Descrierea soluțiilor

1 Problema Zalău01

Autor: Andrei Constantinescu, Vlad Ciuleanu, ETH Zürich

1.1 Subproblema 1: $N = K = 50$

Deoarece $K = N$, ne interesează doar numărul de perechi 1-1 din tot șirul, care este dat de $\frac{\#1\cdot(\#1-1)}{2}$, unde #1 reprezintă numărul de 1-uri din șir. Rezolvând ecuația $\frac{\#1\cdot(\#1-1)}{2} = X$, putem determina numărul de 1-uri necesare și apoi să folosim elemente de bază de combinatorică pentru a calcula eficient numărul de posibilități (răspunsul va fi o combinare ținând cont de numărul de poziții cu `?` și de numărul de `1`-uri prezente în șir).

1.2 Subproblema 2: $N = 10$

Se pot încerca toate cele $O(2^N)$ șiruri posibile și număra cele care respecta proprietatea cerută. Complexitate totala: $O(2^N \cdot N^2)$.

1.3 Subproblema 3: $N = 50, K \leq 6$

Se procedează prin dinamica pe stări exponențiale $dp[i, x, mask]$, reprezentând numărul de posibilități de a alege primele i elemente din șir, astfel încât să avem x perechi de 1-uri de tipul cerut printre ele, iar ultimele $K-1$ elemente din șirul fixat până acum să fie date de masca pe biți $mask$. Complexitate totala: $O(2^K \cdot N \cdot X)$.

1.4 Subproblema 4: $N = 50, K \geq 44$

Fixăm în toate modurile posibile primele și ultimele 6 elemente din șir (sunt maxim 2^{12} moduri). Pentru restul de $50 - 2 \cdot 6 = 38$ de elemente problema devine pur combinatorică, analoagă Subproblemei 1 discutate mai sus.

1.5 Subproblema 5: $N = 40, K \geq 26$

Vom explica o soluție mai generală, pentru $K \geq \frac{N}{2} = 20$. Pentru a simplifica explicația, vom presupune că $N = 2 \cdot K$, dar abordarea generalizează natural.

Împărțim șirul în două bucăți a câte K elemente. În contrast cu soluția de complexitate $O(2^K \cdot N \cdot X)$ prezentată anterior, care fixa elementele în dinamică în ordinea naturală $1, \ldots, N$, aici vom proceda tot prin programare dinamică, dar fixând elementele în ordinea mai atipică

$$1, K+1, \quad 2, K+2, \quad \ldots, \quad K-1, 2 \cdot K - 1, \quad K, 2 \cdot K.$$

De ce facem așa? Pentru că astfel informația auxiliară necesară nu mai este o masca formata din $K - 1$ biți, ci doar două numere $0 \leq C_1, C_2 \leq K$, reprezentând numărul de 1-uri din prima, respectiv a doua, jumătate a șirului. Folosind C_1 și C_2 putem de cate ori fixăm un nou element al șirului să deducem cate perechi noi de 1-uri se creează. Complexitate totală: $O(K^2 \cdot N \cdot X)$.

1.6 Subproblemele 6-7: $N \leq 50$

Vom combina soluțiile Subproblemelor 3 și 5. Pe prima o vom folosi exact ca înainte — îi amintim complexitatea: $O(2^K \cdot N \cdot X)$. Pentru a doua, vom împărți șirul în $\lceil \frac{N}{K} \rceil \approx \frac{N}{K}$ bucăți de K. Aplicând același raționament ca mai sus, adică fixând elementele în dinamică în ordine după restul la împărțirea cu K, și la egalitate după cât, obținem un algoritm în complexitate $O(K^{\frac{N}{K}} \cdot N \cdot X)$.

Acum, care dintre cei doi algoritmi este mai rapid? Răspunsul depinde de care este mai mare între 2^K și $K^{\frac{N}{K}} = 2^{\frac{N}{K} \log_2 K}$. Intuitiv, primul este mai rapid pentru K mic, iar al doilea pentru K mare. Cele două cantități se întâlnesc când $K = \frac{N}{K} \log_2 K$, adică când $K = P$, unde P este soluția ecuației $P = \sqrt{N \log_2 P}$. Dacă pentru $K \leq P$ rulăm primul algoritm, iar pentru $K > P$ îl rulăm pe al doilea, obținem un algoritm de complexitate $O(2^P \cdot N \cdot X)$, reprezentând soluția oficială a problemei. Din păcate, nu există o expresie în formă închisă pentru P, dar pentru o aproximare, putem observa că $\sqrt{N} \leq P \leq \sqrt{N \log_2 N}$, deci complexitatea finală este cuprinsă între $O(2^{\sqrt{N}} \cdot N \cdot X)$ și $O(2^{\sqrt{N \log_2 N}} \cdot N \cdot X)$.