Nantes Université — UFR Sciences et Techniques
Master informatique parcours “optimisation en recherche opérationnelle (ORO)”
Année académique 2022-2023

Travaux Pratiques
Optimisation discrète et combinatoire
Xavier Gandibleux, Anthony Przybylski

Sujet
Problèmes de bin-packing

Sujet et consignes pour conduire vos travaux
(4 séances de 2 × 1h20)

1 Introduction

Les problèmes de bin-packing représentent une famille importante de problèmes d'optimisations discrets. Ils se ventilent en trois grandes classes, dites cas unidimensionnel (1D), bidimensionnel (2D) et cas tridimensionnel (3D). Sa dénomination vient du fait que l'on a des objets à ranger avec des contraintes (d'espace occupé propres aux cas 1D, 2D, 3D, ainsi que des contraintes particulières enrichissant un de ces cas de référence) dans le moins de “boîtes” identiques possibles, appelées bins [1].

Ils permettent de modéliser et ainsi de représenter une multitude de situations concrètes. Citons par exemple la sauvegarde de fichiers numériques en utilisant le moins de supports de stockage possibles (1D), le placement de boîtes sur une palette sans superposition de boîtes en utilisant le moins de palettes possibles (2D), le chargement d'un train de marchandise avec des conteneurs en utilisant le moins de wagons possibles (cas 3D).

On va s'intéresser à la résolution de bin-packing avec un algorithme de branch-and-bound (BnB). La littérature rapporte plusieurs algorithmes de branch-and-bound pour le bin-packing 1D. Plusieurs documents de référence support à votre travail sont mis à disposition sur madoc (un papier de synthèse, une présentation de synthèse, des papiers numérotés de 1 à 4 selon la sophistication de l'algorithme qui y est présenté, une thèse de doctorat). Des documents et ressources vous permettant d'aller plus loin sur les sujets abordés sont également proposés.

Le premier algorithme de BnB, et le plus simple, fut proposé par Eilon et Christofides en 1971 (papier 1 sur madoc). Hung et Brown ont proposé en 1978 un algorithme dans le cas où les bins peuvent présenter différentes capacités (papier 2 sur madoc). En 1989, Martello et Toth ont proposé l'algorithme MTP qui est resté durant près de 10 ans l'algorithme de référence dans la classe des BnB pour le bin-packing 1D (papier 3 sur madoc). En 1997, reprenant plusieurs idées fortes de MTP et ajoutant des composants très efficaces notamment fondés sur une recherche tabou, Scholl et al. ont présenté l'algorithme BISON qui est reconnu encore en 2016 (survey 1 sur madoc) pour être le BnB le plus efficace sur ce problème d'optimisation (papier 4 sur madoc).

La mise en œuvre de vos algorithmes se fera en Julia ou C/C++. Du matériel est mis à votre disposition sur github pour conduire vos travaux (`https://github.com/xgandibleux/odcBB1Dstu`) :
— une collection d'instances numériques issues de la OR-library,
— une structure de données (`datastrucBB1D.jl`),
— un parser permettant de lire les instances de la OR-library (`parserBB1D.jl`),
— un point d'entrée principal (`mainBB1D.jl`).

1. les étudiants qui étaient en licence 3 chez nous en 2021-2022 ont eu à travailler sur des bin-packing dans le cadre du projet du cours de “recherche opérationnelle”. Le projet de licence 3 n'est pas un pré-requis au présent travail mais il sera bon de vous souvenir de votre production afin de la situer par rapport aux consignes ci-après.

Activité :
Faire une lecture complète du présent sujet. Prendre connaissance de l'ensemble des documents et ressources mis à disposition sur madoc et sur github afin de rendre efficace leur utilisation au cours de votre travail.

2 Etudes à conduire

2.1 Problème de bin-packing 1D

Considérons un problème de bin-packing 1D (BB1D) dans lequel :

. n est le nombre total d'objets à considérer,
. i est l'indice d'un objet, avec $i = 1, \dots, n$,
. w_i représente l'espace occupé par l'objet i,
. C est la capacité d'un bin.

On suppose que les valeurs w_i sont des entiers positifs, que C est un entier positif, et que $w_i \leq C$, $i = 1, \dots, n$. Les bins sont disponibles en quantité suffisante et présentent tous la même capacité. Aussi, les objets sont indivisibles.

Résoudre le BB1D consiste à ranger chaque objet dans un bin de manière à utiliser le moins de bin possible.

2.2 Instance didactique

Pour ce BB1D prenons l'instance suivante :

n		w_1	w_2	w_3	w_4	w_5	w_6	w_7		C
7		2	3	3	3	4	4	5		6

Table 1 – Instance didactique constituée de 7 objets et de m bins de capacité $C = 6$

Activité :
Construire à la main une solution admissible de bonne qualité pour l'instance didactique.

Indications :
Ce résultat vous servira de borne primale sur la valeur de la solution optimale de cette instance.

Note :
On supposera dans la suite que dans toute solution réalisable, les valeur d'indices de bins les plus petites indiquent les bins utilisés, c'est-à-dire si les m bins sont indicés par $j = 1, \dots, m$, et que l'on associe à chaque bin une variable y_j qui prend la valeur 1 si le bin j est utilisé (0 sinon), on aura $y_j \geq y_{j+1}$ pour $j = 1, \dots, m-1$

2.3 Borne inférieure sur le nombre de bins nécessaires

Dans la perspective d'encadrer la valeur optimale du BB1D, on souhaite disposer d'une borne duale.

Activité :
Proposer l'expression d'une borne duale et donner la valeur obtenue pour l'instance didactique.

Indications :
Imaginer que les objets peuvent être scindés.

2.4 Déduction de la valeur de la solution optimale

A partir de l'encadrement proposé ci-dessus, est-il possible d'établir la valeur de la solution optimale ?

Activité :
Si cela est possible, dériver de l'encadrement la solution optimale de l'instance didactique.

INDICATIONS :
Utiliser des arguments logiques.

2.5 Modélisations MILP

Posons tous les éléments qui vont permettre d'écrire deux modèles représentant le BB1D.

Indices

- i : objet d'indice i
- j : bin d'indice j

Données

- n : le nombre d'objets à ranger dans les bins, avec $i = 1, \ldots, n$
- m : le nombre de bins, avec $j = 1, \ldots, m$
- w_i : l'espace occupé par l'objet i ($i = 1, \ldots, n$)
- C : la capacité d'un bin

Note : En première approche, prenons $m = n$ ce qui assure l'obtention d'une solution réalisable au problème (qui trivialement serait d'affecter un objet à chaque bin, ce qui conduit à une valeur de solution la plus défavorable). Nous discuterons plus tard de l'ajustement de la valeur à donner à m.

Variables de décision

- $x_{ij} = \begin{cases} 1 \text{ si l'objet } i \text{ est placé dans le bin } j \\ 0 \text{ sinon} \end{cases}$
- $y_j = \begin{cases} 1 \text{ s'il y a un objet rangé dans le bin } j \\ 0 \text{ sinon} \end{cases}$

Modèle P1

$$
\left[
\begin{array}{llllll}
\min & z = \sum_{j=1}^{m} y_j & & & & (1.0) \\
s.c. & \sum_{j=1}^{m} x_{ij} & = & 1 & i = 1, \ldots, n & (1.1) \\
 & \sum_{i=1}^{n} w_i x_{ij} & \leq & C y_j & j = 1, \ldots, m & (1.2) \\
 & x_{ij} & \in & \{0, 1\} & i = 1, \ldots, n \quad , \quad j = 1, \ldots, m & (1.3) \\
 & y_j & \in & \{0, 1\} & j = 1, \ldots, m & (1.4)
\end{array}
\right]
$$

Dans ce modèle P1, (1.0) donne le nombre de bins utilisés, (1.1) assurent que chaque objet est assigné à un bin, (1.2) assurent que la capacité du bin n'est pas dépassée par les objets qui lui sont assignés, (1.3-1.4) donnent les contraintes d'intégrités.

Modèle P2

$$
\left[
\begin{array}{llllllll}
\min & z = & \sum_{j=1}^{m} y_j & & & & & (2.0) \\
s.c. & & \sum_{j=1}^{m} x_{ij} & = & 1 & i = 1,\ldots,n & & (2.1) \\
 & & \sum_{i=1}^{n} w_i x_{ij} & \leq & C & j = 1,\ldots,m & & (2.2) \\
 & & x_{ij} & \leq & y_j & i = 1,\ldots,n & , \quad j = 1,\ldots,m & (2.3) \\
 & & x_{ij} & \in & \{0,1\} & i = 1,\ldots,n & , \quad j = 1,\ldots,m & (2.4) \\
 & & y_j & \in & \{0,1\} & j = 1,\ldots,m & & (2.5)
\end{array}
\right]
$$

Dans ce modèle P2, (2.0,2.1,2.4-2.5) ont respectivement la même signification que (1.0,1.1,1.3-1.4) dans P1, (2.2) assurent que la capacité d'un bin n'est pas dépassée par les objets qui lui sont assignés, (2.3) assurent que un objet peut être rangé dans un bin si celui-ci est utilisé.

2.6 Relaxation linéaire

On va s'intéresser à la relaxation linéaire de P1 et de P2, c'est-à-dire pour $i = 1,\ldots,n$ et $j = 1,\ldots,m$:

$$x_{ij} \geq 0, \; x_{ij} \leq 1, \; y_j \geq 0, \; y_j \leq 1$$

Considérons la solution obtenue en prenant les objets dans l'ordre donné par l'énoncé, et lorsque un objet ne peut pas être pris entièrement, il est scindé ; une première partie fractionnaire est prise pour compléter le bin, sa seconde partie fractionnaire est mise dans un nouveau bin. Cette solution s'écrit pour le problème P1 :

$x_{1,1} = 1$				
$x_{2,1} = 1$				
$x_{3,1} = 1/3$	$x_{3,2} = 2/3$			
	$x_{4,2} = 1$			
	$x_{5,2} = 1/4$	$x_{5,3} = 3/4$		
		$x_{6,3} = 3/4$	$x_{6,4} = 1/4$	
			$x_{7,4} = 1$	
$y_1 = 1$	$y_2 = 1$	$y_3 = 1$	$y_4 = 1$	$y_5 = \ldots = y_m = 0$

et $x_{i,j} = 0$ sinon

Pour cette solution, on obtient $z = \sum_{j=1}^{m} y_j = 4$.

Activité :
Vérifier que cette solution est optimale.

INDICATIONS :
Procéder par calcul algébrique.

Activité :
Confirmer votre déduction en calculant la valeur de la solution optimale relaxée du modèle 1 avec JuMP et un solveur LP (GLPK ou HiGHS ou Gurobi ou Cplex).

INDICATIONS :
-

Activité :
Reprendre les valeurs des x_{ij} correspondant à la solution obtenue relâchée ci-dessus (i.e. la solution obtenue en prenant les objets dans l'ordre donné par l'énoncé, objets pouvant être scindés) et déduite les valeurs de y_j dans le cas du modèle P2. Déduire lequel des deux modèles est plus fort que l'autre.

INDICATIONS :
-

Activité :
Confirmer votre déduction en calculant la valeur de la solution optimale relaxée du modèle 2 avec JuMP et un solveur LP (GLPK ou HiGHS ou Gurobi ou Cplex). Vous observerez que bien que les modèles 1 et 2 sont équivalents en entier, ils ne le sont pas en continu.

INDICATIONS :
-

Vous continuerez avec le modèle qui vous sera apparu le plus fort.

2.7 Un algorithme pour calculer la borne primale

Un très grand nombre d'algorithmes approchés ont été proposés dans la littérature pour calculer une solution admissible au BB1D (vous avez rencontré déjà dans le projet de recherche opérationnelle en licence 3 l'un d'entre eux).

Tous nos objets étant connus a priori (situation dite "off-line"), un prétraitement (tri, groupement, etc) peut leur être appliqué. Supposons que les objets sont triés par espace occupé décroissant ($w_i \geq w_{i+1},\ i = 1, \ldots, n-1$). L'une de ces heuristiques gloutonnes permet d'obtenir une solution réalisable et une borne primale :

. *next-fit decreasing* (NFD) :
range l'objet suivant dans le bin actuel si possible, ou dans un nouveau bin lequel devient le bin courant sinon ;

. *first-fit decreasing* (FFD) :
range l'objet suivant dans le bin utilisé de plus petit indice pouvant accueillir, ou dans un nouveau bin si aucun bin utilisé ne peut accueillir l'objet ;

. *best-fit decreasing* (BFD) :
range l'objet suivant dans un bin utilisé qui peut l'accueillir, de manière à laisser le plus petit espace résiduel possible, ou dans un nouveau bin si aucun bin utilisé ne peut accueillir l'objet ;

Activité :
Implémentez ces trois heuristiques et évaluer les résultats qu'elles obtiennent au regard d'une sélection d'instances numériques. Vos résultats seront à rapporter et à analyser dans votre rapport.

INDICATIONS :
Plusieurs collections d'instances numériques existent pour le BB1D, soulignant l'importance accordée par la communauté scientifique à ce problème. Le document "survey 1" disponible sur madoc liste plusieurs sources et discute des caractéristiques de ces instances. Un ensemble d'instances issues de la OR-library (`http://people.brunel.ac.uk/~mastjjb/jeb/orlib/binpackinfo.html`) sont mises à disposition sur github (`https://github.com/xgandibleux/odcBB1Dstu`). Libre à vous de compléter vos expérimentations en prenant d'autres instances de la littérature.

La valeur obtenue à l'issue du calcul de la borne primale permet de donner une valeur à m.

2.8 Un algorithme de branch-and-bound de référence

A présent, nous abordons la résolution de BB1D avec un algorithme de branch-and-bound. Ce premier algorithme vous servira de référence quand ultérieurement il sera question d'apporter des composants supplémentaires à votre algorithme en vue d'augmenter son efficacité. Il sera donc identifié dans la suite par BNB1.

Activité :
Elaborer, mettre en œuvre et évaluer un algorithme de branch-and-bound pour le BB1D.

Indications :
Deux possibilités se présentent à vous :

— proposer votre propre algorithme pour lequel il conviendra de présenter minutieusement chacun de vos choix de façon à ce qu'il soit reproductible ;
— reproduire le premier algorithme proposé dans la littérature par Eilon et Christofides, celui-ci est décrit dans le document "papier 1" disponible sur madoc.

Pensez à instrumenter votre solveur (nombre de noeuds, valeurs des bornes primales et duales, temps consommé, etc.) afin d'en mesurer son activité. Une analyse au regard d'instances numériques est attendue.

Note :
Des choix techniques de mise en œuvre de votre algorithme vont se présenter à vous. A titre d'exemple, lorsque vous ajouterez des contraintes de séparation à un problème ouvert il est naturel de réoptimiser le problème d'optimisation étendu par cette contrainte, et non pas redémarrer complètement une résolution (comme vu en CM et travaillé en TD). Cette étape de réoptimisation est à votre portée avec l'environnement JuMP/MOI utilisé. Cependant elle peut vous demander de descendre dans les couches basses de ces environnements (voir les court-circuiter en attaquant directement le solveur LP). Le focus de ces TP portant prioritairement sur les éléments algorithmiques travaillés, il ne vous sera pas tenu rigueur si vous restez à un niveau élevé d'utilisation des outils mis à votre disposition. Attendez vous toutefois dans ce cas à des performances peu percutantes (pas de secret, le confort de la simplicité se paie à un moment) et par soucis d'équité au moment des comparaisons à venir en présence de tentatives d'améliorations, vous aurez à maintenir la même solution technique.

2.9 Borne inférieure améliorée sur le nombre de bins nécessaires

Appelons $L1$ la borne inférieure que vous avez discuté en section 2.3. Martello et Toth ont proposé $L2$, une borne inférieure améliorée sur le nombre de bins nécessaires.

Activité :
La borne $L2$ est présentée dans le document "papier 3" et est discutée dans le document "survey 1" disponible sur madoc. Implémentez ces deux bornes et évaluer les résultats qu'elles obtiennent au regard d'une sélection d'instances numériques. Vos résultats seront à rapporter et à analyser dans votre rapport.

Indications :
-

2.10 Tentative d'amélioration de BnB1

La borne $L2$ se présente comme un composant vous permettant d'améliorer les performances de votre BnB1.

Activité :
Intégrez la borne $L2$ dans votre algorithme de branch-and-bound et évaluez son impact sur l'activité de votre solveur, comparativement à la borne $L1$.

Indications :
Cette version revue de votre algorithme portera le nom BnB2.

2.11 Coupes

Une autre voie d'action potentielle sur l'efficacité de votre algorithme se situe au niveau de coupes définissant des inégalités valides. Celles-ci peuvent être ajoutées au problème d'optimisation en vue de resserrer la relaxation linéaire. Le sujet est vaste, il ne sera qu'effleuré dans cette étude.

Activité :
Justifier pourquoi dans le modèle P1 la coupe

$$x_{5,j} + x_{6,j} + x_{7,j} \leq 1 \quad \forall j = 1, \ldots, m$$

est une inégalité valide pour l'instance didactique.

INDICATIONS :
Pour amener une justification, (1) repartir de la discussion en section 2.4 qui a permis de déduire la valeur de la solution optimale et (2) reprendre la solution relâchée discutée dans la section 2.6 pour vérifier que cette inégalité est une coupe pour la solution fractionnaire de valeur 4.

Allons un peu plus loin ; la contrainte de sac-à-dos présente dans la modélisation a fait l'objet d'attention particulière dans la littérature et a conduit à la définition de *cover cuts*. Soit une contrainte de sac-à-dos

$$\sum_{j=1}^{n} a_j x_j \leq b$$

Un ensemble $C \subseteq N = \{1, 2, \ldots, n\}$ est une *couverture (cover)* si

$$\sum_{j \in C} a_j x_j > b$$

Une *couverture est minimale (minimal cover)* si $C \backslash \{j\}$ n'est pas une couverture pour n'importe quel $j \in C$.

Si $C \subseteq N = \{1, 2, \ldots, n\}$ est une couverture alors

$$\sum_{j \in C} x_j \leq \mid C \mid -1$$

est une inégalité valide.

∘ Exemple :
pour la contrainte

$$11x_1 + 6x_2 + 6x_3 + 5x_4 + 5x_5 + 4x_6 + x_7 \leq 19$$

1) les indices de variables $\{1, 2, 3\}$ forment une couverture car

$$11 + 6 + 6 = 23 > 19$$

et donc

$$x_1 + x_2 + x_3 \leq 2$$

forme une inégalité valide.

2) les indices de variables $\{3, 4, 5, 6\}$ forment une couverture car

$$6 + 5 + 5 + 4 = 20 > 19$$

et donc

$$x_3 + x_4 + x_5 + x_6 \leq 3$$

forme une autre inégalité valide.

Si $C \subseteq N = \{1, 2, \ldots, n\}$ est une couverture pour $\sum_{j=1}^{n} a_j x_j \leq b$ alors

$$\sum_{j \in E(C)} x_j \leq \mid C \mid -1$$

est une inégalité valide, où $E(C)$ est une *extended cover* définie par

$$E(C) = C \cup \{j \mid \forall i \in C : a_j \geq a_i\}$$

On parle de *stronger cover cut.*

◦ Exemple :
pour l'inégalité valide

$$x_3 + x_4 + x_5 + x_6 \leq 3$$

les coefficients des variables x_1 et x_2
valent respectivement 11 et 6 et
sont $\geq$ aux coefficients des variables x_3 à x_6 (valant respectivement 6, 5, 5, 4)

et donc

$$x_1 + x_2 + \ldots + x_6 \leq 3$$

forme une inégalité valide.

Ces inégalités de couverture sont discutées dans le document "papier A" (disponible sur madoc). D'autres inégalités valides pour le BB1D sont présentées dans le document "survey 1".

2.12 Tentative d'amélioration de BnB2

Les coupes se présentent comme un composant vous permettant d'améliorer les performances de votre BnB2.

Activité :
Partant d'une relaxation linéaire, et pour une ou plusieurs familles de coupes retenues, générer des inégalités valides à adjoindre au programme linéaire à résoudre en vue d'espérer resserrer la valeur de la relaxation linéaire.

INDICATIONS :
Les coupes peuvent être ajoutées à n'importe quel sous-problème correspondant à un noeud ouvert du branch-and-bound. L'attitude attendue est de générer quelques coupes qui seront ajoutées au problème, et d'observer l'effet de celles-ci. Pour des raisons de temps accordé à ce travail, cette étape n'est pas complètement spécifiée et sera exploratoire. Cette version revue de votre algorithme portera le nom BnB3.

2.13 Etude d'une variante du bin-packing 1D

De nombreuses variantes du BB1D peuvent être imaginées. Parmi celles-ci, citons les trois suivantes :

— des bins de capacité différentes ;
— des contraintes d'incompatibilité entre objets dans un bin ;
— des contraintes de fragilité sur les objets.

Activité :
Mener l'étude de l'une de ces variantes. Vous avez la latitude pour définir ce qui mérite d'être investigué ici.

INDICATIONS :
La thèse disponible sur madoc présente notamment les deux dernières variantes mentionnées. Etant arrivé au terme de ce travail, vous avez suivi un cheminement et grandi en maturité sur l'étude et la résolution de problèmes d'optimisation discrets et combinatoires. Faites donc usage de cette expérience pour répondre avec pertinence à cette activité.

3 Livrable

Vos productions seront dument présentées dans un rapport. Ce dernier, ainsi que les implementations réalisées et les instances numériques élaborées seront à déposer sur madoc pour au plus tard le 13 décembre, date ferme.

4 Résultats d'apprentissage

Les étudiants ayant réalisé avec fruit cette étude seront capables :

— de modéliser des programmes linéaires (situation de pures variables continues, entières ou binaires, et situations mixtes) [M]
— d'apprécier la difficulté inhérente à une situation d'optimisation discrète et retenir les principes fondamentaux pour apporter une réponse en terme d'algorithme de résolution compte tenu de ressources (moyens de calcul, temps de calcul et espace mémoire) disponibles. [M]
— d'identifier et exploiter une propriété présente dans un modèle en vue de concevoir un algorithme de résolution efficace [A]
— d'apporter des traitements pertinents en vue de serrer les bornes autour d'une solution optimale d'un problème [A]
— d'apporter de l'information a priori exogène (coupes ; inégalités valides) en vue de faciliter sa résolution [I]
— de concevoir, implémenter, intégrer un solveur ad-hoc fondé sur l'énumération implicite [A]
— d'utiliser un environnement générique de résolution, d'y intégrer les éléments du problème à résoudre et d'obtenir des solutions argumentées [I]
— de juger du bien-fondé de la mise en oeuvre spécifique dans un environnement ad-hoc [A]
— d'utiliser d'un langage de modélisation algébrique (modèle explicite, modèle implicite implicite) [M]
— d'utiliser d'un solveur MIP (utilisation d'un solveur MIP en tant que solveur autonome et bibliothèque de fonctions usitée dans un algorithme donné) et interpréter finement les résultats issus de l'optimisation [M]
— de traiter un cas d'étude complet allant de l'énoncé d'une situation jusqu'aux recommandations en terme de solutions et de paramètres [A]
— Elaborer un document scientifique et technique accompagnant la solution informatique. [M]