Analyse des algos gloutons d'approximation pour bin packing.

Next fit

12 Soit x_i le premier éléments inséré dans la boite B_{i+1} alors $v_i + x_i > C$ et $x_i \leq v_{i+1}$ donc $v_{i+1} + v_i > C$ et ceci pour tout $0 \leq i \leq m-2$.

13 Soit $V = \sum_{i=0}^{m-1} v_i$ qui correspond au volume total occupé par les objets. On a $\underline{C m^* \geq V}$.

or d'après la question précédente:

$$\sum_{i=0}^{m-2} (v_i + v_{i+1}) > (m-1) C$$

$$\left(2 \sum_{i=0}^{m-1} v_i \right) - v_0 - v_{m-1}$$

or v_0 et $v_{m-1} > 0$ donc on obtient

$$2 C m^* \geq 2V > (m-1) C \quad \Rightarrow \quad 2m^* > m-1$$
$$\Rightarrow \underline{2m^* \geq m}$$

On a bien montré que Next-Fit fournit une 2-approximation au problème bin packing.

First Fit decreasing

14

$$x > C/2$$

Tous les objets insérés dans les boites B_i avec $i < j$ ont un poids supérieur ou égal à x et sont donc tous seuls dans leurs boites respectives et ne peuvent pas être

ensemble dans une même boite. Ainsi il y a au moins $j+1$ objets de poids $> \frac{c}{2}$ et ainsi toute solution, en particulier la solution optimale, devra utiliser au moins $j+1$ boites.

$$m^* \geqslant j+1 \geqslant \frac{2m}{3} \implies m^* \frac{3}{2} \geqslant m.$$

15 Supposons que $x \leqslant \frac{c}{2}$:

x étant de poids maximal dans B_j, c'est le premier elt à y être inséré et tous les elts insérés ds les B_k avec $k \geqslant j$ sont donc considérés après et ont donc un poids $\leqslant \frac{c}{2}$. Ainsi chaque boîte B_k avec $k \geqslant j$ peut contenir au moins 2 elts et en dehors éventuellement de la dernière, elles vont effectivement contenir au moins deux éléments (sinon on n'ouvrirait pas la suivante). De plus, ces éléments ne sont pas insérés dans les B_k avec $k < j$ donc ils ont chacun un poids $> v$

Ainsi, $\forall i \in [\![j, m-2]\!]$, $v_i > 2v$

et $v_{m-1} > v$ donc : $\sum_{i=j}^{m-1} v_i > 2v(m-1-j) + v$

16 Comme précédemment :

$$C m^* \geqslant V = \sum_{i=0}^{m-1} v_i = \sum_{i=0}^{j-1} v_i + \sum_{i=j}^{m-1} v_i$$

$$> \sum_{i=0}^{j-1} v_i + 2v(m-1-j) + v$$

or, $\forall i \in [\![0, j-1]\!]$, $v_i \geqslant C - v$ donc :

$$C m^* > j(C-v) + 2v(m-1-j) + v$$

donc:

$$m^* C > jC + v(2m - 3j - 1)$$

cas 1: Si $m \equiv 0 [3]$ alors $3j = 2m$ et on obtient

$$m^* > j - \frac{v}{C} \quad \text{avec} \quad 0 \leq \frac{v}{C} \leq 1$$

$$> j - 1$$

$$\text{donc} \quad m^* \geq j = \frac{2m}{3} \Rightarrow \frac{3}{2} m^* \geq m$$

cas 2: Si $m \not\equiv 0 [3]$ alors $3j \leq 2m - 1$ donc

$$m^* C > jC \Rightarrow m^* \geq j + 1 \geq \frac{2}{3} m.$$

17 Soit $x_0, \dots, x_{n-1}$ une instance de Partition. Supposons, par l'absurde, qu'il existe un algo polynomial qui fournit une $(\frac{3}{2} - \varepsilon)$ approximation pour Bin Packing

On va alors utiliser cet algorithme avec les poids $(2x_0, \dots, 2x_{n-1})$, $C = \sum_{i=0}^{n-1} x_i$

Si l'algorithme nous renvoie $k = 2$ alors on est sûr qu'on avait une instance positive de partition.

Si l'algorithme nous renvoie $k \geq 3$ alors on sait que le nb optimal de boîtes $k^* \times (\frac{3}{2} - \varepsilon) \geq k \geq 3$ et ainsi $k^* > 2$ ce qui implique que notre instance était négative pour partition.

Ainsi, s'il existe $\varepsilon > 0$ pour lequel on a un algo polynomial qui fournit une $\left(\frac{3}{2} - \varepsilon\right)$ approximation au problème Bin Packing alors on en a déduit un algo polynomial qui résout exactement le problème Partition dont on a montré qu'il était NP-complet et dans ce cas, on déduirait que P = NP.