

Комбинаторика и теория графов

Посов И.А.

Осень 2021 г.

Запись конспекта: Спиридонов А.

Содержание

1	Бинарные отношения	3
1.1	Свойства бинарных отношений	4
1.2	Отношения порядка	7
1.3	Топологическая сортировка	8
1.4	Транзитивное замыкание	9
2	Графы	11
2.1	Связность графа	13
2.2	Количество рёбер, вершин	15
2.3	Планарные графы	17
2.4	Хроматизм	19
2.5	Хроматические многочлены	23
2.6	Эйлеровы графы	26
2.7	Гамильтоновы графы	28
2.8	Длины путей в графах	29
2.9	Алгоритм Беллмана - Форда	33
2.10	Алгоритм Дейкстры	36
2.11	Алгоритм Флойда	39
2.12	Потоки в сетях	42
2.13	Теорема Форда — Фалкерсона	47
2.14	Задача о паросочетаниях	53
2.15	Задача о максимальном контролирующем множестве	56
2.16	Поиск в глубину, в ширину	59
2.17	Компоненты сильной связности	67

1 Бинарные отношения

Определение. M — множество $\neq \emptyset$.

$R \subset M \times M$ — бинарное отношение.

Пояснение

$M \times M$ — множество пар из элементов R .

Допустим, $M = \{a, b, c\}$. Тогда $M \times M = \{(a, a), (a, b), (a, c), (b, a), (b, b), (b, c), (c, a), (c, b), (c, c)\}$

Или $M = \mathbb{N}$

$M \times M = \mathbb{N} \times \mathbb{N} = \{(1, 1), (1, 2), \dots, (42, 17), \dots\}$

Отношение R — это подмножество пар.

Обозначение.

$(x, y) \in R$ — пара (x, y) принадлежит отношению. Вместо $(x, y) \in R$ будем писать xRy .

Вместо $(x, y) \notin R$ будем писать $x \not R y$

Примеры:

1. $M = \mathbb{R}$; $> = R = \{(x, y) : x > y\}$

$(3, 2) \in R \Leftrightarrow 3R2 \Leftrightarrow 3 > 2$

$(3, 4) \notin R \Leftrightarrow 3 \not R 2 \Leftrightarrow 3 \not > 2$

2. $M = \mathbb{R}$ отношение \geq : $7 \geq 6$; $7 \geq 7$; $7 \not \geq 8$

3. $M = \mathbb{R}$ отношение $=$: $7 = 7$; $7 \not = 8$

4. $M = \mathbb{R}$ \approx : $x \approx y \Leftrightarrow |x - y| < 1$

5. $M = \mathbb{R}$ отношение $@$: $x @ y \Leftrightarrow x^2 > y$. $2 @ 2$; $1 \not @ 2$

6. $M = \mathbb{N}$ отношение $:$: $x : y \Leftrightarrow \exists k \in \mathbb{R} : x = ky$. $4 : 2$; $10 : 3$

6.1. $M = \mathbb{Z}$ отношение \vdots : $x \vdots y \Leftrightarrow \exists k \in \mathbb{R} : x = ky$. $7 \vdots 0$; $0 \vdots 0$

7. $M = \mathbb{Z}$ отношение $\equiv \pmod 3$: $0 \equiv 3 \pmod 3$; $1 \not \equiv 8 \pmod 3$

8. $M = \mathbb{N}$ отношение $a \xi b$, если в числе 'a' 'b' цифр: $100 \xi 3$; $238 \not \xi 8$

9. $M =$ прямые на \mathbb{R}^2



отношение \parallel : l_1 параллельно l_2 , если l_1 не пересекает l_2 , или $l_1 = l_2$

10. $M =$ студент ЛЭТИ. $x \succ y$, если средний балл за последнюю сессию x больше чем y

11. $M =$ Пользователи ВКонтакте. $x \rightarrow y$, если пользователь x находится в друзьях у пользователя y

1.1 Свойства бинарных отношений

Определение Бинарное отношение R на множестве M называют *рефлексивным*, если $\forall x \in M : xRx \ ((x, x) \in R)$

Замечание. Отношение не рефлексивно $\Leftrightarrow \exists x : x \not R x$ — контрпример.

Примеры:

$=$: $\forall x : x = x$

\geq : $\forall x : x \geq x$

\approx : $\forall x : x \approx x (|x - x| = 0 < 1)$

$>$: $(2 \not R 2)$

ξ : не рефлексивно $(3 \not R 3)$

Определение Бинарное отношение R на множестве M называют *антирефлексивным*, если $\forall x \in M : x \not R x$

Замечание. Отношение не антирефлексивно $\Leftrightarrow \exists x : xRx$ — контрпример.

Примеры:

$>$: антирефлексивно $\forall x : x \not R x$

ξ : не антирефлексивно $(1 \xi 1)$

Замечание:

ξ не рефлексивно и не антирефлексивно.

Не бывает R , которое одновременно и рефлексивно, и антирефлексивно.

Рассмотрима $\in M \begin{cases} aRa \Rightarrow \text{не антирефлексивно} \\ a \not R a \Rightarrow \text{не рефлексивно} \end{cases}$

Определение Бинарное отношение R на множестве M называют *симметричным*, если $\forall x, y \in M : xRy = yRx$

Замечание. Отношение не симметрично $\Leftrightarrow \exists x, y : xRy, y \not R x$ — контрпример.

Примеры:

$=$: — симметрично $x = y \Leftrightarrow y = x$

\approx : — симметрично $\forall x, y$: если $|x - y| < 1 \Rightarrow |y - x| < 1$

ξ : — не симметрично $100 \xi 3, 3 \not \xi 100$

Определение Бинарное отношение R на множестве M называют *антисимметричным*, если $\forall x \neq y \in M : xRy \Rightarrow y \not R x$

Замечание. Отношение не антисимметрично $\Leftrightarrow \exists x \not\asymp y : xRy, yRx$ — контрпример.

Примеры:

$>$: — антисимметрично $x > y \Rightarrow y \not> x$

Попробуем построить контрпример:

$> : x \not\asymp y : x > y, y > x$ — невозможно

\Rightarrow нет контрпримера \Rightarrow отношение антисимметрично

$\geq : x \not\asymp y : x \geq y, y \geq x$ — невозможно

\Rightarrow нет контрпримера \Rightarrow отношение антисимметрично

$= : x \not\asymp y : x = y, y = x$ — невозможно

\Rightarrow нет контрпримера \Rightarrow отношение антисимметрично

$\equiv \text{mod } 3 : 1 \equiv 4 \text{ mod } 3$ — контрпример \Rightarrow отношение не антисимметрично

Над \mathbb{N} отношение $\dot{:} : x \not\asymp y : x \dot{:} y, y \dot{:} x$ — невозможно

\Rightarrow нет контрпримера \Rightarrow отношение антисимметрично

Над \mathbb{Z} отношение $\dot{:} : 4 \not\asymp -4 : 4 \dot{:} -4, -4 \dot{:} 4$ — контрпример

\Rightarrow отношение не антисимметрично

Определение Бинарное отношение R на множестве M называют *асимметричным*, если $\forall x, y \in M : xRy \Rightarrow y \not R x$

Замечание. Отношение не асимметрично $\Leftrightarrow \exists x, y : xRy, yRx$ — контрпример.

Эквивалентное определение Бинарное отношение R на множестве M называют *асимметричным*, если R — антисимметрично и антирефлексивно.

Примеры:

$>$: — асимметрично $x > y \Rightarrow y \not> x$

■ : — асимметрично (пустое отношение, когда $R = \emptyset$)

"Начальник" на множестве тех, кто работает в ЛЭТИ: отношение асимметрично
 x начальник $y \Rightarrow y$ не начальник x

Определение Бинарное отношение R на множестве M называют *транзитивным*, если $\forall x, y, z \in M : xRy, yRz \Rightarrow xRz$

Замечание. Отношение не транзитивно $\Leftrightarrow \exists x, y, z : xRy, yRz, x \not R z$ — контрпример.

Примеры:

$>$: — транзитивно $x > y, y > z \Rightarrow x > z$

\geq : — транзитивно $x \geq y, y \geq z \Rightarrow x \geq z$

\vdash : — транзитивно $x:y, y:z \Rightarrow x:z \Leftrightarrow (x = ky; y = lz \Rightarrow x = k(lz) \Rightarrow x:z)$

ξ : — не транзитивно $100\xi 3, 3\xi 1, 100\xi 1$

Определение Бинарное отношение R на множестве M называют отношением эквивалентности, если отношение R рефлексивно, симметрично и транзитивно.

Замечание Вместо R — эквивалентно, нужно говорить R — *отношение эквивалентности*.

Пример 1:

$=$: — отношение эквивалентности

$\forall x : x = x$ (Рефлексивность)

$\forall x, y : x = y \Rightarrow y = x$ (Симметричность)

$\forall x, y, z : x = y, y = z \Rightarrow x = z$ (Транзитивность)

\geq : — не отношение эквивалентности

$x \geq y \not\Rightarrow y \geq x (4 \geq 2; 2 \not\geq 4)$

Пример 2:

π (одинаковое количество цифр в числе): — отношение эквивалентности

$\forall x : x\pi x$ (Рефлексивность)

$\forall x, y : x\pi y \Rightarrow y\pi x$ (Симметричность)

$\forall x, y, z : x\pi y, y\pi z \Rightarrow x\pi z$ (Транзитивность)

Определение R — отношение эквивалентности на множестве M , $x \in M$ — класс эквивалентности $x : M_x = \{y | xRy\}$

Примеры:

$= : M_5 = \{5\}$

$\equiv \text{mod } 3 : M_2 = \{2, 5, 8, \dots\} = M_5$

Утверждение R — отношение эквивалентности на множестве M , $\forall x, y \in M : M_x = M_y$ или $M_x \cap M_y = \emptyset$

Доказательство

$\square M_x \cap M_y \neq \emptyset \Rightarrow \exists z \in M_x, z \in M_y \Rightarrow xRz, yRz \Rightarrow$ (Симметричность) $zRy \Rightarrow$ (Транзитивность) xRy

Теперь проверим, что класс $M_x = M_y$. Возьмём $u \in M_x$ и проверим $u \in (?)M_y. u \in M_x \Rightarrow xRu; xRy \Rightarrow$ (Симметричность) $yRu \Rightarrow$

\Rightarrow (Транзитивность) $yRu \Rightarrow u \in M_y$ ■

Следствие R — отношение эквивалентности на множестве M , тогда M разбито на несколько классов элементов (*классов эквивалентности*)

$$M = M_1 \cup \dots \cup M_n$$

$$M_i \cap M_j = \emptyset$$

Примеры:

= на \mathbb{N} :

$$\mathbb{N} = \{1\} \cup \{2\} \cup \{3\} \cup \dots$$

$$\equiv \text{mod } 3 \text{ на } \mathbb{N} : \mathbb{N} = \{0 \ 3 \ 6 \ 9 \ \dots\} \cup \{1 \ 4 \ 7 \ 10 \ \dots\} \cup \{2 \ 5 \ 8 \ 11 \ \dots\}$$

Замечание Если есть $M \neq \emptyset$ разбито на $M_i \neq \emptyset$:

$$M = M_1 \cup \dots \cup M_n; M_i \cap M_j = \emptyset$$

Тогда можно ввести отношение $R: xRy$, если $\exists M_i : x, y \in M_i$

1.2 Отношения порядка

Определение \sqsupset бинарное отношение R транзитивно и антисимметрично:

1) рефлексивно - нестрогий порядок (\succeq)

2) антирефлексивно - строгий порядок (\succ)

Примеры:

$>$ на \mathbb{R} — строгий порядок

\geq на \mathbb{R} — нестрогий порядок

\vdots на \mathbb{N} — нестрогий порядок

Определение $\sqsupset R$ — строгий или нестрогий порядок:

R — линейный порядок, если $\forall x \neq y : xRy$ или yRx

R — частичный порядок, если $\exists x \neq y : \cancel{xRy}, \cancel{yRx}$

Примеры:

$>, \geq$ — линейный порядок

\vdots на \mathbb{N} — частичный порядок ($2 \not\prec 3, 3 \not\prec 2$)

Утверждение R — строгий или нестрогий порядок на конечном множестве $M (|M| < \infty)$. Тогда $\exists x$ — минимальный, то есть $\forall y : x \not\succeq y$

Примеры:

\geq на $\{1, 2, 3, 4, 5\}$:

Минимальный - 1, так как $\forall y \neq 1 : 1 \not\succeq y$

\vdots на $\{2, 3, 4, 5, 6\}$:

Минимальный - 2, так как $\forall y \neq 2 : 2 \succ y$

Минимальный - 3, так как $\forall y \neq 3 : 3 \succ y$

Минимальный - 5, так как $\forall y \neq 5 : 5 \succ y$

Доказательство

Берём x_1 - любой элемент множества.

Если он не минимальный $\Rightarrow \exists x_2 \neq x_1 : x_1 \succ x_2$

Если x_2 не минимальный $\Rightarrow \exists x_3 \neq x_2 : x_2 \succ x_3$

...

Если мы не можем найти минимальный элемент, поскольку множество конечно \Rightarrow в какой-то момент элемент повторится ($x_i = x_j$)

$x_i \succ x_{i+1} \succ x_{i+2} \succ \dots \succ x_{j-1} \succ x_j = x_i$

\succ — транзитивно $\Rightarrow x_i \succ x_{j-1}, x_{j-1} (\neq x_i) \succ x_i$ (невозможно по антисимметричности)

Противоречие: следовательно, такое невозможно и рано ли поздно мы найдём минимальный элемент ■

Определение Отношение R_1 на множестве M расширяет R_2 на множестве M , если $R_2 \in R_1$

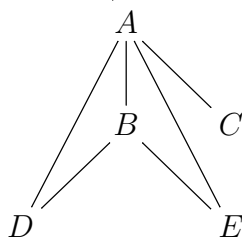
Замечание R_1 "добавляет" пары, где xR_2y

Замечание $xR_2y \Rightarrow xR_1y$

1.3 Топологическая сортировка

Теорема о топологической сортировке

Если \succ — отношение строгого или нестрогого порядка на конечном множестве M , то $\exists \gg$ — отношение линейного порядка на конечном множестве M , такое что \gg расширяет \succ



— нелинейный порядок

Варианты топологической сортировки:

A A

B B

C D

D E

E C

Доказательство Найдём минимальный элемент отношения \succ . \square это $x_1 \in M$. Удаляем x_1 из M : тогда имеем M без x_1 . Очевидно, что новое отношение также антисимметрично, транзитивно и рефлексивно либо антирефлексивно в зависимости от свойств изначального \Rightarrow в нём также есть минимальный элемент x_2 . Удаляем x_2 и продолжаем до тех пор, пока не получим последовательность x_1, x_2, \dots, x_n , где $n = |M|$

Вводим новый порядок $x_i \ll x_j$ для $i < j : x_1 \ll x_2 \ll \dots \ll x_n$

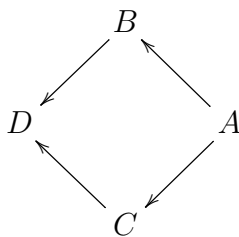
Почему \ll расширяет $<$?

Если $x < y \Rightarrow x$ был удалён раньше $y \Rightarrow x \ll y$ ■

Замечание Этот алгоритм (поиска минимального элемента и его удаление) не самый эффективный. Лучше сделать поиск в глубину с обратной нумерацией.

Замечание Топологическая сортировка - практически очень важная задача.

Например, порядок работы:



1.4 Транзитивное замыкание

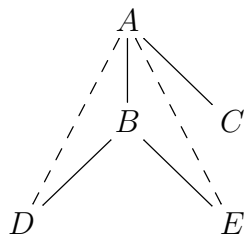
Был порядок - расширяли до линейного (Топологическая сортировка)

Было отношение - расширяем до транзитивного

(Транзитивное замыкание)

Имеется нетранзитивное отношение:

Добавлением пунктирных рёбер мы делаем его транзитивным:



Теорема

$\sqsupset R$ — бинарное отношение на множестве M . $\exists \bar{R}$ — отношение на M .

1) \bar{R} расширяет R ($R \in \bar{R}$)

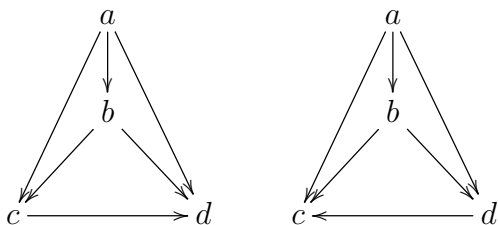
2) \bar{R} транзитивно

3) \bar{R} — минимальное транзитивное расширение, то есть если \tilde{R} — транзитивное расширение R , то $\tilde{R} \supset \bar{R}$

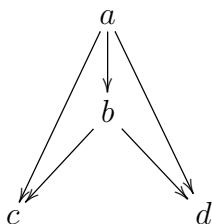
Доказательство (не для алгоритма) Рассмотрим все транзитивные расширения $\{\bar{R}\}$, возьмём $\bar{R} \cap \bar{R}_i$, то есть берём только те пары, которые есть во всех транзитивных расширениях.

Пример:

$M = \{a, b, c, d\} : aRb, bRc, bRd$



Оставим только те пары, которые есть везде — \bar{R} :



Проверим, что \bar{R} подходит под условия теоремы:

0) Почему \bar{R}_i существует?

$\bar{R}_i :=$ полное отношение $M \times M \Rightarrow \exists$

1) \bar{R} расширяет: $\sqsupset xRy \Rightarrow \forall \bar{R}_i : x\bar{R}_iy \Rightarrow x\bar{R}y$

2) $\sqsupset x\bar{R}y, y\bar{R}z \Rightarrow \forall \bar{R}_i : x\bar{R}_iy, y\bar{R}_iz \Rightarrow x\bar{R}_iz \Rightarrow$ транзитивно $x\bar{R}z$

3) $\bar{R} = \bar{R}_i \supset R$, так как $R = \bar{R}_i \cap \dots$ ■

2 Графы

Определение Неориентированный граф $G = (V, E)$,

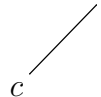
где V — множество (вершины); $E \subset \{(u, v) \mid (u, v) \text{ пара неупорядочена}\}$,
где $u, v \in V$

Замечание Вершины — точки или кружочки; рёбра — линии, неважно какой формы. Важно лишь то, что они соединяют.

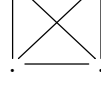
Замечание V — vertex, E — edge

Примеры:

$a \text{ --- } b$



$\begin{matrix} \cdot & \text{---} & \cdot \\ | & & | \\ \cdot & \text{---} & \cdot \\ | & & | \\ \cdot & & \cdot \end{matrix}$ — полный граф



— пустой граф

Определение G — полный граф, если $\forall u, v \in V : (u, v) \in E$

Определение Размер(порядок) графа $|G| =$ количество вершин $|V|$

Замечание

$|V| = n$ — количество вершин

$|E| = m$ — количество рёбер

$G = (n, m)$ граф

Определение Степень вершины $v \in V : \{(v, u) \mid (v, u) \in E\}$

(Количество рёбер с этой вершиной)

Определение k -регулярный граф — граф, где $\forall v \in V : deg(v) = k$

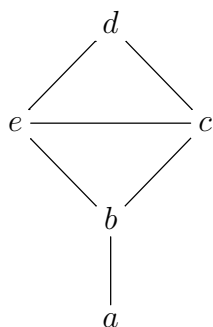
$\begin{matrix} \cdot & \text{---} & \cdot \\ | & & | \\ \cdot & \text{---} & \cdot \\ | & & | \\ \cdot & & \cdot \end{matrix}$ — 3-регулярный граф



Определение Путь в графе — последовательность вершин-рёбер:

$v_1, e_1, v_2, e_2, \dots, v_{n-1}, e_{n-1}, v_n$

$G = (V, E); v_i \in V; e_i \in E; e_i = (v_i, v_{i+1})$



Примеры путей:

- 1) abcd
- 2) ab
- 3) aba
- 4) abcdecdeba

Определение Замкнутый путь — путь, если $v_1 = v_n$
 Не замкнутый, открытый путь — путь, если $v_1 \neq v_n$

Определение Простой путь - если $e_i \neq e_j$ при $i \neq j$

Примеры:

- 4) abcdecdeba: путь замкнутый (начинается и заканчивается в вершине a), и непростой (два раза встречается ребро de)
- 5) bedce: путь простой, но не замкнутый

Путь		Все рёбра разные	Все вершины разные
Замкнутый	Замкнутый путь	Простой замкнутый путь	Цикл
Открытый	Открытый путь	Простой открытый путь	Цепь

Теорема Если \exists путь между вершинами $u, v \Rightarrow \exists$ цепь от u до v

Доказательство \square путь $v_1, e_1, v_2, e_2, \dots, v_{n-1}, e_{n-1}, v_n$. Рассмотрим все пути из этих рёбер и выберем минимальный - это будет цепь.

Иначе $v_i, \dots, v_j, \dots, v_n$.

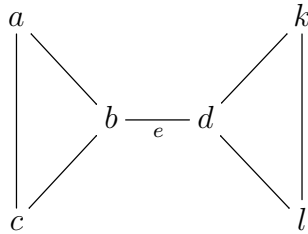
$\square v_i = v_j$.

Укоротим $U : v_i = v_j, \dots, v_n$ *Противоречие!!!*

Теорема Если есть простой замкнутый путь через ребро $e, \Rightarrow \exists$ цикл через e

Доказательство Аналогично ■

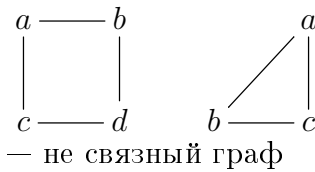
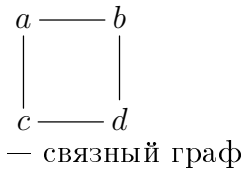
Замечание



$dbacbd$ — не простой путь (e повторяется). Цикла через e нет.

2.1 Связность графа

Определение Граф $G = (V, E)$ связан, если $\forall u, v \in V \exists$ цепь(путь) из U в V



Введём отношение \equiv на вершинах графа: $U \equiv V$, если существует путь из U в V .

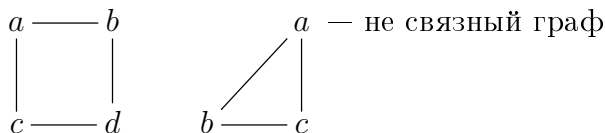
Проверим, что \equiv — отношение эквивалентности:

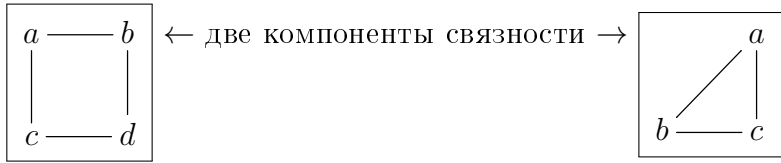
- 1) рефлексивность $U \equiv U$ — верно, путь U
- 2) симметричность.

Путь U : $v_1, e_1, v_2, e_2, \dots, v_{n-1}, e_{n-1}, v_n$. Путь V получем переворотом пути U : v_n, \dots, v_1, e_1, u . Поскольку мы считаем граф неориентированным, $\Rightarrow U \equiv V \Leftrightarrow V \equiv U$

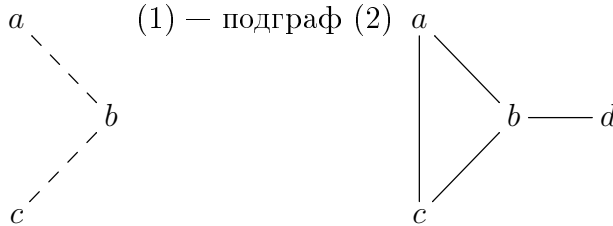
- 3) транзитивность: $U \equiv V, V \equiv W$ — путь $U \equiv W$ получен объединением первого и второго ■

Определение Класс эквивалентности \equiv — "компонента связности"





Определение $G_1 = (V_1, E_1)$ — подграф $G = (V, E)$, если $V_1 \subset V$, $E_1 \subset E$



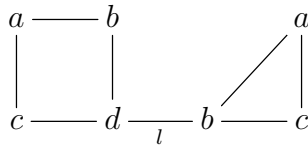
Замечание

G всегда является своим подграфом

\emptyset — подграф чего угодно

Определение $G = (V, E)$. Ребро e называется мостом, если количество компонент связности $G <$ количества компонент связности

$(V, E \setminus \{e\})$

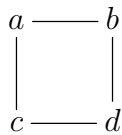


l — мост

Определение Степень связности графа G — минимальное количество рёбер, которые надо выкинуть, чтобы граф стал не связным.

Определение Двусвязный граф — граф, из которого нужно выкинуть хотя бы 2 ребра, чтобы он стал не связным.

Замечание Двусвязный граф \Leftrightarrow нет мостов



— двусвязный граф

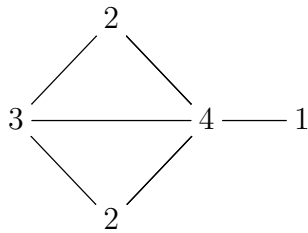
Определение Вершина $u \in U$ называется точкой сочленения, если количество компонент связности $G <$ количества компонент связности $\bar{G} = (V \setminus \{v\}, E \setminus \{(v, u) \mid (v, u) \in E\})$

2.2 Количество рёбер, вершин

Теорема В графе $G = (V, E)$, если $deg(u)$ — степень вершины u , количество рёбер

$$|E| = \frac{1}{2} \sum_v deg(v)$$

Пример:



Количество рёбер: $6 = \frac{1}{2}(3 + 2 + 2 + 4 + 1)$

Доказательство $deg(v)$ — количество рёбер, выходящих из вершины $\sum_v deg(v)$ — все рёбра посчитаны дважды $= 2|E|$ ■

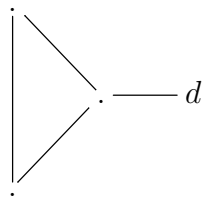
Следствие:

- 1) Сумма степеней вершин всегда чётная
- 2) Вершин нечётной степени - чётное количество

Задача У каждого из 15 инопланетян по 3 руки. Могут ли они взяться за руки так, чтобы ни у кого не осталось свободной руки?

Решение Нет, поскольку это граф из 15 (нечётного числа) вершин степени 3 (нечётной)

Определение Висячая вершина - вершина степени 1



d — висячая вершина

Теорема Если в графе есть рёбра, но нет висячих вершин $\Rightarrow \exists$ цикл
Доказательство Берём ребро $e(u_1, u_2)$.

u_2 — не висячая \Rightarrow из неё есть ещё ребро $e(u_2, u_3)$.

u_3 — не висячая \Rightarrow из неё есть ещё ребро $e(u_3, u_4)$.

Продолжаем, пока очередной u_n не будет равен $u_i : 1 \leq i < n$.

Путь u_i, u_{i+1}, \dots, u_n — цикл, поскольку все рёбра и вершины разные (при первом повторе мы замкнули цикл) ■

Определение Дерево - связный граф без циклов

Пример: $\cdot \text{---} \cdot \text{---} \cdot \text{---} \cdot$

Теорема В любом дереве есть хотя бы висячих 2 вершины

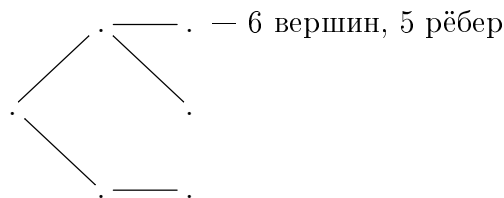
Доказательство Берём любую вершину: если она не висячая — идём по ребру; если опять не висячая - есть ещё ребро ... Циклов нет \Leftrightarrow будет конец — это и есть висячая вершина.

Чтобы найти вторую висячую вершину, нужно начать путь из первой

■

Теорема Если $G = (V, E)$ - дерево, $|V| = 1 + |E|$

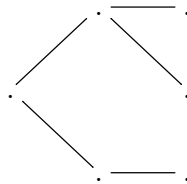
Пример:



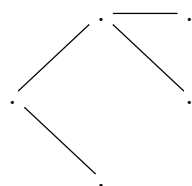
Доказательство по индукции (количество вершин)

База: $|V| = 1, |E| = 0$ \square $|V| = 1 + |E|$

Переход: \square $|V| = n + 1$ $\cdot \text{---} \cdot$ $n+1$ вершина



Найдём висячую вершину и удалим её, с её единственным ребром:

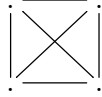


$\bar{G} = (V/\{v\}, E/\{e\})$ — тоже дерево, поскольку граф связан и нет циклов $\Rightarrow |\bar{V}| = 1 + |\bar{E}|$

$|V| = |\bar{V}| + 1; |E| = |\bar{E}| + 1$

$\Rightarrow |V| = 1 + |E|$ ■

□ G — полный граф, $\forall u \neq v \in V$ соединены ребром



если n вершин ($|V| = n$), то рёбер:

1) $C_n^2 = \frac{n(n-1)}{2}$

2) степень всех вершин $n-1$

$$\sum_v \deg(v) = 2|E| \Rightarrow n(n-1) = 2|E|$$

Ответ: $\frac{n(n-1)}{2}$

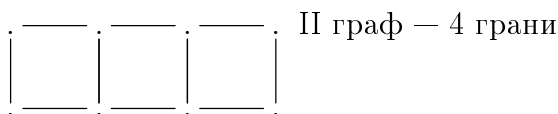
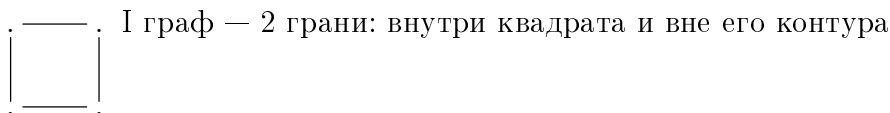
2.3 Планарные графы

Определение G — планарный граф, если его можно нарисовать на плоскости так, чтобы рёбра не пересекались



Оба графа планарны, поскольку планарность — свойство графа, а не рисунка. Можно сказать, что граф A_1, B_1, C_1, D_1 "неправильно" нарисованный

Формула Эйлера Если связный планарный граф $G = (V, E)$ нарисован на плоскости, у него можно посчитать грани. □ их $f, |V| = n, |E| = m$



Тогда $n - m + f = 2$

Проверим:

I граф: $4 - 4 + 2 = 2$

II граф: $8 - 10 + 4 = 2$

Доказательство (индукция по количеству рёбер)

База: G — дерево. У дерева всегда 1 грань. Поскольку вокруг грани всегда цикл, а в дереве циклов нет, то и грань будет только одна - внешняя: $n - (n - 1) + 1 = 2$

Переход: возьмём граф G , для которого мы не знаем, верна ли формула. Если она верна для графа $\bar{G} \Rightarrow$ будет верна и для G

(G и \bar{G} — связанные планарные графы)

G — не дерево \Rightarrow есть цикл. Берём любое ребро цикла - вокруг него две грани. Удалим ребро, получим \bar{G} — тоже связан и планарен.

$\bar{n}, \bar{m}, \bar{f}$ — вершины, рёбра, грани \bar{G}

$$\bar{n} = n; \bar{m} = m - 1; \bar{f} = f - 1$$

По индукционному предположению $\bar{n} - \bar{m} + \bar{f} = 2$

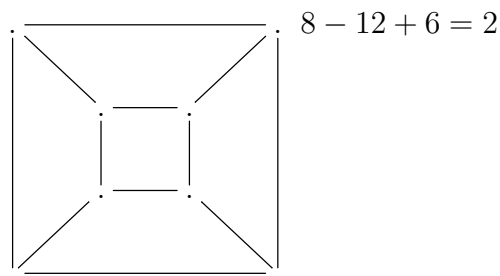
$$\Rightarrow n - (m - 1) + (f - 1) = 2$$

$$\Rightarrow n - m + f = 2 \blacksquare$$

Следствия

1. Неважно, как рисовать планарный граф - количество граней постоянно

2. Про многогранники так же: кубик на плоскости может быть представлен следующим образом



3. Если G — планарный (не обязательно связный) граф, то $n - m + f = 1 + |\text{количество компонент связности } G|$

4. \square у каждой грани вокруг ≥ 3 ребра

$3f \leq \sum_g$ количество рёбер вокруг $g \leq 2m$ (каждое ребро посчитано дважды) \Rightarrow

$$\boxed{3f \leq 2m}$$

Домножаем выражение $\boxed{n - m + f = 2}$ на 3, получаем:

$$3n - 3m + 3f = 6 \Rightarrow 3n - 3m + 2m \geq 6 \Rightarrow 3n - m \geq 6 \Rightarrow$$

$$\boxed{m \leq 3n - 6} \tag{1}$$

Итого, $m \leq 3n - 6$ в связном планарном графе

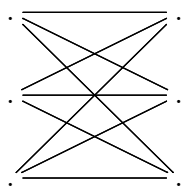
Следствие Полный граф при $n = 5$ не планарен

Доказательство $n = 5, m = \frac{n(n-1)}{2} = \frac{5 \times 4}{2} = 10$

$10 \leq 3 \times 5 - 6 = 9$ — неверно **Противоречие**

Замечание K_n — полный граф на n вершинах

Утверждение Граф $K_{3,3}$ тоже не планарный



Доказательство $n = 6, m = 9 : 9 \leq 3 \times 6 - 6$ верно

Сколько у него должно быть граней, чтобы он был планарным?

$6 - 9 + f = 2 \Rightarrow f = 5$ граней

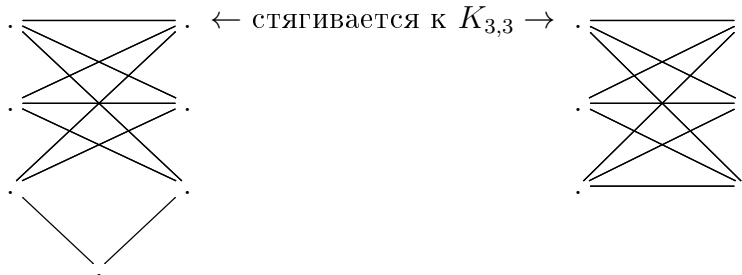
В $K_{3,3}$ все циклы — чётные (ходим слева направо и справа налево)

\Rightarrow у любой грани ≥ 4 ребра

$4f \leq \sum_g \text{количество рёбер вокруг } g \leq 2m \Rightarrow m \geq 2f$

Однако $9 \geq 2 \times 5$ — неверно **Противоречие**

Теорема Понтрягина - Куратовского Граф G планарен \Leftrightarrow он не содержит подграфов, "стягивающихся" к K_5 и $K_{3,3}$



(2)

2.4 Хроматизм

Определение $G = (V, E)$ — граф.

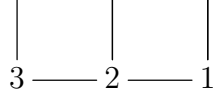
Раскраска графа G в k цветов — это функция $c : V \rightarrow \{1 \dots k\}$

Причём если есть ребро (u, v) , то $c(u) \neq c(v)$

1 — 2 — 3 — раскраска



1 — 2 — 3 — не раскраска (существует ребро 2-2)



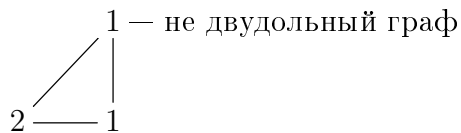
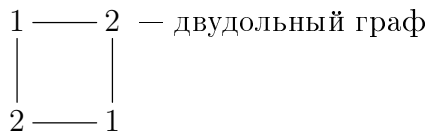
Какие графы можно раскрасить в 1 цвет?

1 1 — графы без рёбер

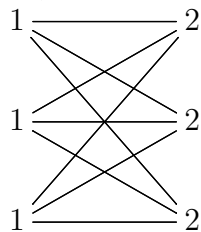
1 1

Какие графы можно раскрасить в 2 цвета?

Определение Граф называется *двудольным*, если его можно раскрасить в 2 цвета



$K_{3,3}$ — двудольный:

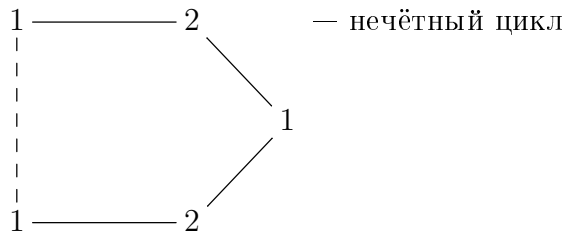


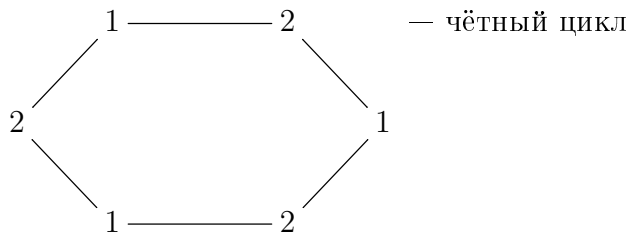
Замечание Двудольные графы часто рисуют из двух частей (долей (принцип построения, как $K_{3,3}$))

Теорема G — двудолен \Leftrightarrow все циклы G имеют чётную длину

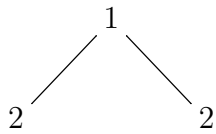
Доказательство

1) Граф двудолен \Rightarrow циклы чётные, иначе получаются две соседние вершины с одним цветом:

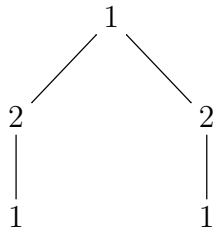




2) Циклы чётные \Rightarrow граф двудолен: "подвесим граф за вершину"
 Выбираем любую вершину и назначаем ей цвет 1. Тогда все вершины, соединённые с ней, будут иметь цвет 2:

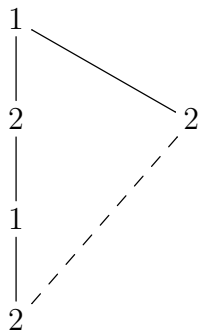


Теперь из всех вершин цвета 2 будем также проводить все рёбра туда, куда их ещё не проводили (рёбра, которые направлены на уже нарисованную ранее вершину, нас не интересуют) - новые рёбра будут иметь цвет 1:



Назначаем цвета "по уровням"
 Почему "обратное ребро" (ведущее в ранее существовавшую вершину) не соединяет одинаковые цвета?

Поскольку в таком случае будет нечётный цикл, а по условию все циклы — чётные:



Определение $\square G = (V, E)$ — граф.

$\psi(G)$ — хроматическое число графа, минимальное количество цветов, в котрое его можно раскрасить

$$\psi \left(\begin{array}{c} \cdot \\ | \\ \cdot \\ | \\ \cdot \end{array} \begin{array}{c} \cdot \\ | \\ \cdot \\ | \\ \cdot \end{array} \right) = 3 ; \psi \left(\begin{array}{c} \cdot \\ | \\ \cdot \\ | \\ \cdot \end{array} \begin{array}{c} \cdot \\ | \\ \cdot \\ | \\ \cdot \end{array} \right) = 2$$

$$\psi \left(\begin{array}{c} \cdot \\ | \\ \cdot \\ | \\ \cdot \end{array} \begin{array}{c} \cdot \\ | \\ \cdot \\ | \\ \cdot \end{array} \right) = 5 \text{ (все должны быть разные)}$$

$\psi(K_n = n)$, где K_n — полный граф на n вершиннах

Замечание Если $k \geq \psi(G)$, то G можно покрасить в k цветов

Утверждение $\psi(G) \leq (\max \deg V + 1)$

Пример:

$$G = \begin{array}{c} 1 \\ | \\ 3 \end{array} \begin{array}{c} 2 \\ | \\ 2 \end{array} \text{ — степени вершин; } \max \deg = 3; \psi(G) \leq 3 + 1 = 4$$

Доказательство индукция по количеству вершин

База: $n = 0, m = 1$ — верно ($\max \deg = 0, \psi(G) \geq 1$)

Переход: $G, v - \max \deg = \Delta$, уберём её — получим $\bar{G} = G$ без v
 $\max \deg \bar{G} \leq \max \deg G = \Delta$

Раскрасим \bar{G} в $\Delta + 1$ цвет

Для v запрещены $\leq \Delta$ цветов $\Rightarrow \geq 1$ цвет разрешены, покрасим в него

■

Утверждение G — планарный граф $\Rightarrow \psi(G) \leq 5$

Доказательство

1) В G есть вершина степени ≤ 5

$$|V| = n; |E| = m$$

$$\text{Если такой нет} \Rightarrow \deg V \geq 6 \Rightarrow \sum \deg V \geq 6n$$

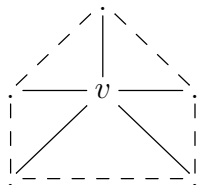
$\Rightarrow 2m \geq 6n \Rightarrow m \geq 3n$, но в планарном графе $m \leq 3n - 6$, согласно доказанному ранее **Противоречие** \Rightarrow точно есть вершина степени ≤ 5

Утверждение Раскрасим в 5 цветов по индукции

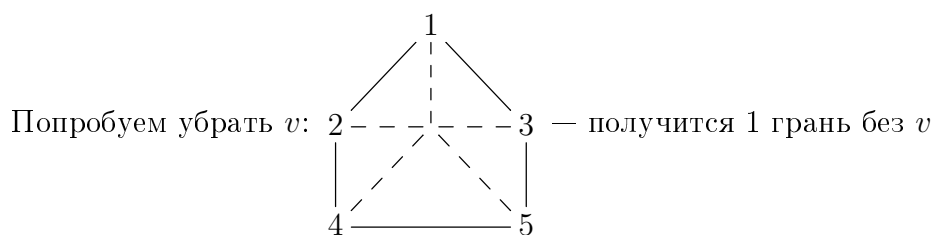
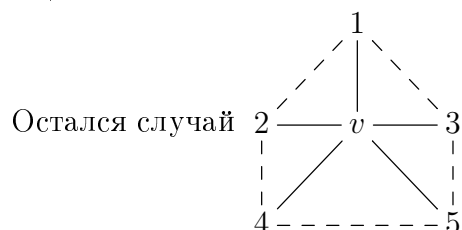
База: графы из 1, 2, 3, 4, 5 вершин точно можно раскрасить

Переход: $\square n$ вершин. По индукционному предположению, для графа с $(n - 1)$ вершинами раскраска есть

Берём $v : \deg v \leq 5$



Раскрасим \tilde{G} без v : если соседи v используют ≤ 4 цветов, \Rightarrow для v есть цвет



Стянем в одну две вершины, которые не соединены ребром и имеют одинаковый цвет — сделаем \tilde{G}

Если v_i, v_j все соединены ребром \Rightarrow есть $K_5 \Rightarrow G$ — не планарен
 \tilde{G} имеет $(n - 2)$ вершин \Rightarrow можно раскрасить

Если мы стянули две вершины, значит они имеют одинаковый цвет \Rightarrow есть цвет для v ■

Утверждение $\psi(G) \leq 4$ (Проблема 4 красок)

Это первая крупная математическая теорема, доказанная с помощью компьютера

2.5 Хроматические многочлены

Определение $\chi(G, k)$ — функция "сколько способов раскрасить G в k цветов"

$$\chi(G, k) = \begin{cases} k = 0 & 0 \\ k = 1 & 0 \\ k = 2 & 2 \\ k = 3 & 6 \\ \text{иначе} & k(k-1) \end{cases}$$

$$\chi(\cdot \text{---} \cdot, k) = k(k-1)$$

$$\chi(\cdot \quad \cdot, k) = k^2$$

1) (Изолированный граф) $\chi(\emptyset_n, k) = k^n$

2) (Полный граф) $\chi(K_n, k) = \underbrace{k(k-1)(k-2)\dots(k-n+1)}_{n \text{ множителей}} = k^n$

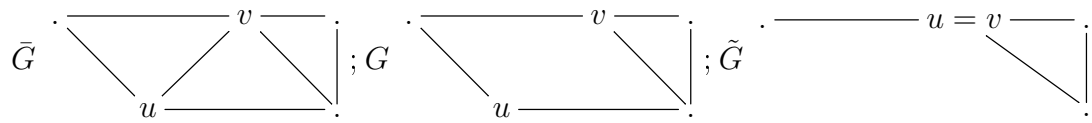
3) (Дерево) $\chi(T_n, k) = k(k-1)^{n-1}$

4) \bar{G} — граф, u, v — вершины, соединены ребром (u, v)

G — \bar{G} без вершин u, v

\tilde{G} — \bar{G} с соединёнными в одну вершинами u, v

Пример:



$$\boxed{\chi(G, k) = \chi(\bar{G}, k) + \chi(\tilde{G}, k)} \quad (3)$$

= (способы раскрасить G , если u, v — разный цвет) +
 + (способы раскрасить G , если u, v — одинаковый цвет)
Следствие

$$\boxed{\chi(\bar{G}, k) = \chi(G, k) - \chi(\tilde{G}, k)}$$

Примеры:

$$\chi\left(\begin{array}{c} \cdot \text{---} \cdot \\ \cdot \text{---} \cdot \\ \cdot \text{---} \cdot \end{array}, k\right) = \chi\left(\begin{array}{c} \cdot \text{---} \cdot \\ \cdot \text{---} \cdot \\ \cdot \text{---} \cdot \end{array}, k\right) + \chi\left(\begin{array}{c} \cdot \text{---} \cdot \\ \cdot \text{---} \cdot \\ \cdot \text{---} \cdot \end{array}, k\right) =$$

$$= k(k-1)(k-2)(k-3) + k(k-1)(k-2)$$

5) $\chi\left(\begin{array}{c} \cdot \text{---} \cdot \\ \cdot \text{---} \cdot \\ \cdot \text{---} \cdot \end{array}, k\right)$ C_n — n вершин

$$c_n = \chi(C_n, k) = k(k-1)^{n-1} - k(k-1)^{n-2} + k(k-1)^{n-3} + \dots$$

$$\dots + (-1)^n k(k-1) + (-1)^{n+1} k$$

Это геометрическая прогрессия:

Начало: $(-1)^{n+1} k$

Множитель q : $-(k-1)$

Слагаемых: n штук

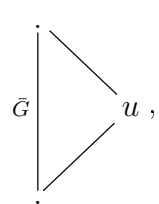
$$S = (-1)^{n+1} k^{\frac{q^n-1}{q-1}} = (-1)^{n+1} k^{\frac{(1-k)^n-1}{1-k-1}} = (-1)^{n+1} k^{\frac{(-1)^n(k-1)^n-1}{-k}} = (-1)^n \left((-1)^n (k-1)^n - 1 \right) = (k-1)^n + (-1)^n (k-1) \blacksquare$$

Утверждения

1) $\sqsupset G$ имеет висячую вершину u

$$\underbrace{\bar{G} \text{ --- } u}_G, \chi(G, k) = \chi(\bar{G}, k) + \underbrace{(k-1)}_{\text{раскрасить } u}$$

2)



$$\underbrace{\bar{G} \text{ --- } u}_G, \chi(G, k) = \chi(\bar{G}, k) + \underbrace{(k-2)}_{\text{раскрасить } u}$$

3) $G = \bar{G} \cup \tilde{G}$; между \bar{G} и \tilde{G} нет рёбер

$$\underbrace{\cdot \text{ --- } \bar{G} \cdot \text{ нет рёбер } \cdot \text{ --- } \tilde{G} \cdot}_G, \chi(G, k) = \chi(\bar{G}, k) \times \chi(\tilde{G}, k)$$

Общий случай рассмотрен здесь

Утверждение $\chi(G, k)$ — это многочлен:

- 1) Старший коэффициент = 1
- 2) Степени = n (количество вершин)
- 3) Знаки чередуются: $k^n - k^{n-1} + k^{n-2} - \dots - 0$
- 4) Младший коэффициент = 0
- 5) Коэффициент при $k^{n-1} = \pm m$ (количество рёбер)

Доказательство Индукция по количеству вершин, при равном количестве вершин — по количеству рёбер

База: возьмём пустой граф из n вершин: $\chi(\emptyset_n, k) = k^n = 0 \times k^{n-1}$

Переход: \bar{G} с рёбрам

$$\chi(\cdot \text{ --- } \cdot, k) = \underbrace{\chi(\cdot \text{ --- } \cdot, k)}_{\text{мало рёбер}} - \underbrace{\chi(\cdot, k)}_{\text{мало вершин}}$$

Работает индукционное предположение

- 1) Старший коэффициент = $(1 \times k^n - \dots) - (k^{n-1} - \dots)$
- 2) Степень = n
- 3) Знаки чередуются: $(k^n - k^{n-1} + k^{n-2} - \dots) - (k^{n-1} + k^{n-2} - \dots)$
- 4) Младший коэффициент = $0 - 0 = 0$
- 5) (Количество рёбер G) $\times k^{n-1} - k^{n-1} =$
 $-(\text{Количество рёбер } G + 1) \times k^{n-1} = -(\text{Количество рёбер } \bar{G}) \times k^{n-1}$

На примере:

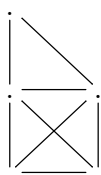
$$\chi\left(\begin{array}{c} \cdot \\ \diagup \quad | \\ \cdot \\ \hline \cdot \\ \diagdown \quad | \\ \cdot \end{array}, k\right) = (k-1) \times \chi\left(\begin{array}{c} \cdot \\ \diagup \quad | \\ \cdot \\ \hline \cdot \\ \diagdown \quad | \\ \cdot \end{array}, k\right) =$$

$$= (k-1)k(k-1)(k-2) = k^4 - 4k^3 + 5k^2 - 2k$$

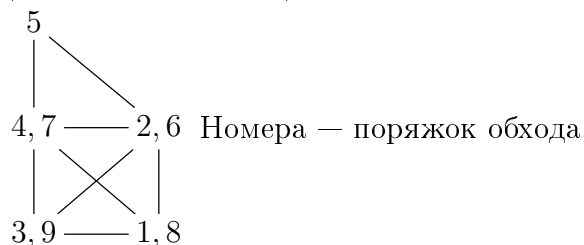
Утверждение $\chi(G)$ — хроматическое число (минимальное число цветов для раскраски)

$\chi(G, k), k = 0, 1, 2, \dots, \chi(G) - 1$ — корни, $\chi(G)$ — не корень

2.6 Эйлеровы графы



Задача — нарисовать, не проводя жважды по одному ребру (по вершине можно):



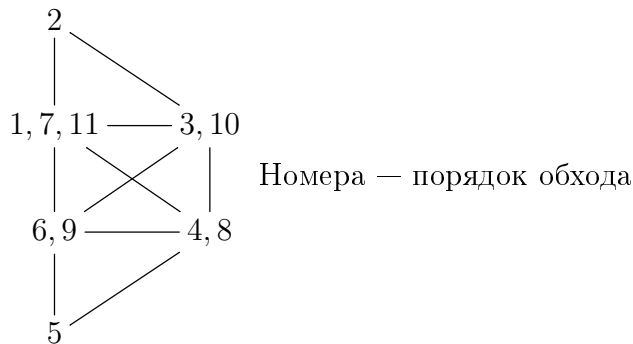
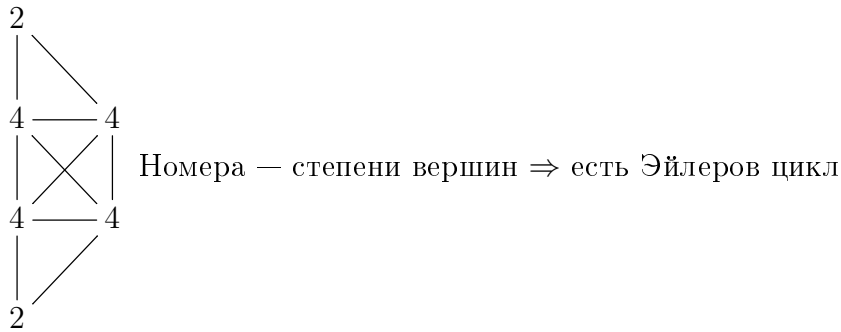
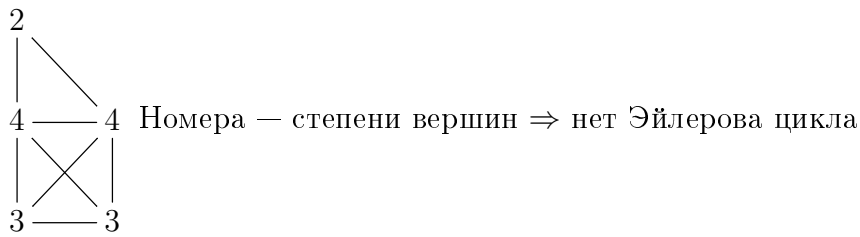
Определение Эйлеров путь — простой путь, содержащий все рёбра

Определение Эйлеров цикл — цикл, содержащий все рёбра

(В обоих случаях не проходим дважды по одному ребру)

Утверждение Граф G содержит Эйлеров цикл \Leftrightarrow граф связан и все степени вершин — чётные

Пример:



Доказательство

\Rightarrow

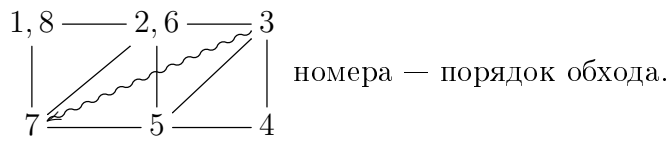
Граф связан, потому что мы прошли по всем вершинам, значит от каждой до каждой можно прийти

Каждая вершина чётна, потому что количество входов в неё за время цикла = количеству выходов

\Leftarrow

Начнём строить цикл: идём из любой вершины, выбираем ребро на каждом шаге, которое ещё не использовали. В каждой вершине по пути использовано чётное число рёбер (k входов, k выходов) + 1 ребро, через которое мы сейчас вошли \Rightarrow использовано чётное количество рёбер \Rightarrow есть ещё ≥ 1 ребро, через которое и нужно уйти.

Такая ситуация складывается на всех вершинах, кроме начальной: из неё мы вышли на 1 раз больше \Rightarrow мы закончим ходить в начальной вершине.



Мы обошли не все вершины, однако вернулись в изначальную.

Нужно выкинуть все просмотренные рёбра, которые мы обошли: в оставшейся части тоже все степени чётные.

Поскольку граф G связан, из изначальной вершины X мы можем попасть в любую другую вершину и ребро.

Повторим процесс из ребра, входящего в первый цикл, из которой ведёт новое ребро.

Объединим два цикла: проходимся по первому циклу до первого пересечения со вторым, проходим по второму, возвращаемся в то же место пересечения с первым и заканчиваем первый.

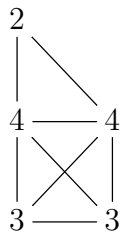
Продолжаем до тех пор, пока все рёбра не объединятся в один цикл.

■

Теорема Граф G содержит Эйлеров путь \Leftrightarrow

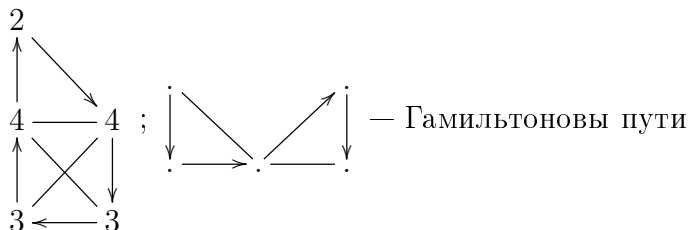
- 1) Граф G связан
- 2) $\left[\begin{array}{l} \text{Степени всех вершин чётны} \\ \text{Степени всех вершин, кроме двух, чётны} \end{array} \right.$

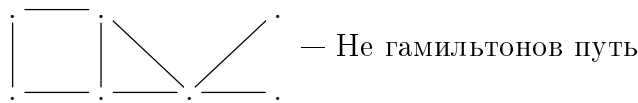
Во втором случае нечётные вершины гарантированно являются началом и концом



2.7 Гамильтоновы графы

Определение Гамильтонов путь/цикл — простой путь/цикл на всех вершинах.

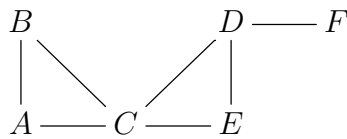




2.8 Длины путей в графах

Определение Длина пути в графе — количество рёбер в пути

Пример



ABCDF — путь от A до F, длина 4 (4 ребра)

ACEDF — путь от A до F, длина 4 (4 ребра)

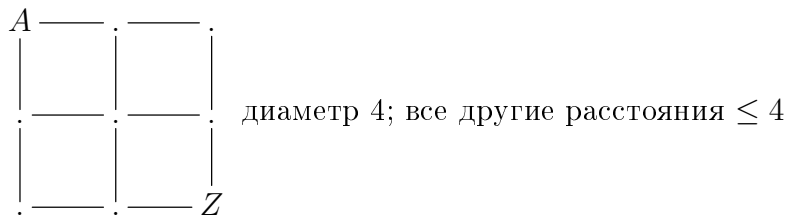
ACDF — путь от A до F, длина 3 (3 ребра)

ABCEDF — путь от A до F, длина 5 (5 рёбер)

Определение Расстояние между вершинами — минимальная длина между вершинами или $+\infty$, если пути нет

Обозначение $d(X, Y)$ — расстояние от X до Y

В примере выше: $d(A, F) = 3$ (достигается на AF)

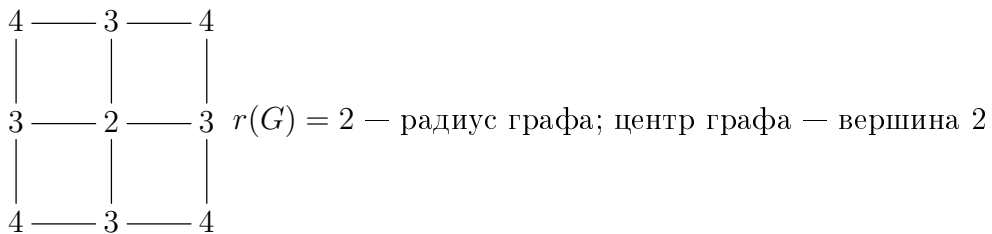


Определение Для каждой вершины графа $G(V, E)$ можно посчитать максимальное расстояние до других вершин

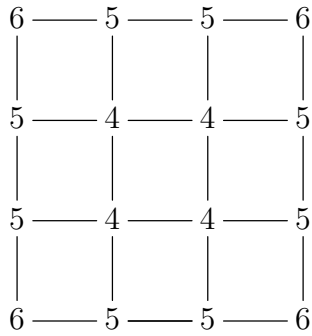
$r(v) := \max \{d(v, s) | s \in V\}$

Радиус $r(G) = \min \{r(v) | s \in V\}$

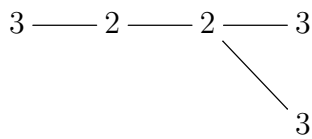
Те вершины, на которых достигается минимум — центр



центров может быть много:



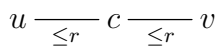
4 центра в вершинах 4; $r(G) = 4$ — радиус графа



2 центра в вершинах 2; $r(G) = 2$ — радиус графа

Утверждение В графе $G(V, E) : d(G) \leq 2r(G)$

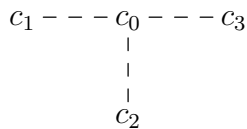
Доказательство $\square c$ — центр графа; $u, v \in V$



$d(c, u) \leq r; d(c, v) \leq r \Rightarrow d(u, v) \leq 2r \Rightarrow d(G) = \underbrace{\max}_{u, v} d(u, v) \leq 2r \blacksquare$

Утверждение В дереве ≤ 2 центров

\square их 3:



Построим пути между c_1 и c_2

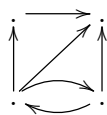
Потом между c_2 и c_3

(В дереве ровно 1 путь между вершинами)

Вершина развилки $c_0 : r(c_0) < r(c_1) = r(c_2) = r(c_3) = r(G) = r$

Замечание Будем иногда дальше использовать ориентированные графы $G = (V, E)$ (рёбра в ориентированном графе иногда называют дугами)

$E \subset \{(u, v) \text{ — упорядоченная пара}\}$

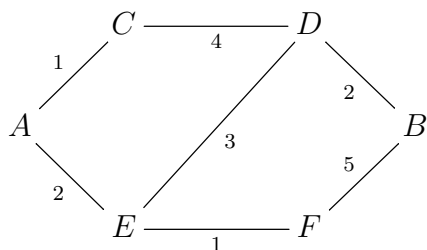


Замечание У рёбер будет вес

$G = (V, E)$; вес — функция $f : E \rightarrow \mathbb{R}$
 то есть число у каждого ребра

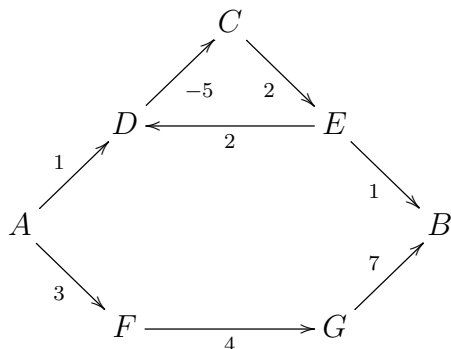
$\cdot \xrightarrow{3} \cdot$; $\cdot \xrightarrow{7} \cdot$

Расстояние на графе считается как минимальная сумма весов по всем путям



$d(A, B) = ?$
 $d(ACDEFB) = 1 + 4 + 3 + 1 + 5 = 14$
 $d(AEFB) = 2 + 1 + 5 = 8$
 $d(AEDB) = 2 + 3 + 2 = 7$
 $\Rightarrow d(A, B) = 7$

Замечание Расстояние во взвешенном графе не всегда существует



$d(F, G) = 4$
 $d(G, F) = +\infty$
 $d(A, B) = ?$
 $d(ADC EB) = 1 - 5 + 2 + 1 = -1$
 $d(ADCEDCEB) = 1 - 5 + 2 + 2 - 5 + 2 + 1 = -2$
 и так далее, минимально $-\infty$

Утверждение В графе есть все расстояния \Leftrightarrow в графе нет циклов отрицательной длины

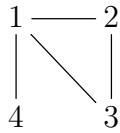
Доказательство Если есть цикл отрицательной длины $\Rightarrow \forall$ две вершины этого цикла не имеют расстояния (или $-\infty$)

Если нет расстояния, то есть для u, v есть пути без положительных циклов сколь угодно маленькие. \square есть путь длинее $n = |V|$ рёбер \Rightarrow повторяются вершины в пути — это и будет отрицательный цикл

Как хранятся графы в компьютере?
 (Представление графа в компьютере)

1. Матрица смежности: таблица вершины \times вершины

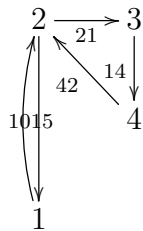
$$a(i, j) = \begin{cases} 0, & \text{если нет ребра} \\ 1, & \text{если есть ребро} \end{cases}$$



	1	2	3	4
1	0	1	1	1
2	1	0	1	0
3	1	1	0	0
4	1	0	0	0

— симметрична для неориентированных графов

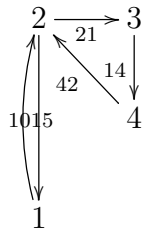
Для графов с весами $a(i, j)$ = вес ребра (i, j) или $+\infty$, если ребра нет



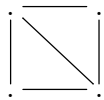
	1	2	3	4
1	$+\infty$	15	$+\infty$	$+\infty$
2	10	$+\infty$	21	$+\infty$
3	$+\infty$	$+\infty$	$+\infty$	14
4	$+\infty$	42	$+\infty$	$+\infty$

Объём памяти: $n^2 = |V|^2$

2. Списки смежности — для каждой вершины храним список соседей



- 1 : 2(15)
- 2 : 1(10), 3(21)
- 3 : 4(14)
- 4 : 2(42)



1 : 2,3,4

2 : 1,3

3 : 2,4

4 : 1,3

Память $\approx |E|$ — количество рёбер

3. Неявные способы

Умеем вычислять всех соседей \forall вершины

Пример: обход конём шахматной доски

Граф:

вершины = клетки (64)

рёбра — вершины, связанные ходом коня

Можно для \forall клетки (вершины) посчитать, куда из неё можно попасть

Задача обхода конём \Leftrightarrow задача поиска Гамильтонова цикла в этом графе

Задача: даны две вершины u, v : найти $d(u, v)$ и путь, на котором достигается это расстояние

Замечание оказывается, что найти путь от u до v — это \approx то же самое, что найти путь от u до всех вершин

2.9 Алгоритм Беллмана - Форда

$G(V, E)$, f — вес, $u \in V$. Найти расстояние $d(u, v) \forall v \in V$

Будем писать $d(v) = d(u, v)$, так как u не меняется

Будем хранить в массиве d текущие найденные расстояния

В начала $d(u) = 0, d(v) = +\infty$, если $v \neq u$

Релаксация ребра $e = (v_1, v_2)$:

$v_1 \text{ — } v_2$

если $d(v_1) + f(v_1, v_2) < d(v_2) \Rightarrow d(v_2) := d(v_1) + f(v_1, v_2)$

Алгоритм:

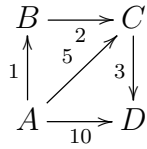
повторить $n - 1$ раз, перебрать все рёбра e

и каждое релаксировать

В неориентированном графе

$v_1 \text{ — } v_2 = v_1 \overset{\curvearrowright}{\leftarrow} v_2$, то есть две релаксации на ребро

Пример 1:



$n = 4$ (4 вершины)
 $A : B(1), C(5), D(10)$
 $B : C(2)$
 $C : D(3)$
 $D :$

	A	B	C	D
d	0	$+\infty$	$+\infty$	$+\infty$

Шаг 1:

	A	B	C	D
AB	0	1	$+\infty$	$+\infty$
AC	0	1	5	$+\infty$
AD	0	1	5	10
BC	0	1	3	10
CD	0	1	3	6

Шаг 2:

	A	B	C	D
AB	0	1	$+\infty$	$+\infty$
AC	0	1	5	$+\infty$
AD	0	1	5	10
BC	0	1	3	10
CD	0	1	3	6

— ничего не изменится

Шаг 3:

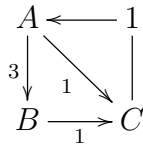
	A	B	C	D
AB	0	1	$+\infty$	$+\infty$
AC	0	1	5	$+\infty$
AD	0	1	5	10
BC	0	1	3	10
CD	0	1	3	6

— ничего не изменится

Время работы $\approx |V| \times |E| \leq |V|^3$

$$\text{Ответ} \left\{ \begin{array}{l} d(A) = 0 \\ d(B) = 1 \\ d(C) = 3 \\ d(D) = 6 \end{array} \right.$$

Пример 2:



A: 3B, 1C

B: 4C

C: 1B

$n = 3 \Rightarrow n - 1 = 2 \Rightarrow 2$ раза запускаем циклы релаксации

Пути из A:

	A	B	C
d	0	$+\infty$	$+\infty$

Шаг 1:

	A	B	C
AB	0	$+\infty$	$+\infty$
AC	0	3	$+\infty$
BC	0	3	1
CB	0	2	1

Шаг 2: AB, AC, BC, CB — ничего не улучшилось

Ответ:

A	B	C
0	2	1

Корректность алгоритма

Теорема В конце массив d содержит расстояния от A

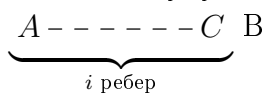
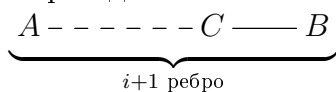
Доказательство После i -го цикла релаксации всех рёбер d хранит числа $d(v) \leq \min(\text{длин путей, в которых } \leq i \text{ рёбер})$

Действительно:

База: $i = 0 = \min(\text{путей из } 0 \text{ рёбер})$ (Только путь A — A)

$d(A) = 0$ — верно; $d(u) = +\infty$

Переход: \square есть оптимальный путь из $i + 1$ ребра



По индукционному предположению $d(C) = \text{dist}(A, C)$

Длина пути A — C — B = $\text{dist}(C) (= d(C)) + \text{dist}(C, B)$

Проверка: $d(C) + \text{dist}(C, B) \leq d(B)$ — верно, так как путь A — C — B оптимальный = $d(B) = d(C) + \text{dist}(C, B)$

Почему $n - 1$ этап?

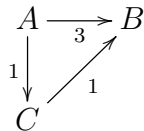
Оптимальный путь не содержит циклов $\Rightarrow \leq n - 1$ ребро ■

Замечание Мы выясняем только расстояния, но путь не называется
Как восстановить путь?

Будем сохранять информацию об успешных релаксациях

$Prev$ — изначально пустой массив вершин

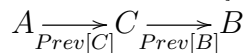
Если релаксация $u \rightarrow v$ успешна, то $Prev[v] = u$ — оптимальный путь в v лежит через u



	A	B	C
d	0	$+\infty$	$+\infty$

	A	B	C
AB	0	$+\infty$	$+\infty$
AC	0	$3/A$	$+\infty$
BC	0	$3/A$	$1/A$
CB	0	$2/C$	$1/A$

Восстановить путь в B :



В общем случае путь $A \rightarrow v$ — это

$$A \rightarrow Prev[Prev[v]] \rightarrow [Prev[v]] \rightarrow v$$

2.10 Алгоритм Дейкстры

В отличие от алгоритма Беллмана - Форда, требуется чтобы все веса $w(e) \geq 0$

Алгоритм Дан граф $G = (V, E)$, $A \in V$

Найти расстояния до всех вершин $d(u) = dist(A, u)$

Алгоритм $d(A) = 0, d(u \neq A) = +\infty$

$P = \emptyset$ — обработанные вершины

Повторяем $n = V$ раз:

Выбрать $u \in V \setminus P$, где $d(u)$ — минимальный

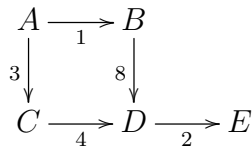
(выбираем из необработанных вершин ту, у которой минимальный d)

$\forall e \in$ ребро из $u, e = (u, v)$

Релаксируем ребро e

$P = P \cup \{u\}$

Пример:



	A	B	C	D	E	
d	0	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$P = \emptyset$
$u = A$		1	3	$+\infty$	$+\infty$	$P = \{A\}$
$u = B$			3	9	$+\infty$	$P = \{A, B\}$
$u = C$				7	$+\infty$	$P = \{A, B, C\}$
$u = D$					9	$P = \{A, B, C, D\}$
$u = E$					9	$P = \{A, B, C, D\}$

Эффективность: $|V| \times |E| \times \underbrace{\log|V|}_{\text{Выбор } \min}$

Корректность

Идея: На каждом шаге $d(u) = \min$ путей $A \overset{\text{обработанные вершины}}{\dashrightarrow} u$

База: шаг = 0; $d(A) = 0$; $d(u) = +\infty$

Переход: Выбираем $u = \text{минимальная вершина из } V \setminus \{P\}$

\square есть оптимальный путь в u

$A \overset{\text{обработанные рёбра}}{\dashrightarrow} \bar{u} \xrightarrow{x} u$

$dist(\bar{u}) = dist(u) - x$

По индукционному предположению $dist(\bar{u}) = d(\bar{u})$

$d(u) \geq dist(u) > dist(\bar{u}) \Rightarrow d(u) > d(\bar{u})$

\Rightarrow **Противоречие** ($d(u)$ был минимальным)

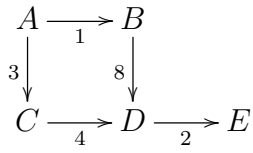
\square оптимальный путь в v идёт через u :

$dist(A, u) + w(u, v) = dist(A, v) \Rightarrow$ релаксация $u \longrightarrow v$ успешна, $d(v)$

получит расстояние \blacksquare

Для восстановления пути нужен аналогичный массив $Prev$

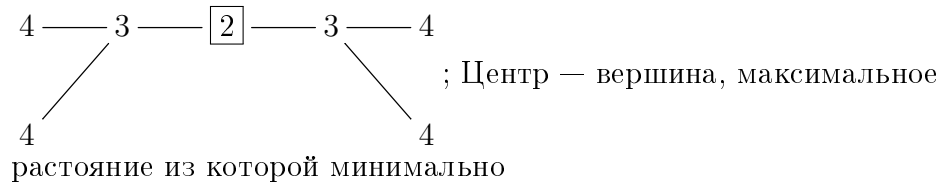
В случае успешной релаксации $u \longrightarrow v$ $Prev[v] = u$



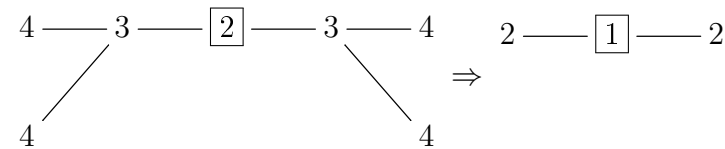
A	B	C	D	E
0	$+\infty$	$+\infty$	$+\infty$	$+\infty$
	1/A	3/A	$+\infty$	$+\infty$
		3/A	9/B	$+\infty$
			7/C	$+\infty$
				9/D

Путь $A \rightarrow E$: $A \xrightarrow{Prev[C]} C \xrightarrow{Prev[D]} D \xrightarrow{Prev[E]} E$

Утверждение В в дереве $1 \leq \text{центров} \leq 2$



Доказательство Если убрать у дерева все висячие вершины, максимальные расстояния уменьшатся на 1



Если продолжать убирание висячих вершин, придём к одной из ситуаций:

□ — центр;

□ —□ — центры



2.11 Алгоритм Флойда

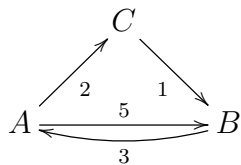
Дан взвешенный граф $G = (V, E)$

Вернуть "таблицу" $d(v, u)$, содержащую расстояния между всеми парами вершин

Составляем матрицу d_0 :

$$d_0(u, u) = 0$$

$$d_0(u, v) = \begin{cases} \infty, & \text{если нет ребра } u - v \\ w(u, v), & \text{если есть ребро } u - v \end{cases}$$



d_0	A	B	C
A	0	5	2
B	3	0	∞
C	∞	1	0

for $k \in V$

 for $u \in V$

 for $v \in V$

 if $d(u, v) > d(u, k) + d(k, v) \Rightarrow d(u, v) = d(u, k) + d(k, v)$

Пример:

$K = A$

	A	B	C
A	0	5	2
B	3	0	5
C	∞	1	0

$K = B$

	A	B	C
A	0	5	2
B	3	0	5
C	4	1	0

$K = C$

	A	B	C
A	0	3	2
B	3	0	5
C	4	1	0

Корректность

Утверждение После шага $Kd(u, v) = \text{mind}(\text{пути})$

$u \longrightarrow \text{пути от } 1 \text{ до } k \longrightarrow v$

Доказательство по индукции для K

База: $K = 0$

$d(u, v) = \min(u \longrightarrow \text{нет} \longrightarrow v)$

Действительно, сначала d содержит длины рёбер

Переход $k \rightarrow k + 1$:

$u \longrightarrow \text{вершины от } 1 \text{ до } k+1 \longrightarrow v$

□ есть оптимальный путь $u \longrightarrow \text{вершины от } 1 \text{ до } k+1 \longrightarrow v$

- 1) В нём нет вершины $k + 1 \Rightarrow$ его длина $= d(u, v)$
- 2) В нём есть вершина $k + 1 \ u \longrightarrow 1 \dots k \longrightarrow k+1 \longrightarrow 1 \dots k \longrightarrow v$

В случае 2) его длина $= d(u, k + 1) + d(k + 1, v)$

Это является ровно проверкой из цикла выше

Наименьший вариант запишется в d

В конце $d(u, v) = \min(u \dots \forall \text{ вершины } \dots v) = \text{dist}(u, v)$

■

Замечание Чтобы восстановить путь, можно ввести массив *through* :

$\text{if } d(u, v) > d(u, k) + d(k, v) \Rightarrow d(u, v) = d(u, k) + d(k, v)$

$\text{through}(u, v) = k$

Для восстановления пути:

$A \xrightarrow{\text{th}(A, I)} I \xrightarrow{\text{th}(I, B)} B$

$\underbrace{\hspace{10em}}_{\text{th}(A, B)}$

И так далее ...

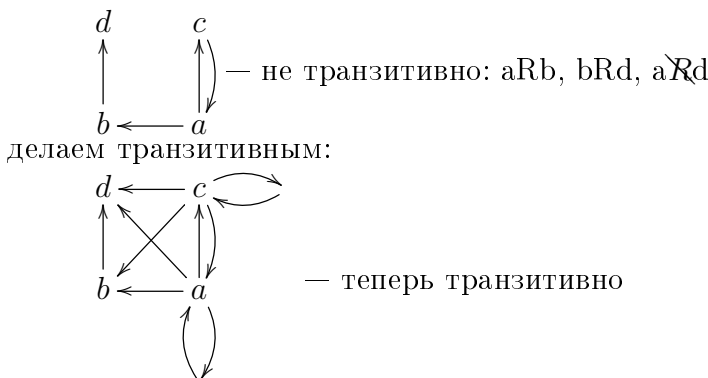
Если ребро (X, Y) не записано \Rightarrow ребро $X - Y$ — оптимальный путь

Замечание

Алгоритм Флойда ищет транзитивные замыкания бинарных отношений

- R — бинарное отношение на M
- \bar{R} — транзитивное замыкание R , если
- 1) $\bar{R} \supset R$
- 2) \bar{R} — транзитивно
- 3) $\forall \tilde{R} : \bar{R} \supset \tilde{R} \supset R$ — не транзитивно

Пример:



Теорема

- R — бинарное отношение на M
- $G = (M, R)$ — граф отношения
- Тогда \bar{R} — это $x\bar{R}y \Leftrightarrow$ есть путь $x \longrightarrow y$ в G

Доказательство

- 1) $\bar{R} \supset R$, так как если $xRy \Rightarrow$ есть путь из 1 ребра $\Rightarrow x\bar{R}y$
 - 2) \bar{R} — транзитивно, так как $x\bar{R}y, y\bar{R}z \Rightarrow x\bar{R}z$ (путь $x \longrightarrow z$ тоже есть)
 - 3) □ $\tilde{R} \supset R, \tilde{R}$ — транзитивно
 - есть путь x в y : $x \longrightarrow \dots \longrightarrow x_n \longrightarrow \dots \longrightarrow y$
- $$\left. \begin{matrix} xRx_1 \Rightarrow x\tilde{R}x_1 \\ x_1Rx_2 \Rightarrow x_1\tilde{R}x_2 \end{matrix} \right\} x\tilde{R}x_2 \Rightarrow x\tilde{R}x_n \Rightarrow x\tilde{R}y$$
- $\Rightarrow \tilde{R} \supset \bar{R} \supset R$



Приминение алгоритма Флойда к графу $G = (M, R)$

$$d_0(x, y) = \begin{cases} 1, & \text{если } xRy \\ \infty, & \text{если } x \not R y \end{cases}$$

Замыкание, $x\bar{R}y$, если $d(x, y) < \infty$

На практике:

Алгоритм транзитивного замыкания

$\bar{R} = R$

for $k < M$

 for $x < M$

 for $y < M$

 if $xRy \ \& \ kRy$

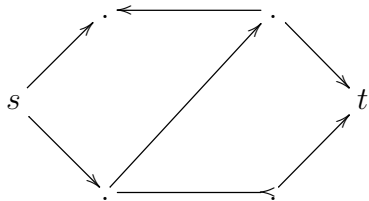
$\bar{R} \leftarrow (x, y)$, то есть сделать $x\bar{R}y$

2.12 Поток в сетях

Определение Сеть — ориентированный граф $G = (V, E)$, $s \in V, t \in V$

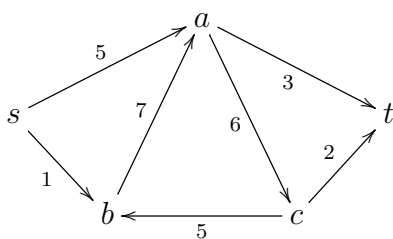
$\neq \exists e = (u, s)$ — ничего не входит

$\neq \exists e = (t, u)$ — ничего не выходит



$c : E \rightarrow \mathbb{N}$ — пропускная способность рёбер

Пример:



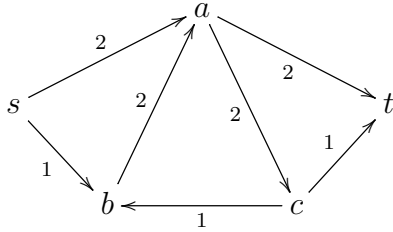
Определение Поток f в сети G — это

$f : E \rightarrow \mathbb{R}$:

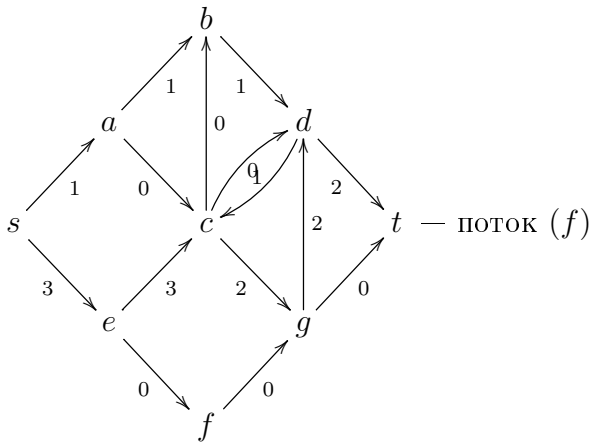
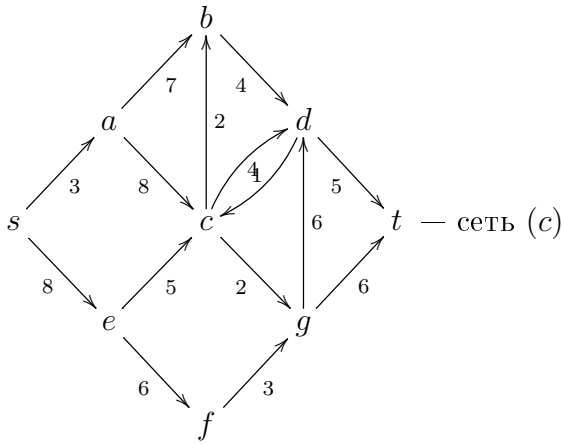
1) $0 \leq f(e) \leq c(e)$

2) $\forall u \neq s, t : \sum_{e=(v,u) \in E} f(e) = \sum_{e=(u,v) \in E} f(e)$

Пример 1:

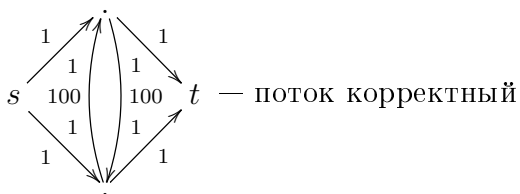


Пример 2:



$$0 \leq f(e) \leq c(e)$$

Пример 3:



Величина потока в примере 1: 4, в примере 2 — 2

Теорема

Дана сеть $(G(V, E), c)$, поток f на G

Тогда $\sum_{u:e=(s,u)} f(e) = \sum_{u:e=(u,t)} f(e)$

Рассмотрим $\sum_{e \in E} f(e) =$

$$\sum_{v \in V} \sum_{e:e=(u,v)} f(e) = \underbrace{\sum_{e:e=(u,s)} f(e)}_0 + \sum_{e:e=(u,t)} f(e) + \sum_{v \in V \setminus \{s,t\}} \sum_{e:e=(u,v)} f(e) =$$

$$= \text{вытекает} + \sum_{v \in V \setminus \{s,t\}} \sum_{e:e=(v,u)} f(e) =$$

$$= \text{вытекает} + \sum_{v \in V} \sum_{e:e=(v,u)} f(e) - \sum_{e:e=(s,u)} f(e) - \sum_{e:e=(t,u)} f(e) =$$

$$= \text{вытекает} + \sum_{v \in V} \sum_{e:e=(v,u)} f(e) - \text{втекает} - 0 =$$

$$= \text{вытекает} - \text{втекает} + \sum_{e \in E} f(e) \Rightarrow$$

$$\sum_{e \in E} f(e) = \text{вытекает} - \text{втекает} + \sum_{e \in E} f(e)$$

$$\cancel{\sum_{e \in E} f(e)} = \text{вытекает} - \text{втекает} + \cancel{\sum_{e \in E} f(e)} \Rightarrow \text{вытекает} = \text{втекает}$$

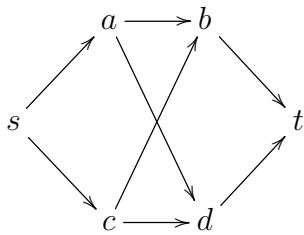
$$\sum_{e:e=(u,t)} f(e) = \sum_{e:e=(s,u)} f(e)$$

Эта величина называется величиной потока $w(f)$

■

Определение Разрез в сети $(G(V, E), c)$

Разрез $G(V_1, V_2) : s \in V_1, t \in V_2, V_1 \cup V_2 = V, V_1 \cap V_2 = \emptyset$

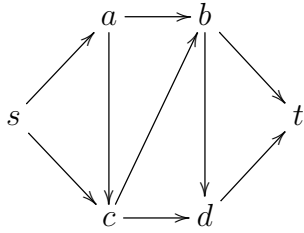


Например, $V_1 = \{s, a, c\}, V_2 = \{b, d, t\}$

Определение Рёбра разреза E_c — все рёбра разреза, которые идут из V_1 в V_2 или наоборот

E_c^+ — прямые рёбра разреза (из V_1 в V_2)

E_c^- — обратные рёбра разреза (из V_2 в V_1)



$$E_c^+ = \{s, a, b\}$$

$$E_c^- = \{c, d, t\}$$

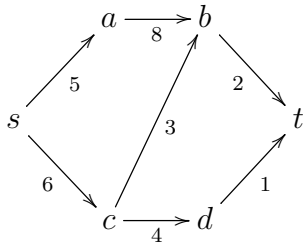
Прямое $E_c^+ = \{sc, ac, bd, bt\}$

Обратное $E_c^- = \{cb\}$

$$E_c = E_c^+ \cup E_c^-$$

Определение Величина разреза = $\sum_{e \in E_c^+} c(e)$

Обозначение $c(G)$



$$V_1 = \{s, a, c\}, V_2 = \{b, d, t\}$$

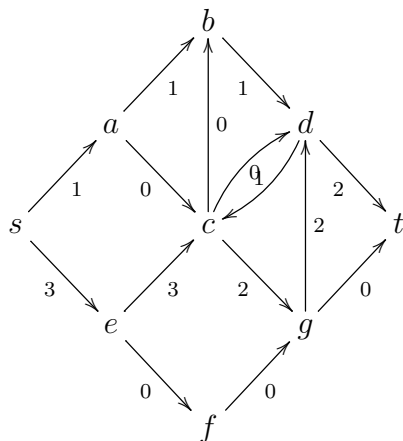
$$C(V_1, V_2)$$

$$c(C) = 8 + 3 + 4 = 15$$

Утверждение Путь есть сеть $(G(V, E), c)$, поток f , разрез $C(V_1, V_2)$

$$\text{Тогда } w(f) = \sum_{e \in E_c^+} f(e) - \sum_{e \in E_c^-} f(e)$$

Пример:



$$V_1 = \{s, e, f, g\}, V_2 = \{a, b, c, d, t\}$$

$$w(f) = 1 + 3 = 2 + 2 = 4$$

$$\sum_{e \in E_c^+} f(e) = 1 + 3 + 0 + 2 = 6$$

$$\sum_{e \in E_c^-} f(e) = 2$$

Действительно, $4 = 6 - 2$

Доказательство

Посчитаем сумму $\sum_{v \in V_1} (\sum_{e: e=(u,v)} f(e) - \sum_{e: e=(v,u)} f(e))$:

1) Для $\forall v \in V_1 \setminus \{s\}$ внутренняя $\sum - \sum = 0$

Для $v = s$: $w(f) = \sum_{e: e=(s,u)} f(e)$

$$2) \sum_{e: e=(u,v), u \in V_1, v \in V_1} (f(e) - f(e)) + \underbrace{\sum_{e \in E_c^+} f(e) + \sum_{e \in E_c^-} [-f(e)]}_{\text{см. условие}}$$

условия ■

Обозначение $w(C, f)$ — величина (размер) потока через разрез

$$= \sum_{e \in E_c^+} f(e) - \sum_{e \in E_c^-} f(e)$$

Замечание $\forall C$ $w(f) = w(C, f)$ — по Теореме

Замечание Будем решать задачу о максимальном потоке в сети (найти $f : w(f) \rightarrow \max$)

Утверждение Дано: $(G(V, E), c)$ — сеть, C — разрез

Тогда $w(f) \leq c(C)$

Доказательство

$$w(f) = w(C, f) = \sum_{e \in E_c^+} f(e) - \sum_{e \in E_c^-} f(e) \leq \sum_{e \in E_c^+} f(e) \leq \sum_{e \in E_c^+} c(e) = c(C)$$

$\Rightarrow w(f) \leq c(C)$ ■

Следствие

В сети G : $w(f_{max}) \leq c(C_{min})$

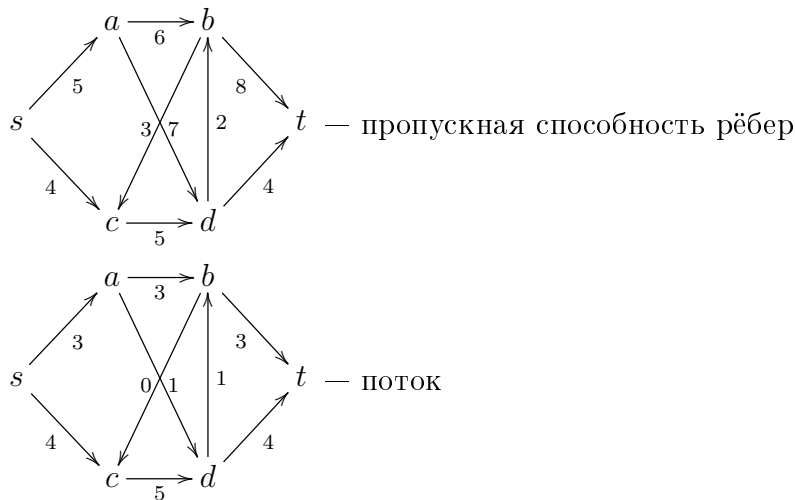
$w(f_{max}) = \max w(f)$ (f — поток)

$c(C_{min}) = \min c(C)$ (C — разрез)

2.13 Теорема Форда — Фалкерсона

В сети $(G(V, E), c)$, $c(e) \in \mathbb{N}$: $w(f_{max}) = c(C_{min})$

(для простоты считаем, что пропускные способности — целые)



Определение Дополнительный граф для потока

\bar{G} имеет $\bar{V} = V$

\bar{E} :

$e = (u, v)$

Если $f(e) < c(e)$

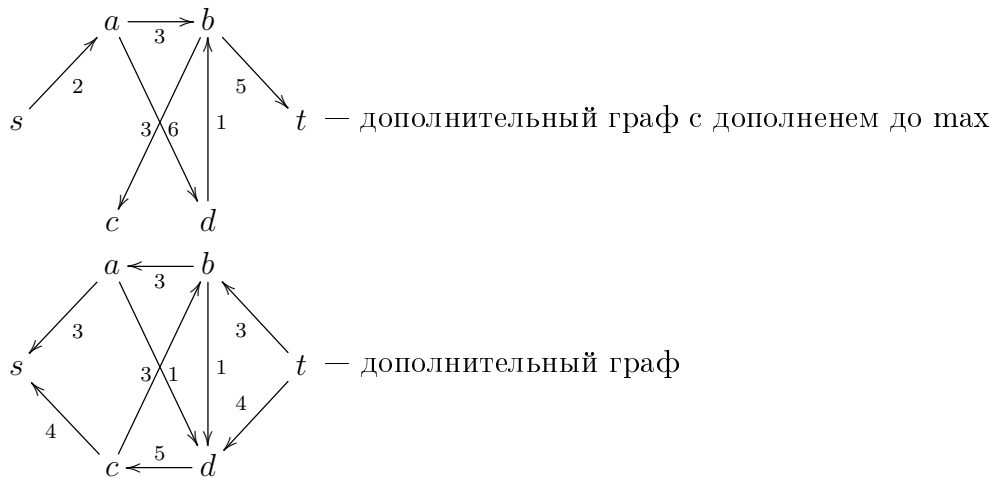
То есть $\bar{e} = (\bar{u}, \bar{v})$

$g(\bar{e}) = cA(e) - f(e)$

Если $0 < f(e)$: $e = (u, v)$

То есть $\tilde{e} = (\bar{v}, \bar{u})$

$g(\tilde{e}) = f(e)$



Доказательство Начнём с 0-го потока и будем его постепенно увеличивать

Построим дополнительный граф \bar{G} и найдём в нём путь из s в t :

$$s \longrightarrow \cdot \longrightarrow \dots \longrightarrow \cdot \longrightarrow t$$

Найдём $\min g(e)$ на этом пути

□ это x

Вычтем в дополнительном графе x на каждом ребре:

$$1) \cdot \longrightarrow \cdot \longrightarrow \xrightarrow{c(e)-f(e) \rightarrow c(e)-f(e)-x} \cdot \quad \bar{f}(e) := f(e) + x$$

$$2) \cdot \longrightarrow \cdot \longrightarrow \xrightarrow[\text{был } f(e)]{f(e)-x} \cdot \quad \bar{f}(e) := f(e) - x$$

Поймём:

- 1) новый поток \bar{f} остался потоком
- 2) величина потока увеличилась на x

Проверим, что это поток:

$$0 \leq \bar{f}(e) \leq c(e)$$

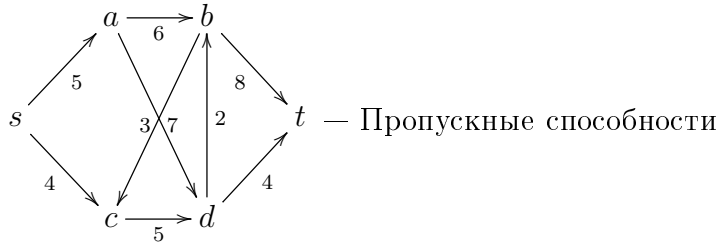
Уменьшаем по обратному, увеличиваем по прямому пути

$$c(e) - (f(e) + x) \geq 0$$

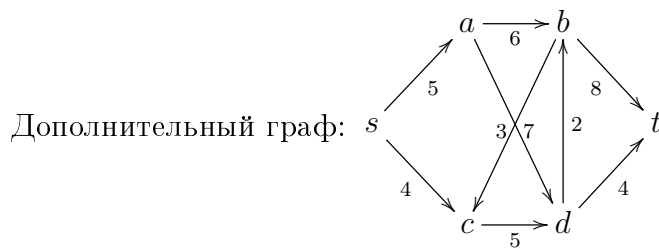
В вершинах верно: $\sum_{\text{входящих}} = \sum_{\text{исходящих}}$

Итого, \bar{f} — поток

Пример:

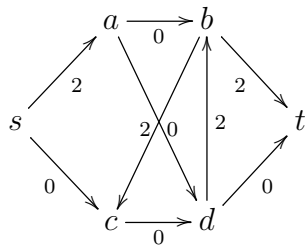


Строим пути
Сначала поток = 0

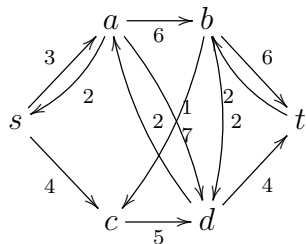


Ищем путь из s в t : $s \xrightarrow{5} a \xrightarrow{3} d \xrightarrow{2} b \xrightarrow{8} t$ — $\min 2$
Добавляем к потоку +2 на каждое из этих рёбер

Поток

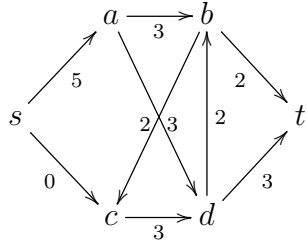


Дополнительный граф

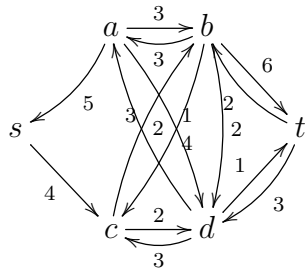


Ищем путь из s в t :
 $s \xrightarrow{3} a \xrightarrow{6} b \xrightarrow{7} c \xrightarrow{5} d \xrightarrow{4} t$ — $\min 3$

Поток



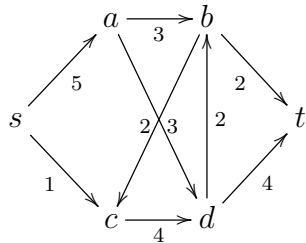
Дополнительный граф



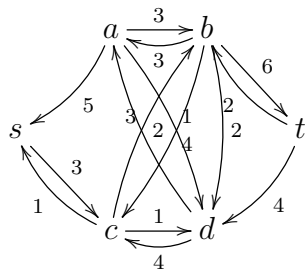
Ищем путь из s в t :

$$s \xrightarrow{4} c \xrightarrow{2} d \xrightarrow{1} t - \min 1$$

Поток



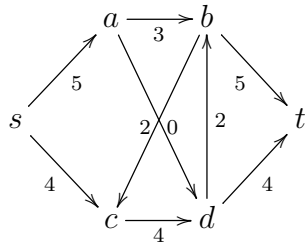
Дополнительный граф



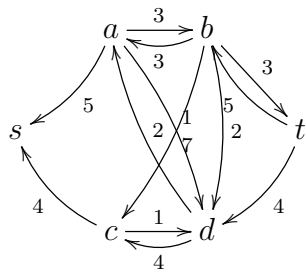
Ищем путь из s в t :

$$s \xrightarrow{3} c \xrightarrow{3} b \xrightarrow{6} t - \min 3$$

Поток



Дополнительный граф



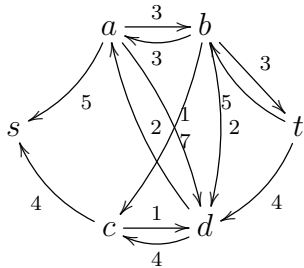
Больше пути из s в t нет

Доказательство теоремы Форда — Фалкерсона *Продолжение*

Почему если пути нет, то поток оптимальный?

$\square V_1$ — вершины, достижимые из s по рёбрам дочернего графа,

$V_2 = V \setminus V_1$



В данном случае $V_1 = \{s\}, V_2 = \{a, b, c, d, t\}$

$t \in V_2$, поскольку нет пути из s в t

Получаем разрез C

В исходном графе есть рёбра из V_1 в $V_2 = E_c^+$

Этих рёбер нет в дополнительном графе

В дополнительном графе веса $= c(e) - f(e) = 0$

Течёт ли что-нибудь в обратном направлении? Нет, иначе V_1 неверен

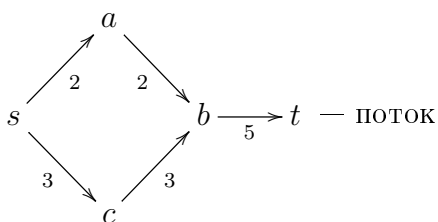
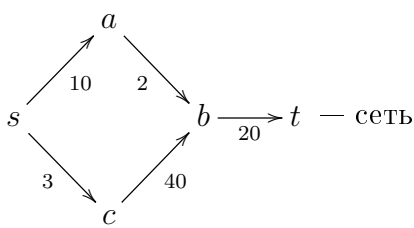
$$c(C) = \sum_{e \in E_c^+} c(e) = \sum_{e \in E_c^+} f(e) = \sum_{e \in E^+} f(e) - \underbrace{\sum_{e \in E^-} f(e)}_0 = c(f)$$

Итого, мы нашли разрез $C : c(C) = c(f)$

Ранее было показано, что $c(c) \geq c(f)$ ($\forall C$ — разрез, $\forall f$ — поток)

Получается: C — минимальный разрез, f — максимальный поток

Пример:



При $V_1 = \{s, a\} : c(C) = 2 + 3 = 5$ — минимальный разрез

При $V_1 = \{a, b, t\} : c(C) = 10 + 40 = 50$

Поток = 5 — максимальный поток

Замечание Метод Форда — Фалкерсона строит минимальный разрез и максимальный поток

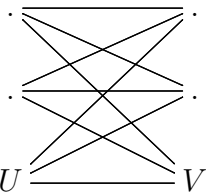
Утверждение Если каждый раз искать путь с минимальным количеством рёбер, то время поиска максимального потока $\sim V^2E$

Без доказательства

Утверждение Для плоской сети (без пересечения рёбер) эффективно искать верхние пути

2.14 Задача о паросочетаниях

Дан двудольный граф $G(U, V, E)$



Определение Паросочетание в G — это $P \subset E$, где рёбра из P не имеют общих вершин

Пример:

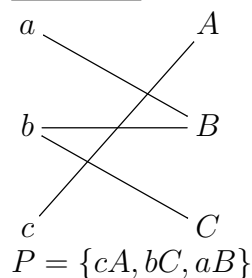


U ————— V

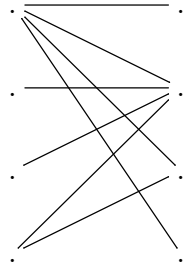
Определение Максимальное паросочетание — $P \subset E$:

$|P|$ — максимальное из возможного

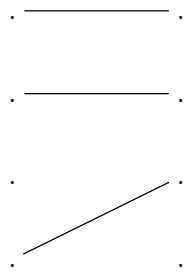
Пример:



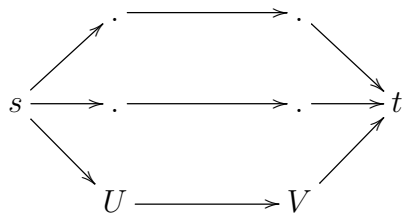
Пример:



4 паросочетания построить невозможно, 3 можно, например:



Сводим задачу о максимальном паросочетании к задаче о потоке:



На рёбрах $U - V$ ставим направление слева направо
 $c(e) = 1$

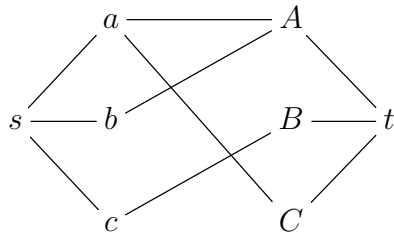
Утверждение Каждому потоку (из $f = 0, 1$) соответствует паросочетание

Доказательство Рёбра с $f(e) = 1$ — рёбра паросочетания
 \rightarrow поток: только одно из рёбер = 1

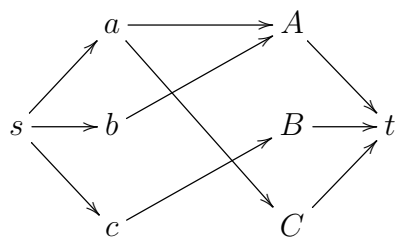
\leftarrow паросочетанию соответствует поток, где $f(e) = 1$, для рёбер паросочетания

Следствие Размер максимального потока равен размеру максимального паросочетания

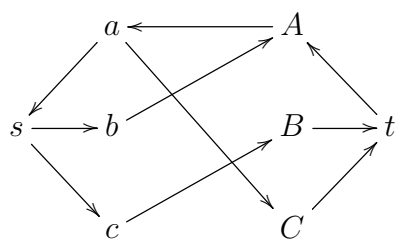
Строим паросочетание методом Форда — Фалкерсона



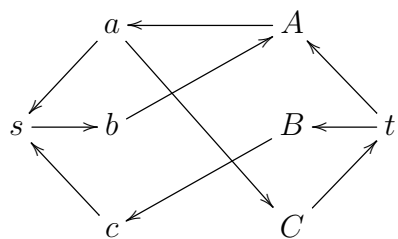
Строим дополнительный граф, но без чисел (всегда 1)



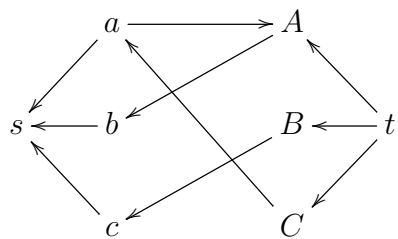
$$s \longrightarrow a \longrightarrow A \longrightarrow t$$



$$s \longrightarrow a \longrightarrow c \longrightarrow B \longrightarrow t$$

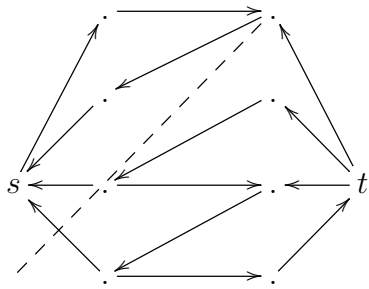
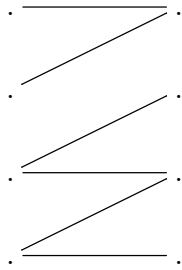


$$s \longrightarrow b \longrightarrow A \longrightarrow a \longrightarrow C \longrightarrow t$$

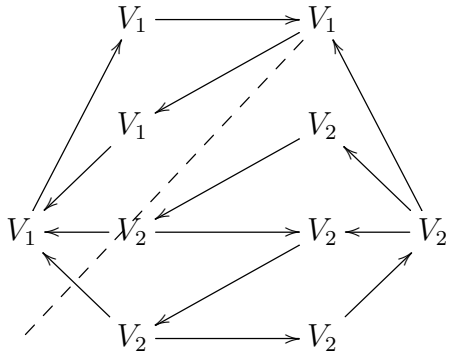


Ответ: aC, bA, cB (рёбра, ориентированные от t к s)

Пример:

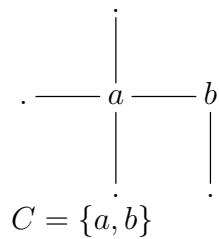


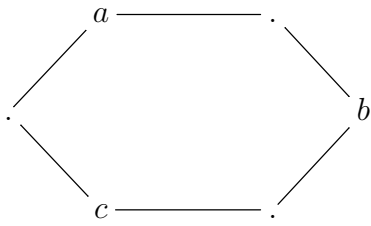
Пунктирная линия (разрез) разделяет 2 группы вершин: V_1 и V_2



2.15 Задача о максимальном контролирующем множестве

Определение $\square G = (V, E); C \subset V$ — контролирующее множество, если $\forall \underbrace{e}_{=(u,v)} \in E : u \in C$ или $v \in C$

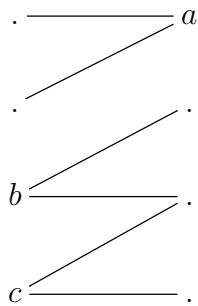




$$C = \{a, b, c\}$$

Замечание $C = V$ — всегда контролирующее множество

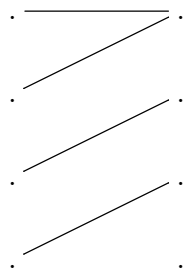
Задача: найти контролирующее множество минимального размера
(будем решать для двудольного графа)



Минимальное $C = \{a, b, c\}$

Утверждение В двудольном графе $G = (U \cup V, E) \sqsupset C$
— контролирующее множество, $\sqsupset P$ — парасочетание: тогда $|C| \geq |P|$

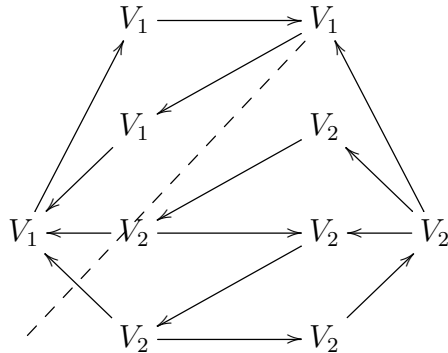
Доказательство



У каждого ребра $e = (u, v) \in P$ есть вершина : $u \in C$ или $v \in C$ ■

Утверждение $\sqsupset G = (U \cup V, E)$ — двудольный граф
Размер максимального парасочетания равен размеру минимального
контролирующего множества

Доказательство Построим максимальное парасочетание по
алгоритму Форда — Фалкерсона и рассмотрим разрез



$$|u| = x; |v| = y; |U \cap V_1| = a; |V \cap V_1| = b$$

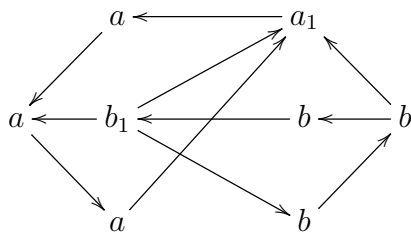
$$c(V_1, V_2) = \sum_{e=(u,v), u \in V_1, v \in V_2} 1 = \underbrace{(x-a)}_{\text{из } S} + \underbrace{b}_{\text{в } T} + \underbrace{n}_{\text{рёбра из } V_2 \text{ в } V_1}$$

Итого, $m = x - a + b + n (\leq m) \leq x - a + b + m \Rightarrow x - a + b \geq 0$

Возьмём в качестве контролирующего множества

$$C = (U \setminus V_1) \cup (V \cap V_1) = (U \cap V_2) \cup (V \cap V_1)$$

Есть ли ребро из $(V \cap V_2)$ в $(U \cap V_1)$?



$$V_1 = \{a\}, V_2 = \{b\}$$

$$\text{Контролирующее множество} = \{a_1, b_1\}$$

Контролирующее множество — это $(U \cap V_2) \cup (V \cap V_1)$

\square есть ребро $e = (u, v)$:

1) $\cdot \longleftarrow \cdot \Rightarrow v \in V_1$ **Противоречие**

1) $u \longrightarrow v$

Как попасть в u ?

Только из v

$\cdot \longrightarrow u \longleftarrow \cdot$ — невозможно

Значит, в этом минимальном разрезе нет рёбер между $(U \cap V_2)$ и $(V \cap V_1)$

Вывод 1: — контролирующее множество $(U \cap V_2)$ и $(V \cap V_1)$

Вывод 2:

$$|P| = C(V_1, V_2) = \sum_{e=(u,v), u \in V_1, v \in V_2} 1 = \underbrace{(x-a)}_{\text{из } S} + \underbrace{b}_{\text{в } T} + \underbrace{0}_{\text{рёбра из } V_2 \text{ в } V_1} = |C|$$



2.16 Поиск в глубину, в ширину

- 1) Структура данных для хранения вершин D — стек или очередь
 $v \rightarrow D$ — положить v в D
 $v := D$ — посмотреть
 $D \rightarrow v$ — убираем элемент из структуры данных
 Стек: первый вошёл, последний вышел
 Очередь: первый вошёл, первый вышел

	Стек	Очередь
$a \rightarrow D$	a	a
$b \rightarrow D$	ba	ba
$c \rightarrow D$	cba	cba
$:= D$	c	a
$D \rightarrow$	ba	cb
$:= D$	b	b

Поиск в ширину (D — очередь)

Поиск в глубину (D — стек)

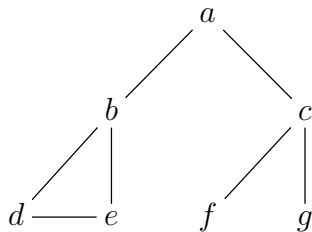
$D \in v_0$ — начальная вершина (были $(v_0) = 1$)

Пока D не пусто

$u := D$

если есть ребро (u, v) : мы ещё были в v , тогда $v \rightarrow D$, были $(v) = 1$

Иначе $D \rightarrow v$



Поиск в глубину:

a
ba
dba
edba
dba
ba
a
ca
fca
ca
gca
ca
a
 \square

Поиск в ширину:

a
ba
cba
cb
dcb
edcb
edc
fedc
gfedc
gfed
gfe
gf
g
 \square

Поиск в глубину (D — стек) и в ширину (D — очередь)

Алгоритм поиска

Дана начальная вершина u

$Used \leftarrow \emptyset$ (обработанные вершины; те, кто был в D)

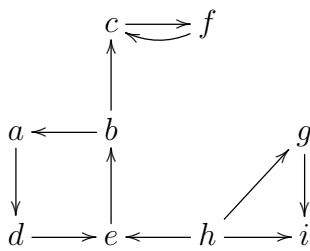
Пока $D \neq \emptyset$

$v := peak D$

Если есть ребро $v - w$, где $w \notin Used$ (ребро дерева поиска)

$D \leftarrow w; Used = Used \cup \{w\}$

Иначе $\leftarrow D$ (убираем вершину из D)



Поиск в глубину из вершины h

Стек

h

he

heb

$heba$

$hebad$

$hebad$

~~$hebad$~~

~~$hebad$~~

$hebc$

$hebcf$

~~$hebcf$~~

~~$hebcf$~~

~~$hebcf$~~

~~$hebcf$~~

h

hg

hgi

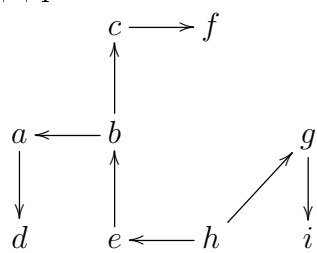
~~hgi~~

~~hgi~~

~~hgi~~

~~hgi~~

Дерево поиска



Поиск в ширину из вершины h

Очередь

h

he

heg

~~heg~~

eg

egb

~~egb~~

gb

gbi

~~gbi~~

bi

bia

$biac$

~~$biac$~~

~~$biac$~~

ac

acd

~~acd~~

cd

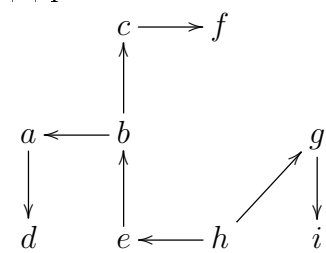
cdf

~~cdf~~

~~cdf~~

~~cdf~~

Дерево поиска



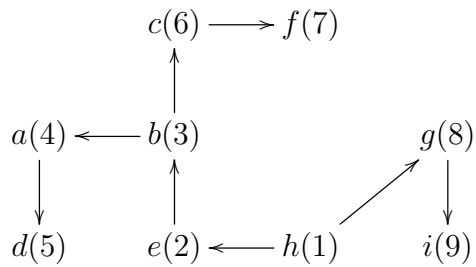
Введём нумерацию:

$f(u)$ — номер, какой по счёту вершина попала в D

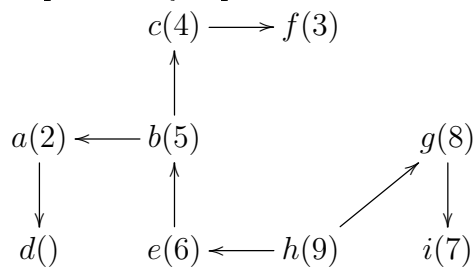
$b(u)$ — обратный номер (номер, какой по счёту вершина ушла из D)

Поиск в глубину

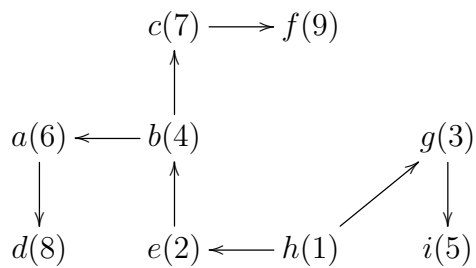
Прямая нумерация



Обратная нумерация



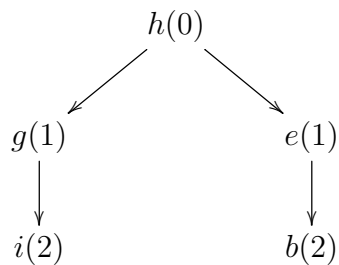
Замечание В случае поиска в ширину $f(u) = b(u)$



Утверждение Поиск в ширину перебирает вершины в том же порядке, что и Алгоритм Дейкстры (веса рёбер = 1)

Действительно, добавление вершинны в D — это релаксация рёбер $v - w$

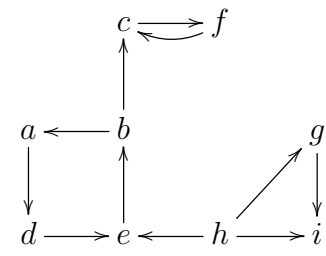
Удаление из D — удаление вершины с минимальным расстоянием



Полный поиск в глубину (Depth-first search, DFS)

Пока есть непосещённая вершина u
поиск в глубину (u)

Пример (прямая нумерация):

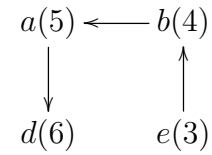


dfs(c)

$c(1) \longrightarrow f(2)$

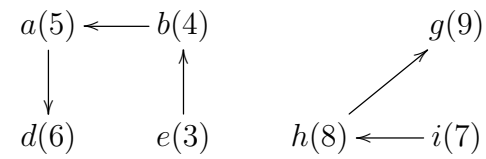
dfs(e)

$c(1) \longrightarrow f(2)$



dfs(i)

$c(1) \longrightarrow f(2)$



Утверждение Пусть G — ориентированный граф без циклов
Пусть есть путь $u - v$ (Нет пути $v - u$, так как нет циклов)
Тогда после полного обхода в глубину (DFS)
 $b(u) > b(v)$

Доказательство

Делаем DFS : куда попали раньше?

1) сначала попали в u

$u \longrightarrow \dots \longrightarrow v$

В стеке будет $u \dots v$

\Rightarrow сначала из стека уйдёт v , потом $u \Rightarrow b(u) > b(v)$ ■

2) сначала v

$u \longrightarrow \dots \longrightarrow v \longrightarrow \dots$

Мы не можем попасть в u , потому что нет циклов

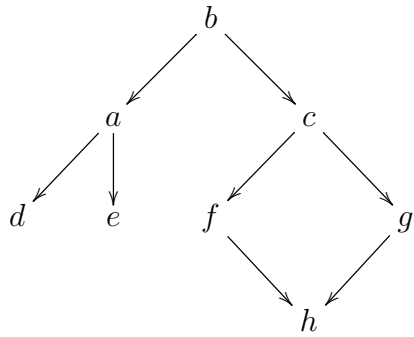
\Rightarrow мы закончим просмотр, так и не попав в u

\Rightarrow номер $b(v)$ присвоится раньше, чем номер $b(u)$ ■

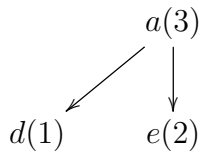
Следствие Алгоритм топологической сортировки

Делаем полный DFS и линейный порядок задаём как $b(u)$

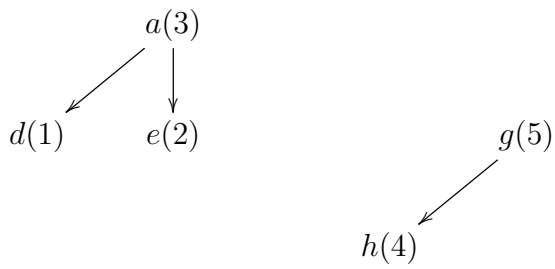
Пример:



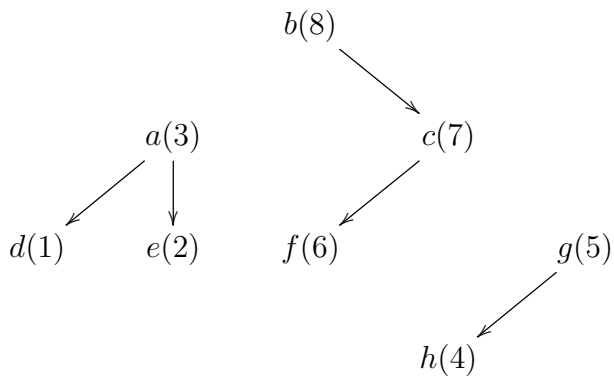
dfs(a)



dfs(g)



dfs(b)



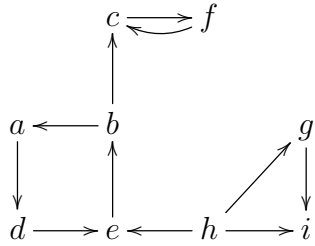
ОТВЕТ: *deahgfcba*

2.17 Компоненты сильной связности

Напомним, что $G(V, E)$ — ориентированный граф.

Введём отношение \leftrightarrow на V :

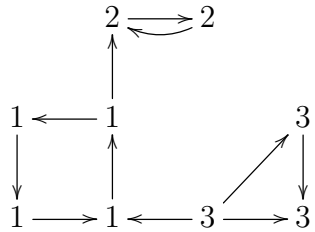
$u \leftrightarrow v :=$ есть путь $u \rightarrow v$ и $v \rightarrow u$



Напомним, что \leftrightarrow — отношение эквивалентности

Классы эквивалентности называются компонентами сильной связности

В данном графе 3 компоненты сильной связности:



Определение \square $G(V, E)$ — ориентированный граф, $G^0(V^0, E^0)$ — граф конденсации, если

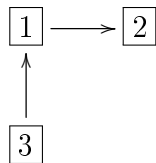
$V^0 = V/\leftrightarrow$ (классы эквивалентности)



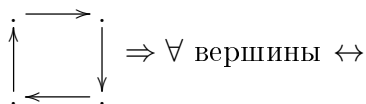
В примере:



$E^0 : u^0 \rightarrow v^0$ есть ребро, если $\exists e = (u, v)$, где $u \in U^0, v \in V^0$



Замечание Граф конденсации G^0 не имеет циклов



Утверждение \square $G(V, E)$ — ориентированный граф, $G^0(V^0, E^0)$ — граф конденсации G

Делаем полный DFS в G

Тогда если в G^0 есть путь из u^0 в v^0 , то $\underbrace{\max}_{u \in U^0} b(u) > \underbrace{\max}_{v \in V^0} b(v)$

Доказательство

Аналогично предыдущему утверждению ■

Следствие **Поиск компонент сильной связности**

1. Полный DFS в G

2. Находим u $b(u) \rightarrow \max$

Делаем DFS по обратным рёбрам G

The End