Какая-то лекция по алгоритмам
Тема: 3 задачки на динамику

Когда-то там

Собрано 10 января 2015 г. в 00:02

Содержание

1 Задача 1(Билет 12)

- **Формулировка**: Из порта нужно увезти груз из n ящиков. Все ящики стоят в ряд, и каждый ящик имеет вес w_i. Требуется увезти все грузы, используя минимальное число перевозок, на корабле грузоподъёмности W, если в любой момент времени разрешается грузить только самый левый или самый правый ящик
- **Решение 1**:
  • Динимика : d_{lr} - минимальное число поездок, которое потребуется, чтобы увезти все грузы, кроме $[l, r)$
  • База : $d_{0n} = 0$
  • Переход : $d_{lr} \to d_{(l+x)(r-y)}$ (погрузили на корабль x ящиков слева и y ящиков справа)
  • Итог : состояний $O(n^2)$, переходов $O(n^2)$ Общее время работы $O(n^5)$ или $O(n^4)$, если использовать частичные суммы
- **Улучшение решения 1**:
  • Заметим, что если мы погрузили на корабль x ящиков слева, то нам всегода выгодно погрузить как можно больше ящиков справа. Максимальный y можно быстро находить, используя метод двух указателей
  • Итог : переходов стало $O(n)$. Общее время работы $O(n^3)$
- **Решение 2**:
  • Динимика : d_{lrf} - минимальный вес, который нужно положить на корабль при $(f+1)$-ой погрузке, чтобы в порту остались грузы $[l, r)$ и было сделано f перевозок
  • База : $d_{0n0} = 0$
  • Переход :

    1. $d_{lrf} \to d_{(l+1)rf}$ (погрузили на корабль левый ящик)
    2. $d_{lrf} \to d_{l(r+1)f}$ (погрузили на корабль правый ящик)
    3. $d_{lrf} \to d_{lr(f+1)}$ (отправили корабль)

  • Итог : состояний $O(n^3)$, переходов $O(1)$ Общее время работы $O(n^3)$.
- **Решение 3**:
  • Динимика : d_{lr} - минимальная пара $< f, w >$, где f - число поездок, а w - вес, загруженный на корабль для $(f + 1)$-ой поездки
  • База : $d_{0n} =< 0, 0 >$
  • Переход :

    1. $d_{lr} \to d_{(l+1)r}$ (погрузили на корабль левый ящик)
    2. $d_{lr} \to d_{l(r+1)}$ (погрузили на корабль правый ящик)

  • Итог : состояний $O(n^2)$, переходов $O(1)$ Общее время работы $O(n^2)$.

- **Поговорим о памяти**
  - В наивном написании решения 3 будет использовано $O(n^2)$ памяти
  - Будем считать динамику в порядке увеличения l, тогда нам достаточно хранить два массива для l и $l+1 \Rightarrow O(n)$ памяти
  - Если требуется восстановить ответ, сделаем это алгоритмом Хиршберга

2 Задача 2(В билетах нет, можно не читать)

- **Формулировка**: Есть квадратная доска $N \times N$. У каждой клетки есть стоимость прохождения w_{ij}. Требуется за минимальную цену добраться из левого нижнего угла в правый верхний, если можно двигаться влево, вправо и вверх на одну клетку
- **Решение**:
  - Динимика : d_{xy} минимальная цена, которая потребуется, чтобы добраться из левого нижнего угла в клетку $[x, y]$
  - База : $d_{00} = w_{00}$
  - Переход : пересчитывать динамику будем сразу для всей строки. Для этого достаточно сделать два цикла : слева направо и справа налево
  - Память : для данного алгоритма достаточно хранить две последние строки $\Rightarrow O(n)$ памяти, но теперь, если мы попытаемся восстеновить ответ алгоритмом Хиршберга, время работы увеличится с $O(n^2)$ до $O(n^2 \log n)$. В качестве альтернативы, можно хранить $\sqrt{n}$ строк на расстоянии $\sqrt{n}$, а между ними честно считать ответ, тогда время работы останется $O(n^2)$, а памяти потребуется $O(n\sqrt{n})$

3 Задача 3(В билетах нет, можно не читать)

- **Формулировка**: На плоскости расположены красные и сние точки. Требуется найти наибольший по площади выпуклый многоугольник на красных точках, внутри которого нет синих точек
- **Решение**:
  - Переберём нижнюю вершину многоугольника и отсортируем вершины выше по углу. Теперь найдём ответ для многоугольника с текущей нижней вершиной
  - Динимика : s_{ij} - максимальная площадь многоугольника, удовлетворяющего условию, и в котором вершины с номерами i и j в порядке увеличения угла являются последней и предпоследней соответственно
  - База : $s_{12} = S_{\triangle(0,1,2)}$
  - Переход : $s_{ij} = \max_k(s_{ki} + S_{\triangle(0,i,j)})$ Нужно проверить, что многоугольник остался выпуклым, и что в добавляемом треугольнике нет синих точек (это можно предподсчитать для всех треугольников)
  - Итог : перебираем все вершины за $O(n)$, $O(n^2)$ состояний динимики и $O(n)$ переходов. Получаем общее время работы алгоритмя $O(n^4)$