

Московский физико-технический институт
Физтех-школа прикладной математики и информатики

ОСНОВЫ КОМБИНАТОРИКИ И ТЕОРИИ ЧИСЕЛ
II СЕМЕСТР

Лектор: *Андрей Михайлович Райгородский*



Автор: *Киселев Николай*
Репозиторий на Github

весна 2025

Содержание

| | | |
|----------|--------------------------------------|----------|
| 1 | И снова графы | 2 |
| 2 | Кратчайшие пути. BFS | 2 |
| 2.1 | BFS (breadth first search) | 3 |
| 2.2 | 0-k BFS | 3 |
| 2.3 | Двусторонний BFS | 4 |
| 2.4 | Алгоритм Дейкстры | 4 |
| 3 | Теория вычислимости | 5 |
| 3.1 | Алгоритм | 5 |
| 4 | Однолетночная машина Тьюринга | 5 |

«««< HEAD

1 И снова графы

Определение 1.1. Пусть G - неориентированный граф. v - точка сочленения (ТС), если после ее удаления количество компонент связности увеличивается хотя бы на 1.

Утверждение 1.1.

1. Если v - корень, то v - точка сочленения $\iff \exists v$ в дереве dfs хотя бы 2 ребенка
2. Если v - не корень, то v - точка сочленения $\iff \exists$ деревесное ребро (v, to) $ret[v] \geq tin[to]$

Доказательство.

1. Корень либо является листом - тогда после его удаления количество компонент не изменится, либо у него есть 2 сына, которые образуют 2 компоненты связности, что удовлетворяет определению ТС.

Подграфы сыновей не связаны, иначе по dfs это была бы 1 общая компонента.

2. Если есть сын to , что $ret[to] \geq tin[v]$, то после удаления v заведомо пропадает путь между to и p , родителем v .

Если же, напротив, для всех сыновей $ret[to] < tin[v]$, то после удаления v сохраняется путь между p и поддеревьями v .

□

2 Кратчайшие пути. BFS

Определение 2.1. Взвешенным графом называется (V, E, w) , где (V, E) - граф, $w : E \rightarrow \mathbb{R}$ Иначе говоря, просто граф с весами на каждом ребре.

Определение 2.2. Весом (стоимостью) пути назовем сумму весов рёбер в нем. $dist(s, t)$ определим, как минимальное значение среди весов путей от s до t .

Важное уточнение!

Замечание.

1. Если пути от s до t нет, то $dist(s, t) = +\infty$
2. Если есть отрицательный цикл, то считаем, что $dist(s, t) = -\infty$

Далее временно считаем, что $\forall e \ w(e) \geq 0$, $w = 1$

2.1 BFS (breadth first search)

Цель: по фиксированной вершине s найти $dist(s, v) \forall v$

Понятно, что $dist(s, s) = 0, dist(s, x) = 1, \forall$ смежных с s вершин. Продолжим цепочку...

1. Заведем массив $d[v]$ - найденная длина пути от s до v .
2. Введем функцию $expand(int v)$ - раскрытие v :

```
for (edge e : g[v]) {
    обновляем d[e.to] через
    d[v] + e.w
    если нужно, кладем e.to в структуру
}
```

3. $d[0 \dots n - 1] = +\infty, d[s] = 0, queue\ q; q.push(s)$

4.

```
while (q не пусто) {
    v = q.front(); q.pop();
    for (edge e: g[v]) {
        if (d[e.to] == +infty) {
            d[e.to] = d[v] + 1;
            q.push(e.to);
        }
    }
}
```

Утверждение 2.1. К моменту рассмотрения последней вершины очереди с $d[v] = k$:

1. До всех вершин $u : dist(s, u) \leq k + 1$ найден правильный ответ ($d[u] = dist(s, u)$)
2. В очереди лежат все вершины с $dist(s, u) = k + 1$ (и только они)

Проводится по индукции. Оставим в качестве упражнения для читателей:)

2.2 0-k BFS

Асимптотика $O(V + E)$

Похожая задача. **Цель:** $\forall v$ найти длину минимального найденного пути от s до v , но теперь веса $w \in \{0, 1, 2, \dots, k\}$

1. Храним в $dp[v]$ длину минимального найденного пути от s до v
2. $d = +\infty, d[s] = 0, q[x]$ - очередь вершин с $d = x$

- ```

3. expand(v):
 if (expanded[v]) return;
 expanded[v] = true;
 for (e: g[v]) {
 y = d[v] + e.w
 if (d[e.to] > y) {
 d[e.to] = y;
 q[y] push(e.to);
 }
 }
}

4. Заведем $q[0], q[1], \dots, q[nk]$, $expanded = false$, $q[0].push(s)$

5. for (x = 0 ... nk) {
 while (q[x] не пусто):
 достаем из нее вершину u и раскрываем
 }

```

Асимптотика:  $O(E + kV)$

**Упражнение.** К моменту завершения рассмотрения очереди  $q[x]$  обработаны и раскрыты все вершины, для которых расстояние не больше  $x$ .

## 2.3 Двусторонний BFS

**Цель:** найти  $dist(s, t)$ . **Общая идея:** Нарастиваем слои по глубине  $k$  от двух вершин, пока области не пересекутся.

**Решение:**

1. Запускаем *BFS* параллельно. Назовем слоем  $k$  множество вершин, до которых можно добраться за  $k$  или меньше ребер.
2. Заметим, что  $dist(s, t) = \min_m (d_s[m] + d_t[m])$ . Так что когда найдется такое  $k$ , что слои от  $s$  и  $t$  пересекутся, мы получим эту вершину  $m$  и, соответственно, путь между  $s$  и  $t$ .

*Проверять пересечение слоев можно проверять быстро через хеш-таблицы.*

## 2.4 Алгоритм Дейкстры

**Цель:**  $w \geq 0$ , *fixs* Найти  $dist(s, v) \forall v$ .

1.  $d = +\infty$ ,  $expanded[v] = false \forall v$ ,  $d[s] = 0$
2. for  $i = 0 \dots n-1$ :
  - пусть  $v$  - вершина с  $\min d[v]$  среди всех нераскрытых
  - Ийййеессслииии  $d[v] = +\infty$ : break

Асимптотика  $O(n + n \log n)$  через Фиб кучу и  $O(m \log n)$  через бин кучу =====

## 3 Теория вычислимости

### 3.1 Алгоритм

Неформально: алгоритм — это процедура, преобразующая данные, закодированные конечными словами, которая тоже имеет конечное описание и выполняется пошагово

**Пример** (Не алгоритм). Метод Ньютона нахождения нуля дифференцируемой. Не является алгоритмом, т.к. нельзя сделать предельный переход, однако, если установить точность с которой мы хотим узнать корень, тогда норм.

**Пример** (Не алгоритм). Метод дележа пирога. Есть пирог, хотим поделить его. Берем нож и несем его над пирогом. Второй человек говорит, когда нам остановиться. Тогда мы и режем пирог. Не является алгоритмом, т.к. время тут не дискретно.

## 4 Одноленточная машина Тьюринга

Одноленточная машина Тьюринга

|     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|
| ... | $m$ | $a$ | $t$ | $l$ | $o$ | $g$ | ... |
|-----|-----|-----|-----|-----|-----|-----|-----|

Есть бесконечная в обе стороны лента (см. выше), в которой хранятся некоторые символы некоторого алфавита. Машина тьюринга представляет собой функцию от этой ленты:  $qa \mapsto rbD$

1.  $q$  — текущее состояние машины
2.  $a$  — символ в ячейке, на которую смотрит машина Тьюринга
3.  $r$  — новое состояние машины
4.  $b$  — новый символ в ячейке
5.  $D$  — направление сдвига  $L, N, R$

Итак, формально:

**Определение 4.1.** Машина Тьюринга — это кортеж  $(\Sigma, \Gamma, Q, q_1, q_0, \delta)$ , где  $\Sigma, \Gamma, Q$  — конечные множества

1.  $\Sigma$  — входной алфавит
2.  $\Gamma \supset \Sigma$  — ленточный алфавит.  $\#$  — пробел, принадлежит  $\Gamma \setminus \Sigma$
3.  $Q \cap \Gamma = \emptyset$  — множество состояний
4.  $q_1, q_0 \in Q$  — начальное и кончное состояния соответственно
5.  $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, N, R\}$

Удобно записывать состояние машины Тьюринга так:

$AqaB$

1.  $a$  — символ, на который мы сейчас смотрим
2.  $q$  — состояние машины в данный момент
3.  $A, B$  — лента до и после символа, на который мы смотрим, соответственно

**Определение 4.2.** Вычисление — последовательность конфигураций, где каждая следующая получается из предыдущей по правилу Машины Тьюринга

**Определение 4.3.** Функция  $f : \Sigma^* \rightarrow \Sigma^*$  называется вычислимой, если существует Машина Тьюринга, такая, что

$$\forall x \begin{cases} f(x) \text{ определена} \Rightarrow M(x) = f(x) \\ f(x) \text{ не определена} \Rightarrow M(x) \text{ не останавливается} \end{cases}$$

Заметим, что Машин Тьюринга счетное множество, а функций  $\Sigma^* \rightarrow \Sigma^*$  континуум  $\Rightarrow \exists$  невычислимые функции.

**Утверждение 4.1.**  $U f$  конечная область определения  $\Rightarrow f$  вычислима

**Утверждение 4.2.**  $f, g$  — вычислимы  $\Rightarrow f \circ g$  — тоже

**Определение 4.4.**  $S \subset \Sigma^*$  разрешимо, если  $\exists$  Машина Тьюринга с бинарным ответом, такая, что  $M(x) \begin{cases} 1, x \in S \\ 0, x \notin S \end{cases}$  Иначе говоря,  $S$  разрешимо, если  $\chi_S(x) = \begin{cases} 1, x \in S \\ 0, x \notin S \end{cases}$  вычислима

**Утверждение 4.3.**  $S$  — конечное  $\Rightarrow S$  разрешимо

**Утверждение 4.4.**  $S, T$  — разрешимы  $\Rightarrow S \cup T, S \cap T, \bar{S}$  — тоже

**Замечание.** Подмножество разрешимого множества может быть неразрешимо.

**Теорема 4.1** (Критерий Разрешимости).  $S$  разрешимо  $\Leftrightarrow S$  можно перечислить по возрастанию

*Доказательство.*

```
 \Rightarrow for i in 0, 1, 2...
 if i in S:
 print i
```

$\Leftarrow$  Пусть  $S$  бесконечно, а на вход подается  $x$ . Печатаем элементы, пока они  $< x$ . Тогда первый элемент, на котором это сломается будет либо равен  $x$ , либо будет  $> x$ . В первом случае выдаем 1, иначе 0. Для конечных оно и так разрешимо.

□

**Определение 4.5.** Множество называется перечислимым, если существует Машина Тьюринга, которая выводит все его элементы  $S$  и только их

**Теорема 4.2** (Поста).  $S$  разрешимо  $\Leftrightarrow S, \bar{S}$  перечислимы

*Доказательство.*

$\Rightarrow$  пробегаемся по  $\Sigma^*$ , и, если элемент  $\in S$ , то выводим его. Аналогично для  $\bar{S}$

$\Leftarrow$  По очереди перечисляем элементы  $S, \bar{S}$ . Рано или поздно мы встретим  $x \Rightarrow$  выведем, где мы его встретили, в  $S$  или  $\bar{S}$ . □

»»»> c7346d4998339b210ee9da181a04419c62922566