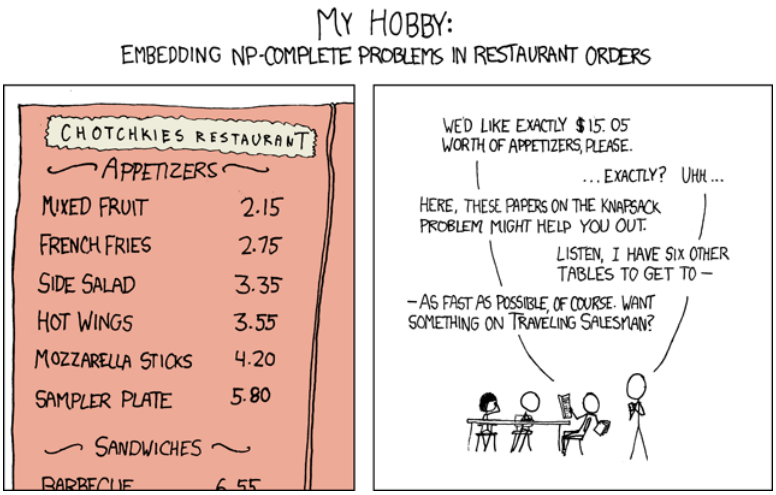
MPI* Info

# TD Bin Packing




Olivier Caffier

**1** Donner une solution optimale pour BINPACKING sur l'instance suivante : $C = 10$ et $X = 2,5,4,7,1,3,8$.

**Corrigé :** Il vient $k = 3$ avec

$$B_0 = \{2,8\}$$
$$B_1 = \{5,4,1\}$$
$$B_2 = \{7,3\}$$

# 1 Caractère NP-complet

**2** Définir le problème de décision BPD associé au problème d'optimisation BINPACKING.

**Corrigé :**

**BPD**

**Entrée :** $C \in \mathbb{N}, X = x_0, \ldots, x_{n-1}, k \in \mathbb{N}$
**Sortie : true** ssi il existe une partition de $X$ en $B_0 \sqcup \ldots \sqcup B_{p-1}$ tq $\forall i, \sum_{x \in B_i} x \leq C$ et $p \leq k$

**3** On considère le problème suivant :

**PARTITION**

- **Instance :** $n$ entiers naturels $x_0, \ldots, x_{n-1}$.
- **Question :** Existe-t-il $I \subseteq [\![0, n-1]\!]$ tel que $\sum_{i \in I} x_i = \sum_{i \notin I} x_i$ ?

Montrer que PARTITION est NP-complet.

**Corrigé :**

**D'une part,**

— On définit $v : X = \{x_0; x_{n-1}\}, < I > \longmapsto$ **true** $\Leftrightarrow \sum_{i \in I} x_i = \sum_{i \notin I} x_i$
— Le problème de décision associé à $v$ est dans $P$
— $|< I >|$ est polynomiale en $\#X$
— Il vient

$$X \in \text{PARTITION} \Leftrightarrow \text{il existe } I \subset [|0; n-1|] \text{ tq. } \sum_{i \in I} x_i = \sum_{i \notin I} x_i$$
$$\Leftrightarrow \exists < I > \in \Sigma^* \text{ tq. } v(X, < I >) = \textbf{true}$$

d'où PARTITION $\in NP$

**D'autre part,** considérons la transformation suivante :

$$\varphi : x_0; \ldots; x_{n-1}, s \longrightarrow x_0; \ldots; x_{n-1}, 2s, \sum_{i=1}^{n-1} x_i$$

- $\Rightarrow$ : Supposons que il existe $B \subset \{x_0; \ldots; x_{n-1}\}$ tq. $\sum_{x \in B} x = s$.
  Notons $S = \sum_{i=0}^{n-1} x_i$. Ainsi,

$$\boxed{S + \sum_{x \in B} x = s + S}$$

et

$$\sum_{x \notin B} x = \sum_{i=0}^{n-1} x_i - \sum_{x \in B} x$$
$$= S - s$$

En ajoutant $2s$ des deux côtés, il vient

$$\boxed{2s + \sum_{x \notin B} x = S + s}$$

D'où

$$\boxed{\boxed{S + \sum_{x \in B} x = 2s + \sum_{x \notin B} x}}$$

Ainsi, on prend

$$I = \{i \in [|0; n-1|] \mid x_i \in B\} \cup \{n+1\}$$
$$I^c = \{i \in [|0; n-1|] \mid x_i \notin B\} \cup \{n\}$$

*Remarque :* Ici, $n$ et $n+1$ sont les indices de $2s$ et $S$ lorsqu'ils sont considérés par le problème PARTITION.

- $\Leftarrow$ : Supposons qu'il existe $I \subset [|0; n+1|]$ tq. $\sum_{i \in I} x_i = \sum_{i \notin I} x_i$ (avec $x_n = 2s, x_{n+1} = S$). Notons $I' = I \cap [|0; n-1|]$.
  - 1er cas : $n, n+1 \in I$. Alors

    $$S + 2s + \sum_{i \in I'} x_i = \sum_{i \notin I'} x_i$$

    or $2s + \sum_{i \in I'} x_i > 0$ et $\sum_{i \notin I'} x_i \leq S$.

    ABS
  - 2ème cas : $n, n+1 \notin I$

    → idem
  - 3ème cas : $n \in I, n+1 \notin I$ (l'autre cas est analogue).
    Ainsi,

    $$S + \sum_{i \in I'} x_i = 2s + \sum_{i \notin I'} x_i$$
    $$\text{d'où } S + \sum_{i \in I'} x_i = 2s + \sum_{i=0}^{n-1} x_i - \sum_{i \in I'} x_i$$
    $$\text{d'où } S + 2\sum_{i \in I'} x_i = 2s + S$$

    Finalement,

    $$\boxed{\boxed{\sum_{i \in I'} x_i = s}}$$

    On prend donc $B = I' \subset [|0; n+1|]$.

    D'où le résultat voulu.

  D'où SubsetSum $\leq_p$ PARTITION

Ainsi, PARTITION $\in NP$ et SubsetSum $\leq_p$ PARTITION (avec SubsetSum $NP$-complet) ce qui permet de conclure.

**4** En déduire que BPD est NP-complet.

**Corrigé :**

**D'une part,**

— On définit $v : C, X, < B_0, \ldots, B_{p-1} >, k \longmapsto$ **true** $\Leftrightarrow \forall i, \sum_{x \in B_i}$ et $p \leq k$

— $| < B_0, \ldots, B_{p-1} > |$ est polynomiale en $\#X$

— Il vient

$$X, k \in BPD \Leftrightarrow \ldots$$
$$\Leftrightarrow \exists P =< B_0, \ldots, B_{p-1} > \in \Sigma^* \text{ tq. } v(C, X, P, k) = \textbf{true}$$

Donc BPD $\in NP$

**D'autre part,** considérons la construction suivante

$$\psi : x_0, \ldots, x_{n-1} \longrightarrow X = x_0, \ldots, x_{n-1}, C = \lfloor \frac{\sum_{i=0}^{n-1} x_i}{2} \rfloor, k = 2$$

- $\underline{\Rightarrow}$ : supposons qu'il existe $I \subset [|0; n-1|]$ tq $\sum_{i \in I} x_i = \sum_{i \notin I} x_i$.

  Alors, il est immédiat que

$$\sum_{i \in I} x_i = \lfloor \frac{\sum_{i=0}^{n-1} x_i}{2} \rfloor$$

$$\text{d'où } \sum_{i \in I} x_i \leq \lfloor \frac{\sum_{i=0}^{n-1} x_i}{2} \rfloor$$

  De même pour $I^c$. On a bien notre partition :

$$B_0 = \{x_i \mid i \in I\}$$
$$B_1 = \{x_i \mid i \not\in I\}$$

→ OK

- $\underline{\Leftarrow :}$ Supposons qu'il existe une partition de $X$ en $B_0 \sqcup B_{p-1}$ tq. $\forall i, \sum_{x \in B_i} x \leq C$ et $p \leq 2$.
  - $\underline{\text{1er cas :}}$ $p = 1$, impossible (car on ne respecterait pas notre inégalité avec la partie entière).
  - $\underline{\text{2ème cas :}}$ $p = 2$. On a

$$\begin{cases} \sum_{x \in B_0} x + \sum_{x \in B_1} x & = \sum_{i=0}^{n-1} x_i \\ \sum_{x \in B_0} x & \leq \lfloor \frac{\sum_{i=0}^{n-1} x_i}{2} \rfloor \\ \sum_{x \in B_1} x & \leq \lfloor \frac{\sum_{i=0}^{n-1} x_i}{2} \rfloor \end{cases} \implies \sum_{x \in B_0} x = \sum_{x \in B_1} x$$

→ OK

Donc PARTITION $\leq_p$ BPD

On a ainsi BPD $\in NP$ et PARTITION $\leq_p$ BPD (avec PARTITION $NP$-complet) ce qui permet de conclure.

# 2 Stratégie gloutonnes

# 3 Analyse de *next-fit*

**5** Montrer pour $i \in [\![0, m-2]\!]$, $v_i + v_{i+1} > C$.

**Corrigé :**

Supposons que $v_i + v_{i+1} \leq C$

Alors d'après le principe de la méthode *next-fit*, tous les objets de la boîte $B_{i+1}$ sont en réalité dans la boîte $b_i$. Ce qui est donc absurde.

**6** En déduire que *next-fit* fournit une 2-approximation pour BINPACKING.

**Corrigé :**

On a

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

or,

$$V = \sum_{i=0}^{m^*-1} v_i \leq \sum_{i=0}^{m^*-1} C = m^* C \tag{1}$$

Or d'après la **5**,

$$\sum_{i=0,i\text{pair}}^{m-2} v_i + v_{i+1} = 2V - v_0 - v_{m-1} > (m-1)C$$

et étant donné que $v_0, v_{m-1} > 0$. on a bien

$$2V > (m-1)C$$

Ainsi (1) nous permet de conclure :

$$\begin{aligned} 2m^*C \geq 2V > (m-1)C \\ \text{i.e } 2m^* > m-1 \\ \text{i.e } m^* \geq \frac{m}{2} \end{aligned}$$

D'où le résultat voulu

# 4 Analyse de *first-fit-decreasing*

**Cas** $x > C/2$

**7** Montrer que $m \leq \frac{3}{2}m^*$

**Corrigé :**

On déduit plusieurs choses du fait que $x > \frac{C}{2}$, notons $k_0$ l'indice de $x$ dans les $(x_k)_{0\leq k\leq n-1}$ :

1. Tous les poids d'indice $k < k_0$ dans rangés **individuellement** dans les boîtes d'indice $i < j$ parce qu'ils sont examinés avant $x$ vu que la liste est triée par ordre décroissant.
2. On utilise donc **au moins** $j+1$ boîtes (car $j+1$ poids $> \frac{C}{2}$)

On en déduit que toute solution au problème, en particulier la solution optimale, utilisera au moins $j+1$ boîtes. Or, par propriété de la partie entière :

$$j = \lfloor \frac{2m}{3} \rfloor \leq \frac{2m}{3} \leq j+1$$

Ainsi,

$$\begin{aligned} m^* \geq j+1 \geq \frac{2}{3}m \\ \text{d'où } m^* \geq \frac{2}{3}m \end{aligned}$$

D'où le résultat recherché

**Cas** $x \leq C/2$ Plusieurs remarques :

1. Tous les éléments $y$ contenus dans les boîtes qui vont de $B_j$ à $B_{m-1}$ vérifient que $y \leq x \leq \frac{C}{2}$.
   Donc chaque boîte de $B_j$ à $B_{m-2}$ contiendra au moins deux éléments.
2. Ces éléments $y$ vérifient aussi que $y > v$ car sinon l'algo aurait choisi de les mettre dans une des boîtes $B_0, \ldots, B_{j-1}$.

**8** Montrer que $\sum_{l=j}^{m-1} v_i > v + 2v(m-j-1)$.

**Corrigé :** En reprenant ces deux remarques :

$$\begin{aligned} \sum_{l=j}^{m-1} v_i &= \sum_{l=j}^{m-2} v_i + v_{m-1} \\ &> \sum_{l=j}^{m-2} 2v + v_{m-1} \\ &> v + 2v(m-j-1) \end{aligned}$$

**9** Conclure.

**Corrigé :** Ainsi,

$$\begin{aligned} Cm^* &\leq \sum_{i=0}^{m-1} v_i \\ &\leq \sum_{l=0}^{j-1} v_i + \sum_{l=j}^{m-1} v_i \\ &> \sum_{l=0}^{j-1} \underbrace{v_i}_{\leq C-v} + v + 2v(m-j-1) \\ &> j(C-v) + v + 2v(m-j-1) \\ &> v(2m-2-2j+1-j) + jC \\ &> jC + v(2m-3j-1) \end{aligned}$$

On a $2m-3j \geq 0$ (définition de la partie entière).

Sans le "-1", on aurait :

$$\begin{aligned} Cm^* > jC &\implies m^* > j \\ &\implies m^* \geq j+1 \geq \frac{2m}{3} \end{aligned}$$

ce qui nous donnerait le résultat voulu.

On est alors obligés de faire une distinction de cas :

- **1er cas :** $m \not\equiv 0[3]$ donc $j < \frac{2m}{3}$, ce qui implique que $2m-3j-1 \geq 0$.
  Alors,

$$\begin{aligned} Cm^* > jC &\implies m^* > j \\ &\implies m^* \geq j+1 \geq \frac{2m}{3} \\ &\implies m \leq \frac{3}{2} m^* \end{aligned}$$

- **2ème cas :** $m \equiv 0[3]$ alors $j = \frac{2m}{3}$, donc

$$\begin{aligned} Cm^* > jC - v &\implies m^* > j - \frac{v}{C} \\ &\implies 0 \leq \frac{v}{c} < 1 \\ &\implies m^* > j-1 \\ &\implies m^* \geq j = \frac{2m}{3} \\ &\implies m \leq \frac{3}{2} m^* \end{aligned}$$

## 5 Difficulté de l'approximation

**10** Soit $\varepsilon > 0$. Par réduction de PARTITION, montrer que s'il existe un algorithme fournissant en temps polynomial une $(\frac{3}{2} - \varepsilon)$-approximation pour BINPACKING, alors $P = NP$.

**Corrigé :** Supposons qu'il existe un tel algorithme $\mathcal{A}$ fournissant en temps polynomial une $(\frac{3}{2} - \varepsilon)$-approximation pour BINPACKING.

Déduisons-en un algo polynomial qui résout PARTITION.

Soit $(x_0, \ldots, x_{n-1})$ une entrée pour PARTITION, on construit l'entrée $(2x_0, \ldots, 2x_{n-1})$ et $C = \sum_{i=0}^{n-1} x_i$ pour $\mathcal{A}$.

- Si $\mathcal{A}$ renvoie 2 boites, alors on renvoie vrai.
- Sinon, on renvoie faux.

➢ Il faut montrer que :

1. si $(x_0,\dots,x_{n-1})$ est une instance positive, alors $\mathcal{A}$ appliqué à $(2x_0,\dots,2x_{n-1})$ et $C=\sum_{i=0}^{n-1} x_i$ va renvoyer 2.
2. sinon, il renverra une valeur $>2$.

DÉMONSTRATION.

1. Dans le cas $m^*=2$, $\mathcal{A}$ renvoie un entier entre 2 et $(\frac{3}{2}-\varepsilon)2<3$, ce qui fait bien 2.
2. Dans ce cas, toute solution optimale utilise au moins 3 boites et donc $\mathcal{A}$ renverra une valeur $\geq 3$.

Bientôt Décembre !