# Algorithmes et Complexité
# Projet - Partie 2

## Description du problème

Dans ce projet on s'interesse au problème de Bin-packing horizontal. On dispose de $n$ boites contenant des objets fragiles qu'on souhaite emballer dans des cartons. Les objets ne peuvent pas être empilés les uns sur les autres, mais peuvent être déposés cote-à-côte dans les cartons. De plus, par soucis de poids, une seule rangé d'objets est autorisée dans chaque carton. Chaque objet $i$ ($i = 1, \ldots, n$) est caractérisé par une hauteur $h_i$ et une largeur $l_i$. Les profondeurs des objets sont identiques, et les embalages (cartons) ont la même profondeur que les objets. On dispose d'un outil qui fabrique des embalages (cartons) sur-mesure avec les dimensions souhaités (largeur et hauteur) pour contenir les objets à mettre dans le même carton.

1. Dans cette partie on suppose que la machine fabrique des cartons d'une largeur fixe $L$, mais la hauteur est réglable, et le nombre de cartons n'est pas limité. Les objets $O_1 \ldots O_N$ sont à ranger dans cette ordre dans les cartons avec les contraintes spécifiques suivantes : (i) $O_1$ doit être le premier dans un carton, et $O_2$ est accolé à $O_1$ ou placé le premier dans un autre carton, et ainsi de suite pour les autres objets, (ii) la largeur totale des objets se trouvant dans le même carton ne doit pas dépasser la largeur fixe $L$ d'un carton. Ainsi un carton contenant les objets $O_i \ldots O_j$, est de dimension $L \times H$, où $H$ est la hauteur maximale des objets $O_i \ldots O_j$. Le problème est de trouver le meilleur placement des objets $O_1 \ldots O_N$ dans les cartons de sorte que la hauteur totale des cartons contenant ces objets soit minimal

$Q_1$. Soit $Tmin(i)$ la hauteur total des cartons de largeur $L$ contenant les objets $O_i \ldots O_N$ ($1 \leq i \leq N$) et $h(i,j)$ la hauteur du carton où sont rangés les objets $O_i \ldots O_j$ ($j \geq i$) on a :

$$h(i,j) = \max_{k \in i \ldots j}(h_k) \text{ si } \sum_{k=i}^{j} l_k \leq L, \text{ et } h(i,j) = +\infty \text{ sinon}$$

Proposer une relation de reccurence pour calculer $Tmin(i)$ en fonction de $Tmin(j{+}1)$ avec $j > i$ et de $h(i,j)$ et préciser l'initialisation de reccurence.

$Q_2$. En déduire l'algorithme de programmation dynamique pour calculer $Tmin(1)$ et donner la complexité de cet algorithme

$Q_3$. Programmer cet algorithme (le choix du langage vous appartient) et tester votre programme sur l'exemple suivant avec $L = 4$.

| I | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| $l_j$ | 1 | 2 | 1 | 1 | 2 | 1 | 3 | 2 |
| $h_i$ | 1 | 2 | 5 | 4 | 3 | 1 | 2 | 4 |

2. On suppose maintenant que les objets $O_1 \dots O_N$ ont la même hauteur. On souhaite ranger les objets $O_1 \dots O_N$ dans $K$ cartons ($K$ une donnée) de largeur réglable et de hauteur fixe identique à celles des objets. L'ordre de placement des objets $O_1 \dots O_N$ dans les cartons est le même que celui décrit dans 1. On cherche à minimiser la plus grande largeur des cartons qui permet de ranger tous les objets $O_1 \dots O_N$ dans $K$ cartons. Autrement dit, il s'agit de former $K$ sous-ensembles d'objets de telle sorte que la largeur du plus large sous-ensemble soit la plus petite possible.
Soit $Lmin(u, v)$ ($1 \leq u \leq N$) la largeur minimale qui permet de ranger les $u$ premiers objets $O_1 \dots O_u$ dans $v$ cartons et on note $l(i, j)$ la somme des largeurs des objets $O_i \dots O_j$ ($l(i, j) = l_i + \dots + l_j$)

$Q_4$. Montrer que, pour tout $u$, avec $1 \leq u \leq N$, on a $Lmin(u, 1) = l(1, u)$.

$Q_5$. Montrer que dans la cas où $u \leq v$ on peut ranger un objet par carton. Etablir la valeur de $Lmin(u, v)$ dans ce cas.

$Q_6$. On s'intéresse maintenant au rangement optimal de $u$ objets dans $v$ cartons, avec $u > v$, où le dernier carton contient les objets $O_j \dots O_u$. Expliquer pourquoi on a, soit $Lmin(u, v) = Lmin(j - 1, v - 1)$, soit $Lmin(u, v) = l(j, u)$

$Q_7$. A partir des questions $Q_4$ - $Q_6$ proposer une relation de récurrence pour calculer $Lmin(N, K)$.

$Q_8$. Décrire l'algorthme de programmation dynamique associé, sa complexité et préciser l'initialisation

$Q_9$. Proposer une implementation de l'algorithme et l'appliquer sur l'exemple suivant avec $K = 3$

| I | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| $l_j$ | 1 | 2 | 1 | 1 | 2 | 1 | 3 | 2 |

3. On conserve ici l'hypothèse où les objets $O_1 \dots O_N$ ont la même hauteur. On dispose maintenant de $K$ cartons identiques de largeur fixe $L$ (donnée) et d'une hauteur identique à celle des objets. On suppose que tous les objets ne peuvent

pas être rangés dans les $K$ cartons ($\sum_{j=1}^{N} l_j > K \times L$) et on souhaite ranger le maximum objets $W$ à determiner, parmi $N$, tout en rangeant les objets comme décrit dans 1.
Soit $Lns(i, j)$ la largeur minimal qu'il faut pour le rangement d'un sous-ensemble ordonné de $j$ objets parmi $i$ objets. Les $j$ objets peuvent être repartis sur plusieurs cartons.

$Q_{10}$. On suppose qu'il y a un seul carton ($K = 1$). Quelle est la combinatoire du problème ?

$Q_{11}$. Donner une formule de reccurence pour calculer $Lns(i, j)$ dans le cas $K = 1$. et l'initialisation de la formule par le calcul de $Lns(i, 1$ et de $Lns(i, i)$. En déduire l'algorithme de programmation dynamique associé et la complexité de cet algorithme. Préciser la façon de déterminer la valeur $W$ cherchée.

$Q_{12}$. Proposer une programmation de l'algorithme et traiter l'exemple suivant avec $L = 5$

| I | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| $l_j$ | 1 | 2 | 1 | 1 | 2 | 1 | 3 | 1 |

$Q_{13}$. On considère ici le cas $K = 2$. Donner la formule de recurrence pour calculer $Lns(i, j)$. En déduire le principe de l'algorithme de programmation dynamique associé, la façon de déterminer la valeur $W$ cherchée et la complexité de cet algorithme.

$Q_{14}$. Proposer une programmation de l'algorithme et traiter l'exemple suivant avec $L = 5$.

| I | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| $l_j$ | 1 | 2 | 1 | 2 | 2 | 1 | 3 | 2 |

## Travail à rendre

Votre travail doit contenir les éléments suivants:

- Un rapport PDF comportant les réponses aux questions $Q_1$, $Q_2$, $Q_4$ - $Q_8$, $Q_{10}$ et $Q_{11}$
- Les fichiers sources de vos programmes ($Q_3$, $Q_9$, $Q_{12}$, $Q_{13}$)
- Un fichier texte contenant les résultats des exécutions demandées en $Q_3$, $Q_9$, $Q_{12}$ et $Q_{13}$