Информатика в 57 школе, 9«М», 10«Д»
Жадные алгоритмы

17 сентября 2022 г.

На занятиях разобрали несколько стандартных задач:

1. Максимальное количество попарно непересекающихся отрезков. Дано множество из n отрезков $[l_1, r_1]$, $[l_2, r_2]$, $\ldots$, $[l_n, r_n]$ на прямой. Необходимо из этих отрезков выбрать как можно большее количество так, чтобы все выбранные отрезки попарно не пересекались.

Идея решения. Будем пытаться идти по отрезкам «слева направо» и среди ещё не выбранных отрезков каким-то образом выбирать оптимальный. Поймём, как именно стоит производить выбор оптимального отрезка и что значит «слева направо», когда речь идёт об отрезках.

Рассмотрим произвольный набор из заданных отрезков, в котором любые два не пересекаются. В этом наборе рассмотрим отрезок с *самым левым правым концом*, обозначим его за s.

Утверждение. Пусть отрезок s' не лежит в рассмотренном наборе и правая граница отрезка s' левее правой границы отрезка s, то есть $r_{s'} < r_s$ Тогда отрезок s в наборе можно заменить на s' и после такого изменения набор останется корректным, то есть никакие два отрезка по-прежнему не будут пересекаться.

Доказательство. Поскольку s — отрезок из набора с самым левым правым концом, то для любого отрезка i из набора выполнено

$$r_s' \leq r_s \leq r_i. \tag{1}$$

Тогда, если какой-то отрезок i из набора пересекается с отрезком s', то это возможно только при $l_i \leq r_{s'} \leq r_i$. Но, в таком случае, по неравенству (1) отрезок i пересекается и с отрезком s, так как $l_i \leq r_{s'} \leq r_s \leq r_i$. По предположению, набор был корректным, поэтому описанное выше возможно только при $i = s$. Значит, при замене s на s' набор останется корректным. □

Таким образом, мы показали, что в любом корректном наборе отрезок с самой левой правой границей можно заменить на отрезок с самой левой правой границей среди *всех* доступных отрезков. Отсюда сразу вытекает алгоритм построения наибольшего по размеру корректного набора:

1. Среди всех отрезков выберем отрезок с самым левым правым концом. Добавим его в набор.
2. Уберём из рассмотрения все отрезки, которые пересекаются с добавленным.
3. Мы свели задачу к аналогичной, но уже с меньшим множеством доступных отрезков. Поэтому, если доступные отрезки ещё остались, то просто переходим к шагу 1. В противном случае оптимальный набор отрезков построен.

В таком виде данный алгоритм работает за время $O(n^2)$, где n — число заданных отрезков. Можно придумать алгоритм за $O(n \log n)$ на сортировку отрезков по правым концам плюс один проход по отсортированному массиву отрезков. Оставим это в качестве упражнения.

2. **Мероприятия.** Дан набор мероприятий, каждое из которых характеризуется временем начала a_i и временем окончания b_i. В каждый момент времени можно проводить не более одного мероприятия. Найти, какое наибольшее число мероприятий можно провести.

Идея решения. Поймите, что это та же самая задача, что и предыдущая.

3. **Покрытие отрезка отрезками.** Дан набор отрезков $S = \{[l_i, r_i] \mid i = 1, \ldots, n\}$ на прямой, а также отрезок $[L, R]$. Найдите минимальное по размеру подмножество S, объединение отрезков которого полностью содержит $[L, R]$.

Идея решения. Рассмотрим точку L, выберем среди всех отрезков, содержащих L, отрезок с максимальной правой границей, пусть это отрезок $[l_i, r_i]$. Остальные отрезки, содержащие L, можно убрать из рассмотрения, так как их просто нет смысла включать в ответ. Задача свелась к аналогичной задаче для точки $r_i \geq L$ и меньшего числа доступных отрезков.

4. **Покрытие множества отрезков точками.** Дан набор отрезков $S = \{[l_i, r_i] \mid i = 1, \ldots, n\}$ на прямой. Найдите минимальное количество точек, которое нужно отметить, чтобы каждый отрезок содержал хотя бы одну отмеченную точку.

Идея решения. Рассмотрим отрезок с самой левой правой границей (то есть с минимальным r_i). На этом отрезке должна быть отмечена какая-то точка. Заметим, что в качестве этой отмеченной точки оптимально выбрать его правую границу. Удалим из рассмотрения все отрезки, содержащие отмеченную точку. Мы свели задачу к аналогичной, но для меньшего числа отрезков.

5a. Даны целые числа x и y. Над ними можно проводить два вида операций:

(1) Увеличить или уменьшить одно из чисел на 1.

(2) Увеличить или уменьшить оба числа на 1 (нельзя одно число увеличить на 1, а другое уменьшить на 1).

За какое минимальное число операций можно оба числе сделать равными нулю?

Идея решения. Рассмотрим случаи:

(a) Числа x и y лежат по одну сторону от нуля.
Тогда оптимально сначала сделать большее по модулю из чисел x, y равным меньшему по модулю с помощью операции (1), после чего сделать оба числа равными нулю с помощью операции (2). Это потребует $|x - y| + \min(|x|, |y|)$ действий.

(b) Числа x и y лежат по разные стороны от нуля. Тогда оптимально сделать x и y равными нулю по отдельности с помощью операции (1), это потребует $|x| + |y|$ действий.

5b. Пусть в условиях предыдущей задачи сделать операцию (1) стоит $a \geq 0$ рублей, а сделать операцию (2) стоит $b \geq 0$ рублей. Найти минимальную стоимость того, чтобы сделать x и y равными нулю.

Идея решения. Заметим, что результат не зависит от порядка операций. Тогда можно считать, что сначала делаются все операции вида (1), потом все операции вида (2). Чтобы с помощью операций вида (2) можно было сделать пару чисел равными нулю, перед применением операций (2) числа уже должны быть равны. Таким образом, с помощью операций вида (1) мы обязаны сделать числа равными. Пусть в результате операций вида (1) мы получили $x = y = C$. Это стоит $|C - x| \cdot a + |C - y| \cdot a$ рублей. Далее, операциями вида (2) мы делаем $x = y = 0$ за $|C| \cdot b$ рублей. Таким образом, нам остаётся найти C, при котором значение выражения

$$f(C) := |C - x| \cdot a + |C - y| \cdot a + |C| \cdot b \tag{2}$$

минимально. Заметим, что $f(C)$ — кусочно-линейная функция от C. Так как линейная функция на отрезке всегда достигает минимума в одном из концов отрезка, то для нахождения минимума $f(C)$ достаточно рассмотреть точки $C = x$, $C = y$ и $C = 0$. Таким образом, ответ на задачу равен $\min\Big(f(x),\ f(y),\ f(0)\Big)$.

Мораль задачи 5b. Вместо того, чтобы смотреть, что будет при разных вариантах того, как соотносятся значения a и b, мы поняли, как концептуально может быть устроен ответ и выяснили, что в итоге задача сводится к перебору конечного числа вариантов.

6. **Футбольная команда.** Для краткости сразу напишем математическую переформулировку задачи. Дан массив a из n неотрицательных чисел $a_1,\ \ldots,\ a_n$. Нужно найти подмножество $T \subseteq \{1,\ \ldots,\ n\}$, с максимальным значением $\sum_{i \in T} a_i$, такое что для любой тройки различных индексов $i, j, k \in T$ для чисел a_i, a_j и a_k выполнено неравенство треугольника: $a_i + a_j \geq a_k$.

Идея решения. Когда задача имеет вид «найдите максимальное значение данной функции на данном классе объектов» и класс объектов параметризуется довольно сложно, есть смысл попробовать зафиксировать какую-то информацию об объекте и попробовать уже с этим ограничением промаксимизировать данную функцию. Данная задача — как раз такой случай. Класс объектов, рассматриваемый в задаче — это множества $T \subseteq \{1, \ldots, n\}$, для которых выполняется неравенство треугольника в указанном выше смысле. А промаксимизировать нам надо $\sum_{i \in T} a_i$. Класс объектов, как мы видим, параметризуется не очень тривиально, поэтому можно попробовать зафиксировать какой-то параметр множества T. В данном случае это будет максимальный элемент среди a_i при $i \in T$. Этот выбор мотивирован тем, что при проверке неравенства треугольника для множества T достаточно проверить, что сумма двух минимальных элементов a_i и a_j при $i, j \in T$ не меньше максимального a_k при $k \in T$.

Без ограничения общности можно считать, что массив a отсортирован по возрастанию. Тогда можно фиксировать не максимальное значение a_i при $i \in T$, а максимальный индекс i при $i \in T$.

Утверждение. Зафиксируем максимальный индекс в множестве T, пусть он равен i. Тогда оптимальное множество T имеет вид отрезка $[l,\ l+1,\ l+2,\ \ldots,\ i-1, i]$ для некоторого $l = l(i) < i$.

Доказательство. Было разобрано на уроке, можно ещё раз его проделать самостоятельно в качестве упражнения.

Из данного утверждения легко сразу получить алгоритм решения задачи: переберём максимальный индекс i, для каждого i найдём с помощью цикла `while` значение $l(i)$ и посчитаем сумму на отрезке $\mathtt{a}[\mathtt{l(i)},\ \mathtt{l(i)+1},\ \ldots,\ \mathtt{i-1}, \mathtt{i}]$, попробуем обновить ответ этим значением.

Такой алгоритм работает за $O(n^2)$. Также на уроке было разобрано улучшение этого алгоритма с помощью бинарного поиска и префиксных сумм за $O(n \log n)$. В качестве домашнего упражнения предлагается придумать, как ускорить этот алгоритм до асимптотики $O(n)$.