Лекция 5

Поиск кратчайших путей между всеми парами вершин графа

(Конспект: А. Куликов)

5.1 Предисловие: постановка задачи

Наша задача такова. Представим, что нам нужно найти кратчайшие пути между парами вершин связного графа (длины всех ребер мы считаем равными 1, так что расстояние между вершинами - это просто количество ребер на пути из одной в другую).

Как известно, алгоритм Дейкстры работает за время $O(n^3)$. Попытаемся найти алгоритм, работающий быстрее.

Будем искать такую матрицу S, что $S_{i,j}$ — это номер первой вершины на кратчайшем пути из вершины i в j. Ясно, что по такой матрице мы будем знать все кратчайшие пути между парами вершин.

5.2 Шаг первый: поиск матрицы расстояний между вершинами

Найдем матрицу расстояний между всеми парами вершин графа (назовем ее D; $D_{i,j}$ — расстояние между вершинами i и j).

Пусть A — матрица смежности исходного графа (размера $n \times n$). Рассмотрим матрицу A^2. Заметим, что $(A^2)_{i,j} = 0$ тогда и только тогда, когда в нашем графе не существует пути из i в j длины не более 2. Действительно, если в графе нет пути длины не более двух из i в j, то множества $N(i)$ и $N(j)$ (множества соседей вершин i и j) не пересекаются, то есть не существует такого k, что $A_{i,k} = A_{k,j} = 1$ (а именно такое k нужно, чтобы $(A^2)_{i,j} = \sum_k A_{i,k}A_{k,j}$ было отлично от нуля).

Теперь рассмотрим квадрат исходного графа G (назовем его G') — такой граф, в котором ребро между вершинами u и v есть тогда и только тогда, когда в графе G существует путь между этими двумя вершинами длины не более двух, причем $u \neq v$. В матрице A' будут стоять единицы только там, где стояли единицы в матрице A или A^2, а на главной диагонали в любом случае поставим все нули.

Особый случай: граф G' - полный. Ясно, что тогда $D = 2 \cdot A' - A$.

В общем же случае будем искать матрицу D рекурсивно. Пусть D' - матрица расстояний для графа G' (найденная рекурсивным вызовом нашего алгоритма для графа G').

Интуитивно ясно, что расстояния в G примерно в 2 раза больше, чем в G'. Примерно, но не совсем, а именно:

Лемма 5.1. *Если $D_{i,j} \vdots 2$, то $D_{i,j} = 2 \cdot D'_{i,j}$; в противном случае $D_{i,j} = 2 \cdot D'_{i,j} - 1$.*

Доказательство. Очевидно. $\square$

Но этого еще не достаточно для нахождения числа $D_{i,j}$ через $D'_{i,j}$, ведь для этого нам нужно знать четность $D_{i,j}$.

Лемма 5.2. *Пусть k — сосед i ($k \in N(i)$). Тогда, если $D_{i,j} \vdots 2$, то $D'_{k,j} \geq D'_{i,j}$. Если $D_{i,j} \not\vdots 2$, то $D'_{k,j} \leq D'_{i,j}$; более того, в этом случае найдется такое k, что $D'_{k,j} < D'_{i,j}$ (ясно, что этой вершиной будет первая вершина на кратчайшем пути из i в j).*

Рассмотрим матрицу M, равную произведению матриц A и D'. Заметим, что $M_{i,j} = \sum_k A_{i,k}D'_{k,j} = \sum_{k \in N(i)} D'_{k,j}$. Таким образом, если $D_{i,j} \vdots 2$, то $M_{i,j} \geq (A^2)_{i,i} \cdot D_{i,j}$; в противном случае — $M_{i,j} < (A^2)_{i,i} \cdot D_{i,j}$ (заметим, что $(A^2)_{i,i}$ — степень вершины i в графе G).

Итак, проверив полученное неравенство для каждого элемента матрицы, мы найдем D для графа G рекурсивно, зная D' для графа G'. Рекурсия закончится не более, чем за $O(\log n)$ шагов, ибо на каждом шаге рекурсии диаметр графа (наибольшее расстояние между вершинами) сокращается примерно вдвое (а точнее? — упражнение!); в итоге мы придем к нашему особому случаю — графу G диаметра 2 (соответственно, полному графу G'), разобранному отдельно.

5.3 Шаг второй: Поиск “виновников” в умножении булевых матриц

Интуитивное соображение: Для расстояния $D_{i,j}$ “виновник” того, что расстояние между i и j именно такое — это путь длины $D_{i,j}$. Но сейчас мы будем искать “виновников” для другой задачи.

Пусть $A \wedge B = P$, где A, B, P — булевы матрицы. Ясно, что если $P_{i,j} = 1$, то для некоторого k выполняется $A_{i,k} = B_{k,j} = 1$. Такое k назовем *“виновником”*.

Рассмотрим теперь матрицу A^*, такую что $A^*_{i,k} = k \cdot A_{i,k}$. Пусть $P^* = A^* \cdot B$ (внимание: здесь умножение целое, а раньше было $\wedge$ — булево). Тогда $P^*_{i,j}$ — сумма всех виновников. Значит, если существует ровно один виновник, то $P^*_{i,j}$ — виновник.

Факт 5.1. *Пусть среди n шаров имеется w белых и $n-w$ черных. Случайно выбирается r шаров. Тогда если $n/(2w) \le r \le n/w$, то вероятность того, что среди выбранных шаров будет ровно один белый, не меньше $1/(2e)$.*

Рассмотрим теперь случайный вектор $R = (r_1, ..., r_n)$, в котором ровно r координат равны 1. Вычислим матрицы $A^{*,R}$ и B^R следующим образом:

$$A^{*,R}_{i,j} := kA_{i,k}r_k, \quad B^R_{k,j} := B_{k,j}r_k.$$

Вычислим также $C^R := A^{*,R} \cdot B^R$. Заметим, что вероятность того, что $C^R_{i,j}$ — виновник, не меньше $1/(2e)$, если соотношение между r и числом виновников w — такое же, как в Факте 5.1.

Чтобы угадать r, для $r = 1, 2, 4, 8, \dots, 2^{\lfloor \log n \rfloor}$ повторим: выбираем случайно вектор R и проверяем, является ли $C^R_{i,j}$ виновником. Эту процедуру повторим $4 \log n$ раз.

Для данных i и j, оценим вероятность того, что за эти итерации мы найдем виновника. Ясно, что когда-нибудь r попадет в отрезок $[n/(2w); n/w]$, при этом по Факту 5.1 мы найдем виновника с вероятностью, не меньшей $1/(2e)$. После $4 \log n$ таких итераций вероятность того, что мы найдем виновника, не меньше $1 - (1 - 1/(2e))^{4 \log n} \le 1 - 1/n$. Таким образом, мат. ожидание количества ненайденных виновников не превосходит $n^2 \cdot 1/n$, т.е. n. На поиск оставшихся виновников мы затратим время $O(n^2)$ (ненайденных виновников, как мы выяснили, не более n).

На все описанные действия мы затратим время $F(n) = O(T(n) * \log^2 n)$, где $T(n)$ — время, необходимое для умножения двух матриц размера $n \times n$ с числами от 0 до n (ясно, что $F(n)$ больше n^2, но меньше n^3).

5.4 Шаг третий: объединяем все вместе

Покажем, как найти матрицу S. Заметим, что первая вершина на кратчайшем пути из i в j — это такое k, что $D_{i,j} = D_{i,k} + 1$. Ясно, что для любой вершины $k \in N(i)$, выполняется одно из трех следующих неравенств:

$$\begin{aligned} &D_{i,j} = D_{i,k} + 1, \\ &D_{i,j} = D_{i,k}, \\ &D_{i,j} = D_{i,k} - 1. \end{aligned}$$

Итак, нас интересуют такие вершины k, что $D_{i,j} - D_{i,k} = 1 \pmod 3$.

Рассмотрим теперь три матрицы $D^{(0)}$, $D^{(1)}$, $D^{(2)}$, такие что $D^{(s)}_{i,k} = 1$ тогда и только тогда, когда $D_{i,k} = (s-1) \pmod 3$. Теперь для (произведения) булевых матриц A и $D^{(s)}$ мы получим нашим алгоритмом матрицы виновников - назовем их $W^{(s)}$. И тогда матрицу S мы сможем найти следующим образом: $S_{i,j} = W_{i,j}^{(D_{i,j} \mod 3)}$.

Итого, время работы алгоритма: $O(T(n) \cdot \log^2 n)$, где $T(n)$ — время на умножение двух целочисленных матриц порядка $n \times n$, содержащих числа от 0 до n^2 (точнее, это их произведение может содержать такие числа — найдите, в каком именно месте алгоритма?..).