

SPb HSE, 1 курс, зима 2024/25

Практика по алгоритмам #19

BFS, Dijkstra

18 февраля

Собрано 18 февраля 2025 г. в 21:30

---

## Содержание

1. BFS, Dijkstra	1
2. Разбор задач практики	3
3. Домашнее задание	7
3.1. Обязательная часть . . . . .	7
3.2. Дополнительная часть . . . . .	8

# BFS, Dijkstra

## 1. Два коня

На шахматной доске  $32 \times 32$  с выколотыми клетками стоят два коня. Кони ходят строго по очереди. За какое минимальное число шагов можно поменять их местами?

## 2. Модификации bfs

- a) 1-2-bfs, веса вещественные.
- b) 0-1-bfs, веса вещественные.
- c)  $k$ - $2k$ -bfs, веса вещественные.

## 3. Ускорение 1- $k$ -bfs

- a)  $\mathcal{O}(E \log k)$
- b)  $\mathcal{O}(E \log \log(Vk))$
- c) (\*)  $\mathcal{O}(E \log \log k)$

## 4. Ускорение Дейкстры

- a)  $\mathcal{O}(E + V \log V)$
- b)  $\mathcal{O}(E \log \log C)$

## 5. Чётные пути

Дан оргграф,  $w_e \geq 0$ . Найдите кратчайший путь из  $s$  в  $t$ , состоящий из чётного числа рёбер.

## 6. Флойд и битовое сжатие

Найдите транзитивное замыкание графа за  $\mathcal{O}(\frac{V^3}{w})$ .

## 7. Матрица расстояний за $\mathcal{O}(\frac{V^3}{w})$

В оргграфе с единичными весами.

## 8. Число кратчайших путей

Дан взвешенный оргграф с положительными весами и вершина  $s$ , нужно для каждой вершины  $v$  найти число кратчайших путей из  $v$  в  $s$ .  $\mathcal{O}(E \log V)$ . А если есть нулевые веса?

## 9. Кратчайший путь по выделенным вершинам

Дан взвешенный оргграф с положительными весами. Найти кратчайший путь из  $s$  в  $t$ , проходящий по всем  $k \leq 10$  выделенным вершинам.

## 10. Обобщение Дейкстры

На каких функциях кроме суммы работает алгоритм Дейкстры?

Рассмотрите функции «\*»,  $\min$ ,  $\max$ ,  $or$ ,  $and$ ,  $gcd$ .

Придумайте общий критерий для функции.

## 11. Максимальный корабль

Даны две параллельные прямые (река). Река течёт слева направо. В реке есть  $n$  островов (точек). Нужно провести по реке круглый корабль максимального радиуса  $R$  так, чтобы он не задел ни одного острова. Корабль изначально находится сильно левее всех островов,

хочет уплыть вправо за горизонт. Найти только число  $R$  (траекторию искать не нужно) за  $\mathcal{O}(n^2 \log M)$ . А за  $\mathcal{O}(n^2)$ ? А строго доказать?

## 12. Платные дороги

Дан взвешенный орграф, вес дороги – время в пути. Дороги бывают платные. Нужно дойти за минимальное время из  $s$  в  $t$ , использовав не более 100 платных дорог (бюджет).

## 13. Веса-пары: кратчайший по $y$ путь с ограничением на $x$

Есть *орграф*, на ребрах этого орграфа написаны пары положительных целых чисел  $\langle x_e, y_e \rangle$ . Нужно найти путь  $\text{path}$  из  $s$  в  $t$  такой, что  $\sum_{e \in \text{path}} x_e \leq A$ , а  $\sum_{e \in \text{path}} y_e \rightarrow \min$ .  $\mathcal{O}(V + E \cdot A)$ .

## 14. Невероятный Дейкстра

У нас есть невзвешенный *орграф*. Для каждого ребра  $e$  известна вероятность  $p_e$ , с которой у нас получится по ребру пройти. Мы начинаем в вершине  $s$ . За ход мы можем выбрать любое ребро и попытаться по нему пройти. Найдите матожидание числа ходов, за которое мы сможем попасть в  $t$ , если будем действовать оптимально.

## 15. (\*) Двусторонний алгоритм Дейкстры

*Историческая справка: Эдсгер Виле Дейкстра – знаменитый голландский учёный. Правильно говорить не «Дейкстра работает», а «алгоритм Дейкстры работает». Ниже приведён пример того, как говорить не следует.*

Ищем кратчайший путь  $s \rightsquigarrow t$  так: запускаем параллельно два Дейкстры – из  $s$  вперед (массив  $d_1, \text{heap}_1$ ) и из  $t$  назад (массив  $d_2, \text{heap}_2$ ). На каждом шаге двигаем меньший из  $\text{heap}_1.\text{min}$ ,  $\text{heap}_2.\text{min}$ . Каждый раз, когда один из Дейкстр улучшает расстояние до  $v$ , он обновляет ответ:  $\text{relax}(\text{ans}, d_1[v] + d_2[v])$ .

Доказать корректность алгоритма в двух случаях. Если алгоритм останавливается, когда

- нашлась вершина  $v$ , обработанная обоими Дейкстрами (найлены верные  $d_1[v]$  и  $d_2[v]$ );
- (\*) текущий ответ  $\text{ans} = \min_v(d_1[v] + d_2[v]) \leq \text{heap}_1.\text{min}() + \text{heap}_2.\text{min}()$ .

## 16. (\*) Topsort в плотных графах

Найдите топологическую сортировку DAG-а за  $\mathcal{O}\left(\frac{V^2}{w}\right)$ .

## 17. (\*) Плохая функция для $A^*$

Придумать функцию оценки  $f$  для  $A^*$ , на которой он будет работать дольше Дейкстры.  $\exists f: A^*$  работает экспоненциально долго.

## 18. (\*) Необычное расстояние

Дан орграф ( $n, m \leq 10^5$ ). у каждого ребра есть целый положительный вес от 1 до 50. Вы начинаете в вершине  $s$ . За ход можно пройти ровно по двум последовательным рёбрам. Если вы прошли по рёбрам с весами  $w_1, w_2$ , вы платите штраф  $(w_1 + w_2)^2$ . Нужно дойти до вершины  $t$ , заплатив минимальный штраф.

# Разбор задач практики

## 1. Два коня

Главное увидеть граф. Вершина – позиции двух коней и битик «кто ходит». Вершин  $1024 \cdot 1024 \cdot 2 \approx 2\,000\,000$ . Ребро – ход коня.

## 2. Модификации bfs

а) 1-2-bfs, веса вещественные.

Три очереди: вершины на расстояниях  $[d, d + 1)$ ,  $[d + 1, d + 2)$ ,  $[d + 2, d + 3)$ .

Когда вынимаем из первой очереди вершину на расстоянии  $d'$  и смотрим из нее на ребро веса  $w$ , будет  $d' + w < d + 3$ . По  $d' + w$  поймем, в какую очередь ее положить.

Корректность, как у 1- $k$ -bfs. Инвариант: на текущий момент верно посчитаны расстояния для всех вершин на расстоянии  $< d + 1$ .

Когда опустела  $[d, d + 1)$  очередь, инвариант верен уже для следующей очереди: если до вершины расстояние  $[d + 1, d + 2)$ , то до предыдущей на кратчайшем пути оно  $[d - 1, d + 1)$ , такие уже обработаны.

б) 0-1-bfs, веса вещественные. Это эквивалентно задаче с произвольными весами.

Были 0-1 веса  $\Rightarrow$  делим все веса на минимальный, получили сколь угодно большие веса.

Были произвольные веса  $\Rightarrow$  делим все веса на максимальный, получили веса 0-1.

В общем, не надо это решать bsf-ом, надо Дейкстрой.

в)  $k$ -2 $k$ -bfs. Можно поделить все веса на  $k$  и получить 1-2-bfs.

А можно не делить явно, а поделить мысленно, и сделать то же, что в 1-2-bfs. То есть будут очереди для расстояний  $[d, d + k)$ ,  $[d + k, d + 2k)$ ,  $[d + 2k, d + 3k)$ .

## 3. Ускорение 1- $k$ -bfs

Напомним, в 1- $k$ -bfs мы обрабатываем каждое ребро один раз, а вот для поиска следующей вершины нам нужна непустая очередь с минимальным расстоянием, на ее поиск может уйти  $k$  операций, итого  $\mathcal{O}(E + kV)$ .

а)  $\mathcal{O}(E \log k)$ : храним номера непустых очередей в бинарной куче.

Время  $V \cdot \text{ExtractMin} + E \cdot \text{Add} = \mathcal{O}(E \log k)$ .

б)  $\mathcal{O}(E \log \log(Vk))$ : храним номера непустых очередей в van Emde Boas tree (все операции за  $\mathcal{O}(\log \log C)$ ). Номера очередей:  $[0, \text{MAXDIST}]$ ,  $\text{MAXDIST} \leq Vk$ .

в)  $\mathcal{O}(E \log \log k)$ : храним два van Emde Boas tree: одно для расстояний  $[ki, k(i + 1))$ , другое для  $[k(i + 1), k(i + 2))$ .

В каждом храним номера от 0 до  $k$ , так что  $\log \log k$ .

## 4. Ускорение Дейкстры

Напомним, Дейкстра работает за  $V \cdot \text{ExtractMin} + E \cdot \text{DecreaseKey}$ .

а)  $\mathcal{O}(E + V \log V)$  – куча Фибоначчи.

б)  $\mathcal{O}(E \log \log C)$  – van Emde Boas.

## 5. Чётные пути

Состояние =  $(v, b)$ , где  $v \in V$ ,  $b \in \{0, 1\}$ .  $b$  отвечает за чётность длины пути в  $v$ . Ищем путь из  $(s, 0)$  в  $(t, 0)$ .

## 6. Флойд и битовое сжатие

Как выглядит обычный фloyd для транзитивного замыкания?

```

1 for k = 0..n-1:
2 for i = 0..n-1:
3     for j = 0..n-1:
4         a[i][j] |= (a[i][k] and a[k][j])

```

Перепишем код, так чтобы использовать битовое сжатие (`a[i]` это теперь битсет).

```

1 for k = 0..n-1:
2 for i = 0..n-1:
3     if (a[i][k])
4         a[i] |= a[k] // a[i][j] |= (a[i][k] and a[k][j])

```

## 7. Матрица расстояний за $\mathcal{O}(\frac{n^3}{w})$

bfs за  $\mathcal{O}(\frac{n^2}{w})$  из каждой вершины.

Поддерживаем `bitset unvisited` еще не посещенных вершин.

Строим слои (как на лекции)  $A_0, A_1, A_2, \dots$

$$A_{d+1} = \cup_{v \in A_d} (g[v] \cap unvisited)$$

Все указанные операции умеет делать `bitset`.

```

1 for (int v = A[d]._Find_first(); v < n; v = A[d]._Find_next(v))
2     A[d+1] |= g[v] & unvisited;
3 unvisited &= ~A[d+1];

```

На самом деле код работает даже чуть быстрее, за  $\mathcal{O}(n + \frac{n}{w} \cdot \text{MAXDIST})$ .

Самостоятельно написать перебор единичных битов битсета можно так: перебираем  $\frac{n}{w}$  групп по  $w$  бит. В каждой не пустой группе мы умеем за  $\mathcal{O}(1)$  взять младший бит.

## 8. Число кратчайших путей

*Решение #1:* запускаем Дейкстру, оставляем только ориентированные ребра  $(a, b, w): d[a] + w = d[b]$ .

Получили так называемый «граф кратчайших путей».

$\forall e: w[e] > 0 \Rightarrow$  граф кратчайших путей ацикличен. Считаем на нем динамикой число путей из  $s$  во все  $v$ .

Если есть нулевые ребра, будут нулевые циклы, по которым можно бесконечно крутиться.

*Решение #2:* по ходу Дейкстры считаем еще и `count[v]` – число путей длины  $d[v]$ .

## 9. Кратчайший путь по выделенным вершинам

Запустим Дейкстру из  $k$  фиксированных вершин и начальной.

Теперь сделаем оргграф только из  $k+2$  вершин – фиксированные, начальная, конечная.

Вес ребер в нем равен кратчайшему пути в исходном графе.

Получили задачу коммивояжера на  $k+2$  вершинах. Итого  $\mathcal{O}(k \cdot \text{Dijkstra} + 2^k k^2)$ .

*Альтернативное решение.* Сразу пустить Дейкстру на графе из  $2^k n$  вершин, где вершина = пара (маска посещенных выделенных, текущая).

## 10. Обобщение Дейкстры

Нам нужны

оптимальность подзадач:  $\forall a, b, c (a \leq b) \Rightarrow f(a, c) \leq f(b, c)$   
 и монотонность:  $\forall a, c f(a) \leq f(a, c)$ .

Примеры:

- Хорошая.  $f(a, b) = a + b$  в  $\mathbb{R}^{\geq 0}$ . Условие « $\geq 0$ » нужно для монотонности.
- Хорошая.  $f(a, b) = \max(a, b)$  в  $\mathbb{R}$ . (минимизируем  $\max$ )
- Плохая.  $f(a, b) = \min(a, b)$  в  $\mathbb{R}$ . (минимизируем  $\min$ )
- Плохая.  $f(a, b) = \text{OR}(a, b)$  в  $\mathbb{Z}^{\geq 0}$ . Нет оптимальности подзадач.
- Плохая.  $f(a, b) = \text{AND}(a, b)$  в  $\mathbb{Z}^{\geq 0}$ . Нет оптимальности подзадач.
- Плохая.  $f(a, b) = \text{gcd}(a, b)$  в  $\mathbb{Z}^{> 0}$ . Нет оптимальности подзадач.
- Если мы хотим максимизировать, следует сперва домножить всё на  $-1$ .
- Хорошая.  $f(a, b) = a \cdot b$  в  $\mathbb{R}^{\geq 1}$ . Условие « $\geq 1$ » нужно для монотонности. Можно прологарифмировать веса, тогда произведение перейдёт в сумму, а 1 перейдёт в 0.

## 11. Максимальный корабль

*Простое решение.* Бинпоиск по ответу, внутри раздуем препятствия и берега, и сделаем вид, что корабль – точка. Построим граф пересечений и проверим, есть ли путь от верхнего берега к нижнему, если есть такой путь, есть разрез реки, через который не проплыть.  $\mathcal{O}(n^2 \log M)$ .

*Корректность простого решения.* Пусть пути (разреза) нет  $\Rightarrow$  существует кривая из  $-\infty$  в  $+\infty$ . Эта импликация математически объясняется так: рассмотрим множество точек достижимых из  $-\infty$ , если оно ограничено справа, посмотрим на границу справа, на разрез... ой, а разреза то  $\nexists$ , ну, значит, неограничено.

*Быстрое решение.* Построим граф с  $n+2$  вершинами – два берега и острова. Вес ребра между вершинами – евклидово расстояние на плоскости.

Максимальный радиус  $R$  равен длине кратчайшего пути от одного берега до другого в этом графе, где вес пути – максимум весов рёбер на пути.

Способы поиска пути: бинпоиск по ответу + `dfs` за  $\mathcal{O}(n^2 \log M)$  или Дейкстра за  $\mathcal{O}(n^2)$ .

*Корректность быстрого решения.* Оно делает ровно то же самое, что простое

## 12. Платные дороги

Построим новый граф. На нём применим Дейкстру. Вершины нового графа – пары  $\langle v, k \rangle$ , где  $k$  – число пройденных платных дорог.

## 13. Невероятный Дейкстра

Матожидание времени «перейти по ребру вперёд» есть  $\frac{1}{p_e}$ .

Пускаем Дейкстру по рёбрам с такими весами.

Чтобы доказать корректность. Мысленно пустим Дейкстру именно из  $t$  в  $s$  по обратным рёбрам. Пусть  $E[v]$  – «матожидание времени дойти до конца (до  $t$ ) из  $v$ ».  $E[v] = \min_e (\frac{1}{p_e} + E[\text{end}_e])$ . Что собственно и найдёт Дейкстра.

## 14. (\*) Двусторонний Дейкстра

Пусть есть путь  $p$  короче, чем мы нашли. Тогда  $\text{len}(p) < c_1 + c_2$ , где

- $c_1 = d_1[i]$ ,  $c_2 = d_2[i]$ , где  $i$  – обработанная обеими сторонами вершина
- $c_1 = \text{heap}_1. \min()$ ,  $c_2 = \text{heap}_2. \min()$

На пути  $p$  есть последняя вершина  $v$ :  $d_1[v] < c_1$ , за ней идет  $u$ :  $d_1[u] \geq c_1$ .

$d_1[u] + d_2[u] = \text{len}(p) < c_1 + c_2 \Rightarrow d_2[u] < c_2 \Rightarrow d_1[v]$  и  $d_2[u]$  к моменту окончания алгоритма

посчитаны верно  $\Rightarrow len(p) = d_1[v] + w(v, u) + d_2[u] < c_1 + c_2$ .

Кто-то из  $v$  и  $u$  был обработан раньше, пусть  $v$ .

Когда второй Дейкстра обработал  $u$ , он сделал  $relax(d_2[v], d_2[u] + w(v, u))$  и сразу после  $relax(ans, d_1[v] + d_2[v])$ . Но  $d_1[v] + d_2[v] < d_1[v] + w(v, u) + d_2[u] < c_1 + c_2 \leq ans$ . Противоречие.

### 15. (\*) **Toposort в плотных графах**

dfs за  $\mathcal{O}(\frac{V^2}{w})$ .

Обрабатываем вершину  $v$ . Помечаем ее посещенной в `bitset unvisited`. Пока `g[v] & unvisited != 0`, делаем `g[v] &= unvisited` и вызываемся рекурсивно от младшей единицы в `g[v]`.

На каждом ребре дерева dfs делаем AND `bitset`-ов и ищем младшую единицу в `bitset`. И то, и то за  $\mathcal{O}(\frac{V}{w})$ , а ребер в дереве dfs  $(V - 1)$ . Итого  $\mathcal{O}(\frac{V^2}{w})$ .

### 16. (\*) **Весы-пары: кратчайший по $y$ путь с ограничением на $x$**

$d[v, a]$  – кратчайший путь из  $s$  в  $v$ , на котором сумма  $x_e \leq a$ .

Раз все  $a_e > 0$ , можно пересчитывать  $d[v, a] = \min_{e=(u,v)} d[u, a - x_e]$  в порядке увеличения  $a$ .

Суммарное число переходов –  $\mathcal{O}(E \cdot A)$ .

### 17. (\*) **Плохая функция для $A^*$**

Пусть кратчайший путь из  $n$  в 1 равен  $n \rightarrow n - 1 \rightarrow \dots \rightarrow 2 \rightarrow 1$ , веса ребер 1.

Также между всеми парами вершин есть ребро веса  $n$ .

Возьмем  $f_v = n^v$ . Тогда  $A^*$  будет работать за  $2^n$ , т.к. сперва пойдёт не в  $n - 1$ , а в 1...

$T(k)$  – время работы от вершины  $k$  до конца.  $T(k) = 1 + T(1) + T(2) + \dots + T(k - 1) = 2^{k-1}$ .

### 18. (\*) **Необычное расстояние**

Давайте думать о задаче в таком ключе: для центральной вершины нам важны лишь веса рёбер, через которые мы проходим. Так что давайте для каждой вершины  $v$  создадим мнимые вершины  $(v, w)$ , где  $w$  – вес последнего ребра по которому мы прошли. Это добавит не более  $\mathcal{O}(M)$  вершин в наш граф. Теперь для каждого ребра  $(u, v, w)$  в изначальном графе создадим ребро  $u \rightarrow (v, w)$  веса 0. Также для каждой вершины  $(u, was)$  и ребра  $(u, v, w)$  создадим ребро  $(u, was) \rightarrow v$  весом  $(was + w)^2$ . Теперь, если запустить алгоритм Дейкстры от вершины 1, то мы получим корректные ответы для всех вершин, так как мы просимулировали новое правило при помощи фейковых вершин и ребер. Также мы создадим не более  $\mathcal{O}(M \cdot maxW)$  ребер, так как для вершины степени  $t$  мы создадим не более  $t \cdot maxW$  ребер, а сумма всех  $t$  равна  $2 \cdot M$ .

# Домашнее задание

## 3.1. Обязательная часть

### 1. (1) Операции над числами

Изначально есть число  $X$ . За один ход можно сделать любое из трёх действий

- a)  $X \rightarrow (X + 1) \bmod N$
- b)  $X \rightarrow (X - 1) \bmod N$
- c)  $X \rightarrow 2X \bmod N$

За минимальное число операций получить число  $Y$ .

Даны  $X, Y, N$ .  $0 \leq X, Y < N \leq 10^6$ .

### 2. (2) Число кратчайших путей

Дан невзвешенный неорграф и два множества вершин –  $A$  и  $B$ ,  $A \cap B = \emptyset$ . Найти (длину кратчайшего пути из  $A$  в  $B$ ) и (количество кратчайших путей из  $A$  в  $B$ )  $\bmod 10^9$ .  $\mathcal{O}(n + m)$ .

(+1) балл за случай  $A = B$  ( $\min_{u,v \in A, u \neq v} d(u, v)$ ).

### 3. (2) Предподсчет

Дан неорграф. Сделайте предподсчет за  $\mathcal{O}(n^3)$ , чтобы за  $\mathcal{O}(1)$  отвечать запросу  $\langle a, b, e \rangle$  – существует ли кратчайший путь из  $a$  в  $b$ , проходящий через ребро  $e$ ?

### 4. (2) Чётный разноцветный путь

Дан оргграф  $n, m \leq 10^5$  у каждого ребра есть вещественный положительный вес и цвет (целое число от 1 до 10). Найти кратчайший путь из  $s$  в  $t$ , состоящий из чётного числа рёбер и хотя бы двух разных цветов.

### 5. (3) Обмен местами

Есть невзвешенный оргграф. Для **каждой пары** вершин  $a, b$  определена функция расстояния  $f(a, b)$ . Вася и Петя стоят в вершинах  $v$  и  $u$  соответственно и хотят поменяться местами, не оказываясь ни в какой момент времени ближе, чем на расстоянии  $d$ . За ход **любой один** из них переходит в смежную вершину. За какое минимальное число ходов они могут это сделать?  $\mathcal{O}(nm)$ . (Частичный балл за  $\mathcal{O}(n^2m)$ )



## 3.2. Дополнительная часть

### 1. (3) Странное расстояние

Найти в орграфе от вершины 1 кратчайшее расстояние до всех вершин. Расстояние от пути  $w_1, w_2, \dots, w_k$  считается как  $\sum_i w_i - \max_i w_i + \min_i w_i$ .  $V, E \leq 10^5$ .

### 2. (3) Заправки

Страна – города и односторонние дороги. У нас есть машина с баком бензина размера  $C$ , в некоторых городах есть заправки с бесконечным бензином. Для каждой дороги известно, сколько бензина нужно потратить, чтобы по ним проехать. Мы стартуем в городе  $s$  с полным баком. До куда мы сможем доехать? (1)  $\mathcal{O}(VE)$ , (+2)  $\mathcal{O}(E \log V)$ .

### 3. (3) Антитест для bfs

Постройте матрицу  $n \times n$ , состоящую из стенок и пустых клеток, на которой при выполнении bfs-а из правого нижнего угла размер очереди будет  $\omega(n)$ . Ходить можно между вершинами смежными по стороне.

### 4. (3) Сумма двух максимальных ребер

Есть взвешенный неограф. Найти путь из  $s$  в  $t$  такой, что сумма двух максимальных ребер на пути минимальна.  $\mathcal{O}((n + m) \cdot \text{poly}(\log n))$ .

### 5. (4) Запросы с уменьшением весов

Дан взвешенный граф с положительными ребрами, в нем выделены вершины  $a$  и  $b$ . Запрос  $\text{Get}(s)$  – минимальная длина пути из  $a$  в  $b$ , причем весом ребра  $e$  считается  $\min(s, w_e)$ .  $s > 0$ . Научиться быстрее, чем « $k$  раз запустить Дейкстру» обработать  $k$  запросов  $\text{Get}(s)$ .