Homework

Panov Ivan, M3139

24-11-2019

Задача 1

Если решать данную задачу за $\mathcal{O}(2^n n^3)$, то динамика содержит три состояния: маска посещенных вершин, начальная вершина и последняя вершина. Но можно заметить, что в такой динамике часто встречаются одинаковые циклы(пути где разные концы). Чтобы избежать лишние состояния можно избавиться от одного состояния в динамике - начальная вершина, и за начальную вершину теперь считать вершину с наименьшим номером. Теперь можно зафиксировать начальную вершину и посчитать динамику учитывая, что она первая. Чтобы запуститься от всех нам понадобится n итерацай, так как любая вершина может быть начальной. Но с таким подходом ассимптотика до сих пор $\mathcal{O}(2^n n^3)$. Но можно заметить, что нам нет смысла перебирать все возможные маски вершин, так как мы знаем, что вершины с номером меньшим чем у начальной, не могут находиться в цикле, поэтому ассимптотику можно оценить как $\mathcal{O}(\sum_{i=0}^{n} 2^i i^2)$. Можно доказать по индукции, что $\sum_{i=0}^{n-1} 2^i i^2 \leq 2^{n+1} n^2$. Следоваельно алгоритм работает за нужную ассимптотику $\mathcal{O}(2^n n^2)$.

Задача 2

Данная задача решается рекурсивно. Обозначим исходные последовательности за a и b. Разобьем a пополам и посчитаем LCS для всех префиксов первой половины a с b и для суффиксов второй a с b. Теперь требуется найти элемент в b, такой, что сумма LCS на префиксе до него и суффиксе после него максимальна. Пусть индекс этого элемента i, теперь требуется запустить этот алгоритм для первой половины a и b от 0 до i, запустить для второй половины a и b от $i+1$ до $length(b)$. Алгоиритм прекращает рекурсивные запуски после, того как $length(a) = 1$, и в данной ситуации нужно либо просто добавить этот единственный ответ, если он лежит в b, или ничего не выводить.

Докажем корректность перехода: когда мы находим i, мы делим последовательности на две части, так чтобы $LCS_1 + LCS_2 = LCS$, тогда можно отдельно решать более маленькую задача для половин, так как конкатенация LCS_i будет искодной LCS.

Оценим асимптотику. Пусть размер певой строки n, второй m. Всего уровней рекурсии будет $\log(n)$. Рассмотрим каждый уровень рекурсии отдельно: на уровне k длина строки a равна $\frac{n}{2^k}$. Чтобы посчитать LCM на каждом уровне для всех пар строк понадобится $\mathcal{O}(\frac{nm}{2^k})$ времени, следовательно суммарно алгорит работает за $\mathcal{O}(nm)$.

Задача 3

Задача 4

Будем решать эту задачу методом динамического программирования. Пусть дан граф из n вершин, тогда заведем $dp[2^n][n]$. В $dp[i][j]$ будем поддерживать следующие значения: i - битовая маска посещенных вершин, j - начальная вершина, $dp[i][j]$ - количество путей начинающихся в вершине j и заканчивающихся в вершине с номером самого младшего единичного бита в i, обозначим его за v. Теперь зная значения такой динамики можно посчитать ответ, нужно просто проверить существование ребра между j и v, и если есть такое ребро прибавить значение динамики к ответу. Переход осуществляется просто прперебором всех вершин соседних к j, но больших v (для сохранения инварианта). В конце надо поделить ответ на 2, так как каждый цикл будет посчитан дважды(если поменять начало и конец местами). Данная динамика верно работает так как мы находим все простые пути, а простые циклы отличаются только ребром между концом и началом пути.