Project Euler Task 78

Козиний Сергей

3 ноября 2016 г.

Problem 78
“

Let p(n) represent the number of different ways in which n coins can be separated into piles. For example, five coins can be separated into piles in exactly seven different ways, so p(5)=7.

OOOOO
OOOO O
OOO OO
OOO O O
OO OO O
OO O O O
O O O O O

Find the least value of n for which p(n) is divisible by one million.
”

Итак, необходимо найти n, для которого p(n) делится нацело на 1000000, при этом p(n) — количество разбиений суммы n на множество неотрицательных целых слагаемых произвольного размера При рассмотрении условий становится понятно, что задачу нахождения $p(n)$ можно разбить на сумму двух подзадач, если $p(n)$ представить в виде $p'(n, n)$, где

$p'(a, b)$ — количество разбиений суммы b на слагаемые размера не более, чем a. Из данного определения можно вывести следующие свойства:

$p'(a+k, a) = p'(a, a)$ (В разбиении суммы на слагаемые не могут участвовать слагаемые большего размера, чем сама эта сумма)

$p'(1, b) = 1$ (Имеется одно разбиение суммы при использовании слагаемых размера 1)

Само разбиение p’(a,b) на подзадачи будет выглядеть так:

$p'(a, b) = p'(a - 1, b) + p'(a, b - a)$

Мы представляем задачу в виде суммы решений задачи, в которой мы задействовали слагаемое наибольшего размера и задачи, в которой это слагаемое задействовано не будет

Поскольку для решения $p(n)$ необходмо вычислять $p'(a, b)$ для

$0 \leq a \leq n$;

$0 \leq b \leq n$

то результаты, можно записывать в двумерный массив размера $n * n$, для избежания построений деревьев рекурсивных вызовов.

Другая важная оптимизация процедуры может заключатся в том, чтобы на каждом шаге i хранить только строку массива, соответвтующую первому аргументу p’, вычисленную на предыдущем шаге, а на каждом новом шаге записывать в эту же самую строку значения p’(i,j) слева направо, т.е для j, начиная с 1.

Предвидеть наперед приблезительный порядок n, такого, что $p(n)\ mod\ 1\ 000\ 000 = 0$ сложно, но для выделения памяти в массиве необходимо задать некоторое m — размер этого массива, число, до которого мы проверим все p(n) на делимость нацело на 1 000 000. В нашей программе n положим равным 100 000 (как будет определено в дальнейшем, искомое n будет меньше этого числа)

В качестве массива для хранения результатов промежуточных вычислений воспользуемся STUArray, поскольку он имеет наивысшую скорость доступа к элементам среди всех рассмотреных аналагов.

```haskell
--Haskell lang
task78 =head $ runST$ (lookup m)
  where
    m = 100000::Int
    lookup n =  do
      res <- (newSTRef [])
      aR <- ((newArray (0,n) 1)::ST s (STUArray s Int Int))
      forM_ [(2::Int)..n] $
                \i -> do
                  forM_ [i..n] $ \j ->
                       do
                         v <- readArray aR (j-i)
                         w <- readArray aR j
                         writeArray aR j ((v+w)`mod` 1000000)
                  p <- readArray aR i
                  if (p == 0)
                  then   modifySTRef res (\x -> x ++ [i] )
                  else return ()
      readSTRef res
```

В этой прогамме выделяется одноменый массив размера 100000, который инициализируется еденицами. Затем, в цикле на каждом шаге i к каждому значению массива с индексом j прибавляется здачение массива с индексом j-i. Если на i шаге внешнего цикла i элемент массива по модулю 1000000 равен 0, то этот результат записывается в хвост переменной res. Если lookup m возвращает пустой список, то искомое значени n больше, чем заданное m

Данное решение являетя неоптимальным, и программа производит вычисления достаточно долго. (В O-нотации затраты памяти составляют $O(n)$, а времени — $O(n^2)$) Для того, чтобы написать программу более оптимальным способом, необходимо воспользоваться Euler Patrition Formula. Однако о её существовании я узнал уже после того как ответ был найден и отправлен на сайт, и поэтому решил предоставить отчёт об этом решении задачи, не внося в него никаких изменений.

```
austrotaxus@small-box:~/EulerProj/reports$ stack exec EulerProj-exe task78
Input the number of needed task
78
55374
```

Проект, содержащий это и другие решения можно найти по адрессу:

`https://github.com/Austrotaxus/EulerProj/`