Студент: Александр Никулин
Дата: 26 октября 2020 г.

**Задача 1.** *Найти в неорграфе простой цикл через данную вершину за $\mathcal{O}(E)$*

*Решение.* Простой цикл - цепь, в которой первая и последняя вершина совпадают. Тогда кажется логичным попробовать найти цикл проходящий через данную вершину с помощью поиска в глубину. Если в процессе обхода мы встретим переход в изначальную вершину, когда она еще серая, то мы встретили цикл. За исключением случаев, когда мы попытались вернуться в вершину, которая является родителем в dfs дереве, например $u->v->u$, который по определению простым циклом не является.

Осталось доказать, что если цикл есть, то поиск в глубину его найдет. Простым он окажется просто по определению поиска в глубину - мы не можем встретить ребра или вершины, по которым уже один раз прошли т.к. они станут черными, а значит мы больше никогда через них не пройдем (т.к. ходим только по белым).

Пусть $v$ - вершина, из которой мы начали поиск в глубину. Тогда существует вершина $u$ которая входит в цикл и из которой есть ребро в $v$. Так как (опять же по определению цикла) между $u$ и $v$ существует белый путь, то в момент выполнения $dfs(u)$ вершина $v$ будет серой, а значит обход попытается перейти в вершину v (она не является родителем, в нее есть ребро) и обнаружит цикл.

**Задача 2.** *Дано дерево $T = \langle V, E\rangle$. За $\mathcal{O}(V + E)$ вычислить для каждого ребра, сколько простых путей проходит через него.*

*Решение.* В дереве между любыми двумя вершинами существует только один путь, он же является и простым. Рассмотрим ребро $(v, u)$. Данное ребро разбивает дерево на два и все пути которые проходят через него соединяют вершины из одной части с вершинами в другой. Т.к. каждое соединяет с каждым это можно выразить как произведение количества вершин в первой части на количество вершин во второй. Пусть $dp[v]$ - количество вершин в поддереве, корнем которого является $v$. Тогда справедливо такое равенство:

$$\text{paths}(v, u) = dp[v] \cdot (|V| - dp[v])$$

Посчитать dp можно рекурсивно, выбрав какую-то вершину корнем, а потом вызваться от ее детей, сложив итоговый результат (т.к. $dp[v] = \sum_{u\in\Gamma(v)} dp[u]$). База рекурсии - лист, для него $dp[l] = 1$.

```
    def dfs(root):
        visited[root] = True

        if not adj[root]:
            dp[root] = 1
        else:
            total = 0
            for neigh in adj[root]:
                if not visited[neigh]:
                    total += dfs(neigh)

            dp[root] = total

        return dp[root]
```

.

Осталось для каждого ребра посчитать количество путей проходящих через него по формуле. Т.к. мы должны посчитать что-то для каждого ребра, а в dfs еще пройти по каждой вершине (по ребрам конечно же тоже), то асимпототику можно оценить как $\mathcal{O}(V + E)$.

**Задача 3.** *Найдите лексикографически минимальный из всех топологических порядков. $V, E \le 10^6$.*

*Решение.* Лексикографически минимальный топологический порядок - порядок, в котором вершины у которых нет входящих ребер (и которые соответственно уже можно поставить в начало) идут в порядке номера id. Чтобы добиться этого достаточно в алгоритме топологической сортировки заменить

обычную очередь на очередь с приоритетом. Также следует пронумеровать вершины натуральными числами / буквами, если это уже не сделано (иначе отсутствует приоритет).

Алгоритм будет следущим. В очередь добавляются вершины с нулевой степенью, после чего извлекаются вершины с минимальным номером. Для их соседей уменьшается количество входящих ребер, если оно зануляется, то они также добавляются в очередь. Алгоритм остается корректен, потому что мы не нарушем основных условий - все вершины идут в порядке ребер от меньших номеров к большим. Меняется лишь порядок вершин, для которых есть возможность поменяться местами без нарушения топсорта. За счет того, что мы ставим вершину с минимальным id в первой уж точно не нарушится условие ребер от меньших номеров к большим (если не нарушалось до этого).