Динамическое транзитивное замыкание

Кузнецов Илья Александрович
371 группа

18.05.2022

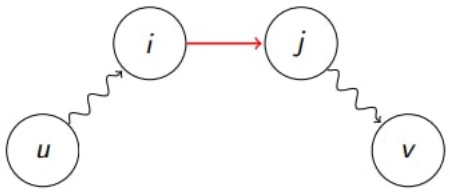
1. На лекции мы научились поддерживать инкрементальное транзитивное замыкание ориентированных графов. Придумайте алгоритм декрементального транзитивного замыкания, работающий за $O(n^2(m+n))$ суммарно за все обновления.

Решение. Инициализация: строим начальную матрицу достижимости M (выполняем транзитивное замыкание), а также списки смежности для исходного графа и инвертированного графа (направление ребер в другую сторону).

При реализации удаления ребра (i, j) нас будут интересовать все такие вершины v, которые стали недостижимы из i после удаления ребра, но которые были достижимы из j до удаления. Для этого используем поиск в глубину (DFS).

Из всех найденных вершин v запустим DFS на графе с обратными ребрами. Так мы определим достижимость каждой вершины u из v. Для всех вершин u, не достижимых из v, необходимо показать это в матрице достижимости $M[u, v] = 0$.

Рассмотрим алгоритм на примере следующего графа.



Если удалить ребро (i, j) в данном графе, то вершиной, которая стала недостижима из i, но была достижима из j, является вершина v. После запуска DFS на ней в инвертированном графе недостижимыми вершинами будут u и i, поэтому в матрице достижимости нужно будет показать $M[u, v] = 0$ и $M[i, v] = 0$.

Algorithm 1 Декрементальное обновление матрицы достижимости

```
1: function REMOVE(i, j)
2:     adj[i].remove(j)
3:     adj_rev[j].remove(i)
4:     res ← dfs(adj, i)
5:     for v : res[v] = 0 ∧ M[j, v] = 1 do
6:         res_rev ← dfs(adj_rev, v)
7:         for u ∈ V do
8:             M[u, v] ← res_rev[u]
```

Доказательство. Так как алгоритм декрементальный, то если $M[i, j]$ однажды стало 0, значит оно никогда не станет 1.

Оценим сложность алгоритма. Сложность DFS — $O(n+m)$ времени, а максимальное число вызовов функции на все обновления равно m, где m — число ребер (если удалятся все ребра в графе).

Таким образом, суммарная сложность на все обновления составляет $O(m(n+m+n)$ времени. Если учесть оценку, что $m <= n^2$, то итоговая сложность равна $O(n^2(m+n))$ времени. $\square$