Министерство образования и науки Российской Федерации
"ТАМБОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ"
ФГБОУ ВПО "ТГТУ"

ВАСИЛЬЕВ В.В., ЩЕРБАКОВА А.В.

ОСНОВЫ ДИСКРЕТНОЙ МАТЕМАТИКИ

Утверждено Методическим советом ТГТУ
в качестве методических указаний для студентов,
обучающихся по инженерным направлениям подготовки

Тамбов 2015

Рецензент
к.ф.-м.н., доцент А.Д. Нахман

Утверждено Методическим советом ТГТУ
(протокол № 1 от 20.01.2015 г.)

ОГЛАВЛЕНИЕ

МАТЕМАТИЧЕСКАЯ ЛОГИКА

§1. Полукольцо. Булева алгебра

Полукольцом $\mathcal{S} = (S, +, \cdot, 0, 1)$ называется множество S с заданными на нем операциями $+$ и $\cdot$, удовлетворяющими условиям:

1. $a + (b + c) = (a + b) + c$;
2. $a + b = b + a$;
3. $a + 0 = a$;
4. $a \cdot (b \cdot c) = (a \cdot b) \cdot c$;
4. $a \cdot 1 = 1 \cdot a = a$;
5. $a \cdot (b + c) = a \cdot b + a \cdot c$;
6. $(b + c) \cdot a = b \cdot a + c \cdot a$;
8. $a \cdot 0 = 0 \cdot a = a$.

Полукольцо называется *идемпотентным*, если $a + a = a$.

Полукольцо называют *симметричным*, если

1. $a \cdot a = a$;
2. $a \cdot b = b \cdot a$;
3. $a + (b \cdot c) = (a + b) \cdot (a + c)$;
4. $1 + a = 1$.

Примеры.

1. $\mathcal{R}^+ = (\mathbb{R} \cup \{+\infty\}, \min, +, \infty, 0)$ – полукольцо;
2. Пусть операции $+$ и $\cdot$ определены таблицами Кэли

$+$	0	1
0	0	1
1	1	1

$\cdot$	0	1
0	0	0
1	0	1

тогда $\mathcal{B} = (\{0, 1\}, +, \cdot, 0, 1)$ – симметричное полукольцо.

Перечислим аксиомы симметричного полукольца (двойственные аксиомы объединены).

1. $a + (b + c) = (a + b) + c$;
2. $a + b = b + a$;
3. $a + a = a$;
4. $a + 0 = a$;
5. $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$;
6. $a \cdot 0 = 0$;
7. $a \cdot (b \cdot c) = (a \cdot b) \cdot c$;
8. $a \cdot b = b \cdot a$;
9. $a \cdot a = a$;
10. $a \cdot 1 = a$;
11. $a + (b \cdot c) = (a + b) \cdot (a + c)$;
12. $a + 1 = 1$.

Таким образом, для симметричных полуколец справедлив *принцип двойственности*: любое тождество в симметричном полукольце остается верным, если сложение заменить умножением, а единицу – нулем и наоборот.

В идемпотентном полукольце *естественное отношение порядка* определяется следующим образом: $x \leqslant y \Leftrightarrow x + y = y$.

Свойства симметричного полукольца.

1. Тождества поглощения:

$$x(x + y) = x, \qquad x + xy = x.$$

Докажем первое тождество.

$$x(x+y) = xx + xy = x + xy = x(1+y) = x \cdot 1 = x.$$

2. В симметричном полукольце

$$x \leqslant y \Leftrightarrow x \cdot y = y.$$

Доказательство. Необходимость. Пусть $x \leqslant y$, тогда $xy = x(x+y) = xx + xy = x(1+y) = x \cdot 1 = x$.

Достаточность. Пусть $x = xy$, тогда $x + y = xy + y = y(x+1) = y \cdot 1 = y$.

3. Для любого x из симметричного полукольца верны неравенства $0 \leqslant x \leqslant 1$.

Доказательство. Из соотношений $0 + x = x$ и $x + 1 = 1$ по определению порядка следуют соотношения $0 \leqslant x$ и $x \leqslant 1$ соответственно.

Булева алгебра – симметричное полукольцо, в котором для любого x существует $\bar{x}$, называемый дополнением x такой, что

$$x + \bar{x} = 1, \quad x \cdot \bar{x} = 0.$$

Обычно сложение обозначают $\vee$, а умножение $\wedge$.

Перечислим аксиомы булевой алгебры.

1. $a \vee (b \vee c) = (a \vee b) \vee c$;	8. $a \wedge (b \wedge c) = (a \wedge b) \wedge c$;
2. $a \vee b = b \vee a$;	9. $a \wedge b = b \wedge a$;
3. $a \vee a = a$;	10. $a \wedge a = a$;
4. $a \vee 0 = a$;	11. $a \wedge 1 = a$;
5. $a \wedge (b \vee c) = (a \wedge b) \vee (a\wedge)$;	12. $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$;
6. $a \wedge 0 = 0$;	13. $a \vee 1 = 1$;
7. $a \vee \bar{a} = 1$;	14. $a \wedge \bar{a} = 0$.

Свойства булевой алгебры.

1. Для любого x существует единственный $\bar{x}$.

Доказательство. Допустим противное, т.е. для x найдется a такое, что $a \wedge x = 0$ и $a \vee x = 1$. Тогда из соотношений $a = a \vee 0 = a \vee (x \wedge x) = (a \vee x) \wedge (a \vee \bar{x}) = 1 \wedge (a \vee \bar{x}) = (x \vee \bar{x}) \wedge (a \vee \bar{x}) = \bar{x} \vee (a \wedge x) = \bar{x} \vee 0 = \bar{x}$ следует, что $a = \bar{x}$.

2. Для любого x выполняется $\bar{\bar{x}} = x$.

Доказательство. Из соотношений $\bar{\bar{x}} \wedge \bar{x} = 0$, $\bar{\bar{x}} \vee \bar{x} = 1$, $\bar{x} \wedge x = 0$, $\bar{x} \vee x = 1$ в силу единственности дополнения следует $\bar{\bar{x}} = x$.

3. Законы де Моргана:

$$\overline{x \vee y} = \bar{x} \wedge \bar{y}, \quad \overline{x \wedge y} = \bar{x} \vee \bar{y}.$$

Для доказательства первого тождества достаточно показать, что $(\bar{x} \wedge \bar{y}) \vee (x \vee y) = 0$, $(\bar{x} \wedge \bar{y}) \vee (x \vee y) = 1$.

$(\bar{x} \wedge \bar{y}) \vee (x \vee y) = (\bar{x} \vee x \vee y) \wedge (\bar{y} \vee x \vee y) = (1 \vee y) \wedge (1 \vee x) = 1 \wedge 1 = 1$.

Пример. 1. Полукольцо $\mathcal{B}$ – булева алгебра.

2. На множестве $\{0,1\}^n$ определим структуру булевой алгебры, положив для $\tilde{\alpha} = \{\alpha_1, \ldots, \alpha_n\}$, $\tilde{\beta} = \{\beta_1, \ldots, \beta_n\}$

$$\tilde{\alpha} \vee \tilde{\beta} = \{\alpha_1 \vee \beta_1, \ldots, \alpha_1 \vee \beta_1\}, \quad \tilde{\alpha} \wedge \tilde{\beta} = \{\alpha_1 \wedge \beta_1, \ldots, \alpha_1 \wedge \beta_1\}$$
$$\bar{\tilde{\alpha}} = \{\bar{\alpha}_1, \ldots, \vec{\alpha}_n\}, \quad \bar{0} = \{0, \ldots, 0\}, \quad \bar{1} = \{1, \ldots, 1\}.$$

Носитель этой булевой алгебры называется *булевым кубом*. Обозначаем эту алгебру $\mathcal{B}^n$. Отношение естественного порядка на алгебре $\mathcal{B}^n$ определяется так: $\tilde{\alpha} \leqslant \tilde{\beta} \Leftrightarrow \alpha_i \leqslant \beta_i$, $i = \overline{1,n}$, то есть $\alpha_i \vee \beta_i = \beta_i$.

3. Пусть A некоторое множество, обозначим 2^A множество его подмножеств. Тогда $S_A = \{2^A, \cup, \cap, \varnothing, A\}$ – булева алгебра.

§2. Булева функция.

Булева функция – отображение вида $f : \{0;1\}^n \to \{0;1\}$.

Булевы константы – 0 и 1. Булевы переменные имеют область значений $\{0;1\}$. Область определения булевой функции – булев куб. Число всех его элементов равно 2^n.

Булева функция задается таблицей. При $n = 1$ имеется 4 функции.

x	f_1	f_2	f_3	f_4
0	0	0	1	1
1	1	0	1	0

Функция f_1 называется *тождественной*, а функция f_4 – *отрицанием*.

При $n = 2$ имеется всего 16 функций, перечислим основные.

x_1	x_2	f_1	f_2	f_3	f_4	f_5	f_6	f_7
0	0	0	0	0	1	1	1	1
0	1	1	0	1	1	0	1	0
1	0	1	0	1	0	0	1	0
1	1	1	1	0	1	1	0	0

Для функций из этой таблицы используют следующие обозначения:

$$f_1(x_1, x_2) = x_1 \vee x_2; \qquad f_5(x_1, x_2) = x_1 \leftrightarrow x_2;$$
$$f_2(x_1, x_2) = x_1 \wedge x_2; \qquad f_6(x_1, x_2) = x_1 | x_2;$$
$$f_3(x_1, x_2) = x_1 \oplus x_2; \qquad f_7(x_1, x_2) = x_1 \downarrow x_2$$
$$f_4(x_1, x_2) = x_1 \to x_2;$$

и названия: f_1 – дизъюнкция, f_2 – конъюнкция, f_3 – сумма по модулю 2, f_4 – импликация, f_5 – эквивалентность, f_6 – штрих Шеффера, f_7 – стрелка Пирса. Пользуясь определениями функций легко убедиться в справедливости следующих равенств $x_1|x_2 = \overline{x_1 \wedge x_2}$, $x_1 \wedge x_2 = \overline{x_1 \vee x_2}$.

Пример.

	x_1	x_2	x_3	f
1	0	0	0	0
2	0	0	0	0
3	0	1	0	0
4	0	1	1	1
5	1	0	0	0
6	1	0	1	1
7	1	1	0	1
8	1	1	1	1

Приведенная здесь функция называется *мажоритарной*. В другой форме эту функцию можно записать так: $f = (0,0,0,1,0,1,1,1)$, выписав столбец f, или $f = \{4,6,7,8\}$, указав номера наборов, на которых принимается значение 1.

Булевы функции равны, если они совпадают как отображения булева куба. Проблема в том, чтобы определить равенство булевых функций независимо от числа переменных.

Пример. Рассмотрим $f(x,y) = x \vee y$ и $g(x,y,z) = xz \vee x\bar{z} \vee yz \vee y\bar{z}$. Преобразовав, получим $g(x,y,z) = x \vee y$.

Обобщая ситуацию, можно ввести понятие фиктивной переменной.

Переменную x_i называют *фиктивной* переменной функции $f(x_1,\ldots,x_n)$, если ее значения не зависят от x_i.

Переменную, не являющуюся фиктивной, называют существенной.

Булевы функции f и g называют *равными*, если на каждом наборе существенных переменных их значения совпадают.

Фиктивные переменные можно удалять и вводить следующим образом

$$\hat{f}(x_1,\ldots,x_n,y) = f(x_1,\ldots,x_n)(y \vee \bar{y}).$$

§3. Формулы и суперпозиции.

Определение формулы основано на понятии суперпозиции.

Пусть булева функция f есть функция от n переменных, а булевы функции $g_1,\ldots,g_n$ – произвольные функции от m переменных.

Определим функцию $f(g_1,\ldots,g_n)$, называемую *суперпозицией* функций $f, g_1,\ldots,g_n$ так, что для любого $\tilde{\alpha} \in \mathcal{B}^m$ имеет место равенство

$$f(g_1,\ldots,g_n)(\tilde{\alpha}) = f(g_1(\tilde{\alpha}),\ldots,g_n(\tilde{\alpha})).$$

Пусть дано ммножество булевых функций F. Тогда формулой над множеством F считаем любую константу и любую булеву переменную. Далее, если известно, что $\Phi_1,\ldots,\Phi_n$ – формулы над F, а f – функция из F от n переменных, то $f(\Phi_1,\ldots,\Phi_n)$ – формула над F. Других формул нет.

Пример. Пусть $F = \{\vee, \wedge, {}^- \}$. Это множество называют *стандартным базисом.* Формулами в нем являются, например, $x \wedge y$, $x \vee y$, $\bar{x}$.

При записи формул опускают скобки, учитывая ассоциативность операций $((x \vee y) \vee z) = (x \vee y \vee z)$. Также опускают внешние скобки и используют соглашение о старшинстве операций, полагая, что самый высокий приоритет имеет отрицание, затем $\wedge$, затем $\vee$. С

учетом сказанного формула $((\overline{x \vee y}) \vee ((y \wedge z) \wedge u))$ может быть переписана так $(\overline{x \vee y}) \vee y \wedge z \wedge u$.

Рассмотрим множество базисных функций $F = \{\oplus, \cdot, 1\}$, которое называют базисом Жегалкина. Приоритет операции $\cdot$ выше, чем $\oplus$.

Примеры формул: $xy \oplus x \oplus y$, $x \oplus 1$, $xyz \oplus xy \oplus xz \oplus yz \oplus x \oplus y \oplus z \oplus 1$.

Пусть множество базисных функций $F = \{|\}$ (штрих Шеффера). Эта операция не ассоциативна, т. е. $x|(y|z) \neq (x|y)|z$.

Дадим определение подформулы. Пусть Ψ – формула над F. Если $\Psi \in F$ или Ψ – переменная, то она собственная подформула. Если Ψ имеет вид $f(\Phi_1, \ldots, \Phi_n)$, где $f \in F$, а Φ_1 – формула над F, то подформулами формулы Ψ будут 1) все формулы Φ_i, $i = \overline{1,n}$; 2) все подформулы формулы Φ_i.

Множество булевых функций F называют 1) *замкнутым*, если любая формула над F представляет функцию из F; 2) *полным*, если любая булева функция может быть представлена формулой над F.

Пример. Над стандартным базисом функции $\bar{}$, $\oplus$, $\to$, $\leftrightarrow$, $|$, $\downarrow$ записываются следующим образом $\bar{x} = x|x$, $x_1 \oplus x_2 = x_1\bar{x}_2 \vee \bar{x}_1 x_2$, $x_1 \to x_2 = \bar{x}_1 \vee x_2$, $x_1 \leftrightarrow x_2 = (\bar{x}_1 \vee x_2)(x_1 \vee \bar{x}_2)$, $x_1|x_2 = \overline{x_1 x_2}$, $x_1 \downarrow x_2 = \overline{x_1 \vee x_2}$.

Назовем *эквивалентными* формулы, представляющие равные функции. Необходимо уметь решать задачу перехода от данной формулы к более простой (в некотором смысле) эквивалентной ей. Также необходимо для любой булевой функции уметь находить представляющую ее формулу.

§4. Дизъюнктивные и конъюнктивные нормальные формы.

Любая формула вида x или $\bar{x}$ над стандартным базисом, где x – произвольная переменная называется *литералом*.

Еще используют обозначение

$$x^\sigma = \begin{cases} x, & \sigma = 1, \\ \bar{x}, & \sigma = 0. \end{cases}$$

Формула вида $K = x_1^{\sigma_1} x_2^{\sigma_2} \ldots x_n^{\sigma_n}$ называется *элементарной конъюнкцией*.

Дизъюнктивной нормальной формой (ДНФ) от переменных $x_1, \ldots, x_n$ называют формулу вида $K_1 \vee \ldots \vee K_m$, где K_i, $i = \overline{1,m}$ – элементарные конъюнкции, содержащие некоторые из литералов x_i.

В случае, когда в каждую конъюнкцию K_i входят литералы, ДНФ называют *совершенной ДНФ* (*СДНФ*). Используя принцип двойственности определяются *конъюнктивная нормальная форма* (*КНФ*) и *совершенная КНФ* (*СКНФ*).

Теорема (Шеннон). Любая булева функция, отличная от 0 (1) представима в виде СДНФ (СКНФ).

Доказательство для функции $f \not\equiv 0$. Рассмотрим для нее множество тех $\tilde{\alpha}$, при которых $f(\tilde{\alpha}) = 1$. Такие $\tilde{\alpha}$ называют *конституентами единицы*. Каждому $\tilde{\alpha}$ сопоставим элементарную конъюнкцию $K_{\tilde{\alpha}} = x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}$, которую также называют конституентой единицы. СДНФ будет иметь вид

$$f = \bigvee_{\tilde{\alpha}} K_{\tilde{\alpha}}.$$

Согласно принципу двойственности, СКНФ для той же функции будет иметь вид

$$f = \bigwedge_{\tilde{\alpha}} D_{\tilde{\alpha}},$$

где $\tilde{\alpha}$ таково, что $f(\tilde{\alpha}) = 0$. Такие $\tilde{\alpha}$ называют *конституентами* нуля. При этом элементарная дизъюнкция сопоставляется $\tilde{\alpha}$ по правилу

$$D_{\tilde{\alpha}} = x_1^{\bar{\alpha}_1} \vee \ldots \vee x_n^{\bar{\alpha}_n}.$$

Теорема доказана.

Из доказанного следует, что любая булева функция выразима через функции стандартного базиса, поэтому он полный.

Пример. Построим СДНФ мажоритарной функции. Конституентами единицы являются наборы $\tilde{\alpha}_1 = (0,1,1)$, $\tilde{\alpha}_2 = (1,0,1)$, $\tilde{\alpha}_3 = (1,1,0)$, $\tilde{\alpha}_4 = (1,1,1)$, им соответствуют $K_{\tilde{\alpha}_1} = \bar{x}_1 x_2 x_3$, $K_{\tilde{\alpha}_2} = x_1 \bar{x}_2 x_3$, $K_{\tilde{\alpha}_3} = x_1 x_2 \bar{x}_3$, $K_{\tilde{\alpha}_4} = x_1 x_2 x_3$. СДНФ имеет вид

$$\bar{x}_1 x_2 x_3 \vee x_1 \bar{x}_2 x_3 \vee x_1 x_2 \bar{x}_3 \vee x_1 x_2 x_3.$$

§5. Построение минимальных ДНФ.

СДНФ, которая строится по таблице булевой функции может быть сложной, т. е. содержит много элементарных конъюнкций и литералов. Необходимо уметь находить простую в некотром смысле ДНФ.

Уточним задачу.

Булеву функцию g называют *импликантой* функции f, если для любых наборов значений переменных из $g = 1$ следует $f = 1$.

Если f представима СДНФ, то любая ее элементарная конъюнкция или их дизъюнкция является импликантой f. Из определения равных булевых функций и импликанты следует, что $f = g$ только тогда, когда каждая из них является импликантой другой.

ДНФ называют минимальной, если она содержит наименьшее число литералов среди всех эквивалентных ДНФ.

Пример. ДНФ $x_1 x_2 \vee \bar{x}_1 x_2$ не является минимальной $x_1 x_2 \vee \bar{x}_1 x_2 = x_2(x_1 \vee \bar{x}_1) = x_2$. Вместо четырех литералов имеем один.

Длиной ДНФ называют количество входящих в нее элементарных конъюнкций.

ДНФ называют *кратчашей*, если она имеет наименьшую длину среди всех эквивалентных.

Рассмотрим метод построения минимальной ДНФ.

Алгоритм Куайна–Мак Клоски.

1. Склейка. Пусть K_1 и K_2 – элементарные конъюнкции, входящие в исходную ДНФ, которая представляет функцию f, причем $K_1 = xK$, $K_2 = \bar{x}K$. Тогда $K_1 \vee K_2 = xK \vee \bar{x}K = (x \vee \bar{x})K = K$.

Установим геометрический смысл простой склейки. Существует взаимнооднозначное соответствие между множеством элементарных конъюнкций и множеством конституент

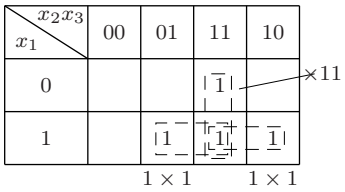
единицы. Простая склейка может быть применена только к таким двум элементарным конъюнкциям $K_{\tilde{\alpha}}$ и $K_{\tilde{\beta}}$, соответствующим $\tilde{\alpha}$ и $\tilde{\beta}$, что для некоторого i

$$\tilde{\alpha} = (\alpha_1, \ldots, \alpha_{i-1}, \alpha_i, \alpha_{i+1}, \ldots, \alpha_n),$$
$$\tilde{\beta} = (\alpha_1, \ldots, \alpha_{i-1}, \bar{\alpha}_i, \alpha_{i+1}, \ldots, \alpha_n)$$

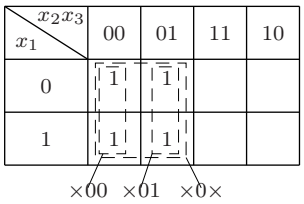
образуют ребро булева куба. Склейку применяют до тех пор пока это возможно. В результате получается сокращенная ДНФ. Ее элементарные конъюнкции являются простыми импликантами.

Для функций трех и четырех переменных процедура склейки наглядно выполняется по картам Карно. Это способ изображения булева куба размерности 3 или 4.

Пример.

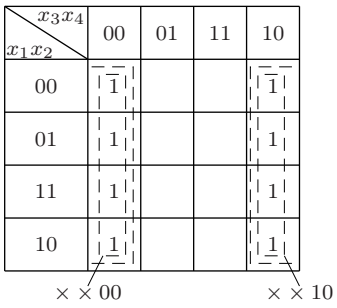


Наличие в карте Карно двух прямоугольников площади 2, находящихся в соседних столбцах или строках, показывает, что функция принимает значение 1 на паре соседних ребер. Они могут быть объединены в прямоугольник площади 4. Ребро $[110, 111]$ записывается в виде $11\times$, по этому обозначению легко получить импликанту x_1x_2, являющуюся результатом склейки.



В данном случае склеивается двумерная грань, ее обозначают $\times 0 \times$. Результат склейки: $\bar{x}_2$.

Можно объединять в один прямоугольник площади 8 два соседних прямоугольника площади 4.



2. Нахождение ядра. Говорят, что элементарная конъюнкция K *покрывает* элементарную конъюнкцию L $(K \succ L)$, если каждый литерал из K содержится в L. Например, $x_1x_2 \succ x_1x_2x_3$. Ясно, что если $K \succ L$, то $K \vee L = K$ согласно тождеству

поглощения. Каждая простая импликанта в ДНФ покрывает элементарную конъюнкцию в СДНФ. На карте Карно этому соответствует прямоугольник, закрывающий 1.

Простая импликанта называется *ядерной*, если она покрывает некоторую элементарную конъюнкцию, не покрываемую никакой другой.

Ни одна ядерная импликанта не может быть удалена из ДНФ. Например, у мажоритарной функции все импликанты ядерные.

3. Отыскание тупиковой ДНФ. Простую импликанту называют *избыточной* относительно ДНФ, содержащей только простые импликанты, если ее можно удалить из ДНФ без потери ее эквивалентности исходной СДНФ.

Любую ДНФ, не содержащую ни одной избыточной импликанты и содержащую все ядерные импликанты, называют *тупиковой*.

§6. Предикаты.

Пусть задано множество M. *n-местным предикатом* на множестве M называется функция $P^{(n)} : M^n \to \mathcal{B}$.

Пример. Пусть $M = \mathbb{N}$.

1. Предикат тождества $E : \mathbb{N}^2 \to \mathbb{B}$ $E(a_1, a_2) = 1 \Leftrightarrow a_1 = a_2$;
2. Предикат порядка $Q : \mathbb{N}^2 \to \mathbb{B}$ $Q(a_1, a_2) = 1 \Leftrightarrow a_1 \leqslant a_2$;
3. Предикат делимости $D : \mathbb{N}^2 \to \mathbb{B}$ $D(a_1, a_2) = 1 \Leftrightarrow a_1 \vdots a_2$;
4. Предикат суммы $S : \mathbb{N}^2 \to \mathbb{B}$ $S(a_1, a_2, a_3) = 1 \Leftrightarrow a_1 + a_2 = a_3$;
5. Предикат произведения $\Pi : \mathbb{N}^2 \to \mathbb{B}$ $\Pi(a_1, a_2, a_3) = 1 \Leftrightarrow a_1 \cdot a_2 = a_3$.

Моделью $\mathfrak{M}$ называется множество M с заданными на нем предикатами $P_1^{(k_1)}, \ldots, P_n^{(k_n)}$. Обозначают $\mathfrak{M} = \langle M, P_1^{(k_1)}, \ldots, P_n^{(k_n)} \rangle$.

Множество M называется *носителем* модели, набор предикатов $\zeta = \langle P_1^{(k_1)}, \ldots, P_n^{(k_n)} \rangle$ называется *сигнатурой* модели, $\tau = \langle k_1, \ldots, k_n \rangle$ называется *типом* модели.

Пример.

1. Модель $\mathfrak{N}_1 = \langle \mathbb{N}, E, S, \Pi \rangle$ является арифметикой натуральных чисел, сигнатура $\zeta = \langle E, S, \Pi \rangle$, тип $\tau = \langle 2, 3, 3 \rangle$.
2. Модель $\mathfrak{N}_2 = \langle \mathbb{N}, Q \rangle$ – упорядоченное множество натуральных чисел, $\zeta = \langle Q \rangle$, $\tau = \langle 2 \rangle$.
3. Модель $\mathfrak{N}_3 = \langle Q, E, S, \Pi \rangle$ – арифметика рациональных чисел.

Любое множество моделей с одной сигнатурой ζ называется *классом* K_ζ.

Пусть на множестве M задан предикат P Операция $\forall$ ставит в соответствие предикату P высказывание $\forall x P(x)$, которое истинно только тогда, когда $P(x) = 1$ при всех $x \in M$. Эта операция называется *навешиванием квантора общности*.

Операция $\exists$ ставит в соответствие предикату P высказывание $\exists x P(x)$, которое истинно только тогда, когда найдется $x \in M$ при котором $P(x) = 1$. Эта операция называется *навешиванием квантора существования*.

Пример. На множестве $\mathbb{N}$ $P(x) = 1$ только тогда, когда x – простое. $\forall x P(x)$ читается "любое число x простое". Это высказывание ложное. Высказывание $\exists x P(x)$ в данном случае истинно.

В случае предиката n переменных операция навешивания квантора определяется так:

$$\forall x_i P(x_1,\dots,x_i,\dots,x_n) = \\ = \begin{cases} 1, & \text{если для любого } x_i \in M \quad P(x_1,\dots,x_i,\dots,x_n)=1, \\ 0, & \text{если существует } x_i \in M \quad P(x_1,\dots,x_i,\dots,x_n)=0; \end{cases}$$

$$\exists x_i P(x_1,\dots,x_i,\dots,x_n) = \\ = \begin{cases} 1, & \text{если существует } x_i \in M \text{такое, что} \quad P(x_1,\dots,x_i,\dots,x_n)=1, \\ 0, & \text{если для любого } x_i \in M \qquad\qquad P(x_1,\dots,x_i,\dots,x_n)=0; \end{cases}$$

После навешивания квантора предикат P зависит от $n-1$ переменной.

Формулой называется

1. $P_i^{(k_i)}(x_1,\dots,x_{k_i})$, если $P_i^{(k_i)} \in \zeta$;

2. $\forall x_i A(x_1,\dots,x_i,\dots x_n)$, $\exists x_i A(x_1,\dots,x_i,\dots x_n)$, где $A(x_1,\dots,x_i,\dots x_n)$ – формула. При этом формула $A(x_1,\dots,x_i,\dots x_n)$ называется областью действия квантора;

3. $\overline{A}(x_1,\dots,x_i,\dots x_n)$, где $A(x_1,\dots,x_i,\dots x_n)$ – формула;

4. $A(x_1,\dots x_n)\wedge B(x_1,\dots x_n)$, $A(x_1,\dots x_n)\vee B(x_1,\dots x_n)$, $A(x_1,\dots x_n)\to B(x_1,\dots x_n)$, $A(x_1,\dots x_n)\leftrightarrow B(x_1,\dots x_n)$, где $A(x_1,\dots x_n)$ и $B(x_1,\dots x_n)$ формулы.

Поясним как вычисляется значение формулы сигнатуры ζ. Пусть $\mathfrak{M}$, A – формула сигнатуры ζ. Если A не содержит свободных вхождений переменных, то, используя определения логических операций и кванторов, можно вычислить значение формулы A.

Пример. Рассмотрим модель $\mathfrak{N} = \langle \mathbb{N}, E, S, \Pi\rangle$. Определим на этой модели значение формулы

$$A(x_2) = \exists x_3(\Pi(x_2,x_2,x_3)\to S(x_2,x_2,x_3))$$

для случая, когда $x_2 = 2$

$$A(2) = \exists x_3(\Pi(2,2,x_3)\to S(2,2,x_3)) = 1, \tag{1}$$

так как $\Pi(2,2,4)\to S(2,2,4)=1$. Если $x_2\neq 2$ и $\Pi(x_2,x_2,a)=1$, то $S(x_2,x_2,a)=0$ для всех $a\in\mathbb{N}$, следовательно, $A(x_2)=0$.

На той же модели $\mathfrak{N}$ формула $\forall x_1x_2x_3x_4(S(x_1,x_2,x_3)\wedge S(x_2,x_2,x_4)\to E(x_3,x_4))$ истинна. Она выражает однозначность операции сложения.

Формула $\forall x_1x_2x_3(\Pi(x_1,x_2,x_3)\wedge \overline{E}(x_1,x_2)\to S(x_1,x_2,x_3))$ на модели $\mathfrak{N}$ является ложным высказыванием, так как при $x_1\neq x_2$ $x_1x_2\neq x_1+x_2$.

Формула A называется *выполнимой на модели* $\mathfrak{M}$, если для некоторых $x_1,\dots,x_k\in M$ $A(x_1,\dots,x_k)=1$.

Формула A называется *выполнимой*, если существует модель, на которой она выполнима.

Формула A называется *истинной на модели* $\mathfrak{M}$, если для всех $x_1,\dots,x_k\in M$ выполняется $A(x_1,\dots,x_k)=1$.

Если формула не выполнима на модели $\mathfrak{M}$, то она называется ложной на модели $\mathfrak{M}$.

Пример. Формула (1) выполнима, но не истинна на модели $\mathfrak{N}$.

Пусть A – формула алгебры предикатов и $F_1^{(k_1)},\dots,F_n^{(k_n)}$, входящие в нее предикаты.

Сигнатуру ζ и класс K_ζ и каждую модель $\mathfrak{M} \in K_\zeta$ назовем *допустимыми* для A, если ζ содержит хотя бы один предикат с k_i-арный предикат $i = \overline{1,n}$.

Если $\mathfrak{M}$ – допустимая модель для формулы A, то можно строить сигнатурные отображения σ множества предикатных переменных $F_i^{(k_i)}$ так, чтобы образом $\sigma F_i^{(k_i)}$ был предикат того же числа аргументов. Такое отображение назовем *сигнатурным*. Таким образом, формуле A сопоставляется σA.

Пример.

$$A = \forall x_1 x_2 x_3 x_4 (F_1(x_1, x_2, x_3) \wedge F_1(x_1, x_2, x_4) \to F_2(x_3, x_4)).$$

Модель $\mathfrak{N}$ является допустимой для A. Определим сигнатурное отображение σ', положив $\sigma' F_1^{(3)} = S$, $\sigma' F_2^{(2)} = E$, тогда

$$\sigma' A = \forall x_1 x_2 x_3 x_4 (S(x_1, x_2, x_3) \wedge S(x_1, x_2, x_4) \to E(x_3, x_4)).$$

При отображении σ'' таком, что $\sigma'' F_1^{(3)} = \Pi$, $\sigma'' F_2^{(2)} = E$

$$\sigma'' A = \forall x_1 x_2 x_3 x_4 (\Pi(x_1, x_2, x_3) \wedge \Pi(x_1, x_2, x_4) \to E(x_3, x_4)).$$

Эти формулы выражают однозначность операций сложения и умножения.

Формула называется *общезначимой*, если она тождественно истинна на любой допустимой модели.

Пример. Формула $F(x_1, \ldots, x_n) \vee \overline{F}(x_1, \ldots, x_n)$ общезначима.

Всякая тавтология является общезначимой формулой, а также общезначимыми являются формулы

$$A(x_1, \ldots, x_n) \to \exists x_1, \ldots, x_n A(x_1, \ldots, x_n);$$
$$\forall x_1, \ldots, x_n A(x_1, \ldots, x_n) \to A(x_1, \ldots, x_n).$$

Формулы A и B называют *эквивалентными*, если формула $A \leftrightarrow B$ общезначима. Обозначают $A = B$.

Часто используются эквивалентности

$$\overline{\forall x_i A(\ldots, x_i, \ldots)} = \exists x_i \overline{A(\ldots, x_i, \ldots)}, \tag{2}$$
$$\overline{\exists x_i A(\ldots, x_i, \ldots)} = \forall x_i \overline{A(\ldots, x_i, \ldots)} \tag{3}$$

Докажем (2). Пусть на некоторой модели имеем $\overline{\forall x_i \sigma A(\ldots, x_i, \ldots)} = 1$, $\forall x_i \sigma A(\ldots, x_i, \ldots) = 0$. Последнее означает, что $\sigma A(\ldots, x_i, \ldots) = 0$ хотя бы для одного $x_i \in M$ или $\overline{\sigma A(\ldots, x_i, \ldots)} = 1$, т. е. $\overline{\exists x_i \overline{A(\ldots, x_i, \ldots)}} = 1$. Таким образом, доказано, что $\overline{\forall x_i A(\ldots, x_i, \ldots)} = 1$ влечет $\exists x_i \overline{A(\ldots, x_i, \ldots)} = 1$. Аналогично доказываются остальные ситуации.

Также часто используются следующие эквивалентности

$$\forall x (A_1(x) \wedge A_2(x)) = (\forall x A_1(x)) \wedge (\forall x A_2(x)),$$

$$\forall x (A_1(x) \vee A_2(x)) = (\forall x A_1(x)) \vee (\forall x A_2(x)),$$

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

$$\exists x \exists y A(x, y) = \exists y \exists x A(x, y).$$

Пример. Докажем $\exists x(F_1(x) \to F_2(x)) = \forall F_1(x) \to \exists x F_2(x)$.

$$\exists x(F_1(x) \to F_2(x)) = \exists x(\overline{F_1}(x) \vee F_2(x)),$$
$$\exists x(\overline{F_1}(x) \vee F_2(x)) = \exists x\overline{F_1}(x) \vee \exists F_2(x),$$
$$\exists x\overline{F_1}(x) \vee \exists F_2(x) = \overline{\forall x F_1}(x) \vee \exists F_2(x)$$
$$\overline{\forall x F_1}(x) \vee \exists F_2(x) = \forall F_1(x) \to \exists x F_2(x).$$

§7. Исчисление высказываний.

Алфавитом называется множество A символов.

Множество всех слов S_A в алфавите A называется *языком*.

Длиной слова называется количество букв в нем.

Если к слову $\alpha \in S_A$ приписать справа слово $\beta \in S_A$, то получится слово $\alpha\beta \in S_A$, называемое *композицией* слов.

Пустое слово обозначается λ.

Слово $\alpha_1 \in S_A$ называется *подсловом* в слове $\alpha \in S_A$, если существуют такие слова $\delta \in S_A$ и $\gamma \in S_A$, что $\alpha = \delta\alpha_1\gamma$.

Алфавитом исчисления высказываний назовем следующее множество символов: 1) Латинские символы; 2) $\wedge$, $\vee$, $\to$, ; 3) (,).

Формулами исчисления высказываний называются только те слова, которые образуются по правилам:

1) Латинский символ – формула; 2) Если A и B – формулы, то $\overline{A}$, $A \vee B$, $A \wedge B$, $A \to B$ также формулы.

Всякое подслово, являющееся формулой, называется *подформулой*.

Аксиомы исчисления высказываний.

1. $A \to (B \to A)$,
2. $(A \to B) \to ((A \to (B \to C)) \to (A \to C))$;
3. $A \wedge B \to A$;
4. $A \wedge B \to B$;
5. $A \to (B \to A \wedge B)$;
6. $A \to A \vee B$;
7. $B \to (A \vee B)$;
8. $(A \to C) \to ((B \to C) \to (A \vee B \to C))$;
9. $(A \to B) \to ((A \to \overline{B}) \to \overline{A})$;
10. $\overline{\overline{A}} = A$.

Правила вывода.

1. Modus ponens (MP). Из A, $A \to B$ выводимо B. Обозначается $A, A \to B \vdash B$;

2. Правило подстановки: из формулы A выводима формула B, которая получается из A заменой каждого вхождения высказывательного переменного $x_1, \ldots, x_k$ формулами $B_1, \ldots, B_k$.

Формула B называется *доказуемой* в исчислении высказываний (*теоремой* исчисления высказываний), если существует конечная последовательность формул $B_1, \ldots, B_t$, в которой $B_t = B$ и каждая из B_i является либо аксиомой, либо получена по правилу вывода из предыдущих формул последовательности. Обозначение: $\vdash B$.

Последовательность $B_1, \ldots, B_t$ называется доказательством, t – длиной доказательства.

Пример. Доказать $\vdash A \to A$.

1. В аксиоме 2 положим $B := B \to A$, $C := A$, получим $(A \to B) \to ((A \to (B \to C)) \to (A \to C))$;

2. аксиома 1 $A \to (B \to A)$;

3. Из п. 1, 2 по MP получаем $(A \to ((B \to A) \to A)) \to (A \to A)$;

4. В аксиоме полагаем $B := B \to A$, получаем $A \to ((B \to A) \to A)$;

5. Из п. 3, 4 по MP получаем $A \to A$.

Формула B называется выводимой из формул $A_1, \ldots, A_s$, если существует такая последовательность формул $B_1, \ldots, B_t$, каждая из которых либо теорема, либо A_i, либо получена из предыдущих правилу modus ponens.

Теорема. Формула B является теоремой исчисления высказываний только тогда, когда она выводима из пустого множества посылок.

Доказательство. Если B – теорема, то для нее можно построить вывод длины 1. Вывод состоит из нее самой. Обратно, если $B_1, \ldots, B_t$, – вывод из пустого множества посылок, то, заменив в этом выводе каждую теорему ее доказательством, получаем вывод B. Теорема доказана.

Пример. 1. Доказать $A \to B \to A$.

1. По условию A;

2. Аксиома 1 $A \to (B \to A)$;

3. Из п. 1, 2 $B \to A$.

Пример. 2. Доказать $A \wedge B \vdash A$.

1. По условию $A \wedge B$;

2. Аксиома 3 $A \wedge B \to A$;

3. Из п. 1, 2 по MP получаем A.

Пример 3. Доказать $A, B \vdash A \wedge B$.

1. По условию имеем A;

2. Аксиома 5 $A \to (B \to (A \wedge B))$;

3. Из п. 1, 2 по MP имеем $B \to A \wedge B$;

4. По условию имеем B;

5. Из п. 3,4 по MP имеем $A \wedge B$.

Аналогично можно доказать

$$\begin{array}{ll} A \wedge B \vdash B; & \overline{\overline{A}} \vdash A; \\ A \vdash A \vee B; & A \vee B \vdash B \vee A; \\ B \vdash A \vee B; & A \wedge B \vdash B \wedge A. \end{array}$$

Вспомогательные правила вывода.

1. Повторение посылки: $T, A \vdash A$;

2. Введение посылки: если $T \vdash A$, то $T, B \vdash A$;

3. Удаление посылки: если $T, A \vdash B$ и $T \vdash A$, то $T \vdash B$;

4. Правило силлогизма: если $T \vdash A_1, \ldots, T \vdash A_k$ и $A_1, \ldots, A_k \vdash B$, то $T \vdash B$;

5. Введение импликации: если $T, A \vdash B$, то $T \vdash A \to B$;

6. Удаление импликации: если $T \vdash A \to B$, то $T, A \vdash B$;

7. Введение конъюнкции: $T, A, B \vdash A \wedge B$;

8. Удаление конъюнкции: $T, A \wedge B \vdash A$, $T, A \wedge B \vdash B$;

9. Введение дизъюнкции: $T, A \vdash A \vee B$;

10. Удаление дизъюнкции: если $T, A \vdash C$ и $T, B \vdash C$, то $T, A \vee C \vdash C$;

11. Введение отрицания: если $T, A \vdash B$ и $T, A \vdash \overline{B}$, то $T \vdash \overline{A}$;

12. Удаление отрицания: если $T, \overline{\overline{A}} \vdash A$;

13. Правило контрапозиции: если $T, A \vdash B$, то $T, \overline{B} \vdash \overline{A}$.

В более общей форме правило 5 может быть записано в виде теоремы дедукции.

Теорема. (дедукции) Если $A_1, \ldots, A_{k-1}, A_k \vdash B$, то $A_1, \ldots A_{k-1} \vdash A_k \to B$.

Доказательство. Пусть $B_1, \ldots, B_t$ – вывод формулы B из $A_1, \ldots, A_k$. Индукцией по t докажем

$$A_k \to A_1, A_k \to A_2, \ldots, A_k \to A_{k-1}, A_k \to A_k \vdash A_k \to B. \tag{4}$$

Пусть $t = 1$, тогда по определению $B = A_i$ или B – теорема. Если $B = A_i$, то $A_k \to B = A_k \to A_i$ и утверждение (4) верно, т. к. $A_k \to A_i$ – посылка.

Если B – теорема, то из B и $B \to (A_k \to B)$ по MP получаем $A_k \to B$.

Допустим, что (4) верно при $t = n$ и докажем при $t = n + 1$. Теперь B может быть теоремой или $B = A_i$ или получена из предыдущих по MP. В первых двух случаях доказательство такое же как и выше. Рассмотрим третий случай.

Итак, для $i, j \leqslant n$ $B_i, B_j \vdash B$ по MP. Тогда $B_j = B_i \to B$. По предположению индукции имеем

$$A_k \to A_1, A_k \to A_2, \ldots, A_k \to A_k \vdash A_k \to B_i \tag{5}$$

$$A_k \to A_1, A_k \to A_2, \ldots, A_k \to A_k \vdash A_k \to B_i \tag{6}$$

С другой стороны, из теоремы

$$\vdash (A_k \to B_i) \to ((A_k \to (B_i \to B)) \to (A_k \to B)), \tag{7}$$

получаемой из аксиомы 2 подстановкой $A := A_k$, $B := B_i$, $C := B$.

Из соотношений (5), (6), (7) по правилу силлогизма получаем:

$$A_k \to A_1, A_k \to A_2, \ldots, A_k \to A_k \vdash A_k \to B.$$

Формула $A_k \to A_k$ является теоремой и ее можно удалить из посылок

$$A_k \to A_1, A_k \to A_2, \ldots, A_{k-1} \to A_k \vdash A_k \to B. \tag{8}$$

Полагаем в аксиоме 1 $A := A_i$, $B := A_k$, получаем $\vdash A_i \to (A_k \to A_i)$. По MP получаем $A_i, A_i \to (A_k \to A_i) \vdash A_k \to A_i$. Отсюда и из (8) по правилу силлогизма получаем искомое. Теорема доказана.

Формулы A и B называют *эквивалентными* и пишут $A = B$ $(A \leftrightarrow B)$, если

$$\vdash (A \to B) \wedge (B \to A).$$

Эквивалентности исчисления высказываний повторяют эквивалентности алгебры высказываний.

Метатеория исчисления высказываний.

Синтаксическая теория, содержащая знак отрицания, называется *непротиворечивой,* если не существует формулы A такой, что A и $\overline{A}$ теоремы этой теории. Иначе теория называется *противоречивой.*

Можно дать эквивалентное определение.

Синтаксическая теория называется *непротиворечивой,* если существует формула, не являющаяся ее теоремой и *противоречивой*, если любая формула является ее теоремой.

Теорема. Всякая формула, являющаяся теоремой исчисления высказываний, является тавтологией в алгебре высказываний.

Теорема. В исчислении высказываний доказуема любая формула, являющаяся тавтологией и исчислении высказываний.

Из приведенных теорем следует, что исчисление высказываний является непротиворечивой теорией.

§8. Исчисление предикатов

Аксиомами исчисления предикатов являются все аксиомы исчисления высказываний 1 – 10, а также

11. $(\forall x)A(x) \to A(y)$,

12. $A(y) \to (\exists x)A(x)$.

Правилами вывода являются:

1) modus ponens;

2) Правило $\forall$-введения. Если в формуле $B(x)$ имеются свободные вхождения x, а в формуле A их нет, то из $A \to B(x)$ выводима формула $A \to (\forall x)B(x)$.

$$A \to B(x) \vdash A \to (\forall x)B(x);$$

3) Правило $\exists$-удаления. Если в формуле $B(x)$ имеются свободные вхождения x, а в формуле A их нет, то из $B(x) \to A$ выводима формула $(\exists x)B(x) \to A$.

$$B(x) \to A \vdash (\exists x)B(x) \to A.$$

Формула B называется *доказуемой* в исчислении предикатов (*теоремой* исчисления предикатов), если существует конечная последовательность формул $B_1, \ldots, B_t$, в которой $B_t = B$ и каждая из B_i является либо аксиомой, либо получена по правилу вывода из предыдущих формул последовательности. Обозначение: $\vdash B$.

Последовательность $B_1, \ldots, B_t$ называется *доказательством*, t *– длиной* доказательства.

Отметим, что множество доказуемых формул не расширится. если при доказательствах пользоваться ранее доказанными формулами.

Докажем два вспомогательных правила вывода.

1. Правило переименования свободных переменных.

Если формула $A(x)$ содержит свободные вхождения x и ни одно их них не содержится в области действия квантора по y, то из $\vdash A(x)$ следует $\vdash A(y)$.

Доказательство. Пусть B – произвольная доказуемая формула исчисления предикатов, не содержащая свободных вхождений x и построим вывод $A(y)$. Сначала построим вывод $A(x)$. По условию он существует и заканчивается формулой $A(x)$. Возьмем аксиому 1:

$$\alpha = A(x) \to (B \to A(x)).$$

По правилу modus ponens $A(x), \alpha \vdash B \to A(x)$. Отсюда по правилу $\forall$-введения выводима формула $B \to (\forall x)A(x)$. Построим вывод формулы B, который существует в виду выбора B. По правилу modus ponens $B, B \to (\forall x)A(x) \vdash (\forall x)A(x)$. Из формулы $(\forall x)A(x)$ и аксиомы 11 выводима формула $A(y)$.

2. Правило переименования связанных переменных.

Если формула $A(x)$ не содержит свободных вхождений y и содержит свободные вхождения x, ни одно из которых не находится в области действия квантора по переменной y, то из $\vdash (\sigma x)A(x)$ следует $\vdash (\sigma y)A(y)$, где σ – любой квантор. Формула $A(y)$ получена из $A(x)$ заменой всех свободных вхожений x на y.

Доказательство. Рассмотрим 2 случая.

I. Пусть $\delta = \forall$. Построим доказательство для формулы $(\forall y)A(y)$.

1. $(\forall x)A(x) \to A(y)$ аксиома 11;
2. $(\forall x)A(x) \to (\forall y)A(y)$ по правилу $\forall$-введения из п. 1;
3. $\vdash (\forall x)A(x)$ по условию.
4. $(\forall y)A(y)$ из п. 2, 3 по modus ponens.

II. Пусть $\delta = \exists$. Построим доказательство для формулы $(\exists x)A(x)$.

1. $A(y) \to (\exists x)A(x)$ аксиома 12;
2. $(\exists y)A(y) \to (\exists x)A(x)$ правило $\exists$-удаление;
3. $(\exists y)A(y)$ по условию;
4. $(\exists x)A(x)$ из п. 2, 3 по modus ponens.

Из этого правила следует

$$\vdash (\sigma x)A(x) \to (\sigma y)A(y).$$

Пример 1. Доказать $\vdash \exists x \forall y A(x,y) \to \forall y \exists x A(x,y)$.

1. $(\forall y)A(x,y) \to A(x,z)$ аксиома 11;
2. $A(x,z) \to (\exists v)A(v,z)$ аксиома 12;
3. $(\exists v)A(v,z) \to (\exists x)A(x,z)$ переименование связанных переменных;
4. $A(x,z) \to (\exists x)A(x,z)$ из п. 2, 3 по правилу силлогизма;
5. $(\forall y)A(x,y) \to (\exists x)A(x,z)$ из п. 1, 4 по правилу силлогизма;
6. $\exists x \forall y A(x,y) \to (\exists x)A(x,z)$ из п. 5 по правилу $\exists$-удаления;
7. $\exists x \forall y A(x,y) \to \forall x \exists z A(x,z)$ из п. 6 по правилу $\forall$-введения;
8. $\forall z \exists x A(x,z) \to \forall y \exists x A(x,y)$ переименование связанных переменных;
9. $\forall x \forall y A(x,y) \to \forall y \exists x A(x,y)$ из п. 7, 8 по правилу силлогизма.

Пример 2. Доказать $\vdash \overline{(\exists x)A(x)} \to (\forall x)\overline{A(x)}$.

1. $\overline{A(y) \to (\exists x)A(x)}$ аксиома 12;
2. $\overline{(\exists x)A(x)} \to \overline{A(y)}$ правило контрапозиции;
3. $\overline{(\exists x)A(x)} \to (\forall y)\overline{A(y)}$ из п. 2 по правилу $\forall$-введения;
4. $(\forall y)\overline{A(y)} \to (\forall x)\overline{A(x)}$ переименование связанных переменных;
5. $\vdash \overline{(\exists x)A(x)} \to (\forall x)\overline{A(x)}$ по правилу силлогизма из п. 3, 4.

Теорема. Исчисление предикатов непротиворечиво, т.е. ни для одной формулы A и $\overline{A}$ не являются одновременно теоремами этой теории.

Теорема. Исчисление предикатов полно относительно алгебры предикатов, т.е. всякая общезначимая формула алгебры предикатов является теоремой исчисления предикатов.

Исчисление высказываний полно в узком смысле, т. е. при добавлении к аксиомам исчисления высказываний недоказуемой в нем формулы получим противоречивую

теорию. В исчислении предикатов это не так. При добавлении к аксиомам исчисления предикатов формулы $(\exists x)A(x) \to (\forall x)A(x)$, которая очевидно не общезначима, получим непротиворечивую теорию S.

§9. Нечеткие множества, отношения и логика.

Пусть E – универсальное множество, $x \in E$, G – некоторое свойство. Обычное подмножество $A \subset E$, элементы которого удовлетворяют свойству G, определяется как множество пар $A = \{\mu_A(x)|x\}$, где $\mu_A(x) = 1$, если x обладает свойством G и 0 – в противном случае.

Если полагать, что $\mu_A(x) \in [0;1]$, то множество A называется *нечетким*. Функция μ_A называется *функцией принадлежности* и показывает степень принадлежности элемента x множеству A.

Пример 1. $A = \{0,3|x_1; 0|x_2; 1|x_3; 0,5|x_4; 0,9|x_5\}$.

Пример 2. Пусть $E = \{1, \ldots, 100\}$ – возрасты. Нечеткое множество "молодой" можно определить так

$$\mu(x) = \begin{cases} 1 & x \in [1;25], \\ \dfrac{1}{1 + \left(\frac{x-25}{5}\right)^2} & x > 25. \end{cases}$$

Число $\sup\limits_{x \in E} \mu_A(x)$ называется *высотой* множества. Если высота равна 1, то множество называется *нормальным*. Нечеткое множество A пусто, если для любого $\in E$ $\mu_A(x) = 0$. Множество A *унимодально*, $\mu_A(x) = 1$ только для одного $x \in E$.

Методы построения функции принадлежности.

Существуют прямые и косвенные методы построения функции принадлежности.

При использовании прямых методов эксперт просто задает для каждого $x \in E$ значение $\mu_A(x)$. Как правило, прямые методы используются для измеримых понятий, таких как скорость, время, расстояние, давление, температура и т.д., или когда выделяются полярные значения.

Во многих задачах при характеристике объекта можно выделить набор признаков и для каждого из них определить полярные значения, соответствующие значениям функции принадлежности, 0 или 1. Для конкретного объекта эксперт, исходя из приведенной шкалы, задает $\mu_A(x) \in [0,1]$, формируя векторную функцию принадлежности $\{\mu_A(x_1), \mu_A(x_2), \ldots, \mu_A(x_n)\}$.

Разновидностью прямых групповых методов построения функции принадлежности являются прямые групповые методы, когда, например, группе экспертов предъявляют конкретный объект, и каждый должен дать ответ: принадлежит или не принадлежит этот объект заданному множеству. Тогда число утвердительных ответов, деленное на общее число ответов, дает значение функции принадлежности объекта данному нечеткому множеству.

Косвенные методы определения значений функции принадлежности используются в случаях, когда нет измеримых элементарных свойств, через которые определяется нечеткое множество. Как правило это методы попарных сравнений. Если бы значения

функции принадлежности были известны, например, $\mu_A(x_i) = w_i,\ i = 1, 2, \ldots, n,$ то попарные сравнения можно представить матрицей отношений $A = (a_{ij})$, где $a_{ij} = \frac{w_i}{w_j}$.

На практике эксперт сам формирует матрицу A, при этом предполагается, что диагональные элементы равны 1, а для элементов, симметричных относительно главной диагонали, $a_{ij} = \frac{1}{a_{ji}}$. В этом общем случае задача сводится к поиску вектора w, удовлетворяющего уравнению вида $Aw = \lambda_{\max} w$, где $\lambda_{\max}$ – наибольшее собственное значение матрицы A.

Использование типовых форм кривых для задания функций принадлежности с уточнением их параметров в соответствии с данными эксперимента.

Использование относительных частот по данным эксперимента в качестве значений принадлежности.

Операции над нечеткими множествами.

Множество A *содержится* в B, если для любого $x \in E\ \ \mu_A(x) \leqslant \mu_B(x)$. Обозначение $A \subset B$.

Множество A *равно* B, если для любого $x \in E\ \ \mu_A(x) = \mu_B(x)$. Обозначение $A = B$.

Множество $\overline{A}$ называется *дополнением* к A, если для любого $x \in E\ \ \mu_{\overline{A}} = 1 - \mu_A(x)$. Очевидно, что $\overline{\overline{A}} = A$.

Пересечением множеств A и B называется множество $A \cap B$ с фукнцией принадлежности $\mu_{A\cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}$.

Объединением множеств A и B называется множество $A \cup B$ с фукнцией принадлежности $\mu_{A\cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}$.

Разностью множеств A и B называется множество $A \setminus B = A \cap \overline{B}$.

Для операций объединеneия, пересечения выполняются свойства коммутативности, ассоциативности, идемпотентности и дистрибутивности, $A \cap \varnothing = A,\ A \cup \varnothing = A,\ A \cap E = A,$ $A \cup E = E,$ законы де Моргана. Однако в отличии от булевой алгебры, здесь $A \cap \overline{A} \neq \varnothing,$ $A \cup \overline{A} \neq E$.

Прямым произведением нечетких подмножеств $A_1 \subset E_1,\ A_2, \subset E_2 \ldots,\ A_n \subset E_n$ называют подмножество $A_1 \times A_2 \times \ldots \times A_n \subset E_1 \times E_2 \times \ldots \times E_n$ с функцией принадлежности $\mu_A(x_1, x_2, \ldots, x_n) = \min\{\mu_{A_1}(x), \ldots, \mu_{A_n}(x)\}$.

Нечеткое n-арное отношение определяется как подмножество $R \subset E = E_1 \times \ldots \times E_n$ с функцией $\mu_R \in [0; 1]$. При $n = 2$ пишут $R : (X, Y) \to [0, 1]$.

Пример 1. $X = \{x_1, x_2, x_3\},\ Y = \{y_1, y_2, y_3, y_4\}$. Нечеткое бинарное отношение может быть задано таблицей

	y_1	y_2	y_3	y_4
x_1	0	0	0,1	0,3
x_2	0	0,8	1	0,7
x_3	1	0,5	0,6	1

Пример 2. Если $X = Y = \mathbb{R}$. Отношение R, для которого $\mu_R(x, y) = e^{-k(x-y)^2}$, при достаточно больших k можно интерпретировать так: x и y близкие друг к другу числа.

Операции над отношениями определяются так же как и операции над нечеткими множествами.

Объединение: $\mu_{R_1 \cup R_2} = \mu_{R_1}(x, y) \vee \mu_{R_2}(x, y)$.

Пересечение: $\mu_{R_1 \cap R_2} = \mu_{R_1}(x,y) \wedge \mu_{R_2}(x,y)$.

Дополнение: $\mu_{\overline{R}}(x,y) = 1 - \mu_R(x,y)$.

Композиция $R_1 \circ R_2$ отношений $R_1 : X \times Y \to [0,1]$ $R_2 : X \times Y \to [0,1]$ определяется функцией принадлежности

$$\mu_{R_1 \circ R_2}(x,y) = \bigvee_y (\mu_{R_1}(x,y) \wedge \mu_{R_2}(y,z)). \tag{9}$$

Пример 3. Найдем композицию отношений

R_1	y_1	y_2	y_3
x_1	$0,1$	$0,7$	$0,4$
x_2	1	$0,5$	0

R_1	z_1	z_2	z_3	z_4
y_1	$0,9$	0	0	$0,2$
y_2	$0,3$	$0,6$	0	$0,9$
y_3	$0,3$	1	0	$0,5$

$R_1 \circ R_2$	z_1	z_2	z_3	z_4
x_1	$0,3$	$0,6$	$0,1$	$0,7$
x_2	$0,9$	$0,5$	1	$0,5$

Элементы последней таблицы вычисляются по формуле (9), например,

$$\begin{aligned}\mu_{R_1 \circ R_2}(x_1,z_1) = (\mu_{R_1}(x_1,y_1) \wedge \mu_{R_2}(y_1,z_1)) \vee \\ \vee (\mu_{R_1}(x_1,y_2) \wedge \mu_{R_2}(y_2,z_1)) \vee (\mu_{R_1}(x_1,y_3) \wedge \mu_{R_2}(y_3,z_1)) = \\ = (0,1 \wedge 0,9) \vee (0,7 \wedge 0,3) \vee (0,4 \wedge 0,1) = \\ = 0,1 \vee 0,3 \vee 0,1 = 0,3.\end{aligned}$$

Нечеткий логический вывод.

Используемый в различных экспертных системах механизм нечетких выводов в своей основе имеет базу знаний, формируемую специалистами предметной области как совокупность нечетких предикатных правил вида:

$$\begin{aligned}&\Pi_1 : \text{ если } x \text{ есть } A_1, \text{ то } y \text{ есть } B_1,\\ &\Pi_2 : \text{ если } x \text{ есть } A_2, \text{ то } y \text{ есть } B_2,\\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\ &\Pi_n : \text{ если } x \text{ есть } A_3, \text{ то } y \text{ есть } B_n,\end{aligned}$$

где x – входная переменная (имя для известных значений данных); y – переменная вывода (имя для значений, которые будут вычислены); A и B – функции принадлежности, определенные на x и y.

Более детально. Знание эксперта $A \to B$ отражает нечеткое причинное отношение посылки и заключения, поэтому его можно назвать нечетким отношением и обозначать через R:

$$R = A \to B,$$

где " $\to$ " называют нечеткой импликацией. Отношение R можно рассматривать как нечеткое подмножество прямого произведения $X \times Y$ полного множества посылок и заключений Y. Таким образом процесс получения (нечеткого) результата вывода B' с использованием данного наблюдения A' и знания $A \to B$ можно представить в виде композиционного правила нечеткий "modus ponens:"

$$B' = A' \circ (A \to B).$$

Пример. Пусть некоторая система описывается следующими нечеткими правилами:

$$\begin{array}{l} \Pi_1: \text{ если } x \text{ есть } A, \text{ то } w \text{ есть } D, \\ \Pi_2: \text{ если } y \text{ есть } B, \text{ то } y \text{ есть } E, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \Pi_n: \text{ если } z \text{ есть } C, \text{ то } w \text{ есть } F, \end{array}$$

где x, y, z – имена входных переменных, w – имя переменной вывода, A, B, C, D, E, F – заданные функции принадлежности.

Пусть заданы конкретные (четкие) значения входных переменных x_0, y_0, z_0. Процесс построения логического вывода происходит в четыре этапа.

На первом этапе на основании данных значений и, исходя из функций принадлежности $A, B, C,$ находятся степени истинности $\alpha(x_0)$, $\alpha(y_0)$, $\alpha(z_0)$ для предпосылок каждого из трех приведенных правил.

На втором этапе происходит "отсекание" функций принадлежности заключений правил D, E, F на уровнях $\alpha(x_0)$, $\alpha(y_0)$, $\alpha(z_0)$.

На третьем этапе рассматриваются функции принадлежности, усеченные на предыдущем этапе, и производится их объединение с использованием операции max, в результате чего получается комбинированное нечеткое подмножество, описываемое функцией принадлежности $\mu(w)$ и соответствующее логическому выводу для выходной переменной w.

Наконец, на четвертом этапе находится, при необходимости четкое значение выходной переменной, например, с применением центроидного метода: четкое значение переменной определяется как центр тяжести кривой $\mu(w)$:

$$w_0 = \frac{\int\limits_{\Omega} w\mu(w)\, dw}{\int\limits_{\Omega} \mu(w)\, dw}.$$

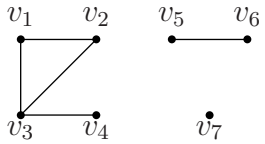
ГРАФЫ
§1. Основные понятия

Неориентированные графы	Ориентированные графы
Задается парой множеств $G = (V, E)$, где V – множество *вершин*, E – множество неупорядоченных пар на V (множество *ребер*) $\{u, v\} \in E$ $u \neq v$.	Задается парой множеств $G = (V, E)$, где V – множество вершин, E – множество упорядоченных пар на V (подмножество в $V \times V$), элементы которого называют *дугами*.
Если ребро $e = \{u, v\}$, то говорят, что ребро e *соединяет* вершины u и v и обозначают $u \mapsto v$	Если $e = (u, v)$, то говорят, что дуга e *ведет* из u в v и обозначают $u \to v$.
Вершины u и v, соединенные ребром $u \mapsto v$, называют *смежными*, а также концами ребра $\{u, v\}$. Если $u \mapsto v$, то говорят, u и v связаны отношением непосредственной достижимости.	Вершины u и v такие, что из u в v ведет дуга $u \to v$, называют *смежными*, u называют *началом*, а v – *концом* дуги (u, v). Если $u = v$, то дугу называют *петлей*. Если $u \to v$, то u и v связаны отношением непосредственной достижимости.
Ребро e называют *инцидентным* вершине v, если v является одним из его концов.	Дугу (u, v) называют исходящей из u и заходящей в v. Дугу (u, v) называют *инцидентной* вершинам u и v.
Степенью вершины v называют число $\deg v$ всех инцидентных ребер	*Полустепенью захода* вершины v называется число $\deg^- v$ заходящих в нее дуг, *полустепенью исхода* – число $\deg^+ v$ исходящих дуг. Степень вершины v – число $\deg v = \deg^+ v + \deg^- v$.
Для вершины v множество $\Gamma(v) = \{x : x \mapsto v\}$ называют *множеством смежных* вершин с v (*окрестностью* v).	Для вершины v множество $\Gamma(v) = \{x : v \to x\}$ называют множеством *преемников* вершины v, а множество $\Gamma^{-1}(v) = \{x : x \to v\}$ – множество *предшественников* вершины v.

Неориентированные графы	Ориентированные графы
Цепь в графе G – последовательность вершин v_0, $v_1, \ldots,$ v_n такая, что $v_i \mapsto v_{i+1}$ для $i = 1, 2, \ldots, n-1$.	*Путь* в орграфе G – последовательность вершин v_0, $v_1, \ldots,$ v_n такая, что $v_i \to v_{i+1}$ для $i = 1, 2, \ldots, n-1$.
Число n называют *длиной* цепи, т. е. это число ребер в цепи. Цепь длины 0 – вершина.	Число n *длина* пути, т. е. число дуг.
Говорят, что вершина v графа G достижима из вершины u, если существует цепь v_0, $v_1, \ldots,$ v_n, где $u = v_0$, $v_n = u$. Также говорят, что цепь соединяет u и v.	Говорят, что вершина v орграфа G достижима из вершины u, если существует путь v_0, $v_1, \ldots,$ v_n, где $u = v_0$, $v_n = u$. v – начало, u – конец пути.
Простая цепь – цепь все вершины которой, может быть, кроме первой и последней различны и все ребра различны.	*Простой путь* – путь, все вершины которого, кроме, может быть, первой и последней различны.
Простую цепь ненулевой длины с совпадающими концами называют *циклом*.	Простой путь ненулевой длины с совпадающими началом и концом называют *контуром*.
Граф без циклов называют *ациклическим*.	Граф без контуров называют *безконтурным*.
Граф называют *связным*, если любые две его вершины соединены цепью.	Орграф называют *связным*, если для любых двух вершин одна достижима из другой.
Компонента связности – максимальный связный подграф.	*Компонента связности* – максимальный связный подграф.

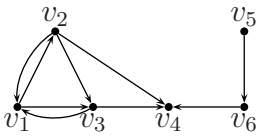
На примерах проиллюстрируем введенные понятия.

Пример 1.



$v_1v_3v_4$ – простая цепь. $v_1v_3v_2v_1v_3v_4$ – не простая цепь, т. к. есть совпадающие ребра. $v_3v_1v_2v_4$ – не цепь. $v_4v_3v_1v_2v_3v_4$ – цепь но не простая, т. к. есть совпадающие ребра. $v_1v_3v_2v_1$ – цикл. $v_4v_3v_1v_2v_3v_4$ – цепь с совпадающими концами, но не цикл.

Пример 2.



$v_1v_2v_3v_4$ – простой путь, $v_1v_2v_3v_1v_3v_4$ – не простой путь, $v_3v_1v_2v_3$ – контур, $v_3v_1v_2v_1v_3$ – замкнутый путь, но не контур, $v_1v_3v_4v_6$ – не путь.

Теорема. Для любой цепи, соединяющей две вершины графа, существует простая цепь, соединяющая те же вершины. (То же верно и для орграфа.)

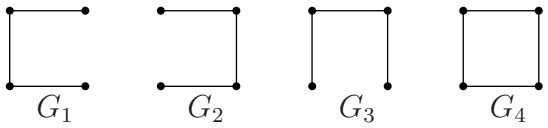
Граф $G_1 = (V_1, E_1)$ называют *подграфом* графа $G = (V, E)$, если $V_1 \subseteq V$, $E_1 \subseteq E$. Обозначают $G_1 \subseteq G$.

Если хотя бы одно из включений строгое, то подграф называют *собственным*. Если $V_1 = V$, то подграф называют *остовным*.

Подграф G_1 называют *порожденным множеством вершин* $V_1 \subset V$, если ребро принадлежит $E_1 \subseteq E$ только тогда, когда его концы принадлежат V_2.

Подграф $G_1 \subseteq G$ называют максимальным подграфом, *обладающим данным свойством* P, если он не является собственным подграфом никакого другого подграфа в G, обладающего свойством P.

Пример 3.



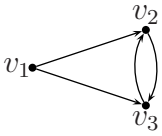
G_1, G_2, G_3 – максимальные ациклические подграфы в G_4.

Орграф называется *сильно связным*, если любые две вершины u и v достижимы друг из друга.

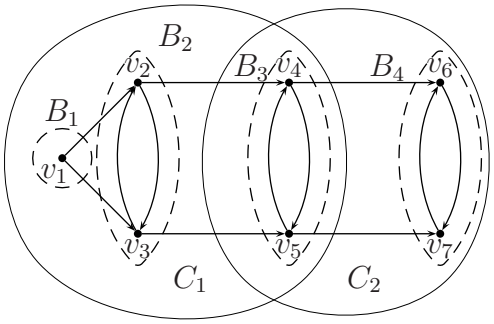
Бикомпонента орграфа – максимальный сильно связный подграф.

Две различные бикомпоненты не пересекаются.

Пример 4. Этот орграф связный.

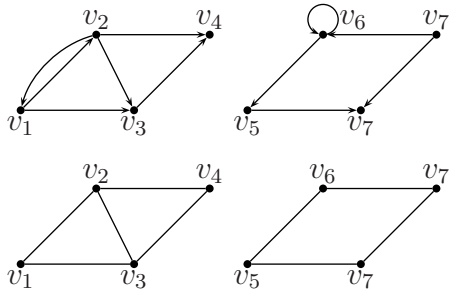


Пример 5. Пунктиром выделены бикомпоненты B_1, B_2, B_3, B_4. Непрерывной линией выделены компоненты связности C_1, C_2.



Граф $G_1 = (V_1, E_1)$ называют *ассоциированным с орграфом* $G = (V, E)$, если множество $V_1 = V$, пара $\{u, v\}$ образует ребро только тогда, когда есть дуга (u, v) или (v, u), при $u \neq v$.

Пример 6. Здесь показан несвязный орграф и граф, ассоциированный с ним.



Орграф называют *слабо связным*, если ассоциированный граф связен. *Компонентой слабой связности* орграфа называют максимальный слабо связный подграф.

§2. Способы представления графов

Допустим, что все вершины и ребра графа или все вершины и дуги (включая петли) орграфа пронумерованы, начиная с единицы. Граф может быть представлен матрицей $m \times n$, где n – число вершин, m – число ребер (дуг).

Матрицей инцидентности графа называется матрица $A = (a_{ij})$, элементы которой задаются равенствами

$$a_{ij} = \begin{cases} 1, & \text{для } i\text{-ой вершины } j\text{-е ребро инцидентно,} \\ 0, & \text{иначе.} \end{cases}$$

Матрицей инцидентности орграфа называется матрица $A = (a_{ij})$, элементы которой задаются равенствами

$$a_{ij} = \begin{cases} 1, & \text{для } i\text{-ой вершины } j\text{-я дуга выходящая,} \\ -1, & \text{для } i\text{-ой вершины } j\text{-я дуга исходящая,} \\ 0, & \text{иначе.} \end{cases}$$

Матрицей смежности графа называется матрица $B = (b_{ij})$, элементы которой определяются равенствами

$$b_{ij} = \begin{cases} 1, & \text{если } i\text{-я и } j\text{-я вершины смежны,} \\ 0, & \text{иначе.} \end{cases}$$

Матрицей смежности орграфа называется матрица $B = (b_{ij})$, элементы которой определяются равенствами

$$b_{ij} = \begin{cases} 1, & \text{если из } i\text{-ой вершины в } j\text{-ю ведет дуга,} \\ 0, & \text{иначе.} \end{cases}$$

Во многих задачах граф задается динамически, т. е. в ходе решения задачи меняется множество вершин и ребер. В этом случае эффективным способом машинного представления графа являются списки смежности.

Рассмотрим орграф. Для задания множества вершин, непосредственно достижимых из v, используют линейный однонаправленный список. Каждый элемент списка содержит число и указатель на следующий элемент списка. Последний элемент содержит пустой указатель.

Матрица достижимости – квадратная матрица, элементы которой определяются равенствами

$$c_{ij} = \begin{cases} 1, & \text{если из } i\text{-ой вершины достижима} j\text{-я}, \\ 0, & \text{иначе.} \end{cases}$$

С помощью матрицы достижимости можно отыскивать компоненты и бикомпоненты связности графа.

По определению в бикомпоненту входят взаимно достижимые вершины. Для двух таких вершин должны выполняться равенства $c_{ij} = c_{ji} = 1$. Поэтому, чтобы найти бикомпоненту, в которую входит i-я вершина орграфа, нужно просмотреть i-ю строку и i-й столбец матрицы достижимости C и сформировать множество $P_i = \{p | c_{ip} = c_{pi} = 1\}$ номеров вершин, порождающих искомую бикомпоненту. Из определения матрицы достижимости вытекает, что в P_i содержатся номера всех вершин данной бикомпоненты. Поскольку две две различные бикомпоненты не пересекаются, вершины с номерами из множества P_i при поиске других бикомпонент можно не рассматривать.

Поиск компонент графа сложнее, рассмотрим его на примере.

Пример. Для графа из примера 5 §1 составим матрицу достижимости.

$$C = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Начнем с поиска бикомпонент. Для первой вершины множество $P_1 = \{1\}$ включает только ее саму. Для второй вершины имеем $P_2 = \{2, 3\}$, для четвертой – $P_4 = \{4, 5\}$ и для шестой $P_6 = \{6, 7\}$. Полученные множества вершин порождают бикомпоненты B_1, B_2, B_3 и B_4, изображенные на рисунке примера 5 §1.

Перейдем к поиску компонент. Используя матрицу C, выпишем для каждой вершины v_i, $i = \overline{1,7}$ множество K_i, состоящее из тех вершин, которые достижимы из i-й или из которых достижима она. В рассматриваемом орграфе в множество K_1 входят вершины, для которых $c_{1j} = 1$ или $c_{j1} = 1$, т. е. $K_1 = \{v_1, v_2, v_3, v_4, v_5\}$. Для вершин v_2 и v_3 $K_2 = K_3 = K_1$. Поскольку все элементы четвертого и пятого столбцов матрицы C равны единице, то $K_4 = K_5 = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$. Для вершин v_6 и v_7 $K_6 = K_7 = \{v_4, v_5, v_6, v_7\}$.

Отметим, что, если некоторая компонента C_p содержит вершины v_i и v_j, то вершина v_m будет принадлежать этой компоненте только тогда, когда $v_m \in K_i \cap K_j$.

Действительно, пусть вершина v_m принадлежит компоненте C_p вместе с вершинами v_i и v_j. тогда либо вершина v_m достижима из вершины v_i, либо, наоборот, v_i достижима из v_m и, следовательно, $v_m \in K_i$. Из аналогичных соображений вытекает, что $v_m \in K_j$. Таким образом, вершина $v_m \in K_i \cap K_j$.

Пусть теперь $v_i, v_i \in C_p$ и $v_m \in K_i \cap K_j$. Тогда $v_m \in C_p$, поскольку либо v_i достижима из v_m, либо v_m достижима из v_i, и для вершин v_j и v_m ситуация аналогична.

Начнем строить компоненты, содержащие v_1. Рассмотрим множество K_1. Так как вершина v_2 принадлежит множеству K_1, найдется по крайней мере одна компонента, в которую входят обе эти вершины. Обозначим ее C_1. Вершина v_3 будет принадлежать этой компоненте только тогда, когда $v_3 \in K_1 \cap K_2$.

Можно заметить, что

$$K_1 \cap K_2 \cap K_3 \cap K_4 \cap K_5 = K_1,$$

поэтому компоненета C_1 есть подграф, порожденный множеством K_1, и вершина v_1 не может входить в какую-либо другую компоненту, отличную от C_1. Поскольку в компоненту C_1 вошли все вершины из множеств K_2 и K_3, то вершины v_2 и v_3 также не входят ни в какие другие компоненты.

Найдем компоненты, отличные от C_1, в которые входит вершина v_4. В множестве K_4 содержится вершина v_5. Пересечению множеств K_4 и K_5 принадлежат вершины v_6 и v_7, не вошедшие в выделенную компоненту C_1. Рассуждая аналогично и учитывая, что $v_4, v_5 \in B_3$, а $v_6, v_7 \in B_4$, получаем, что компонента C_2 порождается множеством вершин

$$K_4 \cap K_5 \cap K_6 \cap K_7 = K_4.$$

Других компонент нет.

§3. Деревья.

Деревом называют связный ациклический граф.

Ордеревом называют бесконтурный граф, у которого полустепень захода каждой вершины не больше 1 и существует одна вершина, у которой полустепень захода 0, называемая *корнем*.

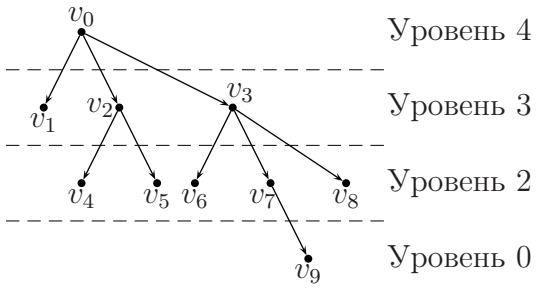
Вершину, не имеющую потомков называют листом.

Произвольный ациклический граф называют *лесом*. Если каждая слабая компонента орграфа является ордеревом, то граф называется орлесом.

Подграф дерева называется поддеревом.

Ордерево, у которого каждая вершина, отличная от корня является листом, называется кустом.

Высота ордерева – наибольшая длина пути из корня в лист, *глубина* $d(v)$ *вершины* ордерева – длина пути из корня в вершину. *Уровень вершины* – разность между высотой и глубиной.



Ордерево называется *бинарным*, если полустепень исхода любой вершины не больше 2, бинарное дерево называется *полным*, если из любой вершины, не являющейся листом, исходит ровно две дуги, а уровни всех листьев совпадают.

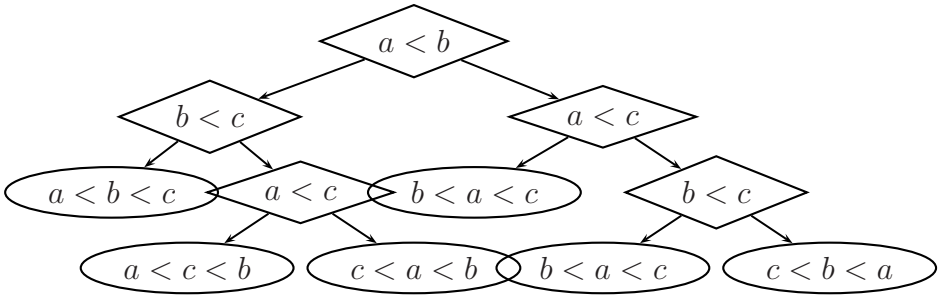
Индукцией по высоте h полного бинарного дерева можно доказать, что в нем 2^h листьев.

Теорема. Бинарное ордерево с n листьями имеет высоту не меньшую $\log_2 n$.

Пример. Задача сортировки. Требуется упорядочить множество

$$\{a_1, a_2, \ldots, a_n\}.$$

Ответом является одна из $n!$ возможных перестановок. Все стравнения, которые могут быть произведены в процессе работы изображаются в виде ордерева. В частности для множества $\{a, b, c\}$ ордерево изображено на рисунке. Движение по левой ветке означает выполнение проверяемого условия, по левой – невыполнение.



Сортируя последовательно, алгоритм пройдет от корня до листа. Число операций сравнения будет увеличиваться пропорционально высоте дерева не менее, чем $\log_2 n!$.

Остовным лесом (*деревом*) графа называют любой остовный подграф, являющийся деревом.

Теорема. У всякого графа существует максимальный остовный лес.

Остовное дерево наименьшего веса.

Граф, у которого каждому ребру (дуге) сопоставлено число, называемое *весом*, называют *взвешенным*.

Алгоритм Краскала вычисляет для данного взвешенного графа остовное минимального веса, т. е. с минимальной суммой ребер.

Алгоритм.

1. Множество ребер H искомого дерева полагаем пустым $H = \varnothing$.

2. Формируем множество $V_S = \{\{v_1\}, \ldots, \{v_n\}\}$, элементами которого является множество вершин исходного остовного леса. Каждая компонента леса состоит из одной вершины.

3. Сортируем множество ребер E исходного графа по возрастанию и формируем из них очередь Q.

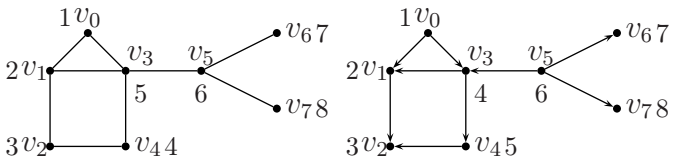
4. Если множество V_S содержит более одного элемента (т. е. остовный лес состоит из нескольких компонент) и очередь Q не пуста, переходим к п. 5, иначе к п. 7.

§4. Методы систематического обхода вершин графа

Необходимо уметь обходить вершины так, чтобы каждая вершина была отмечена один раз.

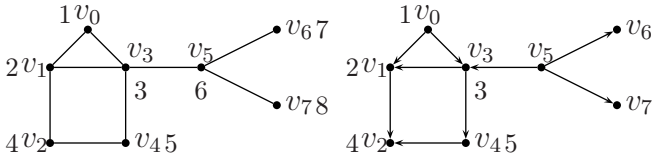
Один из способов обхода – *поиск в глубину*. Вершины нумеруются в том порядке, в котором они встретились. Отправляясь из вершины v_0, действуем так: пусть мы достигли вершину v, отмечаем ее и просматриваем смежные с ней вершины. Если среди смежных есть хотя бы одна, не отмеченная, вершина, продолжаем "путешествие" из первой такой вершины, действуем как сказано выше. Если неотмеченных вершин нет, то возвращаемся в предыдущую и смотрим ее список смежности. "Путешествие" прекратится в тот момент, когда мы вернемся в начальную вершину v_0 и либо все вершины отмечены, либо окажется, что неотмеченные вершины есть, но из v_0 дальше идти некуда. Тогда поиск ведут из неотмеченной вершины или останавливаются (зависит от алгоритма).

Пример.



Поиск в ширину. Здесь правила игры такие: достигнув некоторой вершины v, отмечаем ее. Затем просматриваем ее список смежности и отмечаем все ранее не отмеченные вершины списка. После того как отмечены все вершины из списка смежности вершины, считаем ее обработанной и продолжаем обработку вершин из списка смежности вершины v по указанной методике.

Пример.



§5. Упорядоченные множества

Множество M с заданным на нем отношением $\leqslant$ порядка называют *упорядоченным* и обозначают $(M, \leqslant)$.

Элементы $x, y \in (M, \leqslant)$ называют *сравнимыми*, если $x \leqslant y$ или $y \leqslant x$, иначе несравнимыми.

Элемент $a \in A$ называют *наибольшим* в A, если для любого $x \in A$ $x \leqslant a$.

Элемент $b \in A$ называют *максимальным,* если для любого $x \in A$ имеет место одно из двух: или $x \leqslant b$, или x и b не сравнимы.

Аналогично определяются наименьший и минимальный элементы.

Элемент $a \in A$ называют верхней (нижней) гранью множества B, если для всех $x \in B$ $x \leqslant a$ $(x \geq a)$.

Наименьшую (наибольшую) из верхних (нижних) граней множества A называют *точной верхней (нижней)* границей множества A и обозначают $\sup A$ ($\inf A$).

Последовательность $\{x_i\}_1^\infty$ называют *неубывающей*, если для любого $i \in \mathbb{N}$ выполняется неравенство $x_i \leqslant x_{i+1}$.

Множество $(M, \leqslant)$ называется *индуктивным*, если:

1) оно содержит наименьший элемент;

2) всякая неубывающая последовательность элементов имеет точную верхнюю грань.

Пусть $(M_1, \leqslant)$ и $(M_2, \preccurlyeq)$ – индуктивные множества. Отображение $f : M_1 \to M_2$ называют *непрерывным,* если для любой неубывающей последовательности $\{a_n\}_{n=0}^\infty$ элементов из M_1 выполняется равенство $f(\sup a_n) = \sup f(a_n)$. Отображение $f_1 : M_1 \to M_2$ упорядоченных множеств называют *монотонным*, если для любых $a, b \in M_1$ из $a \leqslant b$ следует $f(a) \leqslant f(b)$.

Теорема 1. Всякое непрерывное отображение одного индуктивного множества в другое монотонно.

Доказательство. Пусть $a \leqslant b$. Образуем последовательность, где $x_1 = a$, $x_2 = b$. Это неубывающая последовательность, $\sup x_n = b$. В силу непрерывности f выполняются равенства $f(b) = f(\sup x_n) = \sup f(x_n) = \sup\{f(a), f(b)\}$. Следовательно, $f(a) \leqslant f(b)$. Теорема доказана.

Элемент $a \in A$ называют *неподвижной точкой* отображения $f : A \to A$, если $f(a) = a$.

Элемент $a \in A$ называется *наименьшей неподвижной точкой*, если он является наименьшим элементом в множестве неподвижных точек отображения f.

Теорема 2. Любое непрерывное отображение f индуктивного упорядоченного множества $(M, \leqslant)$ в себя имеет неподвижную точку.

Этой точкой является $a = \sup\limits_{n \geq 1} f^n(\mathbb{O})$, где $\mathbb{O}$ – наименьший элемент в множестве M.

Полукольцо $\mathcal{S} = (S, +, \cdot, 0, 1)$ называется *замкнутым*, если

1) оно идемпотентно;

2) любая последовательность элементов множества S имеет sup относительно естественного порядка;

3) $\sup(aX) = a \sup(X)$, $\sup(Xa) = \sup(X)a$.

Теорема 3. Если A – конечное подмножество идемпотентного кольца, то $\sup A$ относительного естественного порядка равен сумме элементов из A.

В замкнутом полукольце обозначают $\sum\limits_{n=1}^{n} x_n = \sup\{x_n : n \in \mathbb{N}\}$.

Итерацией (*замыканием*) элемента x в замкнутом полукольце называют $x^* = \sup\limits_{n \geq 0}\{x_n\}$, где $x^0 = 1$. Другое обозначение $x^* = \sum\limits_{n=0}^{\infty} x^n$.

Пример. В полукольце $\mathcal{R}$ для любого элемента x итерация $x^* = 0$.

Теорема 4. Всякое замкнутое полукольцо является индуктивным упорядоченным множеством.

Можно доказать, что отображения $f(x) = ax + b$ и $g(x) = xa + b$, действующие в замкнутом полукольце являются непрерывными, поэтому в нем разрешимы уравнения:

$$x = ax + b, \tag{10}$$

$$x = xa + b. \tag{11}$$

Теорема 5. Наименьшими решениями уравнений (11) и (11) соответственно являются $x = a^*b$ и $x = ba^*$.

Доказательство. Относительно естественного порядка в полукольце наименьшим элементом является 0 – нейтральный по сложению. Согласно теореме 2 решение уравнения (11) $x = \sup\limits_{n\geq 0} f^n(0) = \sum\limits_{n=0}^{\infty} f^n(0)$. Пусть $f(x) = ax + b$, тогда $f^0(0) = 0$, $f^1(0) = b$, $f^2(0) = ab + b = (a + 1)b, \ldots,$ $f^n(0) = (a^{n-1} + \ldots + a^2 + a + 1)b$. Следовательно, $x = \sum\limits_{n=0}^{\infty} a^n b = (\sum\limits_{n=0}^{\infty} a^n)b = a^*b$. Теорема доказана.

§6. Задача о путях во взвешенных графах

Важными задачами анализа графов являются следующие:

1. Вычисление для заданного графа матрицы достижимости. Эту задачу называют задачей поиска транзитивного замыкания.

2. Вычисление наименьших расстояний между всеми парами вершин. Эту задачу называют задачей о кратчайших расстояниях. Расстоянием от вершины v до w по пути S называют сумму меток дуг пути. Эта задача не всегда имеет решение. Если в орграфе есть петли с отрицательной меткой, то, ходя по ней, путь можно делать сколь угодно коротким.

3. Перечисление всех путей между двумя произвольными вершинами.

Все эти задачи можно решать в рамках единого подхода.

Взвешенным орграфом называют пару $W = (G, \varphi)$, где $G = (V, E)$ – граф, $\varphi : E \to \mathcal{K}$ – весовая функция со значениями в идемпотентном кольце $\mathcal{K} = (K, +, \cdot, 0, 1)$. Пусть вершины графа пронумерованы, тогда взвешенный орграф может быть задан *матрицей весов* A с элементами $\varphi(i, j)$ – метки дуг (v_i, v_j). Вычисление итерации A^* этой матрицы позволяет решить все поставленные задачи. В случае выбора полукольца $\mathcal{B}$ получаем решение задачи о транзитивном замыкании, а в случае полукольца $\mathcal{R}$ – решение задачи о кратчайших расстояниях.

Рассмотрим решение общей задачи о путях для произвольного замкнутого полукольца $\mathcal{K}$.

Метка пути из вершины v_i и v_j – произведение в $\mathcal{K}$ меток, входящих в путь дуг в порядке их следования (для пути ненулевой длины) и метка равна 1 для пути нулевой длины.

Стоимость прохождения из вершины v_i в v_j – сумма в $\mathcal{K}$ меток всех путей из v_i в v_j.

Сумма, определяющая стоимость прохождения, есть бесконечная сумма, т. е. sup последовательности меток. Это связано с тем, что множество всех путей, ведущих из одной вершины в другую, бесконечно. Если стоимость прохождения между парой вершин равна нулю, то это означает, что такого пути нет.

Матрица меток дуг является элементом полукольца матриц над K. В этом полукольце определены операции сложения и умножения матриц обычным образом.

Лемма. Элемент $a_{ij}^{(l)}$ матрицы A^l, (A – матрица весов) равен стоимости прохождения из вершины v_i в v_j по всем путям длины l.

Доказательство. Индукция по l. При $l = 0$ утверждение очевидно, т. к. $A^0 = E$ (E – единичная матрица). При $l = 1$ утверждение также очевидно. Далее, $a_{ij}^{(l)} = \sum\limits_{k=1}^{n} a_{ik}^{(l-1)} a_{kj}$.

По предположению индукции $a_{ik}^{(l-1)}$ равен стоимости прохождения из v_i в v_k по всем путям длины $l-1$. Множество путей длины l из вершины v_i в v_j, проходящих через v_k так, что есть дуга из v_k в v_j образуется путем присоединения дуги (v_k, v_j) к каждому пути из v_i в v_k длиной $l-1$. Тогда $a_{ij}^{(l)}$ и есть нужная стоимость. Лемма доказана.

Так как стоимость прохождения между парой вершин (v_i, v_j) равна сумме меток всех путей из v_i в v_j, а эту сумму можно получить суммируя метки путей длины $0, 1, 2, \ldots$, то матрица стоимостей взвешенного орграфа может быть представлена в виде

$$A^* = A^0 + A^1 + \ldots + A^n + \ldots$$

Для вычисления A^* достаточно найти наименьшее решение в $\mathcal{K}$ при всех $j = \overline{1,n}$ системы уравнений

$$\xi = A\xi + \varepsilon_j,$$

где вектор $\varepsilon_j = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix}$ (1 стоит на j-м месте). Решением системы является $\xi = A^*\varepsilon_j$ – j-й столбец матрицы A^*.

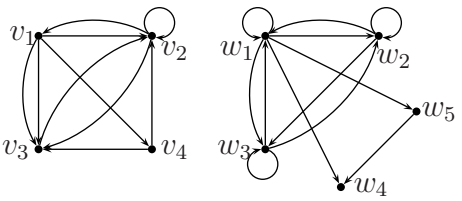
Выясним смысл матрицы A^* для полуколец $\mathcal{B}$ и $\mathcal{R}$. В полукольце $\mathcal{B}$ метка пути всегда равна 1. Следовательно, в A^* элемент $c_{ij} = 1$, если существует хотя бы один путь из вершины v_i в v_j и $c_{ij} = 0$, если пути нет, т. е. матрица стоимостей – матрица достижимости. В полукольце $\mathcal{R}$ метка пути – арифметическая сумма меток дуг. Так как сложение – выбор минимума в $\mathbb{R}^+$, то c_{ij} – наименьшая из меток путей из v_i в v_j.

§7. Морфизмы графов

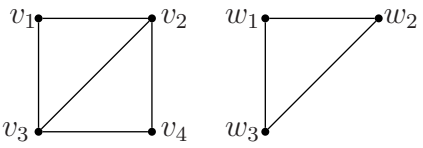
Отображение $h : V_1 \to V_2$ множество вершин графа $G_1 = (V_1, E_1)$ в множестве вершин графа $G_2 = (V_2, E_2)$ называют *гомоморфизмом* графа G_1 в G_2, если для любых двух вершин, смежных в G_1, их образы при отображении h смежны в G_2.

Гомоморфизм, при котором вершины смежны в G_1 только тогда, когда их образы смежны в G_2 называют *изоморфизмом*. Обозначают $G_1 \sim G_2$.

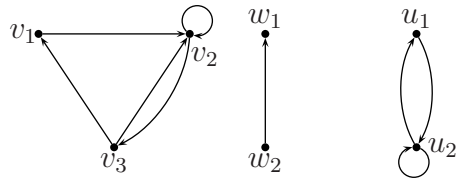
Пример 1. Отображение $h : h(v_1) = w_1,\ h(v_2) = w_2,\ h(v_3) = h(v_4) = w_3$ является гомоморфизмом следующих графов.



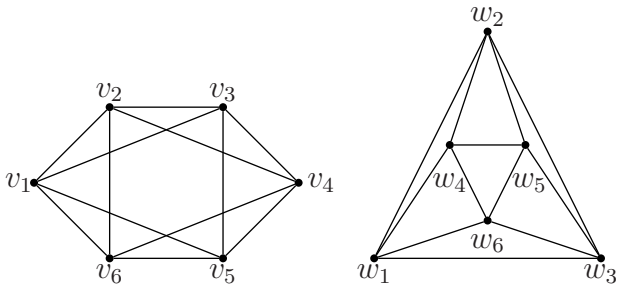
Пример 2. Отображение $h : h(v_1) = w_1,\ h(v_2) = w_2,\ h(v_3) = h(v_4) = w_3$ является гомоморфизмом следующих графов.



Пример 3. Отображение $h : h(v_1) = h(v_2) = w_1,\ h(v_3) = w_2$ не является гомоморфизмом. Отображение $g : g(v_1) = u_1,\ g(v_2) = g(v_3) = u_2$ – гомоморфизм.



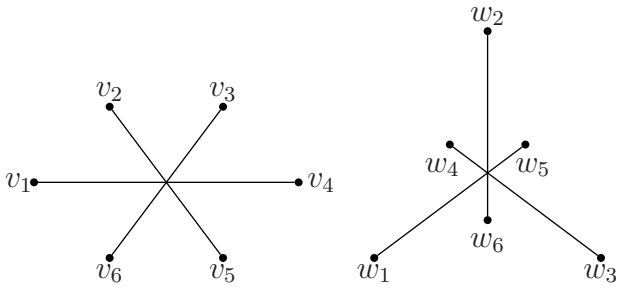
Пример 4. Данные графы изоморфны. Изоморфизм h задается формулой $h(v_i) = w_i$, $i = \overline{1,6}$.



Граф (орграф) $\overline{G} = (V, \overline{E})$ называют дополнением графа $G = (V, E)$, где $\overline{E}$ дополнение E до множества всех неупорядоченных (упорядоченных) пар на V.

Зачастую, чтобы распознать изоморфизм графов, удобно перейти от исходных графов к их дополнениям. Можно доказать, что графы изоморфны только тогда, когда изоморфны их дополнения.

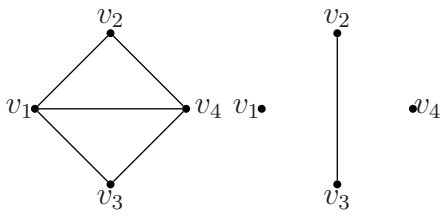
Пример 5. Перейдем к дополнениям графов из примера 4. Очевидно, что дополнения изоморфны.



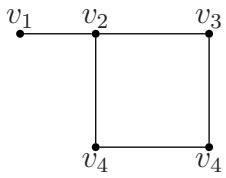
Автоморфизм графа $G = (V, E)$ – любая перестановка множества вершин, являющаяся изоморфизмом G на себя. Можно доказать, что композиция двух автоморфизмов – автоморфизм, а подстановка, обратная автоморфизму – автоморфизм.

Множество автоморфизмов образует группу, которая является подгруппой в группе всех перестановок из n элементов. Ее называют группой автоморфизмов.

Пример 6. Изображены два графа, являющихся дополнениями друг для друга. Группа их автоморфизмов порождается перестановками (14) и (23).



Пример 7. Группа автоморфизмов данного графа является циклической и порождается перестановкой (35).



§8. Топологическая сортировка

Сетью называют бесконтурный орграф.

Можно доказать, что в нем существуют вершины с нулевой полустепенью захода и вершины с нулевой полустепенью исхода. Первые называют *источниками*, вторые – *стоками*.

Уровень вершины сети – натуральное число, определенное следующим образом:

1) если полустепень захода равна 0, то ее уровень равен 0 и наоборот (уровень N_0 – множество всех входов);

2) если множество N_i вершин уровня i определены для всех $i \leqslant k$, то уровень N_{k+1} содержит только те вершины, предшественники которых принадлежат любому уровню от 0 до k, причем есть хотя бы один предшественник уровня k.

Уровень вершины можно интерпретировать как длину максимального пути от источников до вершины.

Порядковой функцией сети $G = (V, E)$ называют отображение, сопоставляющее каждой вершине сети номер ее уровня.

Проблема расположения вершин по уровням называется *топологической сортировкой.*

Топологическая сортировка производится по алгоритму Демукрона.

Перед началом работы алгоритма составляем матрицу смежности. Резервируем массив M, в котором элемент m_i – полустепень захода вершины v_i. Алгоритм выдает массив Ord длины n (число вершин), в котором i-й элемент – номер уровня вершины v_i.

Алгоритм.

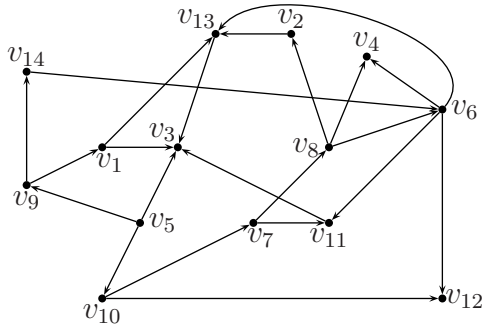
0. Сформировать множество V вершин сети. Значение счетчика уровней $k := 0$. Найти суммы элементов по всем столбцам матрицы смежности и заполняем массив M.

1. Если множество $V \neq \varnothing$, то на шаг 2, иначе, на шаг 3.

2. Определить множество I номеров всех новых нулевых элементов массива M, т. е. таких, что соответствующие этим номерам вершины принадлежат множеству V. Присвоить элементам массива Ord с номерами из множества I номер уровня k и удалить вершины с этими номерами из множества V. Вычесть из массива M строки с номерами из I. Увеличить счетчик уровней $k := k + 1$. Вернуться на шаг 1.

3. Закончить работу.

Пример.



Матрица смежности сети

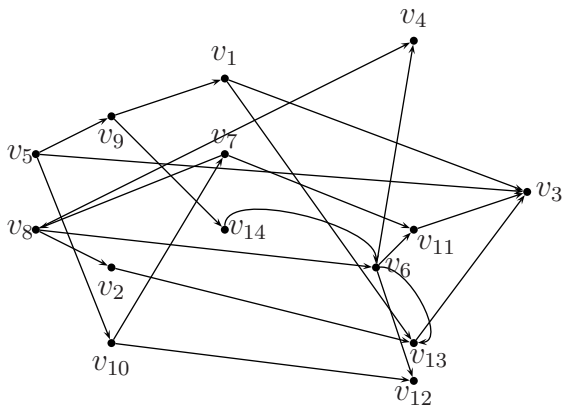
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	0	1	0	0	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	0	0	0	0	0	0	1	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	1	0	0	0	0	0	1	1	0	0	0	0
6	0	0	0	1	0	0	0	0	0	0	1	1	1	0
7	0	0	0	0	0	0	0	0	0	0	1	0	0	0
8	0	1	0	1	0	1	1	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0	0	0	0	0	1
10	1	0	0	0	0	0	1	0	0	0	0	1	0	0
11	0	0	1	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	1	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Далее приводится последовательность значений массива M и уровни N_i. Символом $*$

отмечаются вершины, удаленные из множества V.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	
2	1	5	2	0	2	2	0	1	1	2	2	3	1	$N_0 = \{5, 8\}$
2	0	4	1	$*$	1	1	$*$	0	0	2	2	3	1	$N_1 = \{2, 9, 10\}$
0	$*$	4	1	$*$	1	0	$*$	$*$	$*$	2	1	2	0	$N_2 = \{1, 7, 14\}$
$*$	$*$	3	1	$*$	0	$*$	$*$	$*$	$*$	1	1	1	$*$	$N_3 = \{6\}$
$*$	$*$	3	0	$*$	$*$	$*$	$*$	$*$	$*$	0	0	0	$*$	$N_4 = \{4, 11, 12, 13\}$
$*$	$*$	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$N_5 = \{3\}$

На следующем рисунке приводится данный граф, топологически отсортированный.



§9. Циклы и разрезы

Разрезом называется минимальное множество ребер, удаление которых превращает граф в несвязный.

Суммой графов $G_1 = (V_1, E_1)$ и $G_2 = (V_2, E_2)$ называется граф $G_1 \oplus G_2 = (V_1 \cup V_2, E_1 \triangle E_2)$, где $\triangle$ – симметрическая разность множеств.

Цикл называется *базисным*, если все его ребра, кроме одного являются остовными, а одно реброй – хордой.

Разрез называется *базисным*, если только одно его ребро остовное.

Теорема 1. Цикл и разрез связного графа имеют четное число общих ребер.

Доказательство. Пусть C – цикл, а S – разрез связного графа G, V_1 и V_2 – множества вершин двух связных подграфов $G_1 = (V_G, E_G)$ и $G_2 = (V_S, E_S)$ графа $G \setminus S = (V_G \cap \overline{V}_S, E_G \setminus E_S)$. Если C – подграф графа G_1 или G_2, то, очевидно, что число ребер, общих в C и S, равно нулю, т. е. четному числу.

Предположим, что C и S имеют несколько общих ребер. Будем передвигаться по циклу C из вершины $v_1 \in V_1$. Поскольку передвижение должно кончаться в вершине v_1, необходимо, чтобы каждый раз мы встречались с ребром из S, ведущим нас от вершины в V_1 к вершине в V_2, также должно существовать ребро в S, переводящее нас из вершины в V_2 к вершине в V_1. Это возможно только в случае, когда C и S имеют четное число общих ребер. Теорема доказана.

Рассмотрим граф $G = (V, E)$. Набор всех подмножеств в E, включая $\varnothing$, обозначим W_G. Множество W_G – абелева группа относительно операции $\oplus$. Для каждого $E_i \in W_G$

справедливы равенства $E_i \oplus \varnothing = E_i$, $E_i \oplus E_i = \varnothing$, т. е. $\varnothing$ – нейтральный, E_i – обратный самому себе. Определить умножение вектора на скаляр из $\mathbb{Z}_2$ можно так $1 \cdot E_i = E_i$, $0 \cdot E_i = \varnothing$. Таким образом, W_G – векторное пространство. Каждому элементу E из W_G, содержащему m ребер можно сопоставить m-мерный вектор с координатами из $\mathbb{Z}_2$, в котором i-я координата равна 1, если в E содержится i-е ребро, координата равна 0, если ребра нет. Множество таких векторов обозначим $\mathbb{Z}_2^m$. На множестве $\mathbb{Z}_2^m$ определим сумму векторов $\vec{a} = (a_1, \ldots, a_m)$ $\vec{b} = (b_1, \ldots, b_m)$ равенством $\vec{a} \oplus \vec{b} = (a_1 \oplus b_1, \ldots a_m \oplus b_m)$. Установленное соответствие между W_G и $\mathbb{Z}_2^m$ является изоморфизмом групп.

Обозначим W_C – множество всех циклов, $\varnothing$ и объединений реберно-непересекающихся циклов, W_S – множество всех разрезов, $\varnothing$ и объединений реберно-непересекающихся разрезов.

Теорема 2. W_C является подпространством в W_G.

Доказательство. Степень каждой вершины в цикле и в объединении реберно-непересекающихся циклов четная. Пусть $C_1, C_2 \in W_C$, $C_3 = C_1 \oplus C_2$. Установим, что каждая вершина в C_3 имеет четную степень. Рассмотрим вершину $v \in C_3$, обозначим X_i – множество ребер в C_i, инцидентных v, $|X_i|$ – количество ребер в X_i. Заметим, что $|X_i|$ – четные числа и одно из них может быть нулем. Следовательно, $|X_3| \neq 0$. Так как $C_3 = C_1 \oplus C_2$, то $X_3 = X_1 \triangle X_2$. Поэтому $|X_3| = |X_1| + |X_2| - 2|X_1 \cap X_2|$. Из этого равенства следует, что $|X_3|$ – четно. Следовательно, $C_3 \in W_G$. Теорема доказана.

Размерность подпространств циклов и разрезов

Рангом графа называется количество ребер в остове.

Цикломатическим числом называется количество хорд.

Покажем, что размерности подпространств W_C и W_S равны соответственно цикломатическому числу и рангу графа. Для этого докажем, что множество базисных циклов и базисных разрезов по отношению к некоторому остову являются базисами в W_C и W_S.

Пусть T – остов связног графа G на n вершинах и m ребрах. Ветви T обозначим b_1, $b_2, \ldots,$ b_{n-1}, а хорды c_1, $c_2, \ldots,$ c_{m-n+1}. Пусть C_i и S_i базисный цикл и базисный разрез по отношению к c_i и b_i. Каждый базисный цикл содержит одну хорду, не содержащуюся в других базисных циклах, т. е. базисный цикл не может быть представлен суммой других базисных циклов. Следовательно, циклы C_i независимы. Аналогично с разрезами S_i.

Чтобы доказать, что C_1, $C_2, \ldots,$ C_{m-n+1} (S_1, $S_2, \ldots,$ S_{n-1}) образуют базис подпространства циклов (разрезов) графа G нужно доказать, что любой подграф в подпростарнстве W_C (W_S) можно представить суммой. Рассмотрим подграф $C \in W_C$. Пусть он содержит хорды c_{i_1}, $c_{i_2}, \ldots,$ c_{i_r}. Пусть $C' = C_{i_1} \oplus C_{i_2} \oplus C_{i_r}$. Очевидно, что C' содержит только хорды c_{i_1}, $c_{i_2}, \ldots,$ c_{i_r} и не содержит других хорд остова T. Поскольку подграф C содержит только эти хорды, то $C' \oplus C$ не содержит ни одной хорды. Покажем, что $C' \oplus C = \varnothing$. Если это неверно, то по предыдущим рассуждениям $C' \oplus C$ содержит только ветви и, следовательно, не содержит цикла. С другой стороны $C' \oplus C$ должно быть циклом или объединением реберно-непересекающихся циклов. Получено противоречие, следовательно, $C = C_{i_1} \oplus C_{i_2} \oplus \ldots \oplus C_{i_r}$. Аналогично можно доказать, что каждый подграф можно представить суммой разрезов. Итак, имеет место

Теорема 3. Пусть G – связный граф на n вершинах и m ребрах. В этом случае:

1) Базисные циклы по отношению к остову графа G образуют базис пространства W_C, и, следовательно, размерность W_C равна $m - n + 1$, т. е. цикломатическому числу графа G.

2) Базисные разрезы по отношению к остову графа G образуют базис пространства W_S, и, следовательно, размерность подпространства разрезов графа равна $n - 1$, т. е. рангу графа G.

Было доказано, что цепи и разрезы имеют четное число ребер. Используя то, что каждый подграф в пространстве W_C является либо циклом, либо объединением реберно-непересекающихся циклов, а каждый подграф в пространстве W_S является либо разрезом, либо объединением реберно-непересекающихся разрезов получим теорему.

Теорема 4. Любой элемент в W_C имеет четное число общих ребер с любым элементом в W_S.

Верна и обратная теорема.

Ортогональность подпространств циклов и разрезов

Подпространства W_C и W_S изоморфны подпространствам в $\mathbb{Z}_2^m$, которые соответственно обозначаем W'_C, W'_S. Так как любые два графа $G_1 \in W_C$, $G_2 \in W_S$ имеют четное число ребер, то $w_1 \cdot w_2 = 0$, где $w_1 \in W'_C$, $w_2 \in W'_S$ – векторы, соответствующие G_1, G_2. Таким образом имеет место

Теорема. Подпространства циклов и разрезов ортогональны.

АВТОМАТЫ, ЯЗЫКИ, ГРАММАТИКИ

§1. Алфавит, слово, язык

Алфавит – конечное множество $V = \{a_1, a_2, \ldots, a_n\}$, элементы которого называются *буквами*.

Словом в алфавите V называют элемент множества

$$V^k = \underbrace{V \times V \times \ldots \times V}_{k}.$$

При $k = 0$ имеем множество V^0, состоящее из пустого упорядоченного набора элементов, обозначаемого λ.

Множество всех слов в алфавите V обозначают V^*. *Длиной* слова называют количество букв в нем.

Языком в алфавите V называется подмножество в V^*. Множество V^* называют *универсальным языком*.

Над языками можно производить теоретико-множественные операции объединения, пересечения, дополнения, вычитания.

Конкатенацией слов $x = x_1x_2\ldots x_k$, $y = y_1y_2\ldots y_m$, где x_1, $x_2, \ldots,$ x_k, y_1, $y_2, \ldots,$ $y_m \in V$ называют слово $xy = x_1x_2\ldots x_ky_1y_2\ldots y_m$.

Вхождение слова $x \in V^*$ в слово $y \in V^*$ – упорядоченная тройка слов (u, x, v) такая, что $y = uxv$.

Говорят, что вхождение (u, x, v) и (s, z, t) слов x и z в одно слово y не пересекаются, если существуют такие (возможно пустые) слова p и q, что $y = uxpzt$ или $y = szqxv$.

Вхождение x в y обозначают $x \leqslant y$.

Конкатенацией языков L_1 и L_2 называют язык L_1L_2, состоящий из слов xy, где $x \in L_1$, $y \in L_2$.

Пример. Пусть $V = \{a, b, c\}$, $L_1 = \{ab, bcc, cab\}$, $L_2 = \{ca, bcc\}$, тогда

$$L_1L_2 = \{abca, abbcc, bccca, bccbcc, cabca, cabbcc\},$$

$$L_2L_1 = \{caab, cabcc, cacab, bccbcc, bcccab\}.$$

Языки L_1 и L_2 можно записать в виде $L_1 = ab + bcc + cab$, $L_2 = ca + bcc$. Знаком $+$ обозначена операция объединения языков. Язык L_1L_2 можно вычислить так: $L_1L_2 = (ab + bcc + cab)(ca + bcc) = abca + abbcc + bccca + bccbcc + cabca + cabbcc$.

Степенью n языка L называется язык $L^n = L^{n-1}L$ при $n > 1$, $L^0 = \lambda$.

Итерацией языка L называют объединение всех его степеней

$$L^* = \bigcup_{n=0}^{\infty} L^n.$$

Позитивной итерацией называется язык $L^+ = \bigcup\limits_{n=1}^{\infty} L^n$.

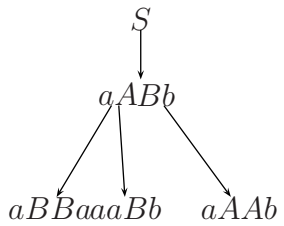
§2. Порождающие грамматики

Рассмотрим математическую модель синтаксиса, которая описывает механизмы порождения и распознавания "правильно" написанных слов. Будем рассматривать языки, которые могут быть бесконечными, но множество порождающих правил конечно. Такие языки называют *перечислимыми*.

Порождающая грамматика задается четверкой $G = (V, N, S, P)$, в которой V называется *терминальным* алфавитом, N – *нетерминальным* алфавитом, $V \cap N = \varnothing$, S – выделенный символ терминального алфавита, называемый *аксиомой*, P – конечное множество правил вывода. Правила вывода записывают в виде $\alpha \to \beta$ или $\alpha \vdash \beta$.

Слово δ *непосредственно выводимо* в грамматике G из слова γ, если существуют такие слова σ и ρ и такое правило $\alpha \to \beta$, что $\gamma = \sigma\alpha\beta$, $\delta = \sigma\beta\rho$. Обозначают $\gamma \vdash \delta$.

Пример. $G_0 = \{\{a, b\}, \{S, A, B\}, S, \{S \to aABb, aA \to aB, aB \to Ba, aA \to aa, aBb \to aabb, aBa \to bBb, Bb \to Ab\}\}$. Способ вывода слов $aBBb$, $aaBb$, $aAAb$ из аксиомы S показан на графе.



Слово δ *выводимо* из γ, если найдутся такие слова $\gamma = \alpha_0$, $\alpha_1, \ldots,$ α_n, где $n \geq 0$, что

$$\alpha_0 \vdash \alpha_1 \vdash \ldots \vdash \alpha_{n-1}\alpha_n.$$

Для данного слова существует много слово слов, выводимых из него непосредственно.

Выводом в грамматике G называют конечную или бесконечную последовательность $\alpha_0, \alpha_1, \ldots, \alpha_n \ldots,$ в которой для любого $i \geqslant 0$ $\alpha_i \vdash \alpha_{i+1}$.

Язык, порождаемый грамматикой G – множество $L(G)$ всех выводимых из аксиомы грамматики терминальных слов.

Две грамматики называются *эквивалентными*, если они порождают один язык.

В сокращенной форме правила $A \to \alpha_1$, $A \to \alpha_2, \ldots$ $A \to \alpha_n$ будем записывать в виде $A \to \alpha_1|\alpha_2| \vdash |\alpha_n$.

Пример 1.

$$G_1 = (\{\text{кот}, \text{пес}, \text{крокодил}, \text{мяукает}, \text{лает}, \text{плачет}\},$$
$$\{\langle\text{предложение}\rangle, \langle\text{подлежащее}\rangle, \langle\text{сказуемое}\rangle\},$$
$$\langle\text{предложение}\rangle, P_1).$$

Множество правил P_1 имеет вид:

$\langle$предложение$\rangle \to \langle$подлежащее$\rangle\langle$сказуемое$\rangle$,

$\langle$подлежащее$\rangle \to$ кот|пес|крокодил ,

$\langle$сказуемое$\rangle \to$ мяукает|лает|плачет .

Построим вывод:

$\langle$предложение$\rangle \vdash \langle$подлежащее$\rangle\langle$сказуемое$\rangle \vdash$ кот$\langle$сказуемое$\rangle \vdash$ кот плачет.

Грамматика – система "определений," "терминов," "понятий." Выделяется самое общее понятие (здесь "предложение"). Оно сводится к менее общим понятиям, до тех пор по не придем к конкретным оъектам. Самое общее понятие – аксиома, другие "понятия" – нетерминалы, конкретные объекты – терминалы. В подобном духе строится определение синтаксиса языков программирования.

Пример 2. Грамматика $G = (\{a, b, 0, 1\}, \{I, D\}, I, P_2)$, правила вывода P_2 : $I \to aD|bD|a|b$, $D \to aD|bD|0D|1D|a|b|0|1$. Эта грамматика порождает простейшие идентификаторы в алфавите из двух букв и двух цифр, т. е. слова, начинающиеся с буквы. Выводы в грамматике G_2 могут быть построены, например, так: $I \vdash aD \vdash abD \vdash ab0D \vdash ab0bD \vdash ab0b1$; $I \vdash a$, $I \vdash b$.

Пример 3. Грамматика $G_3 = (\{(,)\}, \{S\}, S, S \to ()|(S)|SS)$ порождает множество так называемых "правильных скобочных структур", например, $S \vdash SS \vdash (S)S \vdash (())S \vdash (())()$.

§3. Регулярные языки, грамматики и выражения

На множестве языков определим операции объединения и конкатенации, тем самым множество становится полукольцом, которое обозначают $\mathcal{L}(V)$. В полукольце $\mathcal{L}(V)$ рассмотрим подполукольцо, порожденное пустым языком, языком $\{\lambda\}$, всех языков $\{a\}$, $a \in V$ и замкнутое относительно итерации. Это подполукольцо обозначаем $\mathcal{R}(V)$, оно называется *полукольцом регулярных языков.*

Множество регулярных языков в алфавите $V = \{a_1, \ldots, a_n\}$ – замыкание конечного множества $\{\varnothing, \{\lambda\}, \{a_1\}, \ldots, \{a_n\}\}$ относительно операций объединения, конкатенации и итерации.

Можно описать это множество индуктивно:

1) Множества $\varnothing$ и $\{\lambda\}$ считаем регулярными языками в алфавите V;

2) если известно, что P и Q – регулярные языки в алфавите V, то $P \cup Q$ и $P \cap Q$ – регулярные языки;

3) если известно, что P – регулярный язык в алфавите V, то V^* – регулярный язык;

4) других регулярных языков, кроме описанных в п. 1–3.

Регулярной называется грамматика, в которой каждое правило вывода имеет вид $A \to aB$ или $A \to a$, где a или терминал или λ.

Алгебраические операции над регулярными языками можно представить регулярными выражениями. Каждое регулярное выражение представляет регулярный язык, причем языки $\varnothing$, $\{\lambda\}$, $\{a\}$ обозначают $\varnothing$, λ, a. Если выражение p обозначает язык P, а выражение q – язык Q, то $p + q$, pq, p^* обозначают множества $P \cup Q$, PQ, P^*. Для выражений $\alpha\alpha^*$, $\alpha^*\alpha$ используют обозначение α^+ – положительная итерация. Самый высокий приоритет среди операций у итерации, затем у конкатенации и, наконец, у объединения.

Регулярное выражение $a^* + (bc)^n$ означает множество, состоящее из слов вида a^n, $(bc)^n$, $a, b \in V$. Менее удобное обозначение $\{a\}^n \cup (\{b\} \cdot \{c\})^n$.

§4. Конечные автоматы

Одна из наиболее важных задача, решаемых в теории формальных языков следующая. Пусть задана порождающая грамматика G с терминальным алфавитом V и слово x в алфавите V. Спрашивается: принадлежит ли слово x языку, порождаемому грамматикой G? Эту задачу называют проблемой принадлежности для грамматики G.

Решение состоит в разработке распознающего алгоритма. В основе подобного рода алгоритмов лежит математическая модель языка, называемая распознающей моделью, или автоматом.

Автомат можно описать как устройство, состоящее из блока управления, входной ленты, читающей головки автомата и блока внутренней памяти. На ленте, не ограниченной справа и разделенной на ячейки, записываются слова в алфавите V. Буквы занимают ячейки без пропусков. Блок управления может в каждый момент времени находиться в одном из состояний множества Q, а головка установлена на одну ячейку ленты. Такт работы автомата состоит в том, что в зависимости от содержимого обозреваемой ячейки, состояния q, а также содержимого внутренней памяти автомат сдвигает головку влево или вправо на одну ячейку, либо оставляет ее на месте и переходит в новое состояние или остается в прежнем. В общем случае автомат может менять содержимое обозреваемой ячейки.

Конфигурация автомата определяется состоянием блока управления, содержимым обозреваемой ячейки и содержимым внутренней памяти. В общем случае автомат – недетерминированное устройство, т. е. для него из заданной конфигурации возможны переходы в несколько различных. Правила, по которым автомат переходит из одной конфигурации в другую, образуют систему команд автомата. Множество команд предполагают конечным.

Пусть на ленте написано слово $x \in V^*$. Допустим, что среди состояний блока управления выделено специальное состояние, называемое *начальным*, а также подмножество состояний, называемых *заключительными*.

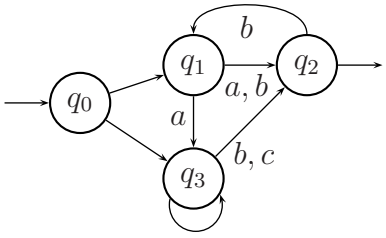
В начальный момент времени блок управления находится в начальном состоянии, головка обозревает первую ячейку ленты. Читая слово x и делая шаг за шагом, автомат, после того как он прочтет последнюю букву на ленте, окажется в состоянии q. Если это состояние заключительное, то говорят, что автомат допустил слово x.

Конечные автоматы – анализирующие модели для регулярных языков. Конечный автомат не имеет внутренней памяти, головка движется только направо. С ленты можно только читать. Кроме того автомат может спонтанно переходить из одного состояния в другое, не читая ленту и не передвигая головку. Такой такт можно рассматривать как переход из одного состояния в другое по пустому слову. Его называют λ-тактом. Команды автомата пишутся в виде $qa \to r$, что означает: из состояния q по символу $a \in V$ или по λ можно перейти в состояние r (возможно $q = r$). Для любой пары (q, r) состояний автомата имеет место следующее: если существует команда $qa \to r$ при $a = \lambda$, то нет ни одной команды для той же пары при $a \in V$ и наоборот.

Конфигурация конечного автомата определяется как упорядоченная пара (q, ay), где q – состояние блока управления, a – символ обозреваемой ячейки, y – слово входного алфавита правее a. Если $a = \lambda$, то непрочитанная часть считается пустой. Автоматы изображают графами. Вершины соответствуют состояниям, дуги определяются системой

командследующим образом: дуга ведет из состояния q в r только, если есть команда $qa \to r$.

Пример. Автомат задан системой команд: $q_0\lambda \to q_1$, $q_0\lambda \to q_2$, $q_1a \to q_2$, $q_1b \to q_2$, $q_1a \to q_3$, $q_2b \to q_1$, $q_3b \to q_2$, $q_3c \to q_2$, $q_3c \to q_3$.



Дано слово $abac$. Это допустимая цепочка, в силу последовательности конфигураций: $(q_0, abac)$, $(q_1, abac)$, (q_2, bac), (q_1, ac), (q_3, c), (q_3, λ). Ни одно слово, начинающееся с ca, не допускается автоматом.

Автомат называется *детерминированным*, если в нем нет дуг с меткой λ и из любого состояния ппо любому входному символу возможен переход в одно состояние.

Язык $L(M)$ конечного автомата M – множество всех слов во входном алфавите, читаемых в M из начального состояния в одно из конечных. $L(M)$ – *допустимый язык.*

Два автомата M_1 и M_2 называются *эквивалентными*, если $L(M_1) = L(M_2)$.

Теорема. Язык допускается конечным автоматом только тогда, когда он порождается регулярной грамматикой.

Для доказательства нужно:

1) указать способ построения регулярной грамматики G по автомату M такой, чтобы язык, порождаемый грамматикой, совпадал с $L(M)$;

2) указать способ построения автомата по грамматике.

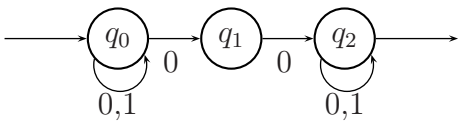
1. Построение грамматики по автомату. Пусть дан автомат

$$M = (V, Q, q_0, F, \delta),$$

где V – терминальный алфавит, Q – множество состояний, q_0 – начальное состояние, F – заключительные состояния, δ – функция переходов $\delta : Q \times (V \cup \{\lambda\}) \to 2^Q$ определена равенством $\{r : q \to_a r\}$, т. е., читая символ a, автомат из состояния q переходит в состояние r.

Определим регулярную грамматику $G = (\widetilde{V}, N, S, P)$, $\widetilde{V} = V$, $N \leftrightarrow Q$ – взаимнооднозначное соответствие, аксиома S сопоставляется начальному состоянию q_0. Множество правил вывода строится по системе команд так: правило вывода $S_q \to aS_r \in P$ только тогда, когда в δ есть команда $qa \to r$; правило вывода $S_q \to a \in P$ только тогда, когда в δ есть команда $qa \to r$, где $r \in F$; правило вывода $S_{q_0} \to \lambda \in P$ только тогда, когда в δ есть команда $q_0a \to r$, где $q_0 \in F$.

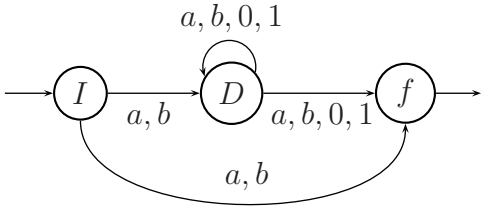
Пример. Для автомата



система команд такова: $q_0 \to q_0$, $q_0 0 \to q_1$, $q_0 1 \to q_0$, $q_1 0 \to q_2$, $q_2 0 \to q_2$, $q_2 1 \to q_2$. Грамматикой является $S_{q_0} \to 0S_{q_0}|1S_{q_0}|0S_{q_1}$; $S_{q_1} \to 0S_{q_2}$, $S_{q_2} \to 0S_{q_2}|1S_{q_2}|0|1$.

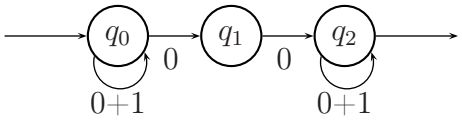
2. Построение автомата по грамматике. Дана грамматика $G = (V, N, S, P)$. Построим автомат $M = (\widetilde{V}, Q, q_0, F, \delta)$ так, чтобы $\widetilde{V} = V$, $Q = N \cup \{f\}$, f – некоторое состояние, не принадлежащее $V \cup N$, $q_0 = S$, множество заключительных состояний $F = \{f\}$. Система команд δ определяется следующим образом: команда $Aa \to B$ принадлежит δ только тогда, когда в множестве P правил вывода есть правило $A \to aB$; команда $Aa \to f$ принадлежит δ только тогда, когда правило $A \to a \in P$.

Пример. Дана грамматика $G = (\{a, b, 0, 1\}, \{I, D\}, I, P)$, правила вывода $I \to aD|bD|a|b$, $D \to aD|bD|0D|1D|a|b|0|1$. Тогда система команд автомата следующая: $Ia \to D$, $Ib \to D$, $Da \to D$, $Db \to D$, $D0 \to D$, $D1 \to D$, $Da \to f$, $Db \to f$, $D0 \to f$, $D1 \to f$, $Ia \to f$, $Ib \to f$.



Согласно общему определению метки пути в размеченном графе, метка пути в конечном автомате есть конкатенация меток входящих в этот путь дуг (в порядке их прохождения). Таким образом, метка любого пути конечной длины есть регулярный язык.

Пример. Для автомата



метка пути q_0, q_0, q_1, q_2 равна конкатенации языков

$$\{0, 1\} \cdot \{0\} \cdot \{0\} = \{000, 100\},$$

которую можно записать в виде регулярного выражения $(0 + 1)00$, а метка пути q_0, q_1, q_2, q_2, q_2 может быть задана таким регулярным выражением: $00(0 + 1)(0 + 1) = 00(00 + 01 + 10 + 11) = 0000 + 0001 + 0010 + 0011$. Если слово x принадлежит метке некоторого пути W, ведущего из вершины q в вершину r автомата M, то говорят, что слово x читается на пути W в M. В этом случае пишут $q \Rightarrow_x r$.

Стоимость прохождения из состояния q в r есть объединение меток всех путей из q в r. Это значит, что элемент c_{qr} матрицы стоимостей есть язык $c_{qr} = \{x : q \Rightarrow_x r\}$.

Рассмотрим задачу нахождения языка конечного автомата. Для этого достаточно вычислить матрицу стоимостей автомата, решая n систем вида

$$\xi_j = A\xi_j + \varepsilon_j,$$

где n – количество состояний, A – матрица меток дуг автомата, ε_j – j-й столбец единичной матрицы, $\varnothing$ – ноль полукольца, λ – единица полукольца. Нам требуется не вся матрица стоимостей, а лишь элементы c_{st}, где s – номер начального состояния, t –

номер заключительного. Поэтому вместо решения нескольких систем, достаточно решить одну

$$\xi = A\xi + \beta,$$

где β – столбец, все координаты которого равны $\varnothing$ (нулю полукольца), кроме координат с номерами $t_1, \ldots, t_m$, которые являются номерами заключительных состояний. Эти координаты равны λ (единица полукольца).

§5. Детермизация конечных автоматов

Теорема. Для любого автомата M может быть построен эквивалентный ему детерминированный автомат M'.

Алгоритм детерминимзации.

1. Удаление λ-переходов.

а) все состояния, кроме начального, в которые заходят только дуги с меткой λ удаляются.

б) множество дуг конечного автомата M' и их меток определяется так: для любых двух состояний $p, r \in Q'$ имеет место $p \to_a r$ только тогда, когда $a \in V$, а в графе M имеет место одно из двух

в) множество заключительных состояний F' конечного автомата M' содержит все состояния $q \in Q'$, т. е. состояния автомата M, не удаляемые могласно п. а), для которых имеет место $q \to^*_\lambda q_f$ для некоторого $q_f \in F$, т. е. либо состояние q само заключительное, либо из него ведет путь ненулевой длины по дунам с меткой λ в одно из заключительных состояний.

Собственно детерминимзация.

Пусть $M = (V, Q, q_0, F, \delta)$ – автомат без λ-переходов. Автомат M_1 определяется так, что его множество состояний – множество подмножеств состояний автомата M. Каждое отдельное состояние автомата M_1 определено как некоторое подмножество множества состояний автомата M. При этом начальным состоянием нового автомата является одноэлементное подмножество, содержащее начальное состояние старого автомата, а заключительными состояниями автомата M' являются такие подмножества в Q, которые содержат хотя бы одну заключительную вершину исходного автомата M. Функция переходов нового автомата определяется так, что из состояния-множества S по входному символу a конечный автомат M_1 переходит в состояние-множество, представляющее собой объединение всех множеств-состояний старого автомата, в которые этот старый автомат переходит по символу a из каждого состояния-множества S.

Среди состояний нового автомата есть состояние $\varnothing$. Значение функции переходов автомата M_1 $\delta_1(S, a) = \varnothing$ только тогда, когда $\forall q \in S$ значение функции переходов автомата M $\delta(q, a) = \varnothing$, т. е. в графе автомата M из каждого такого состояния q не выходит ни одна дуга, помеченная символом a.

Пример.

§6. Минимизация конечных автоматов

Для произвольного конечного автомата может быть построен эквивалентный автомат с минимальным числом состояний. Процедуру построения такого автомата

называют минимизацией автомата. Исходный автомат $M = (V, Q, q_0, F, \delta)$ полагаем детерминированным. Также полагаем, что в автомате нет состояний, не достижимых из начального.

На множестве состояний автомата зададим отношение эквивалентости следующим образом:

1. 0-эквивалентность. Для состояний q_1 и q_2 полагаем $q_1 \equiv^0 q_2$, если они оба заключительныые или оба не заключительные.

2. k-эквивалентность: при $k \geq 1$ полагаем $q_1 \equiv^k q_2$ толькко тогда, когда $q_1 \equiv^{k-1} q_2$ и для любого входного символа a состояния $\delta(q_1, a)$ и $\delta(q_2, a)$ также $k-1$ эквивалентны.

Пример. Минимизировать автомат

Состояния, являющиеся 0-эквивалентными, образуют множества $\{s_0, s_1, s_2\}$ и $\{s_3, s_4, s_5\}$. Состояния, являющиеся 1-эквивалентными, образуют множества $\{s_0, s_1\}$, $\{s_2\}$, $\{s_3, s_4, s_5\}$. Состояния, являющиеся 2-эквивалентными, образуют множества $\{s_0\}$, $\{s_1\}$, $\{s_3, s_4, s_5\}$.

Применение процедуры минимизации может дать два крайних случая:

1. все состояния исходного автомата окажутся эквивалентными и тогда в минимальном автомате окажется одно состояние.

2. итоговое разбиение будет состоять из одноэлементных классов эквивалентности, т. е. исходный автомат уже минимальный.

Если требуется выяснить, эквивалентны ли автоматы M_1 и M_2, то можно минимизировать каждый из них. Если минимальные автоматы M_1 и M_2 имеют множества состояний с разным числом вершин, то автоматы эквивалентны.

Можно доказать, что исходные автоматы эквивалентны только тогда, когда графы их минимальных автоматов изоморфны.

§7. Машина Тьюринга

Машина Тьюринга (A. Turing) определяется кортежем вида

$$T = (V, Q, \lambda, S, L, R, q_0, q_f, \delta),$$

где Q – конечное множество состояний, V – конечный входной алфавит, $\lambda \notin V$ – пустой символ (пробел), $S, L, R \notin V$ – символ направления движения читаюшей головки, $q_0 \in Q$ – начальное состояние, $q_f \in Q$ – заключительное состояние, δ – функция переходов, являющаяся отображением вида

$$\delta : Q \times (V \cup \{\lambda\}) \to 2^{Q \times (V \cup \{\lambda\}) \times \{S,L,R\}}.$$

Функция переходов записывается системой команд

$$qa \to rbM,$$

где $a, b \in V \cup \{\lambda\}$, $M \in \{S, L, R\}$.

Неформально работу машины Тьюринга можно описать так. Машина имеет устройство управления, которое может находиться в одном из состояний множества Q, бесконечную ленту, разделенную на ячейки, в каждой из которых может быть записан символ из алфавита $V \cup \{\lambda\}$, головку чтения-записи, которая в каждой момент

времени обозревает какую-то одну ячейку ленты. Команда $qa \to rbM$ разрешает машине Тьюринга, находящейся в состоянии q, читающей символ a, перейти в состояние r, вместо a записать на ленту символ b и сдвинуть головку в соответствии со значением M.

Понятие машины Тьюринга есть одно из математически точных определений алгоритма.

До построения математического понятия алгоритма мы имеем дело с интуитивным представлением об алгоритме. Это интуитивное понятие может быть охарактеризовано следующими признаками:

1. Детерминированность. Алгоритм – пошагово реализуемая процедура, на каждом шаге которой однозначно определено ее продолжение.

2. Результативность. Работа завершается через конечное число шагов.

3. Массовость. Алгоритм должен быть применим к достаточно широкому множеству входных данных.

Наряду с машиной Тьюринга известны и другие уточнения понятия алгоритма (нормальный алгоритм Маркова, рекурсивные функции и т. п.). Все эти понятия эквивалентны. Кроме того, в теории алгоритмов, которая занимается проблемами построения различных уточнений понятия алгоритма и вычислимости, а также сравнением между собой разных уточнений, основной гипотезой является гипотеза о том, что всякая функция вычислимая в интуитивном смысле, будет вычислима по Тьюрингу. Эта гипотеза называется *тезисом Тьюринга*. В пользу тезиса Тьюринга говорит то, что до сих пор не удалось придумать вычислимую функцию, которую нельзя было бы запрограммировать машиной Тьюринга, а также то, что все известные альтернативные уточнения понятия алгоритма можно свести к машине Тьюринга.

Пример. Машина с алфавитом $V = \{1, \lambda\}$, состояниями $\{q_0, q_1\}$ и командами $q_1 1 \to q_1 1R$, $q_1\lambda \to q_1 1R$ из любой начальной конфигурации будет работать бесконечно, заполняя единицами всю ленту направо.

Композиция машин Тьюринга. Функция f называется вычислимой по Тьюрингу, если существует машина Тьюринга, которая ее вычисляет.

Теорема. Если f_1 и f_2 вычислимы по Тьюрингу, то их композиция $f_2 \circ f_1$ также вычислима по Тьюрингу.

Пусть T_1 – машина, вычисляющая f_1, T_2 – машина, вычисляющая f_2. Множества их состояний $Q_1 = \{q_{10}, q_{11}, \ldots, q_{1n_1}\}$, $Q_2 = \{q_{20}, q_{21}, \ldots, q_{2n_2}\}$. Отождествим состояние q_{21} с q_{10}. Начальным состоянием новой машины T объявим состояние q_{11}, а заключительным – q_{20}. Машину T называют *композицией* машин T_1 и T_2.

Теорема. Всякая функция, вычислимая машиной Тьюринга, вычислима на машине с правой полулентой.

Например, так: всякий раз когда головке прежней машины нужно зайти за левый край ленты, новая машина предварительно сдвигает все написанное на клетку вправо.

Как уже говорилось, словесное описание алгоритма может быть неточным и нуждается в формализации. Поскольку в качестве формализации рассматривают машину Тьюринга, то естественно поставить задачу построения машины Тьюринга U, реализующей алгоритм выполнения программы для машины Тьюринга T. Для машины Тьюринга функции одной переменной формулировка задачи такова: построить машину U, вычисляющую функцию двух переменных и такую, что для любой машины T с системой

команд Σ_T $U(\Sigma_T, \alpha) = T(alpha)$, если $T(\alpha)$ определена (т. е. останавливается при данных α.) Проблема, возникающая при построении этой машины в том, что она должна иметь фиксированный алфавит V_U и множество состояний Q_U. Команды и данные машины T нельзя просто переписать на ленту машины U. Всегда найдутся машина T, алфавиты V_T и Q_T которой содержат больше символов, V_U и Q_U. Выход состоит в том, чтобы символы алфавитов V_T и Q_T кодировать символами в алфавите V_U и Q_U. Будем всегда считать, что $a_1 = 1$, $a_{m_T} = \lambda$ (эти сиволы есть в алфавите любой машины, работающей с числами). Обозначим коды q_i и a_j через $S(q_i)$ и $S(a_j)$ и определим их так: $S(\lambda) = \lambda^{m_T}$, $S(a_j) = a\lambda^{m_T - i - 1}$, для заключительного состояния q_0 $S(q_0) = q\lambda^{n_T - 1}$, $S(q_i) = q\lambda^{n_T - i - 1}1^i$. Код $S(a_j)$ имеет длину m_T, а код $S(q_i)$ – формат n_T. $S(R) = R$, $S(L) = L$, $S(L) = L$, $S(\rightarrow) = \Longrightarrow$.

План построения машины U таков. Будем считать, что машина T всегда работает на правой полуленте. Тогда ленту машины U можно разделить на двве бесконечные полуленты с границей Z между ними. В правой полуленте пишется текущая конфигурация машины T, а в левой полуленте записан код системы команд Σ_T.

Например, начальной конфигурации машины T из примера выше с исходным словом 11 соответствует следующее слово на ленте машины U :

$$q1a1 \rightarrow q1a1Rq1\lambda\lambda \rightarrow q1a1RZq1a1a1.$$

Работа машины U состоит в следующем.

1. Для слова $S(q_i, a_j)$ на правой полуленте выяснить, имеется ли на левой полуленте слово $S(q_i, a_j) \rightarrow$. Если да, то перейти к шагу 2, если нет, то к шагу 10.

2. В найденном слове $S(q_i, a_j)$ заменить $\rightarrow$ на $*$.

3. Выяснить, равен ли R ближайший символ движения справа от $*$. Если да, то переход к шагу 4, если нет, то – к шагу 1.

4. В m_T ячеек правой полуленты, начинающихся с q, записать слово длины m_T, начинающееся с $n_T + 1$-го символа правее $*$. Это слово $S(a'_j)$ и является кодом символа, который машина T печатет по команде $q_i a_j \rightarrow q'_i a'_j R$) После записанного слова напечатать $||$.

5. В n_T ячеек правой полуленты, начинающихся с $||$, записать слово длины n_T. (Это слово $S(q'_i)$ и является кодом состояния, в которое машина переходит по команде $q_i a_j \rightarrow a'_i a'_j R$.)

6. Заменить в правой полуленте $*$ на $\rightarrow$ и перейти к шагу 1.

7. Слово длины m_T на правой полуленте, расположенное левее q, сдвинуть вправо на n_T клеток. Поле сдвинутого слова напечатать метку $||$.

8. В m_T ячеек правой полуленты, начинающихся с $||$, записать слово длины m_T, начинающееся с $n_T + 1$ символа правее $*$. (Это слово $S(a'_j)$.) В клетке, расположенной на $m_T + n_T$ клеток левее начала записанного слова, напечатать $||$.

9. В n_T ячеек правой полуленты, начинающихся с $||$ записать слово длины n_T, начинающееся справа от $*$ (Это слово $S(q'_i)$.) В результате шагов 7, 8, 9 слово $S(a_k q_i a_j)$ на правой полуленте заменяется словом $S(q'_i a_k a'_j)$, что имитирует команду $q_i a_j \rightarrow q'_i a'_j L$.

10. Стереть в правой полуленте код состояния (это состояние будет заключительным для машины T, так как переход к шагу 10 означает, что состояние машины T не встретилось в левых частях команд).

Существование универсальной свшины Тьюринга означает, что систему команд Σ_T можно интерпретировать двояко: либо как описание работы конкретного устройства, либо как программу для машины U. Любой алгоритм можно реализовать либо аппратно – построением схемы, либо программно.

Проблема остановки машины Тьюринга. Алгоримт должен быть результативным, т. е. по любому алгоритму A и данным α можно определить, приведет ли работа алгоритма A к результату. Иначе говоря, нужно построить алгоритм B такой, что $B(A, \alpha) = 1$, если $A(\alpha)$ дает результат и $B(A, \alpha) = 0$, если – нет. В силу тезиса Тьюринга эту задачу можно сформулировать как задачу о построении машины Тьюринга: построить машину T_0 такую, что для любой машины Тьюринга T и любых исходных данных α $T_0(\Sigma_T, \alpha) = 1$, если машина $T(\alpha)$ останавливается и $T_0(\Sigma_T, \alpha) = 0$, если не останавливается. Эта задача называется проблемой остановки.

Теорема. Не существует машины Тьюринга, решающей проблему остановки для произвольной мшины T.

Эта теорема утверждает, что не существует единого алгоритма отладки программ. В каждо конкретном случае вопрос решается индивидуально.