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

ОСНОВЫ КОМБИНАТОРИКИ И ТЕОРИИ ЧИСЕЛ

II СЕМЕСТР

Лектор: *Райгородский*

h\nu

Автор: *Киселев Николай*
Репозиторий на Github

весна 2025

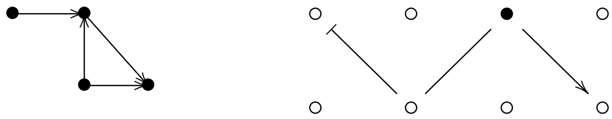
Содержание

«««< HEAD

1 Графы

Определение 1.1. Ориентированный граф $G = (V, E)$, где V - конечное множество. $E \subset V \times V$

Определение 1.2. Неориентированный граф $G = (V, E)$, где V - конечное множество. $E \subset C_v^2$



1.1 Алгоритм dfs (поиск в глубину)

Псевдокод:

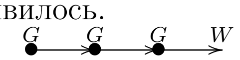
```
vector<vector<int>> g;

vector<int> parent;

vector<int> tin, tout; // время входа и выхода из вершины

vector<string> color; // для покраски вершин
```

Изначально все вершины покрашены в белый цвет - сигнал, что в вершину еще не заходили, цвет серый - вершина в обработке, цвет черный - вершина полностью обработана, больше нас не интересует

```
void dfs(int v) {
  color[v] = "GRAY"; tin[v] = timer; ++timer;
  for (int to: g[v]) {
    if (color[to] == "WHITE"): parent[to] = v; dfs(to);
  }
  tout[v] = timer; ++timer;
  color[v] = "BLACK";
}
```

Лемма 1.1 (О белых путях)**.** *За время с $tin[v]$ до $tout[v]$ dfs посетит все те вершины, которые были достижимы из v по белым путям и перекрасит их в черный цвет.*

Доказательство. Понятно, что перекрасить можем только описанные вершины. Заметим, что GRAY вершины - это в точности стек рекурсии. Значит, в момент $tout[v]$ новых серых не появилось.



Остается доказать, что белые вершины достижимы по белым путям и не могут остаться белыми. Рассотрим вершину u - самую высокую оставшуюся из белых. Тогда ее родитель не мог почернеть без захода в эту вершину. $\square$

Следствие. Пусть изначально все вершины - белые. Тогда после внешнего запуска dfs(s) посетятся все достижимые из s вершины.

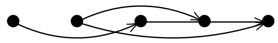
Следствие. В графе $\exists$ цикл, достижимый из $s \leftrightarrow dfs(s)$ в какой-то момент ведет ребро в серую вершину.

Замечание. Мы не пытаемся обойтись 2 цветами - черным и белым, чтобы иметь возможность понять, есть ли цикл в графе

Замечание. Асимптотика алгоритма равно $O(n+m)$, где $n = |V|, m = |E|$.

Определение 1.3. DAG(directed acyclic graph) - ориентированный граф без циклов.

Определение 1.4. Топологическая сортировка графа: перестановка вершин графа, чтобы все ребра вели "слева направо".



Утверждение 1.1. *Топологическая сортировка сущесутвует тогда и только тогда, когда граф - DAG*

Доказательство.

$\rightarrow$ Очев

$\leftarrow$ Алгоритмом: все вершины красим в белый цвет.

```
for (s = 0...n-1)
  if (color[s] == "WHITE") dfs(s)
```

Топологическая сортировка - перестановка вершин в порядке убывания $tout$

Проверим корректность: Достаточно показать, что не может быть ребра из u в v: $tout[u] < tout[v]$. Предположим противное и разберем 2 случая:

(a) $tin[u] < tin[v]$

По лемме о белых путях, к моменту времени выхода из u вершина v уже полностью обработается $\rightarrow tout[v] < tout[u]$. Противоречие.

(b) $tin[v] < tin[u]$ Тогда $\nexists$ пути из v в u. Значит, пол лемме, к моменту $tout[v]$ мы даже не увидим u. А следовательно, $tout[v] < tout[u]$. Противоречие.

□

Определение 1.5. Пусть G - ориентированный граф, $u, v \in V(G)$. Тогда говорим, что u, v сильно связны, если $\exists$ пусть из u в v и из v в u.

Задача. *Сильная связанность - отношение эквивалентности.*

Определение 1.6. Класс эквивалентности по этому отношению - компонента сильной связности (КСС)

1.1.1 Алгоритм Косарайю

Алгоритм выделения КСС за $O(n + m)$

1. dfs от всех вершин, сортируя все вершины в порядке убывания tout
2. В этом порядке запускаем dfs по обратным ребрам (dfs Reversed). Все, что посетим за один такой запуск - очередная КСС.

Корректность?

Доказательство. Ясно, что каждый запуск dfs Reversed обойдет одну или несколько КСС целиком. Но вдруг мы возьмем 2 КСС вместо одной...

Утверждение 1.2.
Пусть C_1, C_2 - две КСС, причем есть ребро из C_1 в C_2. Тогда $max_{x \in C_1}(tout(x)) > max_{y \in C_2}(tout(y))$

Доказательство.

1. $min_{a \in C_1} tin(a) < min_{b \in C_2} tin(b)$

   В этом случае к моменту времени входа в a все вершины в C_1 и C_2 - еще белые. По лемме о белых путях к моменту $tout[a]$ все вершины из C_1 и C_2 покрасятся в черный $\implies tout(a) > max_{y \in C_2}(tout(y))$.

2. $min_{a \in C_1} tin(a) > min_{b \in C_2} tin(b)$

   Тогда к моменту входа в b все вершины из C_1 и C_2 еще белые. Отметим, что не существует пути из b в C_1 (иначе C_1 и C_2 - одна КСС). Значит, к моменту выхода из b вся C_1 еще белая $\implies max_{x \in C_1}(tout(x)) > max_{y \in C_2}(tout(y))$

□

Теперь воспользуемя утверждением и получим искомое. □

Замечание. Пусть алгоритм Косарайю нумерует все КСС в порядке их обнаружения. $id[v]$ - номер КСС, содержащий v. Значит, если есть ребро из a в b, $id[a] \leqslant id[b]$

a, b *сильно связаны* $\implies id[a] = id[b]$