Практика по алгоритмам #12
Кратчайшие пути

---

1. Новые задачи

1. За $\mathcal{O}(n+m)$ найти в неорграфе цикл нечетной длины.
2. Есть ориентированный граф. Для каждой пары вершин a, b определена функция расстояния $f(a,b)$. Вася и Петя стоят в вершинах v и p, соответственно, и хотят поменяться местам, не оказываясь ни в какой момент времени ближе чем на расстоянии d. За какое минимальное число ходов они могут это сделать? Ход – один из них переходит в смежную вершину. $\mathcal{O}(nm)$.
3. Дан взвешенный орграф с положительными весами и выделенная вершина s, нужно для каждой вершины v найти число кратчайших путей из v в s. $\mathcal{O}(m \log n)$.
4. Дан взвешенный орграф с положительными весами. Найти кратчайший путь, проходящий по всем $k \le 10$ выделенным вершинам.
5. В графе почти все ребра имеют неотрицательные веса. Все кроме ребер смежных с s, t, x (какая-то третья вершина). Найдите кратчайший путь из s в t за $\mathcal{O}(m \log n)$.
6. Даны две паралелльных прямых (река). В реке есть n островов (точек). Нужно провести по реке круглый корабль максимального радиуса R так, чтобы он не задел ни одного осторова. Найти R за $\mathcal{O}(n^2 \log n)$.
7. Есть ограф, на ребрах этого орграфа написаны пары положительных чисел $\langle a, b\rangle$. Нужно найти путь $path$ из s в t такой, что $\sum\limits_{e \in path} a_e \le A$, а $\sum\limits_{e \in path} b_e \to \min$. Решение за $\mathcal{O}(n+m \cdot A)$.
8. Есть взвешенный неограф. Найти путь из s в t такой, что сумма двух максимальных ребер на пути минимальна. $\mathcal{O}((n+m) \cdot poly(log))$.
9. Нужно научиться на запрос “уменьшился вес ребра” за $\mathcal{O}(n^2)$ пересчитывать матрицу расстояний.
10. Есть n валют и m обменников. i-й обменник предлагает менять валюту a_i на валюту b_i по курсу c_i/d_i. Можно ли, используя сколь угодно большие начальные сбережения и данные m обменников, сломать финансовую систему и бесконечно обогащаться? Считается, что у обменников есть бесконечное количество денег целевой валюты. $\mathcal{O}(nm)$.
11. Есть ограф. Нужно найти количество необязательно простых путей из a в b
    a) Длины ровно k, здесь $k \le 10^9$.
    b) Длины не более k, здесь $k \le 10^9$.
    c) Длины от l до r, здесь $l \le r \le 10^9$.

2. Разобранное в классе

1. Запускаем `dfs`-ом проверку графа на двудольность. В тот момент, когда `dfs` определил, что граф не двудольный, последнее рассмотренное ребро вместе с деревом обхода `dfs`-а образуют цикл нечетной длины.
2. Строим новый граф. Вершина нового графа – пара вершин исходного графа (a, b), вершина валидна, если $dist(a, b) < d$. Ребро нового графа $(a, b) \to (c, b); (a, d)$, где ребра $a \to c$ и $b \to d$ присутствуют в исходном графе. Решение – запустить `bfs` по валидным вершинам нового графа.
3. Запускаем Дейкстру, оставляем только ориентированные ребра $(a, b, w)\colon d[b] = d[a] + w[e]$. Получили так называемый "граф кратчайших путей". Поскольку $\forall e\colon w[e] > 0$, граф кратчайших путей ацикличен. Посчитаем на нем динамикой количество путей из a в b.
4. Запустим k раз Дейкстру, таким образом мы предподсчитали расстояния между всеми парами рассматриваемых k вершин. Теперь считаем динамику $f[A, v]$ – длина кратчайшего пути, который проходит по всем вершинам из множества A и заканчивается в вершине v. Время работы алгоритма $\mathcal{O}(k \cdot Dijkstra + 2^k k^2)$.
5. Изначально все $d[v]$ равны +∞, затем для каждого ребра $s \to v_i$ веса w_i пишем $d[v_i] = \min(d[v_i], w_i)$. Запускаем Дейкстру, по всему графу без вершины t.
   В конце пишем $d[t] = \min\limits_{e=(v,t,w)} (d[v] + w)$.
6. Построим граф с $n + 2$ вершинами – два берега и острова. Расстояние между вершинами – евклидово расстояние между объектами. Максимальный радиус R равен длине кратчайшего пути от одного берега до другого, где длина пути – минимум весов ребер на пути. Возможные решения для поиска пути: Дейкстра, бинарный поиск по ответу + `dfs`, сортировка ребер + `СНМ`.
7. [?] (дана в ДЗ второй группе)
8. Запустим Дейкстру из s, получим $d_s[v]$. Запустим Дейкстру по обратным ребрам из t, получим $d_t[v]$. В обоих запусках расстояние до вершины – максимум на пути. Теперь переберем максимальное ребро (u, v) на пути, который мы ищем. Если $\max(d_s[v], d_t[u]) \le w(u, v)$, то улучшаем ответ до $\max(d_s[v], d_t[u]) + w(u, v)$.
9. Уменьшим вес ориентированного ребра (a, b) до $x\colon w[a, b] = x \quad \Rightarrow$
   $d[x, y] = \min(d[x, y], d[x, a] + w[a, b] + d[b, y])$.
10. Скажем, что обменник (a_i, b_i, c_i, d_i) – это ребро, соединяющее вершины a_i и b_i), а вес этого ребра равен $\log \frac{c_i}{d_i}$. Если в данном графе есть цикл положительного веса, мы можем бесконечно обогащаться. Вес пути/цикла сейчас равен сумме весов ребер. Можно было не переходить к логарифмам, а использовать вес пути/цикла равный произведению весов $\frac{c_i}{d_i}$. Алгоритмы поиска цикла положительного веса и цикла отрицательного веса получаются друг из друга домножением весов всех ребер на -1.

3. Домашнее задание

3.1. Обязательная часть

1. (1) Дан невзвешенный неорграф и два множества вершин – A и B, $A \cap B = \emptyset$. Найти длину кратчайшего пути из A в B и количество кратчайших путей из A в B. $\mathcal{O}(n+m)$. (1доп) Что если $A = B$? Нас не интересуют пути из вершины в себя.
2. (2) Два коня на шахматной доске $n \times n$ хотят поменяться местами за минимальное количество ходов. Один ход – прыжок одного коня. Кони не могут одновременно стоять в одной клетке. По некоторым клеткам шахматной доски можно ходить, по некоторым – нет.
   a) (1 балл) Решение за $n^4 \cdot 16$.
   b) (1 балл) Ускорим решение в 8 раз.
   c) (1доп) Попробуем ускорить решение еще в 2 раза. Вдруг получится?
3. (2) Дан взвешенный орграф с неотрицательными весами и выделенная вершина s, нужно для каждой вершины v проверить, есть ли хотя бы два различных кратчайших пути из s в v. $\mathcal{O}(m \log n)$.
4. (2) Найти в орграфе кол-во необязательно простых путей из a в b длины ровно $k \le 10^9$.
5. (2) Предподсчет за $\mathcal{O}(n^3)$ и запрос $\langle a, b, e\rangle$ за $\mathcal{O}(1)$ – существует ли кратчайший путь из a в b, проходящий через ребро e?
6. (2) Докажите, что если мы за $T(n)$ умеем пересчитывать матрицу расстояний после удаления одного ребра в орграфе с положительными весами, то мы умеем насчитать матрицу расстояний с нуля за $\mathcal{O}(T(n) \cdot \log n)$. Формально: есть алгоритм, который на вход получает орграф, матрицу расстояний, ребро, которое нужно удалить, возвращает новую матрицу расстояний после удаления. Тогда есть алгоритм, который работает не более чем в $\log n$ раз дольше, на вход получает граф, на выход дает матрицу расстояний.
7. (3) Рассмотрим алгоритм поиска кратчайшего пути в орграфе с отрицательными весами "Форд-Беллман с кучей". Алгоритм: на каждом шаге есть куча вершин с ключом "расстояние до вершины", вынимаем из кучи вершину v с минимальным расстоянием и релаксируем ребра из v, при этом все вершины, до которых улучшилось расстояние, добавляются в кучу. В среднем случае описанный алгоритм ведет себя лучше, чем обычный алгоритм Форда-Беллмана. Постройте тест, на котором "Форд-Беллман с кучей" работает экспоненциально долго. Дополнительный балл можно получить за ацикличный пример.

3.2. Дополнительная часть

1. (5) Есть взвешенный неорграф. Найти путь из s в t такой, что сумма трёх максимальных ребер на пути минимальна. $\mathcal{O}((n+m) \cdot poly(log))$.
2. (5) $\texttt{Get}(s)$ – минимальная длина пути из a в b на графе, с весами ребер $cost_e = \min(s, w_e)$. $w_e, s > 0$, w_e фиксированы. Дан граф, нужно научиться быстрее чем "m раз запустить Дейкстру" обаработать m запросов $\texttt{Get}(s)$.
3. (3) Постройте тест, на котором Форд-Беллман, всегда кладущий вершину, в которую мы пришли по отрицательному ребру, в начало очереди, работает экспоненциально долго.
4. (3) Постройте матрицу $n \times n$, состоящую из стенок и пустых клеток, на которой при выполнении bfs-а из правого-нижнего угла в левый-верхний размер очереди будет $\omega(n)$. Ходить можно между вершинами смежными по стороне.