АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА КАРГО

Решение за 30 точки

С `N` означаваме броя дни на периода на планиране. Дефинираме две матрици с размерност `NxN` `Days` и `Holidays` и ги занулляваме.

Заявките за извършване на превози записваме в матрицата `period`. Ако заявката за превоз е от `i`-ти до `j`-ти ден на периода за планиране, ние добавяме `1` (единица) към елемента `period[i-1][j-1]`. Единица, понеже за един ден един екипаж може да извърши един превоз или казано с други думи – всяка една заявка касае един полет. Ако отново въведем заявка за периода `[i,j]` към елемента `period[i-1][j-1]` ще прибавим още една единица и т. н.

Ако разгледаме примера от условието:

Три заявки, 2-ма пилоти

Първата заявка е каргото да се получи в периода от 4-ти до 5-ти ден от периода включително.

Втората заявка е каргото да се получи в периода от 5-ти до 6-ти ден от периода включително.

Третата заявка е каргото да се получи в периода от 5-ти до 7-ти ден от периода включително.

Матрицата `period` изглежда така:

0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0
      0 **1** 0 0 0 0
      0 **1** **1** 0 0
        0 0 0 0
          0 0 0
            0 0
              0

Очевидно е, че ще използваме само горната триъгълна част от матрицата, тъй като по условие: $1 \le S_i \le E_i \le N$.

В горната триъгълна част на матрицата могат да бъдат кодирани абсолютно всички възможни времеви интервали (заявки), касаещи периода на планиране.

Обхождаме последователно периода на планиране, от първия до последния ден избираме съгласно следното правило:

Сред заявките, които са останали и които могат да бъдат изпълнени в текущия ден, избираме тази, чиито краен срок е най-ранен (най-близък).

Решение за 60 точки

Описаното по-горе решение не може да се използва за решаване на втората подзадача, защото паметта, нужна за описаните матрици, е недостатъчна. За решаването на задачата е достатъчно да се запомнят началата и краищата на заявките без да се интересуваме кое начало с кой край е свързано. За реализацията на това решение могат да се използват едномерни масиви с размерност `N` и необходимата памет значително намалява. След това, както в гореописаното решение, обработваме информацията ден по ден. При обхождане на интервала на планиране поддържаме информация за броя на неизпълнените заявки и броя на активните интервали към текущия момент и от тях правим извод дали заявките могат да се изпълнят, ако се работи и в почивните дни. В случая, когато няма да се работи в събота и неделя, може

да решаваме задачата по аналогичен начин като преобразуваме входните данни (махаме съботите и неделите). Ако началото на някой интервал е в събота или неделя, го преместваме в понеделник, а ако краят на някой интервал е в събота или неделя, го премествам в петък.

Решение за 100 точки

За решаване на последната подзадача можем да работим по начин, аналогичен на описаните по-горе, но обхождането на интервала на планиране да извършваме не ден по ден, а като прескачаме дните, в които не започва и не завършва нито един интервал на заявка. Тъй като при ограниченията на третата подзадача $M \leq 10^6$ и $N \leq 10^9$, за реализацията на това решение е достатъчно да се използват масиви с размерност $2M$. Също така при такова решение времето, необходимо за обхождането на интервала на планиране ще зависи линейно от стойността на M.

Автор: Пано Панов