Problema Rucsacului (general)

Avem *n* obiecte,, fiecare dintre ele avand o greutate si o valoare. Avem un rucsac de capacitate W, trebuie sa gasim o submultime de obiecte astfel incat suma greutatilor sa nu depaseasca pe W iar suma valorilor sa fie maxima.

1. **Problema continua a rucsacului**

* fiecare obiect poate fractionat inainte de a fi adaugat in rucsac.

Solutie:

Tehnica Greedy:  
Sortam obiectele descrescatolr dupa raportul val/greutate. Iteram prin lista de obiecte. Cat timp mai exista loc in rucsac, adaugam integral obiectul. Pe ultimu obiect eventual il fractionam si adaugam doar o parte a sa.

W=50

Obiecte (val/greutate): [30/10; 50/25; 100/40; 15/3; 25/5]

Sortez obiectele: [**25/5**; **15/3**; **30/10**; ***100/40***; 50/25]

Sol: 25+15+30+100\*32/40= 25+15+30+80= **150**

Complexitate: O(n log n)

**Corectitudine:**

1. trebuie sa aratam ca obiectele selectate de noi reprezinta o submultime valida.

E usor de aratat ca suma greutatilor obiectelor nu va depasi niciodata capacitatea rucsacului.

1. Ramane de demonstrat optimalitatea:
   1. Inductie
   2. **Reducere la absurd + “exchange argument”**

Fie ALG - solutia oferita de catre algoritmul descris mai sus

OPT - o solutie diferita de ALG (OPT cea mai apropiata de G solutie optima)

Observatie: Presupunem ca oricare doua obiecte diferite au rapoartele val/greutate diferite.

Deoarece nu conteaza ordinea din rucsac a obiectelor, putem lua OPT ca fiind sortata dupa celasi criteriu ca si ALG.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ALG |  | …. | Gi | …………. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OPT |  | …. | Oi | …………. |

Fie pozitia *i* - prima pozitie pe care OPT difera de ALG. Pe pozitia *i* in ALG se afla Gi in timp ce in OPT se aflta Oi.

Consideram cazul in care grutate de Oi > grutate de Gi . Cazul revers are o justificare aproape identica.4

val/greutate(Oi ) < val/greutate(Gi ) deoarce ALG mereu selecteaza obiectul cu raportul maxim.

Exchange arument

Deoarece grutatea lui Oi este mai mare decat cea a lui Gi - putem alege ca sa eliminam din OPT o fractiune din Oi astfel incat sa il inseram pe Gi in loc.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ALG |  | …. | Gi | …………. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OPT’ |  | …. | Gi | Oi (fractiune) | …………. |

Construim o solutie OPT’ care inlocuieste o bucata din Oi cu Gi .

OPT’ se incadreaza in continuare in capacitatea rucsacului? DA - deoarece am dat deoparte si am adaugat aceiasi cantitate.

Dar OPT’ va avea o valoare totala mai buna decat OPT

Noi facut un “exchange”. Este valid?

DA.

2) 1/0 knapsack problem

Apare o noua constrangere: obiectele trebuie adaugate integral in rucsac, sau lasate pe dinafara.

Ipoteza de lucru:

1. Fiecare obiect are o greutate <= capacitate rucsacului
2. Suma greutatilor obiectelor > capacitatea rucsacului
3. Pot exista mai multe obiecte cu acelasi raport value/weight.

Ne mai intereseaza oare value/wight?

W= 50

Obiectele: [60/10; 100/20; 120/30]

Solutie Greedy: 160

Solutia Oprima: 220

Reamintim de la curs: Acest algoritm (usor modificat) este un algoritm ½ -aproximativ

Solutie exacta pt aceasta problema

sol 1)

Recursivitate naiva: <https://www.onlinegdb.com/HJVoGx_b_>

Complexitate: O(2^n) - este foarte proasta

sol 2) programare dinamica

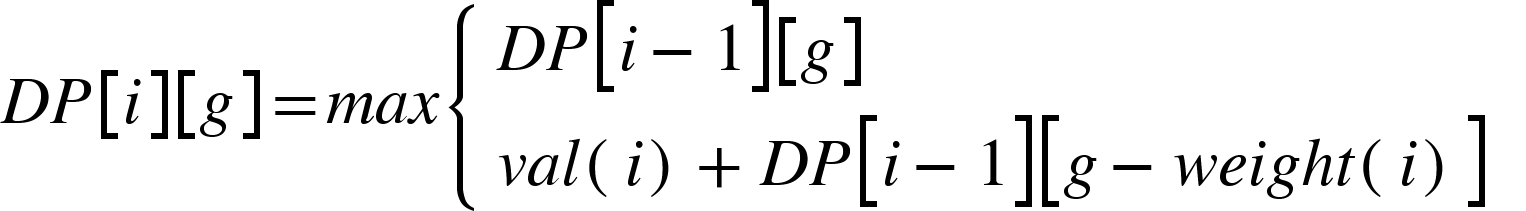
Abordarea consta in urmatorii pasi - pentru a afla valoarea maxima obtinuta din alegerea dintre *n* obiecte fara a depasi W, va trebui sa gasim toate solutiie optime in cazul in care alegem dintre primele *i* obiecte, fara a depasi greutatea *g*. Vom “matura” cu i, respectiv g toate valoarile de la 0 la *n*, respectiv W.

DP[i][g] - valoarea maxima ce poate fi obtinuta selectand din primele *i* obiecte fara a depasi greutatea *g*.

Valori initiale:

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

Ne intereseaza sa gasim DP[n][W]. - aici gasim solutia pt problema noastra



Complexitate:

O(n\*W) - pseudo-polinomiala

Avem posibila o astfel de abordare datorita faptului ca greutatile sunt numere intregi!

--------------------------------------------------------------------------

