

Квантовая дыра

Автор задачи и разработчик: Владислав Власов

Стандартная идея в задачах, в которых надо «собрать» строку из частей — превратить строку в путь в графе. А именно, создадим граф на 2^k вершинах, соответствующих всем двоичным строкам длины k (идея похожа на граф Де-Брёйна, который часто используется в биоинформатике). После чего соединим ребрами строку s_1 и s_2 , если s_2 можно получить из s_1 , откинув первый символ и дописав новый в конец. Например, при $k = 4$ строки «0010» и «0101» будут соединены ребром, так как имеют общий суффикс-префикс длины $k - 1$ «010».

Заметим, что теперь любая двоичная строка длины $n \geq k$ представляется как путь в этом графе. Отдельно рассмотрим случай $n < k$: в таком случае опасность строки всегда будет равна 0, и можно просто вывести любую битовую строку длины n . А при $n \geq k$ строка t может быть представлена как путь по вершинам $t_{1,\dots,k} \rightarrow t_{2,\dots,k+1} \rightarrow \dots \rightarrow t_{n-k+1,\dots,n}$. При чем каждая встреченная на пути вершина — это подстрока длины k данной строки t , поэтому опасность строки — это просто сумма весов вершин на соответствующем ей пути.

Теперь наша задача — найти самый «дешевый» путь длины $n - k + 1$. Воспользуемся методом динамического программирования, пусть $\text{dp}[i][j]$ — минимальная опасность пути длины i , который заканчивается в вершине j . Такую динамику несложно посчитать и восстановить ответ: $\text{dp}[i + 1][v]$ надо обновить через $\text{dp}[i][u] + d_v$ по всем ребрам $u \rightarrow v$.

Альтернативно можно было заметить, что этот алгоритм очень похож на стандартный алгоритм поиска кратчайшего пути Форда-Беллмана. Однако он ищет кратчайший путь в графе со взвешенными ребрами, а у нас взвешенные вершины. Одно можно свести к другому следующим образом: построим граф на 2^{k-1} вершин вместо 2^k , в котором вершины соответствуют битовым строкам длины $k - 1$. После проведем ребро между двумя вершинами, если у них есть общий суффикс-префикс длины $k - 2$, и при наложении они образуют строку длины 2^k . Тогда ребру назначим вес соответствующей битовой строки длины k .

Асимптотика времени работы такого решения — $\mathcal{O}(n \cdot 2^k)$