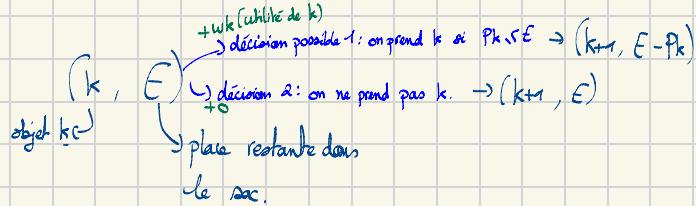


Programmation dynamique.

Représenter une solution réalisable comme une suite de décision prise sur les objets.

Qd on arrive à l'objet k , pour prendre une décision, il faut connaître la somme du sac.



$$0 \leq E \leq P$$

Au début, on a : $(1, P)$
Etat initiale

Un sac réalisable = 1 chemin partant du sommet $(1, P)$ et allant jusqu'au sommet de type $(n+1, E)$ avec $0 \leq E \leq P$.

Valeur du sac = somme des valeurs des arcs du chemin.

$F_k(E)$ = valeur maximale d'un chemin de (k, E) à un sommet $(n+1, E')$.

Notre problème est : calculer $F_n(P) \Rightarrow$ trouver le chemin de $(1, P)$ à un sommet qui donne la valeur max d'abord.

Où on prend pas l'objet de pas d'abord.

Par rapport à notre dessin, au classeur : $\begin{cases} F_k(E) = \max_{\text{arcs}} (w_k + F_{k+1}(E - P_k), 0 + F_{k+1}(E)) \end{cases} \Rightarrow$ récurrence.

Arrivée ici : $F_n(E)$ (dernier objet). Quel est la meilleure décision possible en fonction de E ?

$$F_n(E) = \begin{cases} f_{nn} & \text{si } P_n \leq E \\ 0 & \text{sinon} \end{cases}$$

Application sur les objets suivants:

Complexité, ordre de grandeur: $O(n(P+1))$. Pour la méthode arborescente $O(n^2)$

P_i	1	2	3	4	5
w_i	5	2	4	3	2
$P=10$	50	16	20	33	8

i	1	2	3	4	5
0	0	0	0	0	0
1	0	0	0	0	0
2	16	16	8	8	8
3	33	33	33	33	8
4	33	33	33	33	8
5	50	45	44	44	8
6	50	49	44	44	8
7	66	57	53	44	8
8	83	77	53	44	8
9	83	63	61	44	8
10	93	63	61	44	8

Pour remplir le tableau on doit calculer $f_n(E)$. n vaut 5.

$$\text{Ensuite } F_4(E) = \begin{cases} \text{si } E \geq 3 \\ \max(w_k + f_{k+1}(E - P_k), f_{k+1}(E)) \\ \text{max } (33 + f_5(E-3), f_5(E)) \end{cases}$$

$$\text{max entre } 33 + f_5(E-3) \text{ donc } 8 \text{ sinon: } f_5(E)$$

$$\text{Pour } F_3(E)$$

Algo de programmation dynamique pour le problème du sac à dos (Pétabilité max).

Tableau T_F avec n colonnes, $P+1$ lignes, initialisé à zéro.

Case (E, n) du tableau T_F $\leftarrow T_F [E, n]$ contient $F_{\max}(E)$

3 phases à suivre :

1) Pour E de P_n à P dernière colonne.

$$T_F [E, n] = w_n$$

2) Pour k de $n-1$ à 1

pour E de 0 à P_{k-1}

$$T_F [E, k] = T_F [E, k+1]$$

pour E de P_k à P

$$T_F [E, k] = \max \left[w_k + T_F [E - P_k, k+1], T_F [E, k+1] \right]$$

3) Retrouver les objets pris.

$$E = P$$

pour k de 1 à $n-1$

$$\text{si } T_F [E, k] = T_F [E, k+1]$$

$\rightarrow k$ n'est pas dans le sac optimal.

sinon

$\rightarrow \begin{cases} k \text{ est dans le sac optimal} \\ E = E - P_k \end{cases}$

Si $T_F [n, E] = 0$

$\rightarrow n$ n'est pas dans le sac optimal

sinon

$\rightarrow n$ est dans le sac optimal

Nouvel exemple:

	1	2	3	4	5
P _i	3	4	2	6	3
W _i	10	9	8	7	6
P = 9.					

$$\begin{cases} \text{si } P_k \leq E \\ F_k(E) = \max (w_k + f_{k+1}(E-P_k), \alpha + f_{k+1}(E)) \\ \text{sinon} \\ F_{k+1}(E). \end{cases}$$

E _k	1	2	3	4	5
0	0	0	0	0	0
1	0	0	0	0	0
2	8	8	8	0	0
3	10	8	8	6	6
4	10	9	8	6	6
5	18	14	14	6	6
6	18	17	14	7	6
7	19	17	14	7	6
8	24	17	15	7	6
9	27	23	15	13	6

: étape 3.

Pour $F_1(E)$:

si $P_1 \leq E \Leftrightarrow 3 \leq E$:

$$\max (-10 + f_2(E-3), f_2(e))$$

sinon

$$F_2(e)$$

$$\max (-10 + 17, 23)$$

Solution optimale: {1, 2, 3}

Concepts généraux de prog dynamique.

① Représentation d'une solution réalisable d'un problème comme une suite finie de décisions.

sac à dos: on a une suite de décisions: 1 décision concerne 1 objet.

On appelle une étape avec 1 décision $\rightarrow 1$ phase.

phase k \rightarrow objet k.

nombre d'étapes appelé: pour n \rightarrow horizon

D_k \rightarrow l'ensemble des décisions possibles à la phase k.

$$D_k = \{\text{on prend } k, \text{ on ne prend pas } k\}$$

② Définir l'état du système à chaque phase :

L'état résume ce qu'on a besoin de connaitre des décisions prises des phases 1 à $k-1$ pour savoir ce qu'on a le choix de décider par la suite.

sac à dos: l'état E_k à la phase k c'est le sac restant dans le sac (le poids restant) et pas la liste des objets mis dans le sac.