Анализ на задача "Функции"

CodeIT, 2014-2015, Round 2

Факта, че търсим най-малкото число R, за което е изпълнено условието (има поне K числа X, такива че L ≤ Q(X) ≤ R) трябва да ни наведе на мисълта за двоично търсене по отговора.

Така задачата се свежда до това за дадено число R да намерим броя числа X, за които

L ≤ Q(X) ≤ R.

Първо трябва да направим няколко наблюдения. Q(X) ≤ R≤ 1018. Q(X) = X\*P(X). Можем да забележим, че P(X) ≤ X за всяко X. Следователно P(X) ≤ sqrt(R) ≤ 109.

P(X) е произведение на цифри, следователно простите му множители може да за само 2, 3, 5 или 7. Броят числа P(X) = 2a 3b 5c 7d ≤ 109 е малък (по-точно 5194), следователно можем да ги обходим всичките.

Да се върнем на уравнението L ≤ X\*P(X) ≤ R. Нека си фиксираме числото P(X). Тогава търсим броят числа floor(L / P(X)) ≤ X ≤ ceil(R / P(X)), такива че произведението на цифрите им е P(X).

Можем да намерим всички такива числа като ги построяваме отляво надясно цифра по цифра.

Нека int rec (string prefix, int N, int prod) е такава функция която връща броя числа, които започват с prefix и имат още N цифри, чието произведение е prod.

Нека също така си дефинираме и две променливи max = числото, получено като добавим N девятки към prefix, min = числото, получено като добавим N нули към prefix.

int rec (string prefix, int N, int prod)

{

If (max < floor(L / P)) or (min > ceil(R / P)) then

Return 0

If N == 0 then

If prod == 1 then Return 1

else Return 0

Count = 0

For each D from 1 to 9

If prod % D == 0 then

Count += rec(prefix.append(D), N-1, prod/D)

Return Count

}

Оптимизации:

Можем да ускорим алгоритъма ако добавим меморизация на стойностите които връща фунцкията rec. За целта разглеждаме два случая:

* Ако min ≥ floor(L / P) и max ≤ ceil(R / P) (т.е. всички стойности които можем да генерираме са в интервала), тогава резултатът е един и същ независимо от числото което сме генерирали до сега и можем да игнорираме prefix параметъра.

Остават ни само N и prod като параметри и след като N може да приеме само 18 различни стойности, а prod не повече от 5194, можем да си запишем резултата в масив.

* Ако min ≥ floor(L / P) и max > ceil(R / P) или min < floor(L / P) и

max ≤ ceil(R / P), то числото генерирано в prefix е или префикс на floor(L / P) или е префикс на ceil(R / P), тоест броя случаи които трябва да меморизираме зависи от N.