо *A3*, являются гротескными.
6. Пусть в исчисление высказываний примитивными связками являются $\neg$, $\&$, $\vee$ и $\Rightarrow$. Формулы получены из пропозициональных букв с помощью этих примитивных связок. Каковы бы ни были формулы $A$, $B$ и $C$, следующие формулы суть аксиомы теории $L_1$:

   *A1:* $A \Rightarrow (B \Rightarrow A)$;
   *A2:* $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$;
   *A3:* $A \& B \Rightarrow A$;
   *A4:* $A \& B \Rightarrow B$;
   *A5:* $A \Rightarrow (B \Rightarrow (A \& B))$;
   *A6:* $A \Rightarrow (A \vee B)$;
   *A7:* $B \Rightarrow (A \vee B)$;
   *A8:* $(A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \vee B) \Rightarrow C))$;
   *A9:* $(A \Rightarrow B) \Rightarrow ((A \Rightarrow \neg B) \Rightarrow \neg A)$;
   *A10:* $\neg\neg A \Rightarrow A$.

Правилом вывода теории $L_1$ служит правило *modus ponens (MP)*. Доказать для теории $L_1$:

1) что формула $A \Rightarrow A$ является теоремой;

2) что верна теорема дедукции: если $\Gamma$ - множество формул, $A,B$ - формулы и $\Gamma,A \vdash B$, то: $\Gamma \vdash A \Rightarrow B$;
3) $(A \Rightarrow B),(B \Rightarrow C) \vdash (A \Rightarrow C)$;
4) что формула $B \Rightarrow (A \Rightarrow (A \vee B))$ является теоремой;
5) что формула $B \Rightarrow ((A \& B) \Rightarrow A)$ является теоремой;
6) что формула $A \Rightarrow (\neg A \Rightarrow A)$ является теоремой;
7) что формула $B \Rightarrow (\neg A \Rightarrow A)$ является теоремой.

7. Пусть формула $A$ теории первого порядка является частным случаем тавтологии. Доказать, что $A$ является теоремой в теории первого порядка.
8. Доказать следующую теорему дедукции для теорий первого порядка: Если $\Gamma$ - множество формул, $A,B$ - формулы и $\Gamma,A \vdash B$, и при этом существует такой вывод $B$ из $\{\Gamma,A\}$ в котором ни при каком применении правила обобщения к формулам, зависящим в этом выводе от $A$, не связывается квантором никакая свободная переменная формулы $A$. Тогда: $\Gamma \vdash A \Rightarrow B$.
9. Пусть $A$, $B$ формулы теории первого порядка. Доказать, что в теории первого порядка имеем: $\forall x_1 A \Rightarrow B \vdash \forall x_1 B$.
10. Пусть $A$ формула теории первого порядка. Доказать, что в теории первого порядка имеем: $\vdash \forall x_1 \forall x_2 A \Rightarrow \forall x_2 \forall x_1 A$.

*- Зато я получил классическое образование.*
*- Как это? – спросила Алиса.*
*- А вот как, - отвечал Грифон. – Мы с моим учителем, крабом – старичком, уходили на улицу и целый день играли в классики.*

*Л. Кэррол*

# Глава 5. НЕКЛАССИЧЕСКИЕ ЛОГИКИ

*Если бы это было так, это бы еще ничего, а если бы ничего, оно бы так и было, но так как это не так, так оно и не этак! Такова логика вещей!*

*Л. Кэррол*

## § 1. Трехзначные логики

До сих пор рассматривались высказывания, которые могли принимать лишь два значения: *И* либо *Л* (*1* либо *0*). Однако оказывается, что некоторые явления требуют для своего описания употребления высказываний, принимающих более двух значений.

Например, значением высказывания можно считать одно из трех значений: *истина, неопределенность (нейтрально)* и *ложь*, обозначаемые соответственно *И, Н* и *Л* или *1, ½* и *0*. Такие высказывания будем обозначать через *x, y, z* и т.д., а также этими буквами с числовыми индексами. Их значения в дальнейшем будем записывать символами *1, ½* и *0* соответственно.

В двузначной логике отрицание *истины* есть *ложь*, а отрицание *лжи* вводится как *истина*. Эти определения интуитивно очевидны и однозначны. Для трехзначной логики уже на этапе определения отрицания интуитивно неясно, как, например, ввести отрицание *неопределённости*. В настоящее время имеются разные варианты трёхзначных логик. Рассмотрим некоторые трёхзначные системы.

***Трёхзначная логика Лукасевича***.

Рассмотрим множества высказываний, каждое из которых может принимать только одно из трёх значений: *1, ½, 0*. В качестве операций в трёхзначной логике Лукасевича введены отрицание, обозначаемое *Nx*, конъюнкция (*Kxy*), дизъюнкция (*Axy*), импликация (*Cxy*). Эти операции определены следующим образом:

*Nx=1-x, Kxy=min(x, y), Axy=max(x, y),*

*Cxy=min(1, 1–x+y),* т.е.*: Cxy=1*, если $x \le y$; *Cxy=1–x+y,* если $x>y$. Для проведения сравнений различных логик будем использовать обозначения использовавшиеся для классической логики: для конъюнкции – *«x&y»*, для

дизъюнкции – *«x∨y»*, для импликации – *«x⇒y»* и для эквивалентности – *«x≡y»*. Согласно введенным определениям, получим следующую таблицу истинности. В этой таблице введем также операцию эквивалентности Лукасевича [15].

| *x* | *y* | *Nx* | *x&y* | *x∨y* | *x⇒y* | *x≡y* |
|---|---|---|---|---|---|---|
| *0* | *0* | *1* | *0* | *0* | *1* | *1* |
| *0* | *½* | | *0* | *½* | *1* | *½* |
| *0* | *1* | | *0* | *1* | *1* | *0* |
| *½* | *0* | *½* | *0* | *½* | *½* | *½* |
| *½* | *½* | | *½* | *½* | *1* | *1* |
| *½* | *1* | | *½* | *1* | *1* | *½* |
| *1* | *0* | *0* | *0* | *1* | *0* | *0* |
| *1* | *½* | | *½* | *1* | *½* | *½* |
| *1* | *1* | | *1* | *1* | *1* | *1* |

Рассмотрим, например выражение *N(x&y)*. Легко видеть, что

*N(x&y)= 1- min (x,y)= max(1-x,1-y)=(Nx)∨(Ny).*

Аналогичным образом можно получить, что *N(x∨y)=(Nx)&(Ny)*. Следовательно, в этой логике выполняются законы де Моргана. Имеются и другие сходства с двузначной логикой, но есть и различия, например, не выполняется закон исключённого третьего, т.е. *x∨(Nx)* не всегда истинно; есть и другие различия.

***Трёхзначная логика Гейтинга.***

В двузначной логике являются тавтологиями как *x⇒¬¬x*, так и *¬¬x⇒x*. Из предположения, что тавтологией можно считать только формулу *x⇒¬¬x* Гейтинг разработал новую трёхзначную логику. Пусть имеем вновь множество высказываний, каждое из которых принимает одно из значений: *1, ½* и *0*.

Операции Гейтинга вводятся согласно следующей таблице.

Из таблицы видно, что конъюнкция и дизъюнкция определены следующим образом:

*x& y=min(x, y), x∨y=max(x, y),*

а импликация по формуле: *x⇒y=1,* если *x≤y, x⇒y=y*, если *x>y*.

В этой логике, как и в логике Лукасевича, если оставить только значения *0* и *1* (исключить третье значение – значение *½*), то получим обычную двузначную логику.

| $x$ | $y$ | $Nx$ | $x\&y$ | $x \vee y$ | $x \Rightarrow y$ | $x \equiv y$ |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | ½ | | 0 | ½ | 1 | 0 |
| 0 | 1 | | 0 | 1 | 1 | 0 |
| ½ | 0 | ½ | 0 | ½ | 0 | 0 |
| ½ | ½ | | ½ | ½ | 1 | 1 |
| ½ | 1 | | ½ | 1 | 1 | ½ |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | ½ | | ½ | 1 | ½ | ½ |
| 1 | 1 | | 1 | 1 | 1 | 1 |

***Трёхзначные логики Рейхенбаха, Бочвара и Клини***

В настоящее время имеется много вариантов построения трёхзначных логик, наиболее известными являются пять трёхзначных логик. К ним относятся указанные логики Лукасевича и Гейтинга, а также трёхзначные логики Рейхенбаха, Бочвара и Клини. Для сравнения этих логик положим, что *истина*, *неопределённость* и *ложь* обозначены через *1*, *½* и *0,* соответственно. Отметим, что их создатели обозначили указанные значения по разному. Также единым образом обозначим операции: конъюнкцию – «*&*», дизъюнкцию – «$\vee$», импликацию – «$\Rightarrow$» и эквивалентность – «$\equiv$». В каждой из этих логик есть отрицание, такое, что $\bar{x}=1-x$ (выше в логиках Лукасевича и Гейтинга отрицания для $x$ обозначались через «$Nx$»). Значения для конъюнкции, дизъюнкции, импликации и эквивалентности высказываний определяется по следующей таблице [15].

| $x$ | $y$ | Трёхзначные логики | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Рейхенбаха | | | | Бочвара | | | | Клини | | | |
| | | & | $\vee$ | $\Rightarrow$ | $\equiv$ | & | $\vee$ | $\Rightarrow$ | $\equiv$ | & | $\vee$ | $\Rightarrow$ | $\equiv$ |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | ½ | 0 | ½ | 1 | ½ | ½ | ½ | ½ | ½ | 0 | ½ | 1 | ½ |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| ½ | 0 | 0 | ½ | ½ | ½ | ½ | ½ | ½ | ½ | 0 | ½ | ½ | ½ |
| ½ | ½ | ½ | ½ | 1 | 1 | ½ | ½ | ½ | ½ | ½ | ½ | ½ | ½ |
| ½ | 1 | ½ | 1 | 1 | ½ | ½ | ½ | ½ | ½ | ½ | 1 | 1 | ½ |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | ½ | ½ | 1 | ½ | ½ | ½ | ½ | ½ | ½ | ½ | 1 | ½ | ½ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Из определений операций видно, что во всех этих пяти логиках значения операций совпадают с их значениями для классической двузначной логики, когда аргументы $x$ и $y$ принимают значения из $\{0, 1\}$ и различаются когда значение хотя бы одного из аргументов $x, y$ принимают значение ½. Легко убедиться, что ни в одной из этих пяти логик не выполняется закон

противоречия, т.е. $(x\&\bar{x})$ не всегда ложно, также не имеет места закон исключённого третьего, т.е. $(x\vee\bar{x})$ не всегда истинно.

Указанные логики вводились для различных целей. Так, например, Рейхенбах построил свою логику для описания явлений квантовой механики. По его мнению, говорить об истинном или ложном высказывании правомерно лишь тогда, когда возможно осуществить их проверку. Если нельзя ни подтвердить истинность высказывания, ни опровергнуть его с помощью проверки, то такое высказывание должно оцениваться третьим значением – неопределенно. К числу таких высказываний относятся высказывания о ненаблюдаемых объектах в микромире.

Рейхенбах обозначал «истину» как *1*, «неопределённость» – *2*, «ложность» – *3*. Тавтология принимает значение *1*. В логике Рейхенбаха введены три отрицания:

- циклическое отрицание, обозначаемое «*~A*»;
- диаметральное отрицание, которое обозначим через «$\bar{A}$»;
- полное отрицание, которое обозначим через «*–A*».

Эти операции вводятся по следующей таблице (здесь значения *1*, *2* и *3* вновь переобозначены через *1, ½* и *0*).

| $A$ | $\sim A$ | $\bar{A}$ | $-A$ |
|---|---|---|---|
| *1* | *½* | *0* | *½* |
| *½* | *0* | *½* | *1* |
| *0* | *1* | *1* | *1* |

В логике Рейхенбаха кроме указанных операций, введены ещё альтернативная импликация, квазиимпликация и альтернативная эквивалентность.

Легко проверить, что для циклического отрицания не имеет место закон двойного отрицания, а имеет место закон тройного отрицания, т.е. *~(~(~A))* равносильно *A*. Также для циклического отрицания не имеет место закон исключённого третьего, а имеет место закон исключённого четвёртого, который означает, что $A\vee(\sim A)\vee(\sim(\sim A))$ всегда истинно.

Для диаметрального отрицания, как уже указано, сохраняется закон двойного отрицания, но не имеет место закон исключённого третьего. Имеются и много других, как отличий, так и совпадений этой логики и обычной двузначной логики.

*Кто мешает тебе выдумать порох непромокаемый.*

*Козьма Прутков*

## § 2. Многозначные логики

***Конечнозначная (k-значная, k≥2) логика Поста*** является обобщением двузначной логики, т.е. при $k=2$ получится двузначная логика.

Рассмотрим множество высказываний (переменных), каждое из которых может принимать одно из значений $0,1,2,\ldots,k-1$.

На множестве введённых $k$ – значных высказываний вводятся операции:

1) $\bar{x}=x+1(mod\ k)$ – циклическое отрицание или отрицание Поста, здесь + - сложение по модулю $k$;

2) $Nx=k-1-x$ – отрицание Лукасевича;

3) $I_m(x)=\begin{cases} k-1, & если\ x=m, \\ 0, & если\ x\neq m, \end{cases}$ $m=0,1,\ldots,k-1.$

Функция $I_m(x)$ называется иногда характеристической функцией и обозначается как $x^m$;

4) $x\&y=min(x,y)$ – конъюнкция;

5) $x\vee y=max(x,y)$ - дизъюнкция;

6) $x\times y=x\times y\ (mod\ k)$ – произведение по модулю $k$;

7) $x+y=x+y\ (mod\ k)$ – сумма по модулю $k$;

8) $x\Rightarrow y=\begin{cases} k-1, & если\ 0\le x<y\le k-1, \\ (k-1)-x+y, & если\ 0\le y\le x\le k-1. \end{cases}$

Водятся и другие операции.

Используя введённые операции можно строить суперпозиции этих функций, исследовать их свойства. Также можно вводить нормальные формы, доказать следующее соотношение:

$$f(x_1,\ldots,x_n)=\bigvee_{(a_1,a_2,\ldots,a_n)} I_{a_1}\&\ldots\&I_{a_n}\&f(a_1,a_2,\ldots,a_n),$$

где дизъюнкция берётся по всевозможным наборам значений $(a_1,a_2,\ldots,a_n)$ переменных $(x_1,x_2,\ldots,x_n)$.

Легко доказать теорему:

**Теорема 5.1.** Число различных функций $k$-значной логики, зависящих от $n$ переменных равно $k^{k^n}$.

Система функций $k$-значной логики $\{\varphi_1,\varphi_2,\ldots,\varphi_m\}$ называется функционально полной, если любую функцию $f$ $k$-значной логики можно выразить через функции из $\{\varphi_1,\varphi_2,\ldots,\varphi_m\}$.

Существует и критерий полноты системы функций.

**Теорема 5.2** (о функциональной полноте, теорема А. В. Кузнецова). Для каждой $k$-значной логики существует конечное число замкнутых классов $K_1, K_2, \ldots, K_{r(k)}$ таких, что для полноты системы функций $k$-значной логики $\{\varphi_1, \varphi_2, \ldots, \varphi_m\}$ необходимо и достаточно чтобы $\{\varphi_1, \varphi_2, \ldots, \varphi_m\}$ не содержалась целиком ни в одном из классов $K_1, K_2, \ldots, K_{r(k)}$.

Отметим, что в $k$-значных логиках сохраняются многие свойства и результаты, которые имели место в двузначной логике, но есть и существенные отличия от двузначной логики.

***Многозначная логика Лукасевича.*** В отличие от $k$-значной логики Поста в $k$-значной логике Лукасевича считается, что истинностные значения переменных образуют следующее множество:

$$T_k = \left\{ 0 = \frac{0}{k-1}, \quad \frac{1}{k-1}, \quad \frac{2}{k-1}, \ \ldots, \frac{k-2}{k-1}, \quad \frac{k-1}{k-1} = 1 \right\}. \qquad (5.1)$$

Эти истинностные значения можно интерпретировать как степень (уровень) истинности.

Операции определяются следующим образом.

$$\begin{aligned} &Nx = 1 - x; \\ &x \& y = \min(x, y); \\ &x \vee y = \max(x, y). \end{aligned} \qquad (5.2)$$

Операции импликации и эквивалентности вводится по формулам: $x \Rightarrow y = \min(1, 1 + y - x)$ и $x \equiv y = 1 - |x - y|$. Отметим, что Лукасевич вводил только отрицание и импликацию, а остальные записывал через них.

Для каждого $k$, $k \geq 2$, $k$-значная логика Лукасевича обозначается как $L_k$. В последовательности $L_2, L_3, L_4, \ldots$ этих логик $L_2$ является классической двузначной логикой, логика $L_3$ совпадает с трёхзначной логикой Лукасевича, рассмотренной в предыдущем параграфе. Предельный случай – логика $L_\infty$ является бесконечнозначной логикой, для которой истинностными значениями являются все рациональные числа единичного отрезка $[0,1]$, а операции вводятся по (5.2).

Рассматриваются также и другие многозначные логики, для которых истинностными значениями являются числа отрезка $[0,1]$, но определяемые по формулам отличным от (5.1), операции вводятся как по (5.2), так и по другим формулам. Кроме того, рассматриваются бесконечнозначные логики, для которых истинностными значениями являются уже все действительные числа единичного отрезка $[0,1]$ для которых операции вводятся как по формулам (5.2) так и по формулам отличным от (5.2). Логика, в которой истинностными значениями являются все действительные числа единичного

отрезка *[0,1]*, а операции вводятся по формулам (5.2), считается стандартной логикой Лукасевича – логикой $L_1$ [15].

*Такие слова наводят на всякие мысли, хотя и неясно – на какие (Алиса)*[1].

*Л. Кэррол*

## § 3. Понятие нечеткого множества

Основателем теории нечетких множеств является Л. Заде. Л. Заде писал (см. предисловие к книге [16]): «Теория нечетких множеств – это, по сути дела, шаг на пути к сближению точности классической математики и всепроникающей неточности реального мира, к сближению, порожденному непрекращающимся человеческим стремлением к лучшему пониманию процессов мышления и познания»

Пусть $U$ - произвольное непустое множество в обычном понимании (иногда называемое универсальным множеством), а $A$ является его подмножеством, $A \subseteq U$.

Тот факт, что элемент $x$ множества $U$ есть элемент подмножества $A$ или, как говорят, принадлежит $A$, обычно обозначают так $x \in A$. Для выражения этой принадлежности можно использовать и другое понятие - характеристическую функцию $\mu_A(x)$, значения которой указывают, является ли (да или нет) $x$ элементом $A$:

$$\mu_A(x) = \begin{cases} 1, & \text{если } x \in A, \\ 0, & \text{если } x \notin A. \end{cases}$$

Рассмотрим пример. Пусть $U=(-\infty, \infty)$, $A=[-2,3]$, тогда $\mu_A(x)$ имеет вид изображенный на рис. 5.1.

Очевидны следующие свойства характеристических функций:

1) $(A=B)$ тогда и только тогда, когда $\forall x(\mu_A(x) = \mu_B(x))$;

2) $\mu_{CA}(x) = 1 - \mu_A(x)$;

3) $\mu_{A \cap B}(x) = \begin{cases} 1, & \text{если } x \in A \cap B, \\ 0, & \text{если } x \notin A \cap B \end{cases} =$

$= min(\mu_A(x), \mu_B(x))$;

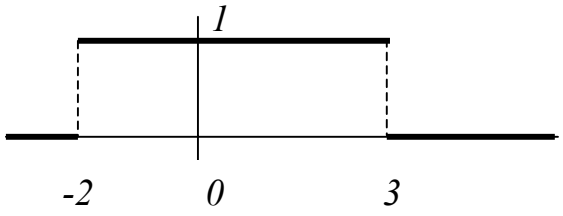



Рис. 5.1.

[1] *Эта фраза Алисы из «Алисы в Зазеркалье» взята из книги: М. Гэри, Д. Джонсон, Вычислительные машины и труднорешаемые задачи, см [10]. В переводе Н. Димуровой произведения «Алиса в Зазеркалье» фраза Алисы приводиться без первых двух слов эпиграфа, да и остальная часть* ***нечетко*** *совпадает с эпиграфом, даже без указанной части.*

4) $\mu_{A\cup B}(x)=\begin{cases}1, & \text{если } x\in A\cup B,\\ 0, & \text{если } x\notin A\cup B\end{cases}$ $= max(\mu_A(x),\ \mu_B(x)).$

Отметим, что рассмотренная характеристическая функция принимает только два значения *0* или *1*.

Представим теперь, что характеристическая функция $\mu(x)$ может принимать любое значение в интервале *[0,1]*. В соответствии с этим мера принадлежности *x* подмножеству *A* может быть любой из *[0,1]*, т.е. *x* может быть элементом *A* более или менее, менее чем более и т.п. Таким образом, понятие принадлежности получает интересное обобщение. Дадим строгое определение.

Пусть *U* - множество и *x* элемент *U*. Тогда *нечетким подмножеством* *A** множества *U* называется множество упорядоченных пар

$A^*=\{(x,\ \mu_{A^*}(x))\}$, где $x\in U,\ \mu_{A^*}(x)\in[0,1];$

функцию $\mu_{A^*}(x)$ называют *функцией принадлежности*, а *U- универсальным* или *базовым множеством*.

Рассмотрим множество людей различного возраста и попытаемся выделить подмножество молодых людей, т.е. задать функцию $\mu(x)$, $0\le\mu(x)\le1$. Ясно, что каждый может ввести свое понимание функции $\mu(x)$. На рис. 5.2 приведены графики некоторых возможных таких функций $\mu(x)$; при этом на оси *0x* указаны возраст в годах, на *0y* – значения функции $\mu(x)$.

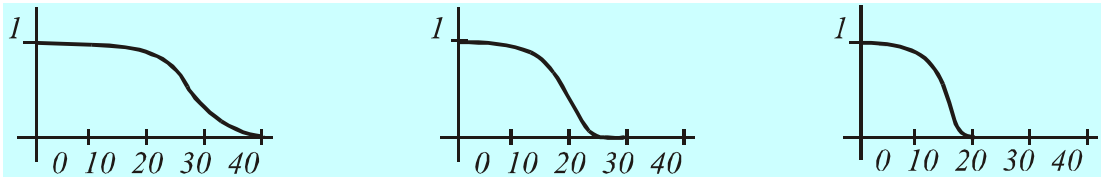



Рис. 5.2.

На множестве (-∞,∞), например, можно ввести понятие действительных чисел очень близких к нулю. Например, можно определить функцию принадлежности $\mu_{A^*}(x)$ нечеткого подмножества *A** действительных чисел очень близких к нулю по формуле: $\mu_{A^*}(x)=\dfrac{1}{1+10x^2}$.

График этой функции, представлен на Рис. 5.3.

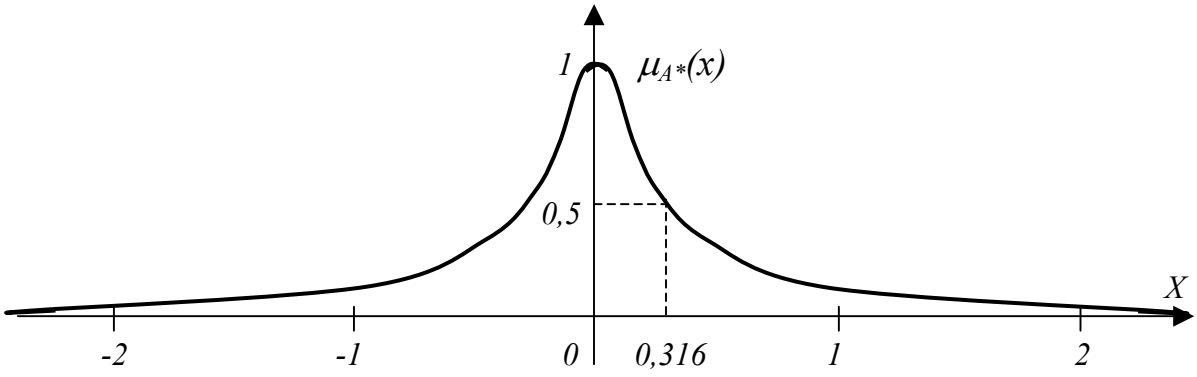



Рис. 5.3.

Ясно, что понятие действительных чисел очень близких к нулю тоже вводится неоднозначно, следовательно, и в этом случае можно получить различные функции принадлежности $\mu_{A^*}(x)$. Таким образом, выбор функции $\mu_{A^*}(x)$, в общем, может быть различным.

*Носителем нечеткого подмножества* $A^*$ называется (обычное) подмножество $B$ множества $U$, содержащее те элементы из $U$, для которых $\mu_{A^*}(x)>0$. Носитель для $A^*$ обозначают как $suppA^*$ $(suppA^*=\{x \in U: \mu_{A^*}(x)>0\})$.

Два нечетких подмножества $A^*$ и $B^*$ множества $U$ называются равными тогда и только тогда, когда $\forall x \in U: \mu_{A^*}(x) = \mu_{B^*}(x)$.

Будем говорить, что $A^*$ содержится в $B^*$, если

$$\forall x \in U: \mu_{A^*}(x) \le \mu_{B^*}(x)$$

и обозначать $A^* \subseteq B^*$.

Считаем, что нечеткие подмножества $A^*$ и $B^*$ множества $U$ дополняют друг друга, если

$$\forall x \in U: \mu_{A^*}(x) = 1 - \mu_{B^*}(x). \qquad (5.3)$$

и обозначать: $B^* = \overline{A}^*$ или $A^* = \overline{B}^*$.

Введем пересечение ($\cap$) и объединение ($\cup$) нечётких подмножеств.

Пусть $A^*$ и $B^*$ нечеткие подмножества множества $U$, тогда их пересечение и объединение есть нечеткие подмножества множества $U$ имеющие соответственно следующие функции принадлежности:

$$\forall x \in U: \mu_{A^* \cap B^*}(x) = \min(\mu_{A^*}(x), \mu_{B^*}(x)), \qquad (5.4)$$

$$\forall x \in U: \mu_{A^* \cup B^*}(x) = \max(\mu_{A^*}(x), \mu_{B^*}(x)). \qquad (5.5)$$

Положим, что универсальное множество $U$ является конечным множеством, например, $U=\{u_1,u_2,\ldots,u_n\}$ и $A^*$ его нечёткое подмножество с функцией принадлежности $\mu_{A^*}(x)$. Тогда имеем $A^*=\{\langle x, \mu_{A^*}(x)\rangle\}$, где $x \in U$, $\mu_{A^*}(x) \in [0,1]$, т. е. $A^*=\{\langle u_1, \mu_1\rangle, \langle u_2, \mu_2\rangle, \ldots, \langle u_n, \mu_n\rangle\}$, где $\mu_i = \mu_{A^*}(u_i)$. В таких случаях используют специальную форму записи, именно пишут:

$$A^* = \mu_1/u_1 + \mu_2/u_2 + \ldots + \mu_n/u_n, \qquad (5.6)$$

или

$$A^* = \sum_{i=1}^{n} \mu_i / u_i .$$

В этих записях указывается элемент универсального множества $u_i$ $(u_i \in U)$ и $\mu_i$ $(\mu_i = \mu_{A^*}(u_i))$ – степень принадлежности элемента $u_i$ нечёткому подмножеству $A^*$. В этих записях символ «+» не означает операцию сложения, а служит разделителем элементов множества $A^*$. Если $\mu_i = 0$, то, как правило, элементы

$\mu_i/u_i$ в (5.6) опускаются. Рассмотрим пример. Пусть универсальное множество $U$ состоит из 10 элементов, означающих возраст (в годах):

$U=\{5, 10, 20, 30, 40, 50, 60, 70, 80, 90\}$.

На этом множестве с помощью следующей таблицы заданы нечёткие подмножества, характеризуемые словами *молодой, взрослый* и *старый.* В таблице для каждого элемента $u_i$ из $U$ указаны степени принадлежности $u_i$ указанным нечетким подмножествам.

| Элементы из $U$ (годы) | Нечёткие подмножества | | |
|---|---|---|---|
| | *Молодой* | *Пожилой* | *Старый* |
| *5* | *1* | *0* | *0* |
| *10* | *1* | *0* | *0* |
| *20* | *0,8* | *0,1* | *0* |
| *30* | *0,5* | *0,3* | *0,1* |
| *40* | *0,2* | *0,5* | *0,3* |
| *50* | *0,1* | *0,7* | *0,5* |
| *60* | *0* | *1* | *0,8* |
| *70* | *0* | *1* | *1* |
| *80* | *0* | *1* | *1* |
| *90* | *0* | *1* | *1* |

Из таблицы можно записать, что носитель нечёткого подмножества $A^*$-*молодой* равно:

$supp(A^*)=supp(молодой)=\{u_i \in U: \mu_i > 0\}=\{5, 10, 20, 30, 40,50\}$.

Нечёткое подмножество $A^*$ можно записать в виде:

$A^*=1/5+1/10+0,8/20+0,5/30+0,2/40+0,1/50$.

Если $B^*$ нечёткое подмножество *пожилой*, то записываем:

$B^*=0,1/20+0,3/30+0,5/40+0,7/50+1/60+1/70+1/80+1/90$.

Пересечение этих подмножеств, очевидно, равно следующему:

$A^* \cap B^*=0,1/20+0,3/30+0,2/40+0,1/50$.

Известно, что для произвольных (обычных, чётких) подмножеств $A$, $B$, и $C$ множества $U$ выполняются следующие соотношения.

$\overline{\overline{A}}=A$ – инволютивность;

$\left.\begin{array}{l} A \cup B=B \cup A \\ A \cap B=B \cap A \end{array}\right\}$ – коммутативность;

$\left.\begin{array}{l} A \cup (B \cup C)=(A \cup B) \cup C \\ A \cap (B \cap C)=(A \cap B) \cap C \end{array}\right\}$ – ассоциативность;

$\left.\begin{array}{l} A \cup (B \cap C)=(A \cup B) \cap (A \cup C) \\ A \cap (B \cup C)=(A \cap B) \cup (A \cap C) \end{array}\right\}$ – дистрибутивность;

$\left.\begin{array}{l} A \cup \varnothing=A \\ A \cup U=U \\ A \cap \varnothing=\varnothing \\ A \cap U=A \end{array}\right\}$ – свойства операций с $\varnothing$ и с $U$;

$\left.\begin{array}{l}\overline{A \cup B} = \bar{A} \cap \bar{B} \\ \overline{A \cap B} = \bar{A} \cup \bar{B}\end{array}\right\}$ – законы де Моргана;

$\left.\begin{array}{l}A \cup A = A \\ A \cap A = A\end{array}\right\}$ –законы идемпотентности;

$\left.\begin{array}{l}A \cap (A \cup B) = A \\ A \cup (A \cap B) = A\end{array}\right\}$ - законы поглощения;

$\left.\begin{array}{l}A \cup \bar{A} = U \\ A \cap \bar{A} = \varnothing\end{array}\right\}$ – свойства дополнения;

Если $A^*$, $B^*$, и $C^*$ нечеткие подмножества универсального (обычного) множества $U$, то можно доказать, что выполняются все приведённые свойства за исключением последних двух соотношений (свойства дополнения), т. е. для нечетких подмножеств имеем следующие соотношения.

1) $\bar{\bar{A}}^* = A^*$ – инволютивность;

$\left.\begin{array}{ll}2) & A^* \cup B^* = B^* \cup A^* \\ 3) & A^* \cap B^* = B^* \cap A^*\end{array}\right\}$ – коммутативность;

$\left.\begin{array}{ll}4) & A^* \cup (B^* \cup C^*) = (A^* \cup B^*) \cup C^* \\ 5) & A^* \cap (B^* \cap C^*) = (A^* \cap B^*) \cap C^*\end{array}\right\}$ – ассоциативность;

$\left.\begin{array}{ll}6) & A^* \cup (B^* \cap C^*) = (A^* \cup B^*) \cap (A^* \cup C^*) \\ 7) & A^* \cap (B^* \cup C^*) = (A^* \cap B^*) \cup (A^* \cap C^*)\end{array}\right\}$ – дистрибутивность;

$\left.\begin{array}{ll}8) & A^* \cup \varnothing = A^* \\ 9) & A^* \cup U = U \\ 10) & A^* \cap \varnothing = \varnothing \\ 11) & A^* \cap U = A^*\end{array}\right\}$ – свойства операций с $\varnothing$ и $U$;

$\left.\begin{array}{ll}12) & \overline{A^* \cup B^*} = \bar{A}^* \cap \bar{B}^* \\ 13) & \overline{A^* \cap B^*} = \bar{A}^* \cup \bar{B}^*\end{array}\right\}$ – законы де Моргана;

$\left.\begin{array}{ll}14) & A^* \cup A^* = A^* \\ 15) & A^* \cap A^* = A^*\end{array}\right\}$ – законы идемпотентности;

$\left.\begin{array}{ll}16) & A^* \cap (A^* \cup B^*) = A^* \\ 17) & A^* \cup (A^* \cap B^*) = A^*\end{array}\right\}$ - законы поглощения.

Здесь $U$ является обычным множеством, для которого полагаем, что его характеристическая функция введена как: $\mu_U(x)=1$ для всех $x \in U$. Множество $\varnothing$ тоже является обычным множеством, для которого полагаем, что его характеристическая функция введена как: $\mu_\varnothing(x)=0$ для всех $x \in U$.

Как уже указано, свойства дополнения в общем случае не выполняются, т.е. существуют $A^*$ и $B^*$ такие, что:

$$A^* \cup \bar{A}^* \neq U,$$
$$B^* \cap \bar{B}^* \neq \varnothing.$$

Используя приведенные соотношения 1)-17), можно проводить упрощения, преобразования.

*Однажды Насреддина спросили:*
*- Ходжа, где истина?*
*- Я не вижу места, где бы она отсутствовала, так что не могу указать точно место, где она находится.*
*- Ходжа Насреддин*[1]

## § 4. Нечеткие высказывания и максиминные операции над ними

*Нечетким высказыванием* называется предложение, относительно которого можно судить о степени его истинности или ложности. *Степень истинности* или *степень ложности* каждого нечеткого высказывания принимает значение из замкнутого интервала *[0,1]*, причем *0* и *1* являются их предельными значениями и совпадают с понятиями лжи и истины для "четких" высказываний. Степень истинности (степень ложности) каждого нечеткого высказывания может принимать, как только некоторые значения из *[0,1],* так и все значения из *[0,1]*.

Примеры нечетких высказываний:

"Два - маленькое число".

"Петров занимается большой общественной работой".

"Волга - хорошая машина".

"Молодая была не молода".

Нечеткие высказывания бывают простыми и составными. Составные высказывания образуются из простых с помощью вводимых операций, таких как отрицание, конъюнкция, дизъюнкция, импликация и других. Операции могут вводиться различными способами. Рассмотрим следующий вариант введения операций.

Отрицанием нечеткого высказывания $A^*$ называется нечеткое высказывание, обозначаемое $\neg A^*$, степень истинности которого определяется выражением

$$\neg A^* = 1 - A^*.$$

Конъюнкцией нечетких высказываний $A^*$, $B^*$ называется нечеткое высказывание, обозначаемое $A^* \& B^*$, степень истинности которого определяется следующим образом:

$$A^* \& B^* = \min(A^*, B^*).$$

Дизъюнкцией нечетких высказываний $A^*$, $B^*$ называется нечеткое высказывание, обозначаемое $A^* \vee B^*$, степень истинности которого находится как

$$A^* \vee B^* = \max(A^*, B^*).$$

[1] *Отметим, что приведенное имя Насреддина не единственно. Его называют в различных краях то Ходжой или Моллой, или Хасаном и, даже, Джохой Насреддином.*

Импликацией нечетких высказываний *A*, B** называется нечеткое высказывание, обозначаемое $A^* \Rightarrow B^*$, степень истинности которого определяется выражением

$$A^* \Rightarrow B^* = max(1-A^*, B^*).$$

Эквивалентностью нечетких высказываний *A*, B** называется нечеткое высказывание, обозначаемое $A^* \equiv B^*$, степень истинности которого определяется выражением

$$A^* \equiv B^* = min(max(1-A^*, B^*), max(1-B^*, A^*)).$$

Введенная нечеткая логика называется нечеткой логикой с максиминными операциями.

Рассматривая *A*, B*, C** и т.д. как нечёткие переменные (пропозициональные буквы), можно ввести понятие формулы в нечеткой логике точно также как вводились пропозициональные формы (формулы логики высказываний). Истинностные значения этих формул определяются согласно соотношений введенных для $\neg$, $\&$, $\vee$, $\Rightarrow$ и $\equiv$. Так, например, имеем:

$$A^* \vee \neg A^* = max(A^*, 1-A^*) = \begin{cases} A^*, & если \ \ A^* \geq 0{,}5 \\ 1 - A^*, & если \ \ A^* < 0{,}5. \end{cases} \qquad (5.7)$$

Из (5.7) следует, что значение $A^* \vee \neg A^*$ всегда не меньше *0,5*. Рассмотрим теперь формулу:

$$A^* \& \neg A^* = min(A^*, 1-A^*) = \begin{cases} A^*, & если \ \ A^* \leq 0{,}5 \\ 1 - A^*, & если \ \ A^* > 0{,}5. \end{cases}$$

Таким образом, истинностное значение для $A^* \& \neg A^*$ будет всегда не больше *0,5*.

Пусть нечеткое подмножество *M* молодых людей задано функцией принадлежности:

$$\mu_{молодой}(x) = \begin{cases} 1, & если \ \ x \in [0;20], \\ \left[1 + \left(\frac{x-20}{4}\right)^2\right]^{-1}, & если \ \ x > 20. \end{cases}$$

Значение функции принадлежности для выбранного значения *x*, положим *x* = Тина, можно рассматривать как истинностное значение для нечёткого высказывания «Тина молода». Тогда истинностное значение нечеткого высказывания «Тина молода» будет равно *0,63*, если ей 25 лет, см. Рис. 5.4 а). Если же Алёне 18 лет, то истинностное значение высказывания «Алёна молода» равно *1*.

Используя нечёткое подмножество *M* можно ввести *нечёткий предикат*, например: «*x* является молодым человеком», т.е.: «$x \in M$».

Можно строить и другие нечёткие предикаты, используя, например, понятия: *пожилой, старый, редкий, красивая, дорогой* и т.п.

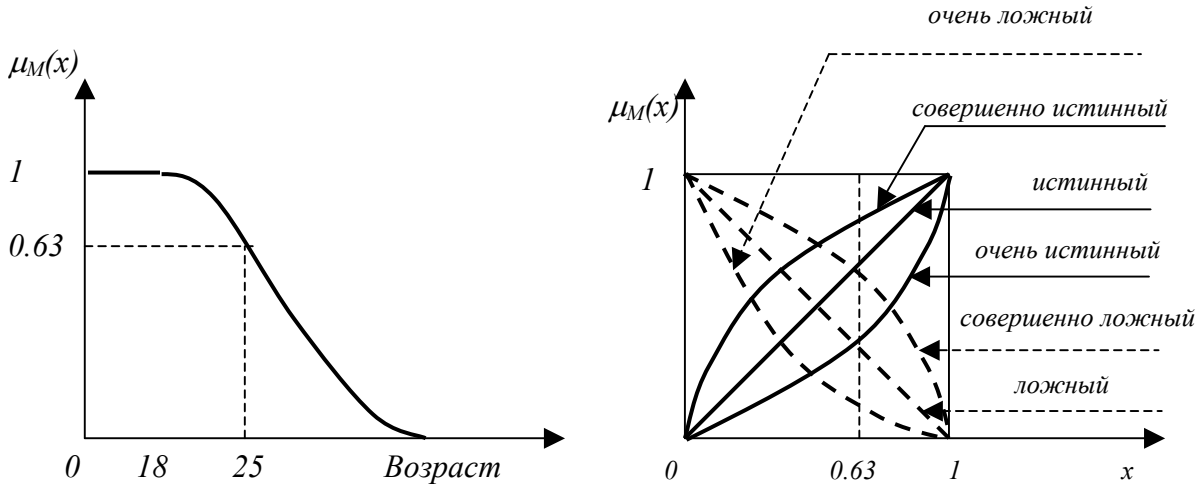
Кроме нечётких предикатов, можно ввести нечёткие кванторы (*почти все, много, несколько* и т.п.), а также нечёткие истинностные значения

(*совершенно истинный, очень истинный, истинный, совершенно ложный, очень ложный, ложный* и т.п.).

Указанные нечёткие истинностные значения вводятся с помощью подходящих нечётких подмножеств, а они – с помощью функций принадлежности. Например, можно положить, что для $x \in [0,1]$ имеем:

$$\begin{aligned} &\mu_{\text{совершенно истинный}}(x)=\sqrt{x}\ ; && \mu_{\text{совершенно ложный}}(x)=\sqrt{1-x}\ ; \\ &\mu_{\text{очень истинный}}(x)=x^2; && \mu_{\text{очень ложный}}(x)=(1-x)^2; \\ &\mu_{\text{истинный}}(x)=x; && \mu_{\text{ложный}}(x)=1-x, \end{aligned} \qquad (5.8)$$

а вне отрезка *[0,1]* значения этих функций равны нулю. Графики этих функций представлены на Рис. 5.4 б).




а) б)

Рис. 5.4

Рассмотрим высказывания:

1) «Тина молода» - совершенно истинно;
2) «Тина молода» - истинно;
3) «Тина молода» - очень истинно;
4) «Тина молода» - очень ложно;
5) «Тина молода» - ложно;
6) «Тина молода» - совершенно ложно.

Для выяснения степени истинности (истинностных значений) высказываний 1)-6) сначала выясним значение $y = \mu_{\text{молодой}}(x)$, затем по $y$ вычисляем значения $\mu(y)$ по (5.8). Пусть Тине 25 лет. Тогда

$$\mu_{\text{молодой}}(25)=0{,}63.$$

Подставляя значение *0,63* в функции (5.8) получим, что истинностные значения для высказываний 1)-6) равны соответственно:

*0,79; 0,63; 0,4; 0,14; 0,37; 0,61.*

Пусть $X=2^A$ множество подмножеств множества $A$ $(A\neq\varnothing)$ и на $X$ заданы обычные операции дополнения, пересечения и объединения, т.е. имеем алгебру $\boldsymbol{A}=\langle X;\ \bar{\ },\ \cap,\ \cup\rangle$. Положим, что $V$ множество обычных (чётких) высказываний (с возможными значениями *1* и *0*) с операциями $\neg$, $\&$ и $\vee$, т. е. имеем алгебру $\boldsymbol{B}=\langle V;\ \neg,\ \&,\vee\rangle$. Легко видеть, что эти алгебры изоморфны, при этом $\varnothing$ $(\varnothing\in X)$ отображается на противоречие, а $A$ на тавтологию. Этот изоморфизм можно продемонстрировать с помощью следующей таблицы.

| | $\boldsymbol{A}=\langle X;\ \bar{\ },\ \cap,\ \cup\rangle$ - алгебра подмножеств множества $A$ | $\boldsymbol{B}=\langle V;\ \neg,\ \&,\vee\rangle$ - алгебра высказываний |
|---|---|---|
| Основное множество | $X$ – множество подмножеств множества $A$ | $V$- множество высказываний |
| Выделенные элементы из основного множества | $\varnothing$ | $П$- противоречие |
| | $A$ | $Т$-тавтология |
| Операции | $\bar{\ }$ - дополнение | $\neg$- отрицание |
| | $\cap$ - пересечение | $\&$ - конъюнкция |
| | $\cup$ - объединение | $\vee$ - дизъюнкция |

При указанном изоморфизме каждый элемент или операция, записанная в некоторой строке таблицы, для одной из алгебр переходит в соответствующий элемент или операцию, записанную в той же строке для другой алгебры.

Легко показать, что существует изоморфизм между стандартной логикой Лукасевича $L_1$ (с максиминными операциями) и алгеброй нечётких подмножеств с операциями дополнения, пересечения и объединения, введёнными по (5.3), (5.4) и (5.5) соответственно. Действительно, функцию принадлежности $\mu_B(x)$, $x\in X$, с помощью которой задаётся нечёткое подмножество $B$ на универсальном множестве $X$, можно интерпретировать как функцию задающую степени истинности (истинностные значения) утверждения «$x$ является элементом подмножества $B$» в $L_1$. Обратно, истинностные значения утверждения «$x$ является $P$» в $L_1$, где $P$ нечеткий предикат (такой, как *молодой, высокий, красивый* и т. п.) можно интерпретировать как значения функции принадлежности нечёткого подмножества со свойством $P$, определённого на $X$. Изоморфизм тогда следует из того, что логические операции в $L_1$, определённые по формулам (5.2), в точности совпадают с операциями для нечётких подмножеств.

Стандартная логика Лукасевича $L_1$ является лишь одной из возможных бесконечнозначных логик. Другие бесконечнозначные логики со значениями на *[0,1]* можно строить, например, вводя иначе, чем в $L_1$ операции. Для

каждого частного случая такой бесконечнозначной логики можно ставить в соответствие изоморфную алгебру нечетких подмножеств с новыми операциями. Таким образом, исследование бесконечнозначных логик равносильно исследованию нечетких подмножеств (алгебры нечетких подмножеств) и наоборот.

Кроме того, для каждой многозначной логики можно ставить в соответствие некоторую изоморфную алгебру нечетких подмножеств с некоторыми операциями.

Рассмотренная выше нечёткая логика, т.е. множество нечётких высказываний с операциями $\neg$, $\&$ и $\vee$ является по существу некоторым расширением понятия многозначной логики. Такая нечёткая логика считается нечёткой логикой в узком смысле. В широком смысле нечёткая логика равнозначна теории нечётких множеств, см. [3] и работы, указанные в [3].

Подробнее о различных нечетких логиках можно прочитать в работах [3, 12, 15, 16, 23].

*Fuzzy logic=computing with words*
*(Нечёткая логика=вычисления посредством слов).*

\- *Л. Заде*

## § 5. Понятие о нечеткой лингвистической логике

Основоположником понятия лингвистической переменной является Л. Заде. Он же заложил основы применения этой переменной к приближенным рассуждениям. Главная цель введения лингвистической переменной и логики, основанной на этих переменных, является формализация приближенных рассуждений, используя теорию нечетких множеств.

В этой логике используются нечёткие количественные понятия (*почти все, много, мало, несколько* и т.п.), нечёткие истинностные значения (*существенно истинный, очень истинный, более или менее истинный, ложный* и т. п.), а также иные нечеткие понятия (*молодой, редкий, дорогой, красивый, почти невозможный, невероятный* и т.п.).

*Лингвистической* называется переменная, значениями которой являются слова или предложения естественного или искусственного языка. Например, *ВОЗРАСТ* - можно рассматривать как числовую переменную, а можно рассматривать как лингвистическую переменную, принимающую следующие лингвистические значения: *очень молодой, молодой, вполне молодой, не молодой, не молодой и не очень старый, старый* и т.п. При этом для каждого из перечисленных значений нужно задать характеристическую функцию, называемую смыслом этого значения.

Более точно лингвистическая переменная описывается набором:

$$(X, T(X), U, G, M)$$

в котором:

*X* - название лингвистической переменной,

*T(X)* - множество лингвистических значений переменной *X*,

*U* - универсальное множество,

*G* - синтаксические правила, порождающие названия переменной, т.е. правила определения синтаксических значений,

*M* - семантические правила, которые ставят в соответствие каждой нечеткой переменной ее смысл *M(X)*, т.е. характеристическую функцию для *X*.

Отметим, что при определении *U* и *T(X)* множество понимается в обычном классическом смысле, а не имеется в виду нечеткие множества. Причём всюду, когда имеем дело с нечеткими множествами, пишется «нечеткое множество», если слово «нечеткое» не записано, то это обозначает, что нечеткости нет. Кроме того отметим, что множество *U* может быть как конечным, так и бесконечным, а множество *T(X)* считается конечным.

Структуру лингвистической переменной можно представить в следующем виде, см. Рис. 5.5.

Трактовка истинности как лингвистической переменной приводит к нечеткой лингвистической логике, которая существенно отличается от двухзначной и многозначной логики. Эта нечеткая логика является основой того, что можно было бы назвать приближенными рассуждениями.

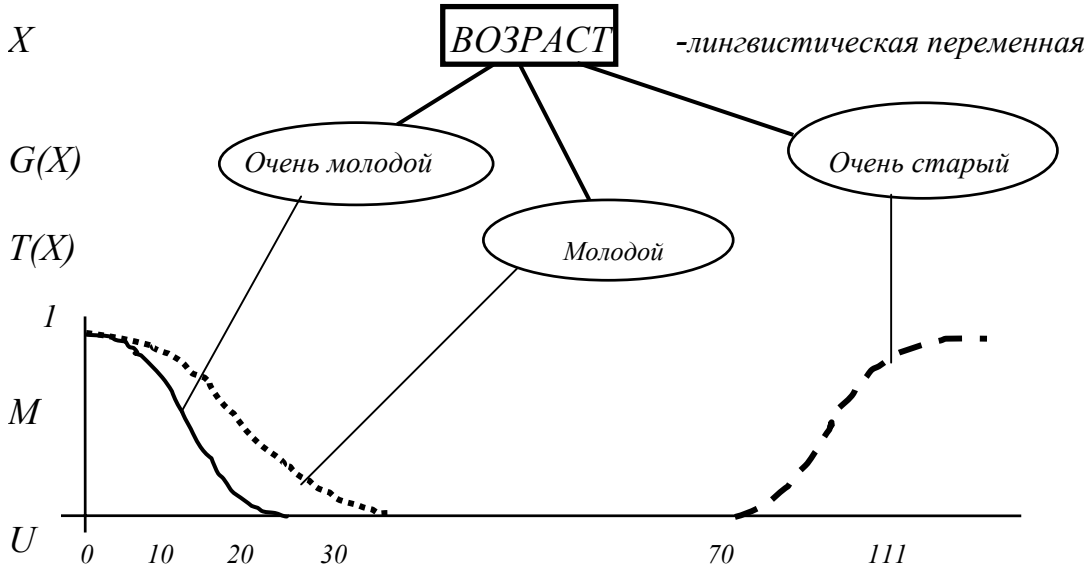



Рис. 5.5.

Лингвистическая переменная *ИСТИННОСТЬ* может принимать, например, следующие лингвистические значения:

*существенно истинный,*
*очень, очень истинный,*
*очень истинный,*
*истинный,*
*не очень истинный,*

*более или менее истинный,*

*...,*

*существенно ложный.*

Смысл каждого значения является некоторой функцией принадлежности на базовом множестве [*0,1*].

Функцию принадлежности значения *истинный*, например, можно задать как в предыдущем параграфе, либо, например, с помощью выражения:

$$\mu_{истинный}(x) = \begin{cases} 0, & при \quad 0 \le x \le a, \\ 2\left(\dfrac{x-a}{1-a}\right)^2, & при \quad a \le x \le \dfrac{a+1}{2}, \\ 1-2\left(\dfrac{x-1}{1-a}\right)^2, & при \quad \dfrac{a+1}{2} \le x < 1. \end{cases}$$

Тогда носителем значения *истинный* является отрезок [*a,1*], см. Рис 5.6. Для значения *ложный* функцию принадлежности, например, можно задать выражением: $\mu_{ложный}(x) = \mu_{истинный}(1-x)$, график см. на Рис. 5.6.

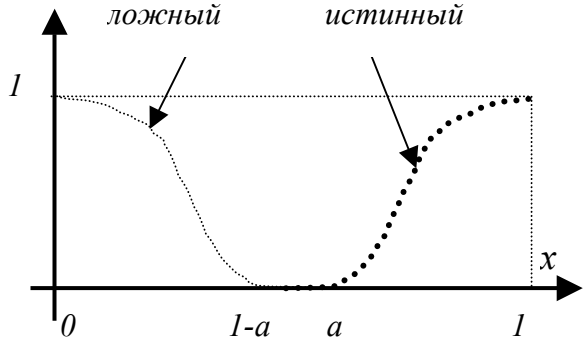



Рис. 5.6.

В некоторых случаях считают, что *U* есть конечное множество, например, *U={0; 0,1; 0,2; 0,3; ...; 0,9; 1},* которое записывают в виде: *U=0+0,1+0,2+0,3+...+0,9+1*.

При таком задании *U* функцию принадлежности значения *истинный* можно определить, например, так:

*истинный=0,5/0,7+0,7/0,8+0,9/0,9+1/1*,

где, например, пара *0,5/0,7* означает, что совместимость значения истинности *0,7* со значением *истинный* равна *0,5*.

Для исследований лингвистических переменных вводятся логические операции - связки $\neg, \&, \vee$. Ясно, что эти операции будут уже не столь тривиальны. Здесь нужно будет различать, например, соединение союзом «и» лингвистических значений (положим, истинный и не истинный) от союза «и» в высказывании «*ИСТИННЫЙ* и не *ИСТИННЫЙ*»

Построенная таким образом нечеткая логика используется в так называемых приближенных рассуждениях. Приближенные рассуждения лежат в основе способности человека понимать естественный язык, разбирать почерк, играть в шахматы, принимать решения в сложной и не полностью определенной среде. Данная логика интенсивно исследуется и находятся ее приложения, в частности, используется в экспертных системах, в системах читающих рукописный текст и т.п.

*Если исключить невозможное, то то что останется, сколь бы невероятным оно ни было, должно быть истиной (Шерлок Холмс).*
*А. К. Дойль*

## § 6. Модальные логики

Назначение различных систем модальной логики состоит в том, чтобы включить в логику так называемые модальности – прежде всего *необходимости* и *возможности*: того что «*должно быть*», и того, что «*может быть*».

Обычно говорят, что высказывание логически необходимо, если его истинность может быть установлена независимо от опыта, или чисто логическим путем. В модальной логике из необходимости высказывания вытекает его истинность, но не наоборот. Высказывание и его отрицание не могут быть вместе необходимыми.

Необходимость является, таким образом, более сильным видом истины, чем фактическая истинность. С самого зарождения логики было подмечено различие между истинными высказываниями, являющимися таковыми, так сказать в силу необходимости, и высказываниями истинными случайными, возможными.

Развитие модальной логики можно разбить на три периода:

К первому периоду относится зарождение модальной логики в античности и некоторое развитие в средневековье. Модальности были введены Аристотелем, который считал, что термин «возможность» имеет различный смысл. Аристотелем введены и исследованы модальные силлогизмы и некоторые другие аспекты модальностей. Проводя исследование аристотелевской логики, Лукасевич заключает, что в работах Аристотеля можно найти все элементы необходимые для построения полной системы модальной логики, однако Аристотель исходил из двузначной логики, в то время как модальная логика не может быть двузначной. К идее многозначной логики Аристотель подошёл вплотную, рассуждая о «будущем морском сражении». Следуя Аристотелю, Лукасевич в 1920 году построил трёхзначную логику. Тем самым выявляется идейная связь между модальными и многозначными логиками.

Второй период связан с появлением работ К. Льюиса (примерно 80 лет назад). В этот период строятся формальные системы (исчисления) модальной логики, выявляются различные черты модальных понятий. Идея Льюиса состояла в проведении различия между связками, выражающими логическую необходимость, и связками, не выражающими такого рода необходимости.

Для третьего периода, начатого работами С. Крипке (конец 1950 годов), существенно выявление внутреннего единства различных систем, казавшихся ранее никак не связанными между собой.

Приведем описание модальной системы *S1* К. И. Льюиса (согласно [13]). В языке исчисления вводятся символы:

1) *p,q,r,…*- символы для высказываний;

2) $\sim, \bullet, \diamond$ - отрицание, конъюнкция (логическое произведение), возможность;

3) *),(* - скобки.

Определение формул: 1) каждый из символов *p,q,r,…* считается формулой; 2) если *A* и *B* формулы, то следующие выражения тоже формулы

$\sim A$ – отрицание *A*;

$A \bullet B$ - конъюнкция *A* и *B*;

$\diamond A$ – возможно *A*;

$A \vee B \overset{def}{=} \sim((\sim A) \bullet (\sim B))$;

$A \supset B \overset{def}{=} \sim A \vee B$ – материальная импликация (отметим, что в материальной импликации высказывание $A \Rightarrow B$ истинно, если *A* ложно, что не всегда удобно. Почему, например, из того, что $2 \times 2 = 5$ следует, что Иванов – студент. В строгой импликации это уже устраняется);

$A \prec B \overset{def}{=} \sim\diamond(A \bullet (\sim B))$ - строгая импликация Льюиса. «$A \prec B$» читается «*A* имплицирует *B*» (здесь $A \prec B$ выражает строгую импликацию в отличие от ранее рассмотренной $A \Rightarrow B$. В строгой импликации из ложности высказывания $2 \times 2 = 5$ не следует, что Иванов – студент, ибо должно быть, что $A \prec B$ необходимо истинно);

$A \div B \overset{def}{=} (A \prec B) \bullet (B \prec A)$, $\div$ знак строгой эквивалентности;

$A \equiv B \overset{def}{=} (A \supset B) \bullet (B \supset A)$ – материальная эквивалентность.

Аксиомы:

A1) $p \bullet q \prec q \bullet p$;

A2) $p \bullet q \prec p$;

A3) $p \prec p \bullet p$;

A4) $(p \bullet q) r \prec p \bullet (q \bullet r)$;

A5) $p \prec \sim(\sim p)$;

A6) $(p \prec q) \bullet (q \prec r) \prec (p \prec r)$;

A7) $p \bullet (p \prec q) \prec q$.

Правила вывода следующие.

Р1. Правило замены строго эквивалентным.

Р2. Вместо любых переменных *p, q, r,…* можно подставить произвольную формулу.

Р3. Введение конъюнкции: из *A, B* выводится $A \wedge B$.

Р4. *Modus pones* со строгой импликацией: из *A* и $A \prec B$ выводится *B*.

Если добавить к аксиомам системы *S1* аксиому

А8) $\Diamond(p \bullet q) \prec \Diamond p$,

то получается система *S2* Льюиса, и *S2* считается системой строгой импликации. Льюисом были предложены и другие системы модальной логики, в частности, системы *S3- S5* и другие.

Отметим, что в классической логике, например, из лжи следует что угодно. Это иногда противоречит нашему содержательному, практическому пониманию логического следования. Для устранения этого парадокса материальной импликации Льюис и создал свои системы со строгой импликацией. Но появились парадоксы и для строгой импликации. Для исключения парадоксов строгой импликации Аккерман Ф. В. построил свою систему модальной логики.

Кроме систем Льюиса, Аккермана существуют системы Лукасевича и некоторые другие системы.

На втором этапе развития современной модальной логики были построены, так называемые, семантики возможных миров. Основным понятием в них является понятие возможного мира. Под возможным миром понимается мыслимое положение дел, или возможный ход развития вещей.

*Не оплакивай, смертный, вчерашних потерь,*
*Дел сегодняшних завтрашней меркой не мерь,*
*Ни былой, ни грядущей минуте не верь,*
*Верь минуте текущей – будь счастлив теперь!*
*Омар Хайям*

*Если бы все прошедшее было настоящим, а настоящее продолжало существовать наряду с будущим, кто был бы в силах разобрать: где причины и где последствия?*
*Козьма Прутков*

## § 7. Временные (темпоральные) логики

*Временные (темпоральные) логики* вводят понятия «*было*»,- «*есть*»,- «*будет*» («*раньше*», «*одновременно*», «*позже*») и считается, что истинностное значение высказывания может быть разным в различные моменты времени. Например, «самолёт летит» истинно во все те моменты времени, когда самолёт летит, и ложно во все те, когда он не летит.

Отрицание высказывания истинно во все те времена, когда само высказывание ложно, и ложно во все те времена, когда само утверждение истинно.

Конъюнкция двух высказываний истинна во всякое время, в которое истинно каждое из этих высказываний, и ложно во всякое другое время.

Остальные операции определяются через отрицание и конъюнкцию.

К высказываниям могут быть применены также, так называемые, временные операторы – $P$ и $F$. Оператор $F$ образует будущее время: $Fq$ читается «будет иметь место случай, что $q$» или «будет $q$» (слабое будущее).

Используя $P$ и $F$ определяется сильное прошлое и будущее;

$Hq \overset{def}{=} \sim P \sim q$ – всегда было $q$ (сильное прошлое);

$Gq \overset{def}{=} \sim F \sim q$ – всегда будет $q$ (сильное будущее).

Последние понятия сходны с принимаемыми в модальной логике понятиями необходимости и возможности.

Отметим, что первые системы темпоральной логики были построены А. Прайором в 1954 и предназначались для реконструкции понятий возможности и необходимости.

Первая логика времени, построенная Прайором, представляет собой пропозициональное исчисление, дополненное формой $Fq$ («будет $q$»), следующими аксиомами

*A1*) $F(p \vee q) \equiv Fp \vee Fq$;

*A2*) $FFp \supset Fp$.

и правилами вывода:

*R1*) $F \vdash GA$,

*R2*) $(A \equiv B) \vdash (FA \equiv FB)$.

Эта система предназначалась для решения вполне определённой задачи – реконструкции представлений Диодора из Меоад о возможности и необходимости. Диодор определял возможное как то, что или является, или через некоторое время будет истинным; невозможное как то, что не является и никогда не будет истинным; и необходимое как то, что и является, и всегда будет истинным.

В настоящее время построены различные темпоральные логики, оперирующие понятиями прошлое, настоящее и будущее. Кроме того выявлено, что временные логики и модальные логики взаимосвязаны.

## § 8. **Вопросы и темы для самопроверки**

1. Трёхзначная логика Лукасевича.
2. Трёхзначная логика Гейтинга.
3. Трёхзначная логика Рейхенбаха.
4. Что общего в трёхзначных логиках и двузначной логике? Какие различия между ними?
5. Многозначная ($k$-значная) логика Поста.
6. Многозначная ($k$-значная) логика Лукасевича. Бесконечнозначные логики, пример введения операций в них.

7. Понятие нечёткого множества.
8. Нечёткие высказывания и максиминные операции над ними.
9. Понятие о лингвистической нечёткой логике.
10. Модальная логика.
11. Временные (темпоральные) логики.

## § 9. Упражнения

Рассмотрим $k$-значную ($k \geq 2$) логику Поста, где, как уже указывалось, имеется множество высказываний (переменных), каждое из которых может принимать за одно из значений $0,1,2,\ldots,k-1$ и на этом множестве высказываний введены операции:

1) $\bar{x} = x+1 \pmod k$ – циклическое отрицание или отрицание Поста, здесь + - сложение по модулю $k$; выражение $\bar{x}$ будем обозначать (в данной работе) также через $\neg x$;

2) $Nx = k-1-x$ – отрицание Лукасевича;

3) $I_m(x) = \begin{cases} k-1, & \text{если } x = m, \\ 0, & \text{если } x \neq m, \end{cases}$ $m=0,1,\ldots,k-1,$

функция $I_m(x)$ называется иногда характеристической функцией и обозначается как $x^m$;

4) $x \& y = \min(x, y)$ – конъюнкция;

5) $x \vee y = \max(x, y)$ - дизъюнкция;

6) $x \times y = x \times y \pmod k$ – произведение по модулю $k$;

7) $x+y = x+y \pmod k$ – сумма по модулю $k$;

8) $x-y = \begin{cases} 0, & \text{если } 0 \leq x < y \leq k-1, \\ x-y, & \text{если } 0 \leq y \leq x \leq k-1; \end{cases}$

9) $x \Rightarrow y = \begin{cases} k-1, & \text{если } 0 \leq x < y \leq k-1, \\ (k-1)-x+y, & \text{если } 0 \leq y \leq x \leq m-1. \end{cases}$

***В упражнениях 1-10 предполагается, что задана k-значная (k≥2) логика Поста.***

1. Выяснить, выполняются ли следующие соотношения:
   а) $N(Nx)=x$; б) $\neg(\neg x)=x$; в) $(x \Rightarrow y) \Rightarrow y = x \vee y$.
2. Выяснить, выполняются ли следующие соотношения:
   а) $(x \Rightarrow y) + \neg y = x \& y$; б) $N(\neg x+y) = (Nx)+(Ny)$;
   в) $\neg(N(\neg x \times \neg y)) = (Nx) \times (\neg y)$.
3. Доказать, что каждая функция $f(x_1,\ldots,x_n)$ $k$-значной ($k \geq 2$) логики Поста представима в виде:

$$f(x_1,\ldots,x_n) = \bigvee_{(a_1,a_2,\ldots,a_n)} I_{a_1} \& \ldots \& I_{a_n} \& f(a_1,a_2,\ldots,a_n),$$

где дизъюнкция берётся по всевозможным наборам значений $(a_1,a_2,\ldots,a_n)$ переменных $(x_1,x_2,\ldots,x_n)$. Правая часть такого представление функции называется совершенной дизъюнктивной нормальной формой (с.д.н.ф.).

4. Построить с.д.н.ф. для следующих функций:

а) $x \vee y$, $k=3$; б) $\neg x \Rightarrow y$, $k=3$; в) $Nx$, $k=5$.

5. Выяснить, полна ли следующая система функций:

$\{0, 1, 2,\ldots, k-1, I_0(x), I_1(x), I_2(x),\ldots, I_{k-1}(x), x\&y, x \vee y\}$.

6. Доказать, что имеет место соотношение:

$$I_j(x)=1+\max_{\alpha \neq k-1-j}\{x+\alpha\}.$$

7. Выяснить, образуют ли полную систему следующие системы функций:

1) $\{n, \neg x, I_0(x), I_1(x), I_2(x),\ldots, I_{k-1}(x), x\&y, x \vee y\}$, здесь $n$ $(0 \le n \le k-1)$-любое целое неотрицательное число не превосходящее $k-1$, т.е. $0 \le n \le k-1$;

2) $\{1, Nx, I_0(x), I_1(x), I_2(x),\ldots, I_{k-1}(x), x\&y, x \vee y\}$;

3) $\{0, \neg x, x\&y, x \vee y\}$.

8. Построить с.д.н.ф. для следующих функций:

а) $x \Rightarrow y$, $k=3$; б) $x \vee (\neg x\&y)$, $k=3$.

9. Выяснить имеют ли место соотношения:

а) $x-y=x-x\&y$; б) $(Nx)-(Ny)=y-x$; в) $x-y=(x\&y)-y$.

10. Выяснить имеют ли место соотношения:

а) $x \times (y+z)=(x \times y)+(x \times z)$, $k=2$; б) $x \times (y+z)=(x \times y)+(x \times z)$, $k=3$.

11. Постройте таблицы истинности в логиках $L_4$ и $L_5$ Лукасевича для следующих выражений:

а) $x \vee \neg x$; б) $x\& \neg x$;

в) $x \vee (\neg x\&y)$; г) $\neg x \Rightarrow y$;

д) $x\&(\neg x \vee y)$; г) $x \vee \neg x \vee \neg\neg x$.

12. Пусть для нечётких подмножеств $A^*$, $B^*$ и $C^*$, определённых на универсальном множестве $U=[0,10]$, их функции принадлежности равны соответственно:

$$\mu_{A^*}(x)=\frac{1}{1+x}, \quad \mu_{B^*}(x)=2^{-x} \quad \text{и} \quad \mu_{C^*}(x)=\frac{1}{1+10(x-2)^2}.$$

Запишите в аналитическом виде и нарисуйте графики функций принадлежности для следующих нечетких подмножеств:

а) $\overline{A}^*$, $\overline{B}^*$, $\overline{C}^*$; б) $A^* \cup B^*$, $A^* \cup C^*$;

в) $B^* \cup C^*$, $A^* \cup B^* \cup C^*$; г) $A^* \cap B^*$, $A^* \cap C^*$;

д) $(A^* \cap B^*) \cup C^*$; е) $A^* \cap \overline{C}^*$.

13. Нечеткие подмножества $A^*$, $B^*$ и $C^*$, на универсальном множестве $U=[0;\infty)$, заданы их функциями принадлежности соответственно:

$$\mu_{A^*}(x)=\frac{1}{1+20x}, \quad \mu_{B^*}(x)=\left(\frac{1}{1+10x}\right)^{1/2} \quad \text{и} \quad \mu_{C^*}(x)=\left(\frac{1}{1+10x}\right)^2.$$

Упорядочите по включению нечёткие подмножества $A^*$, $B^*$, и $C^*$.

14. Докажите, что для нечетких подмножеств $A^*$, $B^*$ и $C^*$, с введёнными операциями дополнения, объединения и пересечения, выполняются:

а) законы дистрибутивности;

б) законы де Моргана;

в) законы поглощения.

15. Пусть нечеткие подмножества $A^*$ и $B^*$ заданы с помощью таблиц в параграфе 3. Запишите в виде (5.6) следующие нечёткие подмножества:

а) $A^* \cup B^*$; б) $A^* \cap \overline{B}^*$;

в) $A^* \cup (B^* \cap \overline{A}^*)$; г) $\overline{A}^* \cup B^*$.

*Делай, как делается*
*(правила для решения задач)*[1].

# Глава 6. ТЕОРИЯ АЛГОРИТМОВ

Теория алгоритмов является одной из ветвей (разделов) математической логики. Первоначально теория алгоритмов возникла в связи с внутренними потребностями теоретической математики. Основания математики, алгебра, геометрия и анализ остаются и сегодня одной из основных областей приложения теории алгоритмов. Другая область ее приложения возникла в 40-х годах в связи с созданием быстродействующих вычислительных машин. Наконец, теория алгоритмов оказалась тесно связанной и с рядом областей лингвистики, экономики и философии.

*Вытапливай воск, но сохраняй мед.*
*Козьма Прутков.*

## §1. Неформальное понятие алгоритма

Понятие алгоритма принадлежит к числу основных понятий математики. Под алгоритмом понимают точное предписание о выполнении в определенном порядке системы операций для решения всех задач некоторого данного типа.

Простейшими алгоритмами являются правила, по которым выполняется та или другая из четырех арифметических операций в десятичной системе счисления. Само слово алгоритм (алгорифм) происходит от имени узбекского математика Аль-Хорезми, который в IX веке систематизировал правила арифметических операций. Полное имя Аль-Хорезми следующее: Аль-Хорезми Абу Абдалла Мухаммед бен Муса аль-Маджуси.

Рассмотрим правила сложения целых чисел в десятичной системе счисления. Этот алгоритм перерабатывает выражение, записанное с использованием цифр 0,1,...,9. Результатом является выражение, записанное с помощью цифр 0,1,…,9. Таким образом, мы имеем процедуру преобразования некоторых символьных входов (цифр, означающих слагаемые) в определенные символьные выходы (цифры, означающие

[1] *рекомендация по методу (правилу, алгоритму) решения задач в древнеегипетском папирусе Ринда (2000 г. до н.э.)*

сумму). Аналогичная ситуация имеет место и для остальных арифметических операций. В общем случае понятно, что алгоритм есть преобразование каких-то входов, записанных с помощью некоторых символов, в выходы, записанные тоже с помощью символов. Грубо говоря, алгоритм - это детерминированная процедура, которую можно применять к любому элементу некоторого класса символьных входов и которая для каждого такого входа дает через конечное число действий (шагов) соответствующий символьный выход.

Существенными чертами неформального понятия алгоритма оказываются следующие:

1 - алгоритм задается как набор инструкций конечных размеров, т. е. его можно описать конечным набором слов и специальных символов;

2 - имеется вычислитель, обычно человек, который умеет обращаться с инструкциями и производить вычисления;

3 – алгоритм имеет некоторое число входных данных;

4 - имеется возможность для выделения, запоминания и повторения шагов вычисления;

5 - для каждого данного входа вычисление (преобразование входа) производится по данным инструкциям;

6 – с помощью алгоритма получается одно или несколько выходных данных.

Легко заметить аналогию с цифровыми вычислительными машинами.

Отметим, что алгоритм обладает также свойством массовости, т.е. он применяется для решения множества однотипных задач, а не одной задачи. Например, правила сложения чисел позволяют производить сложение любых действительных чисел.

Особо отметим, что задание алгоритма предполагает, что процесс применения алгоритма к входам (решение задачи с помощью алгоритма) является механическим, т.е. процедура преобразования входов не требует для своего осуществления никакой изобретательности.

Эти рассуждения выявляют некоторые свойства алгоритма, но не дают достаточно точного определения алгоритма.

*Словами диспуты ведутся.*
*Из слов системы создаются;*
*Словам должны вы доверять, ...*
*(Мефистофель)*
*И. Гете (Фауст)*

## § 2. Алфавит. Слова. Алгоритм в алфавите. Вполне эквивалентные алгоритмы

*Алфавитом* называется всякое непустое множество символов, а сами символы алфавита называются буквами.

Примером алфавита может служить конечное множество символов $\{a,+,?,v\}$ или, например, русский алфавит.

*Словом* в данном алфавите $A$ называется всякая конечная последовательность букв алфавита $A$.

Пустая последовательность букв называется *пустым словом* и обозначается через $\Lambda$.

Если $P$ обозначает слово *abb* и $Q$ обозначает слово *bb*, то пусть $PQ$ обозначает слово *abbbb*, аналогично для любых слов $P$ и $Q$ запись $PQ$ обозначает слово, полученное из слов $P$ и $Q$, если сразу за $P$ записать слово $Q$.

Ясно, что для любого слова $P$ имеем $P\Lambda=\Lambda P=P$.

Алфавит $A$ называется *расширением алфавита* $B$, если $B\subset A$. Если алфавит $A$ есть расширение алфавита $B$, то, очевидно, всякое слово в алфавите $B$ является также словом и в алфавите $A$, обратное верно не всегда.

Будем говорить, что *слово $P$ входит* в слово $Q$, если $Q=R_1PR_2$, где $R_1$ и $R_2$ любые, может быть и пустые слова. Слово $P$ может входить в слово $Q$ несколько раз, *первым вхождением* будем считать самое левое вхождение.

Под *алгоритмом в алфавите* $A$ понимается алгоритм, входами и выходами которого являются слова в алфавите $A$. Таким образом, алгоритм в алфавите $A$ представляет собой потенциально осуществимое преобразование слов в данном алфавите $A$.

Алгоритмы обозначаются заглавными полужирными буквами ($\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$,...).

Пусть $P$ слово в алфавите $A$. Говорят, что алгоритм $\boldsymbol{A}$ применим к слову $P$, если применение его к слову $P$ приводит через конечное число шагов к некоторому слову $Q$. При этом слово $Q$ обозначается через $\boldsymbol{A}(P)$. Если процесс переработки (преобразования) слова $P$ бесконечен, то считается, что алгоритм не применим к этому слову.

Два алгоритма $\boldsymbol{A}$ и $\boldsymbol{B}$ в одном и том же алфавите $C$ называются *вполне эквивалентными в алфавите $D$ ($D \subseteq C$)*, если для любого слова $P$ в алфавите $D$ либо оба алгоритма не применимы к $P$, либо применимы и их результаты совпадают: $\boldsymbol{A}(P) = \boldsymbol{B}(P)$. То, что алгоритмы $\boldsymbol{A}$ и $\boldsymbol{B}$ вполне эквивалентны в алфавите $D$, обозначается следующим образом:

$$\forall P \text{ в } D:\ \boldsymbol{A}(P) \cong \boldsymbol{B}(P).$$

*Введение подходящих абстракций – это для нашей мысли единственный способ организовать сложное и управлять им.*

*Э. Дейкстра (Дисциплина программирования)*

## § 3. Нормальный алгоритм (алгоритм А. А. Маркова)

Опыт изучения и применения математики показывает, что все известные алгоритмы можно разбить на простейшие шаги - элементарные операции. В качестве элементарной операции, на базе которой будут строиться нормальные алгоритмы, выделим подстановку одного слова вместо другого.

Пусть задан алфавит $A$, не содержащий в качестве букв символов "$\bullet$" и "$\rightarrow$", и пусть $P$ и $Q$ слова в алфавите $A$. Тогда выражения $P \rightarrow Q$, $P \rightarrow \bullet Q$ называются *формулами подстановки* в алфавите $A$.

Формула подстановки $P \rightarrow Q$ называется *простой подстановкой* и означает, что вместо $P$ нужно подставить слово $Q$ и перейти к следующей подстановке.

Формула подстановки $P \rightarrow \bullet Q$ называется *заключительной подстановкой* и означает, что вместо $P$ нужно подставить $Q$ и закончить процесс преобразования.

Пусть $P \rightarrow (\bullet) Q$ означает любую из формул подстановки $P \rightarrow Q$ или $P \rightarrow \bullet Q$.

*Нормальный алгоритм в алфавите A* считается заданным, если задана конечная таблица (схема) формул подстановок слов алфавита $A$:

$$\boldsymbol{B} = \begin{cases} P_1 \rightarrow (\bullet) Q_1 \\ P_2 \rightarrow (\bullet) Q_2 \\ \ldots\ldots\ldots\ldots\ldots \\ P_n \rightarrow (\bullet) Q_n \end{cases}$$

$n \geq 1$, $P_i$ и $Q_i$, $(1 \leq i \leq n)$ слова в $A$, причем согласно этой таблице (схеме) формул подстановок можно осуществлять преобразование слов алфавита $A$ следующим образом.

Пусть $R_o$ слово в алфавите $A$. Если ни одно из $P_1, P_2, \ldots, P_n$ не входит в $R_o$, то процесс применения $\boldsymbol{B}$ к $R_o$ заканчивается и его результатом считается слово $R_o$. Если хотя бы одно из $P_1, P_2, \ldots, P_n$ входит в $R_o$, то отыскиваем самую первую (в порядке следования в таблице) формулу подстановки, такую, что $P_k$ входит в $R_o$. Совершаем подстановку слова $Q_k$ вместо самого левого вхождения слова $P_k$ в слово $R_o$. Пусть $R_1$ - результат такой подстановки. Если использованная подстановка $P_k \rightarrow (\bullet) Q_k$ - заключительная, то работа алгоритма заканчивается и его результатом считается $R_1$. Если $P_k \rightarrow (\bullet) Q_k$ - простая формула подстановки, то применим к $R_1$ тот же поиск, который был только что применен к $R_o$, и так далее. В случае, когда через конечное число шагов процесс преобразования закончится, то полученное слово $R_i$ и является результатом. Если же процесс переработки слова $R_o$ бесконечен (никогда не заканчивается), то считаем, что алгоритм не применим к слову $R_o$.

*Нормальный алгоритм над алфавитом A* отличается от нормального алгоритма в алфавите $A$ только тем, что в словах $P_i$ и $Q_i$ $(1 \leq i \leq n)$ могут использоваться не только буквы из алфавита $A$, но и буквы, не принадлежащие алфавиту $A$.

Рассмотрим несколько примеров нормальных алгоритмов.

1. Пусть $A=\{a,b,c\}$. Таблица формул подстановок

$$\boldsymbol{B}=\begin{cases} a \to a & (1) \\ cc \to (\bullet)c & (2) \\ b \to c & (3) \end{cases}$$

задает некоторый нормальный алгоритм $\boldsymbol{B}$ в алфавите $A$. Если взять слово $R_o=bb$, то $\boldsymbol{B}$ преобразует его сначала с помощью подстановки *(3)* в слово $cb$, затем, вновь применяя подстановку *(3)* получим $cc$. Далее, по заключительной подстановке *(2)*, получим $c$. Это слово и будет $\boldsymbol{B}(R_o)$.

Если исходное слово $R_o$ будет содержать букву $a$, то $\boldsymbol{B}$ не применим к $R_o$, ибо подстановка (1) будет применяться безостановочно.

2. Пусть $A=\{a,b,c\}$. Рассмотрим таблицу формул подстановок

$$\boldsymbol{B}=\begin{cases} \beta a \to a\beta & (1) \\ \beta b \to b\beta & (2) \\ \beta c \to c\beta & (3) \\ \beta \to \bullet a & (4) \\ \Lambda \to \beta & (5) \end{cases}$$

Это таблица, очевидно, задает нормальный алгоритм над алфавитом $A$, ибо $\beta \notin A$. Если взять произвольное слово $R_o$ в $A$, то к нему сначала подстановки *(1),(2)* и *(3)* не применимы, так как слов $\beta a$, $\beta b$ и $\beta c$ в слове $R_o$ быть не может. Подставив вместо самого левого пустого слова ($\Lambda$) в $R_o$ слово $\beta$ по *(5)*, имеем $\beta R_o$. Если первой в $R_o$ стоит буква $a$, то применяем подстановку *(1)*, если же первая буква - $b$, то применяем подстановку *(2)*, а если первая $c$, то применяем *(3)* и перемещаем $\beta$ за первую букву в слове $R_o$. Повторяя этот процесс столько раз, сколько букв в слове $R_o$, получим $R_o\beta$. По подстановке *(4)* окончательно имеем $R_oa$, т.е. этот алгоритм приписывает к произвольному слову $R_o$ в алфавите $A$ справа от $R_o$ букву $a$.

В рассмотренном алгоритме $\boldsymbol{B}$ подстановки *(1),(2)* и *(3)* служат для перестановки местами $\beta$ и $a$ или $\beta$ и $b$ или $\beta$ и $c$, т.е. для перестановки $\beta$ с любой буквой алфавита $A$. Тогда можно ввести для нашего алгоритма $\boldsymbol{B}$ более краткую форму записи:

$$\boldsymbol{B'}=\begin{cases} \beta x \to x\beta \quad (x \in A) & (1^*) \\ \beta \to (\bullet)a & (2^*) \\ \Lambda \to \beta & (3^*) \end{cases}$$

Здесь запись *(1*)*: $\beta x \to x\beta$ применена для обозначения подстановок *(1)-(3)* в $\boldsymbol{B}$.

Пусть $A=\{a_1,a_2,...,a_n\}$, $P$, $Q$ и $R$ слова в алфавите $A$. Договоримся, что запись вида $PxQ \to (\bullet)R \ \ (x \in A)$ обозначает таблицу подстановок:

$$\begin{cases} Pa_1Q & \to & (\bullet)R \\ Pa_2Q & \to & (\bullet)R \\ \quad ... & & ... \\ Pa_nQ & \to & (\bullet)R \end{cases}$$

3. Пусть заданы алфавиты $A$ и $B$, буква $\alpha$ не входит в $A$ и в $B$, и пусть $a_1,a_2,...,a_k$ - фиксированные буквы из алфавита $A$, а $Q_1,Q_2,...,Q_k$ - фиксированные слова в алфавите $B$. Рассмотрим нормальный алгоритм в алфавите $A \cup B \cup \{\alpha\}$, задаваемый таблицей

$$\boldsymbol{B}=\begin{cases} \alpha a_i & \to & Q_i\alpha & (i=1,2,...k) \\ \alpha x & \to & x\alpha & x \in A \setminus \{a_1,a_2,...,a_k\} \\ \alpha & \to & \bullet\Lambda & \\ \Lambda & \to & \alpha & \end{cases}$$

Этот алгоритм в любом слове $P$ (алфавита $A$) все встречающиеся там буквы $a_1,a_2,...,a_k$ - заменяет на слова $Q_1,Q_2,...,Q_k$ соответственно. Если же в $P$ букв $a_1,a_2,...,a_k$ нет, то оставляет его без изменения.

Пусть $M=\{1,*\}$. Любое неотрицательное целое (натуральное) число можно записать (обозначить) в алфавите $M$ словом, состоящим из $n$+1 букв $1$:

число $0$ обозначим словом $\overline{0}=1$,

число $1$ обозначим словом $\overline{1}=11$,

.....

число $n$ обозначим словом $\overline{n} = \underbrace{111...1}_{n+1 \text{ единиц}}$ .

Слова $\overline{n}$ будем называть цифрами. Поставим теперь в соответствие всякому вектору $(n_1,n_2,...,n_k)$, где $n_1,n_2,...,n_k$ - натуральные числа, слово $\overline{n}_1 * \overline{n}_2 * ... * \overline{n}_k$ в алфавите $M$, которое обозначим через $\overline{(n_1,n_2,...,n_k)}$. Так, например, $\overline{(1,2,3)}$ обозначает слово $11*1111*111$.

4. Нормальный алгоритм

$$\boldsymbol{A_0}=\begin{cases} * & \to & * \\ \alpha 11 & \to & \alpha 1 \\ \alpha 1 & \to & \bullet 1 \\ \Lambda & \to & \alpha, \end{cases}$$

как легко убедиться, применим только к тем словам в алфавите $M$, которые суть цифры, и переводит любое слово $\overline{n}$ в $\overline{0}$, т.е. $\boldsymbol{A_0}(\overline{n})=\overline{0}$ для любого натурального $n$. Заметим, что алгоритм $\boldsymbol{A_0}$ не применим к пустому слову.

5. Нормальный алгоритм

$$\boldsymbol{A_1}=\begin{cases} * & \to & * \\ \alpha 1 & \to & \bullet 11 \\ \Lambda & \to & \alpha, \end{cases}$$

очевидно, тоже применим только к тем словам в алфавите $M$, которые суть цифры, и преобразует любое слово $\overline{n}$ в слово $\overline{n+1}$, т.е. $\boldsymbol{A_1}(\overline{n})=\overline{n+1}$ для любого натурального $n$. Алгоритм $\boldsymbol{A_1}$, как и $\boldsymbol{A_0}$, не применим к пустому слову.

*Когда я рассмотрел то, что нужно людям при счёте, я нашёл , что всё это есть число.*

*Аль-Хорезми*

## § 4. Функции частично вычислимые и вычислимые по Маркову

Напомним, что функция *f(x)* называется частично определенной на множестве $\boldsymbol{M}$, если значения этой функции определены не для всех $x$ из $\boldsymbol{M}$.

Функция $f$ называется арифметической, если ее значения и значения её аргументов являются целыми неотрицательными числами, т.е. область определения аргументов и область значений функции есть множество натуральных чисел.

Положим, как и раньше, алфавит $M=\{1,*\}$.

Пусть $\varphi$ частично определенная арифметическая функция от $n$ аргументов. Положим, что существует некоторый алгоритм (не обязательно нормальный) $\boldsymbol{A_\varphi}$ в алфавите $M$, позволяющий вычислять значения этой функции всякий раз, когда значение функции существует, т.е. $\boldsymbol{A_\varphi}(\overline{k_1,k_2,...,k_n})=\overline{\varphi(k_1,k_2,...,k_n)}$ тогда и только тогда, когда хотя бы одна из частей этого равенства определена. При этом считаем, что алгоритм $\boldsymbol{A_\varphi}$ не применим к словам, отличным от слов вида $(\overline{k_1,k_2,...,k_n})$. Назовем функцию *$\varphi$ частично вычислимой по Маркову функцией*, если существует нормальный алгоритм $\boldsymbol{B}$ над $M$, вполне эквивалентный $\boldsymbol{A_\varphi}$ относительно алфавита $M$.

Иными словами, $n$-аргументная функция $\varphi$ частично вычислима по Маркову тогда и только тогда, когда существует нормальный алгоритм, позволяющий вычислить значение $\varphi(k_1,k_2,...,k_n)$ для любых совокупностей значений $x_1=k_1$, $x_2=k_2$,..., $x_n=k_n$, при которых $\varphi(k_1,k_2,...,k_n)$ существует.

Если функция определена всюду, т.е. определена для любой совокупности значений своих аргументов, и является частично вычислимой по Маркову, то назовем ее *вычислимой по Маркову*. Таким образом, $n$-аргументная функция $\varphi$ вычислима по Маркову тогда и только тогда, когда существует нормальный алгоритм, позволяющий вычислить значение $\varphi(x_1,x_2,...,x_n)$ для любых совокупностей значений $x_1,x_2,...,x_n$.

В предыдущем параграфе показано, что для всюду определенных арифметических функций

$f(x)=0, \quad \forall x(x\geq 0),$

$f(x)=x+1, \quad \forall x(x\geq 0),$

существуют нормальные алгоритмы, вычисляющие их значения (примеры 4, 5). Следовательно, эти функции являются вычислимыми по Маркову.

*Где начало того конца, которым*
*оканчивается начало.*
*Козьма Прутков*

## § 5. Замыкание, распространение нормального алгоритма

Пусть $\boldsymbol{A}$ - произвольный нормальный алгоритм в алфавите $A$:

$$\boldsymbol{A}=\begin{cases} P_1 \to (\bullet) Q_1 \\ P_2 \to (\bullet) Q_2 \\ \ldots\ldots\ldots\ldots \\ P_n \to (\bullet) Q_n \end{cases}$$

*Замыканием алгоритма* $\boldsymbol{A}$ называется алгоритм $\boldsymbol{A}^{\bullet}$, полученный из $\boldsymbol{A}$ добавлением формулы подстановки $\Lambda\to\bullet\Lambda$ в качестве последней подстановки, т.е.

$$\boldsymbol{A}^{\bullet}=\begin{cases} P_1 \to (\bullet) Q_1 \\ P_2 \to (\bullet) Q_2 \\ \ldots\ldots\ldots\ldots \\ P_n \to (\bullet) Q_n \\ \Lambda \to \bullet\Lambda \end{cases}$$

Нам известно, что любой нормальный алгоритм заканчивает процесс переработки слова либо после применения заключительной подстановки, либо если все слова $P_1,P_2,P_3,\ldots, P_n$ не содержатся в слове, полученном при предыдущем шаге. Подстановка $\Lambda\to\bullet\Lambda$ применима к любому слову. Следовательно, алгоритм $\boldsymbol{A}^{\bullet}$ заканчивает переработку слов всегда по заключительной подстановке, которая есть либо некоторая из подстановок $P_i\to(\bullet)Q_i$ $(1\le i\le n)$ либо $\Lambda\to\bullet\Lambda$.

Отметим, что подстановка $\Lambda\to\bullet\Lambda$, добавленная к $\boldsymbol{A}$ для получения $\boldsymbol{A}^{\bullet}$, стоит последней. Поэтому эта подстановка будет применяться только тогда, когда не применимы все подстановки алгоритма $\boldsymbol{A}$, причем применение $\Lambda\to\bullet\Lambda$ к любому слову не изменяет этого слова. Следовательно, результаты применения алгоритмов $\boldsymbol{A}$ и $\boldsymbol{A}^{\bullet}$ к любому слову в $A$ будут совпадать, т.е. алгоритмы $\boldsymbol{A}$ и $\boldsymbol{A}^{\bullet}$ вполне эквивалентны.

Пусть $\boldsymbol{A}$ - нормальный алгоритм в алфавите $A_1$, а алфавит $A_2$ является расширением $A_1$, т.е. $A_1\subset A_2$. Тогда можно рассмотреть нормальный алгоритм $\boldsymbol{A}^{\#}$ в алфавите $A_2$ с той же самой схемой, что и $\boldsymbol{A}$ . Очевидно,

$$\forall P \text{ в алфавите } A_1\colon \boldsymbol{A}(P) \cong \boldsymbol{A}^{\#}(P), \qquad (1)$$

т.е. $\boldsymbol{A}$ и $\boldsymbol{A}^{\#}$ вполне эквиваленты относительно $A_1$. Нормальный алгоритм $\boldsymbol{A}^{\#}$ будем называть *естественным распространением* $\boldsymbol{A}$ на алфавит $A_2$.

В некоторых случаях удобнее, чтобы алгоритм $\boldsymbol{A}^{\#}$, удовлетворяющий (1), был не применим к тем словам в $A_2$, которые не являются словами в $A_1$. Этого

легко достигнуть, приписав к схеме $\boldsymbol{A}$ сверху формулу вида $x \to x$, где $x$ - любая буква из $A_2 \setminus A_1$. Получившийся нормальный алгоритм называют *формальным распространением* $\boldsymbol{A}$ на алфавит $A_2$. Очевидно, что формальное распространение алгоритма $A$ вполне эквивалентно алгоритму $\boldsymbol{A}$ относительно $A_1$ и не применимо к тем словам в $A_2$, которые не являются словами в $A_1$.

Использование возможности распространения нормального алгоритма на более широкий алфавит позволяет во многих случаях опускать без особого ущерба точности упоминание об алфавитах, в которых строятся конкретные нормальные алгоритмы.

## § 6. Операции над нормальными алгоритмами

*Композиция алгоритмов*. Пусть $\boldsymbol{A}$ и $\boldsymbol{B}$ два алгоритма в алфавите $A$. Композицией алгоритмов $\boldsymbol{A}$ и $\boldsymbol{B}$ в алфавите $A$ называют алгоритм $\boldsymbol{C}$ такой, что $\forall P$ в $A$ : $\boldsymbol{C}(P) \cong \boldsymbol{B}(\boldsymbol{A}(P))$.

Таким образом, композиция алгоритмов $\boldsymbol{A}$ и $\boldsymbol{B}$ представляет собой алгоритм, получающийся в результате последовательного применения алгоритмов к заданному слову $P$, что можно продемонстрировать следующей блок-схемой (рис.6.1)

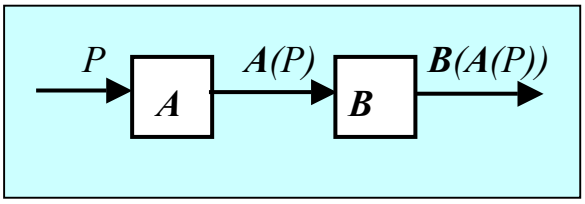



Рис. 6.1.

Композиция алгоритмов $\boldsymbol{A}$ и $\boldsymbol{B}$ обозначается как: $\boldsymbol{C}=\boldsymbol{B} \circ \boldsymbol{A}$

Пусть $\boldsymbol{A}_1,\boldsymbol{A}_2,\ldots,\boldsymbol{A}_n$ - алгоритмы в алфавите $A$, тогда под $\boldsymbol{A}_n \circ \boldsymbol{A}_{n-1} \circ \ldots \circ \boldsymbol{A}_1$ будем понимать следующее:

$\boldsymbol{A}_n \circ (\boldsymbol{A}_{n-1} \circ (\ldots \circ (\boldsymbol{A}_3 \circ (\boldsymbol{A}_2 \circ \boldsymbol{A}_1))\ldots))$.

**Теорема 6.1**. Композиция нормальных алгоритмов $\boldsymbol{A}_1,\boldsymbol{A}_2,\ldots,\boldsymbol{A}_n$ в алфавите $A$ есть снова нормальный алгоритм (над алфавитом $A$).

Ясно, что достаточно провести доказательство для композиции двух алгоритмов.

**Доказательство**. Пусть $\boldsymbol{A}$ и $\boldsymbol{B}$ - нормальные алгоритмы в алфавите $A$. Сопоставим каждой букве $a$ из $A$ новую букву $\overline{a}$, которую назовем *двойником буквы* $a$. При этом считаем, что для любой буквы $a$ из $A$ ее двойник не принадлежит $A$. Пусть $\overline{A}$ - алфавит, состоящий из всех двойников букв алфавита $A$. Выберем какие ни будь две буквы, например, $\alpha$ и $\beta$ , не принадлежащие $A \cup \overline{A}$. Обозначим через $\boldsymbol{A}^{\alpha}$ схему, полученную из схемы нормального алгоритма $\boldsymbol{A}^{\bullet}$ заменой в ней всюду "•" на "$\alpha$".

Обозначим через $\overline{B}^{\beta}$ схему, полученную из $\boldsymbol{B}^{\bullet}$ заменой "•" на "$\beta$", далее всех букв из $A$ - их двойниками и затем заменой всех формул подстановок вида $\Lambda\to Q$ формулами подстановок $\alpha\to\alpha Q$.

Рассмотрим схему:

$$\boldsymbol{C}=\begin{cases} a\alpha\to\alpha a & (a\in A) & (1)\\ \alpha a\to\alpha\,\overline{a} & (a\in A) & (2)\\ \overline{a}b\to\overline{a}\,\overline{b} & (a,b\in A) & (3)\\ \overline{a}\beta\to\beta\,\overline{a} & (a\in A) & (4)\\ \beta\overline{a}\to\beta a & (a\in A) & (5)\\ a\overline{b}\to ab & (a,b\in A) & (6)\\ \alpha\beta\to\bullet\Lambda & & (7)\\ \overline{\boldsymbol{B}}^{\beta} & & (8)\\ \boldsymbol{A}^{\alpha} & & (9) \end{cases}$$

Эта схема представляет собой сокращенную запись некоторого нормального алгоритма $\boldsymbol{C}$ (в алфавите $A\cup\overline{A}\cup\{\alpha,\beta\}$), причем в (1)-(6) буквы $a$ и $b$ означают произвольные буквы из $A$, а строки (8) и (9) означают, что нужно записать все подстановки сначала алгоритма $\overline{B}^{\beta}$, затем алгоритма $\boldsymbol{A}^{\alpha}$.

Покажем, что нормальный алгоритм $\boldsymbol{C}$ таков, что

$\forall P$ в $A$: $\boldsymbol{C}(P)\cong\boldsymbol{B}(\boldsymbol{A}(P))$,

т.е. $\boldsymbol{C}$ есть композиция $\boldsymbol{A}$ и $\boldsymbol{B}$. Пусть задано произвольное слово $P$, например, $P=a_{к1}a_{к2}\ldots a_{кт}$ $(a_{кi}\in A)$. Сначала подстановки (1)-(8) не применимы к $P$, ибо буквы $\alpha$, $\beta$ и буквы-двойники (алгоритм $\overline{B}^{\beta}$ записан в двойниках) не содержатся в слове $P$, а также в (8) нет подстановок вида $\Lambda\to Q$, ибо они заменены на подстановки $\alpha\to\alpha$Q. Если алгоритм $\boldsymbol{A}$ не применим к слову $P$ (перерабатывает его бесконечно), то по (9) и $\boldsymbol{C}$ будет не применимым к $P$. Если $\boldsymbol{A}$ применим к $P$, то в результате мы получили бы некоторое слово $R=\boldsymbol{A}(P)$, пусть, например, $R=b_{n1}b_{n2}\ldots b_{nk}$ $(b_{ni}\in A)$. Нам известно, что алгоритмы $\boldsymbol{A}$ и $\boldsymbol{A}^{\bullet}$ вполне эквивалентны, т.е. результаты их применения к любому слову из $A$ совпадают. В предыдущем параграфе мы отметили, что алгоритм $\boldsymbol{A}^{\bullet}$ всегда заканчивает переработку слова заключительной подстановкой $P_i\to\bullet Q_i$ либо $\Lambda\to\bullet\Lambda$. Так как в $\boldsymbol{A}^{\alpha}$ все "•" заменены на "$\alpha$", то в результате применения (9) получим, что в слове $R$ где-то вклинится буква $\alpha$, т.е. имеем:

$b_{n1}b_{n2}\ldots b_{nq}\alpha b_{n(q+1)}\ldots bn_k$ $(q\le k,\ b_{ni}\in A,\ 1\le i\le k)$.

Применяя $q$ раз подстановку (1) к последнему слову, получим

$\alpha\, b_{n1}b_{n2}\ldots b_{nk}$ $(\alpha\boldsymbol{A}(P))$.

После этого, применяя подстановку (2), и $k$-$1$ раз подстановку (3), придем_к слову

$\alpha\overline{b}_{n1}\,\overline{b}_{n2}\ldots\overline{b}_{nk}$.

Если алгоритм ***B*** не применим к слову $R=\boldsymbol{A}(P)$, то и алгоритм ***C*** будет не применим к $P$ вследствие того, что (8) будет бесконечно перерабатывать слово $\bar{b}_{n1}\,\bar{b}_{n2}\ldots\bar{b}_{nk}$

Если алгоритм ***B*** применим к слову $R=\boldsymbol{A}(P)$, то в результате его применения мы получили бы слово $\boldsymbol{B}(\boldsymbol{A}(P))$, пусть, например,

$$\boldsymbol{B}(\boldsymbol{A}(P)) = c_{m1}\,c_{m2}\ldots c_{ml}\ (c_{mi}\in A,\ 1\le i\le l),$$

тогда, как легко видеть, что в результате применения (8) получим

$$\alpha\,\bar{c}_{m1}\bar{c}_{m2}\ldots\bar{c}_{ms}\,\beta\,\bar{c}_{ms+1}\ldots\bar{c}_{ml}\quad (s\le l;\ c_{mi}\in A,\ 1\le i\le l)$$

Теперь $s$-кратное применение (4) приводит к слову

$$\alpha\,\beta\,\bar{c}_{m1}\bar{c}_{m2}\ldots\bar{c}_{ml}.$$

Далее применяя (5), а затем $(l\text{-}1)$ раз (6), имеем

$$\alpha\,\beta\,c_{m1}\,c_{m2}\ldots c_{m} = \alpha\,\beta\,\boldsymbol{B}(\boldsymbol{A}(P)).$$

Применяя заключительную подстановку (7), получим результат $\boldsymbol{B}(\boldsymbol{A}(P))$.

Таким образом, $\boldsymbol{C}(P)\cong\boldsymbol{B}(\boldsymbol{A}(P))$, и так как слово $P$ было произвольным, то теорема доказана.

***Соединение алгоритмов***.

**Теорема 6.2**. Пусть $\boldsymbol{A}_1,\boldsymbol{A}_2,\ldots,\boldsymbol{A}_n$-нормальные алгоритмы в алфавитах $A_1,A_2,\ldots,A_n$ соответственно и пусть $A$ - объединение этих алфавитов. Тогда существует нормальный алгоритм ***B*** над $A$, называемый соединением алгоритмов $\boldsymbol{A}_1,\boldsymbol{A}_2,\ldots,\boldsymbol{A}_n$, такой, что

$$\forall P \text{ в } A:\ \boldsymbol{B}(P)\cong\boldsymbol{A}_1^{\#}(P)\boldsymbol{A}_2^{\#}(P)\ldots\boldsymbol{A}_n^{\#}(P),$$

где $\boldsymbol{A}_i^{\#}$ есть естественное распространение $\boldsymbol{A}_i$ на $A$.

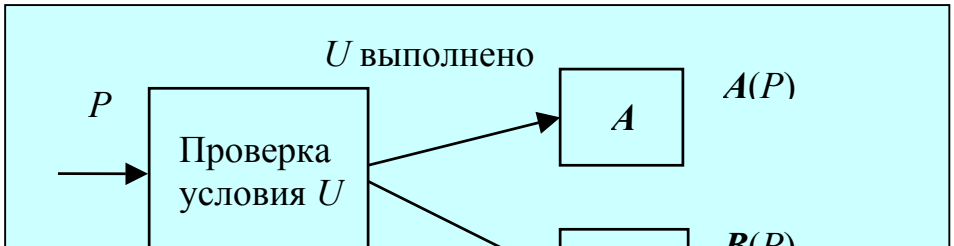
Теорему примем без доказательства. Доказательство см., например, в работе [21].

***Разветвление алгоритмов***. Пусть заданы алгоритмы ***A*** и ***B*** в алфавите $A$ и задано некоторое условие $U$. Тогда можно задать предписание следующего типа: для данного слова $P$ в алфавите $A$ проверить, удовлетворяет ли оно условию $U$, если удовлетворяет, то к $P$ применить алгоритм ***A*** , в противном случае применить к $P$ алгоритм ***B,*** см. Рис 6.2.

Ясно, что условие $U$ можно задавать самым различным образом, например, можно задать так: условие $U$ выполнимо для заданного слова $P$, если $P\neq NP$ (см. главу о сложности вычислений). Решить, выполнимо это условие $U$ или нет, в настоящее время нельзя, ибо указанная проблема еще не решена, несмотря на настойчивые попытки. Поэтому такого рода (типа) условия не будем рассматривать. Будем считать, что условие $U$ всегда задано с помощью какого-то алгоритма ***C*** в алфавите $A$ в следующем виде:

условие $U$ для слова $P$ выполнено, если $\boldsymbol{C}(P)=\Lambda$,

условие $U$ для слова $P$ не выполнено, если $\boldsymbol{C}(P)\neq\Lambda$.



*U* не выполнено

При таком задании условия *U* описанное выше предписание будем считать разветвлением алгоритмов в алфавите *A*. Точнее, пусть ***A, B*** и ***C***-алгоритмы в алфавите *A*. *Разветвлением алгоритмов* ***A*** и ***B*** называется алгоритм ***V*** в *A*, не применимый к *P*, если не применим к *P* алгоритм ***C*** и такой, что`

$$\forall P \text{ в } A : V(P) \cong \begin{cases} \boldsymbol{A}(P), \text{ если } \boldsymbol{C}(P) = \Lambda \\ \boldsymbol{B}(P), \text{ если } \boldsymbol{C}(P) \neq \Lambda \end{cases}.$$

Будем говорить, что это разветвление алгоритмов *управляется алгоритмом* ***C*** .

**Теорема 6.3.** Разветвление нормальных алгоритмов, управляемое нормальным алгоритмом, является нормальным алгоритмом.

Примем без доказательства. Доказательство см., например, в работе [21].

***Повторение алгоритмов.*** Во многих случаях требуется повторить одну и ту же процедуру многократно, каждый раз применяя ее к результату, полученному на предыдущем шаге. Процедуру повторяем до выполнения некоторого условия *U*. Будем считать, что условие *U* задается с помощью алгоритма, как и выше. Такая процедура будет задавать повторение алгоритма. Более точно: пусть ***A*** и ***C*** - алгоритмы в алфавите *A* и пусть $P_o$ - произвольное слово в *A*. Применим к $P_o$ алгоритм ***A***. Получим некоторое слово $P_1=\boldsymbol{A}(P_o)$, если $\boldsymbol{C}(P_1)=\Lambda$, то процесс заканчивается. Если $\boldsymbol{C}(P_1)\neq\Lambda$, то к $P_1$ применяем ***A***, получаем $P_2=\boldsymbol{A}(P_1)=\boldsymbol{A}(\boldsymbol{A}(P_o))$. Если $\boldsymbol{C}(P_2)=\Lambda$, то процесс заканчиваем. Если $\boldsymbol{C}(P_2)\neq\Lambda$, то к $P_2$ применяем ***A***, и т.д. Определенный таким образом алгоритм называется *повторением алгоритма* ***A***, управляемым алгоритмом ***C***.

Повторение алгоритма можно представить следующей блок схемой, см. Рис. 6.3.

Рис. 6.3. Повторение алгоритмов

Также без доказательства примем следующую теорему.

**Теорема 6.4.** Повторение нормального алгоритма, управляемое нормальным алгоритмом, есть нормальный алгоритм.

Основным и важнейшим результатом этого параграфа является то, что различные операции (комбинации) над нормальными алгоритмами снова приводят к нормальному алгоритму.

## § 7. Машина Тьюринга

Стремясь найти точное определение понятия алгоритма, Тьюринг выделил некоторый класс абстрактных машин, о которых высказал предположение, что они пригодны для осуществления любой "механической" вычислительной процедуры. Эти машины называются теперь в честь их изобретателя машинами Тьюринга.

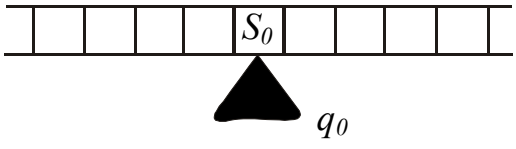



Рис. 6.4.

Пусть имеется лента, потенциально бесконечная в обе стороны* и разделенная на ячейки (квадраты), см. Рис. 6.4. Потенциальная бесконечность ленты понимается в том смысле, что в каждый данный момент времени она имеет конечную длину, и вместе с тем к ней всегда как слева, так и справа могут быть добавлены новые квадраты.

Имеется некоторое конечное множество символов $S_0,S_1,..,S_n$, которое называется *алфавитом машины*. В каждой ячейке может быть записан только один из символов - букв алфавита машины.

Машина обладает некоторым конечным множеством *внутренних состояний* $\{q_0,q_1,...,q_m\}$. В каждый данный момент времени машина находится только в одном из этих состояний. Считаем, что внутренним

*) *Тьюринг определил машину с лентой потенциально бесконечной вправо и ограниченной слева. Потенциальная бесконечность ленты в обе стороны упрощает дальнейшее описание работы машины. Можно показать, что вводимая машина эквивалентна машине, определенной Тьюрингом.*

состоянием $q_0$ обладает каждая машина и $q_0$ называется *начальным состоянием.*

Машина имеет *читающую головку*, которая в каждый данный момент времени находится на одном из квадратов ленты и воспринимает символ, записанный на этом квадрате. Будем предполагать, что среди символов $S_0,S_1,..,S_n$ имеется символ, означающий пустой квадрат, например, $S_0$. Положим, что символ $S_0$ принадлежит алфавиту каждой машины Тьюринга и $S_0$ не может быть записан ни в каком квадрате ленты. Когда мы пишем, что читающая головка обозревает квадрат с символом $S_0$ или записывает в квадрат символ $S_0$, то имеем в виду, что читающая головка соответственно обозревает пустой квадрат (воспринимая его как $S_0$) или оставляет квадрат без символов (воспринимая его как $S_0$). Машина действует не непрерывно, а лишь в дискретные моменты времени.

Если в какой-то момент времени $t$ читающая головка воспринимает квадрат (т.е. стоит на квадрате), содержащий символ $S_i$, и машина находится во внутреннем состоянии $q_j$, то действие машины определено, и она совершает один из следующих четырех действий:

1) головка стирает символ $S_i$ и записывает на том же квадрате символ $S_k$;

2) головка перемещается в соседний слева квадрат;

3) головка перемещается в соседний справа квадрат;

4) машина останавливается.

В случаях 1)-3) машина переходит во внутреннее состояние $q_r$ и готова к действию в следующий момент времени $t$+1.

Первые три из возможных актов действия машины могут быть описаны соответственно следующими упорядоченными четверками символов, которые в дальнейшем будем называть командами:

1) $q_j\ S_i\ S_k\ q_r$;

2) $q_j\ S_i\ L\ q_r$;

3) $q_j\ S_i\ R\ q_r$.

Любая машина имеет конечное (непустое) множество команд.

Машина останавливается, если она находится в некотором внутреннем состоянии $q_j$, читающая головка обозревает какой-то символ $S_k$, а среди команд машины нет команды, начинающейся с $q_jS_k$.

Если на ленте имеется какое-нибудь слово $P$ (в частности, пустое слово), читающая головка помещена на квадрат с первой левой буквой слова $P$ и машина приведена во внутреннее состояние $q_0$, то машина начинает оперировать на ленте: ее головка стирает и пишет символы и перемещается из одного квадрата в другой (соседний). Если при этом машина когда-нибудь останавливается, то находящееся в момент остановки слово на ленте считается результатом применения машины $T$ к данному слову $P$ и обозначается через $T(P)$. Если процесс переработки машиной $T$ слова $P$ бесконечен, то говорят, что машина $T$ не применима к слову $P$.

## § 8. Задание машины Тьюринга

Машина Тьюринга $T$ считается заданной, если задано непустое конечное множество упорядоченных четверок символов (команд), удовлетворяющих условиям:

а) каждая четверка символов принадлежит к одному из трех типов команд, приведенных выше (в § 7),

б) никакие две четверки одной машины не имеют совпадающие первые два символа,

в) среди команд любой машины всегда есть хотя бы одна команда, начинающаяся с $q_0$.

Множество всех символов типа $S_m$, входящих в команды машины, называется алфавитом заданной машины, а входящие в эти команды символы $q_i$ называются внутренними состояниями заданной машины $T$.

Считаем, что в исходном (начальном) состоянии машина обладает внутренним состоянием $q_0$.

Для преобразования слова $P$ машиной $T$ обязательно указывается положение слова на ленте относительно читающей головки. Если это не сделано, то предполагается, что читающая головка находится на первой (самой левой) букве слова $P$.

Рассмотрим несколько примеров.

1. Пусть задана машина $T$ командами:

$q_0 1 R q_1$

$q_1 S_0 1 q_0$,

а на ленте записано слово $P=1$, см. Рис.6.5. Легко убедиться, что машина $T$, начав работу с первой буквы слова $P$, приписывает к нему слева по одной букве $1$ на каждом шаге, никогда при этом не останавливаясь. Следовательно, эта машина неприменима к слову $P$.

2. Машина, заданная командами:

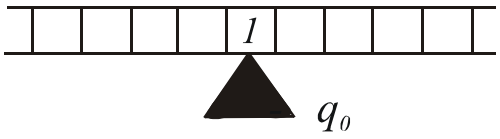



Рис. 6.5

$q_0 S_0 R q_0$

$q_0 S_1 R q_0$

$q_0 S_2 R q_0$

$q_0 S_k R q_0$

……

$q_0 1 1 q_1$,

где ни одно из $S_i$ $(1 \le i \le k)$ не совпадает с символом $1$, двигается по ленте вправо, пока не встретит вхождение (если такое вообще имеется) символа $1$, после чего останавливается.

3. Машина $T$, заданная командами

$q_0 a R q_0$

$q_0 b R q_0$

$q_0 S_0 a q_1$,

как легко убедиться, приписывает к любому слову $P$ алфавита $\{a,b\}$ справа букву $a$ и останавливается.

## § 9. Алгоритм Тьюринга. Вычислимость по Тьюрингу

С каждой машиной Тьюринга можно связать некоторый алгоритм $\boldsymbol{A}_{T,A}$ в алфавите $A$ машины $T$. Возьмем произвольное слово $P$ алфавита $A$ и запишем его слева направо в квадратах чистой ленты, причем так, чтобы первая (самая левая) буква $P$ находилась под читающей головкой. Приведем машину $T$ во внутреннее состояние $q_0$. Машина начинает работать. Если она когда-нибудь остановится, то появившееся в результате на ленте слово алфавита $A$ является значением алгоритма $\boldsymbol{A}_{T,A}$. Такой алгоритм $\boldsymbol{A}_{T,A}$ называется *алгоритмом Тьюринга*.

Будем говорить, что машина Тьюринга $T$ с алфавитом $A$, включающим $1$ и *, *частично вычисляет* частичную арифметическую функцию $f(x_1,x_2,...,x_n)$, если для любых натуральных $k_1,k_2,...,k_n$ и некоторых $r$ и $m$ имеем:

$$A_{T,A}(\overline{k_1,k_2,...,k_n}) = S_0^r \overline{f(k_1,k_2,...,k_n)} S_0^m, \quad (S_0^i = \underbrace{S_0 S_0 ... S_0}_{i \text{ раз}}),$$

тогда и только тогда, когда определена хотя бы одна из частей этого равенства. Это означает, что применение алгоритма Тьюринга $\boldsymbol{A}_{T,A}$ к слову $(\overline{k_1,k_2,...,k_n})$ даст слово, означающее с точностью до слов $S_0^r$ и $S_0^m$ значение функции $f(k_1,k_2,...,k_n)$ ($S_0$ - интерпретируется как изображение пустого квадрата ленты).

Арифметическая функция $f(x_1,x_2,...,x_n)$ называется *вычислимой по Тьюрингу*, если для любых натуральных $k_1,k_2,...,k_n$ и некоторых $r$ и $m$ имеем:

$$A_{T,A}(\overline{k_1,k_2,...,k_n}) = S_0^r \overline{f(k_1,k_2,...,k_n)} S_0^m, \quad (S_0^i = \underbrace{S_0 S_0 ... S_0}_{i \text{ раз}}).$$

Это означает, что применение алгоритма Тьюринга $\boldsymbol{A}_{T,A}$ к слову $(\overline{k_1,k_2,...,k_n})$ даст слово, означающее с точностью до слов $S_0^r$ и $S_0^m$ значение функции $f(k_1,k_2,...,k_n)$, т.е. существует машина Тьюринга, вычисляющая эту функцию для любых значений её аргументов.

Пример. Рассмотрим всюду определенную функцию сложения $f(x,y)=x+y$. Покажем, что эта функция вычислима по Тьюрингу. Для этого построим машину Тьюринга:

$q_0\ 1\ S_0\ q_0$
$q_0\ S_0\ R\ q_1$
$q_1\ 1\ R\ q_1$
$q_1\ *\ 1\ q_2$
$q_2\ 1\ R\ q_2$
$q_2\ S_0\ L\ q_3$
$q_3\ 1\ S_0\ q_3$.

Нетрудно убедиться, что эта машина переводит слово
$\overline{(m,n)} = \underbrace{11...1}_{m+1} * \underbrace{11...1}_{n+1}$ в слово $\underbrace{11...1}_{m+n+1}$, а последнее слово означает

число $m+n$, следовательно, эта машина Тьюринга вычисляет функцию $x+y$. Итак, функция $x+y$ вычислима по Тьюрингу.

*Из разнообразия возникает совершенная гармония.*

*Гераклит*

## § 10. Связь между машинами Тьюринга и нормальными алгоритмами

**Теорема 6.5.** Пусть $T$ - машина Тьюринга с алфавитом $A$. Тогда существует нормальный алгоритм $\boldsymbol{B}$ над $A$, вполне эквивалентный относительно $A$ алгоритму Тьюринга $\boldsymbol{A}_{T,A}$.

**Доказательство**. Каждая конкретная машина Тьюринга содержит конечное число команд вида 1)-3) (§ 7). Выпишем сначала для всех команд вида $q_jS_iS_kq_r$ (если они есть) машины $T$ формулы подстановки $q_jS_i \to q_rS_k$. При этом порядок этих формул подстановок друг относительно друга не существенен, ибо никакие две команды машины $T$ не имели совпадающими первые два символа. Далее для каждой команды вида $q_jS_iLq_r$ (если она есть) выпишем всевозможные формулы подстановки вида $S_lq_jS_i \to q_rS_lS_i$, где $S_l \in A$, и формулу подстановки $q_jS_i \to q_rS_0S_i$. Затем для каждой команды вида $q_jS_iRq_r$ (если она есть) выпишем всевозможные формулы подстановки $q_jS_iS_l \to S_iq_rS_l$, где $S_l \in A$, и формулу подстановки $q_jS_i \to S_iq_rS_0$. Наконец, выпишем всевозможные формулы подстановок $q_i \to \bullet\Lambda$, где $q_i$ - внутреннее состояние заданной машины $T$, и формулу подстановки $\Lambda \to q_0$. Полученная таким образом таблица (схема) формул подстановок определяет некоторый нормальный алгоритм $\boldsymbol{B}$ над $A$.

Покажем, что полученный нормальный алгоритм $\boldsymbol{B}$ вполне эквивалентен относительно алфавита $A$ алгоритму Тьюринга $\boldsymbol{A}_{T,A}$ , т.е. оба алгоритма одинаковым образом преобразуют любое слово $P$ алфавита $A$. Для этого возьмем произвольное слово $P$ в алфавите $A$, пусть, например, $P=S_{k1},S_{k2},...S_{km}$, где $S_{ki} \in A$. Машина Тьюринга находится сначала во внутреннем состоянии $q_O$ и обозревает символ $S_{k1}$. Дальнейшие ее действия определены ее командами. Из подстановок алгоритма $\boldsymbol{B}$ к слову $P$ будет сначала применима только последняя подстановка, в результате которой получим слово

$$q_0\, S_{k1},S_{k2},...S_{km} \qquad (1)$$

Если среди команд машины $T$ имеется команда, по которой стирается буква $S_{k1}$ и заменяется, например, буквой $S_j (S_j \in A)$, т.е. имеется команда $q_0 S_{k1} S_j q_r$, то по подстановке $q_0 S_{k1} \to q_r S_j$ алгоритм преобразует слово (1) в слово

$q_r\, S_j\, S_{k2} \ldots S_{km}$.

Если же машина $T$ не изменяет буквы $S_{k1}$, а читающая головка движется, например, вправо по команде $q_0 S_{k1} R q_r$, то по подстановке $q_0 S_{k1} S_l \to S_{k1} q_r S_l$ $(S_l \in A)$ алгоритма $\boldsymbol{B}$ из (1) получим

$S_{k1} q_r S_{k2} \ldots S_{km}$.

Ясно, что и любому другому такту машины $T$ будет соответствовать такое же преобразование слова с помощью алгоритма $\boldsymbol{B}$, за исключением того, что при преобразовании с помощью $\boldsymbol{B}$ в слове у нас всегда имеется некоторое $q_i$, которого нет на ленте машины. Однако в конце преобразования алгоритм $\boldsymbol{B}$ своей подстановкой $q_i \to \bullet \Lambda$ уберет эту лишнюю букву. Итак,

$\forall P$ в $A$ : $\boldsymbol{A}_{T,A}(P) \cong \boldsymbol{B}(P)$,

что и требовалось доказать.

**Следствие 6.1**. Всякая частично вычислимая (вычислимая) по Тьюрингу функция является частично вычислимой (вычислимой) по Маркову функцией.

**Доказательство**. Пусть функция $f(x_1,x_2,\ldots,x_n)$ вычислима по Тьюрингу и ее вычисляет машина Тьюринга $T$ с алфавитом $A$, содержащим 1 и *. Это означает, что для любых натуральных чисел $k_1,k_2,\ldots,k_n$ найдутся такие слова $R_1$ и $R_2$ (возможно, пустые) в алфавите $\{S_0\}$, что $\boldsymbol{A}_{T,A}(\overline{k_1,k_2,\ldots,k_n}) = R_1 \overline{f(k_1,k_2,\ldots,k_n)} R_2$. В силу доказанной теоремы 6.5 существует нормальный алгоритм $\boldsymbol{B}$ над $A$, вполне эквивалентный относительно $A$ алгоритму $\boldsymbol{A}_{T,A}$, т.е. для любых натуральных чисел $k_1,k_2,\ldots,k_n$ имеем:

$$\boldsymbol{A}_{T,A}(\overline{(k_1,k_2,\ldots,k_n)}) \cong \boldsymbol{B}(\overline{(k_1,k_2,\ldots,k_n)}) \cong R_1 \overline{f(k_1,k_2,\ldots,k_n)} R_2. \qquad (2)$$

Для того, чтобы функция была частично вычислимой по Маркову, нужно, чтобы существовал нормальный алгоритм, который преобразует $\overline{(k_1,k_2,\ldots,k_n)}$ в $\overline{f(k_1,k_2,\ldots,k_n)}$. Наш же нормальный алгоритм $\boldsymbol{B}$ преобразует $\overline{(k_1,k_2,\ldots,k_n)}$ в $R_1 \overline{f(k_1,k_2,\ldots,k_n)} R_2$. Надо как-то изменить алгоритм, чтобы он убирал слова $R_1$ и $R_2$. Пусть $\boldsymbol{B_1}$ - нормальный алгоритм над $\{1,*,S_0\}$, стирающий все вхождения $S_0$ перед первым вхождением $1$ или * во всяком слове в алфавите $\{1,*,S_0\}$. Такой алгоритм можно задать схемой:

$$\boldsymbol{B_1} = \begin{cases} \alpha S_0 \to \alpha \\ \alpha 1 \to \bullet 1 \\ \alpha * \to \bullet * \\ \alpha \to \bullet \Lambda \\ \Lambda \to \alpha \end{cases}$$

Пусть $\boldsymbol{B_2}$ - нормальный алгоритм над $\{1,*,S_0\}$, который стирает все вхождения $S_0$ после последнего вхождения $1$ или * во всяком слове в алфавите $\{1,*,S_0\}$. Например, $\boldsymbol{B_2}$ можно задать схемой:

$$\boldsymbol{B_2} = \begin{cases} \alpha * \to *\alpha \\ \alpha 1 \to 1\alpha \\ \alpha S_0 \to \alpha \\ \alpha \to \bullet \Lambda \\ \Lambda \to \alpha \end{cases}$$

Построим композицию алгоритмов $\boldsymbol{B}$, $\boldsymbol{B_1}$ и $\boldsymbol{B_2}$: $\boldsymbol{C}=\boldsymbol{B_2}\bullet\boldsymbol{B_1}\bullet\boldsymbol{B}$. По доказанной теореме 6.1, алгоритм $\boldsymbol{C}$ является нормальным алгоритмом, как композиция нормальных алгоритмов. Для любых натуральных чисел $k_1,k_2,...,k_n$ имеем

$$\boldsymbol{B}\left(\overline{(k_1,k_2,...,k_n)}\right) \cong \boldsymbol{A_{T,A}}\left(\overline{(k_1,k_2,...,k_n)}\right) \cong R_1\,\overline{f(k_1,k_2,...,k_n)}R_2 .$$

где $R_1$ и $R_2$ - некоторые слова в $\{S_0\}$. Далее

$$\boldsymbol{B_1}\left(R_1\,\overline{f(k_1,k_2,...,k_n)}R_2\right) = \overline{f(k_1,k_2,...,k_n)}R_2;$$

$$\boldsymbol{B_2}\left(\overline{f(k_1,k_2,...,k_n)}R_2\right) = \overline{f(k_1,k_2,...,k_n)}.$$

Отсюда видно, что $f$ есть частично вычислимая по Маркову функция, ее вычисляет нормальный алгоритм $\boldsymbol{C}$ .

**Теорема 6.6.** (обратная теореме 6.5).Пусть $\boldsymbol{B}$- нормальный алгоритм в алфавите $A$, не содержащем $S_0$ и $\delta$. Тогда существует такая машина Тьюринга $T$, что алгоритм Тьюринга $\boldsymbol{A}_{T,A\cup\{So,\delta\}}$ в алфавите $A\cup\{S_0,\delta\}$ обладает следующим свойством: для всякого слова $P$ в $A$ алгоритм $\boldsymbol{A}_{T,A\cup\{So,\delta\}}$ применим к $P$ тогда и только тогда, когда к $P$ применим алгоритм $\boldsymbol{B}$ и при этом $\boldsymbol{A}_{T,A\cup\{So,\delta\}}(P)$ имеет вид $S_0^r\boldsymbol{B}(P)S_0^m$, где $r$ и $m$ целые неотрицательные числа, а $S_0^k = \underbrace{S_0S_0...S_0}_{k\ \textbf{раз}}$.

Согласно сформулированной теореме значения алгоритмов $\boldsymbol{B}$ и $\boldsymbol{A}_{T,A\cup\{So,\delta\}}$ формально различны, так как для машины Тьюринга $S_0$ есть символ пустого квадрата, а в нормальном алгоритме $\boldsymbol{B}$ $S_0$-буква, равноправная с любой другой буквой. Но с точностью до символов $S_0$, которые могут стоять справа и слева от результирующего слова, алгоритмы $\boldsymbol{B}$ и $\boldsymbol{A}_{T,A\cup\{So,\delta\}}$ вполне эквиваленты. Теорему принимаем без доказательства.

**Следствие 6.2.** Всякая частично вычислимая (вычислимая) по Маркову функция частично вычислима (вычислима) по Тьюрингу.

Доказательство следствия сразу получается из теоремы 6.6 и определения вычислимой по Тьюрингу функции.

Из теоремы 6.5 и 6.6 видим, что различные подходы к понятию алгоритмов Тьюринга и Маркова (нормальные алгоритмы) по существу эквивалентны, т.е. то, что можно осуществить с помощью нормального алгоритма, можно осуществить с помощью машины Тьюринга, и наоборот.

Есть еще многоленточные машины Тьюринга и другие модификации (варианты) подхода к понятию алгоритма, такие как машины Поста, машины Минского и др. Однако детальный анализ показывает, что все эти понятия равносильны в том смысле, что то, что можно осуществить (вычислить) с помощью одной из этих машин, можно сделать (вычислить) с помощью машины Тьюринга, а следовательно, и с помощью нормального алгоритма, и наоборот.

*«Знаете что, скрипка?*
*Мы ужасно похожи:*
*Я вот тоже*
*ору -*
*а доказать ничего не умею!»*
*В. Маяковский*

## § 11. Основная гипотеза теории алгоритмов (принцип нормализации или тезис Черча)

С целью дать строгое (с некоторых позиций) определение алгоритма были введены понятия нормального алгоритма и алгоритма (машин) Тьюринга. Было выяснено, что оба эти подхода приводят к равносильным понятиям алгоритма, т.е. то, что можно осуществить с помощью одного из этих алгоритмов, можно осуществить с помощью другого, и наоборот. Естественно задаться вопросом: насколько общими являются эти схемы вычислений с помощью нормального алгоритма или алгоритма Тьюринга и насколько эти понятия близки к интуитивному понятию любого алгоритма? На эти вопросы современная теория алгоритмов предлагает ответ в виде следующей гипотезы:

Для всякого алгоритма $\boldsymbol{B}$ в алфавите $A$ существует вполне эквивалентный ему нормальный алгоритм $\boldsymbol{C}$ над $A$, т.е.

$\forall P$ в $A$: $\boldsymbol{B}(P) \cong \boldsymbol{C}(P)$.

Иначе эта гипотеза формулируется так.

Всякий алгоритм может быть задан посредством некоторой машины Тьюринга и реализован в этой машине.

Эту гипотезу называют *основной гипотезой*, или основным тезисом теории алгоритмов, или *принципом нормализации*, или *тезисом Черча*.

Ясно, что эта гипотеза не носит характера теоремы и не может быть, следовательно, доказана, ибо в нее входит нестрогое (неопределенное) понятие алгоритма. Уверенность в истинности гипотезы основана, главным образом, на опыте. Известные алгоритмы, которые были придуманы в течение многих тысячелетий, могут быть заданы посредством нормального алгоритма (машины Тьюринга). Имеются и другие соображения, подтверждающие правильность основной гипотезы. В § 6 были рассмотрены различные операции над нормальными алгоритмами (аналогичные операции можно рассмотреть и для алгоритмов (машин) Тьюринга), причем оказывалось, что каждый раз результирующий алгоритм снова является нормальным алгоритмом.

Заметим, что внутри самой теории алгоритмов основная гипотеза не применяется. В теории алгоритмов исследуется нормальный алгоритм, машина Тьюринга, машины Поста и т.п., устанавливается связь между ними и т.д. Основная гипотеза только утверждает (постулирует) универсальность понятия нормального алгоритма. Иначе можно сказать, что основной тезис применяется в теории алгоритмов только в рекламных целях: "Любой алгоритм можно представить как нормальный алгоритм".

Основную гипотезу можно рассматривать как уточнение понятия любого алгоритма через более специальное, но строгое понятие нормального алгоритма (машины Тьюринга).

*Никогда не бывает больших дел без больших трудностей.*

*Ф. Вольтер*

## § 12. Проблема алгоритмической неразрешимости

Под *массовой проблемой* будем понимать бесконечное множество однотипных задач. Массовую проблему можно обозначать, например, через $\{a_j\}$, где $a_j$ - некоторая единичная задача.

Переход от расплывчатого понятия алгоритма к строгому определению специальных алгоритмов (нормальный алгоритм, или алгоритм Тьюринга) позволяет уточнить вопрос об алгоритмической разрешимости того или иного круга задач.

Массовая проблема $\{a_j\}$ называется *алгоритмически разрешимой*, если существует алгоритм (нормальный алгоритм или алгоритм Тьюринга) позволяющий решить каждую задачу этой массовой проблемы и *алгоритмически неразрешимой*, если такого алгоритма не существует.

*Проблема алгоритмической (не)разрешимости* формулируется следующим образом: «каждая ли массовая проблема является алгоритмически разрешимой?»

Эта проблема имеет свою историю. Еще великий математик и философ Лейбниц (1646-1716) мечтал о создании всеобщего метода, позволяющего решать любую задачу. Хотя всеобщего алгоритма ему не удалось найти, Лейбниц считал, что наступит время, когда такой алгоритм будет найден. Несмотря на долгие и упорные усилия многих крупных специалистов, трудности остались непреодолимыми. Более того, были обнаружены большие трудности даже при попытке создания алгоритмов для некоторых массовых проблем частного вида (не говоря уже о любой задаче). В результате многочисленных, но безрезультатных попыток создания таких алгоритмов стало ясно, что налицо имеются трудности принципиального характера, и возникло сомнение, для каждого ли класса задач возможно построение решающего алгоритма. Оказалось, что для целого ряда массовых проблем невозможно построить разрешающего алгоритма, т.е. алгоритма, который позволил бы решить все задачи данной массовой проблемы. Первые результаты такого рода были обнаружены в математической логике в работах Геделя, Черча, Тьюринга. Ими были указаны (найдены) примеры алгоритмически неразрешимых массовых проблем.

Таким образом, существуют как алгоритмически разрешимые проблемы, так и алгоритмически неразрешимые массовые проблемы.

К алгоритмически разрешимым массовым проблемам относятся, например, следующие:

-сложение двух и более заданных чисел;

-решение в радикалах уравнений от одной переменной не выше четвертой степени;

-решение систем линейных уравнений с $n$ неизвестными;

-и т.д.

Обнаружение алгоритмически неразрешимых проблем создано в науке такую ситуацию, когда математик, стремящийся к построению желаемого алгоритма, должен считаться с тем, что такого алгоритма может и не существовать. Поэтому параллельно с поиском желаемого алгоритма приходится прилагать усилия для доказательства существования такого алгоритма.

Рассмотрим, например, полиномы, зависящие от произвольного числа переменных $x_1,x_2,...,x_n$ с целыми коэффициентами. Такие полиномы называют диофантовыми в память греческого математика Диофанта, рассмотревшего некоторые из таких полиномов. Будем интересоваться, есть ли у такого полинома целочисленные корни (диофантовы корни). Целочисленными решениями полиномов интересовались еще античные математики, например, в связи с теоремой Пифагора они рассматривали уравнение $x^2+y^2=z^2$. Евклид приводит формулы, позволяющие найти все целочисленные решения этого уравнения. Сам Диофант (Ш в. н.э.) среди многих других уравнений, рассмотрел уравнение $ax^2+bx+c=y^2$ и решил его для некоторых частных случаев.

В эпоху развития анализа диофантовы уравнения привлекали внимание выдающихся ученых, таких как Ферма, Эйлер, Лагранж, Гаусс. В частности, Ферма выдвинул знаменитую гипотезу о том, что уравнение $x^n+y^n=z^n$ при $n>2$ не имеет целочисленных решений (большая теорема Ферма).

На рубеже X1X-XX вв. Гильберт включил проблему диофантовых уравнений в число наиболее важных проблем, которые X1X век оставил XX. Эта проблема была сформулирована им так [27]: "Пусть задано произвольное диофантовое уравнение с произвольными неизвестными и целыми рациональными числовыми коэффициентами. Указать способ, при помощи которого возможно после конечного числа операций установить, разрешимо ли это уравнение в целых рациональных числах". Эта проблема называется *10-й проблемой Гильберта*.

В 1970 году советским математиком Ю.В. Матиясевичем, а несколько позже Г.В.Чудновским было доказано, что эта проблема алгоритмически неразрешима, т.е. алгоритм, который искали, не существует.

Заметим, что не существует алгоритма, позволяющего выяснить наличие целых (диофантовых) корней для произвольного полинома от произвольного числа переменных. Для полиномов частного вида разрешающий алгоритм может существовать.

## § 13. Примеры алгоритмически неразрешимых массовых проблем

В предыдущем параграфе уже рассмотрена одна из алгоритмически неразрешимых массовых проблем - 10-я проблема Гильберта. Рассмотрим еще несколько таких примеров.

1. *Проблема распознавания применимости*. Пусть задан нормальный алгоритм $\boldsymbol{A}$ и слово $P$. Возможны два случая:

1) Алгоритм применим $\boldsymbol{A}$ к слову $P$, т.е. процесс переработки слова $P$ конечен,

2) алгоритм $\boldsymbol{A}$ бесконечно перерабатывает слово $P$, т.е. не применим к этому слову.

Следовательно, возникает задача $a_i$: применим ли $\boldsymbol{A}$ к $P$ или нет. Учитывая, что нормальных алгоритмов $\boldsymbol{A}$ и слов $P$ может быть бесчисленное множество, получаем массовую проблему $\{a_i\}$ - проблему распознавания применимости произвольного нормального алгоритма к произвольному слову.

Далее ставим вопрос: существует ли общий метод для решения всех задач этой проблемы, т.е. есть ли общий метод, который позволил бы выяснить применимость произвольного нормального алгоритма $\boldsymbol{A}$ к произвольному слову $P$. Под общим методом будем понимать либо нормальный алгоритм, либо машину (алгоритм) Тьюринга.

**Теорема 6.7.** Не существует нормального алгоритма ***B***, позволяющего решить все задачи массовой проблемы $\{a_i\}$. Иначе: не существует нормального алгоритма ***B***, который позволил бы выяснить, применим или нет произвольный нормальный алгоритм ***A*** к произвольному слову $P$.

Теорему примем без доказательства.

2. *Проблема эквивалентности слов*. Пусть $A$ - некоторый алфавит, $P$ и $Q$ слова в этом алфавите, $P \to Q$ - ориентированная подстановка, т.е. когда вместо слова $P$ подставляется слово $Q$; $P - Q$ - неориентированная подстановка, т.е. можно подставить либо вместо $P$ слово $Q$, либо наоборот, вместо $Q$ слово $P$. Из бесчисленного множества возможных подстановок в алфавите $A$ задают некоторое конечное множество подстановок указанных видов и называют их допустимыми подстановками.

Два слова $R$ и $S$ в алфавите $A$ называются эквивалентными, если существует конечная последовательность слов $R_1,R_2,...,R_n$ $(R_1=R,\ R_n=S)$ такая, что $R_{i+1}$ получается из $R_i$ в результате одной допустимой подстановки.

Возникает следующая массовая проблема: для любых двух слов в данном алфавите требуется указать, эквивалентны они или нет - проблема эквивалентности слов. Марков и Пост доказали, что данная проблема тоже алгоритмически неразрешима.

3. *Проблема представимости матриц*. Пусть $U_1,U_2,...,U_q$ - матрицы порядка $n \times n$. Будем говорить о матрице $U$ того же порядка, что она представима через $U_1,U_2,...,U_q$, если для некоторого целого положительного $t$ и целых чисел $r_1,r_2,...r_t$ из ряда 1,2,...,$q$ имеет место равенство

$U=U_{r1}*U_{r2}*...*U_{rt}$.

*Общая проблема представимости матриц*. Дано целое положительное число $n$, требуется выяснить, существует ли алгоритм, посредством которого можно было бы узнавать для любой системы матриц $U,U_1,U_2,...,U_q$ порядка $n \times n$, представима ли матрица $U$ через матрицы $U_1,U_2,...,U_q$.

*Частная проблема представимости матриц*. Даны матрицы $U_1,U_2,...,U_q$ порядка $n \times n$, Требуется выяснить, существует ли алгоритм, посредством которого можно было бы узнавать для любой матрицы $U$ того же порядка, представима ли она через $U_1,U_2,...,U_q$.

А. А. Марков построил систему из 102 матриц 6-го порядка, для которой доказал алгоритмическую неразрешимость частной проблемы представимости, откуда сразу следует алгоритмическая неразрешимость и общей проблемы представимости матриц для $n \geq 6$ (позже было доказано для $n \geq 4$).

4. *Проблема неразрешимости логики предикатов*. Черчем доказано, что не существует алгоритма, который для любой формулы логики предикатов устанавливает, логически общезначима она или нет.

5. *Проблема остановки*. Тьюрингом доказано, что не существует алгоритма, позволяющего выяснить остановиться или нет произвольная

программа для произвольного заданного входа. Смысл этого утверждения для теоретического программирования состоит в следующем: не существует общего метода проверки программ на наличие в них бесконечных циклов.

6. Не существует алгоритма, позволяющего установить, вычисляет ли некоторая конкретная программа (на любом языке программирования) постоянную нулевую функцию $Z(x)=0$. Как следствие, можно утверждать, что проблема о том вычисляют ли две произвольные программы одну и ту же одноаргументную функцию, тоже алгоритмически неразрешима. Тем самым получаем, что в области тестирования компьютерных программ имеются принципиальные ограничения.

Сделаем несколько замечаний.

1. Теоремы об алгоритмической неразрешимости утверждают лишь то, что класс рассматриваемых задач достаточно обширен и для них нет одного разрешающего алгоритма, но не утверждается вообще неразрешимость этих задач. Для каждой конкретной задачи, например, для каждого данного слова $P$ может быть и можно выяснить, применим к нему данный нормальный алгоритм или нет, но нет алгоритма, позволяющего выяснить применимость любого нормального алгоритма к любому слову. Это означает, что рассматриваемый класс задач достаточно обширен и надо как-то разумно делить их на подклассы и для них искать разрешающие алгоритмы.

2. Теоремы об алгоритмической неразрешимости показывают, что математика не сводится к построению алгоритмов.

3. Область применения алгоритмов весьма широка и к ней относятся не только вычислительные процессы. Более того, для многих проблем, считающихся трудными, теоретически можно построить алгоритм, но его реализация сопряжена с очень долгим счетом и необходимостью запоминать большое количество промежуточных результатов. Создание быстродействующих вычислительных машин значительно расширило круг решаемых задач.

## § 14. Сведение любого преобразования слов в алфавите к вычислению значений целочисленных функций

Пусть $A$ - алфавит, содержащий $n$ букв. Поставим в соответствие каждой букве $a$ $(a \in A)$ число (геделев номер) $g(a)$ из чисел $1,2,...,n$, причем различным буквам поставим в соответствие различные числа (геделевы номера). Очевидно, что по каждому $g(a)$ всегда можно восстановить букву. Если задано некоторое слово $P$, например, $P=a_{k1}a_{k2}...a_{km}$ $(a_{ki} \in A)$, то сопоставим ему геделев номер:

$$g(P)=2^{b1}\,3^{b2}\,5^{b3}...q_m^{\,bm},$$

где $bi=g(a_{ki})$ и $q_m$ - $m$–е простое число. Легко видеть, что таким образом можно определить геделев номер каждому слову $P$ и по каждому геделеву номеру $g(P)$ легко восстановить слово, которому соответствует этот номер.

Пример. Пусть $A=\{a,b,c\}$. Тогда можно положить $g(a)=1$, $g(b)=2$, $g(c)=3$. Если $P=aaba$, то $g(P)=2^1*3^1*5^2*7^1$. Если же задано число $N=18$, то, представив $18$ в виде произведения степеней простых чисел $18=2*3^2$, получаем, что $18$ является геделевым номером слова $Q=ab$.

Пусть задана последовательность слов в алфавите $A$ (которую можно, например, назвать предложением в алфавите $A$). Тогда этой последовательности слов можно сопоставить последовательность геделевых номеров каждого слова в том же порядке, что и слова, либо, считая, что пробел между словами имеет геделев номер $(n+1)$, сопоставить всей последовательности один геделев номер. Например, Пусть $A=\{a,b,c\}$ и задана последовательность слов $P=abb\ cabb\ aa$, тогда либо

$g(\mathbf{P})=g(abb),\ g(cabb),\ g(aa) = 2^1*3^2*5^2,\ 2^3*3^1*5^2*7^2,\ 2^1*3^1$

либо

$g(\mathbf{P})=2^1*3^2*5^2*7^4*11^3*13^1*17^2*19^2*23^4*29^1*31^1$.

Ясно, что в данных случаях по данному номеру или последовательности номеров можно единственным образом восстановить исходную последовательность слов (предложение).

Как только проведена нумерация, становится понятным, что любое преобразование слов или предложений в алфавите $A$ в слова или предложения $A$ можно свести к вычислению значений функции

$m=\varphi(n)$ или $m=\varphi(n_1,n_2,...n_k)$,

где $n,n_1,n_2,...,n_k$ - геделевы номера преобразуемых слов или предложений, а $m$ - геделев номер результата. Это следует из того, что после введения нумерации можно иметь дело уже только с соответствующими номерами слов или предложений, а не с самими словами или предложениями.

Очевидно, что если у нас есть метод преобразования слов (предложений) алфавита $A$, то есть и метод вычисления значений соответствующей функции. Действительно, чтобы найти значение $\varphi(n)$ при $n=\alpha$, можно по $\alpha$ восстановить слово (предложение), затем с помощью имеющегося метода преобразовать его в слово являющееся результатом и по результирующему слову (предложению) найти геделев номер $\varphi(\alpha)$. Следовательно, $\varphi(\alpha)=\beta$.

Наоборот, если есть метод вычисления функции $\varphi(n)$, то, стало быть, имеется и метод преобразования исходного слова (предложения). Действительно, по записи слова (предложения) можно найти соответствующий ему номер $\alpha$, затем вычислить $\beta=\varphi(\alpha)$ и по $\beta$ определить результирующее слово (предложение).

Таким образом, всякое преобразование слов или предложений алфавита $A$ в слова или предложения того же алфавита можно свести к вычислению значения некоторой функции, и наоборот.

Ранее среди всевозможных преобразований слов алфавита нас интересовали преобразования, которые используют механическую процедуру, т.е. преобразования с помощью алгоритмов. В следующем параграфе займемся изучением функций, значения которых вычисляются с

помощью некоторых "механических" процедур, и установим связь с вычислимостью с помощью нормальных алгоритмов.

## § 15. Примитивно рекурсивные и общерекурсивные функции

Рекурсивное определение функции - это, грубо говоря, определение, в котором значения функции для данных аргументов непосредственно определяются значениями этой же функции для "более простых" аргументов или значениями "более простых" функций. (Понятие "более простой" следует уточнить: например, простейшей функцией можно считать функцию-константу). Такой подход к рассмотрению функций удобен тем, что рекурсивные определения можно рассматривать как алгоритмы.

Здесь, как и при определении вычислимости по Маркову или Тьюрингу, будем рассматривать только арифметические функции.

Теперь приступим к строгому определению примитивно рекурсивных и общерекурсивных функций.

1. Следующие функции называются *исходными (простейшими) функциями:*

1) *нуль функция*: $Z(x)=0$ при $\forall x\ (x\geq 0)$,

2) *функция прибавления единицы*: $N(x)=x+1$ при $\forall x(x\geq 0)$, ясно, что, используя функции $Z$ и $N$, можно получить любую функцию-константу, например:

$1=N(Z(x))$,

$2=N(N(Z(x)))$,

$3=N(N(N(Z(x))))$,

. . . . . .

3) *проектирующие функции* $J_i^n\ (x_1,x_2,...,x_n)=x_i$ при всех $x_1,x_2,...,x_n\geq 0$ $(i=1,2,...,n;\ n=1,2,...)$

2. Следующие правила служат для получения новых функций, исходя из уже имеющихся функций.

*Подстановка*

$f(x_1,x_2,...,x_n)=g(h_1(x_1,x_2,...,x_n),...,h_m(x_1,x_2,...,x_n))$, тогда говорят, что функция $f$ получена с помощью подстановки из функций $g$, $h_1$, $h_2$,..., $h_m$.

*Рекурсия*:

а) $\begin{cases} f(x_1,x_2,\ldots,x_n,0)=g(x_1,x_2,\ldots,x_n), \\ f(x_1,x_2,\ldots,x_n,y+1)=h(x_1,x_2,\ldots,x_n,y,f(x_1,x_2,\ldots,x_n,y)), \end{cases}$

при этом исключается случай $n=0$, для которого:

б) $\begin{cases} f(0)=k, \text{ где } k \text{ - фиксированное целое неотрицательное число}, \\ f(y+1)=h(y,f(y)). \end{cases}$

В случае а) будем говорить, что функция $f$ получена из $g$ и $h$ с помощью рекурсии, а $x_1,x_2,\ldots,x_n$ - параметры рекурсии.

В случае б) говорят, что функция $f$ получена из одной функции $h$ с помощью рекурсии.

Заметим, что функция $f$ всегда определена: в случае а) значение $f(x_1,x_2,\ldots,x_n,0)$ определяется из 1-го равенства, далее, если мы знаем $f(x_1,x_2,\ldots,x_n,y)$, то из 2-го равенства определяем $f(x_1,x_2,\ldots,x_n,y+1)$, аналогично и для случая б).

$\mu$ - *оператор*:

пусть функция $g(x_1,x_2,\ldots,x_n,y)$ такова, что для любых $x_1,x_2,\ldots,x_n$ существует по крайней мере одно значение $y$, при котором $g(x_1,x_2,\ldots,x_n,y)=0$.

Обозначим через $\mu y(g(x_1,x_2,\ldots,x_n,y)=0)$ наименьшее значение $y$ при котором $g(x_1,x_2,\ldots,x_n,y)=0$.

Пусть $f(x_1,x_2,\ldots,x_n)=\mu y(g(x_1,x_2,\ldots,x_n,y)=0)$. Будем тогда говорить, что функция $f$ получена из функции $g$ с помощью $\mu$-оператора, если для любых $x_1,x_2,\ldots,x_n$ существует по крайней мере одно значение $y$, для которого

$g(x_1,x_2,\ldots,x_n,y)=0$.

Функция называется примитивно рекурсивной, если она может быть получена из исходных функций 1), 2) и 3) с помощью конечного числа подстановок и рекурсий.

Функция $f$ называется общерекурсивной, если она может быть получена из исходных функций 1), 2) и 3) с помощью конечного числа подстановок, рекурсий и $\mu$-оператора. Общерекурсивные функции иногда называют рекурсивными функциями.

Из определений очевидно, что каждая примитивно рекурсивная функция является общерекурсивной, но не наоборот.

Отметим, что примитивно и общерекурсивные функции являются всюду определенными функциями, так как исходные функции всюду определены, а подстановки, рекурсии и $\mu$-оператор не меняют всюду определенности.

## § 16. Примитивно рекурсивность некоторых функций. Частично - рекурсивные функции

**Теорема 6.8.** Следующие функции являются примитивно рекурсивными:

1) $x+y$;

2) $x \bullet y$;

3) $x^y$;

4) $\delta(x) = \begin{cases} x-1, & \text{если } x>0, \\ 0, & \text{если } x=0; \end{cases}$

5) $x-y = \begin{cases} x-y, & x \geq y, \\ 0, & x<y; \end{cases}$

6) $|x-y| = \begin{cases} x-y, & x \geq y \\ y-x, & x<y; \end{cases}$

7) $sg(x) = \begin{cases} 0, & x=0, \\ 1, & x \neq 0; \end{cases}$

8) $sg^*(x) = \begin{cases} 1, & x=0, \\ 0, & x \neq 0; \end{cases}$

9) $rm(x,y)$ = остатку от деления $y$ на $x$;

10) $qt(x,y)$ = частному от деления $y$ на $x$;

11) $x!$;

12) $min\ (x,y)$;

13) $min\ (x_1,x_2,\ldots,x_n)$;

14) $max\ (x,y)$;

15) $max\ (x_1,x_2,\ldots,x_n)$.

**Доказательство.**

1) Обозначим первую функцию через $f$. Имеем

$f(x,0)=x+0=x=J_1^1(x)$,

$f(x,y+1)=x+(y+1)=(x+y)+1=f(x,y)+1=N(f(x,y))=h(x,y,f(x,y))$, где $h(x,y,z) = N(z)$.

В результате получили, что $f(x,y)=x+y$ получается рекурсией из примитивно рекурсивных функций $g(x)=J_1^1(x)$ и $h(x,y,z)=N(z)$, следовательно, $f$ - примитивно рекурсивна.

2) Обозначим функцию умножения $x$ на $y$ через $\psi$. Тогда

$\psi(x,0)=x \bullet 0 = 0 = Z(x)$;

$\psi(x,y+1)=x \bullet (y+1)=x \bullet y+x=\psi(x,y)+x=f(\psi(x,y),x)=h^*(x,y,\psi(x,y))$,

где $h^*(x,y,z) = f(x,z)$. Итак, $\psi(x,y)$ получается рекурсией из примитивно рекурсивных функций $g^*(x)=Z(x)$ и $h^*(x,y,z)=f(x,z)$, следовательно, функция $\psi$ - примитивно рекурсивна.

3) Обозначим функцию возведения в степень через $\varphi$. Для арифметических функций полагаем, что $x^0=1$ для любых целых $x$, в том числе и для $x=0$, т. е. $0^0=1$ ($\varphi(0,0)=1$). Получим

$\varphi(x,0)=x^o=1=N(Z(x));$

$\varphi(x,y+1)=x^{y+1}=x^y \bullet x=\varphi(x,y) \bullet x=\psi(\varphi(x,y),x)=h^{**}(x,y,\varphi(x,y)),$

где $h^{**}(x,y,z) = \psi(x,z),$ т.е. $\varphi$ получается рекурсией из примитивно рекурсивных функций $g^{**}(x)=N(Z(x))$ и $h^{**}(x,y,z)=\psi(x,z)$, следовательно $\varphi$ - примитивно рекурсивна.

Аналогичным образом доказывается и для функций 4)-15).

Из теоремы следует, что примитивно рекурсивные функции образуют достаточно широкий класс всюду определенных функций.

Рассмотрим, как можно вычислять значения примитивно рекурсивных функций. Каждая примитивно рекурсивная функция получается из исходных функций 1), 2) и 3) с помощью конечного числа подстановок и рекурсий. Представление функции с помощью подстановки и рекурсии, примененных к исходным функциям, можно рассматривать как набор инструкций для "механического" вычисления значения функций. Следовательно, доказательство примитивно рекурсивности функции является одновременно доказательством существования алгоритма вычисления значений функций. Рассмотрим, например, функцию $\psi(x,y)= x \bullet y$. Выше установлено, что

$\psi(x,0) = Z(x)=0;$

$\psi(x,y+1)=f( \psi (x,y),x),$ где $f(x,y)=x+y.$

Таким образом, значение $\psi(x,y)$ при $y=0$ определено. Зная значение этой функции при некоторых $x$ и $y$, можем определить значение этой функции при $x$ и $y+1$, используя уже схему вычисления значений функции $f(x,y)$, для которой значение при $y=0$ известно, а при $y>0$ можно выразить через функцию $N$, зависящую от аргумента $f(x,y-1)$ и т.д.

Итак, схему задания примитивно рекурсивных функций можно рассматривать как процедуру вычисления ее значений. Вычисление ее значений определяется схемой достаточно просто, явно и осуществляется механическим образом.

**Теорема 6.9.** Множество примитивно рекурсивных функций является счетным множеством, множество общерекурсивных функций тоже является счетным множеством.

**Доказательство.** Все функции константы - примитивно рекурсивны, следовательно, и общерекурсивны. Тогда их не меньше, чем счетное множество. Доказательство того, что их не более чем счетно, приводится с помощью геделевой нумерации, т.е. удается показать, что каждой общерекурсивной функции можно сопоставить некоторое целое неотрицательное число (геделев номер), причем различным функциям

сопоставляются различные числа. Из этой нумерации и следует, что общерекурсивных функций не более чем счетное множество.

Можно доказать следующую теорему.

**Теорема 6.10**. Существуют арифметические всюду определенные функции, не являющиеся общерекурсивными функциями.

Функция $\varphi$ от $n$ аргументов называется *частично рекурсивной*, если она может быть получена из исходных функций 1), 2) и 3) с помощью конечного числа подстановок, рекурсий и $\mu'$ -оператора, где $\mu'$ - оператор определяется так же, как и $\mu$ -оператор, но уже не требуется, чтобы для $\forall x_1, \forall x_2, \ldots, \forall x_n$ существовал $y$ такой, что $g(x_1,x_2,\ldots,x_n,y)=0$, т.е. при некоторых значениях $x_1,x_2,\ldots,x_n$ может и не существовать $y$ такого, что

$g(x_1,x_2,\ldots,x_n,y)=0$.

Очевидно, что всякая общерекурсивная функция является частично рекурсивной, но не наоборот. Можно доказать следующие важные теоремы.

**Теорема 6.11.** Всякая частично рекурсивная функция (общерекурсивная) является частично вычислимой (вычислимой) по Маркову функцией.

**Теорема 6.12**. Всякая частичная функция, частично вычислимая (вычислимая) по Маркову, является частично рекурсивной (общерекурсивной) функцией.

## § 17. Лямбда-исчисление

Прежде чем ввести лямбда-исчисление укажем возможную область её использования в вычислительных моделях.

Различают следующие вычислительные модели:

*императивную (процедурную) вычислительную модель* (императив – повеление, требование), когда имеется последовательная система команд. Языки программирования, такие как АЛГОЛ, ФОРТРАН, С и т.п. основываются на этой вычислительной модели и объединяются под названием императивные или процедурные языки;

*логические вычислительные модели,* основанные на вычислениях с помощью логики предикатов. Примеры таких языков Пролог, Дейталог, Параллельный Пролог;

*функциональные вычислительные модели*, в которых программа рассматривается как множество определений функций. В основу этой модели

положено мак называемое «ламбда-исчисление». Примером языка, основанного на ламбда-исчислении, является язык ЛИСП;

*объектно-ориентированные вычислительные модели*. В императивных (процедурных) и функциональных вычислительных моделях операция принимается за субъект, а вычисления представляются в виде использования операций с некоторыми объектами. Однако если сделать наоборот, т.е. объект операции принять за субъект, а применение операций рассматривать как передачу запроса объекту, который и выполняет эту операцию, то такая модель является объектно-ориентированной, т.е. объекты рассматриваются как процессы. Реализованы в языках PLASMA, Mesa, LOOPS и др.

Теперь перейдем к некоторым понятиям ламбда-исчисления.

Ламбда-исчисление было введено Черчем около 1930 года как один из подходов для обоснования логики и математики. В дальнейшем часть этого исчисления явилось одной из теорий вычислений.

Ламбда-исчисление – это теория, рассматривающая функции как ***правила***, а не как графики и фактически выделяющая вычислительный аспект при введении функций.

Чёрч предложил использовать ламбда-исчисление для формализации понятия эффективной вычислимости. Тьюринг показал, что вычислимость по Тьюрингу эквивалентно ламбда-вычислимости. Таким образом, несмотря на свой очень простой синтаксис, ламбда-исчисление способно реализовать все механически вычислимые функции.

*Синтаксис ламбда-исчисления.*

Выражение в ламбда-исчислении представлено в префиксной форме, т.е. вначале располагается оператор, например, вместо $a+b$ пишется $+ab$, а вместо $a/b+b\times c$ пишется $(+(/ab)(\times bc))$.

Все программы представляются в виде выражений, а процесс выполнения программы заключается в определении значения этого выражения и это называется оценкой выражения.

Оценка выражения производится путём повторения операции выбора и упрощения тех частей этого выражения, которые можно упростить. Такая часть выражения называется *редексом*, а операция упрощения называется *редукцией*.

Процесс редукции завершается, когда выражение, преобразованное редукцией, не содержит больше редекса. Выражение, не содержащее редекса, называется *нормальной формой*.

Например, при оценке выражения (+(/62)(×25)) сначала выбираются редексы и упрощаются соответственно до 3 и 10. Полученное выражение (+3 10) также является редуцируемым, поэтому имеем

(+3 10)→13.

Результат (оценка выражения) 13 невозможно упростить, поэтому он считается нормальной формой.

Применение функции. Применение функции $f$ к аргументу $x$ обозначается как ($fx$). Функция от нескольких аргументов представляется, например, следующим образом ($f(x,y,z)$). Однако функцию от нескольких аргументов можно интерпретировать как результат синтеза функций от одного аргумента. Например (+3 4) можно интерпретировать как (+3)4, т.е. как прибавление тройки к аргументу 4.

*Ламбда-абстракция* – это одно выражение, определяющее функцию. Например, выражение

$(\lambda x.+x1)$

определяет функцию прибавления к переменной $x$ единицы. Запись $\lambda x.$ показывает, что это выражение является ламбда-абстракцией, *формальным параметром* которой является $x$. Выражение, которое следует за точкой (в данном случае это $+x1$) является телом ламбда-абстракции.

Рассмотрим следующее ламбда-выражение:

$(\lambda x.+xy)3$

Здесь $x$ считается связанной переменной, $3$ значением для $x$.

Для оценки этого выражения необходимо, чтобы переменная $y$ уже имела какое-либо значение. Обычно значение свободной переменной определяется во внешнем выражении, содержащем это выражение.

*$\beta$ - преобразование*

Пусть имеем выражение $(\lambda x.+x1)4$, в котором записаны последовательно лямбда-абстракция $(\lambda x.+x1)$ и аргумент $4$. При замене $x$ на $4$ получим $(+4\ 1)$. Это правило преобразования называется $\beta$ - преобразованием.

Примеры $\beta$ - преобразований:

$(\lambda x.+x1)4 \rightarrow +4\ 1$

$(\lambda x.+xx)5 \rightarrow +5\ 5 \rightarrow 10$

$(\lambda x.3)5 \rightarrow 3$

$(\lambda x.(\lambda y./yx))2\ 6 \rightarrow (\lambda y./y2)6 \rightarrow 6\ 2 \rightarrow 3$

$(\lambda f.f3)(\lambda x.+x1) \rightarrow (\lambda x.+x1)3 \rightarrow +3\ 1 \rightarrow 4$,

где $(\lambda x.+x1)$ – аргумент для предыдущей ламбда-абстракции.

Кроме того в ламбда-исчислении вводятся $\alpha$ - преобразование, $\eta$ - преобразование, рекурсии и т.д.

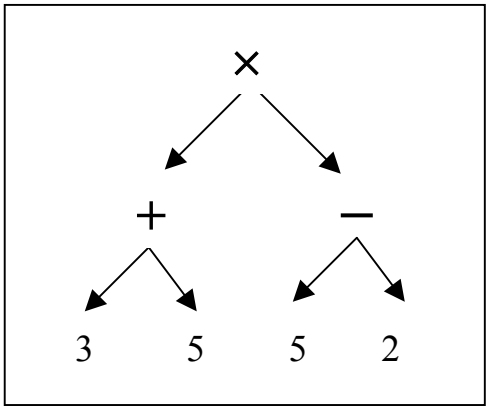



Рис. 6.6.

Если выражение содержит несколько редексов, то возможно одновременное их оценивание, например, если имеем $(x\ (+3\ 5)\ (-\ 5\ 2))$,

то можно представить это выражение в виде следующего графа, см. Рис. 6.6.

Посредством параллельной редукции всех редексов можно увеличить скорость вычислений. При этом появляется возможность параллельной обработки, и

имеются следующие преимущества по сравнению с процедурными языками.

(1) Графовая структура ламбда-исчисления позволяет выявить редексы для параллельной обработки.

(2) Нет необходимости в централизованном управлении операциями редукции, которые нужно делать для процедурных языков.

(3) Связь между редексами можно выполнить полностью унифицировано.

В процедурных языках задачи распараллеливания вычислений требуют специальных программ, в том числе и синхронизацию (по времени) этих вычислений. В языках с ламбда-исчислением этого не нужно. Сама структура программы однозначно определяет оптимальное число параллельных ветвей вычислений.

*Ты значишь то, что ты на самом деле.*
*Надень парик с мильонами кудрей,*
*Стань на ходули, но в душе своей*
*Ты будешь все таким, каков ты в самом деле. (Мефистофель).*

*И. В. Гете (Фауст)*

*Самое непостижимое в этом мире то, что он постижим.*

*А. Эйнштейн*

## § 18. Основные результаты

1. Из теорем 6.5, 6.6, следствий из них, а также теорем 6.11 и 6.12 следует, что класс частично рекурсивных функций совпадает с классом функций частично вычислимых по Маркову и классом частично вычислимых по Тьюрингу функций. Более общее: с помощью детальных рассмотрений показано, что формализации, предложенные Тьюрингом, Марковым (нормальные алгоритмы), а также Клини (частично рекурсивные функции) и другими, эквивалентны, т.е. в каждом случае получается один и тот же класс функций.

Всюду определенные функции, попадающие в этот класс, являются общерекурсивными функциями. Частичные функции этого класса совпадают с классом частично рекурсивных функций. Согласно основной гипотезе теории алгоритмов (принцип нормализации) любой алгоритм вполне эквивалентен нормальному алгоритму. Тогда совокупность частично рекурсивных функций совпадает с совокупностью частичных функций, вычислимых посредством алгоритма.

2. Изучалось обширное множество конкретных функций, интуитивно алгоритмических. Все они оказались частично рекурсивными функциями, т.е. были найдены наборы инструкций для них в какой-нибудь стандартной формализации (нормальные алгоритмы, или Тьюринговская схема, или показана их частично рекурсивность).

3. Доказательство указанных выше результатов обладает следующей общей структурой. В каждом случае тот факт, что один формализованный класс функций содержится в другом, доказывается путем предъявления и обоснования однообразной процедуры, согласно которой для любого набора инструкций $J$ одной формализации указывается набор инструкции $J'$ другой формализации, приводящей к той же самой функции. Эта единообразная процедура оказывается в каждом случае алгоритмической (в неформальном смысле этого слова).

*Если действовать не будешь*
*Ни к чему ума палата.*
*Ш. Руставели (Витязь в тигровой шкуре)*

## § 19. Вопросы и темы для самопроверки

1. Интуитивное понятие алгоритма, его свойства
2. Алфавит, слова, алгоритм в алфавите. Вполне эквивалентные в данном алфавите алгоритмы.
3. Нормальный алгоритм (алгоритм А.А.Маркова), задание, примеры.
4. Функции вычислимые и частично вычислимые по Маркову.
5. Замыкание, распространение нормального алгоритма.
6. Операции над нормальными алгоритмами: композиция, соединение, разветвление, повторение.
7. Машина Тьюринга, общее описание.
8. Задание машины Тьюринга, примеры.
9. Алгоритм Тьюринга. Вычислимость по Тьюрингу.
10. Связь между алгоритмами Тьюринга и нормальными алгоритмами.
11. Основная гипотеза теории алгоритмов (принцип нормализации или тезис Черча). Доказуема ли эта гипотеза?
12. Проблема алгоритмической неразрешимости.
13. Знаете ли вы алгоритмически разрешимые массовые проблемы?
14. Примеры алгоритмически неразрешимых массовых проблем.
15. Сведения любого преобразования слов в алфавите к вычислению значений целочисленных функций.
16. Примитивно рекурсивные и общерекурсивные функции.
17. Примитивно рекурсивность некоторых функций. Вычисление значений примитивно рекурсивных функций.
18. Частично рекурсивные функции, их связь с вычислимостью по Маркову и Тьюрингу.
19. Какие вы знаете вычислительные модели?
20. Ламбда исчисление. Синтаксис ламбда-исчисления.
21. $\beta$-преобразования в ламбда-исчислении.

## § 20. Упражнения

*Нормальные алгоритмы (алгоритм А. А. Маркова)*

1. Применим ли нормальный алгоритм

$$B=\begin{cases}*11\to 1\\ *1\to\bullet 1\\ \Lambda\to 1\end{cases}$$

к слову: а) $P=111$; б) $P=**$; в) $P=11*$; г) $P=*1*1$. Если «да», то указать результат применения.

2. Применим ли нормальный алгоритм

$$B=\begin{cases}*\to *\\ 1\to\bullet 11\\ \Lambda\to *\Lambda\end{cases}$$

к слову: а) $P=11111$; б) $P=*111*1$; в) $P=**$; г) $P=1*1*1$. Если «да», то указать результат применения.

3. Пусть $P=babbbc$ слово в алфавите $A=\{a,b,c\}$. нормальный алгоритм $\boldsymbol{B}$ задан схемой

$$\boldsymbol{B}=\{ab\to a.$$

в какое слово перерабатывает данный алгоритм слово $P$?

4. Пусть задан алфавит $A=\{b,c\}$ и нормальный алгоритм в алфавите $A$: $\boldsymbol{B}=\begin{cases}b\to\Lambda\\ c\to\Lambda\end{cases}$ Применим ли алгоритм $\boldsymbol{B}$ к любому слову в алфавите $A$? Если он применим к некоторому слову $P$, то в какое слово он его перерабатывает?

5. Пусть $A=\{a,b,c\}$ и $\boldsymbol{B}=\begin{cases}b\to\bullet\Lambda\\ a\to\bullet\Lambda\\ c\to c\end{cases}$ Как действует данный алгоритм? Применим ли данный алгоритм к любому слову в алфавите $A$?

6. Построить нормальный алгоритм, стирающий последнюю букву каждого непустого слова $P$ в алфавите $A$. Является ли построенный алгоритм алгоритмом в алфавите $A$ или над алфавитом $A$?

7. Пусть $A$ – русский алфавит. Построить нормальный алгоритм над алфавитом $A$, который преобразует слово «муха» в слово «слон», а любое другое слово в алфавите $A$ в пустое слово. При этом, если слово «муха» входит в некоторое слово $Q$, например $Q$=черемуха, то слово $Q$ алгоритм должен переработать в пустое слово.

8. Пусть $A$ – русский алфавит. Построить нормальный алгоритм над алфавитом $A$, который преобразует слово «слон» в слово «муха», а любое другое слово в алфавите $A$ в пустое слово. При этом, если слово «слон» входит в некоторое слово $Q$, например $Q$=заслон, то слово $Q$ алгоритм должен переработать в пустое слово.

9. Пусть заданы алфавит $A$ и некоторое непустое слово $Q$ в этом алфавите. Описать действие нормальных алгоритмов, задаваемых следующими схемами:

а) $\{\Lambda \to \bullet\Lambda$

б) $\{\Lambda \to \bullet\alpha$

в) $\begin{cases}\alpha \to \bullet\beta \\ \Lambda \to \alpha\end{cases}$

г) $\begin{cases}\alpha \to \alpha \\ \Lambda \to \alpha\end{cases}$

д) $\{\Lambda \to \alpha$

е) $\{\Lambda \to \bullet Q$

ж) $\begin{cases}\alpha x \to x\alpha \\ \alpha \to \bullet Q(\alpha \notin A, x \in A) \\ \Lambda \to \alpha\end{cases}$

з) $\{x \to \bullet\Lambda(x \in A)$

и) $\{x \to \alpha(x \in A, \alpha \notin A)$

10. Показать, что алгоритм, заданный схемой:

$$\begin{cases}\alpha\alpha \to \beta \\ \beta x \to x\beta \\ \beta\alpha \to \beta\ (\alpha, \beta \notin A, x, y \in A) \\ \beta \to \bullet\Lambda \\ \alpha xy \to y\alpha x \\ \Lambda \to \alpha\end{cases}$$

преобразует любое слово в алфавите $A$ в слово, образованное из тех же букв, но в обратном порядке.

11. Пусть $A$ – некоторый алфавит, $\alpha \notin A$; $B,C,D$, - заданные слова в алфавите $A$. Рассмотреть нормальный алгоритм над алфавитом $A$

$$F = \begin{cases}x\alpha \to \alpha x \\ \alpha x \to \alpha \\ \alpha \to \bullet D \\ Bx \to \alpha(\alpha \notin A, x \in A). \\ xB \to \alpha \\ B \to \bullet C \\ \Lambda \to \alpha\end{cases}$$

Показать, что этот алгоритм $F$ применим к любому слову в алфавите $A$, причем $F(P)=D$, $F(B)=C$.

12. Пусть $A=\{a,b,c\}$. Построить нормальный алгоритм, который к любому слову в алфавите $A$ будет приписывать справа слово $abb$.

13. Пусть $A=\{1\}$ и $B=\{1,*\}$. Для всякого натурального числа $n$ определим по индукции $\overline{0} = 1$ и $\overline{n+1} = \overline{n}1$. Таким образом, $\overline{1} = 11$, $\overline{2} = 111$ и т.д. Слова $\overline{n}$ называются цифрами. Поставим теперь в соответствие всякому вектору $(n_1,\ldots,n_k)$, где $n_1,\ldots,n_k$ – натуральные числа, слово $\overline{n_1} * \ldots * \overline{n_k}$ которое

обозначим через $\overline{(n_1,\ldots,n_k)}$. Так, например, $\overline{(3,1,2)}$ обозначает слово $111*11*111$:

а). Показать, что схема

$$F=\begin{cases}*\to *\\ \alpha 11\to \alpha 1\\ \alpha 1\to \bullet 1\\ \Lambda\to\alpha\end{cases}$$

определяет нормальный алгоритм $F$ над алфавитом $B$, применимый только к тем словам в алфавите $B$, которые являются цифрами, и такой, что $F(\overline{n})=0$ для любого $n$.

б) Показать, что нормальный алгоритм $G$ над алфавитом $B$, определяемый схемой

$$G=\begin{cases}*\to *\\ \alpha 1\to \bullet 11,\\ \Lambda\to\alpha\end{cases}$$

применим только к тем словам в алфавите $B$, которые суть цифры, причем $G(\overline{n})=\overline{n+1}$ для любого $n$.

в). Построить схему нормального алгоритма в алфавите $B$, перерабатывающего $\overline{(n_1,n_2)}$ в $\overline{(n_1-n_2)}$.

г). Построить нормальный алгоритм умножения на $2$.

14. Построить нормальный алгоритм над алфавитом $B=\{1,*\}$ для арифметических операций сложения и вычитания.

15. Построить нормальный алгоритм для умножения на фиксированное число $n$.

16. Пусть $A=\{1,*,a,b\}$. Показать, что следующий нормальный алгоритм

$b1\to 1b$ $\qquad$ $*1\to *$

$a1\to 1ba$ $\qquad$ $*\to\Lambda$

$a\to\Lambda$ $\qquad$ $b\to 1$

$1*\to *a$

производит умножение двух чисел, $n$ и $m$, записанных в алфавите $B=\{1,*\}$ в виде одного слова $\underbrace{111\ldots1}_{n}*\underbrace{111\ldots1}_{m}$.

17. Построить нормальный алгоритм $F$ над алфавитом $A$ такой, что для любого слова $P$ в $A$ было $F(P)=PP$.

18. Построить нормальный алгоритм для получения целой части при делении а) на $3$; б) на $n$.

19. Построить схему нормального алгоритма, равного композиции нормальных алгоритмов $F$ и $G$ в алфавите $A=\{1,*\}$:

$$F = \{111 \to \Lambda, \qquad G = \begin{cases} *1 \to 1* \\ * \to \bullet 1 \\ \Lambda \to * \end{cases}$$

Затем применить полученный алгоритм к слову: а) *P=1111111*; б) *P=*1111*; в) *P=1*111*; г) *P=***. Результат проверить, последовательно применяя к заданному слову алгоритм *F*, затем *G*.

20. Пусть имеем нормальный алгоритм *F* над алфавитом *A*, схема которого записана с использованием букв из *A* и произвольного конечного числа букв, не принадлежащих *A*. Обозначим эти буквы, не принадлежащие *A,* через $S_1,S_2,\ldots,S_n$, т.е. $B=A\cup\{S_1,S_2,\ldots,S_n\}$. Тогда алгоритм *F* над алфавитом *A* можно считать заданным в алфавите *B*. Показать, что для рассматриваемого алгоритма *F* существует вполне эквивалентный ему в алфавите *A* нормальный алгоритм, схема которого построена только с использованием букв алфавита $C=A\cup\{\alpha,\beta\}$ $(\alpha,\beta\notin B)$. Можно ли исключить одну из букв $\alpha$ или $\beta$, если $n>1$?

21. Для каждого из приведенных ниже алгоритмов над алфавитом $A=\{a,b\}$ найти вполне эквивалентный ему нормальный алгоритм, схема которого, записана только с использованием букв *a, b,* $\alpha$, $\beta$:

$$F_1 = \begin{cases} \alpha a \to a\varepsilon \\ \alpha b \to b\alpha \\ \alpha \to \beta\gamma \\ x\beta \to \beta x (x \in A) \\ \beta \to aa \\ \gamma \to bb \\ \Lambda \to \alpha \end{cases} \quad F_2 = \begin{cases} \alpha a \to \beta \\ \beta b \to b\gamma \\ \gamma b \to b \\ \gamma \to a \\ \Lambda \to \alpha \end{cases} \quad ; \quad F_3 = \begin{cases} a \to \alpha \\ \alpha b \to \beta \\ \alpha\beta \to \gamma \\ \gamma\gamma \to \delta \\ \delta \to \bullet b \\ \gamma \to a \end{cases}.$$

22. Пусть *A* – некоторый алфавит. Составьте нормальный алгоритм над *A*, позволяющий для произвольных слов *P* и *Q* в алфавите *A* выяснять, одинаковы эти слова или нет. (Указание: Рассмотрите слово *P*Q*(где $*\notin A$) и постройте алгоритм, сравнивающий в словах *P* и *Q* буквы, стоящие первыми слева и справа от ***, затем вторыми от *** и т.д.)

23. Пусть $A=\{0,1,2,\ldots,9\}$. Составьте нормальный алгоритм над *A*, который любое число *n*, записанное в десятичной системе счисления, преобразует в *n+1*.

24. Будем рассматривать нормальные алгоритмы в алфавите *A*. До сих пор мы применяли для их задания «схемы» - столбцы формул подстановок. Однако можно задавать каждый нормальный алгоритм в виде одного слова, которое получается следующим образом. Пусть $\alpha,\beta,\gamma\notin A$. Выпишем друг за другом в порядке очередности формулы подстановок алгоритма *F* заменой стрелки знаком $\alpha$, точки – знаком $\beta$ и присоединением после каждой подстановки знака $\gamma$. Получаемое так слово будем называть изображение алгоритма *F* и обозначать символом $F^*$. Алгоритм *F* называется

самоприменимым, если он применим к своей собственной записи, т.е. к слову $F^*$, и несамоприменимым в обратном случае.

Являются ли самоприменимыми следующие алгоритмы:

а) $F_1 = \begin{cases} a \to b \\ b \to \bullet b \end{cases} (a,b \in A)$

б) $F_2 = \{\Lambda \to \Lambda$

в) $F_3 = \begin{cases} a \to a \\ b \to \bullet \Lambda \end{cases} (a,b \in A)$

г) $F_4 = \begin{cases} ab \to \bullet \Lambda \\ \Lambda \to ab \end{cases} (a,b \in A)$

д) $F_5 = \begin{cases} *11 \to 1 \\ *1 \to \bullet 1 (*,1 \in A) \\ \Lambda \to 1 \end{cases}$

е) $F_6 = \begin{cases} * \to * \\ 1 \to \bullet 11 (*,1 \in A) \\ \Lambda \to * \end{cases}$

*Машины (алгоритмы) Тьюринга*

25. Дана машина Тьюринга

$q_0 1 L q_1$
$q_1 S_0 1 q_0$.

На ленте записано слово $P=1$ и читающая головка находится над этим словом, а машина находится во внутреннем состоянии $q_0$, иначе – задана начальная конфигурация $q_0 1$. Описать работу машины Тьюринга.

26. Задана машина Тьюринга

$q_0 a S_0 q_0$
$q_0 S_0 R q_0$
$q_0 b S_0 q_1$

и начальная конфигурация $q_0 P$. Применима ли данная машина к этой конфигурации, если

1) $P=aabbba$; 3) $P=acb$;
2) $P=cbb$; 4) $P=bac$.

Если машина применима к слову $P$, то чему равняется результат?

27. Пусть $A=\{1,2,3,...,9,0\}$. Построить машину Тьюринга $T_0$, которая любое число $n$ (в десятичной записи) перерабатывала бы в нуль, т.е. $T_0(n)=0$.

28. Пусть $A=\{1,2,3,...,9,0\}$. Построить машину Тьюринга $T_1$, которая любое число $n$ (в десятичной записи) перерабатывала бы в число $n+1$, т.е. $T_1(n)=n+1$.

29. Построить машины Тьюринга $T_1$ и $T_0$, перерабатывающие любые числа $n$ в $0$ и $n+1$ соответственно, при условии, что числа записаны только с использованием алфавита $A=\{1\}$, т.е. $n$ обозначено словом $\overline{n} = \underbrace{111...1}_{n+1}$.

30. Составить команды машины Тьюринга, которая будет считать записанные подряд (без пропусков) палочки и запишет их число:

1) в двоичной системе счисления;
2) в троичной системе счисления;

3) в системе счисления с основанием $n$.

31. На ленте записано число в системе счисления с основанием $n$. Составить команды машины Тьюринга, которая запишет число

1) непосредственно следующее за данным;
2) непосредственно предшествующее денному.

32. На ленте записано некоторое число слов $P_1,P_2,\ldots,P_k$ в алфавите $A$, разделенных звездочками $(*,\ *\notin A)$. Составить команды машины Тьюринга, которая считала бы количество слов и записывала бы их число:

1) в алфавите $\{1\}$;
2) в двоичной системе счисления;
3) в троичной системе счисления;
4) в системе счисления с основанием $n$.

33. Построить машину Тьюринга, которая приписывала бы справа от любого слова $P$ в алфавите $A$ слово $aab(a,b\in A)$.

34. Выяснить в какое слово перерабатывается слово $\overline{m}*\overline{r}$ машиной Тьюринга:

$q_0 1 S_0 q_0$ $\quad$ $q_1 * R q_1$ $\quad$ $q_2 * L q_2$
$q_0 S_0 R q_1$ $\quad$ $q_1 S_0 1 q_2$ $\quad$ $q_2 S_0 R q_0$
$q_1 1 R q_1$ $\quad$ $q_2 1 L q_2$.

(начальной конфигурацией является конфигурация $q_0\overline{m}*\overline{n}$).

35. Построить машину Тьюринга для умножения на $2$.

36. Построить машину Тьюринга для вычисления целой части частного при делении на $3$.

37. Построить машину Тьюринга для вычисления $|x\text{-}y|$.

38. На ленте записано некоторое число палочек (без пропусков). Составить команды машины Тьюринга, которая стирает каждую третью палочку, двигаясь слева направо, стирает каждую третью палочку из оставшихся и т.д. При этом машина должна указать последнюю стираемую палочку.

39. Машины Тьюринга заданные с помощью команд вида

$q_i S_j S_k q_r$
$q_i S_j R\ q_r$
$q_i S_j L\ q_r$

можно задать с помощью пяти-символьных команд вида

$q_i S_j q_r S_k Q,$

объединяющей две команды $q_i S_j S_k q_r$ и $q_i S_j Q\ q_r$, где $Q=R$, $Q=L$ или $Q=S$; при $Q=S$ читающая головка не передвигается ни влево и ни вправо.

Выяснить, применимы ли следующие машины Тьюринга к заданным словам.

1) $q_0 0 q_2 0 R$
$q_0 1 q_0 1 R$
$q_2 0 q_3 0 R$
$q_2 1 q_2 1 L$
$q_3 0 q * 0 S$

$q_3 1 q_2 1 R$, $P_1 = 1110111 = 1^3 01^3$, $P_2 = 10[01]^2 1$;

2) $q_0 0 q_2 1 R$
$q_0 1 q_3 0 R$
$q_2 0 q_3 1 L$
$q_2 1 q_2 1 S$
$q_3 1 q_0 1 S$, $P_1 = 1^3 01^2$, $P_2 = 1^6$;

3) $q_0 0 q_0 1 R$
$q_0 1 q_2 0 R$
$q_2 0 q_0 1 R$
$q_2 1 q_3 1 L$
$q_3 0 q_0 1 L$, $P_1 = 101^2$, $P_2 = 1^2 0^2 1$.

40. Построить в алфавите $\{0,1\}$ машину Тьюринга с пятисимвольными командами, обладающую следующим свойством:

1) машина применима к любому непустому слову в алфавите $\{0,1\}$;
2) машина не применима ни к какому непустому слову в алфавите $\{0,1\}$ и зона работы на каждом слове - бесконечная;
3) машина не применима ни к какому непустому слову в алфавите $\{0,1\}$ и зона работы на каждом слове ограничена одним и тем же числом ячеек, не зависящим от выбранного слова;
4) машина применима к словам вида $1^{3n}$, $n \geq 1$, и не применима словам вида $1^{3n+a}$, $n \geq 1$, $a = 1,2$.

41. Построить в алфавите $\{0,1\}$ машину Тьюринга с пятисимвольными командами, переводящую конфигурацию $K_0$ в $K^*$.

1) $K_0 = q_0 1^n$ $K^* = q^* 1^n 01^n$ $(n \geq 1)$;
2) $K_0 = q_0 0^n 1^n$ $K^* = q^*[01]$ $(n \geq 1)$;
3) $K_0 = 1^n q_0 0$ $K^* = q^* 1^{2n}$ $(n \geq 1)$.

42. Машину Тьюринга можно задать с помощью следующей таблицы:

| | $q_0$ | $q_1$ | | $q_i$ | | $q_n$ |
|---|---|---|---|---|---|---|
| $S_0$ | | | | | | |
| $S_1$ | | | | | | |
| | | | | | | |
| $S_j$ | | | | $q_r S_k Q$ | | |
| | | | | | | |
| $S_m$ | | | | | | |

На пересечении $i$-го столбца и $j$ –ой строки $(i \geq 0, j \geq 0)$ записывается выражение $q_r S_k Q$ являющееся частью команды

$$q_i S_j q_r S_k Q$$

если такая команда есть. Если команды начинающейся с $q_i S_j$ нет, то на пересечении $i$-го столбца и $j$ –ой строки ставится прочерк.

По заданной машине Тьюринга и начальной конфигурации $K_0$ найти заключительную конфигурацию для следующих вариантов.

1)

| | $q_0$ | $q_1$ |
|---|---|---|
| 0 | $q*1S$ | $q_00R$ |
| 1 | $q_10R$ | $q_11L$ |

*а)* $K_0=1^2q_01^301$, *б)* $K_0=1q_01^4$;

2)

| | $q_0$ | $q_1$ | $q_2$ |
|---|---|---|---|
| 0 | $q*0S$ | $q*1L$ | $q_01L$ |
| 1 | $q_11R$ | $q_20R$ | $q_00R$ |

*а)* $K_0=1q_01^5$, *б)* $K_0=q_01^3;01$, *в)* $K_0=10q_01^4$.

43. Доказать, что следующие функции примитивно-рекурсивны:

*а)* $f(x)=x+n$; *б)* $f(x,y)=x+y$;

*в)* $f(x,y)=x^y$; *г)* $f(x,y)=x\times y$;

*д)* $sg(x)=\begin{cases}0 & \text{если } x=0,\\ 1 & \text{если } x>0\end{cases}$; *е)* $\overline{sg}(x)=\begin{cases}0 & \text{если } x>0,\\ 1 & \text{если } x=0\end{cases}$.

# Глава 7. СЛОЖНОСТЬ ВЫЧИСЛЕНИЙ С ПОМОЩЬЮ АЛГОРИТМОВ

## § 1. Понятие о сложности

В предыдущей главе рассматривались проблемы принципиальной возможности вычислений и были исследованы различные подходы к понятию вычислимости. При этом не обращалось внимания на ресурсы времени и памяти. Однако существуют задачи, которые теоретически разрешимы, но при практической реализации требуют столь больших вычислений, что их решение практически неосуществимо. Следовательно, принципиальная алгоритмическая разрешимость ещё не означает практическую реализуемость. Рассмотрим некоторые характеристики вычислений.

Считаем, что при вычислении мы используем алгоритм.

Различают сложность описания алгоритма и исходных данных, и сложность применения алгоритма к исходным данным.

*Сложность описания алгоритма* зависит от выбора того или иного способа задания алгоритма. Если такой способ выбран (машина Тьюринга, нормальный алгоритм или рекурсивное задание и т.д.), то под сложностью алгоритма может быть введена как длина записи алгоритма, так и длина встречающихся выражений и т.д. Например, для машины Тьюринга её сложность может быть введена как число букв внешнего и внутреннего алфавитов.

Введем следующее определение. Говорят, что неотрицательная функция $g(n)$ есть $0(f(n))$ если существует такая постоянная $c$, что $g(n) \le cf(n)$ для всех, кроме конечного (возможно пустого) множества значений $n$, $n \in \{1, 2, 3, \ldots\}$. В этом случае записываем: $g(n)=0(f(n))$.

*Сложность исходных данных* понимается как длина (размер) их записи. Что такое размер входа? Всё зависит от того, что является входом. Размером входа, в общем случае, считают число символов, с помощью которых записан вход.

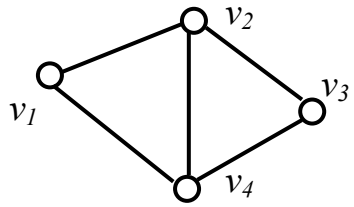
Пусть входом является целое число. Считаем, что число представлено в системе счисления с некоторым фиксированным основанием. В этих

системах число символов, необходимых для представления целого числа $n$ равно $]log_A n[$, где $A \geq 2$ – основание системы, а $]x[$ обозначает наименьшее целое $q$, такое, что $q \geq x$. Известно, что $log_B n=(log_2 n)/(log_2 B)$, здесь $log_2 B$ является константой при фиксированном $B$. Таким образом, число символов, необходимых для представления целого числа $n$ есть $0(log_2 n)$.

Рассмотрим второй пример. Пусть требуется перемножить квадратные $n \times n$ матрицы $A$ и $B$. Ясно, что подходящей мерой сложности исходных данных будет число пропорциональное $l= n^2$, имея в виду, что для запоминания отдельного числа (элемента матрицы) выделен определенный объем памяти.

Во многих задачах входом является граф. Граф $G=(V,X)$ можно, например, задать его матрицей смежностей $A_G=(a_{ij})$ размера $|V| \times |V|$, где $a_{ij}=1$, если ребро $(v_i,v_j) \in X$ и $a_{ij}=0$ в противном случае. Ясно, что максимальное число возможных ребер равно $M= C^2_{|V|}=0(|V|^2)$. Однако многие графы являются разреженными, т. е. число их ребер много меньше $M$. В этом случае лучше задать граф перечислением его ребер, с помощью *списков смежностей*.

При задании списков смежностей для каждой вершины $v \in V$ выписывается множество $A(v)$ $(A(v) \subseteq V)$ вершин, смежных с $v$, см. Рис. 7.1.




$A(v_1)=\{v_2, v_4\}$;
$A(v_2)=\{v_1, v_3, v_4\}$;
$A(v_3)=\{v_2, v_4\}$;
$A(v_4)=\{v_1, v_2, v_3\}$.

Рис. 7.1.

Размер этого представления зависит от суммы длин списков. Каждое ребро вносит вершину в два списка, поэтому сумма длин списков содержит $2|X|$ элементов. Но для различения вершин, как правило, вводятся числовые индексы. Так как имеется $|V|$ вершин, то для индексов потребуется $0(log_2|V|)$ двоичных или десятичных разрядов. Следовательно, при таком представлении графа $G=(V,X)$ потребуется $0(|X| log_2|V|)$ символов.

Более экономной записью информации в ячейках памяти ЭВМ (выделенного для одного числа) можно добиться, что для задания графа $G=(V,X)$ требуется $0(|X|)$ ячеек памяти.

Сложность вычислений с помощью алгоритма понимается как функция от размера входа алгоритма. Для оценки сложности вычислений существует много критериев. Важными критериями являются: временная сложность, характеризующая время, затраченное на вычисление, и ёмкостная

(ленточная) сложность, характеризующая необходимую для вычисления память, используемую для хранения промежуточных результатов.

Кроме того, сложность вычислений зависит от способа формулировки задачи. В качестве примера рассмотрим следующую задачу. Требуется узнать, является ли натуральное число $n$ простым или составным. Чтобы анализировать сложность задачи надо выяснить, как задано число $n$. Если $n$ задано как произведение своих простых делителей: $n=(p_1)^{k1}(p_2)^{k2},\ldots,(p_r)^{kr}$ $k_i \geq 0$, то задачи нет вообще; если $n$ задано в унарной форме, т.е. $n=11\ldots1$, то сложность записи равна $l=n$ (числу единиц), а сложность задачи, как можно показать, равна $\sqrt{l}=\sqrt{n}$ (если использовать известный способ решения этой задачи – решето Эратосфена, состоящий в том, что число $N$ последовательно делят на простые числа $p_1,p_2,\ldots,p_{\sqrt{n}}$. Если ни на одно из этих чисел $N$ не делится, то оно простое, иначе составное). Если число $n$ записано в десятичной форме, то сложность записи числа (длина записи) равна $l=\log_{10}n$, а сложность решения в этом случае равна $\sqrt{n}=2^{l/2}$, т.е. сложность решения представляет собой экспоненту от длины записи числа $n$.

Следует обратить внимание на то обстоятельство, что одной и той же задаче могут соответствовать разные языки, представляющие условия или входные данные задачи. Это связано со способами кодировки данных. Из всех языков представляющих исходную задачу, выбирается «разумный язык» или разумный способ кодировки её условий. Таким образом, каждой задаче соответствует “разумный язык” её представляющий.

## § 2. Временная сложность вычислений (алгоритма)

Временная сложность вычислений (алгоритма) характеризует число операций для решения задачи заданного размера.

При решении однотипных задач с одинаковым размером входа может потребоваться различное число итераций для решения отдельных задач (этого типа), следовательно, и различное число операций.

Определим *временную сложность алгоритма* как число операций в худшем случае по всем входам размера (длины) $n$. Иначе временная сложность алгоритма это функция, которая каждому входу размера $n$ ставит в соответствие максимальное (по всем индивидуальным задачам размера $n$) число операций, затрачиваемых алгоритмом на решение индивидуальной задачи этого размера.

Кроме меры сложности в наихудшем случае вводят *временную сложность алгоритма* $\boldsymbol{A}$ *в среднем* на входе размера $n$:

$$M_{\boldsymbol{A}}(n)=\sum_{a_i \in P_n} p(a_i)\mu(\boldsymbol{A}(a_i)),$$

здесь $p(a_i)$ – вероятность появления задачи $a_i$; $\mu(A(a_i))$ - число операций, затрачиваемых алгоритмом на решение индивидуальной задачи $a_i$; $P_n$ – множество рассматриваемых задач размера $n$, $P_n=\{ a_i\}$

При изучении сложности алгоритма, в основном интересуются его поведением при применении его к очень большим входам. Различие между сложностями $10n^3$ и $9n^3$ считаются несущественными, более важен показатель степени, а не коэффициент. Как уже отмечено, сложность алгоритма оценивается асимптотической сложностью, т.е. порядком роста числа операций при неограниченном росте размера входа. Например, если вход размера $n$ обрабатывается за время $cn^2$, где $c$ – некоторая постоянная, то сложность этого алгоритма есть $0(n^2)$, т.е. постоянная $c$ не содержится в оценке. Для практических задач величина этого коэффициента может быть важна, если они различаются существенно. Если время работы алгоритмов *A1* и *A2* пропорционально, например, $10^{10}n^2$ и $9n^2$ соответственно, то практическое использование алгоритма *A1* проблематично.

Для решения выбранной задачи иногда можно использовать различные алгоритмы.

Под *временной сложностью задачи* понимается временная сложность наилучшего алгоритма, известного для ее решения.

Выясним как изменяется эффективность различных алгоритмов с ростом быстродействия ЭВМ на которых реализуются эти алгоритмы.

Пусть имеем 5 алгоритмов различной сложности для решения одной и той же задачи. Положим. что каждое действие алгоритма осуществляется за 1 млс. Характеристики алгоритмов приведены в следующей таблице [29].

| Алгоритм | Временная сложность алгоритма | Максимальный размер задачи решаемой за указанное время | | |
|---|---|---|---|---|
| | | 1 с | 1 мин | 1 час |
| *A1* | $N$ | *1000* | *60 000* | *36 000 000* |
| *A2* | $N \log_2 N$ | *140* | *4 893* | *2 000 000* |
| *A3* | $N^2$ | *31* | *244* | *1 897* |
| *A4* | $N^3$ | *10* | *39* | *153* |
| *A5* | $2^N$ | *9* | *15* | *21* |

Из этой таблицы видно, что увеличение времени решения задачи, например с 1-ой секунды до одного часа позволяет для алгоритма *A1* увеличить размер решаемой задачи в 3600 раз, а для алгоритма *A5* только в 2,33 раза.

Предположим, что быстродействие ЭВМ возросло в 10 раз. В нижеследующей таблице показано, как при этом возрастут размеры входов [29].

| Алгоритм | Временная сложность алгоритма | Максимальный размер задачи | |
|---|---|---|---|
| | | до ускорения | после ускорения в 10 раз |
| *A1* | *N* | *S1* | *10 S1* |
| *A2* | $N \log_2 N$ | *S2* | *10 S2* |
| *A3* | $N^2$ | *S3* | *3,16 S3* |
| *A4* | $N^3$ | *S4* | *2,15 S4* |
| *A5* | $2^N$ | *S5* | *S5+3,3* |

Из последней таблицы видно, что увеличение быстродействия в 10 раз даёт возможность для алгоритма *A1* увеличить размер входа в 10 раз, а для алгоритма *A5* увеличение размера входа практически не произошло. Таким образом, чем большую временную сложность имеет алгоритм, тем меньшее улучшение даёт увеличение быстродействия.

## § 3. Полиномиальные алгоритмы и задачи. Класс *P*

Считается, что алгоритм – *полиномиальный* или имеет *полиномиальную временную сложность*, если существует полином *p(x)* такой, что на любом входном слове длины *n* алгоритм завершает вычисления после выполнения не более чем *p(n)* операций.

Ясно, что, алгоритмы *A1* и *A2*, временные сложности которых равны, например, $0(n^{3/2})$ и $0(n^2 \log n)$ будет считаться полиномиальными, ибо их сложности ограничены полиномами, т.е. имеют порядок не выше, чем $0(n^2)$ и $0(n^3)$ соответственно.

Говорят, что *задача разрешима за полиномиальное время* или полиномиально разрешима, если для неё существует полиномиальный алгоритм. При этом считается, что задача является «*хорошей*».

Множество всех задач, для каждой из которых существует полиномиальный алгоритм, называется *классом P*.

Среди полиномиальных алгоритмов быстрыми являются линейные алгоритмы, которые обладают сложностью порядка *n*, где *n* – размерность входных данных. К линейным алгоритмам относится школьный алгоритм нахождения суммы десятичных чисел, каждое из которых состоит из *n* цифр. Сложность этого алгоритма – *0(n)*.

В класс *P* кроме линейных задач попадают, например, следующие задачи.

Умножение целых чисел. Школьный метод умножения 2-х *n*-разрядных чисел имеет временную сложность порядка $0(n^2)$. Существует алгоритм Шёнхаге-Штрассена умножения чисел (заданных в двоичной системе счисления) с меньшей сложностью, именно со сложностью порядка $0(n \log_2 n \log_2 \log_2 n)$. Подробнее см. ниже.

Умножение матриц. Обычный метод имеет порядок сложности $0(n^3)$. Существует более быстрый алгоритм умножения матриц - алгоритм Штрассена [2] который имеет сложность порядка $0(n^{\log_2 7})$.

Найти кратчайший путь на графе состоящем из $n$ вершин и $m$ рёбер. Сложность алгоритма $0(mn)$ [28].

Быстрое преобразование Фурье требует $0(n \log_2 n)$ арифметических операций [2].

Задача Прима – Краскала – Кэли. Дано $n$ городов. Нужно соединить все города телефонным кабелем так, чтобы общая длина кабеля была минимальной. Эта задача решается с помощью жадного алгоритма сложности $0(n\log_2 n)$ [28].

Нахождение эйлерова цикла в графе с $m$ рёбрами. Алгоритм нахождения эйлерова цикла имеет сложность порядка $0(m)$, см., например, [22]

Задачи, для которых не существует полиномиального алгоритма, считаются трудно разрешимыми.

Рассмотрим пример определения сложности вычислений (алгоритма) на примерах.

Пусть задано множество $S$, содержащее $n$ действительных чисел. Требуется найти наибольший и наименьший элементы из $S$. Положим, что каждое одно сравнение двух любых чисел осуществляется за одинаковое время.

Один из возможных методов состоит в поиске сначала наибольшего элемента из $S$, а затем – наименьшего. Наибольший элемент можно найти, проводя $n-1$ сравнений, например по следующему алгоритму.

**begin**
MAX←произвольный элемент из $S$;
**for** все другие элементы $x$ из $S$ *do*
**if** $x$>MAX **then** MAX←$x$;
**end**.

В результате $n-1$ сравнений найдётся наибольший элемент. Заметим, что не учитывается время на выборку элемента. Далее начинается поиск наименьшего элемента по аналогичному алгоритму. Если считать эти процедуры независимыми, то вновь потребуется $n-1$ сравнений. В итоге для нахождения наибольшего и наименьшего элементов из $S$ потребуется $2n-2$ сравнений.

Число необходимых сравнений можно уменьшить, если использовать принцип «разделяй и властвуй», который в теории алгоритмов называют ещё стратегией дублирования.

Стратегия дублирования состоит в следующем. Пусть размер задачи (размер входных данных задачи) равен $n$. Разобьём задачу на две подзадачи размера $n/2$ той же структуры, что и исходная задача. Если решения этих задач можно скомбинировать в решение исходной задачи, то получится эффективный алгоритм.

Рассмотрим, как стратегия дублирования даёт ускорение для решения предыдущей задачи. Положим, что число элементов множества $S$ является степенью числа $2$, т.е. $n=2^k$, для некоторого $k$, $k \geq 1$.

Реализуем рекурсивный поиск, при котором множество $S$ разбивается последовательно на два подмножества по следующей процедуре MAXMIN.

**procedure** MAXMIN($S$):
1. **if** $|S|=2$ **then**
   **begin**
2. пусть $S=\{a,b\}$;
3. **return**(MAX($a,b$),MIN($a,b$))
   **end**
   **else**
   **begin**
4. разбить $S$ на два равных подмножества $S_1$ и $S_2$;
5. ($max1, min1$)←MAXMIN($S_1$);
6. ($max2, min2$)←MAXMIN($S_2$);
7. **return**(MAX($max1, max2$). MIN($min1, min2$))
   **end**

В этой процедуре сравнения происходят только на 3-ем шаге, где сравниваются два элемента множества $S$ из которых оно и состоит, и на шаге 7, где сравниваются $max1$ с $max2$ и $min1$ с $min2$. Пусть $T(n)$ – число сравнений элементов множества $S$. Ясно, что $T(2)=1$. Если $n>2$, то $T(n)$ – общее число сравнений, произведённых в двух вызовах процедуры MAXMIN (строка 5 и 6), работающих на множествах размера $n/2$ и ещё два сравнения в строке 7. Таким образом,

$$T(n)=\begin{cases} 1, & \text{если } n=2, \\ 2T(n/2)+2, & \text{если } n>2. \end{cases} \qquad (7.1)$$

Решением рекуррентных уравнений (7.1) служит функция $T(n)=(3/2)n-2$. Таким образом, вместо $2n-2$ сравнений получили $(3/2)n-2$ сравнений чисел, т.е на $(n/2)$ сравнений меньше.

Рассмотрим второй пример. Пусть требуется умножить два $n$ разрядных двоичных чисел. При традиционном (школьном) алгоритме требуется $0(n^2)$ битовых операций. Применим стратегию дублирования и разобьем числа $x$ и $y$ на две равные части:

$x =$ | $a$ | $b$ |

$y =$ | $c$ | $d$ |

Считаем, что $n$ есть степень числа $2$. Тогда

$$xy=(a2^{n/2}+b)(c2^{n/2}+d)=ac2^n+(ad+bc)2^{n/2}+bd. \qquad (7.2)$$

Равенство (7.2) даёт способ вычисления *xy* с помощью четырёх умножений *(n/2)* разрядных чисел и нескольких сложений и сдвигов (умножений на степень числа *2*). Можно получить, что вместо $0(n^2)$ битовых операций нужно $0(n^{\log_2(3)}) \approx 0(n^{1,59})$ битовых операций. Здесь число разбивалось на два блока. Разбивая эти числа на большее число блоков можно получить, что умножение двух чисел имеет сложность $0(n \log_2 n \log_2 \log_2 n)$ для алгоритма Шёнхаге-Штрассена [2].

Абстрактной моделью полиномиального алгоритма является так называемая детерминированная машина Тьюринга. Эта машина в каждый данный момент времени находится в строго определённом состоянии, за один шаг она совершает одно из некоторого конечного множества действий. Затем она переходит в следующее состояние и всё начинается вновь, пока не придёт к ситуации останова.

## § 4. *NP* класс

Наиболее простыми считаются полиномиальные задачи, т.е. задачи класса *P*.

Другим, возможно более широким, «сложностным классом» является *класс NP*. Эта аббревиатура обозначает выражение «разрешимых на Недетерминированной машине Тьюринга за Полиномиальное время». Класс *NP* стали впервые изучать Эдмонде, Кук и Кири. Оказалось, что многие известные задачи принадлежат к *NP* классу.

В обычных машинах Тьюринга (их называют детерминированными, чтобы отличать от недетерминированных) новое состояние, в которое машина переходит на очередном шаге, полностью определяется текущим состоянием и тем символом, который обозревает головка на ленте.

В недетерминированных машинах Тьюринга для каждого состояния может быть несколько следующих состояний, в соответствии с функцией перехода. И в каждом следующем состоянии запускается новая копия данной машины Тьюринга.

Недетерминированность лучше всего понять, рассматривая алгоритм, который производит вычисления до тех пор, пока не доходит до места, в котором должен быть сделан выбор из нескольких альтернатив. Детерминированный алгоритм исследовал бы сначала одну альтернативу, а потом вернулся для рассмотрения следующей альтернативы. Недетерминированный алгоритм может исследовать все альтернативы одновременно, «копируя», в сущности, самого себя для каждой альтернативы. Все копии работают независимо. Если копия обнаруживает, что данный путь безрезультатный, то она прекращает выполняться. Если копия находит требуемое решение, она объявляет об этом, и все копии прекращают работать.

Определим *NP класс* как класс задач, которые можно решить недетерминированными алгоритмами, работающими в течение полиномиального времени.

Чтобы доказать, что некоторая задача принадлежит классу *NP,* достаточно построить недетерминированный алгоритм (алгоритм недетерминированной машины Тьюринга), который решает эту задачу за полиномиальное время.

Пусть имеем, например, задачу выяснения существования в ориентированном графе гамильтонового цикла, длина которого меньше или равна *M*.

Рассмотрим следующий алгоритм.

**begin**

$v_1 \leftarrow 1$ {Пункт отправления}

$S \leftarrow \{2,3,\ldots,n\}$ {Множество вершин которые нужно посетить}

Длина пути $\leftarrow 0$ {Общая длина пути}

$nv \leftarrow 1$ {Число пройденных вершин}

{Пусть преемник $(v_{nv})$ обозначает допустимое, (не содержащее паразитных циклов) множество вершин, в которые можно попасть из $v_{nv}$}

**while** преемник $(v_{nv}) \neq \varnothing$ **do**

**begin**

$v_{nv-1} \leftarrow$ **ВЫБОР**(преемник $(v_{nv})$);

$nv \leftarrow nv+1;$

Длина пути $\leftarrow$ Длина пути + длина дуги $(v_{nv-1}, v_{nv})$

**end**

**if** $nv=n$ **and** Длина пути $\leq$ M **then** успех

**else** неудача

**end**

В этом алгоритме рассмотрение каждого варианта, т.е. последовательности соединённых дугами вершин $v_1, v_{i2}, v_{i3},\ldots v_{in}$ требует $n$ шагов. Следовательно, каждая процедура **ВЫБОР** (иначе каждая копия алгоритма просмотра одного пути) работает не более, чем полиномиальное время, точнее имеет сложность порядка $0(n)$. Таким образом, задача выяснения существования в ориентированном графе гамильтонового цикла, длина которого меньше или равна *M,* является *NP* задачей.

Детерминированная машина Тьюринга является частным случаем недетерминированной машины Тьюринга (которая не имеет копий), поэтому имеем, что $P \subseteq NP$.

Вопрос о том, будет ли $P=NP$ является открытой проблемой теории сложности. Широко распространено мнение, что $P \neq NP$, следовательно, $P \subset NP$.

Примеры задач из класса *NP*:

1) выяснение выполнимости формулы логики высказываний, записанной в к.н.ф.;

2) нахождение целочисленных решений системы линейных уравнений;

3) задача распознавания простого числа;

4) выяснение гамильтоновости графа;

5) задача коммивояжёра;

6) размещение обслуживающих центров (телефон, телевидение, срочные службы) для максимального числа клиентов при минимальном числе центров;

7) оптимальный раскрой (бумага, стальной прокат, отливка), оптимизация маршрутов в воздушном пространстве, инвестиций, станочного парка;

8) составление расписаний, учитывающих определённые условия;

9) оптимальная загрузка ёмкости (рюкзак, поезд, корабль, самолёт) при некоторых условиях;

10) динамическое распределение памяти. Раскроем эту проблему. Пусть заданы $A$ – множество элементов данных, размер $s(a) \in Z^+$, $Z^+=\{1,2,3,\ldots\}$, каждого элемента данных $a \in A$, время поступления $r(a) \in Z_0^+$, $Z_0^+=\{0,1,2,3,\ldots\}$ и время $d(a) \in Z^+$ окончания работы с элементом данных $a \in A$, положительное целое число $D$ – размер области памяти. *Вопрос*. Существует ли для множества элементов данных допустимое распределение памяти? Иначе говоря, существует ли такая функция $\sigma: A \to \{1,2,3,\ldots\}$, что для каждого элемента $a \in A$ интервал

$l(a)=[\sigma(a), \sigma(a)+s(a)-1]$

содержится в $[1, D]$, причём для любых $a, a^* \in A$, если $l(a) \cap l(a^*) \neq \varnothing$, то либо $d(a) \leq r(a^*)$, либо $d(a^*) \leq r(a)$.

В случае, когда размеры всех элементов данных одинаковы, то задача решается за полиномиальное время [10];

11) организация памяти в виде корневого дерева. Задача состоит в следующем [10]. Заданы конечное множество $X$, набор $C=\{X_1,X_2,\ldots,X_n\}$ подмножеств множества $X$, положительное целое число $K$. *Вопрос*. Существует ли такой набор $C=\{X_1^*, X_2^*,\ldots,X_n^*\}$ подмножеств множества $X$, что $X_i \subseteq X_i^*$ при всех $i$, $1 \leq i \leq n$, $\sum_{i=1}^{n} | X_i^* \setminus X_i | \leq K$ и существует ориентированное корневое дерево $T(x,A)$, в котором элементы каждого из подмножеств $X_i^*$, $1 \leq i \leq n$, образуют ориентированный путь?

12) достаточность числа регистров для реализации циклов. Задача состоит в следующем. Заданы множество $V$ параметров циклов, длина цикла $N \in Z^+$, для каждого параметра $v$ начальное время $s(v) \in Z_0^+$ и продолжительность $l(v) \in Z^+$, целое положительное число $K$. *Вопрос*. Достаточно ли $K$ регистров для запоминания параметров циклов? Иными

словами, существует ли такое назначение регистров $f: V \to \{1,2,3,\ldots,K\}$, что если $f(u)=f(v)$ для некоторых $u,v \in V$, то из неравенства $s(u) \le s(v)$ следует, что $s(u)+l(u) \le s(v)$ и $s(v)+l(v)(mod\ N) \le s(u)$?

Если число $K$ заранее фиксировано, то задача разрешима за полиномиальное время [10].

## § 5. *NP*-полные и *NP*-трудные задачи

Рассмотрим сводимость задач. Хотя сведение одних задач к другим является традиционной математической деятельностью, оно до работ С. Кука не было подвержено самостоятельному исследованию. Именно С. Куку принадлежит честь выделения этого понятия как центрального в теории полиномиальной сводимости.

Говорят, что задача $Z_1$ сводится к задаче $Z_2$ если метод решения задачи $Z_2$ можно преобразовать в метод решения задачи $Z_1$. Сводимость называется полиномиальной если указанное преобразование можно осуществить за полиномиальное время.

Если одновременно задача $Z_1$ полиномиально сводится к задаче $Z_2$ и $Z_2$ полиномиально сводится к задаче $Z_1$, то задачи $Z_1$ и $Z_2$ *полиномиально эквивалентны.*

Задача называется *NP-трудной* если каждая задача из *NP* полиномиально сводится к ней.

*NP*-трудная задача имеет тот смысл, что эта задача не проще, чем «самая трудная в *NP*».

В классе *NP* выделяются *NP*-полные задачи. Задача называется *NP-полной*, если она входит в *NP* и каждая задача из *NP* полиномиально сводится к ней (т.е. является *NP*-трудной). ***NP*-полные задачи понимаются как самые трудные задачи из класса *NP***.

Класс *NP*- полных задач обладает следующими свойствами.

1. Никакую *NP*-полную задачу нельзя решить никаким известным полиномиальным алгоритмом, несмотря на настойчивые усилия многих блестящих исследователей.
2. Если бы удалось построить полиномиальный алгоритм для какой-нибудь *NP*-полной задачи, то это бы означало полиномиальную разрешимость каждой *NP*-полной задачи.

Основываясь на этих двух свойствах, многие высказывают гипотезу, что $P \neq NP$. Практическое значение понятия *NP*-полной задачи лежит именно в широко распространенном мнении, что такие задачи по существу труднорешаемы. Следовательно, при их решении в худшем случае потребуется экспоненциальное количество времени и не будет возможности решить на практике такие задачи, за исключением очень малого числа индивидуальных задач.

Первой задачей, для которой было доказано, что она является *NP*-полной, была проблема о выполнимости (см. задачу 1 в § 4), именно, имеет место следующая теорема.

**Теорема 7.1** (теорема Кука). Задача выяснения выполнимости формулы логики высказываний, представленной в к.н.ф., является *NP*-полной.

Все приведённые ранее примеры *NP*-задач (задачи 1-12) являются *NP*-полными задачами. Список *NP*-полных задач в настоящее время содержит несколько сотен задач и продолжает пополняться. Многие вновь установленные *NP*-полные задачи печатаются в журнале Journal of Algorithms. В действительности о большей части задач комбинаторной оптимизации известна либо полиномиальная разрешимость, либо *NP*-полнота, что является ещё одним подтверждением гипотезы $P \neq NP$.

## § 6. Класс *E*

Как уже указано, считается, что алгоритм имеет полиномиальную временную сложность, если существует полином $p(x)$ такой, что на любом входном слове длины $n$ алгоритм завершает вычисления после выполнения не более чем $p(n)$ операций. Если такого полинома не существует, *временная сложность считается экспоненциальной*. Таким образом для них число операций возрастает быстрее значений любого полинома.

Кроме этого определения существует и второе определение: алгоритм имеет экспоненциальную временную сложность если его временная сложность имеет порядок не меньше, чем $ca^n$, где $c > 0$, $a$ $(a > 1)$ - постоянные. Иначе, временная сложность $f(n)$ является экспоненциальной, если существуют постоянные $c > 0$, $a > 1$, что

$ca^n \leq f(n)$

для всех, кроме конечного числа значений $n$.

При первом определении, например задача, имеющая сложность порядка $0(n^{\log_2 n})$ ( $0(n^{\log_2 n}) = 0(2^{(\log_2 n)^2})$ ) будет отнесена к задаче с экспоненциальной временной сложностью, а по второму определению – нет. Отметим, что функция $n^{\log_2 n}$ хотя и растет быстрее любого полинома, но растёт медленнее, чем, например $2^n$.

Большинство экспоненциальных алгоритмов – это просто варианты полного перебора. Экспоненциальные алгоритмы не считаются «хорошими», в отличие от полиномиальных алгоритмов, которые, как уже указывалось, считаются «хорошими». К экспоненциальным задачам относятся задачи, в которых требуется построить множество всех подмножеств данного

множества, все полные подграфы некоторого графа или же все поддеревья некоторого графа.

Некоторые экспоненциальные алгоритмы весьма хорошо зарекомендовали себя на практике. Дело в том, что временная сложность определяется для худшего случая, а многие задачи могут быть не так плохи.

Для решения задачи линейного программирования существует так называемый симплекс-метод (алгоритм), который хотя и экспоненциален в худшем случае – уверено работает на практике для задач достаточно большого размера. Отметим, что для решения задачи линейного программирования Хачиян Л. Г. в 1979 г. предложил алгоритм эллипсоидов, который имеет полиномиальную временную сложность.

Метод ветвей и границ успешно решает задачу о рюкзаке, хотя этот алгоритм имеет экспоненциальную временную сложность.

Примеров экспоненциальных алгоритмов успешно используемых на практике мало. Более того даже для полиномиальных алгоритмов представляют практический интерес алгоритмы сложность которых имеет порядок $n^2$ или $n^3$. Ясно, что алгоритмы со сложностью порядка $n^{100}$ вряд ли будут практически используемы.

Интересно, что экспоненциальный алгоритм может оказаться при малом размере задачи более быстрым, чем полиномиальный, однако с ростом размера задачи полиномиальный становится более быстрым.

Экспоненциальная сложность задачи имеет следующие аспекты:

-для отыскания решения требуется экспоненциальное время;

-искомое решение может быть настолько велико, что не может быть представлено выражением, длина которого была бы ограничена полиномом от длины входа.

Вторая ситуация возникает, например, если в задаче коммивояжера требуется найти все маршруты длины, не превосходящей заданной величины $L$. Может оказаться, что имеется экспоненциальное число маршрутов длины не превосходящей $L$.

## § 7. Ёмкостная (ленточная) сложность алгоритма

Ёмкостная или ленточная сложность алгоритма характеризует необходимую для вычисления память для хранения исходных данных, промежуточных результатов и окончательного результата. При применении машины Тьюринга как модели вычислений, ёмкостная сложность оценивается длиной части ленты используемой для записи исходных данных и вычислений.

Пусть машина Тьюринга вычисляет значение функции $f$.

Активной зоной ленты (машины Тьюринга) в данный момент времени $t$ называется связанная часть ленты, содержащая обозреваемую ячейку и все ячейки в которых имеются символы из внешнего алфавита машины.

Активной зоной при работе машины Тьюринга на числе $n$ называется объединение всех активных зон за время вычисления.

Определим $S_f(x)$ как длину активной зоны при работе машины $T$ на числе $x$, если $f(x)$ определено. В противном случае значение $S_f(x)$ будем считать неопределённым. Функцию $S_f(x)$ назовём *ленточной сложностью*.

Отметим, что в настоящее время уделяется существенно больше внимания временным характеристикам и значительно меньше ёмкостным, хотя эта характеристика тоже существенна при использовании ЭВМ с ограниченной памятью.

Для задач вводятся понятия задач с полиномиальной потребной памятью, с *NP*-потребной памятью и т.д.

Имеет место следующая теорема:

**Теорема 7.2.** Для ленточной (ёмкостной) характеристики сложности вычисления функции $f$ имеет место оценка

$S_f(x) \le x+1+ t_f(x)$,

где $t_f(x)$ -временная сложность вычисления $f(x)$.

**Доказательство**. В начальный момент на ленте машины Тьюринга записано число $x$ (в унарной системе), следовательно, занято $x+1$ клеток ленты. На каждом такте (шаге) активной становится не более одной новой клетки, поэтому получим указанную оценку для $S_f(x)$.

Эта теорема показывает, что если известна временная сложность, то можно оценить сверху ленточную сложность.

Интересно, что нет пределов сложности вычислений. Можно доказать, что какую бы сложную задачу не взять, то существует ещё более сложная задача.

## § 8. Вопросы и темы для самопроверки

1. Понятие о сложности вычислений.
2. Размер исходных данных задачи для случаев, когда входом является число, матрица, граф.
3. Временная сложность алгоритма. Временная сложность задачи.
4. Временная сложность алгоритма в наихудшем случае, временная сложность алгоритма в среднем.
5. Может ли, что для решения одной и той же задачи существовать алгоритмы разной сложности? Если да, то какова сложность задачи?
6. Полиномиальная сложность алгоритма, задачи.
7. Классификация задач по сложности.
8. Класс *P*, примеры задач из этого класса.
9. *NP* класс.
10. Недетерминированная машина Тьюринга, её отличие от детерминированной машины Тьюринга.

11. Задачи из *NP* класса.
12. *NP*-трудные и *NP*-полные задачи, в чем их различие?
13. Примеры *NP*-полных задач.
14. Класс *E*.
15. Емкостная (ленточная) сложность алгоритма, оценка ленточной сложности через временную сложность алгоритма.

## § 9. Упражнения

Рассмотрим известную задачу коммивояжера. Имеется $n$ городов $C=\{c_1,c_2,\ldots,c_n\}$ и известны расстояния $d(c_i,c_j)$ между каждой парой городов $c_i,c_j$ из $C$. Нужно найти обход городов чтобы побывать в каждом городе из $C$ один и только один раз, и вернуться в исходный город, т. е. найти упорядоченный набор $\langle c_{k1},c_{k2},\ldots,c_{kn}\rangle$ и этот набор должен минимизировать суммарное расстояние обхода:

$$S=\sum_{i=1}^{n-1} d(c_{ki},c_{k(I+1)})+d(c_{kn},c_{k1}). \qquad (7.3)$$

Эта задача, очевидно, алгоритмически разрешима, ибо можно перебрать все возможные варианты обходов и выбрать тот, для которого указанное суммарное расстояние минимально. Число обходов $n$ городов равно $M=(n-1)!/2$.

Легко найти, что число обходов $n$ городов равно $M=(n-1)!/2$. Известно [26], что число 50! имеет около 65 десятичных знаков. Пусть для выбора одного варианта обхода городов и вычисления величины $S$ по формуле (7.3) требуется $t_0=0{,}001$ секунд.

1. Оценить какое должно быть быстродействие ЭВМ которая переберет все варианты обхода городов задачи коммивояжера для $n=51$ за миллиард лет непрерывной работы.

2. Оценить время непрерывной работы ЭВМ с **современными** вычислительными возможностями для перебора всех указанных в задаче 1 вариантов.

3. Построить алгоритм нахождения наибольшего элемента множества $S$, содержащего $n$ действительных чисел, где $n$ – степень числа 2, на базе стратегии дублирования. Оценить временную сложность построенного алгоритма.

4. Построить алгоритм нахождения наименьшего элемента множества $S$, содержащего $n$ действительных неотрицательных чисел, на базе стратегии дублирования. Оценить временную сложность построенного алгоритма.

5. Пусть $A$ массив размера $n$, состоящий из целых чисел (положительных и отрицательных), причём $A_1<A_2<\ldots<A_n$. Построить алгоритм нахождения числа $i$, для которого $A_i=i$ (если такое $i$ существует). Каков порядок времени работы алгоритма.

6. Построить алгоритм умножения двух полиномов степени $n$, где $n=2k$, $k \in \{1,2,3,\ldots\}$, по стратегии дублирования и определить вычислительную сложность.

7. Доказать, что сложности умножения, деления, обращения и возведения в квадрат *(M(n), D(n), R(n) и S(n))* целых двоичных чисел размера $n$ совпадают с точностью до постоянных множителей и имеют порядок $0(n\log_2 n)$, если $n$ – степень числа 2.

8. Доказать, что сложности умножения, деления, обращения и возведения в квадрат *(M(n), D(n), R(n) и S(n))* полиномов степени $n$ совпадают с точностью до постоянных множителей и имеют порядок $0(n\log_2 n)$, если $n$ – степень числа 2.

*Никто не обнимет необъятного*
*Козьма Прутков*
*Одна услада в жизни учиться!*
*Петрарка*

## ЛИТЕРАТУРА


1. Акимов О. Е. Дискретная математика. Логика, группы, графы. -М.: А39 Лаборатория Базовых Знаний, 2001 –352 с.
2. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. -М.: Мир, 1979. –536 с.
3. Батыршин И. З. Основные операции нечёткой логики и их обобщения. - Казань: Отечество, 2001. –102 с.
4. Братко И. Программирование на языке ПРОЛОГ для искусственного интеллекта. Пер. с англ. -М.: Мир, 1990. –560 с.
5. Гаврилов Г. П., Сапоженко А. А. Сборник задач по дискретной математике. -М.: Наука, 1977. –368 с.
6. Гетманова А. Д. Учебник по логике. 2-ое издание. -М.: ВЛАДОС, 1995. –186 с.
7. Гильберт Д., Бернайс П. Основания математики. Логические исчисления и формализация арифметики. Пер. с нем. -М.: Наука, 1979. –560 с.
8. Гиндикин С. Г. Алгебра логики в задачах. М.: Наука, 1975. –288 с.
9. Горбатов В.А. Фундаментальные основы дискретной математики. Информационная математика: Учебник для втузов. -М.: Наука. Физматлит, 2000. -544с.
10. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. Пер. с англ. -М.: Мир, 1982. –416 с.
11. Ефимов Н. В. Высшая геометрия. -М.: Физматлит, 1961. –580 с.
12. Заде Л. Понятие лингвистической переменной и его применение к принятию приближенных решений. Пер. с англ. -М.: Мир, 1976. –168с.
13. Ивлев Ю. В. Модальная логика. -М.: Издательство МГУ, 1991. –224с.
14. Клини С. Математическая логика. Пер. с англ. -М.: Мир, 1973. –480с.
15. Klir G. J. and Folger T. A. Fuzzy Sets, Uncertainty and Information. Prentice Hall PTR, Englewood Cliffs, New Jersey, 1988 –356 p.
16. Кофман А. Введение в теорию нечётких множеств. Пер. с франц. -М.: Радио и связь, 1982. -432с.
17. Кузнецов О. П., Адельсон-Вельский Г. М. Дискретная математика для инженера. -М.: Энергоатомиздат, 1988. -480с.
18. Лавров И.А., Максимова Л.Л. Задачи по теории множеств, математической логике и теории алгоритмов. -М.: Наука, 1975. -240с.
19. Логический подход к искусственному интеллекту: от классической логики к логическому программированию: Пер. с франц. Тейз А., Грибомон П., Луи Д. И др. -М.: Мир, 1990. -432с.

20. Лорьер Ж. –Л. Системы искусственного интеллекта. Пер. с франц. - М.: Мир, 1991. –568 с.
21. Мендельсон Э. Введение в математическую логику. Пер. с англ. -М.: Наука, 1984. –320 с.
22. Нефедов В. Н., Осипова В. А. Курс дискретной математики. -М.: Издательство МАИ, 1992. -264 с.
23. Нечеткие множества в моделях управления и искусственного интеллекта. А. Н. Аверкин, Н. З. Батыршин, А. Ф. Блишун и др. Под ред. Д. А. Поспелова. -М.: Наука, 1986. –312с.
24. Новиков П.С. Элементы математической логики. -М.: Наука, 1973. - 400с.
25. Новиков Ф. А. Дискретная математика для программистов. –СПб.: Питер., 2001. –304 с.
26. Попадимитриу Х., Стайглиц К. Комбинаторная оптимизация. Алгоритмы и сложность. Пер. с англ. -М.: Мир, 1985. –512 с.
27. Проблемы Гильберта. Сборник под общей редакцией П. С. Александрова. -М.: Наука, 1969. –240 с.
28. Рейнгольд Э., Нивергельт Ю., Део Н. Комбинаторные алгоритмы. Теория и практика. Пер. с англ. -М.: Мир, 1980. – 478 с.
29. Самсонов Б. Б., Плохов Е. М. и Филоненков А. И. Компъютерная математика (основание информатики). -Ростов-на-Дону: «Феникс», 2002. –512с .
30. Чень Ч., Ли Р. Математическая логика и автоматическое доказательство теорем. Пер. с англ. -М.: Наука, 1983. -360 с.
31. Яблонский С. В. Введение в дискретную математику. -М.: Высшая школа, 2001. -384с.

# ПРИЛОЖЕНИЯ

## Варианты типового задания

1. Запишите приведенное высказывание в виде формулы логики высказываний. Для полученной формулы составьте таблицу истинности.
2. Упростите формулу логики высказываний, используя основные равносильности между формулами.
3. Составьте программу нахождения совершенной конъюнктивной нормальной формы (с.к.н.ф.) на любом известном вам алгоритмическом языке и найдите с.к.н.ф. для заданной булевой функции. Проверьте полученный результат, построив с.к.н.ф. равносильными преобразованиям.
4. Методом резолюций выясните, истинно ли приведённое утверждение. Решите эту задачу, используя два метода из следующих: метод исчерпания уровня, стратегия вычёркивания, лок-резолюция и табличный метод (последний для случая, если заданное множество является множеством хорновских дизъюнктов).
5. Запишите предложение в виде формулы логики предикатов.
6. Привести пример интерпретации, для которой данная формула истинна.
7. Получить предваренные нормальные формы и скулемовские стандартные формы для данных формул.
8. Записать предложения в виде соотношений формул логики предикатов. Методом резолюций выяснить будет ли заключение логическим следствием из посылок. Продемонстрировать результат с помощью диаграмм Эйлера-Венна.
9. Построить нормальный алгоритм для преобразования слова $P$ в слово $Q$, при условии что в каждой подстановке $P_i \rightarrow (\bullet) Q_i$ алгоритма число букв удовлетворяет неравенству: $|P_i| \le n$, $|Q_i| \le n$, где $n=2+[N](mod\ 3)$, здесь $N$ - ваш номер в списке группы, а $[N](mod\ 3)$ означает число $N$ по модулю три.
10. Построить машину Тьюринга для преобразования слова $P$ в слово $Q$.
11. Построить машину Тьюринга, которая будет считать записанные подряд (без пропусков) единицы (их число не превосходит $n$) и запишет их число в системе счисления с основанием $n+1$, здесь $n=3+[N](mod\ 13)$ и $N$ =(ваш номер в списке группы)+(номер вашей группы).
12. Доказать, что приведенная функция является примитивно рекурсивной.
13. Доказать методами исчисления высказываний, что данная формула является теоремой исчисления высказываний.

14. Выяснить, равносильны ли приведенные формулы в трёхзначной логике Лукасевича. Желательно сделать это с помощью разработанной вами программы на любом известном вам алгоритмическом языке.
15. Пусть нечеткие множества *A*, B** и *C** определены на универсальном множестве $U=\{x: 0 \le x \le 10\}$ функциями принадлежности:

$$\mu_{A^*}(x)=\frac{1}{1+nx},\quad \mu_{B^*}(x)=\left(\frac{1}{1+nx}\right)^{1/2},\quad \mu_{C^*}(x)=\left(\frac{1}{1+nx}\right)^{2},$$

здесь $n=1+[N](mod\ 25)$ и $N$ =(ваш номер в списке группы)+(номер вашей группы). Построить (в аналитическом виде и графическом) функции принадлежности для нечетких подмножеств, указанных для вашего варианта.

***Вариант 1***

1. *A* достаточно для *B*, а *B* необходимо для *C* или *A*, но *A* не эквивалентно *C*.
2. $A\&C\vee A\&D\&\neg C\vee A\&\neg C\vee D\&B\&\neg D\vee B\&A\&\neg C\&D\vee B\vee B\&B\&A\&\neg B$.
3. $A\&C\vee A\&B\&\neg C$.
4. $A\Rightarrow(B\Rightarrow C),\ B\vee C\vee A \models (A\Rightarrow C)\vee B$.
5. Все *A* суть не *B*, а некоторые *B* суть *C*, кроме того, существуют *A*, такие что *C*.
6. $\forall x\exists y P(x,y)\Rightarrow\forall x P(x,x)$.
7. $\boldsymbol{A}=\forall x\exists y P(x,y)\Rightarrow\forall x P(x,x)$, $\boldsymbol{B}=\exists x\,\forall y P(x,y)\Rightarrow\exists y\,\forall x P(y,x)$.
8. Ни одно *C* не есть *D*. Все *A* суть *D*. Все *B* суть *C*. Следовательно, все *B* не есть *A*.
9. $P=aabcc,\ Q=aabccdabcdd$.
10. $P=aabc,\ Q=aabccdab$.
11. Смотри условия задачи.
12. $\delta(x)=\begin{cases} x-1, & если\ \ x>0 \\ 0, & если\ \ x=0. \end{cases}$
13. $(\neg B\Rightarrow\neg A)\Rightarrow(A\Rightarrow B)$.
14. $((Nx)\Rightarrow(Ny))\&z,\ (y\Rightarrow x)\&z$.
15. $A^*\cap B^*,\ A^*\cup C^*,\ \overline{A}^*\cap(B^*\cup C^*)$.

***Вариант 2***

1. *A* необходимо для *B*, а *B* достаточно для *C* и *A*, но *A* не эквивалентно *C* или *B*.
2. $A\&\neg(C\vee D)\&(\neg A\vee B\vee\neg C\vee D)\&(\neg A\vee\neg C\vee D\vee C\&\neg B)\vee B\&A\&\neg B$.
3. $\neg A\vee C\vee A\&B\&\neg C$.
4. $\neg P\vee\neg Q\vee R,\ \neg P\vee\neg Q\vee S,\ P,Q \models S$.
5. Некоторые *B* суть не *A*, но ни одно *B* не есть *C*, тогда некоторые не *A* суть не *C*.
6. $\forall x P(x,a)\Rightarrow\exists y\,\forall x P(x,y)$.
7. $\boldsymbol{A}=\exists x\,\forall y P(x,y)\Rightarrow\forall x P(x,x)$, $\boldsymbol{B}=\forall x\exists y\,\forall z P(x,y,z)\Rightarrow\exists y\,\forall z Q(y,z)$.
8. Все *C* не есть *D*. Все *A* суть не *D*. Все *B* суть *C*. Следовательно, все *B* есть *A*.
9. $P=bbcab,\ Q=bbcabccdabcdd$.
10. $P=abab,\ Q=abababcdab$.

11. Смотри условия задачи. 12. $x\text{-}y=\begin{cases} x-y, & \text{если } x \geq y \\ 0, & \text{если } x < y. \end{cases}$

13. $(A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A)$. 14. $(N((Nx)\&y))\&z, (y \Rightarrow x)\&z$.

15. $A^* \cup B^*, A^* \cap C^*, \overline{A}^* \cup (B^* \cap C^*)$.

***Вариант 3***

1. $A$ когда $B$ или $C$, а $B$ необходимо для $A$ или $C$, но из $C$ следует $A$ и $B$.
2. $(A \vee C)\&(D \vee (\neg A \& \neg C)) \vee (\neg D \& \neg A) \vee (A \vee C)\&(\neg C \vee \neg D) \vee B\&A\&\neg A$.
3. $A \Rightarrow (B \Rightarrow C)$.
4. $P \vee Q, \neg P \vee Q, P \vee \neg Q \models P\&Q$.
5. Все $A$ суть не $B$, а некоторые $C$ суть $B$ или некоторые $C$ суть не $A$ и $B$.
6. $\forall x \forall y P(x,y) \Rightarrow \forall x P(x,x)$.
7. $\boldsymbol{A} = \exists y \forall x P(x,y) \Rightarrow \forall x P(a,x)$, $\boldsymbol{B} = \forall x \exists y P(x,y) \Rightarrow \exists z \exists y \forall x Q(y,x,z)$.
8. Некоторые $C$ не есть $D$. Все $A$ суть не $D$. Все $B$ суть $C$. Следовательно, все $B$ суть $A$.
9. $P=ccaab, Q=ccaabccabcdd$. 10. $P=cabc, Q=cabccdab$.
11. Смотри условия задачи. 12. $|x\text{-}y| = \begin{cases} x-y, & \text{если } x \geq y \\ y-x, & \text{если } x < y. \end{cases}$
13. $A \Rightarrow (\neg B \Rightarrow \neg(A \Rightarrow B))$. 14. $((Nx) \vee (Ny))\&z, (N(y\&x))\&z$.
15. $\overline{A}^* \cap B^*, A^* \cup C^*, A^* \cap (B^* \cup C^*)$.

***Вариант 4***

1. Если $A$ то $B$, а $B$ достаточно для $C$ или $A$, но $A$ не эквивалентно не $C$.
2. $(A \vee \neg C)\&(\neg A\&D \vee B\&D \vee \neg A\&\neg D \vee B\&\neg D)\&(A \vee C) \vee B\&A\&\neg B$.
3. $(A \equiv C) \vee A\&B\&\neg C$.
4. $\neg P \vee \neg Q \vee R, P \vee R, Q \vee R \models R$.
5. Ни одно $C$ не есть $D$, но все $A$ суть $D$, а все $B$ суть $C$ или $A$.
6. $(\exists x P(x)) \vee \exists x Q(x) \equiv \exists x (P(x) \vee Q(x))$.
7. $\boldsymbol{A} = \forall x \exists y P(x, f(x,y)) \Rightarrow \forall x P(x,x)$, $\boldsymbol{B} = \forall x \exists y Q(x,a,y) \Rightarrow \exists y \forall x P(y,x)$.
8. Все $A$ суть не $B$, а некоторые $B$ суть $C$, следовательно, не существует $A$, таких что $B$ или $C$.
9. $P=ddaab, Q=ddaabbbabcdd$. 10. $P=ddccc, Q=ddcccbccab$.
11. Смотри условия задачи. 12. $sg^*(x) = \begin{cases} 1, & \text{если } x = 0 \\ 0, & \text{если } x \neq 0. \end{cases}$
13. $(A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)$. 14. $((Nx) \equiv (Ny))\&z, ((Ny) \Rightarrow (Nx))\&z$.
15. $A^* \cap C^*, B^* \cup C^*, \overline{A}^* \cup (B^* \cap C^*)$.

***Вариант 5***

1. $A$ тогда, когда $B$, а $B$ только тогда когда $C$ или $A$, но $A$ не достаточно для $C$.
2. $(A \vee D)\&(B\&\neg C \vee B\&D \vee C\&B \vee \neg B\&D)\&(\neg D \vee A) \vee B\&A\&\neg B$.
3. $A\&C \vee (A \Rightarrow B)\&\neg C$.

4. $P \vee Q,\ R \vee Q,\ R \vee S,\ \neg R \vee \neg P,\ \neg S \vee \neg Q \models R \& Q$.
5. Все $D$ суть $B$, ни одно $A$ не есть не $C$, также ни одно $B$ не есть $C$.
6. $(\exists x P(x)) \vee \exists x Q(x) \equiv \exists x \exists y (P(x) \vee Q(y))$.
7. $\boldsymbol{A} = \forall x \exists y P(x,y) \Rightarrow \forall x Q(a, f(x,y))$, $\boldsymbol{B} = \exists x \exists y P(x,y) \Rightarrow \exists y \forall x P(y,x)$.
8. Некоторые $C$ не есть $D$. Все $A$ суть $D$. Все $B$ суть не $C$. Следовательно, все $B$ есть $A$.
9. $P = ccdda,\ Q = ccdddacdabcdd$.
10. $P = acacb,\ Q = acacbccdab$.
11. Смотри условия задачи.
12. $x!$
13. $\neg A \Rightarrow (A \Rightarrow B)$.
14. $(Nx) \vee ((Ny) \& z),\ x \Rightarrow ((Ny) \& z)$.
15. $C^* \cap \overline{A}^*,\ A^* \cup C^*,\ A^* \cap (B^* \cup C^*)$.

***Вариант 6***

1. $A$ при условии, что для $B$, а $B$ влечёт $C$ и $A$, но $A$ не эквивалентно $C$.
2. $(\neg A \vee B \vee C \vee D) \& (\neg A \vee C \vee D \vee \neg C \& B) \& A \& \neg (\neg C \vee D) \vee B \& C \& \neg B$.
3. $A \Rightarrow C \vee A \& B \& \neg C$.
4. $P,\ Q \vee \neg P,\ \neg R \vee \neg P \vee \neg Q \models \neg R \& P$.
5. Не все $A$ суть не $B$, а некоторые $B$ суть $C$, кроме того, не существует $A$, таких что $B$.
6. $(\forall x P(x)) \& \forall x Q(x) \equiv \forall x (P(x) \& Q(x))$.
7. $\boldsymbol{A} = \forall x \exists y Q(a,x,y) \Rightarrow \forall x P(x,x)$, $\boldsymbol{B} = \forall y \exists x P(x, f(y)) \Rightarrow \exists y \forall x P(y,x)$.
8. Ни одно $C$ не есть $D$. Все $A$ суть не $D$. Некоторые $B$ суть не $C$. Следовательно, все $B$ суть $A$.
9. $P = abca,\ Q = abcacdaaad$.
10. $P = abc,\ Q = abccbaa$.
11. Смотри условия задачи.
12. $min(x,y)$.
13. $(\neg A \Rightarrow \neg A) \Rightarrow (A \Rightarrow A)$.
14. $((Nx) \equiv y) \& z,\ (x \Rightarrow y) \& (y \Rightarrow (Nx)) \& z$.
15. $A^* \cap \overline{B}^*,\ A^* \cup C^*,\ A^* \cap (B^* \cup C^*)$.

***Вариант 7***

1. $A$ необходимо для $B$, а $B$ когда $C$ и $A$, но $A$ не эквивалентно $C$ и $B$.
2. $(A \& C \vee D) \& (\neg A \vee \neg C) \vee D \& A \vee (\neg A \vee \neg C) \& (C \vee \neg D) \vee B \& \neg A \& \neg B$.
3. $\neg A \vee C \vee A \& B \Rightarrow \neg C$.
4. $\neg P \vee \neg Q \vee R,\ \neg P \vee \neg Q \vee S,\ P, Q \models S$.
5. Некоторые $B$ суть не $A$ и ни одно $B$ не есть $C$, но некоторые $A$ суть $C$.
6. $(\forall x P(x)) \& \forall x Q(x) \equiv \forall x \forall y (P(x) \& Q(y))$.
7. $\boldsymbol{A} = \forall x \forall y Q(a,b,x,y) \Rightarrow \forall x P(x,x)$, $\boldsymbol{B} = \exists x \exists y \forall z R(x,y,z) \Rightarrow \exists y \forall z P(y,z)$.
8. Все не $B$ суть $A$. Ни одно $A$ не суть $D$. Все $B$ суть $C$. Следовательно, все $D$ суть $C$.
9. $P = badb,\ Q = badbdabcdd$.
10. $P = abab,\ Q = abababd$.
11. Смотри условия задачи.
12. $min(x_1, x_2, \ldots x_n)$.
13. $(A \Rightarrow A) \Rightarrow (\neg A \Rightarrow \neg A)$.
14. $N((Nx) \vee y) \& z,\ (N(x \Rightarrow y)) \& z$.
15. $\overline{C}^* \cup B^*,\ A^* \cap C^*,\ A^* \cup (B^* \cap C^*)$.

***Вариант 8***

1. $A$ когда $B$ и $C$, а $B$ при условии, что $C$, но из $C$ не следует $A$ и $B$.

2. $A\&\neg C\vee((\neg A\vee D)\&(B\vee D)\&\neg A\&\neg D\&(B\vee\neg D))A\&C\vee C\&\neg C\&D$.
3. $A\Rightarrow B\&C$.
4. $\neg P\vee\neg Q\vee R,\ \neg P\vee\neg R,\ P,\ Q,\ \neg R\vee S,\ \neg R\vee T\models S\&\neg T$.
5. Все $A$ суть не $B$, а некоторые $C$ суть $B$, но некоторые $C$ суть $A$ или не $B$.
6. $\forall x\,\forall y P(x,y)\Rightarrow\forall x P(x,x)$.
7. $\boldsymbol{A}=\exists y\,\forall x P(x,y)\Rightarrow\forall x P(a,x)$, $\boldsymbol{B}=\forall x\exists y P(x,y)\Rightarrow\exists z\exists y\,\forall x Q(y,x,z)$.
8. Все $D$ суть $E$. Все $C$ суть $A$. Ни одно $B$ не есть не $D$. Все $E$ суть не $A$. Следовательно, некоторые $A$ суть $D$.
9. $P=dccaa,\ Q=dccaabccabcdd$.
10. $P=ddbc,\ Q=ddbcccdab$.
11. Смотри условия задачи.
12. $max(x,y)$.
13. $A\Rightarrow(\neg A\Rightarrow\neg(A\Rightarrow A))$.
14. $((Nx)\ \&\ (Ny))\vee z,\ N(y\&x)\vee z$.
15. $\overline{A}*\cup B*,\ A*\cap C*,\ A*\cap(B*\cup C*)$.

***Вариант 9***

1. $A$ необходимо для $B$, а $B$ достаточно для $C$ или $A$, но $A$ не эквивалентно $C$.
2. $\neg A\&B\&\neg C\&D\vee A\vee\neg(C\&D)\vee(\neg A\&\neg C\&\&\neg D\&(C\vee\neg B)\vee\neg A\&B\&A$
3. $(A\equiv C)\vee A\equiv B$.
4. $A\Rightarrow B,\ C\Rightarrow D,\ A\vee C,\ A\Rightarrow\neg D,\ C\Rightarrow\neg B\models(A\vee B)\Rightarrow A\&B$.
5. Все $C$ суть $D$, а все $A$ суть не $D$, но некоторые $B$ суть $C$.
6. $\forall x(P(x)\Rightarrow\neg Q(x))\Rightarrow\neg((\exists x(P(x))\&\forall x Q(x))$.
7. $\boldsymbol{A}=\forall x P(x,x)\Rightarrow\forall x\exists y P(x,f(x,y))$, $\boldsymbol{B}=\forall x\exists y Q(x,a,y)\Rightarrow\exists y\,\forall x P(y,x)$.
8. Все $A$ суть $B$, а некоторые $B$ суть $C$, следовательно, существует $A$, такое что $B$ и $C$.
9. $P=dadab,\ Q=dadabbabcdd$.
10. $P=dadc,\ Q=dadccbccab$.
11. Смотри условия задачи.
12. $max(x_1,x_2,\ldots x_n)$.
13. $(A\Rightarrow A)\Rightarrow((\neg A\Rightarrow A)\Rightarrow A)$.
14. $(Nx\equiv Ny)\Rightarrow z,\ ((Ny)\Rightarrow(Nx))\Rightarrow z$.
15. $A*\cap C*,\ \overline{A}*\cup C*,\ A*\cup(B*\cap C*)$.

***Вариант 10***

1. $A$ необходимо для $B$, а $B$ только тогда когда $C$ или $A$, но $A$ не достаточно для $C$.
2. $A\&D\vee(B\vee\neg C)\&(B\vee D)\&(C\vee B)\&(\neg B\vee D)\vee\neg D\&A\vee B\&A\&\neg A$.
3. $A\vee C\vee(A\Rightarrow B)\&C$.
4. $A\vee B,\ A\Rightarrow B,\ B\Rightarrow(C\Rightarrow\neg D),\ A\Rightarrow D\models\neg(A\&C)$.
5. Все $D$ суть не $B$, но ни одно $A$ не есть $C$, а некоторые $B$ не есть $C$.
6. $(\exists x P(x))\vee\exists x Q(x)\equiv\exists x\exists y(P(x)\vee Q(y))$.
7. $\boldsymbol{A}=\forall x\exists y P(x,y)\Rightarrow\forall x Q(a,f(x,y))$, $\boldsymbol{B}=\exists x\exists y P(x,y)\Rightarrow\exists y\,\forall x P(y,x)$.
8. Некоторые $C$ суть $D$. Все $A$ суть $D$. Все $B$ суть не $C$. Следовательно, все $B$ есть $A$.
9. $P=ccaad,\ Q=ccaaddacdcdd$.
10. $P=cacab,\ Q=cacabcdc$.
11. Смотри условия задачи.
12. $rm(x,y)$=остатку от деления $y$ на $x$.
13. $(A\Rightarrow\neg A)\Rightarrow A$.
14. $(Nx)\Rightarrow((Ny)\&z),\ x\vee((Ny)\&z)$.

15. $C^* \cup \overline{A}^*$, $A^* \cup C^*$, $A^* \cup (B^* \cap C^*)$.

***Вариант 11***

1. $A$ необходимо для $B$, а $B$ достаточно для $C$ или $A$, но $A$ не эквивалентно $C$.
2. $B \& D \vee (C \vee \neg A) \& (\neg A \vee \neg B) \& (A \vee C) \& (A \vee \neg B) \vee B \& \neg D \vee B \& A \& \neg B$.
3. $A \& C \vee ((A \vee B) \Rightarrow \neg C)$.
4. $B \Rightarrow C$, $\neg A \equiv B$, $\neg C \equiv D \models (C \Rightarrow B) \Rightarrow (D \Rightarrow A)$.
5. Если все $A$ суть $B$, а некоторые $B$ суть $C$, тогда существуют $A$, такие что $C$.
6. $P(x) \Rightarrow \forall y P(y)$.
7. $\boldsymbol{A} = \forall x \exists y Q(x,a,y) \Rightarrow \exists x P(x,f(x))$, $\boldsymbol{B} = \forall x \exists y P(x,y) \Rightarrow \forall y \forall x Q(a,y,x)$.
8. Ни одно $C$ не есть $D$. Все $A$ суть $D$. Все $B$ суть $C$. Следовательно, все $B$ не есть $A$.
9. $P=aabcc$, $Q=aabccdabcdd$.
10. $P=aabc$, $Q=aabccdab$.
11. Смотри условия задачи.
12. $qt(x,y)$=частному от деления $y$ на $x$.
13. $(\neg A \Rightarrow \neg A) \Rightarrow (A \Rightarrow A)$.
14. $(Nx \Rightarrow Ny) \vee z$, $(y \Rightarrow x) \vee z$.
15. $A^* \cap C^*$, $A^* \cup B^*$, $C^* \cap (B^* \cup \overline{A}^*)$.

***Вариант 12***

1. $A$ необходимо для $B$, а $B$ достаточно для $C$ и $A$, но $A$ не эквивалентно $C$ или $B$.
2. $(A \vee (C \vee B \& C)) \& \neg (C \& D) \& C \& \neg D \& (C \vee \neg C \& \neg D \vee D) \vee B \& A \& \neg A$.
3. $\neg A \Rightarrow C \vee A \& B \& \neg C$.
4. $P \vee Q \vee R$, $\neg P$, $\neg Q$, $\neg R \vee S \models S$.
5. Если некоторые $B$ суть не $A$, но ни одно $B$ не есть $C$, то некоторые $A$ суть не $C$.
6. $\exists x P(x) \Rightarrow \forall x P(x)$.
7. $\boldsymbol{A} = \exists x \forall y P(x,y) \Rightarrow \forall x Q(a,x,x)$, $\boldsymbol{B} = \forall x \exists y \forall z S(x,f(y),z) \Rightarrow \exists y \forall z Q(y,z)$.
8. Все $C$ не есть $D$. Все $A$ суть не $D$. Все $B$ суть $C$. Следовательно, все $B$ есть $A$.
9. $P=bcab$, $Q=bcabccdabcda$.
10. $P=aba$, $Q=ababcdab$.
11. Смотри условия задачи.
12. $x+y \times z$.
13. $(A \Rightarrow A) \Rightarrow (\neg A \Rightarrow \neg A)$.
14. $N((Nx) \vee y) \& z$, $N((y \Rightarrow x) \vee z)$.
15. $A^* \cup B^*$, $\overline{A}^* \cap C^*$, $A^* \cup (B^* \cap C^*)$.

***Вариант 13***

1. $A$ достаточно для $B$ или $C$, а $B$ необходимо для $A$ и $C$, но из $C$ следует $A$ либо $B$.
2. $B \vee D \& (C \& \neg A \vee \neg A \& \neg B \vee A \& C \vee A \& \neg B) \& (B \vee \neg D) \vee \neg A \& A \& \neg B$.
3. $A \equiv (B \equiv C)$.
4. $\neg P \vee Q$, $P \vee R$, $\neg Q \models R$.
5. Если все $A$ суть $B$, а некоторые $C$ суть $B$, то некоторые $C$ суть не $A$.

6. $\forall x \exists y(P(x) \equiv \neg P(y))$.
7. $\boldsymbol{A} = \forall z \exists y \forall x Q(x,y,z) \Rightarrow \forall x P(a,x)$, $\boldsymbol{B} = \forall x \exists y P(x,y) \Rightarrow \forall z \exists y \forall x Q(y,f(a,x),z)$.
8. Некоторые $C$ не есть $D$. Все $A$ суть не $D$. Все $B$ суть $C$. Следовательно, все $B$ суть $A$.
9. $P=dcaab$, $Q=dcaabccabcdd$.
10. $P=dabc$, $Q=dabccdab$.
11. Смотри условия задачи.
12. $x \times (y+z)$.
13. $A \Rightarrow (\neg A \Rightarrow \neg(A \Rightarrow A))$.
14. $((Nx) \vee (Ny)) \Rightarrow z$, $N(y \& x) \Rightarrow z$.
15. $\overline{A}^* \cap \overline{B}^*$, $A^* \cup C^*$, $A^* \cup B^* \cup C^*$.

***Вариант 14***

1. Если $A$ то $B$ либо $C$, а $B$ достаточно для $C$, но $A$ не эквивалентно $C$.
2. $\neg(C \vee D) \vee A \& C \& (B \vee C) \vee C \vee \neg D \vee C \& (\neg C \vee D) \& D \vee D \& A \& \neg D$.
3. $(A \equiv C) \vee (A \Rightarrow B)$.
4. $\neg P \vee \neg Q \vee R$, $P \vee R$, $Q \vee R \models R$.
5. Когда ни одно $C$ не есть $D$, а все $A$ суть $D$, то все $B$ суть $C$.
6. $\forall x(P(x) \Rightarrow \neg Q(x)) \Rightarrow \neg((\forall x P(x)) \& \exists x Q(x))$.
7. $\boldsymbol{A} = \forall x \exists y Q(x,f(x,y,a)) \Rightarrow \forall x P(x,x)$, $\boldsymbol{B} = \forall x \exists y Q(x,a,y) \Rightarrow \exists y \forall x P(y,x)$.
8. Все $A$ суть не $B$. Некоторые $B$ суть $C$. Следовательно, не существует $A$, таких что $B$ или $C$.
9. $P=dcdaa$, $Q=dcdaabccabcd$.
10. $P=dcdca$, $Q=dcdcacbaca$.
11. Смотри условия задачи.
12. $(x+y)^z$.
13. $(A \Rightarrow A) \Rightarrow ((\neg A \Rightarrow A) \Rightarrow A)$.
14. $(Nx \Rightarrow Ny) \vee z$, $(y \Rightarrow x) \vee z$.
15. $\overline{A}^* \cap C^*$, $B^* \cup C^*$, $\overline{A}^* \cap B^* \cap C^*$.

***Вариант 15***

1. $A$ только тогда, когда $B$, а $B$ необходимо для $C$ или $A$, но $A$ не достаточно для $C$.
2. $A \& B \& C \vee A \& B \& \neg A \& C \vee (A \vee B) \& (C \vee D) \& (A \vee \neg B) \vee D \& B \vee C \& A \& \neg C$.
3. $(A \equiv C) \vee (A \Rightarrow B)$.
4. $A \Rightarrow (C \Rightarrow B)$, $D \Rightarrow A$, $C \models D \Rightarrow B$.
5. Когда все $D$ суть $B$, а некоторые $A$ есть $C$ или не $B$, то ни одно $B$ не есть $C$.
6. $(\exists x P(x)) \vee \exists x Q(x) \equiv \exists x \exists y (P(x) \vee Q(y))$.
7. $\boldsymbol{A} = \exists y \forall x Q(a,f(x,y)) \Rightarrow \forall x \exists y P(x,y)$, $\boldsymbol{B} = \exists x \exists y P(x,a(y)) \Rightarrow \exists y \forall x P(y,x)$.
8. Некоторые $C$ не есть $D$. Все $A$ суть $D$. Все $B$ суть не $C$. Следовательно, все $B$ суть $A$.
9. $P=ccda$, $Q=ccdacdaacdd$.
10. $P=accb$, $Q=accbccaab$.
11. Смотри условия задачи.
12. $(x+y)!$
13. $(\neg A \Rightarrow A) \Rightarrow (\neg A \Rightarrow A)$.
14. $(Nx) \vee ((Ny) \vee z)$, $x \Rightarrow ((Ny) \vee z)$.
15. $C^* \cap \overline{A}^*$, $A^* \cup \overline{B}^*$, $A^* \cap (B^* \cup C^*)$.

***Вариант 16***

1. $A$ достаточно для $B$, а $B$ влечёт $C$ либо $A$, но $A$ не эквивалентно $C$.

2. $(A \vee B \vee C) \& (\neg A \vee B \vee A \vee C) \& (A \& B \vee C \& D \vee A \& \neg B) \& (D \vee B) \vee B \& C \& \neg C$.
3. $A \Rightarrow (C \vee A) \equiv B$.
4. $C \Rightarrow (A \vee B)$, $D \Rightarrow (B \vee C) \models A \vee B$.
5. Когда не все $A$ суть не $B$, а некоторые $B$ суть $C$, тогда не существует $A$, таких что не $B$.
6. $\exists x \forall y P(x,y) \Rightarrow \forall y \exists x P(x,y)$.
7. $\boldsymbol{A} = \forall x \exists y P(x,f(y)) \Rightarrow \exists x P(x,x)$, $\boldsymbol{B} = \forall y \exists x Q(x,f(y)) \Rightarrow \exists y \forall x P(y,x)$.
8. Ни одно $C$ не есть $D$. Все $A$ суть $D$. Некоторые $B$ суть $C$. Следовательно, все $B$ суть $A$.
9. $P=abbc$, $Q=abbccdacad$.
10. $P=abb$, $Q=abbcbac$.
11. Смотри условия задачи.
12. $x^2+y$.
13. $\neg\neg(A \Rightarrow A) \Rightarrow (A \Rightarrow A)$.
14. $((Nx) \Rightarrow y) \equiv z$, $(x \vee y) \equiv z$.
15. $A^* \cap \overline{B}^*$, $A^* \cup C^*$, $(A^* \cap B^*) \cup \overline{C}^*$.

***Вариант 17***

1. Когда $A$ необходимо для $B$, а $B$ достаточно для $C$ и $A$, тогда $A$ не эквивалентно $C$ или $B$.
2. $(\neg A \vee \neg C) \& (C \vee D) \vee B \& \neg A \& \neg B \vee D \& (A \& C \vee D) \& (\neg A \vee \neg C) \vee D \& A$.
3. $\neg A \vee C \Rightarrow A \& B \Rightarrow \neg C$.
4. $A \Rightarrow (B \Rightarrow C)$, $B \vee C \vee D \models (A \Rightarrow C) \vee D$.
5. Если некоторые $B$ суть $A$, а ни одно $B$ не есть $C$, то все не $A$ суть $C$.
6. $\exists x(P(x) \& \forall y(P(y) \Rightarrow Q(x,y)))$.
7. $\boldsymbol{A} = \exists y \forall x Q(a,x,y) \Rightarrow \forall x P(x,x)$, $\boldsymbol{B} = \exists x \exists y R(x,y) \Rightarrow \exists y \forall x P(y,x)$.
8. Все не $B$ суть $A$. Ни одно $A$ не суть $D$. Все $B$ суть $C$. Следовательно, все $D$ суть $C$.
9. $P=ddad$, $Q=dbaddaccdd$.
10. $P=bdd$, $Q=bddabad$.
11. Смотри условия задачи.
12. $(x+y) \times z$.
13. $(A \Rightarrow A) \Rightarrow (\neg A \Rightarrow \neg A)$.
14. $N((Nx) \& y) \vee z$, $((N(x)) \Rightarrow y) \vee z$.
15. $\overline{C}^* \cap B^*$, $A^* \cup C^*$, $(A^* \cup B^*) \cap C^*$.

***Вариант 18***

1. $A$ если $B$ и $C$, а $B$ при условии, что $C$, но $C$ достаточно для $A$ и $B$.
2. $(A \vee \neg C) \& (\neg A \& D \vee B \& D \vee \neg A \vee \neg D \vee B \& \neg D) \& (A \vee C) \vee C \& \neg C \& D$.
3. $A \& \neg C \Rightarrow B \& C$.
4. $A \vee B$, $C \Rightarrow B$, $B \Rightarrow A$, $A \Rightarrow C \models B \& C \& A$.
5. Если все $A$ суть не $B$, а некоторые $C$ суть $B$, то некоторые $C$ суть не $A$ или $B$.
6. $\exists x \forall y(P(x,y) \Rightarrow \forall z Q(x,y,z))$.
7. $\boldsymbol{A} = \exists z \exists y \forall x Q(y,x,z) \Rightarrow \forall x P(a,x)$, $\boldsymbol{B} = \forall x \exists y P(x,y) \Rightarrow \exists y \forall x P(x,y)$.
8. Все $D$ суть не $E$. Все $C$ суть $A$. Ни одно $B$ не есть $D$. Все $E$ суть $A$. Следовательно, некоторые $A$ суть $D$.
9. $P=caa$, $Q=caabccabcdd$.
10. $P=ddc$, $Q=ddcccdab$.
11. Смотри условия задачи.
12. $x \times y^z$.

13. $A \Rightarrow A \Rightarrow \neg\neg(A \Rightarrow A))$. 14. $((Nx)\&(Ny)) \Rightarrow (Nz)$, $N(y \vee x) \Rightarrow (Nz)$.
15. $\overline{A}^* \cup \overline{B}^*$, $A^* \cap C^*$, $A^* \cap B^* \cap C^*$.

***Вариант 19***

1. $A$ эквивалентно не $B$, а $B$ необходимо для $C$ или $A$, но из $A$ не следует $C$ и $B$.
2. $\neg(C\&D) \vee \neg A\& \neg C\& \neg D \vee (C \vee \neg B)\&A \vee \neg A\&B\&A \vee \neg A\&B\& \neg C\&D \vee A$.
3. $\neg(A \equiv C) \vee A \Rightarrow B$.
4. $A \Rightarrow (B \vee C)$, $A \vee B$, $B \Rightarrow A$, $B \Rightarrow D \models C \vee D$.
5. Если некоторые $A$ суть $B$, но все $B$ суть $C$, то существует $A$, такое что $B$ и $C$.
6. $(\forall x P(x) \Rightarrow \exists x Q(x)) \equiv \exists x \exists y (P(x) \Rightarrow Q(y))$.
7. $\boldsymbol{A} = \forall x \exists y Q(x,a,y) \Rightarrow \forall x \exists y P(x,f(x,y))$, $\boldsymbol{B} = \forall x P(x,x) \Rightarrow \exists y \forall x P(y,x)$.
8. Не все $C$ суть $D$. Все $A$ суть $D$. Все $B$ суть не $C$. Следовательно, некоторые $B$ есть $A$.
9. $P=ddab$, $Q=ddabbabcdd$. 10. $P=ddc$, $Q=ddccbccab$.
11. Смотри условия задачи. 12. $x \times y + x \times z$.
13. $\neg(A \Rightarrow A) \Rightarrow ((A \Rightarrow A) \Rightarrow B)$. 14. $(Nx) \Rightarrow (y \Rightarrow z)$, $(z \Rightarrow (y \Rightarrow (Nx))$.
15. $A^* \cap C^*$, $\overline{A}^* \cup C^*$, $A^* \cup \overline{B}^* \cup C^*)$.

***Вариант 20***

1. $A$ только тогда, когда $B$, а $B$ достаточно для $C$ или $A$, но $A$ не эквивалентно $C$.
2. $(B \vee \neg C)\&(B \vee D)\&(C \vee B)\&(\neg B \vee D) \vee \neg D\&A \vee B\&A\& \neg A \vee A\&D$.
3. $A \vee C \Rightarrow (A \Rightarrow B)$.
4. $A \vee B$, $A \Rightarrow B$, $B \Rightarrow (C \Rightarrow \neg D)$, $A \Rightarrow D \models \neg(A\&C)$.
5. Когда не все $D$ суть $B$, а ни одно $A$ не есть $C$, тогда некоторые $B$ не есть $C$.
6. $(\forall x P(x) \Rightarrow \forall x Q(x)) \equiv \exists x \forall y (P(x) \Rightarrow Q(y))$.
7. $\boldsymbol{A} = \forall x Q(a,f(x,y)) \Rightarrow \forall x \exists y P(x,y)$, $\boldsymbol{B} = \exists x \exists y P(x,y) \Rightarrow \exists y \forall x P(y,x)$.
8. Все $C$ суть $D$. Некоторые $A$ суть $D$. Все не $B$ суть $C$. Следовательно, все $B$ есть $A$.
9. $P=ccad$, $Q=ccaadcacdabcdd$. 10. $P=cab$, $Q=cabccdcc$.
11. Смотри условия задачи. 12. $max(x_1,x_2,x_3)$.
13. $(A \Rightarrow \neg A) \Rightarrow (A \Rightarrow \neg A)$. 14. $(Nx) \Rightarrow (y\&z)$, $x \vee (y\&z)$.
15. $C^* \cup \overline{A}^*$, $\overline{A}^* \cap B^*$, $A^* \cap B^* \cap C^*$.

***Вариант 21***

1. Как $A$ так и $B$, а $B$ необходимо для $C$ или $A$, но $A$ не эквивалентно $C$.
2. $(A \vee B \vee \neg A\& \neg B)\&(B \vee C \vee B\&D)\& \neg(A \vee D)\&B\& \neg A \vee B\&A\& \neg B$.
3. $A\&C \vee A\&B \vee \neg C$.
4. $A \Rightarrow (B \Rightarrow C)$, $D \Rightarrow E \Rightarrow A$, $C \Rightarrow F \models B \Rightarrow (F \vee D)$.

5. Если все $A$ суть не $B$, а некоторые $B$ суть $C$, то существуют не $A$, такие что $C$.
6. $\exists x(P(x)\& \forall y(P(y)\Rightarrow Q(x,y)))$.
7. $\boldsymbol{A}=\forall xP(x,a)\Rightarrow \exists y\,\forall xP(y,x)$, $\boldsymbol{B}=\forall x\exists yP(f(x,y),y)\Rightarrow\exists y\,\forall xP(y,x)$.
8. Ни одно $C$ не есть $D$. Все $A$ суть $D$. Некоторые $B$ суть $C$. Следовательно, все $B$ не есть $A$.
9. $P=aabbc$, $Q=aabbccdabcdd$.
10. $P=aabd$, $Q=aabdcdab$.
11. Смотри условия задачи.
12. $(x+y)\times z$.
13. $(A\Rightarrow(A\Rightarrow A))\Rightarrow(\neg(A\Rightarrow A)\Rightarrow\neg A)$.
14. $((Nx)\Rightarrow y)\&z$, $((Ny)\Rightarrow x)\&z$.
15. $A^*\cap B^*$, $A^*\cup C^*$, $\overline{A}^*\cap(B^*\cup C^*)$.

***Вариант 22***

1. $A$ необходимо для $B$, а $B$ достаточно для $C$ и $A$, но $A$ не эквивалентно $C$ либо $B$.
2. $(\neg A\vee\neg C\vee D\vee C\&\neg B)\&\ A\&\ \neg(C\vee D)\&(\neg A\vee B\vee\neg C\vee D)\vee B\&A\&\neg B$.
3. $\neg A\vee C\vee B\Rightarrow\neg C$.
4. $(A\Rightarrow B)\Rightarrow(C\Rightarrow D)$, $(D\Rightarrow E)\Rightarrow F\models A\vee F$.
5. Когда некоторые $B$ суть не $A$, а ни одно $B$ не есть $C$, тогда некоторые не $A$ суть $C$.
6. $\exists x\,\forall y(P(x)\Rightarrow\neg Q(x,y))$.
7. $\boldsymbol{A}=\exists x\,\forall yR(a,f(x,b),y)\Rightarrow\forall xP(x,x)$, $\boldsymbol{B}=\forall x\exists yQ(x,y)\Rightarrow\exists y\,\forall xP(y,x)$.
8. Все $C$ есть не $D$. Все $A$ суть $D$. Все $B$ суть $C$. Следовательно, некоторые $B$ есть не $A$.
9. $P=bca$, $Q=bcacdaccdd$.
10. $P=abc$, $Q=abcbccab$.
11. Смотри условия задачи.
12. $(x+y)+z$.
13. $(A\Rightarrow A)\Rightarrow(\neg(A\Rightarrow A)\Rightarrow B)$.
14. $N((Nx)\Rightarrow y)\vee z$, $(y\Rightarrow(Nx))\vee z$.
15. $A^*\cup C^*$, $\overline{A}^*\cap B^*$, $\overline{A}^*\cup(B^*\cap C^*)$.

***Вариант 23***

1. $A$ когда $B$ либо $C$, а $B$ необходимо для $A$ и $C$, но из $C$ не следует $A$.
2. $(A\vee C)\&(D\vee(\neg A\&\neg C))\vee(\neg D\&\neg A)\vee(A\vee C)\&(\neg C\vee\neg D)\ \vee B\&A\&\neg A$.
3. $A\Rightarrow(B\equiv C)$.
4. $A,\ B\Rightarrow C,\ D\Rightarrow C,\ B\vee(A\Rightarrow D)\models C$.
5. Если все $A$ суть не $B$. Некоторые $C$ суть $B$, тогда некоторые $C$ суть $A$.
6. $\exists x\,\forall yP(x,y)\Rightarrow\ \forall y\exists xP(x,y)$.
7. $\boldsymbol{A}=\exists y\,\forall xR(x,b,y)\Rightarrow\forall xP(a,x)$, $\boldsymbol{B}=\exists z\exists y\,\forall xQ(y,x,z)\Rightarrow\forall x\exists yP(x,y)$.
8. Некоторые $C$ есть $D$. Все $A$ суть не $D$. Все $B$ суть $C$. Следовательно, все $B$ суть $A$.
9. $P=aab$, $Q=aabccabcdd$.
10. $P=abc$, $Q=abccdab$.
11. Смотри условия задачи.
12. $x\times(y+z)$.
13. $(A\Rightarrow(A\Rightarrow B))\Rightarrow(\neg(A\Rightarrow B)\Rightarrow\neg A)$.
14. $((Nx)\vee(Ny))\Rightarrow z$, $N(y\&x)\Rightarrow z$.
15. $\overline{A}^*\cap B^*$, $\overline{A}^*\cup C^*$, $A^*\cap(B^*\cup C^*)$.

***Вариант 24***

1. Если $A$ то $B$ либо $C$, а $B$ достаточно для $C$, но $A$ не эквивалентно $C$.
2. $(\neg A\&\neg D\vee B\&\neg D)\&(A\vee C)\vee B\&A\&\neg B\vee(A\vee\neg C)\&(\neg A\&D\vee B\&D)$.
3. $(A\equiv\neg C)\vee A\&B\&C$.
4. $A\vee B,\ A\Rightarrow C,\ B\Rightarrow D,\ D\Rightarrow C\models A\&C$.
5. Когда все $A$ суть $B$, а некоторые $B$ суть не $C$, тогда не существует $A$, таких что $B$ или $C$.
6. $\forall x\exists yP(x,y))\Rightarrow\exists y\,\forall xP(x,y)$.
7. $\boldsymbol{A}=\forall x\exists yP(x,f(x,y))\Rightarrow\forall xP(a,x)$, $\boldsymbol{B}=\forall x\exists yQ(f(x,a),y)\Rightarrow\exists y\,\forall xP(y,x)$.
8. Ни одно $C$ не есть $D$. Все $A$ суть $D$. Все $B$ суть $C$. Следовательно, некоторые $B$ есть $A$.
9. $P=ddab,\ Q=ddabbbacdd$.
10. $P=ddc,\ Q=ddcbccab$.
11. Смотри условия задачи.
12. $x\times(y+2)$.
13. $A\Rightarrow(\neg(A\Rightarrow A)\Rightarrow\neg(A\Rightarrow(A\Rightarrow A))$.
14. $(Nx)\equiv((Ny)\&z),\ (Ny)\Rightarrow((Nx)\&z)$.
15. $A^*\cap C^*,\ B^*\cup C^*,\ \overline{A}^*\cup B^*\cup C^*)$.

***Вариант 25***

1. $A$ только тогда, когда $B$, а $B$ когда $C$ или $A$, но $A$ не достаточно для $C$.
2. $(A\vee\neg D)\&(B\&\neg C\vee B\&\neg D\vee C\&B\vee\neg B\&\neg D)\&(D\vee A)\vee B\&A\&\neg B$.
3. $A\vee C\vee(A\Rightarrow B)\&\neg C$.
4. $C\Rightarrow(B\Rightarrow A),\ B\vee D,\ C\models A\vee D\vee C$.
5. Если некоторые $D$ суть $B$, а ни одно $A$ не есть не $C$, то все $A$ суть $D$.
6. $\exists x\exists y(P(x)\&P(y)\&Q(x,y))$.
7. $\boldsymbol{A}=\forall x\exists yP(x,b)\Rightarrow\forall xQ(a,f(x,y))$, $\boldsymbol{B}=\exists x\exists yP(x,y)\Rightarrow\exists y\,\forall xR(f(x,a),y,x)$.
8. Некоторые $C$ суть $D$. Все $A$ суть $D$. Все $B$ суть не $C$. Следовательно, все $B$ есть $A$.
9. $P=cda,\ Q=cdacdabcdd$.
10. $P=accb,\ Q=accbccdab$.
11. Смотри условия задачи.
12. $x!$
13. $\neg A\Rightarrow(A\Rightarrow A)$.
14. $(Nx)\vee(y\&z),\ x\Rightarrow(y\&z)$.
15. $C^*\cap\overline{A}^*,\ A^*\cup C^*\cup B^*,\ A^*\cap(B^*\cup C^*)$.

## Тесты для самоконтроля

По каждой из глав 1-4 и 6 предложены отдельные тесты для самоконтроля, а по главам 5 и 7 один тест. Каждый тест содержит 10 заданий. *Все задания имеют 5 вариантов ответов, из которых нужно выбрать только один. На листе бумаги запишите номера заданий теста и для каждого задания напишите номер выбранного Вами ответа.*

Тест по логике высказываний (тест № 1)

1. Пусть $x$, $y$ и $z$ переменные со значениями из $(-\infty,\infty)$. Указать какое из следующих выражений является высказыванием

| | | | | |
|---|---|---|---|---|
| 1) $x+y=z$ | 2) $x+y>0$ | 3) $x^2>y$ | 4) $2\times2=5$ | 5) $2+3$ |

2. Пусть $x$ и $y$ переменные со значениями из $(-\infty,\infty)$. Указать какое из следующих выражений **не** является высказыванием

| | | | | |
|---|---|---|---|---|
| 1) $2\times2=4$ | 2) $sin(x)>y$ | 3) $5>10$ | 4) $2\times2=5$ | 5) $2+3=6$ |

3. Указать какое из следующих выражений является символьной записью высказывания: «($B$ тогда, когда $A$) и (без $B$ нет и $A$)»

*1)* $(A\Rightarrow B)\&(\neg B\Rightarrow \neg A)$; *2)* $(B\Rightarrow A)\&(\neg B\Rightarrow \neg A)$; *3)* $(A\Rightarrow B)\&(\neg B\& \neg A)$;
*4)* $(B\Rightarrow A)\&(\neg B\& \neg A)$; *5)* $A\equiv B$.

4. Указать какое из следующих выражений является тавтологией (тождественно истинной)

| | | | | |
|---|---|---|---|---|
| 1) $A\&B\vee C\&\neg A$ | 2) $A\vee C\&\neg A\ \&B$ | 3) $A\&\neg A\vee C\&A$ | 4) $A\vee \neg A$ | 5) $B\&A\vee C\&\neg A$ |

5. Выражение $(A\vee B)\&C\vee A\&(B\vee C)\&B$ при $B=И$ равносильно:

| | | | | |
|---|---|---|---|---|
| *1)* $A\&B$ | *2)* $C\vee A$ | *3)* $A$ | *4)* $C$ | *5)* $C\&\neg A$ |

6. Значения $A,B,C$ и $D$ для системы $\begin{cases} (A\vee C)=Л, \\ (A\equiv(B\&D))=Л \end{cases}$ равны:

| | | | | |
|---|---|---|---|---|
| *1)* $A=Л,B=Л,$ $C=И,D=Л$ | *2)* $A=Л,B=И,$ $C=И,D=Л$ | *3)* $A=И,B=Л,$ $C=И,D=Л$ | *4)* $A=Л,B=И,$ $C=Л,D=И$ | *5)* $A=И,B=Л,$ $C=И,D=И$ |

7. Используя важнейшие пары равносильных пропозициональных форм, упростите следующую форму: $A\vee A\vee A\vee(B\Rightarrow C)\&B\&A\vee C$ и укажите, с какой из следующих форм совпадает результат.

| | | | | |
|---|---|---|---|---|
| 1) $B\&A\vee C$ | 2) $A\vee C$ | 3) $B\vee C$ | 4) $(B\Rightarrow C)\vee C$ | 5) $A\vee B$ |

8. К.н.ф. для $A\Rightarrow B\equiv C$ равна

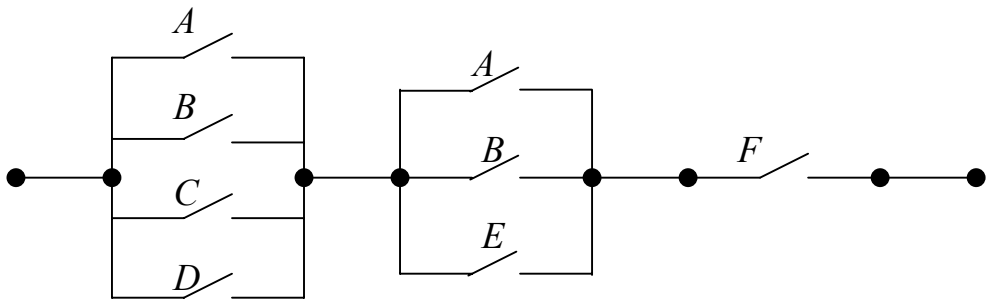
| | | | | |
|---|---|---|---|---|
| 1) $(A\vee B)\&(B\vee\neg A)\&(C\vee A\vee\neg B)$ | 2) $(B\vee A)\&(C\vee\neg A)$ | 3) $(\neg A\vee B\vee\neg C)\&(A\vee B)\&(A\vee C)$ | 4) $(A\vee\neg B\vee C)\&(B\vee C)\&A$ | 5) $(A\vee C)\&(\neg B\vee C)\&(\neg A\vee B\vee\neg C)$ |

9. С.к.н.ф. для булевой функции $f(A,B,C)$ значения которой представлены в следующей таблице

| $A$ | $B$ | $C$ | $f(A,B,C)$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

равна 1) $(\neg A\vee B\vee\neg C)\&(\neg A\vee B\vee C)\&(A\vee\neg B\vee C)\&(A\vee B\vee\neg C)\&(A\vee B\vee C)$;
2) $\neg A\vee B\vee\neg C\&\neg A\vee B\vee C\&A\vee\neg B\vee C\&A\vee B\vee\neg C\&A\vee B\vee C$;
3) $(A\vee B\vee C)\&(A\vee B\vee\neg C)\&(\neg A\vee B\vee C)$;
4) $(A\vee B\vee C)\&A\vee B\vee\neg C\&\neg A\vee B\vee C$;
5) $(\neg A\vee B\vee C)\&(A\vee\neg B\vee C)\&(A\vee B\vee\neg C)\&(A\vee B\vee C)$.

10.Контактная схема




представима в виде выражения:
1) $(A\vee B\vee C\vee D)\&(A\vee B\vee E)\&F$;
2) $(A\&B\&C\&D)\vee(A\&B\&E)\vee F$;
3) $A\vee B\vee C\vee D\&A\vee B\vee E\&F$;
4) $(A\vee B\vee C\vee D)\&(A\vee E)\&B\&F$;
5) $(A\vee B)\&(C\vee D)\&(A\vee B)\&E\&F$.

Тест по логике предикатов (тест № 2)

1. Пусть $x$, $y$ и $z$ переменные со значениями из $(-\infty,\infty)$. Указать какое из следующих выражений является двуместным предикатом

| | | | | |
|---|---|---|---|---|
| 1) $x+y=z$ | 2) $\sin(x+y)>z$ | 3) $x^2>z+y$ | 4) $2\times 2=4$ | 5) $x>y$ |

2. Пусть $x$, $y$ и $z$ переменные со значениями из $(-\infty,\infty)$. Указать какое из следующих выражений **не** является предикатом

| | | | | |
|---|---|---|---|---|
| 1) $x+y=z$ | 2) $sin(x)+y$ | 3) $x^2>y$ | 4) $2\times2=4$ | 5) $x^2<y$ |

3. Предложение «*Для каждого x выполнимо P(x), но не существует x, что Q(x)*» в символическом виде представимо в виде:

1) $(\forall xP(x))\vee\exists x\neg Q(x);$ 2) $\forall xP(x)\equiv\neg\exists xQ(x);$
3) $\forall xP(x)\equiv\exists x\neg Q(x);$ 4) $(\forall xP(x))\&\neg\exists xQ(x);$
5) $\forall x(P(x)\Rightarrow\neg\exists xQ(x)).$

4. Пусть даны предикаты на множестве натуральных чисел:

*P(x): «x простое число»,*
*D(x,y): «x делится на y».*

Предложение: «*Любое простое число не делится на 2, а также не делится на 3*» в символьной форме записывается в виде:

1) $(\forall xD(x,y))\vee\exists xP(x);$
2) $\forall x(\neg D(x,2)\&\ \neg D(x,3)\ \Rightarrow P(x));$
3) $\forall x(P(x)\Rightarrow\neg D(x,2)\vee\neg D(x,3));$
4) $\forall x(D(x,y))\Rightarrow\neg P(2)\&\ \neg P(3));$
5) $\forall x(P(x)\Rightarrow\neg D(x,2)\&\neg D(x,3)).$

5. Формула $(\exists xP(x))\&P(y)$ в интерпретации:

$M=\{...,-2,-1,0,1,2,...\}$, $P(x)$: «$x$ – простое число»

является

1) выполнимой; 2) логически общезначимой;
3) ложной; 4) противоречием;
5) истинной.

6. Формула $\neg\exists x\,\forall yA$ равносильна формуле

| | | | | |
|---|---|---|---|---|
| 1) $\exists x\,\forall y\,\neg A;$ | 2) $\forall x\exists y\,\neg A;$ | 3) $\forall x\,\forall y\,\neg A;$ | 4) $\forall x\exists y\,A;$ | 5) $\forall x\,\forall y\,A.$ |

7. Формула $\neg((\exists xA)\&\forall xD)$ равносильна формуле

1) $(\exists x\neg A)\&\forall x\neg D;$ 2) $(\forall x\neg A)\vee\exists x\neg D;$
3) $(\exists x\,A)\Rightarrow\forall x\neg D;$ 4) $(\forall x\,A)\equiv\exists x\neg D;$
2) $(\forall x\neg A)\&\exists x\,D.$

8. Предваренная нормальная форма для формулы $\forall yA(y)\Rightarrow\forall x\exists zB(x,z)$ равна

1) $\forall y\,\forall x\exists z(\neg A(y)\vee B(x,z)),$ 2) $\forall y\exists x\,\forall z(\neg A(y)\vee B(x,z)),$
3) $\exists y\,\forall x\exists z(\neg A(y)\vee B(x,z)),$ 4) $\exists y\exists x\,\forall z(\neg A(y)\vee B(x,z)),$
5) $\exists z\,\forall y\,\forall x(\neg A(y)\vee B(x,z)).$

9. Какая из следующих формул не является логически общезначимой?

1) $\exists x\,\forall y\boldsymbol{A}\Rightarrow\forall y\exists x\boldsymbol{A};$ 2) $(\boldsymbol{A}\vee\forall x\boldsymbol{B}(x))\equiv\forall x(\boldsymbol{A}\vee\boldsymbol{B}(x));$
3) $(\boldsymbol{A}\&\exists x\boldsymbol{B}(x))\equiv\exists x(\boldsymbol{A}\&\boldsymbol{B}(x));$ 4) $((\exists x\boldsymbol{B}(x))\vee\exists x\boldsymbol{C}(x))\equiv\exists x(\boldsymbol{B}(x)\vee\boldsymbol{C}(x));$
5) $\forall y\exists x\boldsymbol{A}\Rightarrow\exists x\,\forall y\boldsymbol{A}.$

10. Формула $\neg\exists x\,\forall y\exists z\,\forall u A$ равносильна формуле

1) $\exists x\,\forall y\exists z\,\forall u\neg A;$ 2) $\forall x\exists y\,\forall z\exists u\,A;$

3) $\forall x\,\forall y\,\forall z\,\forall u\neg A;$ 4) $\forall x\exists y\,\forall z\exists u\neg A;$

5) $\forall x\,\forall y\exists z\,\forall u\,A.$

Тест по логическому следствию и методу резолюций (тест № 3)

1. Произвольная формула $B$ является логическим следствием формулы $A$ тогда и только тогда, когда

1) $A\Rightarrow B$ - тавтология; 2) $A\Rightarrow B$ - выполнимая формула;

3) $A\Rightarrow B$ - противоречие; 4) $A\&B$ - тавтология;

5) $A\vee B$ - тавтология.

2. Если $C$ является логическим следствием $A$ и $B$, тогда при любых $A$, $B$ и $C$

1) $A\vee B\vee C$ является тавтологией; 2) $A\&B\Rightarrow C$ является противоречием;

3) $A\vee B\vee C$ является противоречием; 4) $A\&B\&C$ является тавтологией;

5) $A\&B\Rightarrow C$ является тавтологией;

3. Укажите, какое из следующих утверждений истинно

| 1) | 2) | 3) | 4) | 5) |
|---|---|---|---|---|
| $A\&B\models B\&\neg A;$ | $A\&B\models A\&\neg A;$ | $A\&B\models A;$ | $A\&B\models B\Rightarrow A;$ | $A\&B\models\neg B.$ |

4. Укажите, какое из следующих утверждений истинно (при произвольных формулах $A$ и $B$)

1) $A,\ A\Rightarrow B\models\neg B;$ 2) $A,\ A\Rightarrow B\models B;$ 3) $A,\ A\Rightarrow B\models\neg B\&B;$

4) $A,\ A\Rightarrow B\models\neg A;$ 5) $A,\ A\Rightarrow B\models A\&\neg A.$

5. Укажите, какое из следующих утверждений **ложно** (при произвольных формулах $A$ и $B$)

1) $A\&B\&C\models A;$ 2) $A\&B\&C\models B;$ 3) $A\&B\&C\models A\&B;$

4) $A\&B\&C\models\neg A;$ 5) $A\&B\&C\models A\&B\&C.$

6. Методом резолюций выяснить выполнимо или нет следующее множество дизъюнктов: $M=\{P\vee R\vee S,\ \neg P\vee S,\ \neg R,\ \neg S\}$. Кроме того, указать, сколько всего дизъюнктов содержится в выводе, считая и исходные дизъюнкты (при реализации метода исчерпания уровня)

1) $M$ выполнимо, вывод содержит меньше 22 дизъюнктов;

2) $M$ невыполнимо, вывод содержит меньше 22 дизъюнктов;

3) $M$ выполнимо, вывод содержит 30 дизъюнктов;

4) $M$ невыполнимо, вывод содержит 30 дизъюнктов;

5) $M$ невыполнимо, вывод содержит более 30 дизъюнктов.

7. Указать сколько и какие бинарные резольвенты можно получить из дизъюнктов $D_1=P\vee\neg T\vee S$, $D_2=\neg P\vee T$.

1) одну резольвенту: $R_1=P\vee\neg P\vee T\vee S$;
2) одну резольвенту: $R_1=\neg T\vee T\vee S$;
3) две резольвенты: $R_1=\neg T\vee T$, $R_2=\neg P\vee P$;
4) две резольвенты: $R_1=T\vee S$, $R_2=P\vee S$;
5) две резольвенты: $R_1=\neg T\vee T\vee S$, $R_2=\neg P\vee P\vee S$.

8. Для литералов множества дизъюнктов $M=\{P\vee R\vee S,\ \neg P\vee S,\ \neg R,\ \neg S\}$ ввести индексами последовательно числа 1,2,…,7. Лок-резолюцией выяснить выполнимо или нет множество дизъюнктов $M$ и сколько всего дизъюнктов содержится в выводе, считая и исходные дизъюнкты.

1) $M$ невыполнимо, в лок-выводе содержится 10 дизъюнктов;
2) $M$ выполнимо, в лок-выводе содержится 8 дизъюнктов;
3) $M$ невыполнимо, в лок-выводе содержится 7 дизъюнктов $M$;
4) $M$ выполнимо, в лок-выводе содержится 10 дизъюнктов;
5) $M$ невыполнимо, в лок-выводе содержится 17 дизъюнктов.

9. Сколемовская стандартная форма для формулы: $\exists x(A(x)\Rightarrow\forall y\exists zB(y,z,a))$ равна

1) $\exists y\,\forall y\exists z(\neg A(x)\vee B(y,z,a))$,
2) $\neg A(x)\vee B(y,z,a)$,
3) $\neg A(x)\vee B(y,f(y),a)$,
4) $\forall y\,(\neg A(b)\vee B(y,f(y),a))$,
5) $\neg A(b)\vee B(y,f(y),a)$,

10. Для силлогизма Camestres по 4-ой фигуре, который в символьной записи имеет вид

$$\forall x\,(P(x)\Rightarrow M(x)),$$
$$\forall x\,(M(x)\Rightarrow\neg S(x)),$$
$$\text{----------}$$
$$\forall x\,(S(x)\Rightarrow\neg P(x))\ .$$

укажите, какие дизъюнкты можно получить для проверки правильности силлогизма методом резолюций.

1) $\neg P(x)\vee M(x)$, $\neg M(y)\vee\neg S(y)$, $S(z)\vee\neg P(z)$;
2) $\neg P(x)\vee M(x)$, $\neg M(y)\vee\neg S(y)$, $S(a)$, $P(a)$;
3) $\neg P(x)\vee M(x)$, $\neg M(y)\vee\neg S(y)$, $S(a)\vee\neg P(a)$;
4) $\neg P(x)\vee\neg M(x)$, $\neg M(y)\vee\neg S(y)$, $\neg S(a)\vee\neg P(a)$;
5) $\neg P(x)\vee M(x)$, $\neg M(y)\vee\neg S(y)$, $S(a)\&P(a)$.

Тест по дедуктивным теориям (тест № 4)

1. Укажите, что не нужно задавать при введении исчисления высказываний

1) алфавит;
2) правила образования формул;

3) аксиомы;
4) правила доказательств;
5) правила действия с кванторами.

2. Последовательность $A_1$, $A_2$,...,$A_n$ формул считается выводом в **произвольной** формальной аксиоматической теории (в логическом исчислении) если

1) для каждого $i$ $(1 \le i \le n)$ формула $A_i$ есть либо аксиома теории, либо непосредственное следствие каких-либо предыдущих формул этой последовательности по одному из правил вывода этой теории;
2) для некоторых $i$ $(1 \le i \le n)$ формула $A_i$ есть либо аксиома теории, либо непосредственное следствие каких-либо предыдущих формул этой последовательности по одному из правил вывода этой теории;
3) формула $A_k$ получена из формул $A_{k-2}$ и $A_{k-1}$ по одному из правил вывода этой теории;
4) формула $A_k$ получена из формул $A_{k-2}$ и $A_{k-1}$ по правилу вывода *MP (modus ponens)*;
5) для каждого $i$ $(1 \le i \le n)$ формула $A_i$ есть либо аксиома теории, либо непосредственное следствие каких-либо предыдущих формул этой последовательности по правилу вывода *Gen.*

3. Дана последовательность формул исчисления высказываний:

*а*) $A \Rightarrow ((A \Rightarrow A) \Rightarrow A)$,
*б*) $(A \Rightarrow ((A \Rightarrow A) \Rightarrow A)) \Rightarrow ((A \Rightarrow (A \Rightarrow A)) \Rightarrow (A \Rightarrow A))$,
*в*) $(A \Rightarrow (A \Rightarrow A)) \Rightarrow (A \Rightarrow A)$,
*г*) $A \Rightarrow A$,
*д*) $A \Rightarrow (A \Rightarrow A)$.

Укажите, какое из следующих утверждений **истинно.**

1) последовательность формул *а*), *б*), *в*), *г*), *д*) является выводом для формулы $A \Rightarrow (A \Rightarrow A)$;
2) последовательность формул *а*), *б*), *в*), *д*), *г*) является выводом для формулы $(A \Rightarrow A)$;
3) последовательность формул *в*), *а*), *б*), *д*), *г*) является выводом для формулы $A \Rightarrow (A \Rightarrow A)$;
4) последовательность формул *а*), *б*), *в*), *г*), *д*) не содержит формул вывода ни для какой формулы;
5) последовательность формул *в*), *б*), *а*), *д*), *г*) является выводом для формулы $A \Rightarrow (A \Rightarrow A)$.

4. Пусть имеем ту же последовательность формул, что и в предыдущей задаче. Укажите, какое из следующих утверждений **ложно.**

1) последовательность формул *а*), *б*), *в*), *д*), *г*) является выводом для формулы $(A \Rightarrow A)$;

2) последовательность формул *б*), *а*), *в*), *д*), *г*) является выводом для формулы $(A \Rightarrow A)$;
3) последовательность формул *а*), *б*), *д*), *в*), *г*) является выводом для формулы $(A \Rightarrow A)$;
4) последовательность формул *а*), *б*), *в*), *г*), *д*) является выводом для формулы $(A \Rightarrow A)$;
5) последовательность формул *д*), *б*), *а*), *в*), *г*) является выводом для формулы $(A \Rightarrow A)$;

5. Пусть имеем следующие правила выводов исчисления высказываний:

*а*) $A \Rightarrow B,\ B \Rightarrow C \vdash A \Rightarrow C;$
*б*) $A \Rightarrow (B \Rightarrow C), B \vdash A \Rightarrow C;$
*в*) если $G, A \vdash B$, то $G \vdash A \Rightarrow B;$
*г*) если $G, A \vdash B$, то $G, \neg B \vdash \neg A;$
*д*) $A \& B \vdash A;$
*е*) $A, B \vdash A \& B;$
*ж*) $A \vdash A \vee B;$
*з*) $A,\ A \Rightarrow B \vdash B;$
*и*) если $A \vdash C$ и $B \vdash C$, то $A \vee B \vdash C$;
*к*) если $A \vdash B$ и $A \vdash \neg B$, то $\vdash \neg A$.

Укажите, какое из них является исходным правилом вывода, а не доказуемым и какое является теоремой дедукции.

1) правило *б)* - исходное, а *к)* - теорема дедукции;
2) правило *а)* - исходное, а *е)* - теорема дедукции;
3) правило *з)* - исходное, а *в)* - теорема дедукции;
4) правило *г)* - исходное, а *д)* - теорема дедукции;
5) правило *ж)* - исходное, а *а)* - теорема дедукции.

6. Множество теорем исчисления высказываний (теории $L$) совпадает с

1) множеством выполнимых формул теории $L$;
2) множеством тавтологий теории $L$;
3) множеством противоречий теории $L$;
4) множеством формул теории $L$, для которых существует с.к.н.ф.;
5) множеством формул теории $L$ записанных без связки $\neg$.

7. Пусть исчисление высказываний обозначена как теория $L$. Укажите, какое из следующих утверждений **ложно.**

1) теория $L$ непротиворечива, полна в широком смысле и является разрешимой теорией;
2) теория $L$ непротиворечива, полна в узком смысле и является разрешимой теорией;
3) теория $L$ непротиворечива, полна в широком и узком смыслах и, кроме того, $L$ - разрешимая теория;
4) теория $L$ противоречива, полна в широком смысле и является разрешимой теорией;

5) теория $L$ непротиворечива, полна в широком смысле, является разрешимой теорией и система её аксиом независима.

8. Укажите, чем могут отличаться различные теории первого порядка

1) логическими аксиомами;
2) исходными правилами выводов;
3) совокупностью предметных переменных;
4) собственными аксиомами;
5) наличием или отсутствием кванторов.

9. Пусть $T$ – множество теорем, а $\Phi$ множество формул дедуктивной теории, содержащей исчисление высказываний; $A$ – формула этой теории. Теория считается противоречивой, если

1) ($T=\Phi$)&($\exists A$, что доказуемы как $A$, так и $\neg A$);
2) ($T\neq\Phi$)&($\exists A$, что доказуемы как $A$, так и $\neg A$);
3) ($T\neq\Phi$)&(не существует $A$, что доказуемы как $A$, так и $\neg A$);
4) ($T=\Phi$)&(не существует $A$, что доказуемы как $A$, так и $\neg A$);
5) ($T\neq\Phi$)&(для любой $A$ доказуемы как $A$, так и $\neg A$).

10. Пусть $K_1$ исчисление предикатов первого порядка. Укажите, какое из следующих утверждений **истинно.**

1) теория $K_1$ непротиворечива, неполна в широком смысле и является разрешимой теорией;
2) теория $K_1$ непротиворечива, полна в узком смысле и является разрешимой теорией;
3) теория $K_1$ непротиворечива, полна в широком и узком смыслах и, кроме того, $K_1$ - разрешимая теория;
4) теория $K_1$ противоречива, полна в широком смысле и является разрешимой теорией;
5) теория $K_1$ непротиворечива, полна в широком смысле, не полна в узком смысле и является неразрешимой теорией.

Тест по теории алгоритмов (тест № 5)

1. Результат применения нормального алгоритма

$$\begin{cases} ab \to c \\ bb \to \bullet d \\ cc \to b \end{cases}$$ к слову $P=abcbad$ равен

| | | | | |
|---|---|---|---|---|
| 1) $da$ | 2) $dad$ | 3) $dd$ | 4) $cccd$ | 5) $ab$ |

2. Результат применения нормального алгоритма

$$\begin{cases} ab \to d \\ bc \to \bullet a \\ dd \to b \end{cases}$$ к слову $P=abdca$ равен

| | | | | |
|---|---|---|---|---|
| 1) $dad$ | 2) $da$ | 3) $dd$ | 4) $ccd$ | 5) $aa$ |

3. Результат применения машины Тьюринга $T_1$:

$q_0\, a\, S_0\, q_0$
$q_0\, S_0\, R\, q_0$
$q_0\, b\, S_0\, q_1$
$q_1\, S_0\, R\, q_1$
$q_1\, c\ \ c\ \ q_2$

к слову $P=abcc$ равен (в начальный момент читающая головка машины обозревает первую букву слова $P$)

| | | | | |
|---|---|---|---|---|
| *1) abc* | *2) cc* | *3) bc* | *4) ab* | *5) bcc* |

4. Результат применения машины Тьюринга $T_1$ (см. предыдущую задачу) к слову $P=abc$ равен (в начальный момент читающая головка машины обозревает первую букву слова $P$)

| | | | | |
|---|---|---|---|---|
| *1) abc* | *2) ab* | *3) bc* | *4) c* | *5) b* |

5. Для каждого нормального алгоритма существует вполне эквивалентный ему

1) алгоритм Тьюринга;
2) алгоритм Евклида;
3) алгоритм Квайна;
4) композиция заданного нормального алгоритма и некоторого фиксированного алгоритма Тьюринга;
5) композиция алгоритмов Тьюринга и Евклида.

6. Пусть $M$ - множество функций частично вычислимых по Маркову,
$T$ - множество функций частично вычислимых по Тьюрингу.
Какое из следующих утверждений **истинно**?

1) $(M\subset T)\&(M\neq T)$, 2) $(T\subset M)\&(T\neq M)$ 3) $T=M$,
4) $T\neq M$, 5) $T\cap M=\varnothing$.

7. Пусть $M$ - множество функций вычислимых по Маркову,
$T$ - множество функций вычислимых по Тьюрингу,
$OR$ - множество общерекурсивных функций. Какое из следующих утверждений **истинно**?

1) $(M\neq T)\&(T=OR)$, 2) $(M=T)\&(M\neq OR)$, 3) $(M\neq T)\&(M=OR)$,
4) $(M\neq T)\&(T\neq OR)$, 5) $T=M=OR$.

8. Машина Тьюринга имеет

1) (бесконечную ленту)&(конечный внешний алфавит)&(конечный внутренний алфавит);
2) (бесконечную ленту)&(бесконечный внешний алфавит)&(конечный внутренний алфавит);
3) (бесконечную ленту)&(бесконечный внешний алфавит) &(бесконечный внутренний алфавит);

4) (конечную ленту)&(бесконечный внешний алфавит)&(конечный внутренний алфавит);
5) (конечную ленту)&(конечный внешний алфавит)&(бесконечный внутренний алфавит).

9. Арифметическая функция $f(x,y) = x+y$
1) (не вычислима по Тьюрингу)&(вычислима по Маркову)&(является общерекурсивной);
2) (вычислима по Тьюрингу)&(вычислима по Маркову)&(является общерекурсивной);
3) (не вычислима по Тьюрингу)&(не вычислима по Маркову) &(является общерекурсивной);
4) (не вычислима по Тьюрингу)&(не вычислима по Маркову) &(не является общерекурсивной);
5) (вычислима по Тьюрингу)&(вычислима по Маркову)&(не является общерекурсивной).

10. Укажите, какая из перечисленных ниже проблем является алгоритмически разрешимой
1) проблема диофантовых корней,
2) проблема эквивалентности слов,
3) проблема остановки,
4) проблема разрешимости логики предикатов,
5) проблема нахождения решения задачи коммивояжёра.

Тест по неклассическим логикам и сложности вычислений (тест № 6)

1. Конъюнкция и дизъюнкция в трехзначной логике Лукасевича, вводятся следующим образом:
1) $x\&y=max(x,y)$, $x\vee y=min(x,y)$;
2) $x\&y= min(x,y)$, $x\vee y= max(x,y)$;
3) $x\&y= x\times y(mod\ 3)$, $x\vee y= x+y(mod\ 3)$;
4) $x\&y=(x\vee y)+1(mod\ 3)$, $x\vee y= max(x,y)$;
5) $x\&y= min(1, max(x,y))$, $x\vee y= max(1, min(x,y))$.

2. Число различных функций $k$ – значной логики, зависящих от $n$ переменных равно

| | | | | |
|---|---|---|---|---|
| 1) $n\times k$ | 2) $n^k$ | 3) $k^n$ | 4) $k^{k^n}$ | 5) $n^{n^k}$ |

3. Рассмотрим $k$ значную $(k>2)$ логику Поста, где циклическое отрицание введено как $\neg x=x+1\ (mod\ k)$, а отрицание Лукасевича как $Nx=k-1-x$. Укажите, какое утверждение истинно
1) $N(Nx) = x$ и $\neg(\neg x) = x$;
2) $N(Nx) \neq x$ и $\neg(\neg x) = x$;
3) $N(Nx) = x$ и $\neg(\neg x) \neq x$;

4) $N(Nx) \neq x$ и $\urcorner(\urcorner x) \neq x$;
5) $N(Nx) \neq \urcorner(\urcorner x) \neq x$.

4. Рассмотрим *k* значную логику Поста, где переменные принимают значения *0, 1, ...k-1*. Импликация в этой логике вводится следующим образом:

$$x \Rightarrow y = \begin{cases} k-1, & \text{если } 0 \le x < y \le k-1, \\ (k-1)-x+y, & \text{если } 0 \le y \le x \le m-1. \end{cases}$$

Пусть *k=3*. Обозначим значения *0, 1*, и *2* через *0, ½* и *1* соответственно. Укажите, какое из следующих утверждений истинно.

1) эта импликация совпадает с дизъюнкцией логики Рейхенбаха;
2) эта импликация совпадает с импликацией логики Бочвара;
3) эта импликация совпадает с импликацией логики Клини;
4) эта импликация совпадает с импликацией логики Гейтинга;
5) эта импликация совпадает с импликацией логики Лукасевича.

5. Пусть задана лингвистическая переменная, описываемая набором:

*(X, T(X), U, G , M)*

в котором:

*X* - название лингвистической переменной,

*T(X)* - множество лингвистических значений переменной *X*,

*U* - универсальное множество,

*G* - синтаксические правила, порождающие названия переменной, т.е. правила определения синтаксических значений,

*M* - семантические правила, которые ставят в соответствие каждой нечеткой переменной ее смысл *M(X)*, т.е. характеристическую функцию для *X*.

Укажите, какое из следующих утверждений истинно

1) *U* – нечеткое множество, а *T(X)* – обычное множество;
2) *U* – обычное множество, а *T(X)* – нечеткое множество;
3) *U* – нечеткое множество и *T(X)* – нечеткое множество;
4) *U* и *T(X)* – обычные множества;
5) *U* – обычное конечное множество, а *T(X)* – обычное бесконечное множество.

6. Пусть*]x[* обозначает наименьшее целое *q*, такое, что $q \ge x$. Укажите, каково минимальное число символов нужно для представления числа *n*, заданного в десятичной системе счисления

| | | | | |
|---|---|---|---|---|
| 1) $n$ | 2) $ln(n)$ | 3) $]log(n)[$ | 4) $ln(log(n))$ | 5) $]ln(n)[$ |

7. Укажите, какой наименьший порядок (из записанных) имеет размер представления в ЭВМ графа с *n* вершинами и *m* ребрами

| | | | | |
|---|---|---|---|---|
| 1) $0(n \times m)$ | 2) $0(ln(n))$ | 3) $0(m^n)$ | 4) $0(m \times log(n))$ | 5) $0(n^m)$ |

8. Рассмотрим задачу о минимальном соединении. Дано *n* городов. Нужно соединить все города телефонной связью так, чтобы общая длина телефонных линий была минимальной. На языке теории графов эта задача формулируется следующим образом. Дан полный граф с *n* вершинами и известны длина каждого ребра. Требуется найти остовный подграф (связный подграф без циклов, содержащий все вершины исходного графа) имеющий минимальную длину, т.е. имеющий минимальную сумму длин ребер.

Эту задачу можно решить, перебирая все остовные подграфы данного полного графа, и выбирая тот остовный подграф который имеет минимальную длину. Известно, что число всех остовных подграфов полного графа равно $n^{(n-2)}$. Кроме алгоритма перебора всех остовных подграфов данного графа, указанную задачу можно решить, так называемым жадным алгоритмом, число шагов которого есть *0(n log(n))*.

Укажите, какое из следующих утверждений истинно

1) задача о минимальном соединении имеет экспоненциальную временную сложность;
2) задача о минимальном соединении имеет полиномиальную временную сложность;
3) алгоритм перебора всех остовных подграфов данного графа имеет полиномиальную временную сложность;
4) жадный алгоритм имеет линейную временную сложность;
5) жадный алгоритм имеет экспоненциальную временную сложность.

9. Проблема выяснения выполнимости произвольной формулы *A* логики высказываний (пропозициональной формы), представленной в конъюнктивной нормальной форме,

1) является алгоритмически неразрешимой;
2) не является *NP* – полной задачей;
3) является *NP* - полной задачей;
4) является задачей, не принадлежащей классу *NP*;
5) является задачей, не имеющей решения.

10. Укажите, какое из следующих утверждений **ложно**

1) задача является *NP* полной, если она входит в *NP* и каждая задача из *NP* полиномиально сводится к ней;
2) *NP* класс задач содержит задачи которые можно решить недетерминированными алгоритмами, работающими в течении полиномиального времени;
3) задача является *NP* трудной, если каждая задача из *NP* полиномиально сводится к ней;
4) если одновременно задача $Z_1$ полиномиально сводится к задаче $Z_2$ и $Z_2$ полиномиально сводится к задаче $Z_1$, то задачи $Z_1$ и $Z_2$ полиномиально эквивалентны;

5) задача имеет экспоненциальную временную сложность, если хотя бы один из алгоритмов её решения имеет экспоненциальную временную сложность.

## Ответы к тестам самоконтроля

| № теста | № вопроса теста | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| | № ответа на данный вопрос теста | | | | | | | | | |
| 1 | 4 | 2 | 1 | 4 | 2 | 4 | 2 | 5 | 3 | 1 |
| 2 | 5 | 2 | 4 | 5 | 1 | 2 | 2 | 3 | 5 | 4 |
| 3 | 1 | 5 | 3 | 2 | 4 | 2 | 5 | 3 | 4 | 2 |
| 4 | 5 | 1 | 2 | 4 | 3 | 2 | 4 | 4 | 1 | 5 |
| 5 | 2 | 5 | 2 | 4 | 1 | 3 | 5 | 1 | 2 | 5 |
| 6 | 2 | 4 | 3 | 5 | 4 | 3 | 4 | 2 | 3 | 5 |