Analyse du problème du Sac-à-dos

Denis TRYSTRAM

Notes de cours **Algorithmique avancée**

ENSIMAG 2A - filière Apprentissage, octobre 2016

Définition

KnapSack.
Instance : n objets décrits par des pairs d'entiers (ω_i, b_i) et un entier K (la capacité du sac).
Question : Déterminer un sous-ensemble d'objets qui maximise le bénéfice total (somme des b_i).

On appelle *densité* le rapport qualité-prix : $\delta_i = \frac{b_i}{w_i}$.

Mathématiquement, le problème s'écrit :
$max(\Sigma_{1 \leq i \leq n} b_i x_i)$
sous la contrainte : $\Sigma_{1 \leq i \leq n} \omega_i x_i \leq K$
où x_i est une variable booléenne qui vaut 1 si i est placé dans le sac et 0 sinon.

Complexité

La version décision de KnapSack est un problème NP-complet.

La preuve est simple par une réduction à partir du problème 2Partition dont la définition est rappelée ci-dessous :

2Partition.
Instance : n entiers n_i dont la somme S est paire.
Question : Existe-t-il une partition des entiers en deux sous-ensembles A_1 et A_2 telle que $\Sigma_{i \in A_1} n_i = \Sigma_{i \in A_2} n_i$?

La vérification d'une solution de KnapSack se fait immédiatement en comparant la somme des poids des objets du sac à sa capacité, il est donc dans $\mathcal{NP}$.

La réduction τ est la suivante :

On transforme une instance $\mathcal{I}$ de 2Partition en une instance de KnapSack où $\omega_i = b_i = n_i$ et $K = \frac{S}{2}$.

On vérifie facilement que cette instance est positive *ssi* $\tau(\mathcal{I})$ l'est.

Branch-and-Bound

Ici, on doit préciser la politique de séparation et le calcul des bornes. Le problème est une recherche de maximum, on doit donc chercher une méthode (rapide) pour calculer borne inférieure d'une solution partielle.

- Séparation (on rappelle qu'une bonne politique de séparation est d'obtenir rapidement une solution réalisable avec un bon profit).

  On trie les objets dans l'ordre des densités décroissantes et on explore l'arbre des configrations en distinguant si l'objet courant est dans le sac ou non.

- Evaluation.

  Des bornes inférieures de la solution optimale sont obtenues au fur et à mesure de l'exploration.

  Une borne supérieure du bénéfice est obtenue en additionnant la valeur courante à la meilleure solution *fluide* possible sur les objets restants. Elle sert à éliminer des branches de l'arbre.

Programmation dynamique

Ce problème peut être résolu par programmation dynamique en considérant la relation de sous-optimilité suivante :
$B(1,p) = 0$ si l'objet ne rentre pas$p < \omega_1$ et $B(1,p) = b_1$ sinon
$B(k,p) = B(k-1,p)$ si $p < \omega_k$ et $max(B(k-1,p), B(k-1,p-1) + b_k)$

où $B(k,p)$ est le meilleur bénéfice possible pour sélectionner les k premiers objets dans un sac de capacité p.

Heuristique

L'algorithme glouton qui consiste à prendre un à un les objets dans l'ordre des densités décroissantes semble raisonnable. Cependant, on peut montrer que cette politique est arbitrairement mauvaise en considérant l'instance particulière suivante :

Sac de capacité $K = x$, deux objets $(\omega_1 = 1, b_1 = 1)$ et $(\omega_2 = x - 1, b_2 = x)$.

La densité du premier est plus grande que celle du second, donc c'est celui-là qui est sélectionné, l'autre ne peut plus rentrer dans le sac. Ainsi, le bénéfice est de 1.

Si on avait mis le second, le bénéfice aurait été de x (que l'on peut prendre aussi grand que l'on veut).

Algorithme d'approximation

Nous allons montrer maintenant comment améliorer la solution glouton pour obtenir un algorithme avec une garantie. On considère le premier objet qui ne rentre pas dans le sac en appliquant la politique *qualité-prix*. On note j cet objet. On a alors la propriété suivante :

$OPT \leq \Sigma_{i<j} b_i + b_j$

En effet, on obtient une borne supérieure du bénéfice optimal en considérant la politique continue (fluide) en sélectionnant les meilleurs objets possibles jusqu'à ce que le sac soit plein.

Ceci correspond à : $OPT \leq \Sigma_{i<j} b_i + b'_j$ où b'_j est la partie de l'objet coupé qui sature le sac, elle est bien entendu inférieure à b_j.

On en déduit l'algorithme suivant :

- Sélectionner les objets du sac-à-dos par la politique qualité-prix
- Remplir le sac en considérant les objets dans cet ordre, prenant la plus grande valeur entre le bénéfice obtenu de cette façon et le bénéfice de l'objet suivant j.

Proposition. cette politique est une $\frac{1}{2}$-approximation.

La preuve découle directement de l'inégalité précédente.
Le coût de cette politique est $f = max(b_j, \Sigma_{i<j} b_i)$
comme $2.max(A, B) \geq A + B$, on a bien :
$2f \geq \Sigma_{i<j} b_i + b_j \geq OPT$.