Динамическое транзитивное замыкание

Иванов Кирилл
371 группа

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

Решение. При удалении нас будут интересовать вершины u, которые недостижимы из i после удаления ребра (i, j) и которые были достижимы из j до удаления ($M[j, u] = 1$). Из каждой такой вершины u запустим поиск в глубину в графе с инвертированными ребрами, при помощи которого определим достижимость каждой вершины v из u. Заметим, что достижимость в инвертированном графе эквивалентна достижимости в исходном: $\exists v \pi u = (v, w_1), (w_1, w_2), \dots, (w_n, u) \Leftrightarrow \exists u \pi^{-1} v = (u, w_n), (w_{n-1}, w_{n-2}), \dots, (w_1, v)$.

Algorithm 2 Декрементальное обновление матрицы достижимости

```
1: function REMOVE(i, j)
2:     g[i].remove(j)
3:     g_rev[j].remove(i)
4:     dfs(i, g)                                  ▷ Заполняем used
5:     for u : ¬used[u] ∧ M[j, u] = 1 do
6:         dfs(u, g_rev)                          ▷ Заполняем used_rev
7:         for v ∈ V do
8:             M[v, u] ← used_rev[v]
```

Анализ сложности.

- Поскольку алгоритм декрементальный, если ячейка матрицы M стала 0, то она никогда не станет 1.
- Если условие на строке 5 не выполнено ни для одной вершины, то строчки 6–8 не будут исполнены. Если же найдется вершина u, удовлетворяющая этому словию, то как минимум $M[i, u]$ станет нулем – такое может произойти не более n^2 раз из-за декрементальности и условия $M[j, u] = 1$ на строке 5, то есть строчки 6–8 могут быть вызваны не более n^2 раз на все апдейты. Отсюда получаем стоимость строчек 5-8 на все апдейты – $O(n^2(T_{dfs}(n, m) + n)) = O(n^2(n + m))$.
- Строки 2–4 стоят $O(n^2\ T_{dfs}(n, m)) = O(n^2(n + m))$
- Итого: суммарное время на все апдейты – $O(n^2(n + m))$