Séparation et évaluation : 0-1 Knapsack

Florian Bourse

On s'intéresse à la résolution du problème d'optimisation du sac-à-dos 0-1 : à partir d'un ensemble d'objets définis par une valeur v_i et un poids w_i, on veut maximiser la valeur des objets emportés en ayant une contrainte sur le poids maximal W que l'on peut emporté. Plus formellement, le problème est défini par le problème d'optimisation linéaire en nombres entiers (integer linear programming) suivant :

0-1-knapsack :

Instance : un ensemble de couples $(w_i, v_i)_{i \in [\![0;k-1]\!]}$ de flottants, et un flottant W.
Solution : un ensemble de booléens $(x_i)_{i \in [\![0;k-1]\!]} \in \{0; 1\}^k$ tels que

$$\sum_{i=0}^{k-1} x_i w_i \leq W.$$

Optimisation : maximiser

$$\sum_{i=0}^{k-1} x_i v_i$$

On considère aussi la relaxation continue du problème, dans laquelle on s'autorise à prendre n'importe quelle fraction d'un objet.

0-1-knapsack*, relaxation continue :

Instance : un ensemble de couples $(w_i, v_i)_{i \in [\![0;k-1]\!]}$ de flottants, et un flottant W.
Solution : un ensemble de flottants $(x_i)_{i \in [\![0;k-1]\!]} \in [0; 1]^k$ tels que

$$\sum_{i=0}^{k-1} x_i w_i \leq W.$$

Optimisation : maximiser

$$\sum_{i=0}^{k-1} x_i v_i$$

Question 1. Montrer qu'une solution $\mathbf{x}^*$ de 0-1-knapsack* est aussi une solution de 0-1-knapsack. En déduire que la solution optimale $\mathbf{x}^*_{\text{opt}}$ de 0-1-knapsack* est nécessairement meilleure que la solution optimale $\mathbf{x}_{\text{opt}}$ de 0-1-knapsack.

Question 2. Écrire une fonction qui permet de trier une liste de couples $(w_i, v_i)_{i \in [\![0;k-1]\!]}$ par valeurs $\frac{v_i}{w_i}$ décroissante. Quelle est la complexité de ce tri ?

Question 3. Proposer un algorithme glouton résolvant le problème 0-1-knapsack* et dont la complexité est dominée par celle du tri de la question précédente.

Question 4. Écrire une fonction qui prend en entrée une instance $\big((w_i, v_i)_{i\in[\![0;k-1]\!]}, W\big)$ de 0-1-knapsack* et qui renvoie la valeur optimale $V_{\text{opt}} = \sum_{i=0}^{k-1} x^*_{\text{opt},i} v_i$ d'une solution optimale. On pourra supposé que la liste des (w_i, v_i) est donnée par ordre de $\frac{v_i}{w_i}$ décroissant.

On va maintenant implémenté un algorithme de recherche exhaustive avec retour-sur-trace pour résoudre le problème 0-1-knapsack.

Question 5. Écrire une fonction récursive qui résoud une instance de 0-1-knapsack par retour-sur-trace.

On veut maintenant améliorer notre recherche grâce à un algorithme de type séparation et évaluation (branch and bound). On va utiliser la résolution de 0-1-knapsack* pour éliminer les branches qui n'ont pas besoin d'être explorées.

Question 6. Écrire une fonction qui résoud une instance de 0-1-knapsack par séparation et évaluation.