

Первый курс, весенний семестр 2019/20

Практика по алгоритмам #1

Поиск в глубину (dfs)

17 января

Собрано 16 января 2022 г. в 14:03

---

## Содержание

1. Поиск в глубину (dfs)	1
2. Разбор задач практики	2
3. Домашнее задание	4
3.1. Обязательная часть . . . . .	4
3.2. Дополнительная часть . . . . .	5

# Поиск в глубину (dfs)

## 1. Достижимость множеств

Даны два множества вершин:  $A$  и  $B$ , за  $\mathcal{O}(V+E)$  проверить, есть ли путь из какой-нибудь вершины  $a \in A$  в какую-нибудь вершину  $b \in B$ .

## 2. Циклы

- Найти цикл в орграфе через данное ребро за  $\mathcal{O}(E)$ .
- Найти цикл в орграфе через данную вершину за  $\mathcal{O}(E)$ .
- Найти в неориентированом графе какой-нибудь цикл за  $\mathcal{O}(E)$ .
- Найти цикл в неорграфе через данное ребро за  $\mathcal{O}(E)$ .
- Найти цикл в неорграфе через данную вершину за  $\mathcal{O}(E)$ .
- Найти в неориентированом графе какой-нибудь цикл за  $\mathcal{O}(V)$ .

## 3. Гамильтонов путь

Дан ациклический орграф, найти в нем гамильтонов путь за  $\mathcal{O}(V+E)$ .

## 4. Единственность топсорта

Дан ациклический орграф, проверить единственность топсорта за  $\mathcal{O}(V+E)$ .

## 5. Ациклическая ориентация

Ориентировать неорграф так, чтобы он стал ациклическим, за  $\mathcal{O}(V+E)$ .

## 6. Сильносвязная ориентация

Ориентировать неорграф так, чтобы он стал сильносвязным, за  $\mathcal{O}(V+E)$ .

## 7. Нечетный цикл

За  $\mathcal{O}(V+E)$  найти в неорграфе цикл нечетной длины.

## 8. Разделение на две клики

Разделить неорграф на две клики или проверить, что это невозможно, за  $\mathcal{O}(V^2)$ .

## 9. Разбиение на дружелюбные доли

У каждой вершины не более 3 врагов.

Разбить на 2 доли так, чтобы с вершиной в долю попало не более 1 врага.  $\mathcal{O}(V+E)$ .

## 10. Транзитивное замыкание

Дан орграф, построить матрицу достижимости.  $V, E \leq 10^5$ .

## 11. (\*) Четный цикл

За  $\mathcal{O}(V+E)$  найти в неорграфе цикл четной длины.

## 12. (\*) Максимальное число различных циклов

Есть неорграф, нужно выбрать max число простых циклов так, чтобы каждый следующий содержал хотя бы одно новое ребро.  $\mathcal{O}(E)$ .

## Разбор задач практики

### 1. Достижимость множеств

dfs из всех вершин  $A$ , проверить, что посещена хотя бы одна из  $B$ .

### 2. Циклы

а) **Цикл в орграфе через ребро.**  $e = (v, u)$ , ищем путь из  $u$  в  $v$ .

б) **Цикл в орграфе через вершину.** Запускаем dfs из  $v$ , цикл есть  $\Leftrightarrow$  мы нашли в процессе dfs ребро обратно в  $v$ .

Способ #2. Удалим вершину, найдем путь из ее исходящих соседей в ее входящих соседей.

Способ #3. Раздвоим вершину на  $v_{in}$  и  $v_{out}$ , все исходящие ребра проведем из  $v_{out}$ , все входящие в  $v_{in}$ , найдем путь из  $v_{out}$  в  $v_{in}$ .

в) **В неорграфе какой-нибудь цикл.**  $\text{dfs}(v, \text{parent})$ , если нашли ребро в посещенную вершину, не равную  $\text{parent}$ , то цикл. Когда в  $\text{dfs}(v, \text{parent})$  нужно пойти в вершину  $u$ , запускаем  $\text{dfs}(u, v)$ .

г) **Цикл в неорграфе через ребро.**  $e = (v, u)$ , ищем путь из  $u$  в  $v$ , не проходя по  $e$ . Это просто запуск  $\text{dfs}(u, \text{parent}=v)$ .

д) **Цикл в неорграфе через вершину.** Так же, как в орграфе, но запрещаем в dfs ходить в предков.

е) **В неорграфе какой-нибудь цикл за  $\mathcal{O}(V)$ .** В графе с  $V$  ребрами всегда есть цикл, считаем только первые  $V$  ребер.

Но можно заметить, что стандартный способ с dfs тоже работает за  $\mathcal{O}(V)$ . Пока мы не нашли цикл, все посещенные dfs-ом ребра образуют дерево, итого мы успеем посмотреть как раз только на  $V$  ребер.

### 3. Гамильтонов путь

Делаем topsort, проверяем, что из каждой вершины есть ребро в следующую.

### 4. Единственность топсорта

Делаем topsort, проверяем, что из каждой вершины есть ребро в следующую  $=)$

И правда, если между какими-то соседями нет ребра, их можно поменять местами.

### 5. Ациклическая ориентация

Ориентируем ребра от меньшей вершины к большей.

Способ #2. Запустим dfs, ориентируем все ребра вниз, то есть ребра дерева и обратные из предка в потомка.

В обоих случаях цикла нет, ибо на любом пути либо строго растет номер вершины, либо глубина в дереве dfs.

### 6. Сильносвязная ориентация

Ориентируем ребра так, как на них впервые посмотрит dfs.

Получится, что ребра дерева dfs будут ориентированы вниз, а обратные вверх.

Из корня достижимо всё. Откуда угодно можно попасть в корень, потому что откуда угодно можно подняться выше: если нельзя, то между поддеревом вершины и тем, что над ней, есть только одно ребро, тогда ориентация была исходно невозможна.

### 7. Нечетный цикл

Красим граф в два цвета. Если видим ребро, ведущее из текущую вершину в уже

покрашенной в тот же цвет – нашли нечетный цикл, он лежит на стеке рекурсии.  
Если нет такой ситуации, покрасили в два цвета, нет нечетного цикла.

### 8. Разделение на две клики

Инвертируем все ребра: если между парой вершин нет ребра, добавим, иначе уберем. Теперь надо разбить на два независимых множества, то есть на две доли, то есть покрасить в два цвета.

*Важное замечание:* это работает за  $\mathcal{O}(V^2)$ , в инвертированном графе ребер  $V^2 - E$ .

### 9. Разбиение на дружелюбные доли

Разобьем как-нибудь. Метод локальных оптимизаций: если у какой-то вершины в её доле 2 врага, перекинем её в другую долю.

Каждый раз уменьшается суммарное число рёбер-врагов внутри долей, поэтому надо  $\mathcal{O}(E)$  шагов.

Чтобы быстро находить плохие вершины, сделаем это dfs-ом.

```
void dfs( int v ) {
    if (bad(v)) {
        color[v] ^= 1;
        for (int x : g[v]) dfs(x);
    }
}
for (int v = 0; v < n; v++)
    dfs(v);
```

Заметим, что тут нет пометок посещенных вершин, но рекурсивные вызовы возникают только после перекидывания вершины, поэтому время  $\mathcal{O}(V + E)$ .

### 10. Транзитивное замыкание

Сконденсируем граф. Из каждой вершины достижима ее ксс, нужно узнать, что еще.

Храним bitset вершин, достижимых из данной. Делаем dfs, ответ для вершины – OR bitset'ов ее детей.

### 11. Чётный цикл

Мы умеем разбивать кактус на циклы. Если наш граф – кактус, алгоритм успешно отработает за  $\mathcal{O}(V + E)$  и нам останется проверить чётности найденных циклов. Если наш граф не кактус, то алгоритм найдёт в какой-то момент два пересекающихся по ребру цикла. Заметим, что или один из этих двух циклов чётный, или они оба нечётные, но благодаря их пересечению мы можем выделить чётный.

### 12. Максимально число различных циклов

Рассмотрим все циклы, образованные обратными ребрами dfs.

# Домашнее задание

## 3.1. Обязательная часть

### 1. (2) Покраска в 4 цвета

За  $\mathcal{O}^*(2^n)$  проверить, можно ли покрасить граф в 4 цвета.

*Подсказка: в два цвета красить очень просто.*

### 2. (2) Минимизация максимального ребра на цикле

Дан неорграф с целыми положительными весами на рёбрах. Найти в неорграфе такой цикл, что максимальный вес ребра этого цикла минимален.  $\mathcal{O}((V+E) \log E)$ .

*Решить, используя только известное нам.*

### 3. (2) Разноцветные двери

Есть комнаты и двери между комнатами. Нужно каждую дверь покрасить с одной стороны в зеленый цвет, с другой в оранжевый цвет так, чтобы для каждой комнаты количества зеленых и оранжевых дверей в комнате отличались не более чем на один.

Решить за время  $\mathcal{O}(\text{poly}(\log(V, E)))$ , (+1)  $\mathcal{O}(V+E)$ .

### 4. (2) Ещё один вариант разбиения на две доли

Разбить множество вершин неорграфа на две доли так, чтобы у каждой вершины был сосед в другой доле.  $\mathcal{O}(V+E)$ .

### 5. (3) Ромбики

Найти в неорграфе “ромбик с диагональю”: 4 вершины  $a, b, c, d$  с 5 рёбрами  $ab, bc, cd, da, ac$ .

Оценка: (1)  $\mathcal{O}(V^3)$ , (1)  $\mathcal{O}(VE)$ , (1)  $\mathcal{O}(\frac{VE}{w})$ , (+1)  $\mathcal{O}(E^{3/2})$ .

### 6. (3) Покраска в 3 цвета

Дан изначально пустой неорграф. В него добавляются рёбра и вершины. При этом поддерживается инвариант, что степени всех вершин не более 5. Поддерживать покраску вершин в 3 цвета так, чтобы у каждой было не более одного соседа того же цвета. При добавлении новых ребра/вершины обновлять покраску за амортизированное  $\mathcal{O}(1)$ .

## 3.2. Дополнительная часть

### 1. (3) Минимизация максимальной исходящей степени

Дан неорграф. Ориентировать его так, что максимальная исходящая степень была минимальна.  $\mathcal{O}(E^2)$ . Можно еще на полилог.

### 2. (3) Сильная связность

Сильная связность (задача с доп практики). Решение за  $\mathcal{O}(V+E)$ : построим конденсацию, выделим стоки и истоки. Затем найдем за  $\mathcal{O}(V+E)$  максимальное по включению множество непересекающихся по вершинам путей из истоков в стоки. То есть, не существует еще одного пути из истоков в стоки, не пересекающегося с уже выбранным. Пусть  $i$ -й путь начинается в  $a_i$ , заканчивается в  $b_i$ , всего путей  $k$ . Добавим ребра  $(b_1, a_2), (b_2, a_3), \dots, (b_n, a_1)$ . Это неполное описание решения. Задача в том, чтобы довести его до конца, доказать корректность, доказать линейность времени работы.

### 3. (3) Существование пути

Напишите детерминированное решение, которое имеет доступ к оракулу  $g(i, j)$  – есть ли ребро между  $i$  и  $j$ , и должно, используя полилогарифм от  $n$  памяти проверить наличие пути из  $a$  в  $b$  в графе из  $n$  вершин.