Комбинаторика и теория графов
Лекционный материал

Посов Илья Александрович

запись конспекта: Штейнберг Эмиль
дата лекции: осенний семестр 2021/2022

1 Бинарные отношения

1.1 Определение

Определение. Пусть m и n — некие непустые множества, R — подмножество произведения множеств $n \times m$, то есть всего множества пар из элементов R. *Бинарным отношением* называется это множество R.

Пример. Если $n = \{a, b, c\}$, то $n \times n = \begin{Bmatrix} aa & ba & ca \\ ba & bb & bc \\ ca & cb & cc \end{Bmatrix}$. Если $m = \mathbb{N}$, то $m \times m$ — бесконечное множество из пар натуральных чисел.

1.2 Обозначение

Бинарное отношение между двумя элементами x и y обозначается $(x, y) \in R$, однако мы будем использовать более привычное обозначение: xRy.

1.3 Примеры бинарных отношений

1. Пусть $m = \mathbb{R}$. Возьмем такие x и y, что $x > y$. Тогда:

   $3R2$, ~~$3R4$~~ — отношение "больше".

2. Пусть $m = \mathbb{R}$. Возьмем такие x и y, что $x \geq y$. Тогда:

   $7R6$, $7R7$, ~~$7R8$~~ — отношение "больше или равно".

3. Пусть $m = \mathbb{R}$. Возьмем такие x и y, что $x = y$. Тогда:

$7R7$, ~~$7R8$~~ — отношение "равно".

Примечание. Далее будем обозначать подмножество R более привычно: как выполняемое соотношение, например вместо $3R2$ будет использоваться $3 > 2$.

4. Пусть $m = \mathbb{R}$. Зададим бинарное отношение "примерно равно"($\approx$) при условии $|x - y| < 1$. То есть, $5.5 \approx 5$, $213.1 \approx 212.11$, ~~$1 \approx 2$~~.

5. Пусть $m = \mathbb{R}$. Зададим бинарное отношение "$\#$", которое будет означать, что $x^2 > y$. То есть, $2\#3$, $4\#15$, ~~$3\#9$~~.

6. Пусть $m = \mathbb{R} \cup \{0\}$. Зададим бинарное отношение делимости ($\vdots$), что будет означать, что найдется такое p, что $x = py$. То есть, $4 \vdots 2$, $100 \vdots 25$, ~~$11 \vdots 4$~~.

7. Пусть m — множество прямых на плоскости. Зададим бинарное отношение параллельности ($\parallel$), которое будет означать, что прямые l_1 и l_2 либо не пересекаются, либо совпадают. То есть $l_1 \parallel l_2$ — прямые параллельны (не пересекаются или совпадают), ~~$l_1 \nparallel l_2$~~ — прямые не параллельны (пересекаются).

8. Пусть m — множество прямых на плоскости. Зададим бинарное отношение перпендикулярности ($\perp$), которое будет означать, что прямые l_1 и l_2 пересекаются под прямым углом ($90°$). То есть $l_1 \perp l_2$ — прямые перпендикулярны (пересекаются под прямым углом), ~~$l_1 \perp l_2$~~ — прямые не перпендикулярны (пересекаются не под прямым углом).

9. Пусть m — студент в ЛЭТИ. Зададим бинарное отношение "$\triangleright$", которое будет означать, что средний балл студента x больше, чем студента y. То есть, $x \triangleright y$ — средний балл больше, ~~$x \triangle y$~~ — средний балл меньше или равен.

10. Пусть m — пользователь Одноклассников. Зададим бинарное отношение "$\heartsuit$", которое будет означать, что пользователь x находится в друзьях у пользователя y. То есть, $x \heartsuit y$ — пользователи в друзьях, ~~$x \heartsuit y$~~ — пользователи не в друзьях.

Бинарные отношения можно задавать на любом множестве, главное, чтобы сопоставляемые элементы были элементами из одного множества.

1.4 Свойства

1. **Определение.** Бинарное отношение R называют *рефлексивным*, если $\forall x \in m$ выполняется условие xRx.

   Замечание. Если подобрать один контрпример, то отношение не является рефлексивным. Это является примером того, что обычно обратное доказать проще, чем доказать прямо.

   Пример. "$\geq$"— рефлексивно, "$=$"— рефлексивно, "$>$"— не рефлексивно.

2. **Определение.** Бинарное отношение R называют *антирефлексивным*, если $\forall x \in m$ выполняется условие $\cancel{xRx}$.

   Замечание. Существуют отношения не рефлексивные и не антирефлексивные, однако одновременно рефлексивных и нерефлексивных не может существовать (доказательство очевидно).

   Пример. "$>$"и "$<$"— антирефлексивны, "$\#$"(из примера 5) — не рефлексивно и не антирефлексивно.

3. **Определение.** Бинарное отношение R называют *симметричным*, если $\forall x, y \in m$ выполняется условие $xRy = yRx$.

   Пример. "$=$"— симметрично, "$>$"и "$<$"— не симметрично.

4. **Определение.** Бинарное отношение R называют *антисимметричным*, если $\forall x \neq y \in m$ выполняется условие $xRy = \cancel{yRx}$.

   Пример. "$\geq$"и "$\leq$"— асимметричны.

5. **Определение.** Бинарное отношение R называют *асимметричным*, если $\forall x, y \in m$ выполняется условие $xRy = \cancel{yRx}$.

   Пример. "$>$"и "$<$"— асимметричны.

   Утверждение. *Множество R – асимметрично означает то, что оно антисимметрично и и антирефлексивно.*

   Замечание. $\square$ (пустое отношение) — асимметрично (не содержит ни одной пары на множестве R).

6. **Определение.** Бинарное отношение R называют *транзитивным*, если $\forall x, y, z \in m$ при $xRy,\ yRz$ выполняется условие xRz.

Пример. "$>$, $\geq$"и "$<$, $\leq$"— транзитивны, $\perp$ — не транзитивно.

Примечание. Свойства так же зависят от выбранного множества. Например, если взять делимость из примера 6, то оно является антисимметричным, однако для целых чисел ($x \in \mathbb{Z}$) отношение не будет являться антисимметричным ($4 \vdots -4 = -4 \vdots 4$).

1.5 Отношение эквивалентности

Определение. Отношение R называется *отношением эквивалентности*, если R — рефлексивно, симметрично и транзитивно.

Пример. Отношение равенства ("=") является отношением эквивалентности на любом множестве.

1. $\forall x\ x = x$ — рефлексивно.
2. $\forall x, y\ x = y \Leftrightarrow y = x$ — симметрично.
3. $\forall x, y, z\ x = y = z$ — транзитивно.

Также отношения $\parallel$, $\equiv_n$ являются отношениями эквивалентности.
Отношение "больше или равно" ("$\geq$") не отношение эквивалентности.

Проверим следующее отношение:
Отношение $\uparrow$ на множестве $\mathbb{N}$: выполняется условие $x \uparrow y$, если x и y содержат одинаковое количество цифр.

Пример. $2 \uparrow 5,\ 12 \uparrow 42,\ \cancel{3 \uparrow 33}$.

1. $\forall x\ x \uparrow x$ — рефлексивно.
2. $\forall x, y\ x \uparrow y \Leftrightarrow y \uparrow x$ — симметрично.
3. $\forall x, y, z\ x \uparrow y \uparrow z$ — транзитивно.

Отношение $\uparrow$ является отношением эквивалентности.

1.6 Классы

Определение. Если R является *отношением эквивалентности* на множестве $\mathbb{M}$, и $x \in \mathbb{M}$, то $\mathbb{M}_x$ — это набор таких y, что xRy.

Пример. Равенство ("="): $\mathbb{M}_5 = 5$, $\equiv_3$: $\mathbb{M}_2 = \{2, 5, 8, \dots\}$, параллельность ("$\parallel$"): $\mathbb{M}_l$ содержит все прямые, параллельные l.

Утверждение. *Если R — отношение эквивалентности на множестве $\mathbb{M}$, то при $\forall x, y \in \mathbb{M}$ $\mathbb{M}_x = \mathbb{M}_y$ или $\mathbb{M}_x \cap \mathbb{M}_y \neq \emptyset$.*

Доказательство. Если $\mathbb{M}_x \cap \mathbb{M}_y \neq \emptyset$, то существует $z \in \mathbb{M}_x$ и $\mathbb{M}_y$. Тогда xRz и yRz, а по свойству симметричности zRy и по свойству транзитивности xRy.

Теперь проверим, что класс $\mathbb{M}_x = \mathbb{M}_y$. Возьмем $u \in \mathbb{M}_x$, проверим, что $u \in \mathbb{M}_y$. Мы знаем, что из $u \in \mathbb{M}_x$ следует, что xRu, а также и xRy, тогда по симметричности yRx, а по транзитивности yRu, значит $u \in \mathbb{M}_y$, то есть множества одинаковые. □

Следствие. *Если R — отношение эквивалентности на множестве $\mathbb{M}$, тогда $\mathbb{M}$ разбито на несколько классов эквивалентности, которые не пересекаются.*

$$\mathbb{M} = \mathbb{M}_1 \cup \mathbb{M}_2 \cup \dots \cup \mathbb{M}_n$$

Пример. Возьмем $\equiv_3$ на множестве $\mathbb{N}$. То есть:

$$\mathbb{N} = \{0, 3, 6, \dots\} \cup \{1, 4, 7, \dots\} \cup \{2, 5, 8, \dots\}$$

Замечание. Если есть $\mathbb{M} \neq \emptyset$, разбитое на $\mathbb{M}_i \neq \emptyset$, тогда можно ввести отношения R.

$$xRy, \text{ если для } \mathbb{M}_i \; x, y \in \mathbb{M}_i$$

Пример. Рассмотрим все прямые на плоскости — это множество $\mathbb{M}$. Тогда среди всех прямых найдется бесконечное количество параллельных прямых. Здесь, отношение эквивалентности R является параллельность ("$\perp$"), а классом $\mathbb{M}_i$— их направление.

1.7 Отношения порядка

Для простого понимания: имеется в виду не эквивалентность, а степени сравнения — *выше, лучше, сильнее, быстрее, важнее* и т.д.

Определение. Пусть дано бинарное отношение R, которое транзитивно, антисимметрично. Тогда, если оно:

1. Рефлексивно, то R — *нестрогий порядок* ($\preceq$ / $\succeq$).

2. Антирефлексивно, то R — *строгий порядок* ($\prec$ / $\succ$).

Примечание. Введенные понятия:

Если $a \succ b,\ b \succ c$, то $a \succ c$ — транзитивность.
Если $a \succ b$, то $\cancel{b \succ a}$ — антисимметричность.

Пример. Отношение $>$ на $\mathbb{R}$ — строгий порядок, $\geq$ на $\mathbb{R}$ — нестрогий порядок,
$\vdots$ на $\mathbb{N}$ — нестрогий порядок (число всегда делится само на себя).

Определение. Пусть R — строгий или нестрогий порядок. Тогда R называется *линейным порядком*, если $\forall x \neq y$, то xRy или yRx.

Замечание. Если существуют такие x и y, что $\cancel{xRy}$ или $\cancel{yRx}$, то R называют *частичным порядком.*

Пример. Рассмотрим некоторые примеры:

1. $>$ — линейный порядок.

2. $\geq$ – линейный порядок.

3. $\vdots$ — частичный порядок.

Утверждение. *Если $\succ$ — строгий или нестрогий порядок на конечном множестве $\mathbb{M}$ ($\forall \alpha \in \mathbb{M}\ |\alpha| < \infty$). Тогда существует минимальный x, и для $\forall y \neq x$ $\cancel{x \succ y}$.*

Пример. Возьмем отношение $\geq$ на множестве $\{1, 2, 3, 4, 5\}$. Тогда минимальный $x = 1$, так как $\forall y$ $\cancel{1 \geqslant y}$.

Доказательство единственности. Берем x_1 — любой элемент отношения $\succ$ конечного множества $\mathbb{M}$. Если он не минимальный, значит существует другой минимальный $x_2 \neq x_1$, такой что $x_1 \succ x_2$. Если же x_2 не минимальный, то существует другой минимальный $x_3 \neq x_2$, такой что $x_2 \succ x_3$. И так далее...

Если мы не можем найти минимальный элемент, значит по свойству конечности множества $\mathbb{M}$ последний x_i будет являться минимальным (точнее говоря, существующие x_i будут повторяться).

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

Отношение $\succ$ — транзитивно, то есть $x_i \succ x_{j-1} \succ x_i$. А это невозможно по антисимметричности. $\square$

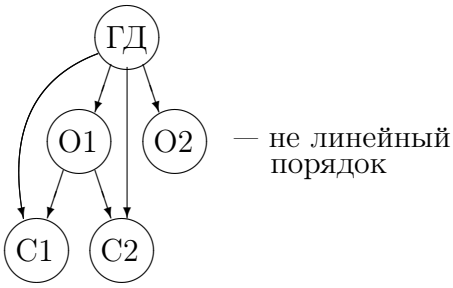
2 Топологическая сортировка

Определение. Отношение R_1 на множестве $\mathbb{M}$ *расширяет* R_2 на $\mathbb{M}$, если $R_2 \subset R_1$.

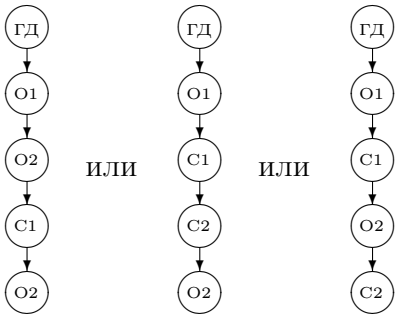
Замечание. R_1 добавляет пары где xRy. То есть из xR_2y следует, что xR_1y.

Теорема (о топологической сортировке)**.** *Если отношение порядка $\succ$ — строгое или нестрогое на конечном множестве $\mathbb{M}$, то существует $\gg$ — отношение линейного порядка на $\mathbb{M}$, такое что $\gg$ расширяет $\succ$.*

Пример. Пусть есть отношение порядка подчинения сотрудников:



— не линейный порядок



$\cdots$

топологическая сортировка

где ГД — генеральный директор, О — начальник отдела, С — сотрудник.

Доказательство. Найдем минимальный элемент отношения $\succ$ (пусть это $x_1 \in \mathbb{M}$) и удалим его из множества. Теперь имеем ограниченное отношение $\succ |_{\mathbb{M}-\{x_1\}}$. Очевидно, что это новое отношение имеет те же свойства, что и изначальное (антисимметрично, транзитивно и рефлексивно/антирефлексивно). В нем тоже есть минимальный элемент x_2, который мы удаляем и получаем ограниченное множество $\succ |_{\mathbb{M}-\{x_1,x_2\}}$. Продолжаем...

В какой-то момент по свойству конечности множество $\mathbb{M} - \{\forall x_i\}$ станет пустым. Итого, имеем последовательность $\{x_1, x_2, \ldots, x_n\}$, где $n = |M|$ — размер исходного множества $\mathbb{M}$.

Вводим новый порядок $x_i \ll x_j$ для $i < j$:

$$x_1 \ll x_2 \ll \cdots \ll x_n$$

Почему $\ll$ расширяет $\prec$? Если $x \prec y$, то x был удален из множества раньше y, следовательно $x \ll y$. $\square$

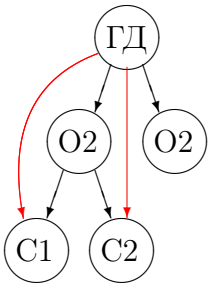
Замечание. Алгоритм поиска минимума и удаления не самый эффективный. Более эффективно будет сделать поиск в глубину и построить обратную нумерацию.

Замечание. Топологическая сортировка — практически важная задача. Как пример зависимостей: нельзя расдать листовки, пока они не напечатаны, при этом нельзя распечатать листовки, пока нет чернил и бумаги.

3 Транзитивное замыкание

По решению задачи топологической сортировки мы расширяли порядок до линейности. Теперь перед нами стоит задача расширить отношение до транзитивности.

Пример. Пусть есть отношение подчиненности:



где ГД — генеральный директор, О — начальник отдела, С — сотрудник.

Черным цветом показана изначальная связь. Мы можем сказать, что ГДRО1 и О1RС1, но отсюда не следует, что ГДRС1.

Для этого в множество необходимо добавить пару ГДRС1, чтобы отношение стало транзитивным (красный цвет). Аналогично для ГДRС2.

Теорема. *Пусть R — отношение на множестве $\mathbb{M}$ и существует такое отношение $\overline{R}$ на том же множестве, что:*

1. *$\overline{R}$ расширяет R ($R \subset \overline{R}$).*
2. *$\overline{R}$ — транзитивно*
3. *$\overline{R}$ — минимальное транзитивное расширение, то есть если $\tilde{R}$ — транзитивное расширение R, то $\tilde{R} \supset \overline{R}$.*

Условное доказательство. Рассмотрим все транзитивные расширения отношения $\{\overline{R_i}\}$ и посчитаем R как пересечение всех $\overline{R_i}$ (берем те ребра, которые есть только у транзитивного расширения).

Пример. Пусть множество $\mathbb{M} = \{a, b, c, d\}$ и на нем есть отношения aRb, bRc, bRd. Мы можем его любым способом достроить до транзитивного (к примеру, достроим отношения aRc, aRd, cRd). Минимальным элементом $\overline{R}$ будет являться пересечение всех таких транзитивных отношений, и оно подходит под все условия:

1. $\overline{R}$ расширяет R (пусть xRy, тогда $\forall \overline{R_i}\ x\overline{R_i}y$, значит $x\overline{R}y$).
2. $\overline{R}$ — транзитивно (пусть $x\overline{R}y$, а $y\overline{R}z$, то $\forall \overline{R_i}\ x\overline{R_i}y, y\overline{R_i}z$, значит $x\overline{R_i}z$, то есть $x\overline{R}z$).
3. $\overline{R}$ — минимальное транзитивное расширение (так как пересечение находится $\forall \overline{R_i}$).

0. Существует ли $\overline{R_i}$?
   Скажем, что R_1 — полное отношение $= \mathbb{M} \times \mathbb{M}$. Получили, что расширить можно в любом случае.

□

4 Графы

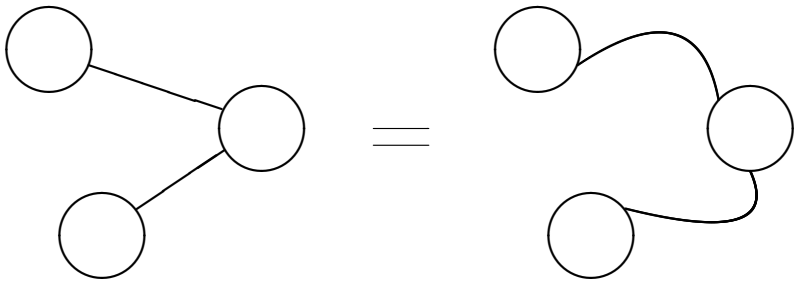
4.1 Неориентированный граф

Определение. *Неориентированный граф* G — это объект $(\mathbb{V}, \mathbb{E})$ от двух аргументов, где $\mathbb{V}$ — множество *вершин графа*, а $\mathbb{E} \subset \{(u, v),\ u, v \in \mathbb{V}\}$ — множество неупорядоченных пар из двух вершин (*ребра графа*). Граф обладает хотя бы двумя вершинами, соединенными между собой.

Замечание. Как рисовать:

1. Вершины обозначаются точками ($\cdot$) или кругами ($\circ$).
2. Ребра обозначаются линиями между двумя вершинами. Важен только факт соединения.

Пример. Граф (важно только наличие соединений, форма определяется фантазией):



Определение. Граф G называется *полным*, если $\forall u, v \in \mathbb{V}\ (u, v) \in \mathbb{E}$.

Примечание. $\mathbb{V}$ — от слова *vertex* (англ. вершина), $\mathbb{E}$ — от слова *edge* (англ. ребро).

Определение. *Размер* (порядок) графа определяется как количество вершин:

$$|G| = |\mathbb{V}| = n$$

Есои количество ребер $|\mathbb{E}| = m$, то иногда говорят, что G — это (n, m)-граф.

Определение. *Степень вершины* $v \in \mathbb{V}$ — это количество ребер, которым она принадлежит.

$$\deg v = |\{(v, u) \mid (v, u) \in \mathbb{E}\}|$$

Определение. *k-регулярным графом* называется граф, все степени вершин которого равны k.

4.2 Путь в графе

Определение. *Путь в графе* — последовательность вершин-ребер

$$v_1, e_1, v_2, e_2, \ldots, v_n$$

Причем, каждый e ведет от вершины v_i к $v_i + 1$.

Пример. a, b, c, d — подразумевается путь в графе от a к b, от b к c, и так далее.

Определение. *Замкнутый путь* — такой путь, если первая и последняя вершина одинаковые.

Определение. *Незамкнутый путь* — такой путь, если первая и последняя вершина не совпадают.

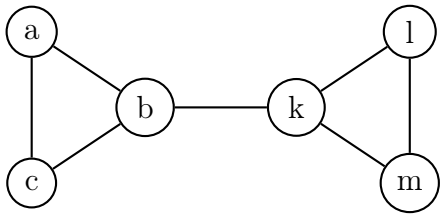
Определение. *Простой путь* — такой путь, который содержит только различные ребра.

Пример. Рассмотрим некоторые примеры путей:

1. b, e, d, c, e — простой путь, так как ребра не повторяются (однако вершина e повторяется).

2. $a, b, c, d, e, c, d, e, b, a$ — непростой замкнутый путь.

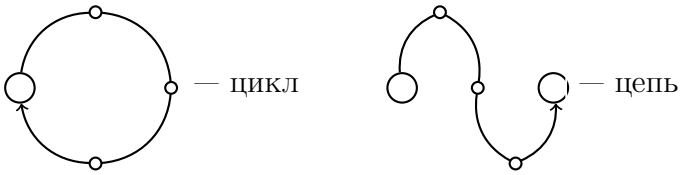
3. Рассмотрим граф G:



   Путь k, b, a, c, b, k — не простой путь, так как ребро $e = (b, d)$ повторяется. А путь b, c, a, b, k, l, m, k — простой, так как ребра не повторяются.

Определение. *Циклом* называется замкнутый путь в графе, все вершины которого разные. *Цепью* называется открытый путь в графе, все вершины которого разные (кроме первой с последней).

Пример. Рассмотрим следующие графы:



Теорема. *Если между вершинами u и v существует путь, то существует и цепь между этими вершинами.*

Доказательство. Пусть есть путь $u, e_1, v_1, e_2, v_2, \ldots, e_n, v$. Рассмотрим все такие возможные пути и возьмем самый короткий. Поймем, что это и есть *цепь*. Представим, что какие то вершины совпали:

$$u \ldots v_i \ldots v_j \ldots v, \ v_i = v_j$$

Тогда среднюю часть можно убрать, и тогда это не самый короткий путь. Противоречие. □

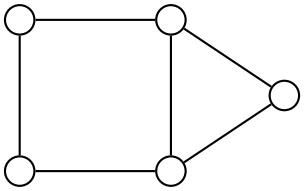
Теорема. *Если есть простой замкнутый путь через ребро e, то есть и цикл через это ребро.*

Доказательство. Аналогично предыдущей теореме, можно найти самый короткий путь, где ребро не повторяется. □

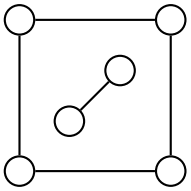
4.3 Связность графа

Определение. Граф G — *связан*, если $\forall u, v \in \mathbb{V}$ существует цепь из u в v.

Пример. Рассмотрим следующие примеры:



связный



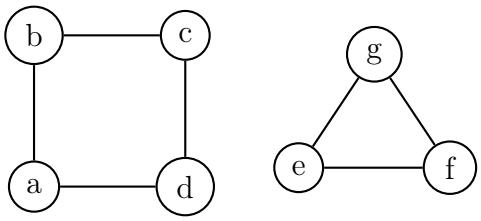
не связный

Введем отношение $\equiv$ на вершинах графа: скажем, что $u \equiv v$, если существует путь из u в v. Проверим, что $\equiv$ — отношение эквивалентности:

1. Рефлексивно — $u \equiv u$ (верно).
2. Симметрично — $u \equiv v$, следовательно $v \equiv u$ (верно).
3. Транзитивно — $u \equiv v$, $v \equiv w$, значит $u \equiv w$ (верно).

Определение. Классы эквивалентности $\equiv$ — *компоненты связности.*

Пример. Пусть имеется граф:



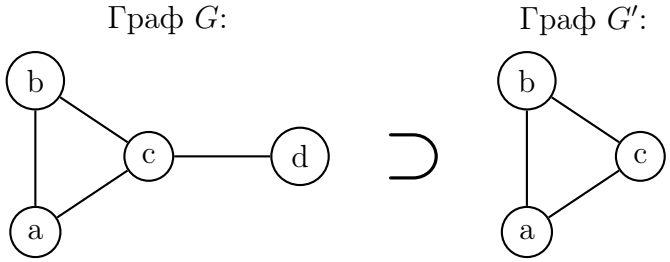
$a \equiv c$
$d \equiv a$
~~$a \equiv f$~~
~~$e \equiv g$~~

Определение. G' — называется *подграфом* G, если из изначального графа выделить некоторые вершины, то есть он является подмножеством изначального множества вершин и ребер. По-научному, подграф G' определяется выражением:

$$G' = (\mathbb{V}\backslash\{v\}, \mathbb{E}\backslash\{(v,u) \mid (v,u) \in \mathbb{E}\})$$

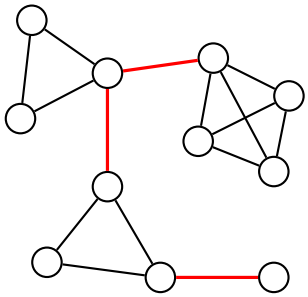
Замечание. G является своим подграфом, а пустой граф $\emptyset$ — подграфом любого графа.

Пример. Граф G' является подграфом графа G:



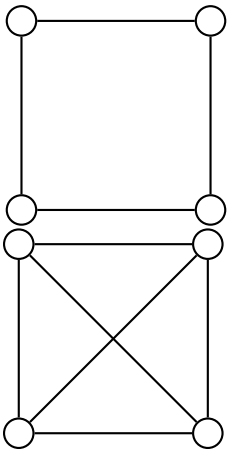
Определение. Пусть существует граф G. Ребро e называется *мостом*, если при его удалении из графа количество компонентов связности $G <$ количество компонентов связности подграфов G.

Пример. В графе G ребра, выделенные красным, являются мостами:



Определение. *Степень связности графа* G — это минимальное количество ребер, которое нужно удалить, чтобы граф G стал несвязным. *Двусвязный граф* — граф степени связности не меньше двух, то есть **граф без мостов**.

Пример. Некоторые примеры графов n-ной степени связности:



— двусвязный граф

— граф 3-ей степени связности

Определение. Вершина $v \in \mathbb{V}$ графа G называется *точкой сочленения*, если при ее удалении из графа (с ребрами) количество компонентов связности $G <$ количество компонентов связности подграфов G. В частном случае, точкой сочленения является вершина моста.

Сосчитаем количество ребер в графе.

Теорема. *В графе $G = (\mathbb{V}, \mathbb{E})$ количество ребер определяется как полусумма всех степеней вершин графа.*

$$|\mathbb{E}| = \frac{1}{2} \sum_{v \in \mathbb{V}} \deg v$$

Доказательство. Мы знаем, что $\deg v$ — количество ребер, выходящих из вершины. Складывая их, мы посчитали все ребра по два раза, так как у ребра ровно две вершины. Следовательно, чтобы получить количество ребер, нужно разделить это число на 2. □

Следствие. *Сумма степеней вершин графа G, а также количество его вершин с нечетной степенью всегда четные.*

Задача. На Землю прилетели 15 трехруких инопланетян. Могут ли они взяться за руки так, чтобы не было ни одной свободной руки?

Решение. Нет, нельзя, так как количество вершин нечетной степени — 15, а должно быть четное количество.

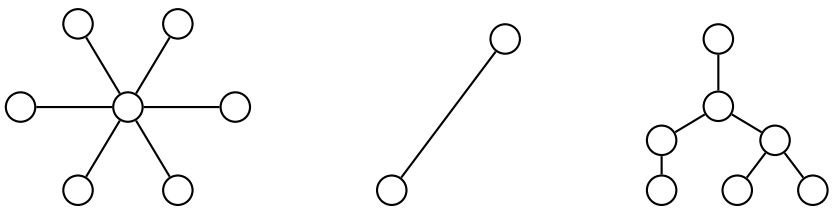
Определение. *Висячая вершина v* — это вершина степени 1 ($\deg v = 1$).

Теорема. *Если в графе G нет висячих вершин, то существует цикл.*

Доказательство. Берем ребро $e = (u_1, u_2)$ из конечного графа G. Мы знаем, что u_2 — не висячая вершина, значит из нее есть какое-то другое ребро $e_1 = (u_2, u_3)$. u_3 — тоже не висячая, и из нее тоже есть новое ребро. И так далее. Тогда по условию, в какой-то момент в графе вершина u_n нового ребра e_n точно совпадет с какой-то старой вершиной u_i, $1 \leq i < n$. Получили цикл $u_i, u_{i+1}, \ldots, u_n$. □

Определение. *Дерево* — связный граф без циклов.

Пример. Примеры деревьев:



Теорема. *В любом дереве хотя бы две висячие вершины.*

Доказательство. Берем любую вершину. Если она не висячая, идем по ребру к следующей. Если снова не висячая, то продолжаем идти из нее. Так как циклы в графе отсутствуют, то в какой-то момент мы достигнем конца графа в висячей вершине. Теперь начнем путь из найденной вершины, и повторим алгоритм. Опять же, в какой-то момент мы упремся в висячую вершину. Итого, получили как минимум две висячие вершины. □

Теорема. *Если $G = (\mathbb{V}, \mathbb{E})$ — дерево, то $|\mathbb{V}| = |\mathbb{E}| + 1$.*

Доказательство по индукции. Пусть меняется количество вершин.

1. База: количество вершин $|\mathbb{V}| = 1$. Следовательно, $|\mathbb{E}| = 0$, то есть $|\mathbb{V}| = |\mathbb{E}| + 1$.

2. Продолжение: рассмотрим произвольное дерево и найдем висячую вершину и удалим ее вместе с единственным ее ребром. При этом граф остался деревом, так как циклы не появились, и он остался связным, а $|\mathbb{E}|$ и $|\mathbb{V}|$ уменьшились на единицу. Продолжая, получим дерево из одной вершины, а по базе теорема сходится.

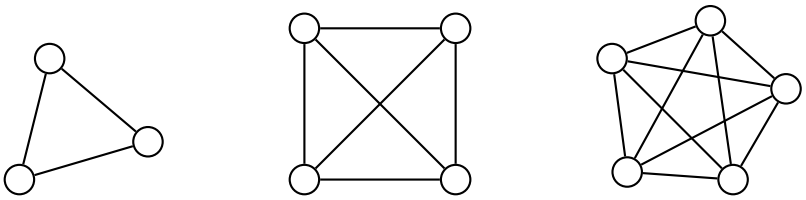
□

5 Полный граф

Теорема. *Если есть n вершин, то:*

1. *$C_n^2 = \frac{n(n-1)}{2}$ ребер.*
2. *Степени всех вершин $n - 1$. Так как $\sum_{v \in \mathbb{V}} \deg v = 2|E|$, то $|E| = \frac{n(n-1)}{2}$*

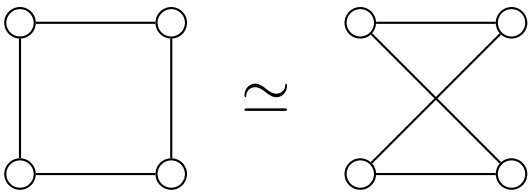
Пример. Следующие графы являются полными:



6 Планарные графы

Определение. *Планарные графы* — это те графы, которые можно нарисовать на плоскости так, чтобы ребра не пересекались.

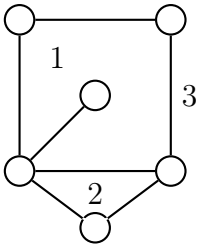
Пример. Пример "правильного" и "неправильного" планарных графов:

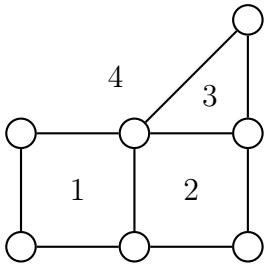
правильный граф  неправильный граф

Теорема (**Формула Эйлера**). *Если связный планарный граф* $G = (\mathbb{V}, \mathbb{E})$ *нарисован на плоскости, то у него можно посчитать грани* f. *Пусть* $|\mathbb{V}| = n, |\mathbb{E}| = m$. *Тогда:*

$$n - m + f = 2$$

Задача. Посчитать грани следующих графов:

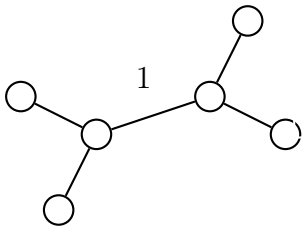




Доказательство. Индукция по количеству ребер.

База: G — дерево. Понятно, что у него одна грань.

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

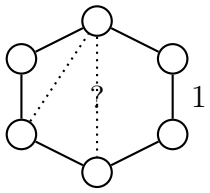


$\Rightarrow\ 6 - 5 + 1 = 2$

Переход: G — неизвестно. Если G' имеет меньше ребер, то верно. Если G — не дерево, значит есть цикл. Возьмем любое ребро цикла — вокруг него точно 2 грани. Теперь удалим это ребро и получим G', G'

тоже связен и планарен. Тогда n' — вершины, m' — ребра, f' — грани графа G'. Количество вершин не изменилось: $n' = n - 1$. Количество ребер и граней стало меньше на 1: $m' = m - 1$, $f' = f - 1$. Получили:

$$n' - m' + f' = 2 \quad \Leftrightarrow \quad n - (m-1) + (f-1) = n - m + f = 2$$



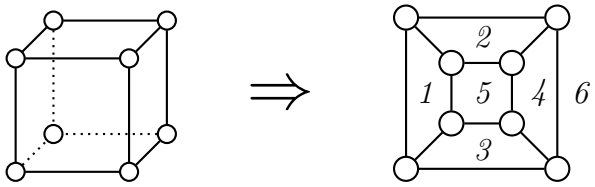
□

Следствие. *Некоторые следствия из формулы Эйлера:*

1. *Не важно, как рисовать планарный граф, количество граней постоянно.*

2. *Теорема про многогранники. Если взять куб, то по формуле Эйлера:* $8 - 12 + 6 = 2$

   Становится понятно если отобразить куб в виде планарного графа:



3. *Если граф G планарен (не обязательно связан), то:*

$$n - m + f = 1 + |\textit{компоненты связности}|$$

4. *У каждой грани как минимум 3 ребра.*

   Доказательство. Посчитаем количество ребер у грани. Можно заметить, что каждое ребро посчитано один или два раза. Тогда:

$$2m \geq \sum_{f \in \mathbb{F}} |\mathbb{E}| \text{ вокруг } f \geq 3f$$

   Следовательно:

$$3f \leq 2m$$

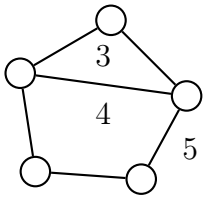
Тогда:

$$n - m + f = 2 \qquad (\times 3)$$

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

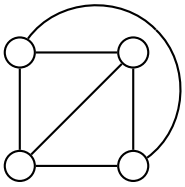
Итого получили, что $m \leq 3n - 6$ в связном планарном графе G.

Физический смысл: ребер не может быть очень много.

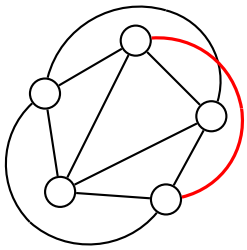


□

5. *Полный граф при $n \geq 5$ не планарен.*



граф планарен

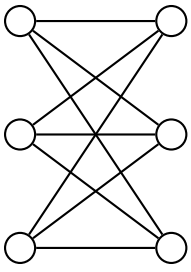


граф не планарен

Доказательство. Если $n = 5$, то $m = \frac{5 \cdot 4}{2} = 10$. Тогда $10 \leq 3 \cdot 5 - 6 = 9$ — неверно, значит граф не планарен. Любой граф с большим количеством не планарен тем более. □

Замечание. Пусть K_5 — полный граф с количеством вершин, равным 5.

Утверждение. *Граф $K_{3,3}$ — тоже не планарен.*



Доказательство. В графе $n = 6, m = 9$: $9 \leq 3 \cdot 6 - 6 = 12$ — сходится.

Рассмотрим количество граней при планарности: $6 - 9 + f = 2$, следовательно $f = 5$ граней. В $K_{3,3}$ все циклы четные (ход лево-право или право-лево). Следовательно у грани должно быть как минимум 4 ребра.

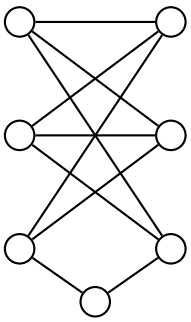
$$4f \leq \sum_{f \in \mathbb{F}} |\mathbb{E}| \text{ вокруг } f = 2m$$

m должно быть хотя бы больше $2f$, но $9 \not\geq 2 \cdot 5$. □

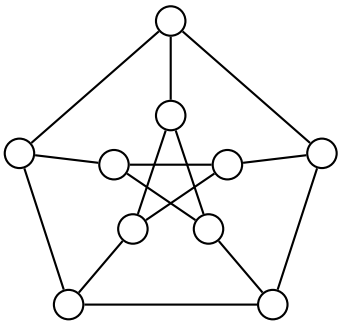
Теорема (Понтрягина-Куратовского)**.** *Граф G планарен только, если он не содержит подграфов G'_i, стягивающихся к K_5 и к $K_{3,3}$.*

Примечание. Стягивающийся к G граф — граф G_1, похожий на граф G при определенных манипуляциях с ребрами.

Пример. Пример стягивающихся графов:



стягивается к $K_{3,3}$



стягивается к K_5

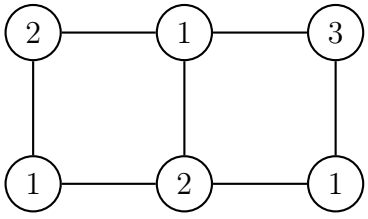
7 Хроматизм

Определение. Пусть $G = (\mathbb{V}, \mathbb{E})$ — граф. *Раскраска графа G* в k цветов это

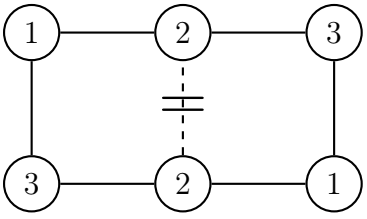
$$\text{функция } c : \mathbb{V} \to \{1 \dots k\}$$

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

Пример. Раскраска графа:



граф раскрашен в 3 цвета



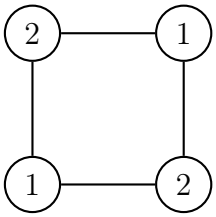
не раскраска графа

Задача. Какие графы можно раскрасить в 1 цвет?
Ответ: графы без ребер.

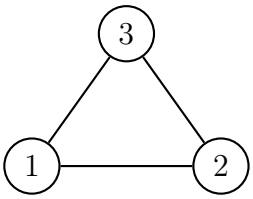
Задача. Какие графы можно раскрасить в 2 цвета?

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

Пример. Двудольные графы:



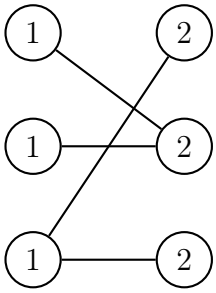
да



нет

Примечание. Граф $K_{3,3}$ — двудолен.

Замечание. Двудольные графы часто рисуют из двух частей (две доли), связанных только в одном направлении (рабочие/задания, студенты/оценки).



Теорема. *Граф G — двудолен только, если всего его циклы имеют четную длину.*

Доказательство четности циклов из двудольности. Если бы цикл был нечетный, то конечные цвета будут повторятся, и их нельзя будет соединить. То есть должно быть одинаковое количество разных цветов. □

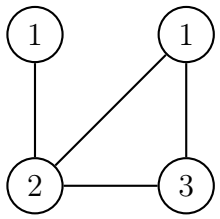
Доказательство двудольности из четности циклов. "Подвесим граф за вершину". Тогда эта вершина связана только с вершинами другого цвета. Следующая вершина опять с другим цветом. И так далее, рассматриваем ребра, которые не идут назад — назначаем цвета по уровням графа в глубину. Почему обратные ребра ничего не портят? Они не соединяют одинаковые цвета, потому что иначе цикл был бы нечетный. □

8 Хроматизм

Определение. Если $G = (\mathbb{V}, \mathbb{E})$ — граф, тогда $\chi(G)$ — *хроматическое число*, то есть минимальное количество цветов, в которые можно раскрасить граф G.

Пример. Примеры расчетов хроматических чисел:

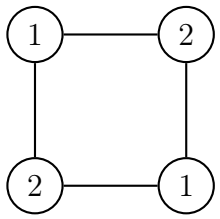
Граф G_1



$$\chi(G_1) = 3$$

можно раскрасить минимум в 3 цвета

Граф G_2



$$\chi(G_2) = 2$$

можно раскрасить минимум в 2 цвета

Следствие. *Хроматическое число полного графа K равно количеству его вершин:*

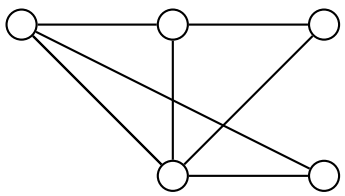
$$\chi(K_{|\mathbb{V}|}) = |\mathbb{V}|$$

Замечание. Если $k \geq \chi(G)$, то граф можно раскрасить в k цветов.

Утверждение 1. *Хроматическое число графа G всегда не больше максимальной степени вершины плюс один.*

$$\chi(G) \leq \max \deg v + 1$$

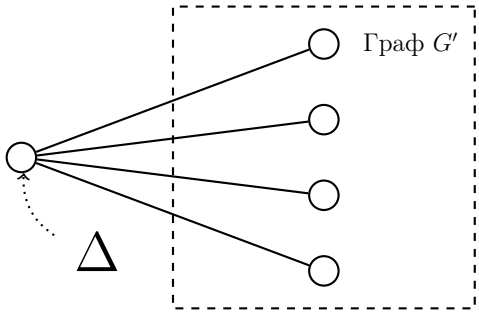
Граф G



$$\chi(G) \leq 4 + 1 = 5$$

$\max \deg v = 4$

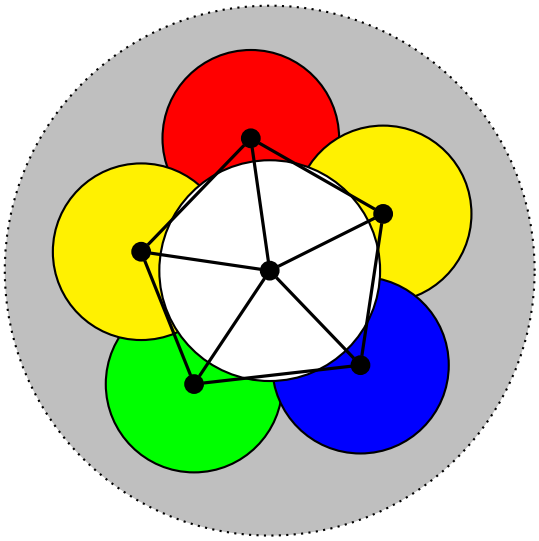
Доказательство. Индукция по количеству вершин.
База. $n = 0, m = 1$ — верно, так как $\max \deg = 0 \Rightarrow \chi(G) \geq 1$.
Переход. Граф G, v — вершина максимальной степени $= \Delta$. Удаляем ее и получаем подграф G'. Максимальная степень подграфа точно не больше степени изначального графа. Попробуем раскрасить в $\Delta + 1$ цвет. Однако цвет запрещен. Значит в Δ цветов раскрасить можно. Продолжая, дойдем то единственной вершины.



□

Утверждение 2. *Если граф G — планарный, то его можно раскрасить не более чем в 5 цветов.*

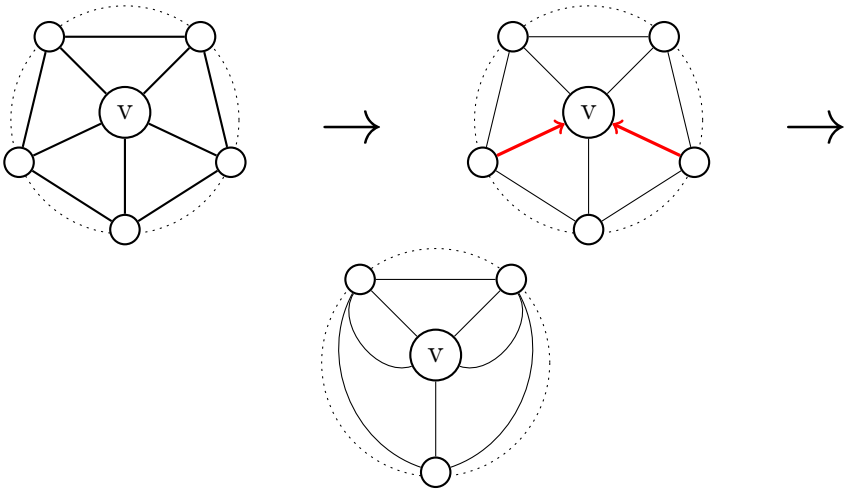
Задача. Во сколько цветов достаточно раскрасить страны на географической карте, чтобы любые две соседние не имели один цвет?



Доказательство. В G есть вершина степени ≤ 5. Если нет, то все степени ≥ 6, значит сумма степеней всех вершин равна хотя бы $6n$ ($n = |\mathbb{V}|$). Мы знаем, что сумма степеней вершин равна удвоенному количеству ребер.

Соответственно ребер хотя бы $3m$, однако так не бывает, ведь количество ребер должно быть меньше или равно $3n - 6$.

Раскрашиваем в пять цветов по индукции.
База. Графы из 5 вершин точно можно раскрасить в пять цветов.
Переход. Пусть есть граф G с $n > 5$ вершин. Предполагаем, что для него есть раскраска. Берем вершину v, у которой степень ≤ 5. Рассмотрим граф без этой вершины и раскрасим его. Если мы для каждой соседней вершины v_i используем не более четырех цветов, то раскрашиваем ее пятым — следовательно, для v есть цвет. Если для соседних вершин используются все пять цветов, то снова попробуем убрать вершину из этого графа — получили одну грань. Возьмем случайные две вершины, не соединенные ребром, и стянем их к v — получим граф $\tilde{G}$. Этот граф также остается планарным, при этом у него будет $n-2$ вершины. Значит, две стянутых вершины можно раскрасить в один цвет, следовательно для v также будет цвет.



Примечание. Между какими-то двумя вершинами точно нет ребра, в другом случае граф будет стягивающимся к K_5, то есть он не был бы планарным.

Задача. Во сколько цветов на самом деле можно раскрасить планарный граф?

Гипотеза (Проблема 4-ех красок)**.** *Любой планарный граф можно раскрасить в 4 цвета.*

□

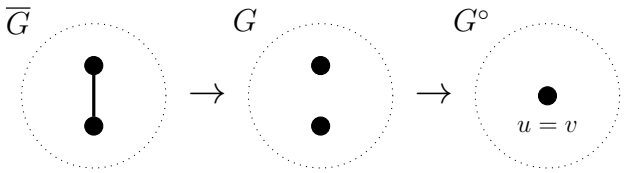
9 Хроматические многочлены

Определение. Пусть $\chi(G, k)$ — это функция, которая возвращает количество способов раскраски графа G по k цветам.

Утверждение. *Рассмотрим некоторые функции:*

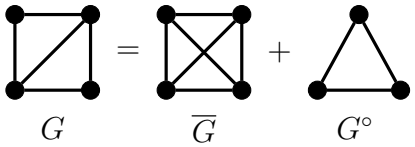
1. *Граф без ребер:* $\chi(\emptyset_n, k) = k^n$
2. *Полный граф:* $\chi(K_n, k) = k(k-1)(k-2)\dots(k-n+1) = \frac{k!}{(k-n)!} = k^{\underline{n}}$
3. *Дерево:* $\chi(T_n, k) = k(k-1)^{n-1}$ *(по методу подвешивания вершины)*
4. *Граф* $\overline{G} = (\mathbb{V}, \mathbb{E})$ *с каким-то ребром* $e = (u, v)$. *Граф* $G = \overline{G} \backslash e$. *Граф* $G = G^\circ$ *— стянутый в одну вершину. Можно заметить, что количество способов раскрасить* G *в* k *цветов. Вершины* u *и* v *имеют либо одинаковый цвет, либо разный цвет. Если имеют разный цвет, то таких способов столько же, сколько раскрасить* $\overline{G}$. *Если цвет одинаковый, то столько же, сколько и* G°. *То есть:*

$$\chi(G, k) = \chi(\overline{G}, k) + \chi(G^\circ, k)$$



Следствие (к пункту 4)**.** *Обратное условие:*

$$\chi(\overline{G}, k) = \chi(G, k) - \chi(G^\circ, k)$$



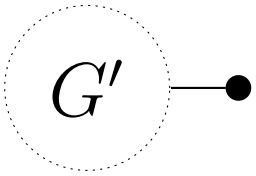
$$\chi(\overline{G}, k) = k(k-1)(k-2)(k-3) + k(k-1)(k-2) = k(k-1)(k-2)^2$$

5. *Цикл:* $\chi(C_n, k) = \chi(T_n, k) - \chi(C_{n-1}, k) = \chi(T_n, k) - \chi(T_{n-1}, k) + \chi(C_{n-2}, k) = \dots$ *— останавливаемся на цикле длиной три, иначе не получим цикл. В финале получим сумму, являющуюся геометрической прогрессией с шагом* $1-k$ *и первым членом* $(-1)^n \cdot k(k-1)$*:*

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

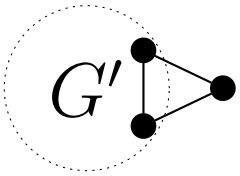
Утверждение. *Пусть есть граф G и он имеет висячую вершину v, и $G' = G \backslash v$. Тогда:*

$$\chi(G, k) = \chi(G', k) \cdot (k - 1)$$



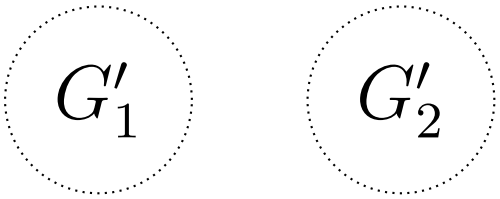
Утверждение. *Если в графе G есть вершина v, образующая треугольник с двумя случайными вершинами, то:*

$$\chi(G, k) = \chi(G', k) \cdot (k - 2)$$

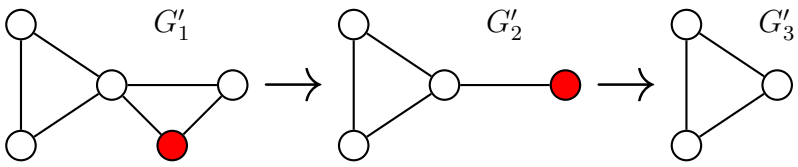


Утверждение. *Пусть $G = G'_1 \cup G'_2$, при этом отсутствуют ребра между G'_1 и G'_2. Тогда:*

$$\chi(G, k) = \chi(G'_1, k) \cdot \chi(G'_2, k)$$



Пример. Рассмотрим преобразование графа G. Красные вершины удаляются:

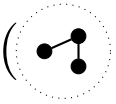
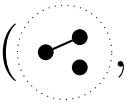
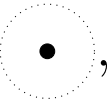


$$\chi(G'_1, k) = \chi(G'_2, k)(k - 2) = \chi(G'_3, k)(k - 1)(k - 2) = k(k - 1)^2(k - 2)^2$$

Утверждение. *$\chi(G, k)$ — хроматический многочлен, обладающий данными свойствами:*

1. *Старший коэффициент всегда равен 1;*
2. *Степень многочлена = n (количество вершин = n);*
3. *Знаки многочлена всегда чередуются;*
4. *Младший коэффициент всегда равен 0;*
5. *Количество ребер графа G равно абсолютному значению коэффициента при k^{n-1}.*

Доказательство. Индукция по количеству вершин.
База. Рассматриваем пустой граф из n вершин: $\chi(\emptyset, k) = k^n$ — это действительно многочлен, удовлетворяющий всем свойствам.
Переход. Пусть есть граф с m ребер. Рассмотрим следующий пример:

$\chi($  $, k) = \chi($  $, k) - \chi($  $, k)$

Ребер стало меньше. Рассмотрим свойства:

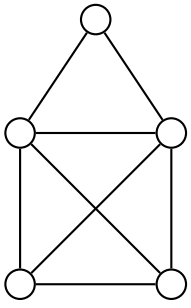
1. Коэффициент остался: $1 \cdot k^n - k^{n-1} + \cdots$;
2. Степень осталась n — следует из предыдущего;
3. Знаки чередуются: $k^n - (k^{n-1} - k^{n-2} - \cdots) + \cdots = k^n - k^{n-1} + k^{n-2} + \cdots$;
4. Младший коэффициент: $0 - 0 = 0$;
5. Количество ребер: $-\text{ребра} \cdot k^{n-1} - k^{n-1} = -k^{n-1} \cdot (\text{ребра} + 1)$.

□

Утверждение. *Хроматическое число многочлена $\chi(G)$ равно первому значению корня многочлена, при котором не получается ноль.*

10 Эйлеровы графы

Задача. Нарисовать данный граф, не проводя по одному ребру дважды:

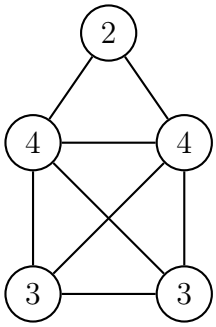


Определение. *Эйлеров путь* — простой путь, содержащий все ребра.

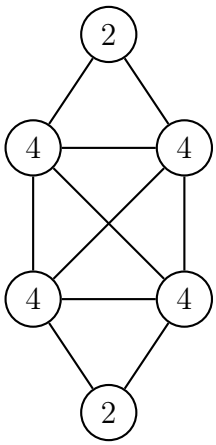
Определение. *Эйлеров цикл* — цикл, содержащий все ребра.

Утверждение. *Пусть G содержит эйлеров цикл, тогда G связен, и* $\deg v$ *— четная* $\forall v \in \mathbb{V}$.

нет циклов

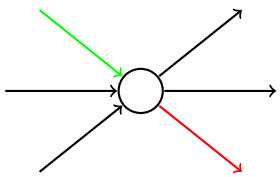


есть цикл

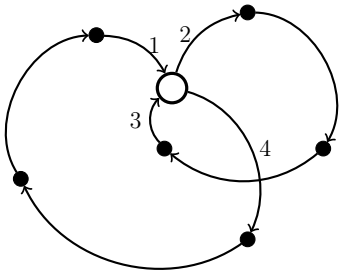


Прямое доказательство. Если граф G связен, то до каждой вершины можно дойти. Мы в нее вошли и вышли одинаковое количество раз, соответственно в любом случае степень такой вершины будет четной. □

Обратное доказательство. Начнем строить цикл. Идем из любой вершины, выбираем ребро, которое еще не использовалось. Вершину мы могли посещать до этого, то есть мы пришли в нее, и вышли одинаковое количество раз. Пусть мы в нее пришли еще раз, получим нечетную степень. Следовательно должно быть еще одно ребро, а если мы застряли, то мы из этой вершины начали (ведь вышли из нее только один раз). Итого, каждая степень четная. □



Замечание. Если есть два цикла в одной вершине, то их можно объединить:

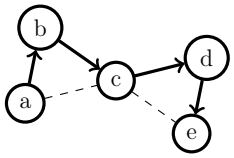


Теорема. *Граф G содержит эйлеров путь, если он связен, а также если степени всех вершин четные, или степени любых двух из них нечетные, а все остальные четные.*

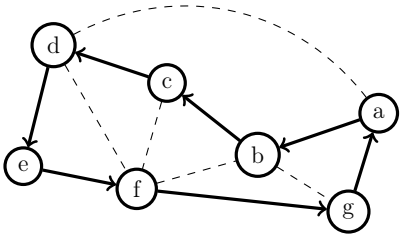
Примечание. Необходимо начать эйлеров путь из одной из нечетных вершины, при этом концом должна быть вторая вершина.

Определение. *Гамильтонов путь/цикл* — простая цепь/цикл по всем вершинам, не повторяясь.

Пример. Примеры гамильтоновых пути и цикла:



abcde — гамильтонов путь

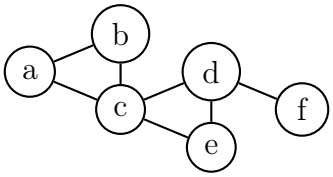


abcdfga — гамильтонов цикл

11 Длина, диаметр, радиус

Определение. *Длина пути* в графе G — это количество ребер в пути.

Пример. Рассмотрим следующий пример:



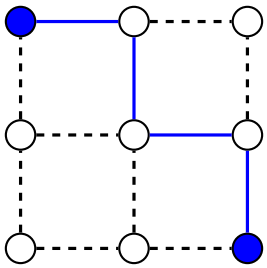
путь от a до f:
abcdf — длина 4
acrdf — длина 4
acdf — длина 3
abcedf — длина 5

Определение. *Расстоянием между вершинами* $d(u, v)$ называется минимальная длина пути между вершинами u и v. Частный случай, длина равна $+\infty$, если пути между этими вершинами не существует.

Пример. Из предыдущего примера: $d(a, f) = 3$.

Определение. *Диаметром графа* называется максимальное расстояние между вершинами графа.

Пример. В примере выше: расстояние $d(a, f)$ является максимальным, следовательно диаметр графа $G = 3$.
Еще один пример:



— диаметр 4

Заметим, что остальные расстояния (**важно — НЕ пути**) ≤ 4.

Определение. Для каждой вершины графа $G = (\mathbb{V}, \mathbb{E})$ можно посчитать максимальное расстояние до других вершин. Обозначим это *радиусом вершины*:

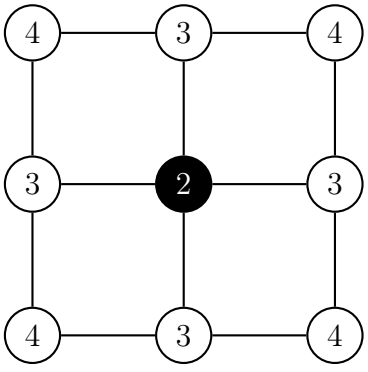
$$r(v) = \max\{d(u, v) \mid u \in \mathbb{V}\}$$

Тогда, *радиусом графа* является минимальное значений $r(v)$:

$$r(G) = \min\{r(v) \mid v \in \mathbb{V}\}$$

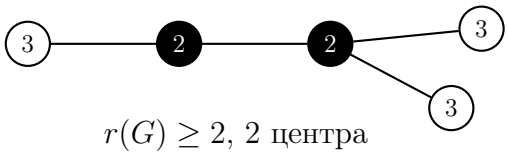
Определение. Вершины с минимальным радиусом называют *центрами графа.*

Пример. Из прошлого примера:



Таким образом, радиус графа $r(G) = \min\{4, 3, 4, 3, 2, 3, 4, 3, 4\} = 2$. Центр помечен черным цветом.

Примечание. Центров графа может быть несколько.



$r(G) \geq 2$, 2 центра

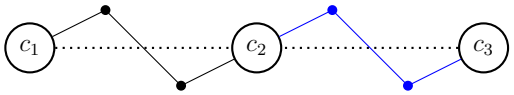
Утверждение. *В любом графе* $G = (\mathbb{V}, \mathbb{E})$ $d(G) \leq 2r(G)$.

Доказательство. Пусть c — центр графа, $c, u, v \in \mathbb{V}$.

Заметим, что $d(c, u) \leq r(G),\ d(c, v) \leq r(G)$, следовательно по транзитивности $d(u, v) = d(c, u) + d(c, v) \leq 2r(G)$. При этом максимум среди двух вершин $d(u, v)$ также $\leq 2r(G)$. □

Утверждение. *В дереве может быть не более двух центров.*

Доказательство от обратного. Пусть центров в дереве 3 : $\{c_1, c_2, c_3\}$. Построим пути между c_1 и c_2, а затем между c_2 и c_3 (вспоминая, что в дереве может быть только один путь).



Если оба полученных пути проходят через одну вершину "развилка"c_0, то ее радиус меньше остальных центров.

$$r(c_0) < r(c_1) = r(c_2) = r(c_3) = r(G)$$

Следовательно, на самом деле, вершина c_0 является центром, а $r(G) = r(c_0)$, что значит, что центров не 3, а уже 1. Пришли к противоречию. □

Условное доказательство. Еще один вариант доказательства: удалим *листья* дерева (висячие вершины) и получим дерево, в котором все оставшиеся расстояния уменьшились на единицу. Соответственно, так как уменьшились все расстояния, то центр (или центры) не изменились. Продолжая удалять снова висячие вершины, мы дойдем до дерева, в котором останется одна или две вершины. Соответственно, эти вершины и будут центрами. □

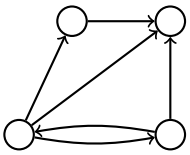
12 Утверждения про ориентированные графы

Замечание. Далее иногда будут использоваться ориентированные графы.

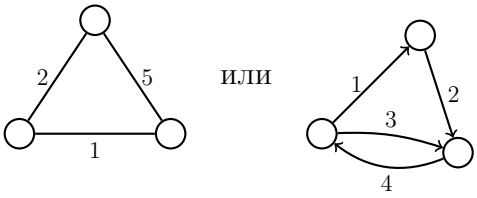
Определение. В ориентированном графе (*орграфе*) $G = (\mathbb{V}, \mathbb{E})$ множество ребер $\mathbb{E}$ является множеством упорядоченных пар.

Примечание. Ребра в ориентированных графах часто называют дугами.

Пример. Ориентированный граф:



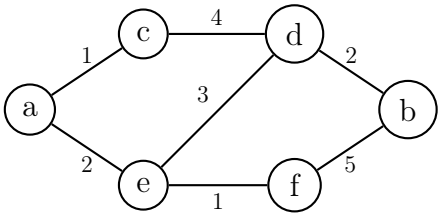
Замечание. Существует *взвешенный граф* — такой граф $G = (\mathbb{V}, \mathbb{E})$, в котором у каждого ребра есть *вес*, то есть функция f, которая сопоставляет ребрам $e \in \mathbb{E}$ вещественное число $a \in \mathbb{R}$.



Определение. Расстояние на графе с весами считается как минимальная сумма весов по всем путям.

$$d = \min \sum_{i}^{|\mathbb{E}|} a_i$$

Пример. Рассмотрим следующий ориентированный граф и найдем расстояние от a до b:



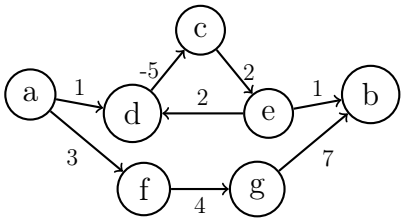
$$d(acdb) = 1 + 4 + 2 = 7$$
$$d(acdefb) = 1 + 4 + 3 + 1 + 5 = 14$$
$$d(aefb) = 2 + 1 + 5 = 8$$
$$d(aedb) = 2 + 3 + 2 = 7$$

$$d(a, b) = 7$$

Ответ: $d(a, b) = 7$.

Замечание. Расстояние во взвешенном графе не всегда удается посчитать.

Пример. Рассмотрим следующий ориентированный граф и найдем расстояние от a до b:



$$d(f, g) = 4$$
$$d(g, f) = +\infty$$
$$d(a, b) = ?$$

$$d(adceb) = 1 - 5 + 2 + 1 = -1$$
$$d(adcedceb) = 1 - 5 + 2 + 1 + $$
$$+(2 - 5 + 2) + 1 = -2$$

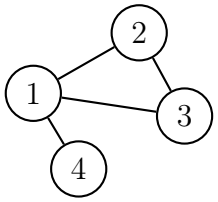
Если продолжить так проходить по циклу бесконечное количество раз, то и расстояние будет стремиться к $-\infty$. То есть расстояние не будет найдено. Условно, $d(a, b)$ можно обозначить $-\infty$.

Утверждение. *В графе есть все расстояния в том случае, если в графе нет циклов отрицательной длины.*

Доказательство. Пусть цикл существует. Значит по нему можно пройтись n раз, а при $n \to \infty$ $d \to -\infty$, откуда следует, что любые две вершины данного цикла не имеют расстояния. Другими словами, если не расстояния, то для двух вершин существуют сколь угодно маленькие пути. □

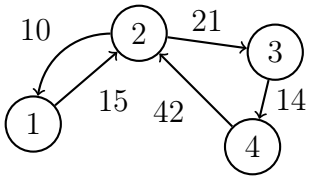
13 Представление графа в компьютере

1. Матрица смежности: таблица $|\mathbb{V}|^2$, где каждая ячейка $a_{i,j} = \begin{cases} 0 \\ 1 \end{cases}$ (содержит 0/1 для описания отсутствия/наличия ребра). Эта матрица всегда симметрична для неориентированного графа.



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

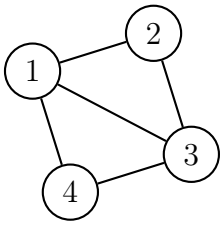
Для графов с весами ячейка матрицы будет содержать вес ребра, или $+\infty$, если ребра нет.



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

Объем памяти: $|\mathbb{V}|^2$.

2. Списки смежности: для каждой вершины задается множество смежных с ней вершин.



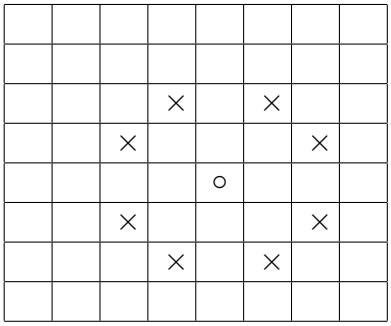
$1: \quad 2, 3, 4$
$2: \quad 1, 3$
$3: \quad 2, 4$
$4: \quad 1, 3$

Объем памяти: $\approx |E|$.

3. Матрицы и списки инцидентности.

4. Неявные способы.

Задача (Обход шахматной доски конем)**.** Пусть есть шахматная доска. Она является графом из 8×8 вершин, а ребра соединяют вершины, в которые может походить конь.



Можно для любой клетки рассчитать, куда можно из нее попасть. Смысл задачи: поиск гамильтоного цикла в графе.

14 Алгоритмы теории графов

Задача. Дано две вершины u и v. Найти расстояние $d(u, v)$ и восстановить путь, на котором достигается это расстояние.

Замечание. Оказывается, что найти путь от u до v — это задача, аналогичная поиску пути от u до всех остальных вершин.

14.1 Алгоритм Форд-Беллмена

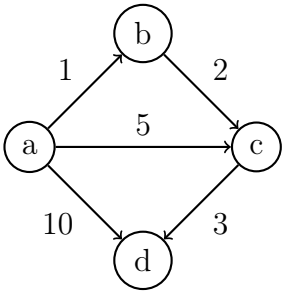
Дан граф $G = (\mathbb{V}, \mathbb{E})$, вершина $u \in \mathbb{V}$ и вес каждого ребра $f_i(e)$. Необходимо найти расстояния $d(u, v)$ для любой вершины $v \in \mathbb{V}$. Будем писать $d(v) = d(u, v)$, так как u — константная вершина, она не меняется. Будем хранить в массиве d текущие найденные расстояния. Начальные условия:

$$d(u) = 0, \qquad d(v) = +\infty, \ (v \neq u)$$

На каждом шаге делается релаксация ребра $e = (v_1, v_2)$. Если $d(v_1) + f(v_1, v_2) < d(v_2)$, тогда меняем $d(v_2)$ на $d(v_1) + f(v_1, v_2)$. Повторяем $n - 1$ раз с перебором всех ребер e, каждое из которых релаксируется.
Время работы алгоритма: $\approx |\mathbb{V}| \cdot |\mathbb{E}| \leq |\mathbb{V}|^3$

Замечание. В неориентированном графе каждое ребро считается и релаксируется как два ребра.

Пример. Возьмем следующий граф:



Составим список смежности:

$$\begin{array}{ll} A: & B(1), C(5), D(10) \\ B: & C(2) \\ C: & D(3) \\ D: & \emptyset \end{array}$$

Инициализируем алгоритм:

шаг	A	B	C	D
0:	0	$+\infty$	$+\infty$	$+\infty$
1_{AB}:	0	1	$+\infty$	$+\infty$
1_{AC}:	0	1	5	$+\infty$
1_{AD}:	0	1	5	10
1_{BC}:	0	1	3	10
1_{CD}:	0	1	3	6
2:	*	*	*	*
3:	*	*	*	*

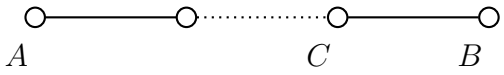
Ответ: $d(v) = 6$.

Примечание. Если после шага в следующем шаге данные не изменились, то работу алгоритма можно прервать.

14.1.1 Корректность алгоритма Форда-Беллмана

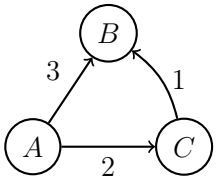
Теорема. *В конце выполнения алгоритма массив d будет содержать расстояния от вершины.*

Доказательство. Оказывается, после каждой итерации цикла релаксации всех ребер, массив d хранит числа $d_i(v) \leq \min$ длин путей, в которых $\leq i$ ребер. Действительно:
База. $i = 0$, min(путь из 0 ребер) — такой путь только из вершины, саму в себя. $d(A) = 0,\ d(u) = +\infty$.
Переход. Пусть есть оптимальный путь из $i + 1$ ребра.

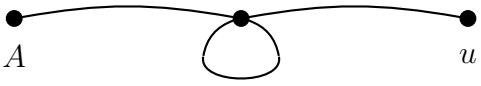


$A \to C = i$ ребер $\qquad$ $A \to B = i + 1$ ребер

По предположению, $d(c) = dist(A, C)$. Длина пути $A \to C \to B$ равна $dist(A, C)$ + вес(CB). Проверка: $d(C)$ + вес$(CB) \leq d(k)$. Оптимальный путь окажется меньше или равным взятого пути, что значит, что значение массива d гарантировано обновится до оптимального.



Примечание. Почему проходится $n - 1$ этапов? Оптимальный путь не содержит циклов.



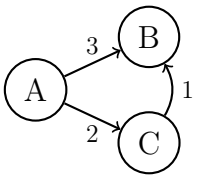
□

Примечание. Достоинство алгоритма Форда-Беллмана заключается в том, что он работает с любыми весами.

Замечание. Мы вычисляем только расстояния, но сам путь при этом неизвестен. Как восстановить путь?

14.1.2 Восстановление пути

Для того, чтобы по алгоритму Форда-Беллмана можно было узнать путь, будем хранить информацию об успешных релаксациях. Пусть есть изначально пустой массив вершин $prev$. Если релаксация вершин $u \to v$ успешна, то $prev(v) = u$ — оптимальный путь в v лежит через u.



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

Восстановить путь в B:

$$prev(prev(B)) \to prev(B) \to B = A \to C \to B$$

В общем случае путь $A \to v$:

$$A \to prev(\dots) \to \dots \to prev(prev(v)) \to prev(v) \to v$$

14.2 Алгоритм Дейкстры

Замечание. В отличие от Алгоритма Форда-Беллмана этот алгоритм требует, чтобы вес каждого ребра был положительным.

Дан граф $G = (\mathbb{V}, \mathbb{E})$, $A \in \mathbb{V}$. Найти расстояния до всех вершин $d(u) = dist(A, u)$. В начале мы точно знаем, что $d(A) = 0$, $d(u \neq A) = +\infty$. Будем обходить все вершины. Зададим множество обработанных вершин p, которое изначально будет пустым.

Повторяем $n = |\mathbb{V}|$ раз: выбрать из всех вершин, кроме обработанных, где $d(u)$ — минимально.После этого необходим цикл: для каждого $e = (u, v) \in \mathbb{E}$ релаксируем ребро e. После выполнения цикла $p = p_u\{u\}$ — то есть вершина тоже стала обработанной. Идем к следующей итерации.

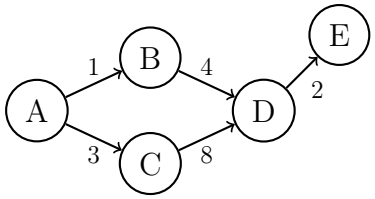
14.2.1 Эффективность алгоритма

В алгоритме производится обход каждой вершины. При этом каждое ребро мы проходим только один раз. Минимальные элемент можно найти по алгоритму с логарифмической сложностью. Итого:

$$|\mathbb{V}| \cdot \log |\mathbb{V}| \quad \Leftrightarrow \quad O(n) = n \log n$$

14.2.2 Пример вычислений

Пример. Возьмем следующий граф:



В графе есть пять вершин, значит по алгоритму нужно пройтись по циклу пять раз.

			A	B	C	D	E	
		d	$\boxed{0}$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$p = \emptyset$
Ит. 1:	$\min = A$			$\boxed{1}$	3	$+\infty$	$+\infty$	$p = \{A\}$
	$A_0 \to_1 B_{+\infty}$				$\boxed{3}$	9	$+\infty$	$p = \{A, B\}$
	$A_0 \to_3 C_{+\infty}$					$\boxed{7}$	$+\infty$	$p = \{A, B, C\}$
Ит. 2:	$\min = B$						$\boxed{9}$	$p = \{A, B, C, D\}$
	$B_1 \to_8 D_{+\infty}$							$p = \{A, B, C, D, E\}$
Ит. 3:	$\min = C$							
	$C_3 \to_4 D_9$							
Ит. 4:	$\min = D$							
	$D_7 \to_2 E_{+\infty}$							

14.2.3 Корректность алгоритма

Идея алгоритма. На каждом шаге $d(u)$ равно минимуму среди путей из A в u, в которых используются только обработанные вершины. Докажем по индукции.

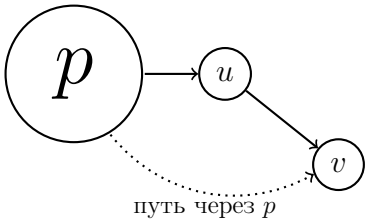
База. Шаг нулевой: $d(A) = 0$, $d(u) = +\infty$ — инициализация соответствует алгоритму.

Переход. Пусть есть какое-то множество обработанных вершин p. В нем точно будет A. Также есть оставшиеся необработанные вершины за пределами множества p. Выбрали минимум $u = \min v \in \mathbb{V}\backslash\{p\}$. Пусть есть следущий оптимальный путь в u:

$$A \to \cdots \to \overline{u} \to \cdots \to u$$

При этом расстояние до $\overline{u}$ равно d из обработанных вершин, а $dist(\overline{u}) = dist(u) - x$. По индукционному предположению, $dist(\overline{u}) = d(\overline{u})$. Тогда, получили, что $d(u) > d(\overline{u})$, то есть можно было бы выбрать $d(\overline{u})$ — противоречие с тем, что $d(u) = \min$.

Пути через p уже были рассмотрены на предыдущих шагах, соответственно рассматриваются пути по необработанным вершинам.



Если необработанная вершина — искомая (v), то мы уже нашли нужное расстояние. Осталось рассмотреть другую вершину (u):

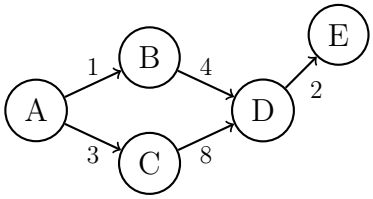
$$dist(A, u) + \omega(u, v) = dist(A, v)$$

Релаксация $u \to v$ успешна, и $d(v)$ получит оптимальное расстояние. $\square$

14.2.4 Восстановление пути

По аналогии с алгоритмом Форда-Беллмана, нужен массив вершин $prev$. То есть при успешной релаксации $u \to v$ запоминаем $prev(v) = u$.

Пример. Обращаясь к примеру выше:



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]

Получившийся путь $A \to E$:

$$prev(C) \to prev(D) \to prev(E) \to E = A \to C \to D \to E$$

14.3 Алгоритм Флойда

Дан граф $G = (\mathbb{V}, \mathbb{E})$. В алгоритме реализуется таблица размером $|\mathbb{V}| \times |\mathbb{V}|$, состоящая из расстояний $d(u, v)$. Таким образом находятся <u>все</u> расстояния от любой вершины u к любой другой вершине v.

Инициализация алгоритма. Для инициализации берется начальная таблица d_0, удовлетворяющая следующим условиям:

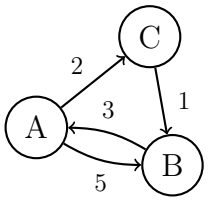
$$\begin{cases} d(u,u) = 0 \\ d(u,v) = \begin{bmatrix} \infty, & u \not\to v \\ \text{вес}(u,v), & u \to v \end{bmatrix} \end{cases}$$

После инициализации начинается перебор вершин. Для каждой вершины $k \in \mathbb{V}$ перебираем вершины $u \in \mathbb{V}$, а для каждой вершины v, соответственно, еще вершину v. И для такой вершины начинаем "релаксировать"ребра, то есть если $d(u,v) > d(u,k) + d(k,v)$, то $d(u,v)$ меняем на $d(u,k) + d(k,v)$.

По аналогии с программированием получается тройной цикл:

$$\begin{array}{l} \text{foreach } k \text{ in } \mathbb{V}: \\ \quad \text{foreach } u \text{ in } \mathbb{V}: \\ \quad\quad \text{foreach } v \text{ in } \mathbb{V}: \\ \text{if } (d(u,v) > d(u,k) + d(k,v)) \quad \Rightarrow \quad d(u,v) = d(u,k) + d(k,v) \end{array}$$

Пример. Разберем алгоритм на примере:



d_0	A	B	C
A	0	5	2
B	3	0	∞
C	∞	1	0

Начинаем обход:

1. $k = $ A

	A	B	C
A	0	5	2
B	3	0	$\boxed{5}$
C	∞	1	0

2. $k = $ B

	A	B	C
A	0	5	2
B	3	0	5
C	$\boxed{4}$	1	0

3. $k = $ C

	A	B	C
A	0	$\boxed{3}$	2
B	3	0	5
C	4	1	0

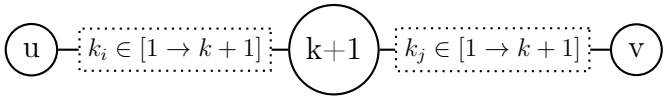
14.4 Корректность алгоритма Флойда

Утверждение. *После шага k в $d(u,v)$ содержится минимальная длина пути $u \to v$, в котором содержатся только вершины от 1 до k.*

Доказательство по индукции. Индукция по вершинам k.
База. Пусть $k = 0$. Если путь $u \to v$ существует, и в нем используются вершина до k, при этом $k = 0$ означает, что вершин таких нет, то любое $d(u,v)$ уже является минимальным.
Переход. Необходимо рассмотреть все пути $u \to v$, которые содержат вершины от 1 до $k+1$. Допустим, что есть оптимальный путь $u \to v$ через $k+1$ вершин. Есть два случая:

1. В таком пути нет вершины $k+1$, значит этот путь имеет длину $d(u,v)$ (по индукционному предположению).

2. Этот путь содержит вершину $k+1$. Тогда путь имеет длину $d(u, k+1) + d(k+1, v)$.



Получившийся путь и является условием, проверяющимся в алгоритме. Меньший вариант записывается, то есть в этом случае точно будет расстояние $d(u, v)$.

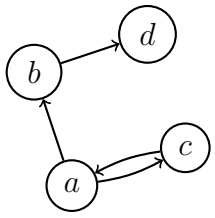
Получили, что в конце алгоритма для каждого случая в таблице будет точно определено расстояние $u \to v$. □

Замечание. Путь можно восстановить с помощью определения двумерного массива $through$: при выполнении условия цикла в алгоритме массива заполняется данной обрабатываемой вершиной k. Если в массиве не указан путь для двух вершин, значит оптимальным путем является само ребро между ними.

Замечание. Алгоритм Флойда также работает для поиска транзитивного замыкания бинарных отношений.

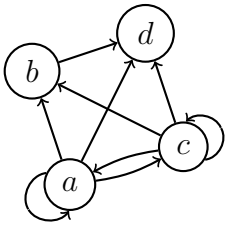
Утверждение. *Пусть R — бинарное отношение на множестве $\mathbb{M}$. $\overline{R}$ является транзитивным замыканием R, если оно расширяет отношение R, является транзитивным и минимальным.*

Пример. Рассмотрим не транзитивное отношение R на следующем примере:



aRb, bRd, но ~~aRd~~
aRc, cRa, но ~~aRa~~

Чтобы сделать отношение транзитивным, нужно добавить все подобные ребра, удовлетворяющие такому условию:



— транзитивное замыкание $R = \overline{R}$

Теорема. *Пусть есть бинарное отношение R на множестве $\mathbb{M}$, а $G = (\mathbb{M}, \mathbb{R})$ — граф этого отношения. Тогда транзитивное замыкание $R = \overline{R}$ — это такое отношение $x\overline{R}y$, которое показывает, что существует путь из x в y.*

Доказательство. Докажем свойства транзитивного замыкания:

1. $\overline{R} \supset R$, так как если xRy, то есть путь из одного ребра, тогда $x\overline{R}y$.
2. $\overline{R}$ — транзитивно, так как если $x\overline{R}y$ и $y\overline{R}z$, то $x\overline{R}z$: очевидно, что если существует путь $x \to y$ и $y \to z$, то из x можно попасть в z.
3. Пусть $\overline{\overline{R}}$ — транзитивно, и существует путь из x в y:

   $$x \to x_1 \to x_2 \to x_3 \to \cdots \to y$$

   Тогда, если xRx_1, то $x\overline{\overline{R}}x_1$. При этом если x_1Rx_2, то $x_1\overline{\overline{R}}x_2$, а значит $x\overline{\overline{R}}x_2$. И так далее, по транзитивности; в итоге получится, что $x\overline{\overline{R}}y$. Таким образом, $\overline{\overline{R}} \supset \overline{R} \supset R$.

□

Применим алгоритм Флойда к графу $G = (\mathbb{M}, \mathbb{R})$. Инициализация алгоритма имеет вид

$$\begin{cases} d_0(x, y) = 1, \text{ если } xRy \\ d_0(x, y) = \infty, \text{ если } \cancel{xRy} \end{cases}$$

В результате алгоритма будет построена матрица отношений между вершинами, выраженными в количестве переходов. Если значение на пересечении первой и второй вершин (столбца и строки матрицы) будет конечным числом, то путь между данными вершинами существует. Если оно равно ∞ — то не существует.

Замечание. На практике не обязательно считать количество переходов, если нас интересует только информация о наличии или отсутствии пути между двумя вершинами. Достаточно один раз зафиксировать наличие одного пути.

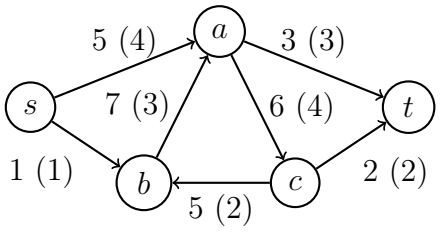
15 Потоки в сетях

Определение. *Сеть* — это ориентированный граф $G = (\mathbb{V}, \mathbb{E})$, в котором $s, t \in \mathbb{V}$, причем не существует ребер вида $e_s = (u_i, s)$ и $e_t = (t, v_i)$.

Другими словами можно сказать, что в вершину s не входят никакие ребра, а из вершины t — не выходят. Также существует функция c, сопоставляющая каждому ребру e натуральные числа $n \in \mathbb{N}$, называющаяся *пропускной способностью ребра.*

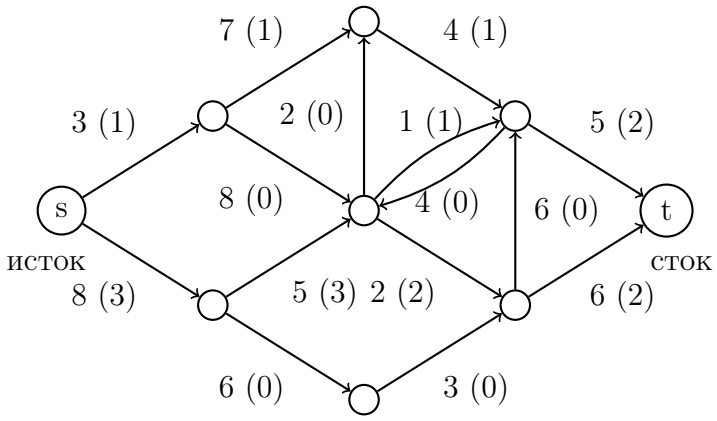
Определение. *Поток* f в сети G это функция, сопоставляющая ребрам e действительные числа $r \in \mathbb{R}$, причем значение функции не превосходит пропускную способность ребра, а алгебраическая сумма потоков в вершинах v, таких что $v \neq s$ и $v \neq t$, должна быть нулем.

Пример. Рассмотрим следующий пример. Вес ребра является пропускной способностью, а в скобках указан поток:



16 Потоки в сетях

Пример. Рассмотрим следующий пример. Текущий поток будем обозначать в скобках:



Поток $f = 4$ (из истока вытекает в сумме 4, в сток — втекает 4).

Теорема. *Дана сеть* (G, c)*, поток* $f \to G$*. Тогда*

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

Доказательство. Рассмотрим сумму всех ребер $\sum_{e \in \mathbb{E}}$. Мы знаем, что алгебраическая сумма потоков в $\forall u \backslash s, t$ равна нулю. Так как мы рассматриваем все вершины, кроме s и t, а s — единственная вершина только с исходящими ребрами, и t — единственная вершина только с входящими ребрами, то сумма всех таких сумм ребер, соединяющих вершины $\forall u \backslash s, t$, равна 0. Соответственно, вытекающий поток будет равен втекающему. □

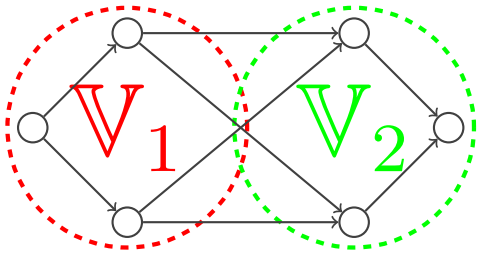
Замечание. Величина, равная вытекающему из s и втекающему в t, называется *величиной потока*.

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

Определение. *Разрез* $\mathbb{C}$ в сети (G, c), где G — граф $(\mathbb{V}, \mathbb{E})$, — это деление общей сети на два непересекающихся множества вершин, при этом исток попадает в первое множество, а сток — во второе.

$$\mathbb{C} = (\mathbb{V}_1, \mathbb{V}_2) \ : \begin{cases} s \in \mathbb{V}_1 \\ t \in \mathbb{V}_2 \\ \mathbb{V}_1 \cup \mathbb{V}_2 = \mathbb{V} \\ \mathbb{V}_1 \cap \mathbb{V}_2 = \emptyset \end{cases}$$

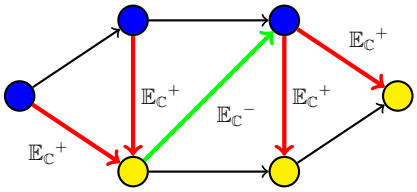
Пример. Разрез:



Определение. *Ребрами разреза* $\mathbb{E}_\mathbb{C}$ называется множество всех ребер, которые соединяют ребра из $\mathbb{V}_1$ и $\mathbb{V}_2$.

$\mathbb{E}_\mathbb{C}{}^+$ — прямые ребра разреза $(\mathbb{V}_1 \to \mathbb{V}_2)$.
$\mathbb{E}_\mathbb{C}{}^-$ — обратные ребра разреза $(\mathbb{V}_1 \leftarrow \mathbb{V}_2)$.

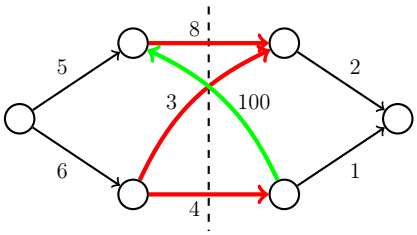
Пример. Для удобства множества $\mathbb{V}_1$ и $\mathbb{V}_2$ обозначены синим и желтым цветом, соответственно, прямые ребра — красным, обратные — зеленым.



Определение. *Величина разреза* равна сумме всех пропускных способностей прямых ребер.

$$c(\mathbb{C}) = \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{+}} c(e)$$

Пример. Дана сеть (G, c) и разрез $\mathbb{C}$ — условная пунктирная линия, отделяющая два множества вершин $\mathbb{V}_1$ (левая половина) и $\mathbb{V}_2$ (правая половина).

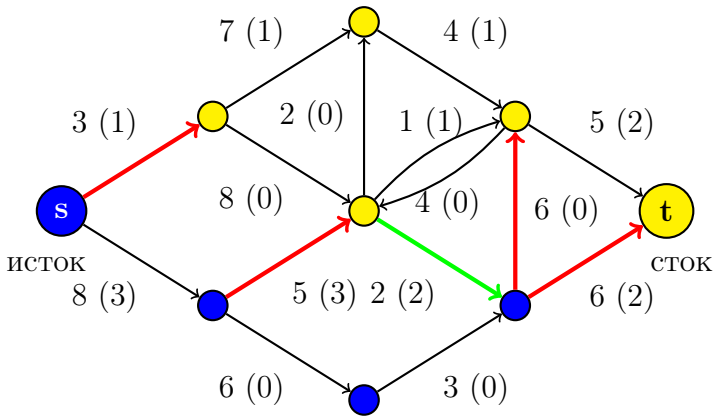


$$c(\mathbb{C}) = 8 + 3 + 4 = 15$$

Утверждение. *Пусть есть сеть* (G, c), *где* G — *граф* $(\mathbb{V}m\mathbb{E})$, *поток* f *в сети, разрез* $\mathbb{C} = (\mathbb{V}_1, \mathbb{V}_2)$. *Тогда*

$$w(\mathbb{C}, f) = \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{+}} f(e) - \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{-}} f(e)$$

Пример. Возьмем пример из начала, в котором множество вершин $\mathbb{V}_1$ обозначено синим цветом, а множество вершин $\mathbb{V}_2$ — желтым. Прямые ребра обозначены красным цветом, обратные — зеленым.



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

$$\sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{+}} f(e) = 1 + 3 + 0 + 2 = 6, \qquad \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{-}} f(e) = 2$$

Действительно, $w(\mathbb{C}, f) = \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{+}} f(e) - \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{-}} f(e) = 6 - 2 = 4$

Доказательство. Посчитаем сумму

$$\sum_{v \in \mathbb{V}_1} \Big(\sum_{e:\ e=(u,v)} f(e) - \sum_{e:\ e=(v,u)} f(e) \Big)$$

Рассмотрим два способа:

1. Для $\forall v \in \mathbb{V}_1 \backslash \{s\}$ внутренняя сумма равна нулю. Для $v = s$ получается
$$w(f) = \sum_{e:\ e=(s,u)} f(e)$$
2. Рассмотрим сумму по дугам ($u \in \mathbb{V}_1$, $v \in \mathbb{V}_2$):
$$\sum_{e=(u,v)} (f(e) - f(e)) + \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{+}} f(e) - \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{-}} f(e) = 0 + w(\mathbb{C}, f) = w(\mathbb{C}, f)$$

□

Замечание. В любом разрезе $w(f) = w(\mathbb{C}, f)$.

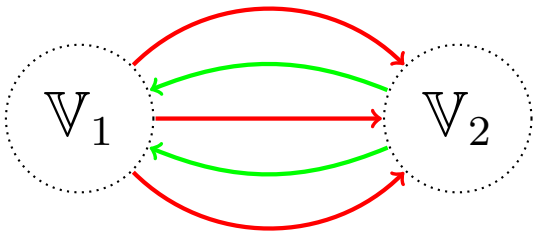
Задача. Решается задача о максимальном потоке в сети, то есть такой поток f, что $w(f) \to \max$.

Утверждение. *Дана сеть (G, c), разрез $\mathbb{C}$. Тогда величина потока не превосходит величину разреза.*

$$w(f) \le c(\mathbb{C})$$

Доказательство. По утверждению о величине потока:

$$w(f) = w(\mathbb{C}, f) = \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{+}} f(e) - \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{-}} f(e) \le \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{+}} f(e) \le \sum_{e \in \mathbb{E}_{\mathbb{C}}{}^{+}} c(e)$$



Зная, что $\mathbb{E}_{\mathbb{C}}{}^{+}c(e) = c(\mathbb{C})$, получили, что

$$w(f) \leq c(\mathbb{C})$$

□

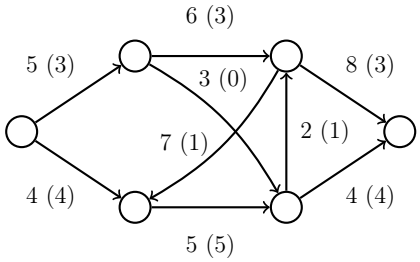
Следствие. *В сети* (G, c) $w(f_{\max}) \leq c(\mathbb{C}_{\min}) \Leftrightarrow \max w(f) \leq \min c(\mathbb{C})$.

Теорема (Форда-Фалкерсона)**.** *На самом деле, в сети* (G, c)*, где* G *— граф* $(\mathbb{V}, \mathbb{E})$*,* $c(e) \in \mathbb{N}$ *(для простоты считаем, что пропускные способности положительные целые):*

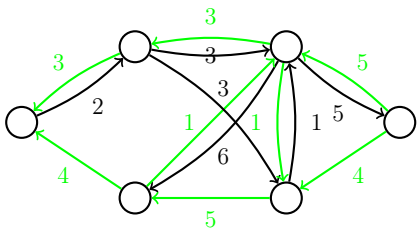
$$w(f_{\max}) = c(\mathbb{C}_{\min})$$

Определение. *Дополнительный граф* $\overline{G}$ для потока — это взвешенный граф, который имеет вершины $\overline{\mathbb{V}} = \mathbb{V}$, а ребра $\overline{\mathbb{E}}$, такие что если $f(e) < c(e)$, где $e = (u, v) \in \mathbb{E}$, то существует ребро $e' = (u', v')$, для которого вес $g(e') = c(e) - f(e)$. Если $f(e) > 0$ для $e = (u, v)$, то нужно добавить обратное ребро $e'' = (v', u')$, а его вес $g(e'') = f(e)$.

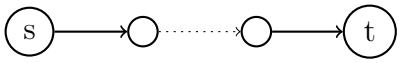
Пример. Изначальный граф:



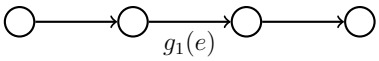
Дополнительный граф:



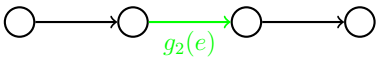
Доказательство. Начнем с нулевого поток и будем его постепенно увеличивать. Построим дополнительный граф $\overline{G}$, найдем в нем путь из s в t и минимальный вес $g(e) = x$ в нем.



Вычтем в дополнительном графе x на каждом ребре.



или



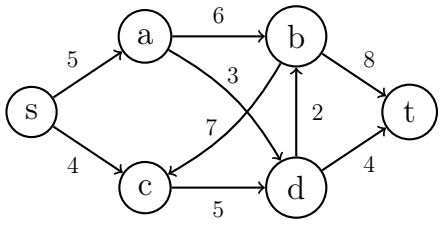
$$g_1(e) = c(e) - f(e) - x, \quad g_2(e) = f(e) - x$$

Поймем, что новый поток f' остался потоком и величина нового потока увеличилась на x. Проверяем, что это поток $0 \leq f'(e) \leq c(e)$. Если уменьшаем по обратному, то он остался положительным, если увеличиваем по прямому, то он не может превысить $c(e)$.

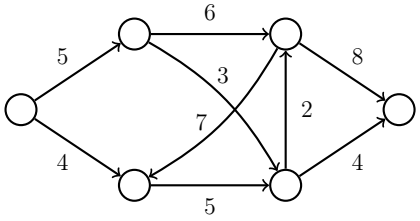
Сумма входящих потоков в вершину должна быть равна сумме выходящих, так как изменение затрагивает весь путь через такую вершину, следовательно величина потока какого-то входящего ребра увеличилось на x, ровно как величина какого-то выходящего.

Значит f' — поток. □

Пример. Рассмотрим граф из предыдущей лекции:

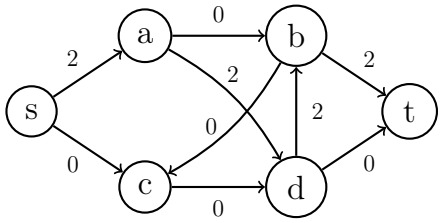


Найдем для него максимальный поток. Будем строить дополнительные графы:

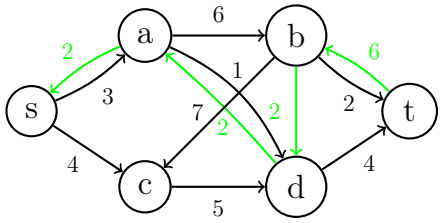


Ищем путь из s в t:
1) $s \to a \to d \to b \to t$
$f_1 = 2(\min = d \to b)$
$\forall e \in 1)\, f(e) + 2$

Таким образом общий поток стал выглядеть как

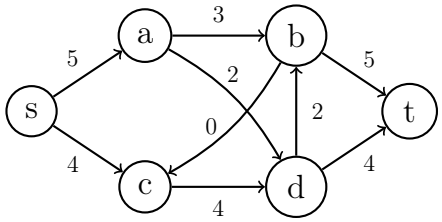


а дополнительный граф — это



Теперь продолжаем искать пути в дополнительном графе.

Финальный поток будет иметь вид:



$$f = 9$$

Если нарисовать дополнительный граф, то можно будет понять, что мы не можем увеличить поток, так как путей из s не будет. Это следует из того, что дуги $s \to a$ и $s \to c$ имеют поток, равный их пропускной способности, следовательно пропустить больший поток не получится.

Продолжение доказательства теоремы Форда-Фалкерсона. Почему, если нет путей, то поток является максимальным?

Пусть множество $\mathbb{V}_1$ — вершины, достижимые из s по ребрам дополнительного графа $\mathbb{V}_2 = \mathbb{V} \backslash \mathbb{V}_2$. Если путей в t из s нет, то $t \in \mathbb{V}_2$, а $\mathbb{V}_1 \cap \mathbb{V}_2 = \emptyset$, что означает, что получен разрез $\mathbb{C}$. Прямые ребра ${\mathbb{E}_{\mathbb{C}}}^+$ в конечном дополнительном графе отсутствуют, так как путей из $\mathbb{V}_1$ в $\mathbb{V}_2$ не существует. Определим размер разреза:

$$c(\mathbb{C}) = \sum_{e \in {\mathbb{E}_{\mathbb{C}}}^+} c(e) = \sum_{e \in {\mathbb{E}_{\mathbb{C}}}^+} f(e) - \sum_{e \in {\mathbb{E}_{\mathbb{C}}}^-} f(e) = c(f)$$

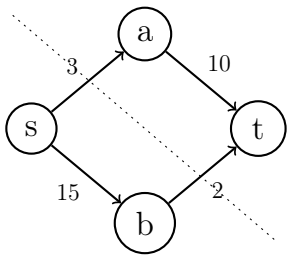
Для любого разреза $\mathbb{C}$ его величина всегда больше или равна величине любого потока f.

$$c(\mathbb{C}) \geq c(f)$$

Значит поток можно только уменьшить, так как $c(\mathbb{C})$ — минимальный разрез, а $c(f)$ — это максимальный поток, следовательно f является максимальным потоком. □

Следствие. *Метод Флойда-Фалкерсона строит минимальный разрез и максимальный поток.*

Пример. Минимальный разрез и максимальный поток:



$$c(\mathbb{C}) = c(f) = 5$$

Утверждение 3. *Если каждый раз искать путь с минимальным количеством ребер, то время поиска максимального потока пропорционально* $\mathbb{V}^2\mathbb{E}$.

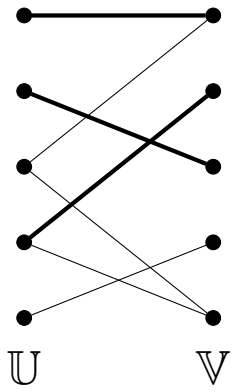
Утверждение 4. *Эффективный алгоритм для плоской (планарной) сети, то есть без пересечения ребер: поиск самого верхнего пути.*

16.1 Паросочетания

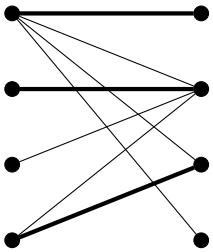
Определение. *Паросочетанием* называется подмножество ребер $\mathbb{P} \subset \mathbb{E}$, такое что ребра из $\mathbb{P}$ не имеют общих вершин.

Определение. *Максимальное паросочетание* — это подмножество ребер $\mathbb{P} \in \mathbb{E}$, такое что количество паросочетаний $|\mathbb{P}| \to \max$ из возможных.

Задача (о паросочетаниях)**.** Дан двудольный граф $G = (\mathbb{U}, \mathbb{V}, \mathbb{E})$.

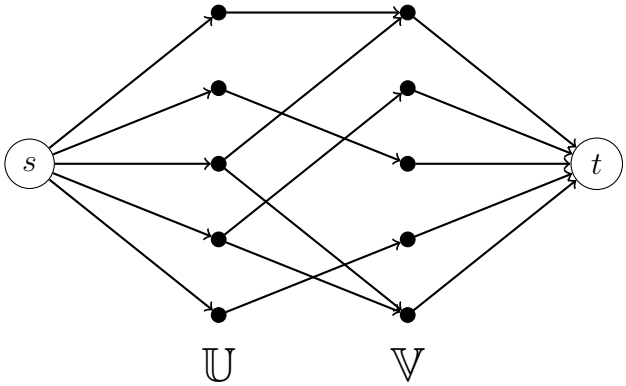


Пример. Разберем пример:



В этом примере нельзя получить 4 паросочетания, так как две вершины второго множества — висячие — соединены с одной вершиной первого множества. Следовательно можно получить максимум 3 (помечены толстыми линиями).

Задача (о паросочетаниях, через потоки)**.** Оказывается, что задачу о нахождении максимального паросочетания можно привести к виду задачи о нахождении потока. Для этого задаем вершину s, дуги **из которой** соединяют все вершины $u \in \mathbb{U}$, и вершину t дуги **в которую** соединяют все вершины $v \in \mathbb{V}$. Ребра данного двудольного графа приводим к виду ориентированных в направлении из множества $\mathbb{U}$ в множество $\mathbb{V}$.

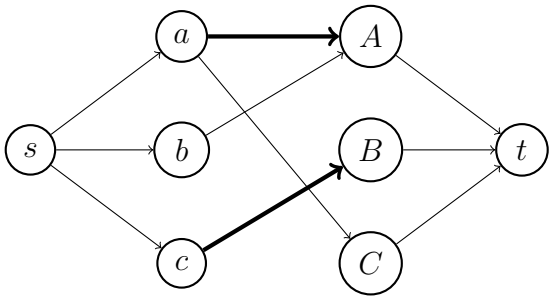


Утверждение. *Каждому потоку из* $\{0,1\}$ *соответствует паросочетание.*

Доказательство. Ребра с $f(e) = 1$ это ребра паросочетания, так как входящий поток равен выходящему, а выходить в таком случае может максимум $f(e) = 1$ для каждой вершины. Значит, вершины пересекаться не будут, что и требуется от задачи. □

Следствие. *Максимальный поток в такой задаче определяет размер максимального паросочетания.*

Пример. Рассмотрим пример. Строим паросочетание по методу Флойда-Фалкерсона:



1. Сначала берем путь $s \to a \to A \to t$, тогда находим одно паросочетение (a, A). После этого дуги этого пути разворачиваем.

2. Пути $s \to b \to A \to t$ и $s \to a \to C \to t$ невозможны, так как теперь прийти в t не получится: содержащиеся дуги развернуты — $s \leftarrow a$ и $A \leftarrow t$.

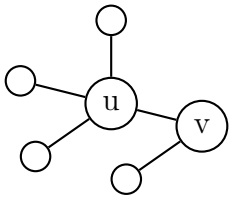
3. Берем путь $s \to c \to B \to t$, получая еще одно паросочетание (c, B).

Больше прийти в t нельзя ни по какому пути. Алгоритм завершается. Значит, размер максимального паросочетания равен 2. При этом $\mathbb{P} = \{(a, A);\ (c, B)\}$.

17 Контролирующее множество

Определение. Пусть есть граф $G = (\mathbb{V}, \mathbb{E})$, тогда $\mathbb{W} \in \mathbb{V}$ называется *контролирующим множеством*, если $\forall e = (u, v) \in \mathbb{E}$ устроено так, что $u \in \mathbb{W}$ или $v \in \mathbb{V}$.

Пример. Контролирующим множеством $\mathbb{W}$ будут являться вершины u и v.

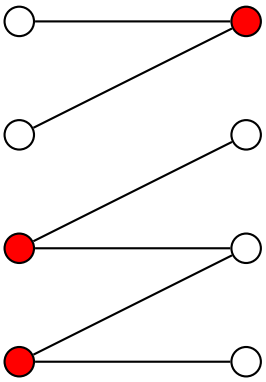


$$\mathbb{W} = \{u, v\}$$

Замечание. $\mathbb{W} = \mathbb{V}$ — всегда контролирующее множество.

Задача (о минимальном контролирующем множестве)**.** Будем решать эту задачу для двудольного графа.

Пример. Рассмотрим пример:



Здесь минимальным контролирующем множеством являются вершины, окрашенные в красный цвет.

Утверждение. *В любом двудольном графе* $G = (\mathbb{U} \cup \mathbb{V}, \mathbb{E})$ *размер контролирующего множества не может быть меньше максимального количества паросочетаний.*

$$|\mathbb{W}| \geq |\mathbb{P}|$$

Доказательство. Пусть есть случайный двудольный граф, у в котором вершины имеют только единичную степень. Тогда контролирующим множеством можно выбрать ровно в два раза меньше таких вершин. При этом, по определению максимального паросочетания, эти оставшиеся вершины будут соединены с соответствующими только единожды, то есть у каждого ребра из множества паросочетаний точно есть вершина из контролирующего множества. □

Утверждение. *На самом деле, размер контролирующего множества равен максимальному количеству паросочетаний.*

$$|\mathbb{W}| = |\mathbb{P}|$$

Доказательство. Построим на графе максимальное паросочетание с помощью задачи о потоках (по алгоритму Форда-Фалкерсона). Рассмотрим получившийся разрез $\mathbb{C}$.

Пусть u — ребра из s, v — ребра до t. Ребро из $(v \cap \mathbb{V}_1)$ в $(u \cap \mathbb{V}_2)$ не может идти, так как $\mathbb{V}_1 \cap \mathbb{V}_2 = \emptyset$ по определению разреза. Если ребро идет в обратную сторону, то до него дошли ранее через v и множество $\mathbb{V}_1$ должно было измениться. Пришли к противоречию. Следовательно, в минимальном разрезе нет ребер между $(v \cap \mathbb{V}_1)$ и $(u \cap \mathbb{V}_2)$, и минимальным контролирующим множеством является

$$\mathbb{W} = (u \cap \mathbb{V}_2) \cup (v \cap \mathbb{V}_1)$$

□

Сделаем некоторые выводы:
Вывод 1. Из построения максимального паросочетания следует построение минимальное контролирующее множество.
Вывод 2. Размер разреза определим с вводом переменных:

- $|u| = x$
- $|u \cap \mathbb{V}_2| = a$
- $|v \cap \mathbb{V}_1| = b$

Тогда:

$$|\mathbb{C}| = \sum_{e=(u,v)} 1 = x - a + b,$$

если $u \in \mathbb{V}_1$ и $v \in \mathbb{V}_2$.

18 Обход графа

Под *обходом графа* имеется в виду **поиск в глубину** и **поиск в ширину**, а также связанные с этим алгоритмы.

18.1 Общие положения

Определимся с тем, как устроен обход графа. В первую очередь необходима линейная структура данных D: *стек* или *очередь*. Для них есть операции: положить вершину v в D ($v \to D$), посмотреть вершину ($\prec D$) и достать ее оттуда ($D \to$).

- Под *стеком* подразумевается структура "первый вошел — последний вышел".
- Под *очередью* подразумевается структура "первый вошел — первый вышел".

Пример. Рассмотрим такие примеры:

Стек	
$a \to D$	a
$b \to D$	ba
$c \to D$	cba
$\prec D = c$	
$D \to$	ba
$\prec D = b$	

Очередь	
$a \to D$	a
$b \to D$	ab
$c \to D$	abc
$\prec D = a$	
$D \to$	bc
$\prec D = b$	

18.2 Алгоритм

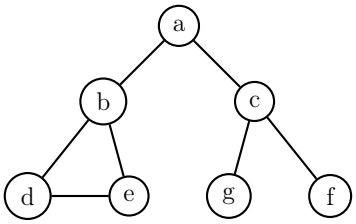
Для поиска в глубину используется $D =$ стек, а для поиска в ширину — $D =$ очередь. Рассмотрим алгоритм:

1. Поиск всегда начинается с определенной вершины. Добавляем в структуру D начальную вершину v_0: $v_0 \to D$;
2. Пока D не пуст, рассматриваем вершину в ней: $\prec D$.

   Если есть ребро $e = (u, v)$, такое что мы не были в v (нет пометки), то кладем вершину в структуру D (и ставим пометку): $v \to D$;

   Иначе нужно достать вершину из D: $D \to$.

Пример. Рассмотрим обход на примере графа G:

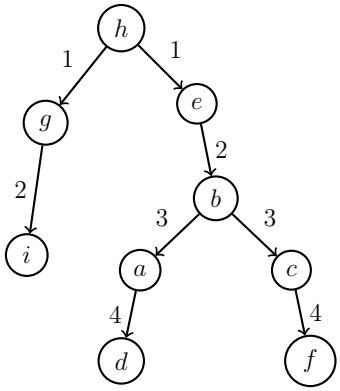


В глубину	В ширину
a	a
ba	ab
dba	abc
$edba$	bc
dba	bcd
ba	$bcde$
a	cde
ca	$cdef$
fca	$cdefg$
ca	$defg$
gca	efg
ca	fg
a	g
$\emptyset$	$\emptyset$

19 Алгоритмы, основанные на обходе графов

Утверждение. *Поиск в глубину перебирает вершины в таком же порядке, что и Алгоритм Дейкстры (все ребер берутся за единицу).*

Доказательство. Действительно, добавление вершины в D (стек) является аналогом релаксации ребра (u, v), а удаление вершины из D — убирание вершины с минимальным расстоянием.



□

Задача (пример)**.** Найти путь в лабиринте от начальной точки до конечной (стены помечены квадратом, начальная точка кружком, конечная – крестом):

						○
	□	□	□	□	□	
					□	
□	□	□	□		□	
	□	□	□	□	□	
				×	□	

⇒

6	5	4	3	2	1	○
7	□	□	□	□	□	1
8	9	10	9	8	□	2
□	□	□	□	7	□	3
10	9	8	7	6	5	4
11	□	□	□	□	□	5
12	13	14	15	×	□	6

Результат (путь помечен стрелочками):

						○
	□	□	□	□	□	↓
					□	↓
□	□	□	□		□	↓
↓	←	←	←	←	←	←
↓	□	□	□	□	□	
→	→	→	→	×	□	

19.1 Полный поиск в глубину

Алгоритм заключается в том, что пока есть непосещенная вершина u, нужно выполнять поиск в глубину от этой вершины (актуально для несвязных графов и ориентированных графов).

Примечание. Будем использовать сокращения:

- Поиск в глубину — DFS (depth-first search)
- Поиск в ширину – BFS (breadth-first search)

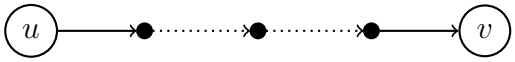
Утверждение. *Пусть $G = (\mathbb{V}, \mathbb{E})$ — ориентированный граф без циклов, в котором есть путь $u \to v$ (но нет пути $v \to u$). Тогда после полного DFS*

$$b(u) > b(v),$$

где b — обратный номер вершины.

Доказательство. Делаем DFS, тогда возникает вопрос — в какую вершину мы попали раньше?

1. Сначала попали в вершину u



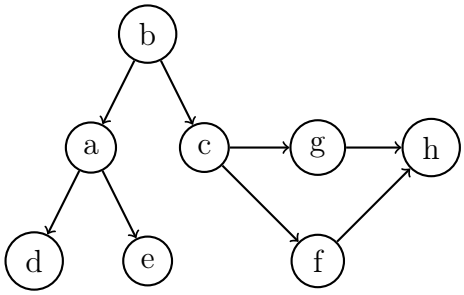
В стеке будет путь $u \to \cdots \to v$, значит сначала из стека уйдет v, а потом u.

2. Сначала попали в вершину v
   Циклы в графе отсутствуют, следовательно мы закончим поиск из v, так и не попав в u. Соответственно, номер вершине v мы присвоим раньше, и из стека она уйдет раньше.

Получили, что в обоих случаях вершина v будет иметь меньший обратный номер, чем вершина u. □

Следствие. *С помощью этого утверждения выполняется эффективный* алгоритм топологической сортировки.

Пример (топологическая сортировка)**.** Отсортируем следующий граф:



Ответ. Топологическая сортировка имеет вид:

d	e	a	h	g	f	c	b
1	2	3	4	5	6	7	8

Замечание. Можно заметить, что все имеющиеся ребра графа в табличной записи идут справа налево.

20 Компоненты сильный связности

Определение. Если есть граф $G = (\mathbb{V}, \mathbb{E})$, то на нем можно ввести отношение взаимной достижимости:

$$u \leftrightarrow v, \text{ если есть пути } u \to v, v \to u$$

Компонентами сильной связности называются подграфы основного графа G, среди которых найдутся такие, что из одного подграфа нельзя попасть в другой (что называется *сильной связью*).

Примечание. Компоненты сильной связности можно представить в виде *графа конденсации* — то есть тем графом, в котором одна компонента сильной связности представляется одной вершиной.

Замечание. Граф конденсации не имеет циклов.

Утверждение. *Пусть граф $G = (\mathbb{V}, \mathbb{E})$ — ориентированный, и имеет G° — граф конденсации G. Проведем полный DFS для графа G. Тогда, если в G° есть путь из $u^\circ \to v^\circ$, то $\forall u \in u^\circ, v \in v^\circ$*

$$\max_{u \in u^\circ} b(u) > \max_{v \in v^\circ} b(v)$$

Доказательство. Аналогично прошлому утверждению. □

Следствие (Поиск компонент связности)**.** *Алгоритм состоит из двух шагов:*

1. *Производится полный DFS.*
2. *Находим $b(u) \to \max$. Делаем DFS по обратным ребрам.*