# Алгоритмы и структуры данных

## Алгоритмы на графах

д.т.н., проф. Трифонов Петр Владимирович

# Содержание лекции

# Основные понятия I

| Неориентированные графы | Ориентированные графы |
|---|---|
| *Неориентированный граф* задается парой $G = (V, E)$, где $V$ — конечное множество вершин, и $E$ — множество неупорядоченных пар из $V$, называемое множеством ребер графа. Будем считать, что все элементы $E$ содержат по две различные вершины | *Ориентированный граф* задается парой $G = (V, E)$, где $V$ — конечное множество вершин, а $E \subset V^2$ — множество дуг графа. |
| Если $e = \{u, v\} \in E$, то говорят, что ребро $e$ соединяет вершины $u$ и $v$. | Если $e = (u, v) \in E$, то говорят, что дуга $e$ ведет из вершины $u$ в вершину $v$. |
| Вершины $u$ и $v$, соединенные некоторым ребром, называют смежными. | Вершины $u$ и $v$, такие что из $u$ в $v$ ведет некоторая дуга, называют смежными, причем $u$ называют началом дуги, а $v$ — ее концом. Если начало и конец дуги совпадают, то дугу называют петлей. |

# Основные понятия II

| | |
|---|---|
| Ребро называют инцидентным вершине $v$, если она является одним из его концов. | Дугу $(u, v)$ называют заходящей в вершину $v$ и исходящей из $u$. Дугу, заходящую или исходящую из некоторой вершины, называют инцидентной ей. |
| Степенью вершины $\deg v$ называют число инцидентных ей ребер. | Полустепенью захода вершины называют число заходящих в нее дуг. Полустепенью исхода вершины $v$ называют число исходящих из нее дуг. Степенью вершины называют сумму степеней захода и исхода. |
| Множество $\Gamma(v) = \{u \mid \{u, v\} \in E\}$ называют множеством вершин, смежных с $v$ | Для вершины $v$ множество $\Gamma(v) = \{x \in V \mid (v, x) \in E\}$ называют множеством ее преемников, а множество $\Gamma^{-1}(v) = \{x \in V \mid (x, v) \in E\}$ — множеством предшественников. |

# Основные понятия III

| | |
|---|---|
| *Цепь* в неориентированном графе — последовательность вершин $v_0, v_1, \ldots, v_n$, такая что $\{v_i, v_{i+1}\} \in E$ для всех $i$ | *Путь* в ориентированном графе — последовательность вершин $v_0, v_1, \ldots, v_n$, таких что $(v_i, v_{i+1}) \in E$ |
| Говорят, что вершина $v$ достижима из вершины $u$, если существует цепь $v_0, \ldots, v_n : v_0 = u, v_n = v$ Таким образом, определено рефлексивное, симметричное и транзитивное отношение достижимости. Простая цепь — цепь, все вершины которой, кроме, возможно, нулевой и последней, попарно различны | Говорят, что вершина $v$ достижима из вершины $u$, если существует путь $v_0, \ldots, v_n : v_0 = u, v_n = u$. Таким образом, определено рефлексивное, транзитивное, но в общем случае не симметричное и не антисимметричное отношение достижимости. Простой путь — путь, все вершины которого, кроме, возможно, нулевой и последней, попарно различны. |
| Простая (произвольная) цепь ненулевой длины с совпадающими началом и концом, называется циклом (замкнутой цепью). | Простой (произвольный) путь ненулевой длины, начало и конец которого совпадает, называют контуром (замкнутым путем). |

# Основные понятия IV

| | |
|---|---|
| Неориентированный граф, не содержащий циклов, называют ациклическим | Ориентированный граф, не содержащий контуров, называют бесконтурным. |

# Простые цепи и пути

**Теорема**

*Для любой цепи (пути), соединяющей две вершины неориентированного (ориентированного) графа, существует простая цепь (путь), соединяющая те же вершины.*

**Доказательство.**

Пусть $u$ и $v$ — концы рассматриваемой цепи. Если $u = v$, то утверждение очевидно, как и в том случае, когда эта цепь простая. Иначе предположим, что вершина $w$ повторяется в цепи несколько раз. Удалим из цепи все элементы между первым и последним вхождением в нее $w$, включая последнее. Полученная последовательность также является цепью. Будем действовать аналогично, пока она не станет простой. В силу конечности множества вершин этот процесс завершится за конечное число шагов. □

# Подграфы I

**Определение**

Неориентированный (ориентированный) граф $G' = (V', E')$ называют подграфом неориентированного (ориентированного) графа $G = (V, E)$, если $V' \subset V, E' \subset E$ и $E' \subset V' \times V'$.

- Если хотя бы одно из этих включений является строгим, подграф называют собственным
- Если $V' = V$, то подграф называют остовным.
- Неориентированный граф называется *связным*, если любые его вершины связаны цепью. Ориентированный граф называют связным, если для любых двух его вершин $u$ и $v$ вершина $u$ достижима из $v$ или $v$ достижима из $u$.
- *Компонентой связности* графа называется максимальный его связный подграф

# Подграфы II

- Ориентированный граф называют *сильно связным*, если любые две его вершины достижимы друг из друга
- Бикомпонентой ориентированного графа называют его максимальный сильно связный подграф
- Неориентированный граф $G' = (V', E')$ называют ассоциированным с ориентированным графом $G = (V, E)$, если $V' = V$ и $E' = \{\{u, v\} | (u, v) \in E\}$
- Ориентированный граф называют *слабо связным*, если ассоциированный с ним неориентированный граф связный
- Компонентой слабой связности ориентированного графа называется называется его максимальный слабо связный подграф

# Матрица инциденций

- Пусть $n = |V|$, $m = |E|$
- Неориентированный граф может быть представлен в виде $n \times m$ матрицы

$$A = ||a_{ij}|| : a_{ij} = \begin{cases} 1 & \text{если для } i\text{-й вершины } j\text{-е ребро инцидентно} \\ 0 & \text{иначе} \end{cases}$$

- Ориентированный граф может быть представлен в виде матрицы

$$A = ||a_{ij}|| : a_{ij} = \begin{cases} 1 & \text{если для } i\text{-й вершины } j\text{-я дуга выходящая} \\ -1 & \text{если для } i\text{-й вершины } j\text{-я дуга заходящая} \\ 0 & \text{иначе} \end{cases}$$

- Определенная таким образом матрица носит название матрицы инциденций

# Матрица смежности

- Представление графа в виде матрицы инциденций является крайне неэкономным, т.к. каждый ее столбец содержит ровно два ненулевых элемента
- $n \times n$ матрица смежности вершин $B = ||b_{ij}||$. Для неориентированных графов

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

Для ориентированного графа

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

- для неориентированного графа $B = B^T$

# Список смежности и матрица достижимости

- Список смежности: для каждой вершины может быть указан список вершин, непосредственно достижимых из нее
- Матрица достижимости: $n \times n$ матрицу $V = ||v_{ij}|| : v_{ij} = 1$ тогда и только тогда, когда $j$-я вершина достижима из $i$-ой.

# Задачи о путях на графе

- Вычисление матрицы достижимости. Эквивалентна нахождению транзитивного (и рефлексивного) замыкания отношения непосредственной достижимости вершин на графе
- Вычисление наименьших расстояний между всеми парами вершин на графе.
  - Всем дугам графа сопоставлены некоторые веса
  - Расстоянием между двумя вершинами по некоторому пути является сумма весов дуг, входящих в него
  - Минимальным расстоянием между вершинами называется минимальное из расстояний между ними по всем возможным путям.

Задача не всегда имеет решение, как, например, в случае наличия в графе петли с отрицательным весом. Каждое включение этой петли в путь будет приводить к уменьшению его “длины”.

# Полукольцо

## Определение

*Полукольцом* называется алгебра $\mathcal{S} = (S, +, \cdot, 0, 1)$, такая, что для произвольных элементов $a, b, c$ множества $S$ выполняются следующие равенства (аксиомы полукольца):

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

Полукольцо называется идемпотентным, если для любого $a \in S$ выполняется $a + a = a$

# Взвешенный ориентированный граф

**Определение**

Под взвешенным ориентированным графом понимается пара $W = (G, \phi)$, где $G = (V, E)$ — ориентированный граф, а $\phi : E \to \mathcal{R}$ — весовая функция (или функция разметки), множеством значений которой является некоторое идемпотентное полукольцо $\mathcal{R} = (R, \oplus, \cdot)$, причем $\phi(e) \neq 0$

Взвешенный граф может быть задан матрицей меток дуг

$$A = ||a_{ij}|| : a_{ij} = \begin{cases} \phi((i.j)) & \text{если из вершины } i \text{ в вершину } j \text{ ведет дуга} \\ 0 \in R & \text{иначе} \end{cases}.$$

# Полукольца и графы

- Введем полукольцо $\mathcal{R}_+ = (\mathbb{R}^+ \cup \{+\infty\}, \min, +)$, где $\min(a, b)$ –операция нахождения минимума двух вещественных чисел
  - Умножение ($\odot$): сложение вещественных чисел
  - Сложение ($\oplus$): операция нахождения минимума
  - $+\infty$ — нулевой элемент (нейтральный относительно операции сложения в полукольце). Он же может рассматриваться как нулевой элемент относительно операции умножения в полукольце.
- Булево полукольцо $\mathcal{B} = (\{0, 1\}, +, \cdot)$, в котором в качестве операции сложения используется операция логического ИЛИ, а умножения — логического И.

Матрицы расстояний между узлами и достижимости могут быть найдены из итерации матрицы $A$, т.е. матрицы $A^* = \sum_{i=0}^{\infty} A^i$, где вычисления производятся, соответственно, в полукольцах $\mathcal{R}_+$ или $\mathcal{B}$.

# Поиск матриц достижимости и расстояний между узлами

- Метка пути, ведущего из вершины $v_i$ в вершину $v_j$ — произведение в полукольце $R$ меток входящих в него дуг в порядке их следования
- Метка пути нулевой длины — единица полукольца
- Стоимость прохождения из вершины $v_i$ в вершину $v_j$ — сумма в полукольце $R$ меток всех путей, ведущих из вершины $v_i$ в вершину $v_j$ Таких путей может быть бесконечное (но не более чем счетное) множество. Поэтому будем использовать точную верхнюю грань последовательности таких сумм
- При отсутствии путей между двумя вершинами стоимость прохождения между ними равна 0.

# Корректность алгоритма

**Теорема**

*Элемент $a_{ij}^{(l)}$ матрицы $A^l, l \geq 0$ равен стоимости прохождения из вершины $v_i$ в вершину $v_j$ по всем путям длины $l$.*

**Доказательство.**

При $l = 0$ утверждение очевидно, т.к. $A^0 = I$. При $l = 1$ утверждение также очевидно. Пусть $a_{ij}^{(l-1)}$ равен стоимости прохождения из вершины $i$ в вершину $j$ путями длины $l - 1$. Тогда

$$a_{ij}^{(l)} = \sum_{k=1}^{n} a_{ik}^{(l-1)} \odot a_{kj}.$$

Утверждение леммы вытекает из того, что все пути длины $l$ получаются присоединением одной дуги в конец путей длины $l - 1$. □

# Идемпотентные полукольца

- Идемпотентное полукольцо: $\forall x \in \mathcal{R} : x \oplus x = x$.
- отношение порядка $\preceq$: $x \preceq y \Leftrightarrow x \oplus y = y$
  - В силу идемпотентности сложения это отношение рефлексивно
  - Из $x \preceq y$ и $y \preceq x$ вытекает, что $x \oplus y = y$ и $y \oplus x = x$, откуда, в силу коммутативности сложения, $x = y$, т.е. это отношение антисимметрично
  - Из $x \preceq y$ и $y \preceq z$ вытекает, что $x \oplus y = y$ и $y \oplus z = z$, откуда $x \oplus z = x \oplus (y \oplus z) = (x \oplus y) \oplus z = y \oplus z = z$, т.е. $x \preceq z$, что означает транзитивность

# Упорядоченные множества

**Определение**

Элемент $x$ множества $A$ с отношением порядка $\preceq$ называется *минимальным*, если $\nexists y \in A : y \preceq x \wedge y \neq x$. *Наименьшим элементом* упорядоченного множества называется такой его элемент $x \in A$, что $\forall y \in A : x \preceq y$.

**Определение**

Пусть $(A, \preceq)$ — упорядоченное множество и $B \subset A$. Элемент $a \in A$ называется верхней (нижней) гранью множества $B$, если $\forall x \in B : (x \preceq a)$ (или $x \succeq a$). Наименьший (наибольший) элемент множества верхних (нижних) граней является точной верхней (нижней) гранью множества $B$ и обозначается $\sup B$ ($\inf B$).

**Определение**

Упорядоченное множество $(A, \preceq)$ называется *индуктивным*, если оно содержит наименьший элемент и всякая неубывающая последовательность элементов этого множества имеет точную верхнюю грань.

# Непрерывные отображения

**Определение**

Пусть $(M_1, \preceq)$ и $(M_2, \sqsubseteq)$ — индуктивные упорядоченные множества. Отображение $f : M_1 \longrightarrow M_2$ называется *непрерывным*, если для любой неубывающей последовательности $\alpha_1, \ldots, \alpha_n, \ldots$ элементов множества $M_1$ образ ее точной верхней грани равен точной верхней грани последовательности образов $f(\alpha_1), \ldots, f(\alpha_n), \ldots$, т.е. $f(\sup \alpha_n) = \sup f(\alpha_n)$.

**Определение**

Отображение $f : M_1 \longrightarrow M_2$ упорядоченных множеств называется *монотонным* (сохраняющим порядок), если $\forall a, b \in M_1 : a \preceq b \Rightarrow f(a) \sqsubseteq f(b)$.

Это определение не следует путать с определением монотонных функций, используемым в матанализе. Функция $f : \mathbb{R} \longrightarrow \mathbb{R}$, монотонная в смысле матанализа, будет являться монотонной в вышеуказанном смысле, только если она является неубывающей.

# Свойства непрерывных отображений

**Теорема**

*Всякое непрерывное отображение одного индуктивного упорядоченного множества в другое монотонно.*

**Доказательство.**

Пусть $f : M_1 \longrightarrow M_2$ непрерывно, $M_1, M_2$ — индуктивные множества. Пусть $a, b \in M_1, a \preceq b$. Образуем неубывающую последовательность $\{x_n\}_{n\in\mathbb{N}}, x_1 = a, x_n = b, n \geq 2$. В нее $\sup x_n = b$. В силу непрерывности $f(b) = f(\sup x_n) = f(\sup\{a, b\}) = \sup\{f(a), f(b)\} \Rightarrow f(a) \sqsubseteq f(b)$. □

# Примеры

**Пример**

Функция $f = \begin{cases} 0,5x, & 0 \leq x < 0.5 \\ 0,5 + 0,5x, & 0.5 \leq x \leq 1 \end{cases}$ является монотонной, но не непрерывной.

**Пример**

Пусть $f(x) = x^2$ определена на $[-1, 1]$. Рассмотрим последовательность $x_i = -1/i, i \in \mathbb{N}$. $\sup x_i = 0$, но $\sup f(x_i) = 1 \neq f(\sup x_i) = 0$, т.е. отображение не является непрерывным в смысле вышеприведенного определения

# Неподвижная точка отображения

**Определение**

Элемент $a \in A$ называется *неподвижной точкой* отображения $f : A \longrightarrow A$, если $f(a) = a$. Элемент $a$ называется наименьшей неподвижной точкой отображения $f : A \longrightarrow A$, если он является наименьшим элементом множества неподвижных точек $f$.

# Теорема о неподвижной точке I

**Теорема (Клини-Тарского)**

*Любое непрерывное отображение $f$ индуктивного упорядоченного множества $(M, \preceq)$ в себя имеет наименьшую неподвижную точку.*

Пусть $\mu \in M$ — наименьший элемент множества $M$. Пусть $f^0(x) = x, f^n(x) = f(f^{n-1}(x)), n > 0$. Рассмотрим последовательность элементов $M$

$$\{f^n(\mu)\}_{n \geq 0} = \{\mu, f(\mu), \ldots, f^n(\mu), \ldots\}. \tag{1}$$

Докажем, что эта последовательность неубывающая. Т.к. $\mu$ — наименьший элемент, $\mu \preceq f(\mu)$. Пусть для некоторого $n f^{n-1}(\mu) \preceq f^n(\mu)$. Т.к. $f$ непрерывно, по теореме 11 оно является монотонным, следовательно $f^n(\mu) = f(f^{n-1}(\mu)) \preceq f(f^n(\mu)) = f^{n+1}(\mu)$. Таким образом, последовательность (1) является неубывающей. Тогда по определению

# Теорема о неподвижной точке II

индуктивного множества она имеет точную верхнюю грань $a = \sup_{n\geq 0} f^n(\mu)$, т.е. $\forall n : f^n(\mu) \preceq a$. Ясно, что $a$ является также верхней гранью для любой подпоследовательности (1), в т.ч. для $n \geq k > 0$. Пусть $b$ — какая-то иная верхняя грань такой последовательности с усеченным началом, т.е. $\forall n \geq k : x_n \preceq b$. Т.к. исходная последовательность неубывающая, $x_p \preceq x_k, p = 0..k-1$, а следовательно $x_p \preceq x_k \preceq b$. Т.к. $a = \sup_{n\geq 0} f^n(\mu)$, $a \preceq b$, т.е. является точной верхней гранью любой последовательности вида (1) с усеченным началом.
В силу непрерывности $f$ имеем

$$f(a) = f(\sup_{n\geq 0} f^n(\mu)) = \sup_{n\geq 0} f(f^n(\mu)) = \sup_{n\geq 0} f^{n+1}(\mu)$$

Но

$$\sup_{n\geq 0} f^{n+1}(\mu) = \sup\{f^1(\mu), f^2(\mu), \ldots\} = \sup_{n\geq 1} f^n(\mu) = a,$$

# Теорема о неподвижной точке III

т.е. $a$ является неподвижной точкой.

Докажем, что найденная вышеописанным способом неподвижная точка является наименьшей. Пусть $\exists y \in M : f(y) = y$. Т.к. $\mu \preceq y$, а $f$, будучи непрерывным, монотонно, то $f(\mu) \preceq f(y) = y, f^2(\mu) \preceq f^2(y) = y$ и т.д., т.е. $\forall n \geq 0 : f^n(\mu) \preceq y$, т.е. $y$ является верхней гранью последовательности $\{f^n(\mu)\}$. Т.к. $a$ является точной верхней гранью, $a \preceq y$, т.е. $a$ — наименьшая неподвижная точка.

### Пример

Пусть $f(x) = \frac{1}{2}x + \frac{1}{4} : [0,1] \longrightarrow [0,1]$. Применяя метод, использованный в доказательстве теоремы, получим $f^0(0) = 0, f^1(0) = 1/4, f^2(0) = 3/8, f^n(0) = \frac{2^n-1}{2^{n+1}} \underset{n\to\infty}{\longrightarrow} 1/2$.

# Точная верхняя грань подмножества идемпотентного полукольца

**Теорема**

*Если $A$ — конечное подмножество носителя идемпотентного полукольца, то $\sup A = \sum_{a_i \in A} a_i$, где при суммировании используется операция сложения $\oplus$ из сигнатуры полукольца*

**Доказательство.**

Пусть $n = |A|$ и $a = \sum_{a_i \in A} a_i$. Тогда для произвольного $a_j \in A$ получим $a_j \oplus a = a_j \oplus (a_1 \oplus \cdots \oplus a_j \oplus \cdots \oplus a_n) = a_1 \oplus \cdots \oplus a_j \oplus a_j \oplus \cdots \oplus a_n = a_1 \oplus \cdots \oplus a_j \oplus \cdots \oplus a_n = a$, т.е. $a_j \preceq a$, т.е. $a$ является верхней гранью $A$. Рассмотрим произвольную верхнюю грань $b$ этого множества. Тогда для любого $a_i \in A$ справедливо $a_i \preceq b$, т.е. $a_i \oplus b = b$. Таким образом, $b \oplus a = (b \oplus a_1) \oplus (a_2 \oplus \cdots \oplus a_n) = b \oplus a_2 \oplus \cdots \oplus a_n = \cdots = b$. Следовательно, $a \preceq b$, т.е. $a$ является точной верхней гранью $A$ □

# Пример

В полукольце $\mathcal{R}^+$ точной верхней гранью множества $A$ в смысле отношения $\preceq$ является его наименьший элемент в смысле обычного числового порядка.

# Замкнутое полукольцо

## Определение

Полукольцо $\mathcal{S} = (S, \oplus, \cdot)$ называется *замкнутым*, если:

1. оно идемпотентно
2. любая последовательность его элементов имеет точную верхнюю грань относительно $\preceq$
3. Операция умножения сохраняет точные верхние грани последовательностей, т.е. $a \sup X = \sup(aX), (\sup X)a = \sup(Xa)$

# Полукольца как индуктивные упорядоченные множества

- Замкнутые полукольца являются индуктивными упорядоченными множествами
  - наименьшим элементом служит нуль полукольца
  - точной верхней гранью произвольной (в частности, неубывающей) последовательности $\{x_n\}_{n\in\mathbb{N}}$ является бесконечная сумма $\sum_{n\in\mathbb{N}} x_n$
  - операция $f_a(x) = ax$ умножения на произвольный фиксированный элемент $a$ непрерывна, т.к. сохраняет точные верхние грани.
  - Для любого $a$ отображение $f_a(x)$ имеет наименьшую неподвижную точку.
- Итерацией элемента $x$ полукольца $\mathcal{R}$ будем называть точную верхнюю грань последовательности всех степеней $x$, т.е., в силу теоремы 17,

$$x^* = \sum_{i\geq 0} x^i.$$

# Решение уравнений в полукольцах

**Теорема**

*Наименьшими решениями уравнений*

$$x = ax \oplus b \tag{2}$$

*и*

$$x = xa \oplus b \tag{3}$$

*в замкнутом полукольце являются, соответственно,* $x = a^*b$ *и* $x = ba^*$.

Рассмотрим отображение $f_{a,b}(x) = ax + b$. Поиск решения уравнения (2) сводится к нахождению наименьшей неподвижной точки этого отображения. Воспользуемся методом, приведенным в доказательстве теоремы Клини-Тарского. А именно, будем искать решение как $x = \sup_{n\geq 0} f^n_{a,b}(\mu)$, где $\mu = 0$ — наименьший элемент полукольца. Видно, что $f^0_{a,b} = 0, f^1_{a,b}(0) = b, f^2_{a,b}(0) = ab \oplus b = (a \oplus 1)b, \ldots, f^n_{a,b}(0) = (a^{n-1} \oplus \cdots \oplus a^0)b$.

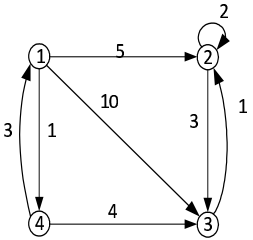
# Итерация матрицы

итерация матрицы может быть найдена из систем уравнений

$$\xi = A\xi + e_j, j = 1..n,$$

где $e_j$ — $j$-ый единичный вектор. Их решения имеют вид $\xi_j = A^* e_j$, т.е. $\xi_j$ является $j$-ым столбцом $A^*$.

# Пример: поиск матрицы достижимости I




Матрица смежности $A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$.

Система уравнений в полукольце $\mathcal{B}$ для определения первого столбца матрицы $A^*$:

$$\begin{cases} \xi_1 = \xi_2 \oplus \xi_3 \oplus \xi_4 \oplus 1 \\ \xi_2 = \xi_2 \oplus \xi_3 \oplus 0 \\ \xi_3 = \xi_2 \oplus 0 \\ \xi_4 = \xi_1 \oplus \xi_3 \oplus 0 \end{cases} \Rightarrow \begin{cases} \xi_2 = \xi_2 \oplus \xi_3 \\ \xi_3 = \xi_2 \\ \xi_4 = \xi_2 \oplus \xi_3 \oplus \xi_4 \oplus 1 \end{cases}$$

# Пример: поиск матрицы достижимости II

В соответствии с теоремой 19 наименьшее решение уравнения $\xi_2 = \xi_2 \oplus \xi_3$ может быть получено как $\xi_2 = 1^*(\xi_3 \oplus 0)$. В полукольце $\mathcal{B}$ итерация любого элемента равна 1, поэтому $\xi_2 = \xi_3$. Таким образом, получим

$$\begin{cases} \xi_3 &= \xi_3 \\ \xi_4 &= \xi_3 \oplus \xi_4 \oplus 1 \end{cases}$$

Аналогичным образом находим $\xi_3 = 0$ и $\xi_4 = \xi_4 \oplus 1$, $\xi_4 = 1^* \cdot 1 = 1$, откуда следует $\xi_1 = 1$. Таким образом, первый столбец матрицы $A^*$ равен $(1\,0\,0\,1)^T$.
Для нахождения второго столбца составим систему уравнений

$$\begin{cases} \xi_1 &= \xi_2 \oplus \xi_3 \oplus \xi_4 \oplus 0 \\ \xi_2 &= \xi_2 \oplus \xi_3 \oplus 1 \\ \xi_3 &= \xi_2 \oplus 0 \\ \xi_4 &= \xi_1 \oplus \xi_3 \oplus 0 \end{cases}$$

# Пример: поиск матрицы достижимости III

Действуя аналогично, получим

$$\begin{cases} \xi_2 &= \xi_2 \oplus \xi_3 \oplus 1 \\ \xi_3 &= \xi_2 \oplus 0 \\ \xi_4 &= \xi_2 \oplus \xi_3 \oplus \xi_4 \oplus 0, \end{cases}$$

откуда $\xi_2 = 1^* \cdot (\xi_3 \oplus 1) = \xi_3 \oplus 1$, $\xi_3 = 1^* \cdot 1 = 1$, $\xi_4 = 1$ и $\xi_1 = 1$. Третий и четвертый столбцы могут быть вычислены аналогично. Окончательно, получим

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$

Таким образом, данный граф является связным и имеет две бикомпоненты связности: $\{1, 4\}$ и $\{2, 3\}$.

# Пример: поиск матрицы кратчайших расстояний I

Пусть матрица меток дуг имеет вид $\begin{pmatrix} \infty & 5 & 10 & 1 \\ \infty & 2 & 3 & \infty \\ \infty & 1 & \infty & \infty \\ 3 & \infty & 4 & \infty \end{pmatrix}$. Воспользуемся полукольцом $\mathcal{R}^+$. Заметим, что нулем этого полукольца является $\infty$, а единицей — число 0. Для нахождения первого столбца матрицы $A^*$ решим систему уравнений

$$\begin{cases} \xi_1 = 5 \cdot \xi_2 \oplus 10 \cdot \xi_3 \oplus 1 \cdot \xi_4 \oplus 0 \\ \xi_2 = 2 \cdot \xi_2 \oplus 3 \cdot \xi_3 \oplus \infty \\ \xi_3 = 1 \cdot \xi_2 \oplus \infty \\ \xi_4 = 3 \cdot \xi_1 \oplus 4 \cdot \xi_3 \oplus \infty \end{cases}$$

Из первого уравнения в силу определения операции сложения в этом полукольце непосредственно вытекает $\xi_1 = 0$. Из второго уравнения получим

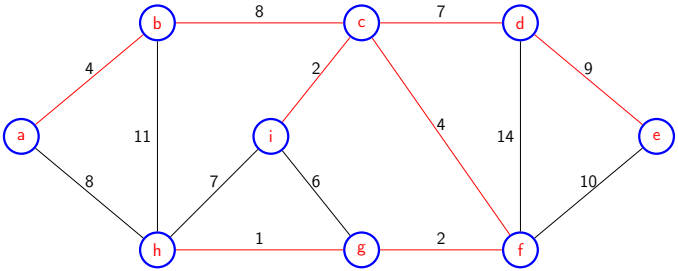
# Пример: поиск матрицы кратчайших расстояний II

$\xi_2 = 2^* \cdot (3 \cdot \xi_3 \oplus \infty) = 3 \cdot \xi_3$. Подставляя это выражение в третье уравнение, получим $\xi_3 = 1 \cdot (3 \cdot \xi_3) \oplus \infty = 4 \cdot \xi_3 \oplus \infty$, откуда $\xi_3 = 4^* \cdot \infty = \infty$. Из четвертого уравнения получим $\xi_4 = 3 \cdot 0 \oplus 4 \cdot \infty = 3 \oplus \infty = 3$. Таким образом, первый столбец матрицы $A^*$ имеет вид $(0\,\infty\,\infty\,3)^T$. Аналогичным образом могут быть найдены оставшиеся столбцы.

Окончательно, матрица расстояний равна $A^* = \begin{pmatrix} 0 & 5 & 5 & 1 \\ \infty & 0 & 3 & \infty \\ \infty & 1 & 0 & \infty \\ 3 & 5 & 4 & 0 \end{pmatrix}$.

# Минимальное остовное дерево

- Неориентированный граф $G' = (V', E')$ называют подграфом неориентированного графа $G = (V, E)$, если $V' \subset V, E' \subset E$ и $E' \subset V' \times V'$.
- Если $V' = V$, то подграф называют остовным.
- Ациклический остовный подграф называют остовным (покрывающим) деревом
- Пусть $w(u, v)$ — вес ребра $(u, v) \in E$. Минимальное остовное дерево: $W(G') = \sum_{(u,v)\in E'} w(u, v) \to \min$



# Алгоритм построения минимального остовного дерева

- МОД растет путем добавления к нему ребер по одному
- Инвариант: Пусть $A$ — множество ребер. Перед каждой итерацией $A$ образует некоторое подмножество ребер некоторого МОД
- На каждом шаге найдем безопасное ребро $(u, v)$, которое можно добавить в $A$ без нарушения инварианта

**Algorithm 1:** GenericMST(G,w)

```
1 A = ∅
2 while A не является МОД do
3     Найти безопасное для A ребро (u, v)
      A = A ∪ {(u, v)}
4 return A
```

# Разрез графа

- Разрезом $(S, V \setminus S)$ неориентированного графа $(V, E)$ называется разбиение $V : V = S \cup (V \setminus S), S \subset V$
- Ребро $(u, v)$ пересекает разрез $(S, V \setminus S)$, если один из концов ребра принадлежит $S$, а второй — $V \setminus S$
- Разрез согласован с множеством $A$ по ребрам, если ни одно ребро из $A$ не пересекает его
- Ребро, пересекающее разрез, является легким, если оно имеет минимальный вес, среди всех ребер пересекающих этот разрез

# Условие безопасности ребер I

**Теорема**

*Пусть $G = (V, E)$ — связный неориентированный граф с весовой функцией $w : E \to \mathbb{R}$. Пусть $A \subset E$ — подмножество ребер, входящих в некоторое МОД. Пусть $(S, V \setminus S)$ — разрез, согласованный с $A$ по ребрам, а $(u, v)$ — легкое ребро, пересекающее этот разрез. Тогда $(u, v)$ безопасно для $A$*

Пусть $G' = (V, E')$ — МОД, включающее $A$. Если $(u, v) \in E'$, теорема доказана. Иначе попытаемся построить другое МОД $G'' = (V, E'') : (u, v) \in E''$.
$(u, v)$ образует цикл с ребрами на пути от $u$ к $v$ в $E'$. На этом пути есть как минимум 1 ребро $(x, y) \in E'$, пересекающее разрез. $(x, y) \notin A$, т.к. разрез согласован с $A$ по ребрам. Т.к. $(x, y)$ — единственный путь от $u$ к $v$ в $E'$, его удаление разбивает $G'$ на два компонента. Добавление $(u, v)$ восстанавливает разбиение, образуя новое остовное дерево с $E'' = (E' \setminus \{(x, y)\}) \cup \{(u, v)\}$.

# Условие безопасности ребер II

Т.к. $(u, v)$ — легкое ребро, пересекающее $(S, V \setminus S)$, и $(x, y)$ также его пересекает,
$w(u, v) \leq w(x, y) \Rightarrow w(G'') = w(G) - w(x, y) + w(u, v) \leq w(G')$.
Т.к. $G'$ — МОД, $w(G') \leq w(G'') \Rightarrow w(G') = w(G'') \Rightarrow G''$ — МОД
$A \subset E', (x, y) \notin A \Rightarrow A \subset E'' \Rightarrow A \cup \{(u, v)\} \subset E''$. Т.к. $G''$ — МОД, $(u, v)$ безопасно для $A$

# Алгоритм построения минимального остовного дерева

- $A$ всегда ациклическое
- $(V, A)$ является лесом
- Число итераций равно $|V| - 1$

**Algorithm 2:** GenericMST(G=(V,E),w)

```
1 A = ∅
2 while A не является МОД do
3     Найти безопасное для A ребро (u, v) ∈ E
      A = A ∪ {(u, v)}
4 return A
```

**Следствие**

*Пусть $G = (V, E)$ — связный неориентированный граф с весовой функцией $w : E \to \mathbb{R}$. Пусть $A \subset E$ входит в МОД и пусть $C = (V_C, E_C)$ — связный компонент в лесу $G_A = (V, A)$. Если $(u, v)$ — легкое ребро, соединяющее $C$ с некоторым другим компонентом $G_A$, то ребро $(u, v)$ безопасно для $A$.*

**Доказательство.**

Разрез $(V_C, V \setminus V_C)$ согласован с $A$, $(u, v)$ — легкое ребро этого разреза $\Rightarrow$ оно безопасно $\square$

# Операции над непересекающимися множествами

- Пусть дан набор множеств $S_i : S_i \cap S_j = \emptyset$ при $i \neq j$
- Каждое множество идентифицируется некоторым однозначно заданным представителем
- Операции
  - *MakeSet*$(x)$ — создать множество с единственным представителем $x$
  - *Union*$(x, y)$ — объединить множества, содержащие $x$ и $y$. Множества не должны пересекаться. Представителем объединенного множества назначается произвольный его элемент
  - *FindSet*$(x)$ — возвратить указатель на представителя множества, содержащего $x$
- Реализация на основе односвязных списков: каждое множество соответствует списку
  - Элементы списка содержат объект, указатель на следующий член, указатель на представителя
  - Каждый список содержит указатели Head на представителя и Tail на последний элемент
  - Сложность *MakeSet* — $O(1)$, *FindSet* — $O(1)$
  - Union: переместить короткий список в конец длинного, обновив в каждом элементе короткого списка указатель на представителя

# Лес непересекающихся множеств

- Каждое множество может быть представлено в виде дерева, где узлы соответствуют элементам множеств
- Каждый член указывает на родительский узел
- Корень дерева считается представителем множества
- MakeSet(x) — создать дерево с 1 узлом; сложность $O(1)$
- FindSet(x) — проход к корню дерева
- Union(x,y) — корень одного дерева указывает на корень другого
- Последовательность операций Union может привести к созданию дерева, являющегося линейной цепочкой узлов

# Лес непересекающихся множеств: эвристики

- Объединение по рангу: корень дерева с меньшим числом узлов должен указывать на корень дерева с большим числом узлов
- Ранг каждого узла — верхняя граница высоты его поддерева
- Сжатие пути: перевешивание узлов непосредственно к корню
- Сложность $m$ операций *Union* над $n$ объектами $O(m\alpha(n))$, где $\alpha(n) = O(\log n)$

**Algorithm 3:** FindSet(x)

```
1 if x ≠ x.p then
2     x.p=FindSet(x.p)
3 return x.p
```

**Algorithm 4:** MakeSet(x)

```
1 X=new Node
2 X.val=x;X.p=X;X.rank=0
```

**Algorithm 5:** Union(x,y)

```
1 Link(FindSet(x),FindSet(y));
```

**Algorithm 6:** Link(x,y)

```
1 if x.rank>y.rank then
2     y.p=x
3 else
4     x.p=y
5     if x.rank=y.rank then
6         y.rank=y.rank+1
```

# Алгоритм Крускала

**Algorithm 7:** MSTKruskal(G,w)

```
1 A = ∅
2 for v ∈ V[G] do
3     MakeSet(v)
4 Отсортировать ребра из E по
    возрастанию w
5 for (u, v) ∈ E в порядке возрастания
    веса do
6     if FindSet(u)≠FindSet(v) then
7         A := A ∪ {(u, v)}
8         Union(u,v)
9 return A
```

- Непересекающиеся множества вершин различных деревьев в лесу $(V, A)$
- На каждом шаге добавляется ребро минимального веса, соединяющее различные деревья
- Сложность операций в строке 6 $O(|E|\alpha(V))$
- Общая сложность операций над непересекающимися множествами $O((|V| + |E|)\alpha(|V|))$
- Сложность сортировки $O(|E| \log |E|)$
- $|E| \geq |V| - 1 \Rightarrow$ сложность $O(|E| \log |E|)$

# Алгоритм Прима

**Algorithm 8:** MSTPrim(G=(V,E),w,r)

```
1 for u ∈ V do
2     u.key = ∞; u.parent = NULL
3 r.key=0;Q=V
4 while Q ≠ ∅ do
5     u=Q.ExtractMin()
6     for v ∈ Adj(u) do
7         if v ∈ Q ∧ w(u,v) < v.key
            then
8             v.parent=u;v.key=w(u,v)
```

- Ребра в $A$ всегда образуют единое дерево с корнем $r$
- На каждом шаге к дереву добавляется легкое ребро, соединяющее его с оставшейся частью графа
- В приоритетной очереди $Q$ хранятся вершины, не вошедшие в дерево
- ПО использует упорядочение по key. $v.key$ равно минимальному весу среди всех ребер, соединяющих вершину $v$ с вершинами дерева

$$A = \{(v, v.parent) | v \in V \setminus \{r\} \setminus Q\}$$

# Алгоритм Прима

**Algorithm 9:** MSTPrim(G=(V,E),w,r)

```
1 for u ∈ V do
2     u.key = ∞; u.parent = NULL
3 r.key=0;Q=V
4 while Q ≠ ∅ do
5     u=Q.ExtractMin()
6     for v ∈ Adj(u) do
7         if v ∈ Q ∧ w(u, v) < v.key
            then
8             v.parent=u;v.key=w(u,v)
```

- Перед каждой итерацией *While* $A = \{(v, v.parent) | v \in V \setminus \{r\} \setminus Q\}$
- Вершины, уже помещенные в МОД, принадлежат $V \setminus Q$
- Для всех $v \in Q$: если $v.parent \neq NULL$, то $v.key < \infty$ и $v.key$ равен весу легкого ребра $(v, v.parent)$, соединяющего $v$ с некоторой вершиной, уже находящейся в МОД
- Реализация ПО на основе двоичной кучи: строки 1–3 — $O(|V|)$, строка 5 — $O(|V| \log |V|)$. Число итераций *for* — $O(|E|)$. Проверка $v \in Q$ с помощью битовой маски — $O(1)$. Обновление ключей — $O(\log |V|)$. Итого $O(|V| \log |V| + |E| \log |V|)$

# Фибоначчиева куча

- Цель: обеспечить сложность операций, не связанных с удалением элементов, $O(1)$
- Операции *Insert*, *Minimum*, *ExtractMin*, *Union*
- Если над элементами ФК не выполнять *DecreaseKey* и *Delete*, структура аналогична двоичной куче
- Поддержка строгой структуры ФК откладывается до момента, когда это будет удобным

# Структура фибоначчиевой кучи

- ФК — набор деревьев
- Каждый узел дерева содержит указатель на родителя и на один из дочерних узлов
- Дочерние узлы объединены в циклический двусвязный список (список дочерних узлов). Порядок узлов в списке произволен
- Удаление элементов из списка и объединение списков имеют сложность $O(1)$
- $x.degree$ — количество дочерних узлов
- $x.mark = true$, если $x$ терял дочерние узлы начиная с момента, когда $x$ стал дочерним узлом другого узла. Вновь созданные узлы не помечены, $x.mark$ снимается, если узел становится дочерним
- $Q.min$ — корень дерева с минимальным ключом. Если куча пуста, $Q.min = NULL$
- Корни всех деревьев образуют циклический двусвязный список (список корней)
- $Q.n$ — число узлов в ФК

# Биномиальные деревья

- Биномиальное дерево $B_0$ состоит из единственного узла
- Биномиальное дерево $B_k$ состоит из двух биномиальных деревьев $B_{k-1}$, причем корень одного из них является крайним левым дочерним узлом корня второго дерева
- Неупорядоченное биномиальное дерево $B_k$ состоит из двух биномиальных деревьев $B_{k-1}$, причем корень одного из них является *произвольным* дочерним узлом корня второго дерева
- ФК — набор неупорядоченных биномиальных деревьев

# Вставка узла

**Algorithm 10:** FibHeapInsert(Q,x)

```
1 x.degree=0;x.p=NULL;x.child=NULL;x.left=x;x.right=x;x.mark=false
2 Присоединить x к списку корней Q
3 if Q.min = NULL ∨ x.key < Q.min.key then
4 |   Q.min=x
5 Q.n=Q.n+1
```

Сложность $O(1)$

# Объединение двух ФК

**Algorithm 11:** FibHeapUnion($Q_1, Q_2$)

```
1 Q=MakeFibHeap();Q.min=Q_1.min
2 Добавить список корней Q_2 к списку корней Q
3 if Q_1.min = NULL ∨ Q_2.min ≠ NULL ∧ Q_2.min.key < Q_1.min.key then
4     Q.min = Q_2.min
5 Q.n = Q_1.n + Q_2.n
6 delete Q_1;delete Q_2
7 return Q
```

Сложность $O(1)$

# Извлечение минимального узла

**Algorithm 12:** FibExtractMin($Q$)

```
1  z=Q.min
2  if z ≠ NULL then
3      for каждого дочернего по отношению к z
         узлу x do
4          добавить x в список корней
             Q;x.p=NULL
5      Удалить z из списка корней Q
6      if z.right=z then
7          Q.min=NULL
8      else
9          Q.min=z.right
10         Consolidate(Q)
11     Q.n=Q.n-1
12 return z
```

Consolidate: многократное исполнение следующих шагов, пока все корни не будут иметь различные поля degree

1. Найти в списке два корня $x, y$ с одинаковой степенью, $x.key \leq y.key$
2. Удалить $y$ из списка корней, сделав его дочерним узлом $x$. $x.degree$ при этом увеличивается

# Уменьшение ключа

**Algorithm 13:** FibHeapDecreaseKey(Q,x,k)

```
1 if k>x.key then
2     Error: Новый ключ больше старого
3 x.key=k;y=x.parent
4 if y ≠ NULL ∧ x.key < y.key then
5     Cut(Q,x,y)
6     CascadingCut(Q,y)
7 if x.key<Q.min.key then
8     Q.min=x
```

Сложность $O(1)$

**Algorithm 14:** Cut(Q,x,y)

```
1 Удалить x из дочернего списка y, уменьшив
    y.degree
2 Добавить x в список корней Q
3 x.p=NULL; x.mark=false
```

**Algorithm 15:** CascadingCut(Q,y)

```
1 z=y.parent
2 if z ≠ NULL then
3     if y.mark=false then
4         y.mark=true
5     else
6         Cut(Q,y,z)
7         CascadingCut(Q.z)
```

# Заключение

- Матрицы достижимости и расстояний могут быть найдены единообразно из итерации матриц смежности/дуг в соответствующем полукольце
- Алгоритма Крускала и Прима нахождания минимального остовного дерева