Билеты Теория Графов - 4

Тимур Адиатуллин | telegram, github

Содержание

1 Специальные бинарные отношения. Связь между понятием отношения и понятием графа. Орграфы и бинарные отношения.

Бинарные отношения

Пусть A — непустое множество. **Бинарным отношением** на A называется любое подмножество

$$R \subseteq A \times B.$$

Пару $(a, b) \in R$ принято обозначать как $a\,R\,b$.

Специальные свойства бинарных отношений

- **Рефлексивность:**

$$\forall a \in A:\ (a, a) \in R.$$

- **Антирефлексивность:**

$$\forall a \in A:\ (a, a) \notin R.$$

- **Симметричность:**

$$\forall a, b \in A:\ (a, b) \in R \Rightarrow (b, a) \in R.$$

- **Антисимметричность:**

$$(a, b) \in R \text{ и } (b, a) \in R \Rightarrow a = b.$$

- **Транзитивность:**

$$(a, b) \in R \text{ и } (b, c) \in R \Rightarrow (a, c) \in R.$$

Основное определение

Графом $G(V, E)$ называется совокупность двух множеств — непустого множества V (множества вершин) и множества E двухэлементных подмножеств множества V (E — множество рёбер),

$$G(V, E) \stackrel{\text{def}}{=} \langle V; E\rangle, \quad V \neq \emptyset, \quad E \subseteq 2^V \ \&\ \forall e \in E\ (|e| = 2).$$

ЗАМЕЧАНИЕ. Легко видеть, что любое множество E двухэлементных подмножеств множества V определяет симметричное бинарное отношение на множестве V. Поэтому можно считать, что

$$E \subseteq V \times V, \quad E = E^{-1}$$

и трактовать ребро не только как множество $\{v_1, v_2\}$, но и как пару (v_1, v_2).

Число вершин графа G обозначим p, а число рёбер — q:

$$p \stackrel{\text{def}}{=} p(G) \stackrel{\text{def}}{=} |V|, \quad q \stackrel{\text{def}}{=} q(G) \stackrel{\text{def}}{=} |E|.$$

Если хотят явно упомянуть числовые характеристики графа, то говорят: (p, q)-граф.

Ориентированный граф (орграф)

Если элементами множества E являются упорядоченные пары (т. е. $E \subseteq V \times V$), то граф называется **ориентированным** (или **орграфом**). В этом случае элементы множества V называются **узлами**, а элементы множества E — **дугами**.

Орграфы и свойства отношений

Свойства отношения естественно интерпретируются в терминах орграфов:

- **Рефлексивность** соответствует наличию петли в каждой вершине.
- **Антирефлексивность** означает отсутствие петель.
- **Симметричность** означает, что каждая дуга имеет дугу в обратном направлении.
- **Антисимметричность** означает отсутствие пар противоположных дуг между различными вершинами.
- **Транзитивность** означает: если есть дуги $a \to b$ и $b \to c$, то должна быть дуга $a \to c$.

Соответствие графов и отношений

- Полный граф — универсальное отношение.
- Неорграф — симметричное отношение.
- Дополнение графов — дополнение отношений.
- Изменение всех направлений дуг — обратное отношение.

Итог

Бинарное отношение на множестве A полностью эквивалентно ориентированному графу на том же множестве вершин. Свойства отношения имеют естественные графовые интерпретации, что делает орграфы удобным инструментом для визуализации и анализа отношений.

2 Определение графа. Смежность. Диаграммы. Псевдо-, гипер- и мультиграфы. Виды графов. Связность

Основное определение

Графом $G(V, E)$ называется совокупность двух множеств — непустого множества V (множества вершин) и множества E двухэлементных подмножеств множества V (E — множество рёбер),

$$G(V, E) \stackrel{\text{def}}{=} \langle V; E\rangle, \quad V \neq \emptyset, \quad E \subseteq 2^V \ \& \ \forall e \in E \ (|e| = 2).$$

Смежность

Пусть v_1, v_2 — вершины, $e = (v_1, v_2)$ — соединяющее их ребро. Тогда вершина v_1 и ребро e

7.1.3. Смежность

Пусть v_1, v_2 — вершины, $e = (v_1, v_2)$ — соединяющее их ребро. Тогда вершина v_1 и ребро e **инцидентны**, ребро e и вершина v_2 также инцидентны. Два ребра, инцидентные одной вершине, называются смежными; две вершины, инцидентные одному ребру, также называются смежными.

Множество вершин, смежных с вершиной v, называется **множеством смежности** (или **окрестностью**) вершины v и обозначается $\Gamma^+(v)$:

$$\Gamma^+(v) \stackrel{\text{Def}}{=} \{u \in V \mid (u, v) \in E\}, \quad \Gamma^*(v) \stackrel{\text{Def}}{=} \Gamma^+(v) + v.$$

ЗАМЕЧАНИЕ. Если не оговорено противное, то символ Γ без индекса подразумевает Γ^+, то есть саму вершину в окрестность не включают.

Очевидно, что $u \in \Gamma(v) \iff v \in \Gamma(u)$. Если $A \subset V$ — множество вершин, то $\Gamma(A)$ — множество всех вершин, смежных с вершинами из A:

$$\Gamma(A) \stackrel{\text{Def}}{=} \{u \in V \mid \exists\, v \in A \ (u \in \Gamma(v))\} = \bigcup_{v \in A} \Gamma(v).$$

Диаграммы

Обычно граф изображают диаграммой: вершины — точками (или кружками), рёбра — линиями.

Пример. На рис. 7.4 приведён пример диаграммы графа, имеющего четыре вершины и пять рёбер. В этом графе вершины v_1 и v_2, v_2 и v_3, v_3 и v_4, v_4 и v_1, v_2 и v_4 смежны, а вершины v_1 и v_3 не смежны. Смежные рёбра: e_1 и e_2, e_2 и e_3, e_3 и e_4, e_4 и e_1, e_1 и e_5, e_2 и e_5, e_3 и e_5, e_4 и e_5. Несмежные рёбра: e_1 и e_3, e_2 и e_4.

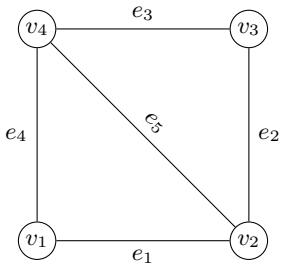

Рис. 1: Диаграмма графа

Типы графов

1. **Псевдограф** — граф с петлями.
2. **Мультиграф** — граф с кратными рёбрами.
3. **Гиперграф** — дуги являются множествами с одним и более элементами.
4. **Нумерованный граф** — если существует функция, отображающая множество вершин или рёбер в множество чисел или символов.

- Если элементом множества E может быть пара одинаковых (не различных) элементов V, то такой элемент называется **петлёй**, а граф — **графом с петлями** (или **псевдографом**).
- Если E является не множеством, а мультимножеством, содержащим некоторые элементы по несколько раз, то такие элементы называются **кратными рёбрами**, а граф — **мультиграфом**.

- Если элементами множества E являются не обязательно двузначные, а любые (непустые) подмножества множества V, то такие элементы называются **гипердугами**, а граф — **гиперграфом**.
- Если задана функция $F : V \to M$ и/или $F : E \to M$, то множество M называется **множеством пометок**, а граф — **помеченным** (или **нагруженным**). В качестве множества пометок обычно используются буквы или целые числа. Если функция F инъективна, то есть разные вершины (рёбра) имеют разные пометки, то граф называют **нумерованным**.

Специальные графы

- **Тривиальный граф** — состоит из одной вершины.
- **Циклический граф с k вершинами** — обозначается C_k.
- **Полный граф** — содержит все возможные рёбра между вершинами, обозначается K_p.

Число рёбер в полном графе:

$$q(K_p) = \frac{p(p-1)}{2}.$$

Колёса и двудольные графы

- **Колесо** — граф, обозначаемый W_n.
- **Двудольный граф** — граф, множество вершин V которого можно разбить на два непересекающихся множества V_1 и V_2 так, что каждое ребро из E инцидентно вершинам из V_1 и V_2. Множества V_1 и V_2 называются **долями графа**.

Связность графа

Говорят, что две вершины в графе **связаны**, если существует соединяющая их (простая) цепь. Граф, в котором все вершины связны, называется **связным**.

Связность является **эквивалентностью**. Классы эквивалентности по отношению связности — это **компоненты связности** графа:

$$k(G).$$

Граф, состоящий только из вершин, называется **вполне несвязным**.

3 Понятие изоморфизма. Изоморфизм графов (+2 теоремы). Инварианты графа.

Гомоморфизм

Пусть $\mathcal{A} = \langle A; \varphi_1, \ldots, \varphi_m \rangle$ и $\mathcal{B} = \langle B; \psi_1, \ldots, \psi_m \rangle$ — две алгебры одного типа (одинаковые векторы арностей). Если существует функция $f : A \to B$, такая, что

$$\forall i \in \{1, \ldots, m\} : \quad f(\varphi_i(a_1, \ldots, a_n)) = \psi_i(f(a_1), \ldots, f(a_n)),$$

то говорят, что f — **гомоморфизм** из $\mathcal{A}$ в $\mathcal{B}$.

Действие гомоморфизма можно изобразить с помощью диаграммы:

$$\begin{array}{ccc} A & \longrightarrow & A \\ \downarrow & & \downarrow \\ B & \longrightarrow & B \end{array}$$

Рис. 2: Коммутативная диаграмма: $f \circ \varphi = \psi \circ f$

Диаграмма называется **коммутативной**, потому что условие гомоморфизма можно переписать с помощью суперпозиции функций:

$$f \circ \varphi = \psi \circ f.$$

Изоморфизм

Пусть $\mathcal{A} = \langle A; \varphi_1, \ldots, \varphi_m \rangle$ и $\mathcal{B} = \langle B; \psi_1, \ldots, \psi_m \rangle$ — две алгебры одного типа, и $f : A \to B$ — изоморфизм или гоморфизм с биекцией. Тогда алгебры $\mathcal{A}$ и $\mathcal{B}$ изоморфны:

$$\mathcal{A}^f \sim \mathcal{B}.$$

Теорема 1. Если $f : A \to B$ — изоморфизм, то $f^{-1} : B \to A$ тоже является изоморфизмом.

Теорема 2. Отношение изоморфизма на множестве однотипных алгебр является эквивалентностью.

Изоморфизм графов

Говорят, что два графа $G_1(V_1, E_1)$ и $G_2(V_2, E_2)$ **изоморфны** (обозначается $G_1 \sim G_2$ или $G_1 = G_2$), если существует биекция $h : V_1 \to V_2$, сохраняющая смежность:

$$e_1 = (u, v) \in E_1 \iff e_2 = (h(u), h(v)) \in E_2.$$

Теорема

Изоморфизм графов есть отношение эквивалентности

Теорема 1.

Графы изоморфны тогда и только тогда, когда их матрицы смежности вершин получаются друг из друга одновременными перестановками строк и столбцов.

Теорема 2.

Графы (орграфы) изоморфны тогда и только тогда, когда их матрицы инцидентности получаются друг из друга произвольными перестановками строк и столбцов.

Инварианты

Ну найдете сами

4 Элементы графов: подграфы, валентность, маршруты, цепи, циклы. Метрические характеристики графа. Особенности алгоритмов теории графов

Подграфы

Граф $G'(V', E')$ называется **подграфом** (или **частью**) графа $G(V, E)$ (обозначается $G' \subseteq G$), если

$$V' \subseteq V \quad \& \quad E' \subseteq E.$$

Если $V' = V$, то G' называется **остовным подграфом** графа G.

Если $V' \subset V$, $E' \subset E$ и $(V' \neq V \vee E' \neq E)$, то граф G' называется **собственным подграфом** графа G.

Подграф $G'(V', E')$ называется **правильным подграфом** графа $G(V, E)$, если он содержит все возможные рёбра графа G между вершинами из V':

$$\forall\, u, v \in V' \quad ((u, v) \in E \Rightarrow (u, v) \in E').$$

Правильный подграф $G'(V', E')$ графа $G(V, E)$ определяется подмножеством вершин V'.

Замечание.

Иногда подграфами называют только правильные подграфы, а неправильные подграфы называют **изграфами**.

Степень вершины

Количество рёбер, инцидентных вершине v, называется **степенью** (или **валентностью**) вершины v и обозначается $d(v)$:

$$\forall v \in V \quad 0 \leq d(v) \leq p - 1, \qquad d(v) = |\Gamma^+(v)|.$$

Таким образом, степень $d(v)$ вершины v совпадает с количеством смежных с ней вершин. Количество вершин, не смежных с v, обозначается $\overline{d}(v)$. Ясно, что:

$$\forall v \in V \quad d(v) + \overline{d}(v) = p - 1.$$

Обозначим минимальную степень вершины графа G через $\delta(G)$, а максимальную — через $\Delta(G)$:

$$\delta(G(V, E)) \stackrel{\text{def}}{=} \min_{v \in V} d(v), \qquad \Delta(G(V, E)) \stackrel{\text{def}}{=} \max_{v \in V} d(v).$$

Очевидно, что $\delta(G)$ и $\Delta(G)$ являются **инвариантами** графа.

Если степени всех вершин равны k, то граф называется **регулярным степени** k:

$$\delta(G) = \Delta(G) = k, \qquad \forall v \in V \quad d(v) = k.$$

Степень регулярности обозначается $r(G)$. Для нерегулярных графов $r(G)$ не определено.

Маршруты, цепи, циклы

Маршрутом в графе называется чередующаяся последовательность вершин и рёбер, начинающаяся и заканчивающаяся вершиной:

$$v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k,$$

в которой любые два соседних элемента инцидентны, причём однородные элементы (вершины, рёбра) через один — смежны или совпадают.

Маршруты, цепи, циклы

Если $v_0 = v_k$, то маршрут называется **замкнутым**, иначе — **открытым**.

Если все рёбра различны, то маршрут называется **цепью**. Если все вершины (а значит, и рёбра) различны, то маршрут называется **простой цепью**.

В цепи $v_0, e_1, \ldots, e_k, v_k$ вершины v_0 и v_k называются **концами цепи**. Говорят, что цепь с концами u и v **соединяет вершины** u и v.

Цепь, соединяющая вершины u и v, обозначается $\langle u, v \rangle$. Если нужно указать граф G, которому принадлежит цепь, то добавляют индекс: $\langle u, v \rangle_G$.

Нетрудно показать, что если существует какая-либо цепь, соединяющая вершины u и v, то существует и **простая цепь**, соединяющая эти вершины.

Замкнутая цепь называется **циклом**; **замкнутая простая цепь** называется **простым циклом**.

Число циклов в графе G обозначается $z(G)$. Граф без циклов называется **ациклическим**.

Для орграфов **цепь** называется **путём**, а **цикл — контуром**.

Путь в орграфе из узла u в узел v обозначается:

$$\langle \vec{u}, v \rangle.$$

Метрические характеристики графа

Длина маршрута — количество рёбер в нём. Маршрут M имеет длину k тогда и только тогда, когда $|M| = k$.

Расстоянием между вершинами u и v, обозначаемым $d(u, v)$, называется длина кратчайшей цепи $\langle u, v \rangle$, а сама кратчайшая цепь называется **геодезической**:

$$d(u, v) = \min_{\{\langle u,v \rangle\}} |\langle u, v \rangle|.$$

Если цепи нет, расстояние считается бесконечным.

Ярус — множество вершин на расстоянии n от вершины v.

Диаметр графа — длина самой длинной геодезической цепи:

$$D(G) = \max_{u,v \in V} d(u, v).$$

Эксцентриситет $e(v)$ вершины v в связном графе — максимальное расстояние от v до других вершин.

Радиус графа $R(G)$ — наименьший эксцентриситет:

$$R(G) = \min_{v \in V} e(v).$$

Вершина называется **центральной**, если её эксцентриситет совпадает с радиусом. Множество центральных вершин называется **центром графа**.

5 Способы задания графа. Представление графов в ЭВМ. Обходы графов.

Представление графов в программах

Следует ещё раз подчеркнуть, что конструирование структур данных для представления в программе объектов математической модели — это основа искусства практического программирования.

Мы приводим четыре различных базовых представления графов. Выбор наилучшего представления определяется требованиями конкретной задачи. Более того, на практике используются, как правило, некоторые комбинации или модификации указанных представлений, общее число которых необозримо. Но все они так или иначе основаны на тех базовых идеях, которые описаны в этом разделе.

Требования к представлению графов

Известны различные способы представления графов в памяти компьютера, которые различаются объёмом занимаемой памяти и скоростью выполнения операций над графами. Представление выбирается, исходя из потребностей конкретной задачи.

Далее приведены четыре наиболее часто используемых представления с указанием характеристики $\eta(p, q)$ — объёма памяти для каждого представления, где p — число вершин, а q — число рёбер.

7.4.2. Матрица смежности

Представление графа с помощью квадратной булевой матрицы

$$M : \text{array } [1..p, 1..p] \text{ of } \{0, 1\},$$

отражающей смежность вершин, называется **матрицей смежности**, где

$$M[i, j] = \begin{cases} 1, & \text{если вершина } v_i \text{ смежна с вершиной } v_j, \\ 0, & \text{если вершины } v_i \text{ и } v_j \text{ не смежны.} \end{cases}$$

Для матрицы смежности объём памяти:

$$\eta(p, q) = \mathcal{O}(p^2).$$

Пример. Матрицы смежности графов G и D:

$$G = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix} \qquad D = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

Замечание. Матрица смежности графа симметрична относительно главной диагонали, поэтому достаточно хранить только верхнюю (или нижнюю) треугольную часть.

Матрица инцидентций

Представление графа с помощью матрицы

$$H : \text{array } [1..p, 1..q] \text{ of } \{0, 1\},$$

а для орграфов:

$$H : \text{array } [1..p, 1..q] \text{ of } \{-1, 0, 1\},$$

отражающей инцидентность вершин и рёбер, называется **матрицей инцидентций**, где для неориентированного графа:

$$H[i, j] = \begin{cases} 1, & \text{если вершина } v_i \text{ инцидентна ребру } e_j, \\ 0, & \text{в противном случае.} \end{cases}$$

А для ориентированного графа:

$$H[i, j] = \begin{cases} 1, & \text{если узел } v_i \text{ инцидентен дуге } e_j \text{ и является её концом,} \\ -1, & \text{если узел } v_i \text{ инцидентен дуге } e_j \text{ и является её началом,} \\ 0, & \text{если узел } v_i \text{ и дуга } e_j \text{ не инцидентны.} \end{cases}$$

Для матрицы инцидентций объём памяти:

$$\eta(p, q) = \mathcal{O}(pq).$$

Пример. Матрицы инцидентций графов G и D:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix} \qquad D = \begin{pmatrix} -1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 0 & -1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 & 1 \end{pmatrix}$$

Замечание. Для связных графов $q > p$, поэтому матрица смежности несколько компактнее матрицы инцидентций.

7.4.4. Списки смежности

Представление графа с помощью списочной структуры, отражающей смежность вершин и состоящей из массива указателей

$$G : \text{array } [1..p] \text{ of } \uparrow N,$$

на списки смежных вершин, где элемент списка описывается структурой:

$$N = \text{record } \{v : 1..p; \quad n :\uparrow N\} \text{ end record},$$

называется **списком смежности**.

В случае представления неориентированных графов:

$$\eta(p, q) = \mathcal{O}(p + 2q),$$

а в случае ориентированных графов:

$$\eta(p, q) = \mathcal{O}(p + q).$$

Замечание. Массив G также можно представить списком.

Пример. Списки смежности для графа G (слева) и орграфа D (справа) представлены на рисунке ниже.

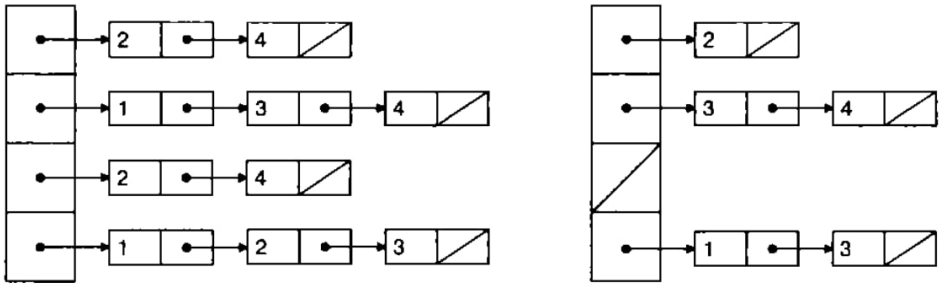

Рис. 3: Списки смежности для графа G и орграфа D

7.4.5. Массив дуг

Представление графа с помощью массива структур

$$E : \text{array } [1..q] \text{ of record } \{b, e : 1..p\} \text{ end record},$$

отражающего список пар смежных вершин (или, для орграфов, узлов), называется **массивом рёбер** (или **массивом дуг**).

Для массива рёбер (или дуг) объём памяти:

$$\eta(p, q) = \mathcal{O}(2q).$$

Замечание. Для представления графов с изолированными вершинами может понадобиться хранить также число p, если только система программирования не позволяет извлечь это число из массива структур E.

Пример. Представление с помощью массива рёбер (дуг) показано в следующей таблице: слева — для графа G, справа — для орграфа D.

Граф G		Орграф D	
b	e	b	e
1	2	1	2
1	4	2	3
2	3	2	4
2	4	4	1
3	4	4	3

Замечание. Указанные представления пригодны для графов и орграфов, а после некоторой модификации — также и для псевдографов, мультиграфов и гиперграфов.

Обходы графов

Обход графа — систематическое перечисление его вершин.

Теорема.

Если граф G связан и конечен, то поиск в ширину и поиск в глубину обходят все вершины по одному разу.

Суть методов:

- **Стек** — используется в поиске в глубину (DFS).
- **Очередь** — используется в поиске в ширину (BFS).

Алгоритм обхода графа:

1. Дан граф. Выбираем начальную вершину случайно или задаём вручную.
2. Создаём массив, где отмечаем пройденные вершины (изначально все равны 0).
3. Помещаем начальную вершину в структуру (стек или очередь) и отмечаем её.
4. Заходим в цикл:
   (a) Извлекаем вершину из структуры.
   (b) Просматриваем все смежные вершины:
      - Если вершина не отмечена — помещаем её в структуру и отмечаем.
   (c) Повторяем, пока структура не опустеет.

6 Операции над графами: локальные, алгебраические.

7.3.4. Операции над графами (продолжение)

4. **Удаление вершины** v из графа $G_1(V_1, E_1)$ (обозначение: $G_1(V_1, E_1) - v$, при условии $v \in V_1$) даёт граф $G_2(V_2, E_2)$, где

$$V_2 = V_1 \setminus \{v\}, \quad E_2 = E_1 \setminus \{e = (v_1, v_2) \mid v_1 = v \vee v_2 = v\}.$$

   Пример: $C_3 - v = K_2$.

5. **Удаление ребра** e из графа $G_1(V_1, E_1)$ (обозначение: $G_1(V_1, E_1) - e$, при условии $e \in E_1$) даёт граф $G_2(V_2, E_2)$, где

$$V_2 = V_1, \quad E_2 = E_1 \setminus \{e\}.$$

   Пример: $K_2 - e = K_2$.

6. **Добавление вершины** v в граф $G_1(V_1, E_1)$ (обозначение: $G_1(V_1, E_1) + v$, при условии $v \notin V_1$) даёт граф $G_2(V_2, E_2)$, где

$$V_2 = V_1 \cup \{v\}, \quad E_2 = E_1.$$

   Пример: $K_2 + v = K_2 \cup K_1$.

7. **Добавление ребра** e в граф $G_1(V_1, E_1)$ (обозначение: $G_1(V_1, E_1) + e$, при условии $e \notin E_1$) даёт граф $G_2(V_2, E_2)$, где

$$V_2 = V_1, \quad E_2 = E_1 \cup \{e\}.$$

8. **Стягивание (правильного) подграфа** A графа $G_1(V_1, E_1)$ (обозначение: $G_1(V_1, E_1)/A$, при условии $A \subset V_1, v \notin V_1$) даёт граф $G_2(V_2, E_2)$, где

$$V_2 = (V_1 \setminus A) \cup \{v\},$$

$$E_2 = E_1 \setminus \{e = (u, w) \mid u \in A \vee w \in A\} \cup \{e = (u, v) \mid u \in \Gamma(A) \setminus A\}.$$

   Пример: $K_4/C_3 = K_2$.

9. **Размножение вершины** v графа $G_1(V_1, E_1)$ (обозначение: $G_1(V_1, E_1) \uparrow v$, при условии $v \in V_1, v' \notin V_1$) даёт граф $G_2(V_2, E_2)$, где

$$V_2 = V_1 \cup \{v'\},$$

$$E_2 = E_1 \cup \{(v, v')\} \cup \{e = (u, v') \mid u \in \Gamma^+(v)\}.$$

   Пример: $K_2 \uparrow v = C_3$.

7 Упорядочение дуг и вершин орграфа. Алгоритм Фалкерсона.

Упорядочивание дуг и вершин графа

Под **упорядочиванием ациклического орграфа** понимается такое разбиение его вершин на группы, при котором:

1. Вершины первой группы не имеют предшествующих, а последней — последующих.
2. Вершины любой другой группы не имеют предшествующих в следующей группе.
3. Вершины одной и той же группы дугами не соединяются.

Такое разбиение всегда возможно.

Алгоритм Фалкерсона

1. Находим истоки — они образуют первую группу, нумеруем их в произвольном порядке.
2. Вычёркиваем все пронумерованные вершины и дуги.
3. Повторяем первый шаг для оставшегося графа — новая группа, новая нумерация.
4. Продолжаем, пока не будут упорядочены все вершины.

Аналогично можно упорядочить и дуги.

Матричный способ

1. Берём матрицу смежности.
2. Находим столбцы, состоящие из нулей — это первая группа.
3. Вычёркиваем найденные столбцы и соответствующие строки.
4. Повторяем, пока не упорядочим все вершины.

8 Выявление маршрутов с заданным количеством ребер. Определение экстремальных путей на графах. Метод Шимбелла. Волновые алгоритмы.

9 Связность: компоненты связности, точки сочленения. Вершинная и реберная связность (мосты и блоки, меры связности).

Связность: компоненты связности, точки сочленения

Теорема

Граф связан тогда и только тогда, когда его нельзя представить в виде объединения двух графов.

Компоненты связности

Классы эквивалентности по отношению связности называются **компонентами связности** графа. Число компонент связности обозначается $k(\mathcal{G})$.

Точка сочленения

Вершина графа называется **точкой сочленения**, если её удаление увеличивает число компонент связности.

Мост

Мостом называется ребро, удаление которого увеличивает число компонент связности.

Замечание В любом нетривиальном графе существует по крайней мере две вершины, которые не являются точками сочленения.

Вершинная и реберная связность (мосты и блоки, меры связности)

Теорема 1

Пусть $G(V, E)$ — связный граф и $v \in V$. Тогда следующие утверждения эквивалентны:

1. v — точка сочленения.
2. $\exists\, u, w \in V$ такие, что $u \neq w$ и $v \in \langle u, w\rangle_G$.
3. $\exists\, U, W \subseteq V \setminus \{v\}$ такие, что $U \cap W = \emptyset$, $U \cup W = V \setminus \{v\}$ и для всех $u \in U$, $w \in W$ любые пути $\langle u, w\rangle_G$ проходят через v.

Следствие

Если вершина инцидентна мосту и не является висячей, то она является точкой сочленения.

Теорема 2

Пусть $G(V, E)$ — связный граф и $x \in E$. Тогда следующие утверждения эквивалентны:

1. x — мост.
2. x не принадлежит ни одному простому циклу.
3. $\exists\, u, w \in V$ такие, что все пути $\langle u, w\rangle_G$ содержат x.
4. $\exists\, U, W \subseteq V$ такие, что $U \cap W = \emptyset$, $U \cup W = V$ и для всех $u \in U$, $w \in W$ любые пути $\langle u, w\rangle_G$ содержат x.

Вершинная связность

Вершинной связностью графа называется наименьшее число вершин, удаление которых приводит к несвязному или тривиальному графу. Обозначается $\chi(G)$.

Рёберная связность

Рёберной связностью графа называется наименьшее число рёбер, удаление которых приводит к несвязному или тривиальному графу. Обозначается $\lambda(G)$.

10 Теорема Менгера (с доказательством). Непересекающиеся цепи и разделяющие множества. Варианты теоремы Менгера. Теорема Холла.

Теорема Менгера

Пусть u и v — несмежные вершины в графе G. Наименьшее число вершин в множестве, разделяющем u и v, равно наибольшему числу вершинно непересекающихся простых цепей $\langle u, v\rangle$:

$$\max |P(u,v)| = \min |S(u,v)|$$

где:

- $P(u,v)$ — множество вершинно непересекающихся цепей $\langle u, v\rangle$;
- $S(u,v)$ — разделяющее множество вершин (удаление которых приводит к разбиению графа так, что u и v оказываются в разных компонентах связности).

я ебал это доказывать

Разрез

Разделяющее множество рёбер называется **разрезом**.

Варианты теоремы Менгера

Теорема

Для любых двух несмежных вершин u и v графа G наибольшее число рёберно-непересекающихся цепей $\langle u, v\rangle$ равно наименьшему числу рёбер в $\langle u, v\rangle$-разрезе.

Теорема

Граф G является n-связным тогда и только тогда, когда любые две несмежные вершины соединены не менее чем n вершинами, образующими вершинно-непересекающиеся простые цепи.

Теорема Холла

Паросочетание

Паросочетанием (или независимым множеством рёбер) называется множество рёбер, никакие два из которых не смежны. Паросочетание называется **максимальным**, если никакое его надмножество не является независимым.

Пусть $G(V_1, V_2, E)$ — двудольный граф. **Совершенным паросочетанием** из V_1 в V_2 называется паросочетание, покрывающее все вершины множества V_1.

Теорема Холла

Совершенное паросочетание существует тогда и только тогда, когда

$$\forall A \subseteq V_1 \quad (|A| \leq |\Gamma(A)|),$$

11 Нахождение кратчайших путей: алгоритм Дейкстры, алгоритм Беллмана-Форда.

Алгоритм Дейкстры

Алгоритм Дейкстры используется для поиска кратчайших путей от заданной вершины-истока s до всех остальных вершин графа.

Сложность

$$\mathcal{O}(p^2)$$

Результат

Алгоритм возвращает вектор расстояний $d[v]$ от истока s до каждой вершины $v \in V$.

Шаги алгоритма

1. **Инициализация:**
   - Для всех вершин $v \in V$ задать $d[v] := \infty$.
   - Для истока s задать $d[s] := 0$.
   - Создать множество необработанных вершин $Q := V$.
2. **Основной цикл:** Пока $Q \neq \emptyset$:
   (a) Выбрать вершину $u \in Q$ с минимальным значением $d[u]$.
   (b) Удалить u из Q.
   (c) Для каждого соседа v вершины u, где $v \in Q$:
      - Вычислить новое расстояние: temp $:= d[u] + w(u, v)$.
      - Если temp $< d[v]$, то обновить: $d[v] :=$ temp.

Алгоритм Беллмана–Форда

Алгоритм Беллмана–Форда используется для поиска кратчайших путей от заданной вершины-истока s до всех остальных вершин графа, допускающего отрицательные веса рёбер.

Сложность

$$\mathcal{O}(pq)$$

Результат

Алгоритм возвращает вектор расстояний $d[v]$ от истока s до каждой вершины $v \in V$.

Инициализация

- Для всех вершин $v \in V$ задать $d[v] := \infty$.
- Для истока s задать $d[s] := 0$.

Основной алгоритм

1. Повторить следующие шаги $|V| - 1$ раз:
   - Для каждого ребра $(u, v) \in E$:
     - Вычислить новое расстояние: temp $:= d[u] + w(u, v)$.
     - Если temp $< d[v]$, то обновить: $d[v] :=$ temp.

Проверка на наличие отрицательных циклов

- Для каждого ребра $(u, v) \in E$:
  - Если $d[u] + w(u, v) < d[v]$, то в графе существует отрицательный цикл.

12 Нахождение кратчайших путей: алгоритм Флойда-Уоршалла. Алгоритм нахождения максимального пути.

Алгоритм Флойда–Уоршалла

Алгоритм Флойда–Уоршалла используется для нахождения кратчайших расстояний между всеми парами вершин во взвешенном графе (возможно с отрицательными весами, но без отрицательных циклов).

Сложность

$$\mathcal{O}(p^3)$$

Результат

Алгоритм возвращает матрицу расстояний размером $p \times p$.

Инициализация

- Для всех пар вершин (i, j) задать $d[i][j] := \infty$.
- Для всех i задать $d[i][i] := 0$.
- Для каждого ребра (u, v) с весом w задать $d[u][v] := w$.

Основной алгоритм

- Для каждой вершины k от 1 до n:
  - Для каждой пары вершин (i, j):
    - Вычислить: temp $:= d[i][k] + d[k][j]$.
    - Если temp $< d[i][j]$, то обновить: $d[i][j] :=$ temp.

Проверка на отрицательные циклы (опционально)

Если после выполнения алгоритма существует i такое, что $d[i][i] < 0$, то в графе присутствует отрицательный цикл.

Алгоритм нахождения максимального пути

Для нахождения максимального пути в графе можно использовать полный перебор всех возможных путей от текущей вершины ко всем достижимым из неё вершинам.

Идея

Перебрать все возможные пути от текущей вершины до всех последующих, достижимых из неё, и выбрать путь с максимальной суммарной длиной (весом).

Применение

Подходит для ориентированных ациклических графов (DAG), где отсутствуют циклы, и можно использовать динамическое программирование или топологическую сортировку для оптимизации.

13 Потоки в сетях: определение потока, разрезы. Теорема Форда и Фалкерсона. Алгоритм Форда-Фалкерсона. Коммуникационные сети.

Сеть

Пусть $G(V, E)$ — орграф с одним источником $s \in V$ и одним стоком $t \in V$, где $s \neq t$. **Сетью** $S = (G; c)$ называется орграф G с заданной функцией $c : E \to R_+$, сопоставляющей каждой дуге $e \in E$ неотрицательное действительное число $c(e)$, называемое **пропускной способностью**.

Поток

Потоком в сети $S = (G; c)$, где $G(V, E)$, называется функция $f : E \to R_+$, приписывающая каждой дуге e неотрицательное число $f(e)$ — **поток по дуге** e, при выполнении следующих условий:

1. $f(e) \leq c(e)$ — поток не превышает пропускную способность;
2. Для любой вершины $v \in V \setminus \{s, t\}$ сумма входящих потоков равна сумме исходящих:

$$\sum_{(u,v)\in E} f(u, v) = \sum_{(v,w)\in E} f(v, w).$$

Разрезы

Пусть $S = (G; c)$ — сеть, где $G = (V, E)$. Если $X \subset V$, $s \in X$, $t \notin X$, и $Y = V \setminus X$, то множество

$$R = \{e \in E \mid e = (v, w),\ v \in X,\ w \in Y\}$$

называется **разрезом** сети S и обозначается $R = (X, Y)$.

Пропускная способность разреза

Пропускной способностью разреза $R(X, Y)$ называется неотрицательное число

$$c(R) = \sum_{e\in R} c(e).$$

Теорема Форда–Фалкерсона

Формулировка

Пусть $S = (G; c)$ — сеть, где $G = (V, E)$. Величина максимального потока $p_{\max}$ в сети S совпадает с минимальной пропускной способностью $r_{\min}$ её разрезов:

$$p_{\max} = r_{\min}.$$

Алгоритм Форда–Фалкерсона

Трудоёмкость

$$\mathcal{O}(q \cdot p_{\max})$$

Шаги алгоритма

1. **Инициализация:**
   - Для всех рёбер (u, v) установить $f(u, v) := 0$.
   - Задать исток s и сток t.
2. **Поиск увеличивающего пути:** Пока существует путь p из s в t в остаточной сети (где $c_f(u, v) > 0$):
   (a) Найти минимальную остаточную пропускную способность:

$$c_f(p) := \min\{c_f(u, v) \mid (u, v) \in p\}.$$

   (b) Для каждого ребра $(u, v) \in p$:

- Увеличить поток: $f(u, v) := f(u, v) + c_f(p)$.
- Уменьшить обратный поток: $f(v, u) := f(v, u) - c_f(p)$.

3. **Завершение:** Когда увеличивающих путей больше нет, максимальный поток равен сумме потоков из s в смежные вершины (или в t).

Коммуникационные сети

- Моделью компьютерной сети может служить ориентированный граф, чьи узлы представляют компьютерные компоненты, а дуги — коммуникационные линии связи. Каждая дуга снабжена весом, обозначающим пропускную способность этой линии.
- Процедура *статической маршрутизации* учитывает информацию о пропускной способности линии для определения фиксированного пути передачи между узлами. В целях оптимизации таких путей применяют алгоритм, близкий к алгоритму Дейкстры. Однако задержки могут возникать при сбоях и превышении пропускной способности сети.
- Процедура *динамической маршрутизации* постоянно корректирует пропускную способность линий с учётом текущих потребностей. Набор правил или протокол позволяет узлам решать, когда и куда передавать новую информацию.
- Каждый узел поддерживает свою таблицу путей — задача оптимизации рассредоточена по всей сети.

14 Эвристические алгоритмы. Алгоритм A*. Метод ветвей и границ

Эвристика

Эвристикой называется алгоритм решения задачи, включающий практический метод, не являющийся гарантированно точным или оптимальным, но достаточный для получения решения и позволяющий ускорить процесс вычислений.

- Не гарантирует лучшее решение.
- Не гарантирует наличие решения.
- Может дать неверное решение.

Алгоритм A*

Порядок обхода вершин определяется *эвристической функцией* $f(x)$, которая представляет собой сумму двух компонент:

$$f(x) = g(x) + h(x),$$

где:

- $g(x)$ — функция стоимости достижения вершины x из начальной вершины (может быть эвристической или точной);
- $h(x)$ — эвристическая оценка расстояния от вершины x до целевой вершины.

Функция $h(x)$ должна быть *допустимой эвристикой*, то есть не должна переоценивать расстояние до цели. Например, в задаче маршрутизации $h(x)$ может представлять собой расстояние до цели по прямой линии.

Интуиция

На каждом шаге алгоритм оценивает, насколько текущая вершина приближает к цели, используя функцию $f(x)$, и выбирает путь, минимизирующий эту оценку.

Ну короче типо мы смотрим на каждом шаге насколько мы вообще приблизились к цели или отдалились с помощью какой-либо эвристической функции и это учитываем при выборе пути

Метод ветвей и границ

Метод ветвей и границ является развитием метода полного перебора, но с отсевом подмножеств, заведомо не содержащих оптимальных решений.

Идея

На каждом шаге элементы разбиения анализируются: содержит ли подмножество оптимальное решение или нет. Если решается задача минимизации, то проверка осуществляется сравнением нижней оценки значения целевой функции с верхней оценкой функционала.

Рекорд

Допустимое решение, дающее наименьшую верхнюю оценку, называется *рекордом*. Если нижняя оценка целевой функции на данном подмножестве не меньше текущего рекорда, то это подмножество не содержит лучшего решения и может быть отброшено. Если значение целевой функции меньше рекорда, рекорд обновляется.

Завершение

Если все элементы разбиения просмотрены, алгоритм завершает работу, и текущий рекорд является оптимальным решением. Если нет — выбирается перспективное множество, которое подвергается дальнейшему разбиению. Процесс продолжается, пока не будут просмотрены все элементы.

15 Поток минимальной стоимости. Алгоритм определения потока минимальной стоимости

я ебал

16 Транспортная задача. Алгоритмы Диница и Кинга. Задачи многокритериальной оптимизации.

Транспортная задача

Пусть имеется m пунктов производства однородного продукта: $A_1, A_2, \ldots, A_m$ и n пунктов его потребления: $B_1, B_2, \ldots, B_n$. В каждом пункте A_i производится a_i единиц продукта, а в каждом пункте B_j потребляется b_j единиц. Необходимо составить оптимальный план перевозок.

Математическая модель

Пусть x_{ij} — объём продукта, перевозимого из A_i в B_j. Тогда общая стоимость перевозок, которую требуется минимизировать:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \to \min$$

Задачи многокритериальной оптимизации

Пусть имеется несколько целевых функций, которые необходимо максимизировать. Основная сложность заключается в том, что максимум одного критерия может сопровождаться неудовлетворительными значениями других.

Методы сведения к однокритериальной задаче

- Выделение главного критерия.
- Линейная свёртка.
- Максминная свёртка.
- Метод идеальной точки.
- Оптимальность по Парето, по Слейтору, по Никоретте.

Метод последовательных уступок

1. Находим максимальное значение самого важного критерия путём решения однокритериальной задачи.
2. Назначаем допустимое отклонение для первого критерия.
3. В ограниченной области ищем наилучшее значение второго критерия.
4. Повторяем процедуру для всех критериев по убыванию важности.

Алгоритмы Диница и Кинга

Алгоритмы Диница и Кинга используются для поиска максимального потока в сети.

Алгоритм Диница

Сложность: $\mathcal{O}(p^2q)$

Инициализация

- Для всех рёбер (u, v) установить поток $f(u, v) := 0$.
- Построить остаточную сеть G_f с пропускными способностями $c_f(u, v) := c(u, v) - f(u, v)$.

Фаза построения слоистой сети (BFS)

Пока существует путь из s в t в остаточной сети:

- Построить слоистую сеть (level graph) обходом в ширину из s.
- Для каждой вершины v вычислить $level[v]$ — расстояние от s до v.
- Ребро (u, v) включается, если $level[v] = level[u] + 1$ и $c_f(u, v) > 0$.

Фаза блокирующего потока (DFS)

Пока в слоистой сети существует путь из s в t:

- Найти блокирующий поток с помощью обхода в глубину.
- Для каждого найденного пути p:
  - Найти минимальную остаточную пропускную способность:

$$c_f(p) := \min\{c_f(u,v) \mid (u,v) \in p\}.$$

  - Увеличить поток вдоль пути: $f(u,v) := f(u,v) + c_f(p)$.
  - Обновить обратные рёбра: $f(v,u) := f(v,u) - c_f(p)$.

Завершение

Если BFS не достигает t, алгоритм завершает работу. Максимальный поток равен сумме потоков из s в смежные вершины.

Алгоритм Кинга

Сложность: $\mathcal{O}(nm)$

Игра на двудольном графе

Рассмотрим неориентированный двудольный граф $G = (U, V, E)$, где $|U| = |V| = n$, $|E| = m$.

Правила игры

Инициализация: Алгоритм-игрок назначает дугу каждой вершине из множества U.
Ходы соперника:

- Удалить любую дугу из графа (*edge-kill*) — очки не начисляются.
- Удалить любую вершину из множества V вместе с инцидентными ей дугами — начисляется по одному очку за каждую удалённую дугу.

Ходы алгоритма-игрока:

- Назначить дугу вершине $u \in U$, если дуга ещё не назначена.
- Назначить новую инцидентную дугу вершине u — старая дуга теряет специальный статус, а соперник получает дополнительное очко.

Стратегия алгоритма-игрока

Пусть l — минимальный коэффициент (степень) вершин u. Обозначим $U' \subseteq U$ — множество вершин с коэффициентом больше l.

Для каждой вершины $v \in V$ определим оценку $r(v)$ — число назначенных дуг (u, v), где u имеет минимальный коэффициент.

Разделим диапазон значений $r(v)$ на уровни: Уровень i содержит вершины v, для которых $r(v) \in [r_i, r_{i+1})$, где $r_i = 2^i$.

При назначении дуги вершине $u \in U'$, алгоритм выбирает дугу, инцидентную вершине v с минимальным уровнем.

17 Связность в орграфах (сильная, односторонняя и слабая связность, компоненты сильной связности). Алгоритм выделения компонент сильной связности.

18 Деревья. Свободные деревья. Основные свойства деревьев (с доказательствами). Код Прюфера.

19 Ориентированные, упорядоченные и бинарные деревья. Свойства ордерева. Эквивалентное определение ордерева. Упорядоченные деревья.

20 Представление деревьев в ЭВМ. Обходы бинарных деревьев. Алгоритм симметричного обхода бинарного дерева

21 Деревья сортировки. Ассоциативная память, способы реализации ассоциативной памяти. Алгоритм поиска в дереве сортировки.

22 Выровненные, заполненные и полные деревья. Сбалансированные деревья. Алгоритм бинарного (двоичного) поиска.

23 Информационные деревья. A- и B-деревья. Красно-черные деревья.

24 Кратчайший остов. Алгоритм построения остова экстремального веса. Алгоритм Краскала. Алгоритм Прима. Алгоритм Борувки. Число остовов в связном обыкновенном графе. Задача Штейнера

25 Фундаментальные циклы и разрезы. Фундаментальная система циклов и циклический ранг. Фундаментальная система разрезов и коциклический ранг. Подпространства циклов и коциклов

26 Эйлеровы циклы. Эйлеровы графы. Алгоритм Флери. Оценка числа эйлеровых графов.

27 Гамильтоновы циклы. Теорема Дирака. Задача коммивояжера.

28 Гиперграфы. Двойственные гиперграфы. Циклы и реализации.

29 Задачи маршрутизации. VRP. Классификация. Методы решения задач VRP.

30 Независимые и покрывающие множества вершин и ребер. Теорема о связи чисел независимости и покрытий

31 Построение независимых множеств вершин. Поиск с возвратами. Улучшенный перебор. Доминирующие множества. Доминирование и независимость. Задача о наименьшем покрытии.

32 Ядро графа. Алгоритм Магу. Спектры графов.

33 Разметка графа. Грациозная, счастливая разметки. Раскраска графа. Примеры задач. Хроматическое число. Алгоритмы раскрашивания. Двойственный граф

34 Планарность. Укладка графов. Эйлерова характеристика. Теорема о пяти красках.

35 Элементы сетевого планирования: критические пути, работы, резервы. Линейные графики. Алгоритм сетевого планирования