Tema 1 Algoritmi Avansati

Moroianu Theodor

23 martie 2021

1 Exerciţiul Knapsack

Cerinta 1

Consideram functia urmatoare, scrisa in limbajul *C++*:

```
int SumaMax(vector <int> w, int K)
{
    vector <int> viz(K + 1);
    viz[0] = 1;
    for (auto i : w)
        for (int j = K; j >= i; j--)
            viz[j] |= viz[j - i];

    int ans = K;
    while (!viz[ans])
        ans--;
    return ans;
}
```

Din cod se observa imediat ca are o complexitate de $\Theta(N * K)$, unde N este numarul de obiecte.
Tot din cod se vede ca spatiul folosit este $\Theta(K)$.

Lema 1. *Cand am procesat primele i elemente, $viz_x = 1$ daca si numai daca exista o submultime a multimii $\{ W_1, \ldots W_i \}$ cu suma x.*

Demonstrație. Demonstram lema prin inductie.
Cazul $i = 0$ este elementar: $W_x = 1$ daca si numai daca $x = 0$.

Cand adaugam un element Q in multime, apar doua cazuri:

- Nu consideram elementul Q pentru a obtine suma x. Observam ca algoritmul nostru trateaza acest caz ne-setand niciodata o valoare de 1 in 0.
- Consideram elementul Q pentru a obtine suma x. Observam de asemenea ca algoritmul nostru trateaza acest caz setand $viz'_i = viz_i \vee viz_{i-Q}$.

Asadar, algoritmul prezentat mai sus ofera o solutie corecta. □

Cerinta 2

Consideram functia urmatoare, scrisa in limbajul *C++*:

```cpp
int K, ans = 0, act;
cin >> K;

while (cin >> act)
{
    ans += act;
    if (ans > K)
        ans -= act;
    else if (ans < act)
        ans = act;
}

cout << "Answer is " << ans << '\n';
```

Din cod se observa ca are o complexitate de $\Theta(N)$, unde N este numarul de obiecte citite. Tot din cod se vede ca spatiul folosit este $\Theta(1)$, mai exact 3 variabile.

Lema 2. *Algoritmul prezentat mai sus este* 2*-aproximativ.*

Demonstrație. Consideram secventa de obiecte citite (exceptandul pe K) S, si OPT solutia optima.
Exista doua cazuri:

- $\exists\ i$ a.i. $S_i > \frac{OPT}{2}$.
  Altfel spus, exista un element mai mare decat solutia optima impartita la 2. Algoritmul nostru ne garanteaza ca va gasi un raspuns mai mare sau egal cu S_i, considerand cazurile cand raspunsul este format dintr-un singur element.

- $\forall\ i\ S_i \leq \frac{OPT}{2}$.
  In acest caz, avem garantia ca atata timp cat raspunsul partial este mai mic decat $\frac{OPT}{2}$ putem sa mai adaugam un element.
  Asadar, cum suma elementelor este cel putin OPT si noi putem mereu adauga elemente cat timp suma noastra este mai mica decat $\frac{OPT}{2}$ stim ca algoritmul o sa aduca raspunsul in intervalul $[\frac{OPT}{2} + 1, OPT]$.

Observam ca in ambele situatii gasim un raspuns satisfacator. $\square$

Codul pentru cele doua exemple se gaseste in *ZIP*-ul atasat.