1/0 Knapsack problem

n obiecte date prin (val, weight); un rucsac de capacitate W. Sa gasesc submultimea de obiecte de profit maxim care incap in rucsac.Nu am voie sa tai obiectele.

* algoritm pseudo-polinomial in timp O(n\*W)

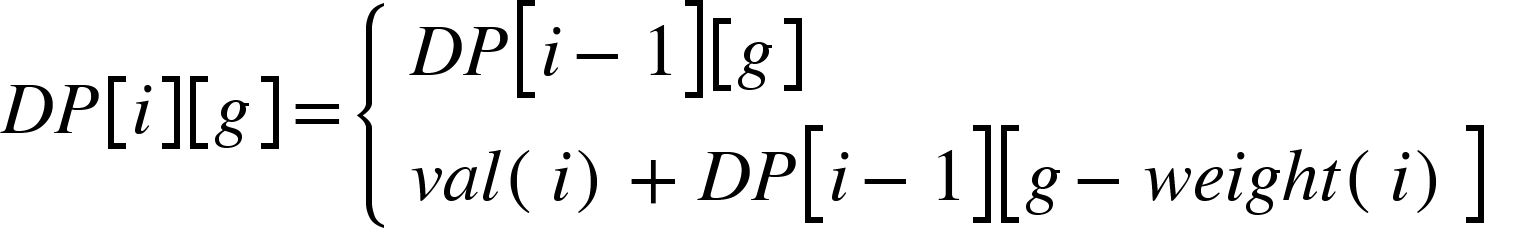
Un algoritm bazat pe programare dinamica

DP[i][g] - suma maxima ce poate fi obtinuta din primele *i* obiecte, fara a depasi greutatea totala *g*

DP[n][W] - va fi raspunsul pt problema noastra

Cum calculam DP?

DP[i][0]=DP[0][g]=0



Input:  
n - numarul de obiecte

W - capacitatea rucsacului

w[] - greutatile obiectelor

val[]-valorile al obiectelor.  
  
Output:  
S - profitul maxim obtinut;

x[] - un vector de *n* elemente de 1 si 0 cu semnificatia ca daca x[i]=1 atunci obiectul *i* a fost selectat.

Ex: W=50; w={10,20,30} val={60,100,120}

Output: S=220 x: [0,1,1]

alernativ: S=220; Obiectele 2,3;

2) Joc   
Fie urmatoarele reguli de joc:

2 jucatori au in fata un sir de numere S. La fiecare mutare jucatorul curent alege din S unul dintre cele doua capete, il “rupe” din S, iar valoarea numerica o adauga la punctajul obtinut. Apoi celălalt jucator va trebui sa mute. Jocul se termina atunci cand S este nul iar castigatatul este jucatorul care a strans un punctaj maxim.

1. Daca S este de lungime para, exista vreo tactica de joc astfel incat primul jucator sa isi asigure cel putin un punctaj egal?

Se uita la paritatea indicilor si alege mereu oelementele de pe pozitiile pare/impare astfel incat isi asigura cel putin un punctaj egal cu adversarul.

1. Sirul este de o lungime oarecare. Jucatorul 1 doreste sa obtina un punctaj maxim/ sa joace mutari optime.

T[i][j] - punctajul maxim care poate fi obtinut de catre jucatorul care se afla la mutare si are in fata ochilor sirul S[i…..j].

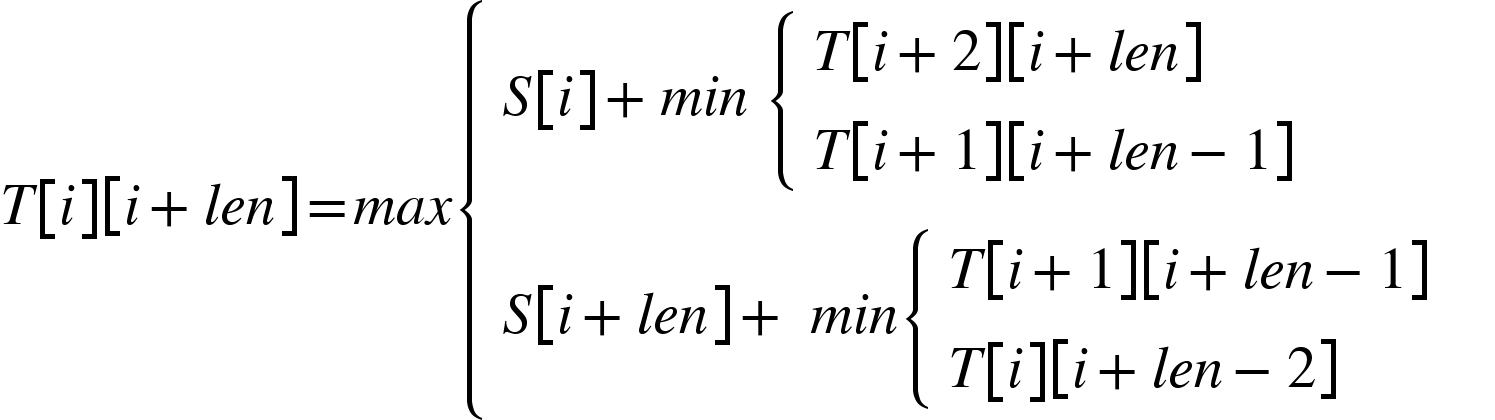
T[1][n] - este valoarea de interes pentru problema noastra

conditie: [i,j] - un interval valid

T[i][i]=S[i]

T[i][i+1]=max(S[i],S[i+1])

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | ... | i | i+1 | i+2 | ... | j-2 | j-1 | j | ... | n |



Complexitate: O(n^2)

Atentie - ordinea in care calculati elementele matricii

i, i; i,i+1; i,i+2; i,i+3….\