Первый курс, осенний семестр 2018/19
Практика по алгоритмам #9

Динамика

3 ноября

Собрано 4 ноября 2018 г. в 21:33

Содержание

Динамика

1. **Игра с камнями**
   Есть кучка из n камней. Двое играют в игру. Кто не может ходить, проиграл. Кто выиграет при оптимальной игре обоих?
   a) За ход можно брать любое число камней из $A = \{a_1, a_2, \ldots a_k\}$.
   b) Первый может брать $a_1, a_2, \ldots a_k$ камней за ход, второй – $b_1, \ldots b_m$.
2. **Шаблоны**
   Подходит ли строка под шаблон из `*?x` за $\mathcal{O}(nm)$.
3. **Задача о министерстве**
   Есть таблица $n \times m$ клеток, в каждой клетке натуральное число. Найти путь min веса из $(1, 1)$ в (n, m), если можно ходить вверх, вправо и влево. $\mathcal{O}(nm)$, восстановить ответ.
   a) $\mathcal{O}(nm)$ памяти.
   b) $\mathcal{O}(n^{1/2}m)$ памяти.
   c) `(*)` $\mathcal{O}(n^{1/3}m)$ памяти.
4. **Профессор и железные яйца**
   У профессора есть k яиц и n этажное здание. Он хочет узнать максимальное x: если яйцо бросить с x-го этажа, оно не разобьётся. Не разбившиеся яйца можно переиспользовать. Минимизировать число бросков в худшем случае. $\mathcal{O}(n^2k) \to \mathcal{O}(nk \log n) \to \mathcal{O}(nk) \to \mathcal{O}(n \log n)$.
5. **Динамика по подотрезкам**
   a) **Произведение матриц.** Нужно посчитать произведение матриц $A_1 A_2 \ldots A_n$ за минимальное число операций. Матрицы размеров $n \times k$ и $k \times m$ умножаются за nkm операций и дают матрицу размера $n \times m$.
   b) **Свертка.** Дана строка из латинских букв длины n, нужно ее запаковать в максимально короткую, используя правило $n(S) = \underbrace{SS\ldots S}_{n}$.

      Пример: `NEERCYESYESYESNEERCYESYESYES` $\to$ `2(NEERC3(YES))`.
6. **Пираты!**
   Судно атакуют пираты. Для каждого пирата известны его азимут a_i и время t_i, через которое пират приплывет и совершит непотребство.
   Однако, у судна есть лазерная пушка, которой оно защищается. У пушки есть начальный азимут a и угловая скорость вращения ω. Пушка уничтожает все объекты, на которые она сейчас направлена.
   Помогите судну определить порядок уничтожения пиратов за $\mathcal{O}(n^2)$, чтобы не допустить непотребства.
7. **Быстрая покраска забора**
   Дан массив длины n, который мы хотим получить. Числа в массиве от 1 до C. Получить его из массива $[0, 0, \ldots, 0]$ минимальным числом операций «покраска отрезка». $\mathcal{O}(n^3C) \to \mathcal{O}(n^3)$.
8. **Задачи про паросочетание**
   Найти максимальное по весу паросочетание (веса на рёбрах)
   a) на дереве за $\mathcal{O}(n)$.
   b) на цикле за $\mathcal{O}(n)$.
   c) на связном графе из n вершин и n ребер за $\mathcal{O}(n)$.

9. **(∗) Взорвите дерево!**
   Найти в дереве минимальное по *суммарному весу* множество вершин, чтобы расстояние от любой вершины до одной из выбранных было не более d.
10. **(∗) Триангуляция многоугольника**
    Найдите кратчайшую триангуляцию выпуклого многоугольника.
11. **(∗) Переливания**
    Есть три стакана размеров $1 \leqslant A \leqslant B \leqslant C \leqslant N$.
    Получить ровно x минимальным числом переливаний за $\mathcal{O}(N^2)$.
12. **(∗) Профессор и железные яйца**
    Решите для заоблачных зданий $k, n \leqslant 10^9$.

Разбор задач практики

1. **Игра с камнями**
   a) $f[i]$ – есть ли гарантированный способ выиграть у того, кто сейчас ходит, когда i камней. $f[i] = !f[i-a_1] \vee !f[i-a_2] \vee \ldots \vee !f[i-a_k]$.
   b) $f[i,p]$ – есть ли гарантированный способ выиграть у игрока p, когда i камней. $f[i,1] = !f[i-a_1,2] \vee !f[i-a_2,2] \vee \ldots \vee !f[i-a_k,2]$. Аналогично $f[i,2]$.

2. **Шаблоны**
   $f[i,j]$ – подходит ли строка $s[0,i)$ под шаблон $p[0,j)$.
   Если $p[j-1]$ – буква, то $f[i,j] = f[i-1,j-1] \wedge p[j-1] == s[i-1]$.
   Если $p[j-1]$ `== '?'`, то $f[i,j] = f[i-1,j-1]$.
   Если $p[j-1]$ `== '*'`, то звездочка ест либо ничего, либо что-то: $f[i,j] = f[i,j-1] \vee f[i-1,j]$.

3. **Задача о министерстве**
   $f[i,j]$ – минимальный путь в (i,j). $f[i] \to f[i+1]$:
   a) Сначала $f[i+1,j] = f[i,j] + \texttt{cost}[i,j]$.
   b) Проходим слева направо и $\texttt{relax}(f[i+1,j], f[i+1,j-1] + \texttt{cost}[i,j])$.
   c) Проходим справа налево и $\texttt{relax}(f[i+1,j], f[i+1,j+1] + \texttt{cost}[i,j])$.

4. **Профессор и железные яйца**
   $f[n,k]$ – минимальное число бросков. При броске с этажа i в худшем случае будет $\max(f[i-1,\ k],\ f[n-i,\ k-1])$. Назовем $\texttt{pos}[n,k]$ оптимальное i для n,k.
   Утверждение #1: $f[n,k] \leqslant f[n+1,k]$. То есть, $f[i-1,k]$ возрастает, $f[n-i,k-1]$ убывает. Значит, $\texttt{pos}[n,k]$ там, где впервые станет $f[i-1,k] \geqslant f[n-i,k-1]$, либо на 1 ниже.
   Утверждение #2: $\texttt{pos}[n,k] \leqslant \texttt{pos}[n+1,k]$. Обоснование: если $j < \texttt{pos}[n,k]$, то $f[j-1,k] < f[n-j,k-1] \leqslant f[n+1-j,k-1] \Rightarrow j \leqslant \texttt{pos}[n+1,k]$.
   Значит, при каждом k можно перебирать n и $\texttt{pos}[n,k]$ двумя указателями. $\mathcal{O}(nk)$.
   Наблюдение: ответ всегда можно найти за $\lceil \log n \rceil$ бросков, значит, при $k > \lceil \log n \rceil$ можно отвечать сразу $\lceil \log n \rceil$ (меньше нельзя, n вариантов ответа).

5. **Динамика по подотрезкам**
   a) **Произведение матриц.**
   $f[l,r]$ – минимальное число действий, нужное для перемножения отрезка $A_l A_{l+1} \ldots A_r$.
   Пусть размер i-й матрицы – $a_i \times a_{i+1}$.
   Тогда в результате умножения отрезка $[l..r]$ получится матрицы размера $a_l \times a_{r+1}$.
   Перебираем, какое умножение m будет последним на отрезке $[l,r]$, получаем
   $$f[l,r] = \min_m f[l,m-1] + f[m,r] + a_l \cdot a_m \cdot a_{r+1}$$
   Когда мы ищем $f[l,r]$, нужно чтобы уже были найдена f на всех подотрезках.
   Либо ленивая динамика, либо перебираем отрезки в порядке возрастания длины.
   Для восстановления ответа храним $m[l,r]$ – оптимальное m для $[l,r]$.
   Восстанавливаем рекурсивно: `ans(l, r) = (ans(l, m[l,r] - 1))(ans(m[l,r], r))`
   b) **Свертка.**
   Предподсчитаем `lcp[i,j]` за $\mathcal{O}(n^2)$.
   `ans[i,len]` – ответ на задачу для подстроки `[i..i+len)`.
   - Изначально `ans[i,len] = len` (не пытаемся сжимать подотрезок).

- Разделяем отрезок на две независимых части.
```
relax(ans[i, len], ans[i, m] + ans[i + m, len - m])
```
- Сворачиваем весь отрезок в повторение подстроки длины `m`.
```
if (len % m == 0 && lcp[i, i + m] >= len - m)
  relax(ans[i, len], length(len / m) + ans[i, m] + 2)
```
Эту часть можно ускорить, предподсчитав для каждого `len` список его делителей.

6. **Пираты!**
На любом отрезке последним умрет пират с одного из краев. $fl[l,r]$ – минимальное время, за которое можно успеть зачистить отрезок $[l,r]$, если последним будет убит пират на позиции l. Аналогично $fr[l,r]$.
Переходы: $fl[l,r] + |a[r+1] - a[l]| \to fr[l, r+1]$. Еще три аналогичных. Важно следить, что мы успеваем убить последнего пирата, иначе ответ $+\infty$.
На самом деле можно хранить только одну $f[l,r]$, где за последнего убитого пирата будет отвечать индекс l. Неоднозначности с тем, какую из половин круга выбрать, нет – ту, в которой находится начальное положение пушки.

7. **Быстрая покраска забора**

a) $\mathcal{O}(n^3C)$. Нужно получить цвета $a[i]$. $f[l,r,c]$ – минимальное число покрасок отрезка $[l,r]$, если он сейчас весь покрашен в цвет c.
Либо $a[r] = c$, тогда не надо красить последнюю клетку, и $f[l,r,c] = f[l,r-1,c]$.
Либо надо красить какой-то суффикс в цвет $a[r]$, $f[l,r,c] = \min\limits_m(1 + f[l,m-1,c] + f[m,r,a[r]])$. Этот суффикс может совпасть со всем отрезком, поэтому для отрезка $[l,r]$ надо всегда сначала считать $f[l,r,a[r]]$.

b) $\mathcal{O}(n^3)$. Теперь вместо использования c мы будем считать, что отрезок весь покрашен в цвет $a[l]$.
Либо $a[r] = a[l]$, тогда $f[l,r] = f[l,r-1]$.
Либо надо красить какой-то суффикс в цвет $a[r]$. Логично, чтобы этот суффикс начинался с такого m, что $a[m] = a[r]$, лишнее начало все равно перекрасят. То есть $f[l,r] = \min\limits_{a[m]=a[r]}(1 + f[l,m-1] + f[m,r])$.

8. **Задачи про паросочетание**
Найти максимальное по весу паросочетание (веса на рёбрах)

a) На дереве за $\mathcal{O}(n)$: считаем две величины
$f_0[v]$ – вес максимального паросочетания в поддереве v, не покрывающего v.
$f_1[v]$ – вес максимального паросочетания в поддереве v, покрывающего v.
Если нам неважно, покрывать вершину v или нет, то будем писать $f[v] = \max(f_0[v], f_1[v])$.
Пусть дети вершины v – $x_1, x_2, \ldots, x_k$, веса рёбер (v, x_i) – w_i, тогда:
$f_0[v] = \sum_i f[x_i]$
$f_1[v] = \sum_i f[x_i] + \text{ребро вниз} = \sum_i f[x_i] + \max_i \big(w_i - f[x_i]\big)$

b) На цикле за $\mathcal{O}(n)$.
Цикл отличается от дерева только одним ребром. Давайте переберём, включаем ли мы в паросочетание первое ребро. В обоих случаях останется дерево, решим для него задачу. Останется даже не дерево, а путь, массив, на массиве можно проще:
$f_0[i] = f[i+1], f_1[i] = w_i + f_[i+2], f_i = \max(f_0[i], f_1[i])$.

c) На связном графе из n вершин и n ребер за $\mathcal{O}(n)$.

Что это за граф такой? Это цикл, на котором растут деревья. Выделим цикл, деревья, сперва для деревьев запустим (а), затем для цикла (б).

9. **(*) Взорвите дерево!**
Сначала считаем динамику, учитывая расстояние только вниз. Затем считаем вторую динамику, уже учитывая пути вверх, используя предыдущую.

10. **(*) Триангуляция многоугольника**
Динамика по подотрезкам. Триангулируем многоугольник на вершинах $i, i+1, \ldots, j$ (в нем есть сторона (j, i)). Либо проводим диагональ (i, k), либо $(j, i+1)$. Итого $f[i, j] = \min(w[j, i+1] + f[i+1, j], \min\limits_k w[i, k] + f[i, k] + f[k, j])$.

11. **(*) Переливания**
Наивное решение за $\mathcal{O}(N^3)$: состояние – сколько налито в каждом стакане.
Замечаем, что в реальности достижимы только состояния, где один из стаканов либо пуст, либо полон.

12. **(*) Профессор и железные яйца**
Учимся считать динамику $n[k, s]$ – max высоту, для которой хватит s бросков k яиц. Дальше замечаем, что ее можно считать возведением в степень матрицы $k \times k$. Вспоминаем, что можно считать $k \leqslant \log n$, итого $\mathcal{O}(\log^4 n)$.

Домашнее задание

3.1. Обязательная часть

1. **(2) Сумма кубов**
   Разбить число n на сумму минимального числа кубов натуральных чисел, $n \leqslant 10^6$

2. **(2) Центроиды**
   Дано дерево из n вершин.
   Найти все вершины v: при удалении v размеры получившихся деревьев будут не более $\frac{2n}{3}$.

3. **(2) Зануление массива**
   Рассмотрим операцию `zero(i): ans += a[next(i)]*a[i]*a[prev(i)], a[i] = 0`.
   Массив циклический, перед i-м идёт элемент $(i-1) \bmod n$, после i-го идёт элемент $(i+1) \bmod n$. Найти последовательность операций, максимизирующую `ans`.
   a) **(1)** $\mathcal{O}(n^3)$.
   b) **(2)** $\mathcal{O}(n)$.

4. **(2) Дубы**
   Дуб можно срубить, если либо оба его соседа ниже, либо оба соседа выше. Крайние дубы рубить нельзя. Срубить минимальное число дубов, чтобы оставшиеся образовали неубывающую последовательность.

5. **(1) Правильные скобочные подстроки**
   Сколько существует строк из n круглых скобок, которые являются подстрокой правильной скобочной последовательности?

6. **(2) Сколько существует счастливых билетов?**
   Счастливый билет – строка из $2n$ цифр такая, что сумма цифр в первой половине равна сумме цифр во второй.

7. **(2) Разбивай и властвуй**
   Сколько разбиений числа N на неупорядоченные слагаемые?
   Слагаемых должно быть от A до B, величины слагаемых от L до R.
   $\mathcal{O}(N^3)$. **(+1)** балл за $o(N^3)$.

3.2. Дополнительная часть

1. **(3) Большая сумма кубов**
   Разбить число до 10^9 на сумму минимального числа кубов.

2. **(3) Паросочетание в кактусе**
   Найти максимальное по весу паросочетание в кактусе за $\mathcal{O}(n)$.
   Кактус – граф, в котором каждое ребро принадлежит не более чем одному циклу.

3. **(4) Редукция дерева**
   Посчитать количество поддеревьев данного дерева из n вершин, содержащих корень дерева и ровно $k \leqslant n$ вершин. Поддеревом здесь называется связный подграф.
   a) **(2)** $\mathcal{O}(n^3)$.
   b) **(3)** $\mathcal{O}(n^2)$.
   c) **(4)** $\mathcal{O}(nk)$.

4. **(3) Упорядоченный Хаффман**
   Дана строка s над алфавитом Σ.
   Каждой букве x из строки нужно сопоставить двоичную строку c_x так, чтобы:
   a) $x < y \Rightarrow c_x < c_y$,
   b) никакой c_x не является префиксом никакого c_y,
   c) $|c_{s_1}| + |c_{s_2}| + \ldots + |c_{s_n}| \to \min$.

   $\mathcal{O}(|\Sigma|^2 + |s|)$

5. **(3) Идеальный шаблон**
   Найти минимальный шаблон такой, что под него не подходит первая строка, но при этом подходит вторая. $\mathcal{O}(n^3)$.