Problème du sac à dos

- On dispose de n objets de poids positifs $w_1, w_2, \ldots, w_n$ et de valeurs positives $v_1, v_2, \ldots, v_n$, respectivement.

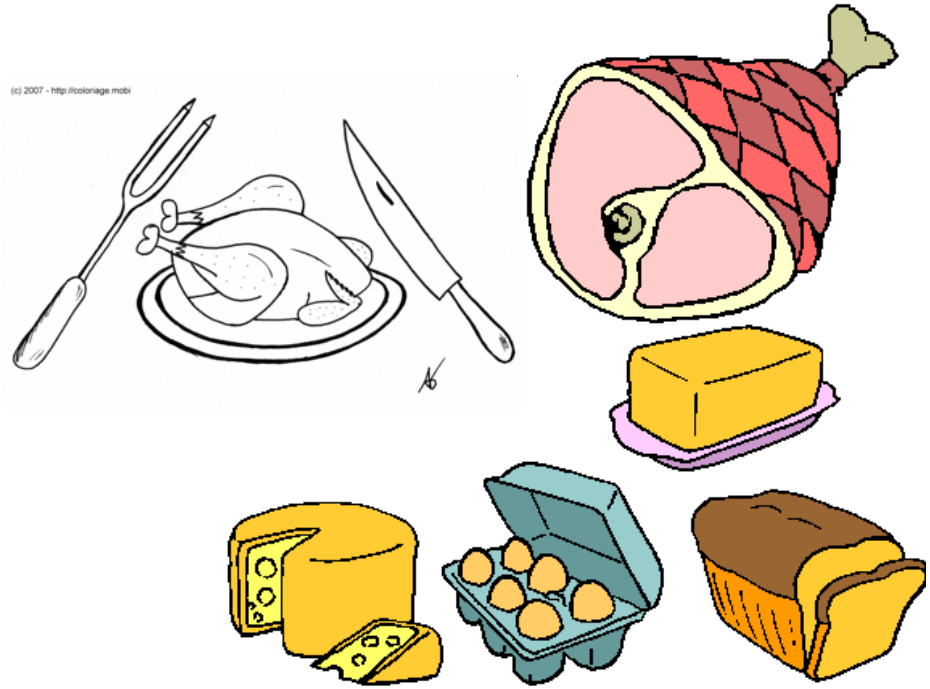
- On a un sac à dos de capacité maximale en poids de W.

Problème: Remplir le sac à dos de sorte de maximiser la valeur des objets inclus tout en respectant la contrainte de poids

http://madamebellefeuille.blogspot.ca/2012/03/dans-mon-sac-dos.html

Problème du sac à dos

- On suppose qu'on peut apporter une fraction x_i de chaque objet i



Problème: Maximiser $\sum_{i=1}^{n} x_i v_i$ tel que $\sum_{i=1}^{n} x_i w_i \leq W$

et $0 \leq x_i \leq 1$

Problème du sac à dos

Stratégie vorace:

- Sélectionner chaque objet à tour de rôle dans un certain ordre
- Mettre la plus grande fraction possible de cet objet dans le sac (sans dépasser la capacité maximale du sac)
- Arrêter quand le sac est plein

Fonctions de sélection possibles:

1) Choisir à chaque étape l'objet de plus grande valeur

2) Choisir à chaque étape l'objet le plus léger

3) Choisir à chaque étape l'objet dont la valeur par unité de poids est maximale

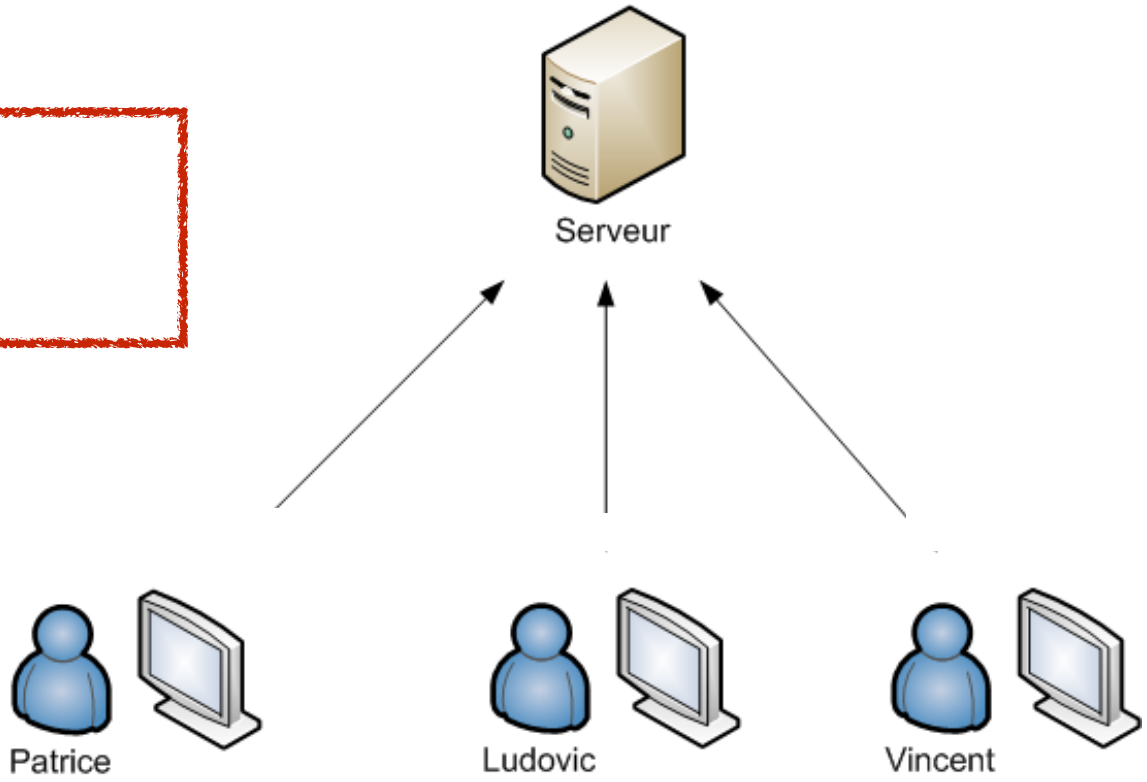
Problème du sac à dos

Théorème: Si les objets sont choisis par ordre décroissant de valeur par unité de poids ($\frac{v_i}{w_i}$), alors l'algorithme du sac à dos vorace trouve une solution optimale.

Problème de la file d'attente simple

- On a un serveur et n clients
- Chaque client i a besoin d'un temps de service t_i
- Si un client i doit attendre, son temps d'attente est dénoté a_i

Problème: Minimiser $\sum_{i=1}^{n} t_i + a_i$



http://web.univ-pau.fr/~puiseux/enseignement/python/tutoQt-zero/qt15/tutoriel-3-11396-0-communiquer-en-reseau-avec-son-programme.html

Problème de la file d'attente simple

Théorème: Si on sert les clients selon un ordre croissant du temps de service demandé, l'algorithme vorace trouver une solution optimale.