Математика и информатика

Романова Наталья Юрьевна,
каф. ИТОиМ, 217-17-68,
гл. корпус, 3-49, 3-54

Uchebka_14@mail.ru

Аксиоматический метод

© Составитель: доцент кафедры ИТОиМ КГПУ им. В.П. Астафьева
Романова Н.Ю.

Аксиоматический метод построения научной теории **заключается в следующем:**

- выделяются основные понятия,

- формулируются аксиомы теории,

- все остальные утверждения выводятся логическим путём, опираясь на них.

- ***аксиомы*** **– утверждения, не требующие доказательства**
- ***основные понятия*** **- это элементарные понятия, которые нельзя определить через другие.**

Наиболее убедительным примером применения аксиоматического метода явился математический трактат "Начала" древнегреческого математика Евклида (ок. 300 г. до н.э.).

Суть *логических рассуждений* представляет собой цепочку утверждений, каждое из которых либо является *исходной посылкой* (*постулатом*, *аксиомой*, *гипотезой*), либо получается из предыдущих утверждений с помощью определённых правил - правил логического вывода:

исходная посылка $\xrightarrow{\text{правила логики}}$ вывод

Какими же правилами *вывода* пользуются люди в логически правильных рассуждениях? Сформулируем лишь некоторые наиболее простые из них, которыми мы пользуемся постоянно и зачастую неосознанно.

- **Правило** ***индукции*** **- переход от частного к общему.**
- **Правило** ***дедукции*** **- переход от общего к частному.**
- **Правило** ***отделения*** **(modus ponens):**

"Если истинно утверждение x и истинно, что из x следует y, то истинно и утверждение y".

- **Правило** ***силлогизма*** **(barbara):**

"Если истинно утверждение, что из x следует y, и истинно утверждение, что из y следует z, то истинно и утверждение, что из x следует z"

- **Правило** ***эквивалентной замены:***

"Если утверждение x истинно и в него входит утверждение y, о котором известно, что оно эквивалентно другому утверждению z, то истинно и утверждение, полученное из x заменой любых вхождений y на z". Это правило аналогично часто используемому в математике правилу замены "на равное"

И т. д

Формализация математических теорий.

Для формализации математической теории нужно выполнить следующие шаги:

1) Ввести *символы*, которые необходимы для записи предложений теории.

2) С помощью этих символов графически изобразить предложения теории в виде строк символов - *формул*. При этом нужно выделить некоторые из этих формул в качестве *аксиом* теории.

3) Следующий шаг более трудный - описать те *средства логики* (правила вывода), которые применяются для получения теорем.

Решающего успеха в деле формализации логики добился в 1847 году английский математик *Джордж Буль* (1815-1864), построив *алгебру логики*, названную в его честь *булевой*. Он превратил математическую логику в *алгебру высказываний*.

Булева алгебра - наука о действиях над

сказываниями (суждениями).

произвел революцию в науке, о которой сам не
дозревал. То, во что он превратил логику, было в
льнейшем положено в основу построения
ектронно-вычислительных устройств. Из всей
тики именно Булева алгебра получила самое
ольшое практическое применение в технике.

Алгебра высказываний

- *Алгебра высказываний* **или** *булева алгебра* **рассматривает способ образования одних высказываний из других, более простых, с помощью так называемых** *логических операций*. **Поскольку высказывания рассматриваются только с точки зрения их логического значения (*И-истина* или *Л-ложь*), то и логические операции рассматриваются как средство вычисления логического значения сложного высказывания по логическим значениям составляющих его простых высказываний.**

Высказыванием (суждением) *называется предложение, которое является или может оказаться либо **истинным**, либо **ложным***

Высказывания А, В, С и т. д. Могут принимать только 2 значения:

И или Л

Например:

А = Л, В = И

Пусть *A* и *B* обозначают некоторые произвольные высказывания. Обычно в логике рассматривают следующие логические операции над этими высказываниями:

Операция	**Название операции**	**Краткое прочтение полученного высказывания**	**Полное прочтение полученного высказывания**
$\neg A$ *или* $\overline{A}$	***отрицание***	не A	неверно, что A.
$A \wedge B$ (*или* $A\&B$)	***конъюнкция***	A и B	верно, что A, и верно, что B.
$A \vee B$	***дизъюнкция***	A или B	верно, что A, или верно, что B.
$A \oplus B$	***исключающая дизъюнкция***	A или B (но не A и B)	верно, что A, или верно, что B. (но не одновременно)
$A \leftrightarrow B$	***эквивалентность***	A эквивалентно B	верно, что A, тогда и только тогда, когда верно, что B.
$A \rightarrow B$	***Импликация***	A влечёт B (*A - условие*, *B-следствие)*	если верно, что A, то верно, что B.

Можно попытаться прочесть полученные высказывания, когда *A* - это высказывание *"Шумел камыш"*, а *B* - высказывание *"Деревья гнулись"*.

- **Таблица, которая определяет, какие значения принимают высказывания, полученные с помощью этих логических операций, если исходные высказывания принимают значения *И* или *Л*, называется *таблицей истинности*:**

Так выглядит ***таблица истинности*** для элементарных логических операций:

A	B	$\neg A$	$A \wedge B$	$A \vee B$	$A \leftrightarrow B$	$A \rightarrow B$
И	*И*	*Л*	*И*	*И*	*И*	*И*
И	*Л*		*Л*	*И*	*Л*	*Л*
Л	*И*	*И*	*Л*	*И*	*Л*	*И*
Л	*Л*		*Л*	*Л*	*И*	*И*

- Сложные формулы состоят из нескольких логических операций (например, $(A \vee B) \wedge B \rightarrow C$). Все операции выполняются в порядке возрастания "старшинства " :

¬, Λ , V, →, ↔

Этот порядок может быть изменен с помощью скобок

- Когда переменные принимают различные значения *И* и *Л*, формула тоже принимает какие-то логические значения, определяемые элементарными логическими операциями.

С помощью таблицы истинности проверим закон «отрицания отрицания»:

A	¬A	¬¬A	¬¬A↔A
И	Л	И	И
Л	И	Л	И

Логические операторы в запросах поисковым системам:

- Логическое И - оператор **AND**
- Логическое ИЛИ - оператор **OR**
- Логическое НЕ - оператор **NOT**
- Логическое «БЛИЗКО» - оператор **NEAR**

Из курса лекций по предмету «Информационная культура»

Операторы в поисковых системах

Оператор	Rambler	Яndex	Aport
И	**Пробел** **AND** **&**	**Пробел**; **&** - в пределах предложения, **&&** - в пределах документа	**Пробел** **AND** **&** **+** **И**
ИЛИ	**OR** **|**	**|**	**OR** **|** **ИЛИ**
НЕ	**NOT**	**~**	**NOT** **-** **НЕ**
Группировка	**( )**	**( )**	**( )**

Используя обозначения для логических операций можно записать ***некоторые законы логики***, например:

- **¬(*A* ∧ ¬*A*)** - *закон противоречия* ("Никакое высказывание не может быть одновременно и истинным, и ложным");
- **(*A*→*B*) ↔(¬*B*→¬*A*)** - *закон контрапозиции* (используется для доказательства "от противного");

Свойства операций:

- **¬¬*A*↔*A*** - *закон двойного отрицания* («отрицание отрицания");
- **¬(*A* ∧ *B*)↔(¬*A* V ¬*B*)**, **¬(*A* V *B*)↔(¬*A* ∧ ¬*B*)** - *законы Де Моргана*;
- *коммутативность*; *ассоциативность*; *дистрибутивность* конъюнкции и дизъюнкции;

***Замена операций*:**

- **(*A*↔*B*) ↔(*A*→*B*)∧(*B*→*A*)**;
- **(*A*→*B*) ↔(¬*A* V *B*)**;
- **(*A* ∧ *B*) ↔¬(¬*A* V ¬*B*)**;
- **(*A* V *B*) ↔¬(¬*A* ∧ ¬*B*)** - (по де Моргану)

Переместительный $x \vee y = y \vee x$ $x \wedge y = y \wedge x$	**Поглощения** $x \vee (x \wedge y) = x$ $x \wedge (x \vee y) = x$
Сочетательный $x \vee (y \vee z) = (x \vee y) \vee z$ $x \wedge (y \wedge z) = (x \wedge y) \wedge z$	**Склеивания** $(x \vee y) \wedge (\neg x \vee y) = y$ $(x \wedge y) \vee (\neg x \wedge y) = y$
Распределительный $x \wedge (y \vee z) = x \wedge y \vee y \wedge z$ $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$	**С инверсией** $x \vee \neg x =$ И(1) $x \wedge \neg x =$ Л(0)
Де Моргана $\neg(x \vee y) = \neg x \wedge \neg y$ $\neg(x \wedge y) = \neg x \vee \neg y$	**С константой** $x \vee$ Л (0)= x $x \wedge$ Л (0) = Л (0) $x \vee$ И (1) = И (1) $x \wedge$ И (1) = x
Идемпотенции $x \vee x = x$ $x \wedge x = x$	**Отрицания отрицания** $\neg \neg x = x$

- Э.Шредер (немецкий математик) первым предложил арифметизировать алгебру логики: заменить «И» на 1, «Л» - на 0, т. е. практически перейти к двоичной системе счисления.

Таблица истинности и логические действия в *арифметическом виде* (*Эрнст* ***Шрёдер*** *«исчисление высказываний»):*

A	B	$\neg A$	$A \wedge B$	$A \vee B$	$A \rightarrow B$	$A \leftrightarrow B$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

$\neg$	*Отрицание*	$1 - A$
$\wedge$	*Конъюнкция*	$A \times B$ или AB
$\vee$	*Дизъюнкция*	$A + B - A \times B$ или $A + B - AB$
$\rightarrow$	*Импликация*	$1 - A + A \times B$ или $1 - A + AB$
$\leftrightarrow$	*Эквивалентность*	$1 - (A - B) \times (A - B)$ или $1-(A-B)(A-B)$

Построение логической функции по ее таблице истинности
(На примере сложения двоичных чисел)

- Пусть нам необходимо сложить двоичные числа X и Y. Через P и Z обозначим первую и вторую цифру суммы: X + Y = PZ. Таблица истинности, определяющая результат сложения, имеет вид:

X	Y	P	Z
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

К Схеме одноразрядного сумматора

Сконструируем функции P(X,Y) и Z(X,Y) по этой таблице:

$$P(X,Y)=X \wedge Y; \quad Z(X,Y)=(X \vee Y) \wedge \neg(X \wedge Y).$$

Логические элементы компьютера

Логический элемент компьютера — это часть электронной логической схемы, которая реализует элементарную логическую функцию.

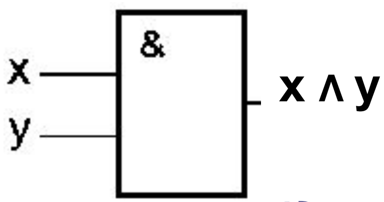
Логическими элементами компьютеров являются электронные схемы *И, ИЛИ, НЕ, И—НЕ, ИЛИ—НЕ* и другие (называемые также вентилями), а также *триггер*.

Триггер **— это электронная схема, широко применяемая в регистрах компьютера для надёжного запоминания одного разряда двоичного кода. Триггер имеет два устойчивых состояния, одно из которых соответствует двоичной единице, а другое — двоичному нулю.**

Каждый логический элемент имеет свое условное обозначение, которое выражает его логическую функцию, но не указывает на то, какая именно электронная схема в нем реализована

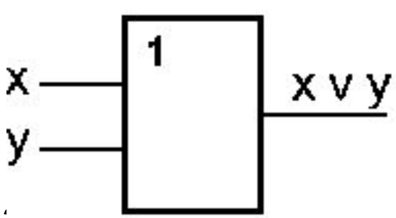
С х е м а   И (конъюнктор)

Схема И реализует конъюнкцию двух или более логических значений.



С х е м а   ИЛИ (дизъюнктор)

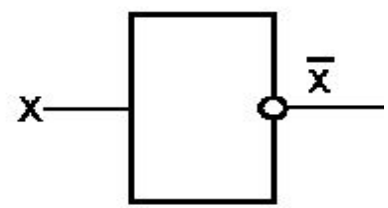
Схема ИЛИ реализует дизъюнкцию двух или более логических значений.



Знак "1" на схеме — от устаревшего обозначения дизъюнкции как **">=1"**

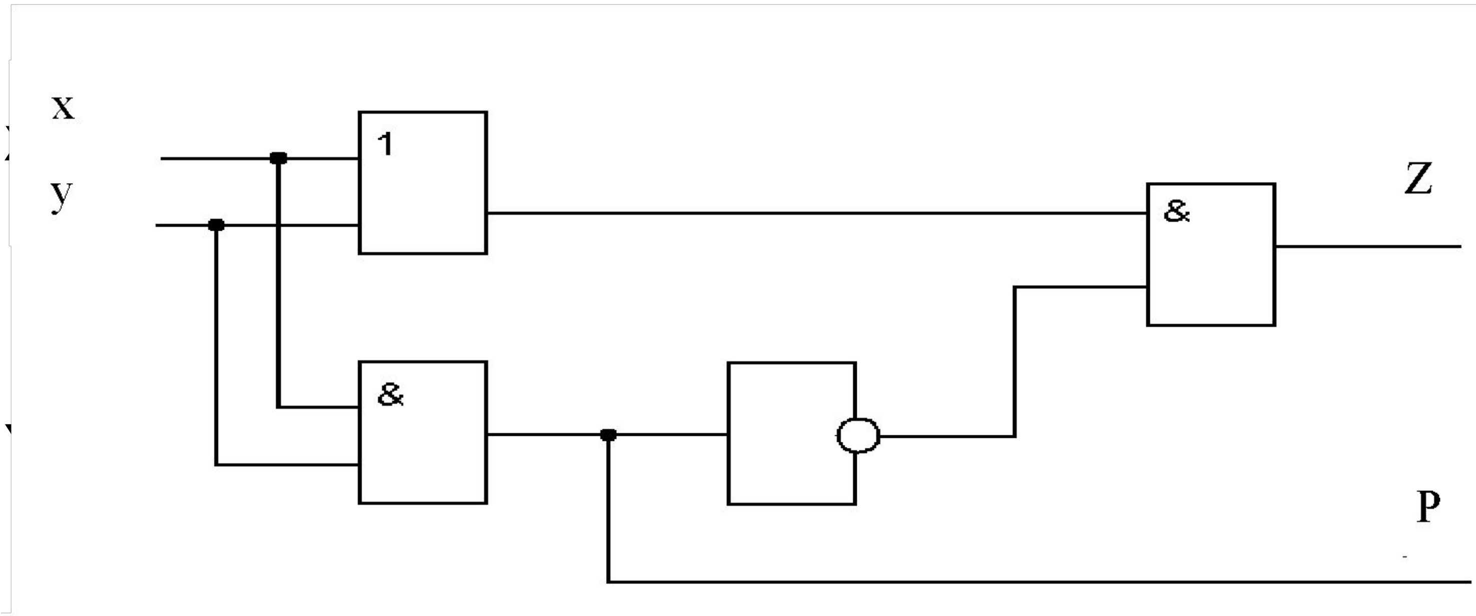
С х е м а   НЕ (инвертор)

Схема **НЕ** реализует операцию отрицания.



С помощью вентилей можно реализовать ***любую логическую функцию***, описывающую работу устройств компьютера – построить *функциональную схему*. Обычно у вентилей бывает от двух до восьми входов и один или два выхода.

С помощью вентилей можно построить функциональную схему одноразрядного сумматора (см. логическую функцию сложения двоичного числа):

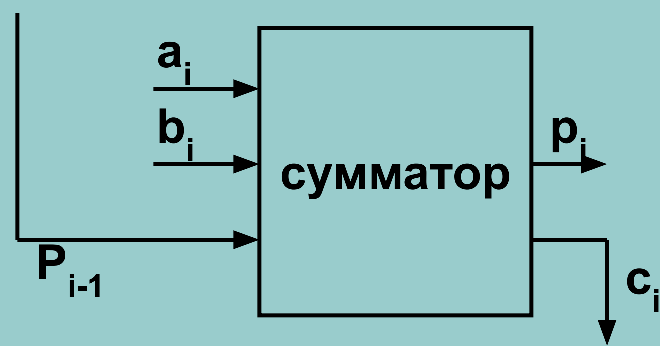


$$P(X,Y)=X \wedge Y;$$

$$Z(X,Y)=(X \vee Y) \wedge \neg(X \wedge Y).$$

Сумматор

Сумматор — это электронная логическая схема, выполняющая суммирование двоичных чисел:



При сложении чисел А и В в одном *i*-ом разряде приходится иметь дело с тремя цифрами:

1. цифра a_i первого слагаемого;

2. цифра b_i второго слагаемого;

3. перенос p_{i-1} из младшего разряда.

В результате сложения получаются две цифры:

1. цифра c_i для суммы;

2. перенос p_i из данного разряда в старший.

Многоразрядный двоичный сумматор, предназначенный для сложения многоразрядных двоичных чисел, представляет собой комбинацию *одноразрядных сумматоров*.

- **ВСЕ ОСТАЛЬНЫЕ ОПЕРАЦИИ, ПРОИЗВОДИМЫЕ ЭВМ, СВОДЯТСЯ К БОЛЬШОМУ ЧИСЛУ ПРОСТЕЙШИХ АРИФМЕТИЧЕСКИХ И ЛОГИЧЕСКИХ ОПЕРАЦИЙ, аналогично тому, как операцию умножения можно свести к большому числу операций сложения.**

Сумматор служит, прежде всего, центральным узлом ***арифметико-логического устройства***.

В современных ЭВМ арифметико-логическое устройство объединяется с управляющими устройствами в единую схему - ***процессор***.

Процессор- центральная микросхема ЭВМ, осуществляющая операции по обработке информации и управляющая работой остальных устройств.

АЛУ + УУ = процессор – основная микросхема компьютера.