Седьмое задание по алгоритмам

Ситдиков Руслан

2 апреля 2018 г.

Задача 1

Пусть наш ориентированный граф **not DAG**, то есть три или более вершин составляют в нем хотя бы один цикл. Тогда найдутся хотя бы две вершины, такие что они составляют цикл в графе, то есть из вершины x_1 есть ориентированное ребро в x_2, а из x_2 есть ориентированное ребро в x_1. Условие наличия ориентированного ребра из вершины x_i в x_j $(x_i > x_j) \Rightarrow$ Нам осталось провести ориентированное ребро из $x_j \to x_i$, для этого должно выполняться неравенство $x_j > x_i$, но по условию $x_i > x_j$, следовательно, наш граф ациклический следует, что мы имеем **DAG**.

Задача 2

База индукции
Проверим для n = 2 вершин:

Имеем $v_1 \to v_2$ простой путь состоит из одного ребра, то есть выполняется $n - 1 = 2 - 1 = 1$.

Предположение индукции
Пусть верно для $n = k$ вершин:

То есть мы имеем для k вершин турнира простой путь $k - 1$.

Доказательство для $k + 1$
Мы добавляем еще одну вершину в наш граф. Пусть для определенности наш путь будет занумерован от v_1 по v_k.
1)Если выполняется $v_k \to v_{k+1}$, то наш путь $v_1, ..., v_{k+1}$ искомый.
2)Если не выполняется $v_k \to v_{k+1}$,то пусть $i \in \{0, \ldots, n\}$ — максимальное число такое, что для любого $j \leq i$ имеется дуга из v_j в v_{k+1}. Тогда

$$v_1, \ldots, v_i, v_{k+1}, v_{i+1}, \ldots, v_k$$

— искомый ориентированный путь.

Алгоритм
Возьмем любую вершину нашего графа. Возьмем связанную с ней вторую вершину.
Эти две вершины составляют турнир. Пусть w_i будет i вершиной нашего простого пути.
Возьмем следующую вершину вершину:

1)Если из нее исходят ориентированные ребра$(v_j \to w_i, v_j \to w_k....)$ в наш составленный путь, то v_j вставляем в начало нашего пути.

2)Если в нее исходят ориентированные ребра$(w_i \to v_j ... w_k \to v_j)$ из нашего турнира,то v_j встает в конце пути.
3)Если ориентированные ребра как исходят из новой вершины, так и

заходят в нее, то мы вставляем нашу вершину между двумя в нашем пути так, что $w_i \to v_j \to w_{i+1}$ при первом же выполнении этого условия.

Корректность

Каждый раз мы добавляем по одной вершине так, что наш путь лишь дополняется новыми вершинами без нарушения условия его создания.

Асимптотика

В худшем случае нам придется находить вершины, между которыми можно вставить новую вершину. Итого:

$$c + 2c + 3c + 4c + ... + nc = O(v^2)$$

-где v - кол-во вершин в графе.

Задача 3

а)Пусть вершины графа отмечены числами их открытия в массиве открытия.

Если вершины соединены в графе ребром так, что вершины(по описанию, одна из них предок, а другая потомок) удовлетворяют условиям $d[i] < d[j]$ и $f[j] < f[i]$,а также $j > i + 1$, то это ребро будет прямым.

б)Пусть вершины графа отмечены числами их открытия в массиве открытия.

Если вершины соединены в графе(по описанию) и в вершину, которая закрылась раньше $f[i] < d[j]$,чем вторая , направлено из второй ориентированное ребро, то это ребро будет перекрестным.

Задача 4

Алгоритм

Для того чтобы разделить наш ограф на области,нужно найти в нем такие компоненты сильной связности, что можно разделить наш граф только этими компонентами сильной связности.

Компоненты сильной связности в графе G можно найти с помощью поиска в глубину в 3 этапа:

1)Построить граф H с обратными (инвертированными) рёбрами.

2)Выполнить в H поиск в глубину и найти f[u] — время окончания обработки вершины u.

3)Выполнить поиск в глубину в G, перебирая вершины во внешнем цикле в порядке убывания f[u].

Корректность

Находя цикл в оргафе(то есть компоненты сильной связности),мы можем заменить его на одну вершину так,чтобы не нарушалась структура исходного графа. То есть заменяя все наши циклы на новые вершины,мы получаем орграф из новых вершин.

Асимптотика

Для того, чтобы инвертировать все ребра в графе, представленном в виде списка потребуется O(V+E) действий. Для матричного представления графа не нужно выполнять никакие действия для его инвертирования. Количество ребер в инвертированном равно количеству ребер в изначальном графе, поэтому поиск в глубину будет работать за O(V+E) Поиск в глубину в исходном графе выполняется за O(V+E). В итоге получаем, что время работы алгоритма O(V+E).

Задача 5

Алгоритм

Возьмем алгоритм обхода в ширину нашего лабиринта. Проходя через каждую вершину, мы оставляем в ней монетку и рассматриваем все уровни(вершины, которые соединены с той, на которой мы находимся в данный момент).

Корректность

Запуская наш алгоритм, мы проходимся по всем вершинам в худшем случае. То есть мы точно должны наткнуться на выход. Для того,чтобы повтороно не проходить через те же вершины, мы помечаем их монетками.

Асимптотика

Пусть из каждой комнаты в другую ровно по одному коридору, тогда общее кол-во вершин(комнат) не превосходит $\frac{m}{2}+1 \Rightarrow$ асимптотика нашего алгоритма будет $O(m)$