MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DE KHEMIS-MILIANA

Faculté des Sciences et de la Technologie

Département de Mathématiques et informatique

MEMOIRE

Présenté par

ACHOUR HENIA

Pour obtenir

LE DIPLOME DE MASTER

Spécialité : Mathématiques

Option : Mathématiques appliquées et Traitement du Signal

Intitulé

Problème de découpe guillotine à deux dimensions non contraint

Dédicaces

Ce mémoire est particulièrement dédiée

Á mes chèrs parents,

Á mon cher frère Omar,

Á mes soeurs Salwa, Hadjira, Fairouze et Samah,

Á mes tante Najia et son mari et tous ses enfants,

Á mes amis Khadra, Zahra, Sherifa,Khadija et Fatima, Fula et Naima.

Remerciements

Je remercie Allah le tout puissant de m'avoir donné la volonté et le courage pour mener à bien ce travail.

Je tien à remercier vivement notre promoteur Mme Ait abdesselam maya pour avoir accepter de diriger ce travail et pour ses précieux conseils et ses encouragements.

Mes vifs remerciements s'adressent à tous les membres de jury Sakri Redha et Kali Abdesselam qui nous ont fait l'honneur d'examiner ce travail.

Je salue et remercie mes parents auxquels je dédie ce travail.

Je remercie également tous mes collègues qui m'ont permis de travailler dans une ambiance excellente.

Merci enfin, à tous ceux qui m'ont aidé durant ces deux années par leurs conseils et leurs encouragements.

Résumé

Le thème abordé dans nos travaux est le problème de la découpe guillotine à deux dimension qui consiste à effectuer des coupes verticales ou horizontales sur un support rectangulaire en allant d'une arête à son opposée tout en étant parallèle aux deux arêtes restantes. Dans ce mémoire nous présentons en premier lieu la définition, la modélisation et les différents types au problème, en suite nous donnons une typologie des problèmes de découpe et d'assemblage. Troisièmement un état de l'art présente les méthodes de résolution exact appuyant sur les méthodes par séparation et evaluation et la programmation dynamique et propose egalement plusieurs méthodes de résolution approchées tel que l'algorithme de génération de bande. En fin nous présentant le problème de sac à dos rectangulaire ainsi qu'une méthode de résolution basé sur le principe des points de découpe.

Abstract

The theme in our work is the guillotine cutting two-dimensiona that is to perform vertical or horizontal sections on a rectangular support going from one edge to the opposite while being parallel to the two remaining edge. In this these we will see first some general notions and concepts for cutting issue and its modeling and the different types, we will see later in a typology (classification) cutting and packing problems. Thirdly a state of the art presents exact solution methods based on separation methods and assessment and dynamic programming and offers several resolution methods Also approached as the band generation algorithm. We end with the rectangular knapsack problem and a method of resolution based on the principle of cutting points.

Table des matières

Introduction Générale

Les problèmes de découpe et d'assemblage sont des problèmes combinatoires, ils sont classés dans la catégorie des problèmes NP-complets et admettent de nombreuses applications en industrie, en systèmes multiprogrammes et multiprocesseurs, ces problèmes se présentent lorsqu'on se propose d'optimiser l'utilisation d'un ou de plusieurs entités disponibles en y plaçant des sous entité en stock, ces problèmes sont connus sous le nom de problèmes de découpe à deux dimension (non) contraint.

Le thème abordé dans nos travaux est celui de la découpe guillotine à deux dimension qui consiste à effectuer des coupes vertical ou horizontales sur un support rectangulaire en allant d'un arête à son opposé tout en étant parallèle au deux arêtes restantes.

Ce mémoire comporte quatre chapitres, le premier chapitre est consacré à quelques notions générales et concepts de base pour le problème de découpe ainsi que sa modélisation et ces différents types.

Dans le deuxième chapitre nous verrons une typologie (classification) des problèmes de découpe et d'assemblage

Le troisième chapitre étant un état de l'art présente des méthodes de résolution exact s'appuyant sur les méthode de séparation et évaluation et la programmation dynamique et propose également plusieurs méthodes de résolution approchées tel que l'algorithmes de génération de bande.

En fin dans le dernier chapitre nous présentant le problème de sac à dos rectangulaire ainsi qu'une méthode de résolution basé sur le principe des points de découpe.

Chapitre 1

Notions générales sur les problèmes de découpe

1.1 Introduction

Le problème de découpe a été étudié pour la première fois par Kantorovick [10] dont le but était de le modéliser sous une forme cohérente. Gilmore et Gomory [7] ont repris ces travaux pour la résolution de quelques unes de ces variantes et les ont généralisés aux problèmes de découpe à deux dimensions.

Ces problèmes différent entre eux suivant le support, le matériel utilisé et le type de découpes effectuées. Parmi ces différents types on s'intéresse particulièrement à la découpe guillotine.

1.2 Notions générales sur les problèmes de découpe

Le problème de découpe est un problème que l'on rencontre courament dans certaines industries tels que: la boiserie, la métallurgie et le prêt-à-porter.

Un mauvais plan de découpe peut entraîner d'importantes pertes matérielles.

Le problème de découpe depend des éléments suivants :

1.2.1 Le support initial

Le support est l'un des éléments de base du problème de découpe. les supports diffèrent entre eux par leur :

Forme géométrique :

- Si la forme du support est circulaire, il est appelé cercle initial de dimension (Ra) où Ra est son rayon.

- Si le support est de forme rectangulaire, il est appelé dans ce cas rectangle initial, de dimensions (L, W) où L représente la longueur et W sa hauteur.

Dans les deux cas on parle de problème de découpe à deux dimensions.

La longueur (ou la hauteur) est négligée si elle est très grande par rapport aux dimensions des pièces à produire. Dans ce cas le problème peut être représenté par une bande, on parlera donc de problème de découpe à une dimension.

Un problème de découpe est à trois dimensions si son support est caractérisé par une base rectangulaire de dimension (L, W) et d'une épaisseur E (voir figure 1).

Homogénéité et régularité :

Un support est homogène si sa surface est partout la même et il est régulier s'il ne présente aucun défaut.

Dissymétrie :

Sur certain support comme les tissus à motifs, les pièces à découper ont souvent une orientation imposée, on parlera alors de pièce fixée.

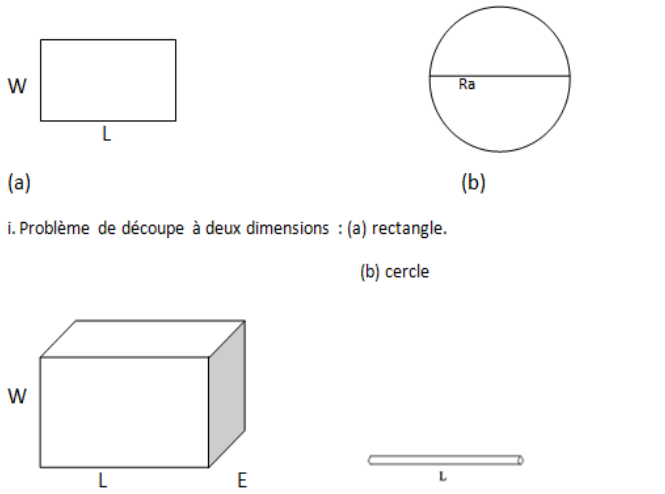


Figure 1: Les différents types de supports

1.2.2 Les pièces à découper

Les caractéristiques principales d'une pièce à découper sont :

- Sa forme géométrique.
- le respect des contraints liées au support et au coût de production.

Les pièces sont dites fixées si l'orientation est imposée si non les rotations sont permises et les pièces peuvent être pivotées de 90°.

1.2.3 Les différents types de découpe

En pratique, les utilisateurs sont souvent contraints par le matériel de découpe disponible, i.e. la façon de procéder pour la réalisation d'un plan de découpe est différente.

i. La découpe guillotine :

Si on suppose que le support est une plaque rectangulaire, alors la découpe est effectuée par la dissection en allant d'un coté à son apposé parallèlement aux deux autres.

ii. La découpe non guillotine :

En générale cette découpe engendre une solution meilleure que celle réalisée par les découpes du type guillotine, en effet cette découpe consiste à utiliser le même procédé que dans la découpe guillotine, et de plus, elle peut être effectuée tout en marquant des arrêts d'atteindre le coté opposé du (sous) rectangle à découper.

iii. La découpe non orthogonale :

cette découpe ne prend pas en considération l'orientation des pièces. Cette fois les pièces peuvent être pivotées et translatées (on effectue des rotations sur les pièces, donc elles ne sont pas fixées).

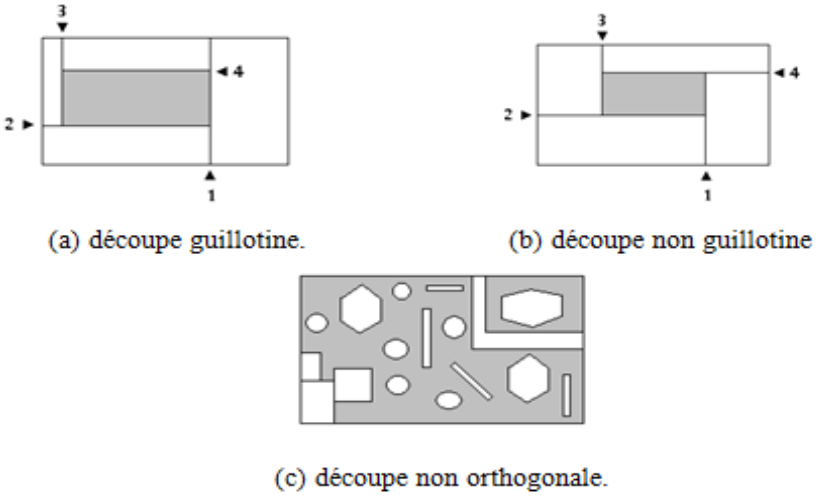


Figure 2: différents types de découpe

1.3 Problème de découpe guillotine à deux dimensions (DGDD) :

Parmi les différents types de découpe nous allons interesser au problème de découpe guillotine. Soit les hypothèses suivantes :

1. D'une part, on suppose que les découpes effectuées sur le rectangle initial, pour les différents problèmes précédents, sont du type guillotine et que les pièces de l'ensemble S sont fixées (i.e la pièce de longueur l et de hauteur w n'est pas la même que celle qui possède la longeur w et la hauteur l).

2. D'autre part, on suppose que les dimensions L, W, l_i et w_i, pour $i = 1, ..., n$, sont des entiers.

1.3.1 Formulation et modélisation du problème :

Une instance du problème de découpe guillotine à deux dimensions consiste à découper n petites pièces rectangulaires à partir d'un rectangle initial R de dimensions (L, W), où chaque pièce $i, i = 1, \ldots, n$, est caractérisée par sa longueur , sa hauteur et un profit (poids) c_i, $i = 1, \ldots, n$. L'objectif est de maximiser la somme des utilités des pièces produites.

Soit le vecteur $(x_1 ... x_n)$ constitué d'entiers naturels correspondant à un modèle de découpe guillotine s'il est possible de découper x_i pièce de type $i, i = 1, \ldots, n$, à partir du support initial de façon que les pièces ne se chevauchent pas.

Le problème de découpe guillotine à deux dimensions consiste à maximiser la valeur du modèle de découpe, i.e:

$$DGDD \begin{cases} \max \sum_{i=1}^{n} c_i x_i \\ \text{tel que } (x_1, ..., x_n) \text{ correspand à un modèle de } DG \end{cases}$$

Toutes les pièces produites qui diffèrent des pièces à découper sont considérées comme des chutes.

Définition 1.3.1 *Le problème DGDD est dit non pondéré si le profit de chaque pièce est égale à sa surface c-à-d : $c_i = l_i w_i$ pour $i = 1, \ldots, n$.*

L'objectif du problème de découpe guillotine $(DGDD)$ dans ce cas est de minimiser la chute .

Définition 1.3.2 *Le problème DGDD est dit pondéré s'il existe un type de pièce pour lequel le profit est différent de sa surface*

c-à-d :

$$\exists l, 1 \leq l \leq n \text{ tel que } c_i \neq l_i w_i \text{ pour } i = 1, \ldots, n.$$

L'objectif dans ce cas est de maximiser la somme des utilités sur l'entité en stock.

Définition 1.3.3 *Soit M l'ensemble fini des vecteurs représentant les différents modèles de découpe réalisable pour le problème DGDD, soit ε un élément de M tel que $\varepsilon = (\varepsilon_1, ..., \varepsilon_n)$, où chaque ε_i désigne le nombre d'apparition de la $i^{ème}$ pièces de S dans le modèle de découpe ε.*

(a) version non pondérée :

La fonction objectif est représentée par l'application F définie par :

$$\begin{aligned} F &: M \to \mathbb{N} \\ \text{Tel que } F^* &= F(\varepsilon) = LW - \sum_{i=1}^{n} c_i \varepsilon_i \end{aligned}$$

Où $c_i = l_i w_i$ désigne la surface de la pièce $i = 1, \ldots, n$.

La solution optimale du problème $DGDD$ dans ce cas consiste à trouver le couple (ε^*, F^*) tel que :

$$F(\varepsilon^*) = \min_{\varepsilon \in M} F(\varepsilon)$$

L'utilité du modèle de découpe réalisable ε^* est égale à F^*, ε^* est le modèle de découpe optimale qui minimise la chute sur le rectangle initial.

(b) version pondérée

La fonction objectif est représentée cette fois comme suit :

$$\begin{aligned} F &: M \to \mathbb{N} \\ \text{Tel que } F^* &= F(\varepsilon) = \sum_{i=1}^{n} c_i \varepsilon_i \end{aligned}$$

Où $c_i \neq l_i w_i$ désigne le profit associé à la pièce $i = 1, \ldots, n$.

La solution optimale revient à déterminer le couple (ε^*, F^*) tel que :

$$F(\varepsilon^*) = \max_{\varepsilon \in M} F(\varepsilon)$$

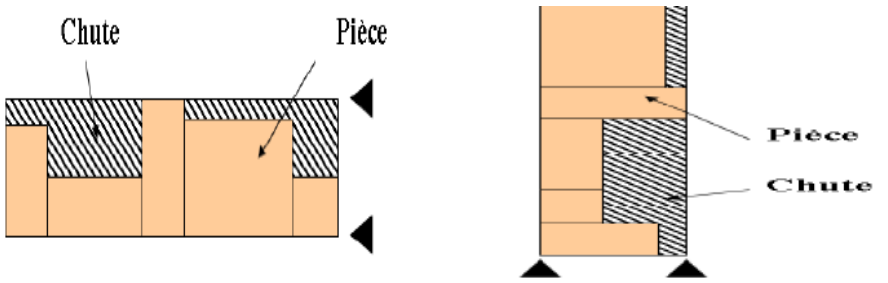
L'utilité du modèle de découpe réalisable ε^* est égale à F^* , ε^* est le modèle de découpe optimale qui maximise la somme des utilités sur l'entité en stock.

1.3.2 Les principaux modèles de bandes

1. **Une bande générale horizontale (resp. verticale):** est une succetion d'un ensemble de pièces rectangulaires de telle façon qu'il existe une ligne horizontale (resp.verticale) ayant les propriétés suivantes :

- Ele touche toutes les pièces sur leur longueur (resp. hauteur).

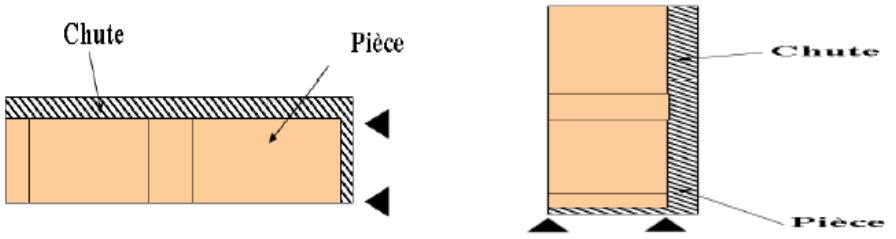
- Un des demi-plans défini par celle-ci contient toutes les pièces (voir figure 3).



(a). Bande générale horizontale. (b). Bande générale verticale.

Figure 3: Bandes générales horizontale et verticale.

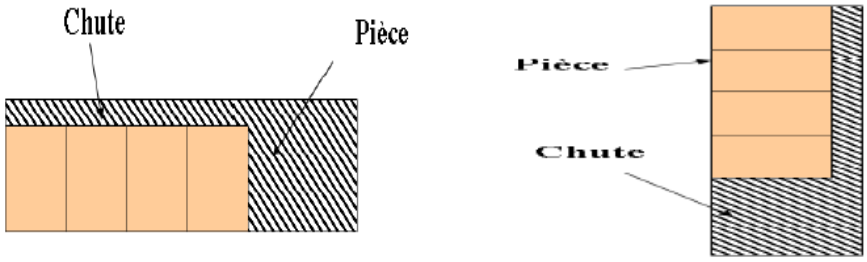
2. **Une bande uniforme:** est une bande pour laquelle existe exactement deux lignes horizontales (resp. verticales) ayant les propriétés précédentes. Dans ce cas, toutes les pièces découpées de la bande sont de même hauteur (resp. longueur) (voir figure 4).



(a). Bande uniforme horizontale. (b). Bande uniforme verticale.

Figure 4: Bandes uniformes horizontale et verticale

3. Une bande homogène horizontale (resp. verticale): est une bande uniforme où il n'y a qu'un seul type de pièces participant.



(a). Bande homogène horizontale. (b). Bande homogène verticale.

Figure 5: Bandes homogènes horizontale et verticale.

4. Une bande optimale de longueur α et de hauteur w (resp. de longueur l et de hauteur β): est une bande générale occupant la surface maximum de la bande (α, w) (resp. (l, β)).

1.3.3 Les principaux modèles de découpes

Un modèle de découpe réalisable s'obtient en combinant un certain nombre de bandes horizontales ou verticales. Comme pour les modèles de bandes il existe différents modèles de découpe:

1. **Un modèle de découpe uniforme:** est un modèle constitué par la combinaison de bandes uniformes verticales et horizontales.

2. **Un modèle de découpe générale:** est un modèle constitué par la combinaison de bandes générales verticales et horizontales.

3. **Un modèle de découpe homogène:** est une dissection uniforme pour laquelle la solution est caractérisée par la répétition d'un seul type de pièces de l'ensemble S (voir figure 6).

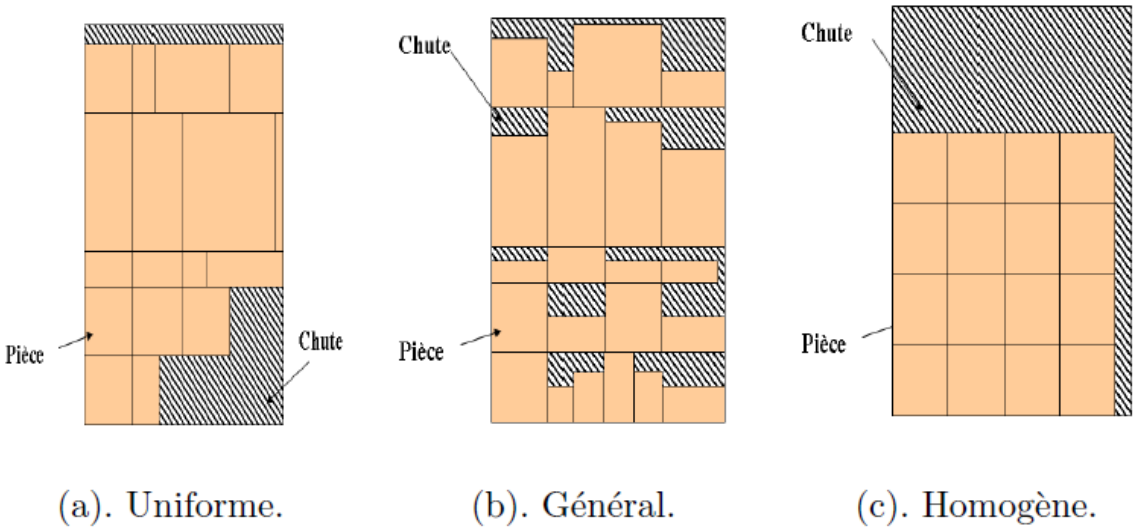


Figure 6: Les différents types de modèles de découpe.

1.3.4 Les points de découpe

Les points de découpes représentent les abscisses (respectivement les ordonnés) sur lesquelles on peut effectuer des coupes verticales (respectivement horizontales). Ce procédé a été introduit par Herz [3] et repris ensuite par un certain nombre d'auteurs.

Les découpes verticales sont limitées par l'ensemble P qui est constitué des combinaisons linaires entre les différentes longueurs de chaque type de pièces de S, idem pour les découpes

horizontales qui sont limitées par l'ensemble Q des combinaisons linéaires sur les différentes hauteurs des pièces de S. Ces ensembles sont définis par :

$$\begin{aligned} P &= \left\{ p \in \mathbb{N} / p = \sum_{i=1}^{n} l_i x_i \leq L, x_i \in \mathbb{N} \right\} \\ Q &= \left\{ q \in \mathbb{N} / q = \sum_{i=1}^{n} w_i x_i \leq W,\ x_i \in \mathbb{N} \right\} \end{aligned}$$

1.3.5 Les différentes variantes du problème de découpe guillotine à deux dimensions (DGDD)

Le problème de $DGDD$ peut engendrer plusieurs variantes dues aux différentes contraintes imposées et à leurs combinaisons.

Le problème de découpe guillotine à deux dimensions contraint (DGDD-C)

Il consiste à déterminer un modèle de découpe tel que le nombre d'apparition de chaque type de pièce $i, i = 1, \ldots, n$ ne doit pas excéder une valeur qu'on notera b_i. Ce problème peut être modélisé comme suit sous forme non linéaire :

$$\begin{cases} \max \sum_{i=1}^{m} \sum_{j=1}^{J} \sum_{k=1}^{K} c_{ijk} a_{ijk} & (1) \\ \sum_{j=1}^{J} l_j \varepsilon_j \leq L & (2) \\ \sum_{k=1}^{K} w_j \mu_j \leq W & (3) \\ \sum_{i=1}^{m} a_{ijk} \leq \varepsilon_j \mu_j \qquad \text{pour chaque } j, k & (4) \\ \sum_{j=1}^{J} \sum_{k=1}^{K} a_{ijk} b_i \qquad \text{pour chaque } i & (5) \\ \text{avec } \varepsilon_j, \mu_j, a_{ijk} \geq 0,\ \text{entier},\ i = 1...m, j = 1...J, k = 1...K & (6) \end{cases}$$

Tel que : J, K représente le nombre des différentes longueurs et hauteurs.

$$c_{ijk} = \begin{cases} c_i & \text{si } l_i \leq l_j \text{ et } w_i \leq w_k \\ 0 & \text{si non} \end{cases}$$

ε_j: Le nombre de fois où la longueur l_i a été découpée de L.

μ_k : Le nombre de fois où la hauteur w_k a été découpée de W.

a_{ijk} : Le nombre de rectangle de dimension $l_j \times w_k$ contenant la pièce de type i.

La fonction objectif (1) maximise la valeur totale des pièces à découper, les contraintes (2) et(3) garantissent le non dépassement des dimensions du support initial, la contrainte (4) limite la variable a_{ijk} à $\varepsilon_j.\mu_k$, la cinquième contrainte concerne l'accessibilité des pièces, et la sixiéme et dernière contrainte fait référence à l'intégrité des variables.

On trouve dans la littérature plusieurs travaux réalisés dans ce domaine entre autre Christodides et Whitlock [2] qui ont présentés la méthode « Depth first search » pour résoudre des instances de petite taille.

Le problème de découpe guillotine à deux dimensions non contraint (DGDD-NC)

Le problème est noté ainsi lorsque la valeur n'est pas imposée. Ce problème peut être modélisé comme pour le cas contraint en suppriment les contraintes (4) et (5) ce qui permet d'avoir le modèle quadratique en nombre entier suivant :

$$\begin{cases} \max \sum_{j=1}^{J} \sum_{k=1}^{K} c_{jk} \varepsilon_j \mu \\ \sum_{j=1}^{J} l_j \varepsilon \leq L \\ \sum_{k=1}^{K} w_k \mu_k \leq W \\ \text{avec } \varepsilon_j, \mu_k \geq 0, \text{ intéger } j = 1...J, k = 1...K. \end{cases}$$

Cette variante du problème de $DGDD$ a été résolue à l'optimum par J.E.Beasley [1] et Gilmores et Gomory [8].

Le problème de découpe guillotine à deux dimensions avec contrainte de niveau K (K-DGDD) Une autre variante du problème est de considérer une contrainte sur le nombre totale de coupe, i.e. la somme des coupes guillotine verticales et horizontales engendrant une pièce de type i, $i = 1, \ldots, n$ ne doit pas dépasser une certaine constante $K < \infty$. Dans ce genre de cas le problème est appelé $K - DGDD$ et K représente le nombre d'étape de découpe (ED). la variante $2 - DGDD$ avec la supposition que la première coupe de la première étape est parallèle à l'axe des longueurs et la première coupe de la deuxième étape est parallèle à l'axe des hauteurs peut être modélisée comme suit [14] :

$$
\begin{cases}
\max \sum_{i=1}^{m} \sum_{j=1}^{P} \sum_{k=1}^{Q} c_i x_{ijk} & (7) \\
\sum_{k=1}^{Q} W_k \leq W & (8) \\
\sum_{i=1}^{m} x_{ijk} \leq 1, \text{ pour tout } j, k & (9) \\
\sum_{i=1}^{m} \sum_{j=1}^{P} l_i x_{ijk} \leq L, \text{ pour tout } k & (10) \\
\sum_{i=1}^{m} w_i x_{ijk} \leq W_k, \text{ pour tout } j, k & (11) \\
\sum_{j=1}^{P} \sum_{k=1}^{Q} x_{ijk} \leq b_i, \text{ pour tout } i & (12) \\
W_k \geq W_{k+1}, \text{ pour tout } k & (13) \\
\text{avec } x_{ijk} \in \{0,1\}, W_k \geq 0, i = 1...m, j = 1...J, k = 1...K. & (14)
\end{cases}
$$

Tel que:

$l_{\min}$: correspond à la plus petite longueur des pièces.

$w_{\min}$: Correspond à la plus petite hauteur des pièces.

P : le nombre maximum de bandes verticales tel que $P = \lfloor L/l_{\min} \rfloor$.

Q : le nombre maximum de bandes horizontales tel que $Q = \lfloor W/w_{\min} \rfloor$.

l_j : Longueur de la $j^{ème}$bande verticale pour $j = 1, \ldots, P$.

w_k : hauteur de la $k^{ème}$bande horizontale pour $k = 1, \ldots, Q$.

$$x_{ijk} = k \begin{cases} 1_i \text{ si la pièce de pièce } i \text{ est placèe dans le rectangle } l_j \times w_k \\ 0 \qquad \text{si non} \end{cases}$$

La fonction objectif (7) maximise la valeur totale des pièces à découper. La contrainte (8) garantie le fait que les hauteurs des bandes ne dépassent pas la hauteur du support initial, la contrainte (9) impose le placement d'une seule pièce dans chaque rectangle $l_j \times w_k$, les contraintes (10) et (11) garantissent que les dimensions de chaque pièce placé dans le rectangle $l_j \times w_k$ ne dépassent pas les dimensions de ce dernier, la 12^e contrainte fait référence à l'accessibilité des pièces, la contrainte (13) élimine quelque symétrie et la 14^e et dernière contrainte précise que les valeurs l_j et w_k ne sont pas forcément des entiers. La figure 7 montre une solution $2 - DGDD$ qui est produite en appliquant les phases suivantes :

Phase1: le rectangle est découpé suivant sa longueur en un ensemble de bandes verticales.

Phase 2: chacune de ces bandes verticales est prise individuellement et elle est découpée suivant sa longueur.

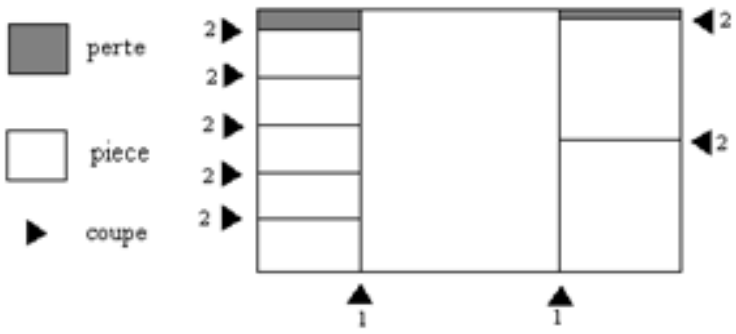


Figure 7: Représentation d'une solution quand k est fixée à 2.

Parmi les travaux réalisés dans ce sens on cite [12] qui ont traité le cas $K - DGDD - C$. Lodi et Monaci [11] ont proposé deux programmes linéaires en nombre entier pour la résolution du problème $2 - DGDD$.

Chapitre 2

Une meilleure typologie des problèmes de découpe

2.1 Introduction

Une typologie est une organisation systématique des objets en catégories homogènes sur la base de donnée, un ensemble de critères caractérisant. Il est orienté vers la pratique et destiné à faire face à "importante" objet réel en premier lieu tout hypothétique et/ou "moins important" objets réels pourraient être négligés [9].

2.2 Structure des problèmes de découpe

Les problèmes de découpes sont composés de deux ensemble:

- un ensemble de grands objets (entrée, approvisionnement)
- un ensemble de petits articles (production, la demande)

qui sont définis de manière exhaustive dans une, deux, trois ou un plus grand nombre (n) de dimensions géométriques.

Sélectionnez certains ou tous les petits articles, les regrouper en une ou plusieurs sous-ensembles et assigner chacun des sous-ensembles résultant à l'un des objets de grande taille tels que le géométrique condition est, c'est à dire les petits articles de chaque sous-ensemble doivent être posés sur la grande correspondante objet de telle sorte que:

• tous les petits articles du sous-ensemble se situent entièrement dans l'objet de grande taille.

• les petits articles ne se chevauchent pas, et on optimise une fonction objective (unidimensionnelle ou multidimensionnelle) donnée.

Nous notons que une solution de ce problème peut résulter en utilisant certaines ou tous les objets de grande taille, et certains ou tous les petits articles, respectivement.

Formellement, cinq sous-problèmes peuvent être distingués, qui bien entendu, doivent être résolus simultanément afin de parvenir à un "mondiale" optimale:

• problème de sélection en ce qui concerne les grands objets.

• problème de sélection en ce qui concerne les petits articles.

• problème de regroupement concernant la petite sélectionné éléments,

• problème d'allocation concernant la cession des sous-ensembles de petits articles aux grands objets,

• problème de mise en forme en ce qui concerne l'agencement des petits articles sur chacun des grands objets sélectionnés par rapport à la condition géométrique.

2.3 Critères pour la définition des types des problèmes (modifié)

1. **Dimensionnalité**

   Nous distinguons des problèmes à une, deux et trois dimensions. Dans la littérature, de temps en temps, aussi des problèmes avec plus de trois géométrique dimensions sont prises en compte.

   Les problèmes de ce type ($n > 3$) sont considérés comme variantes, ici.

2. **Type de mission**

   Encore une fois, comme dans Dyckhoff (1990), nous introduisons deux situations de base, dont nous préférons parler de la maximisation du production (valeur de sortie) et la minimisation entrée (valeur), respectivement.

   • La maximisation du production (valeur)

   Dans le cas de la maximisation du production (valeur), un ensemble de petits articles doit être affectée à un ensemble donné de grands objets.

   L'ensemble des objets de grande taille insuffisante pour contenir tous les petits articles.

   Tous les objets volumineux doivent être utilisées (en d'autres termes: il n'y a pas de sélection problème en ce qui concerne les objets de grande taille), à laquelle une sélection (un sous-ensemble) des petits articles de maximale. La valeur doit être affectée.

   •La minimisation des entrées (valeur)

Encore une fois, un ensemble donné de petits articles doit être affecté à un ensemble de grands objets. Contrairement avant, dans le cas d'entrée (valeur) minimisation de l'ensemble des objets de grande taille est suffisante pour recevoir tous les petits articles. Tous petits articles doivent être assignés à une sélection (un sous-ensemble) d'objet de grande taille (s) d'un minimum de "valeur". Ici, "la maximisation du production (valeur) " et "la minimisation entrée (valeur)" sont utilisés en général, non spécifique manière. Lorsque le traitement des problèmes spécifiques, la "valeur" des objets / éléments doit être défini plus précise et peut être représenté par les coûts, les recettes, ou les quantités de matériaux.

Souvent, la valeur des objets / éléments peuvent être considérés comme directement proportionnelle de leur taille de telle sorte que la fonction objective considère longueur (problèmes unidimensionnels), région (problèmes en deux dimensions) ou volume (en trois dimensions problèmes) maximisation (sortie) ou minimisation (entrée).

Dans de tels cas, il pourrait également possible de traduire à la fois "la maximisation du production (valeur) " et "la minimisation entrée (valeur)" dans "La réduction des déchets", c'est à dire la minimisation de la taille totale des parties inutilisées de la (sélectionné) grand objets. Dans l'environnement des problèmes de découpe souvent le terme "tri-minimisation des pertes" est utilisé.

Nous tenons également à souligner que pour définir des types de problèmes de base que ces deux situations sera considéré ici. Bien sûr, les problèmes rencontrées dans la pratique et / ou discutés dans le littérature peut être caractérisé par le fait qu'un problème de sélection existe à l'égard à la fois à grande objets et de petits articles. Cela nécessite une étendue fonction objectif

qui combine les recettes et les coûts ("la maximisation du profit"). Aussi situations exister dans lequel plus d'une fonction objectif peuvent être pris en compte.

1. Encore une fois, les problèmes de ce type seront considérés comme des variantes problème ici.

3. Assortiment de petits articles

En ce qui concerne la gamme des petits articles nous distinguons trois cas, les petites articles savoir identiques , un assortiment faiblement hétérogène de petite articles, et un assortiment fortement hétérogène de petits articles:

• Les petits articles identiques

Tous les articles ont la même forme et la même taille. Dans le cas de la maximisation du rendement, on peut supposer que ce type (unique) d'article a une demande illimitée.

•Assortiment faiblement hétérogène

Les petits articles peuvent être regroupés en relativement quelques classes (en relation avec le nombre total d'articles), pour laquelle les éléments identiques à rapport à la forme et la taille. Par définition, petite articles de forme et de taille identique qui nécessitent différentes orientations sont traités comme étant différents sortes d'objets. La demande de chaque type d'élément est relativement importante, et peut ou ne pas être limitée par une borne supérieure.

• Assortiment fortement hétérogène

L'ensemble de petits articles est caractérisé par le fait que seuls très peu d'éléments de forme identique et la taille. Si cela se produit, les éléments sont traités comme des éléments individuels. Par conséquent, la demande de chaque élément est égal à un.

Pour la définition des problèmes standard, nous supposons que l'ensemble des petits articles est uniformément structurée, c'est à dire qu'il ne contient pas d'éléments avec une grande exigences et d'autres avec de petites demandes.

4. Assortiment de grands objets

Pour l'assortiment des grands objets nous présentons les cas suivants :

•Un seul grand objet

L'ensemble des grands objets se compose d'un unique élément. Les dimensions de l'objet peuvent être fixé ou variable.

•Plusieurs grands objets

Dans le cas de plusieurs grands objets, seules les dimensions fixes seront considérées. Par analogie aux catégories présentées pour l'assortiment des petits articles, nous distinguons :

- De grands objets identiques faiblement hétérogènes

- Un assortiment fortement hétérogène de grands objets

2.4 Les types de problèmes de base, intermédiaires et raffinés

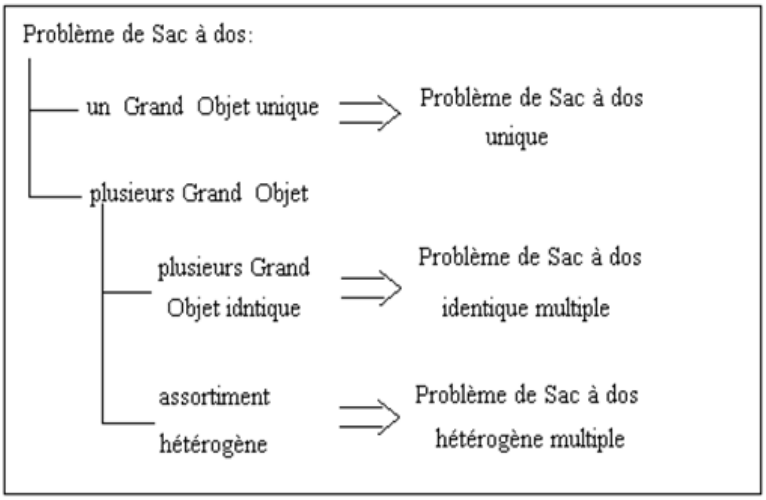
Types de problèmes de base

Types de problèmes de base de découpes sont développées par combinaison des deux critères "type de mission " et " assortiment de petits articles".

Types de la maximisation des sorties Problèmes de type de la maximisation de sortie ont en commun que les grands objets ne sont fournis en quantités limitées qui ne permettent pas pour le logement tous les petits articles. Comme la valeur des articles adaptés doit être maximisée, tous les objets de grande taille sera utilisé. En d'autres termes, il ya généralement un problème de sélection en ce qui concerne les petits articles, mais aucun concernant les objets de grande taille.

- **problème de sac à dos**

Il représente une catégorie de problème qui est caractérisée par un assortiment fortement hétérogène des petits articles qui doivent être assignés à un ensemble donné de grands objets. La disponibilité des grands objets est limitée tels que quelques petits articles seulement peuvent être adaptés et leur valeur doit être maximisée



Figue 8: Problème de sac à dos de types intermédiaires.

Types de la minimisation d'entrée Problèmes de type de la minimisation d'entrée sont caractérisés par le fait que la fourniture du grand objet est assez grande pour accueillir tous les

petits articles. Leurs demandes doivent être satisfaites entièrement, de sorte qu'aucun problème de sélection en ce qui concerne le petit article existe. La valeur du grand objet nécessaire pour accueillir tous les petits articles doit être minimisée.

- **Problème de découpe stock**

Les problèmes de cette catégorie exigent qu'un assortiment faiblement hétérogène de petits articles est complètement attribué à une sélection de grands objets de valeur minimale, le nombre ou la taille totale. L'extension des objets de grande taille est fixée dans toutes les dimensions.

Nous soulignons que nous ne faisons pas d'hypothèses sur la gamme des objets de grande taille. Il peut être constituée d'objets identiques, mais il pourrait aussi être un assortiment faiblement ou fortement hétérogène.

2.5 Conclusions

Les problèmes de découpe fournissent un champ d'application très vaste et très riche en nouvelles idées ce qui a poussé un grand nombre de chercheur à orienter leurs travaux vers ce domaine.

On remarque aussi qu'il existe une forte dualité entre les problèmes de découpe et d'assemblage. pour les problèmes de découpe, les grands objets sont pleins et doivent être vidés en découpant de petits articles. Par contre pour les problèmes d'assemblage, les grands objets sont vides et doivent être remplis par de petits articles.

Chapitre 3

Etat de l'art

3.1 Introduction

Durant ces dernières décennies plusieurs chercheurs se sont intéressés au problème de découpe. Certains ont opté pour des méthodes exactes préférant avoir une solution optimale quelque soit le temps d'exécution, alors que d'autre ont plutôt opté pour des méthodes approchées sacrifiant la solution optimale au profit du temps d'exécution.

Dans ce chapitre nous allons citer quelque méthode exacte ainsi que quelques heuristiques.

3.2 Les méthodes de résolution exactes

Nous présentons d'abord quelques méthodes exactes, ces méthodes donnent une garantie de trouver la solution optimale pour une instance de taille finie dans un temps limité et de prouver son optimalité (Puchinger et Raidl 2005).

3.2.1 La méthode séparation et évaluation (Branch and Bound)

L'algorithme de séparation et évaluation [13], plus connu sous son appellation anglaise Branch and Bound (B&B) (Land et Doig 1960), repose sur une méthode arborescente de recherche d'une solution optimale par séparations et évaluations, en représentant les états solutions par un arbre d'états, avec des noeuds, et des feuilles.

Le branch-and-bound est basé sur trois axes principaux :

– L'évaluation,

– La séparation,

– La stratégie de parcours.

L'évaluation

L'évaluation permet de réduire l'espace de recherche en éliminant quelques sous ensembles qui ne contiennent pas la solution optimale.

L'objectif est d'essayer d'évaluer l'intérêt de l'exploration d'un sous-ensemble de l'arborescence. Le branch-and-bound utilise une élimination de branches dans l'arborescence de recherche de la manière suivante : la recherche d'une solution de coût minimal, consiste à mémoriser la solution de plus bas coût rencontré pendant l'exploration, et à comparer le coût de chaque nœud parcouru à celui de la meilleure solution. Si le coût du nœud considéré est supérieur au meilleur coût, on arrête l'exploration de la branche et toutes les solutions de cette branche seront nécessairement de coût plus élevé que la meilleure solution déjà trouvée.

La séparation

La séparation consiste à diviser le problème en sous-problèmes.

Ainsi, en résolvant tous les sous-problèmes et en gardant la meilleure solution trouvée, on est assuré d'avoir résolu le problème initial. Cela revient à construire un arbre permettant d'énumérer toutes les solutions. L'ensemble de neouds de l'arbre qu'il reste encore à parcourir comme étant susceptibles de contenir une solution optimale, c'est-à-dire encore à diviser, est appelé ensemble des nœuds actifs.

La stratégie de parcours

- **La largeur d'abord**: Cette stratégie favorise les sommets les plus proches de la racine en faisant moins de séparations du problème initial. Elle est moins efficace que les deux autres stratégies présentées,

- **La profondeur d'abord**: Cette stratégie avantage les sommets les plus éloignés de la racine (de profondeur la plus élevée) en appliquant plus de séparations au problème initial. Cette voie mène rapidement à une solution optimale en économisant la mémoire,

- **Le meilleur d'abord**: Cette stratégie consiste à explorer des sous problèmes possédant la meilleure borne. Elle permet aussi d'éviter l'exploration de tous les sous-problèmes qui possèdent une mauvaise évaluation par rapport à la valeur optimale.

3.2.2 Programmation dynamique

L'idée principale de la programmation dynamique consiste à décomposer le problème initial en sous-problèmes de petites tailles. L'évaluation de ces derniers permet par la suite, d'éliminer les

moins intéressants de l'espace de recherche. Ainsi, les techniques de la programmation dynamique permettent de trouver une succession de coordonnées de décisions afin d'atteindre la solution optimale.

Gilmore et Gomory ont proposé la première fonction récursive basée sur la programmation dynamique pour le problème de découpe à deux niveaux. Cette formulation a été améliorée par Beasley pour répondre au problème de découpe guillotine non contraint [15].

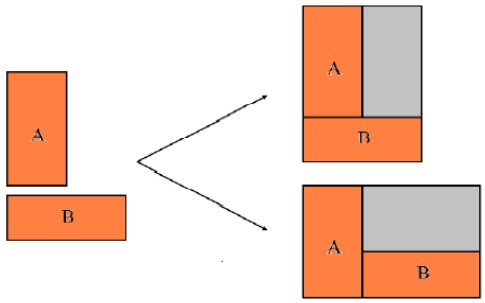


Figure 9: Constructions horizontales et verticales

3.3 La méthode de résolution Heuristiques (approchée)

3.3.1 Les algorithmes de génération de bandes

Les algorithmes de génération de bandes (SGA) procèdent en deux étapes, dans la première étape, ces algorithmes génèrent un ensemble de bandes. Et dans une deuxième étape ils cherchent une bonne combinaison de ces bandes.

Une bande (α, β) avec $0 < \alpha < L$ et $0 < \beta < W$, est obtenue en appliquant une découpe guillotine sur le rectangle initial (L, W). La première découpe peut être horizontale ou verticale. Si la première découpe est horizontale alors la bande générée est une bande horizontale (L, β). De

la même manière, si la première bande est verticale alors la bande générée est verticale (α, W). Dans ce qui suit et sans perte de généralité, on considère que la première découpe est horizontale [15].

Génération de bandes uniformes

La première étape des algorithmes SGA consiste à générer une bande uniforme $(L, \overline{w_j})$ pour chaque hauteur $\overline{w_j}$, $j \in J$, $J = 1, ..., r$. Les bandes uniformes générées sont construites en fonction de la solution optimale du problème de sac-à-dos borné suivant:

$$(KP_{L,\overline{w_j}}) = \begin{cases} f_{\overline{w_j}}(L) = \max \sum_{i \in S_{\overline{w_j}}} c_i x_{ij} \\ \text{S.c.} \quad \sum_{i \in S_{\overline{w_j}}} l_i x_{ij} \leq L \end{cases}$$

Où $S_{\overline{w_j}} = S_{\overline{w_j}^u}$, x_{ij} est le nombre d'apparitions de la pièce du type i dans la bande $(L, \overline{w_j})$ et $f_{\overline{w_j}}(L)$ le profit de la bande. $S_{\overline{w_j}^u} = i, i \in I, w_i = \overline{w_j}$, cela signifie que $S_{\overline{w_j}^u}$ est l'ensemble de types de pièces dont la hauteur est égale à $\overline{w_j}$.

La résolution du $\left(KP_{L,\overline{W}}\right)$ en utilisant la programmation dynamique donne la solution optimale de chaque $\left(KP_{L,\overline{w_j}}\right), j \in J$.

Dans la deuxième étape, les algorithmes SGA, déterminent le nombre d'occurrences y_i de chaque bande uniforme afin de construire le meilleur plan de découpe sans violer la borne supérieure b_i de chaque type de pièce $i \in I$. De cette manière, un plan de découpe optimal est construit en fonction de la solution du problème de sac-à-dos suivant :

$$(BKP_{L,W}) = \begin{cases} g_L(W) = \max \sum_{j \in J} f_{\overline{w_j}}(L) y_i \\ \text{s.c.} \ \sum_{j \in J} \overline{w_j} y_j \leq W \end{cases}$$

Le nombre d'apparitions y_i de la bande $(L, \overline{w_j})\,, j \in J$ dans le rectangle (L, W) est borné par a_j exprimé par :

$$a_j = \min\left\{\frac{W}{\overline{w_j}}, \min_{i \in S_{\overline{w_j}}}\left\{\frac{b_i}{x_{ij}},\ avec\ x_{ij} > 0\right\}\right\}$$

Génération de bandes générales

Pour générer des bandes générales, Hifi et M'hallah ont utilisé le même problème de sac-à-dos précédent. Cependant, les pièces appartenant à une bande, peuvent avoir des hauteurs différentes.

Dans ce cas, $S_{\overline{w_j}} = S_{\overline{w_j}^g}$, avec $S_{\overline{w_j}^g} = i \in I,\ w_i \leq \overline{w_j}$, cela signifié que $S_{\overline{w_j}^g}$ est l'ensemble pièces dont la hauteur est inférieure ou égale à $\overline{w_j}$.

Dans le même article, les auteurs proposent le programme linéaire en nombres entiers suivant pour combiner N bandes générales et r bandes uniformes afin de construire un plan de découpe général.

$$(IP_{L,W}) = \begin{cases} h_L(W) = \sum_{j \in J} \sum_{m=0}^{N_j+1} f_{\overline{w_j}^m}(L)\, y_j^m \\ \text{s.c. } \sum_{j \in J} \sum_{m=0}^{N_j+1} \overline{w_j} y_j^m \leq W \\ y_j^m \in N, j \in J, m = 0, ..., N_j + 1 \end{cases}$$

Où $x_{ij}^{N_j+1}, i \in I$ est la valeur de la solution optimal de $\left(BK_{L,\overline{w_j}}\right), j \in J$ et $f_{\overline{w_j}}^{N_j+1}(L)$ représente la valeur de la fonction objectif qui lui correspond.

y_j^m représent le nombre d'occurrences du $m^{ème}$ type de bandes $(L, \overline{w_j})$ dans le rectangle (L, W) et x_{ij}^m représente le nombre d'occurrences de la $i^{ème}$ pièce dans cette bande.

Le problème $(IP_{L,W})$ etant $NP-$difficile, les auteurs utilisent un algorithme glouton pour

trouver une approximation.

3.3.2 Un algorithme amélioré par génération de bandes

L'idée de base de l'algorithme amélioré de génération de bandes ($ESGA$) consiste a diviser le rectangle initial (L, W) en deux sous-réctangles (L, β) et $(L, W - \beta)$. Le premier sous-rectangle est résolu en utilisant l'algorithme SGA général et le deuxième sous-rectangle est résolu en utilisant une procédure alternative.

Par ailleurs, $ESGA$ détermine le point de division en utilisant une procédure de discrétisation sur les hauteurs des pièces et un calcul d'une borne supérieure des sous-rectangles et en comparant cette dernière a une borne inférieure de départ .

La procédure de discrétisation

La procédure de discrétisation limite la division sur un ensemble fini de sous-rectangles qui participent potentiellement à la meilleure solution.

Cette technique, utilisée aussi par Christofides et Whitlock pour le problème non contraint, est basée sur la construction de l'ensemble des combinaisons linéaires sur les hauteurs des pièces.

Pour un sous-rectangle (L, β), l'ensemble des combinaisons linéaires sur les hauteurs est :

$$Q(L, \beta) = \left\{ y \left| y = \sum_{i \in S_\beta} w_i z_i \leq \beta, z_i \leq \min \left\{ b_i, \left\lfloor \frac{L}{l_i} \right\rfloor \left\lfloor \frac{\beta}{w_i} \right\rfloor, z_i \in \mathbb{N} \right\} \right. \right\}$$

où

$$S_\beta = \{ i \in I \,|\, w_i \leq \beta \}$$

La borne supérieure

Dans l'algorithme $ESGA$, la borne supérieure permet de faire une estimation du profit généré par la découpe d'un sous-rectangle. Soit $\beta \in Q_{(L,W)}$ une découpe valide et (L, β) le sous-rectangle courant généré par cette découpe (L, β) avec les points (L_0, β_0) définis par :

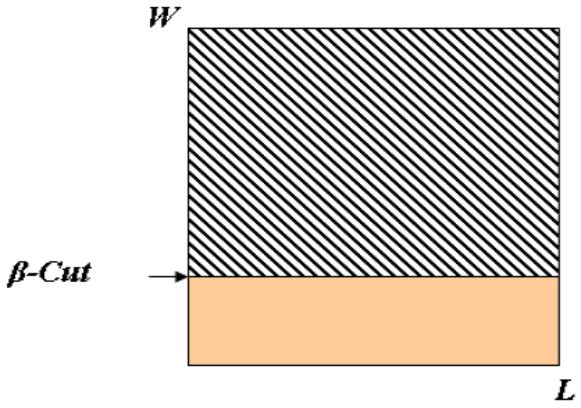


Figure 10: Division en sous-rectangles (β-cut).

$$L_0 = \max_{x \in P(L,\beta)} \{x\} \text{ et } \beta_0 = \max_{y \in P(L,\beta)} \{y\}$$

où

$$P(L,\beta) = \left\{ x \left| x = \sum_{i \in I} l_i z_i \leq L, w_i \leq \beta, z_i \leq \min \left\{ b_i, \left\lfloor \frac{L}{l_i} \right\rfloor \left\lfloor \frac{\beta}{w} \right\rfloor \right\}, z_i \in \mathbb{N} \right. \right\}$$

La borne supérieure $U_{(L,\beta)}$ du sous-rectangle (L, β) est obtenue suivant la solution optimale du problème de sac-à-dos KP_U, cela signifie que:

$$U_{(L,\beta)} = \max \left\{ \sum_{j \in J} f_{\overline{w_j}}(L) z_j \left| \sum_{j \in J} w_j z_j \leq \beta, z_j \in \mathbb{N}, j = 1 \in J \right. \right\}$$

où $f_{\overline{w_j}}(L)$ est le profit généré par la bande générale $(L, \overline{w_j})$ et z_j le nombre d'occurrences de cette bande dans le sous-rectangle (L, β).

Le choix du point de division

Dans l'algorithme $ESGA$, les auteurs utilisent la programmation dynamique pour résoudre KP_U, ils calculent toutes les bornes supérieures des sous-rectangles possibles (L, y) avec $0 \leq y \leq W$.

Par ailleurs, pour chaque $\beta-cut$ qui généré le sous-rectangle courant (L, β) et le sous-rectangle restant $(L, W-\beta)$, le processus d'évaluation suivant est lancé :

$$Lawer_{(L,\beta)} + U_{(L,W-\beta)} \leq Best_{(L,W)}$$

où $Best_{(L,W)}$ est la meilleure solution courante de (L, W) et $Lawer_{(L,\beta)}$ la meilleure solution réalisable courante obtenue sur le sous-rectangle (L, β) (voir figure 10).

Ce processus permet de décider, si cette découpe est susceptible d'améliorer la meilleure solution courante de (L, W). Cette découpe est négligée dans le cas contraire.

3.3.3 Un algorithme hybride de génération de bandes

L'algorithme $HESGA$ peut être considéré comme une amélioration de l'algorithme $ESGA$, les auteurs introduisent dans cette approche basée sur une recherche locale. Le rectangle initial (L, W) est considéré comme la racine et les pièces représentent les feuilles. A chaque étape du développement de l'arbre de recherche, l'algorithme exécute la procédure $ESGA$ sur les deux sous-rectangles (L, β) et $(L, W-\beta)$.

Afin de réduire le temps de résolution, les auteurs introduisent deux stratégies ($HC1$ et $HC2$):

Supposons que le rectangle (L, W) peut être divisé en deux sous-rectangles (L, β) et $(L, W-\beta)$

- **HC1**: La première stratégie réduit l'ensemble $Q(L, W)$ en utilisant une approche qui exploite la contrainte de la borne supérieure (b_i) de chaque pièce.

- **HC2**: La deuxième stratégie concerne la borne supérieure $U_{(L,W-\beta)}$ de chaque sous-rectangle, cette stratégie introduit un paramètre ρ dans le processus d'évaluation du profit généré par un sous-rectangle comme suit :

$$Lawer_{(L,\beta)} + \rho \left[U_{(L,W-\beta)}\right] \leq Best_{(L,W)}$$

où $Best_{(L,W)}$ est la meilleure solution courante de (L, W), $Lawer_{(L,\beta)}$ la meilleure solution réalisable courante obtenue sur le sous-rectangle (L, β) et ρ une constante.

Chapitre 4

Problème de sac-à-dos rectangulaire

4.1 Introduction

Nous allons présente un algorithmes basés sur la programmation dynamique pour le probleme de sac à dos rectangulaire (RK). Cette algorithme résout la formule de récurrence proposé par Beasley, cette algorithme a été testé sur des instance de la literature [4].

4.2 Le problème de sac-à-dos rectangulaire

Le problème de sac-à-dos rectangulaire (RK) a été largement étudié de puis les années soixante. Gilmore et Gomory [5, 6] ont étudié ce problème .

En 1972, Herz [3] a présenté un algorithme récursif pour obtenir un modèle canonique, en utilisant ce qu'on appelle des points de discrétisation . Christofides et Whitlock [2] ont proposé une approche utilisant la programmation dynamique pour calculer l'ensemble des points de

discrétisation . Certains journaux considèrent également les procédures de recherche de l'arbre exactes pour ce problème.

Beasley [1] a proposé une approche de programmation dynamique utilisant les points de discrétisation de Herz .

Avant de presenter l'algorithmes d'implémentation pour la résolution du problème RK, je suis présenté d'abord quelques concepts et résultats.j'ai implementé des formules de récurrence proposées par Beasley combinés avec le concept de points de discrétisation de Herz [3].

Soit $I = (L, W, l, w, v)$ une instance du problème de RK. Nous considérons que L, W, l et w sont tous des entiers. Si ce n'est pas le cas, une instance intégrante équivalente peut être obtenue par une mise à l'échelle appropriée. Un point de discrétisation de la langueur (respectivement, de la hauteur) est une valeur $i \leq L$ (respectivement, $j \leq W$) qui peut être obtenu par une combinaison conique d'entier de $l_1...l_m$ (respectivement, $w_1...w_m$). Nous désignons par P (respectivement Q) l'ensemble de tous les points de discrétisation de la langueur (respectivement, de la hauteur). D'près Herz, un modèle canonique est un modèle obtenue par des coupes qui sont faites à partir des points de discrétisation. Nous notons qu'il suffit de ne considérer que les modèles canoniques (pour chaque modèle qui est pas canonique, il ya un un équivalent qui est canonique).

Pour les consulter, les fonctions suivantes seront utiles.

Pour le rationnel $x \leq L$, soit $p(x) = max(i|i \in P, i \leq x)$ et pour le rationnel $y \leq W$, soit $q(y) = max(j|j \in Q, j \leq y)$. En utilisant ces fonctions, il est facile de vérifier que la formule de récurrence ci-après, proposé par Beasley [1], peut être utilisée pour calculer la valeur $V(l, w)$ d'un modèle canonique guillotine optimale d'un rectangle de dimensions (l, w). Dans cette formule, $v(l, w)$ représente les pieces les plus grande valeur qui peut être coupé dans un rectangle de

dimensions (l, w), il est 0 si aucune pièce ne peut être découpé de rectangle, Ainsi, $V(L, W)$ est la valeur d'une solution pour une instance $I = (L, W, l, w, v)$.

$$V(l,w) = \max \left\{ \begin{array}{c} v(l,w) \\ \max\{V(l',w) + V(p(l-l'),w) \,|l' \in P \text{ et } 0 < l' \leq l/2\} \\ \max\{V(l,w') + V(l,q(w-w')) \,|w' \in Q \text{ et } 0 < w' \leq w/2\} \end{array} \right\} \tag{4.1}$$

4.2.1 Points de discrétisation

Nous présentons deux algorithmes pour trouver les points de discrétisation: les algorithmes DEE (discrétisation par énumération explicite) et DDP (discrétisation en utilisant la programmation dynamique). Dans l'algorithme DEE, D représente la longueur (ou hauteur) du support initial et $d_1, ..., d_m$ représentent les longueurs (ou les hauteurs) des pieces. L'algorithme DEE peut être mis en œuvre pour exécuter en temps $O(m\delta)$ où δ représente le nombre de combinaisons de nombre entier $d_1, .., d_m$ de valeur maximum D. Il n'est pas difficile de construire des instances pour les quelles une énumération explicite peut prendre un temps exponentiel. Mais si nous pouvons garantir que $d_i > \frac{D}{k} (i = 1, ..., M)$, alors la somme de m coefficients de n'importe quelle combinaison d'entier de $d_1, ..., d_m$ de valeur maximum D est inférieure ou égale à k. Ainsi, pour un k fixé, l'algorithme DEE est polynomiale en m.

Algorithme DEE

Entrée: D (longueur ou hauteur), $d_1, ..., d_m$.

Sortie: un ensemble ρ de points (de la longueur ou de la hauteur) de discrétisation.

$\rho = \emptyset, k = 0$

Tant que $k \geq 0$ faire

Pour $i = k+1$à m faire $z_i = \left\lfloor \left(D - \sum_{j=1}^{i-1} d_j z_j \right) / d_i \right\rfloor$.

$\rho = \rho \cup \left\{ \sum_{j=1}^{m} z_j d_j \right\}$.

$k = \max\left(\{ i | z_i > 0, 1 \leq i \leq m \} \cup \{-1\} \right)$.

Si $k > 0$ alors $z_k = z_k - 1$ et $\rho = \rho \cup \left\{ \sum_{j=1}^{k} z_j d_j \right\}$.

Retour ρ.

L'idée de base de cet algorithme est de résoudre un problème de sac à dos dans lequel chaque élément i a un poids et une valeur $d_i (i = 1, ..., M)$, et le sac à dos a une capacité D. La technique de programmation dynamique pour le problème du sac à dos donne une solution optimales de sac à dos avec une capacités (entier) prenant des valeurs de 1 à D. Il est facile de voir que si j est un point de discrétisation si et seulement si le sac à dos avec de capacité j admet une solution optimale j.

Algorithme DDP

Entrer: $D, d_1, ..., d_m$.

Sortier: un ensemble ρ de points de discrétisation

$\rho = \{0\}$.

Pour $j = 0$ à D faire $c_j = 0$.

Pour $i = 1$ à m faire

Pour $j = d_i$ à D

Si $c_j < c_{j-d_i} + d_i$ alors $c_j = c_{j-d_i} + d_i$.

Pour $j = 1$ à D

Si $c_j = j$ alors $\rho = \rho \cup \{j\}$.

Rotour ρ.

Nous notons que l'algorithme DDP exige un temps $O(m\ D)$. Ainsi, la mise à l'échelle (si nécessaire) pour obtenir une instance intégrante peut rendre l'utilisation de DDP inadapté dans la pratique. D'autre part, l'algorithme est adapté pour DDP dans lesquels D est faible. Si D est grande, mais les dimensions des éléments ne sont pas petite par rapport aux dimensions du rectangle initial, l'algorithme DEE a une performance satisfaisante.

4.3 Un algorithme de programmation dynamique pour le problème de RK

Nous allons maintenant décrire l'algorithme DP qui résout la formule de récurrence (4.1). Bien qu'il semble y avoir un moyen simple pour résoudre la formule de récurrence, nous croyons que l'implementation que nous décrivons dans ce qui suit peut être très efficace dans la pratique [4].

Soit $l_{\min}$ (respectivement, $w_{\min}$) la longueur minimale (respectivement hauteur) des éléments de l'instance.

Soit P_0 l'ensemble des valeurs $i \in P$ tel que $i \leq L - l_{\min}$ et soit Q_0 l'ensemble des valeurs $j \in Q$ tel que $j \leq W - w_{\min}$.

Soit $P_1 = P_0 \cup \{L\}$ et $Q_1 = Q_0 \cup \{W\}$. Nous pouvons utiliser les ensembles P_1 et Q_1 à la place des ensembles P et Q dans la récurrence ci-dessus et, éventuellement, obtenir une amélioration dans le temps de resolution, car aucun piece ne peut être à droite (respectivement, en haut) d'une découpe verticale (respectivement, horizontale) faite dans une position supérieure

à $L - l_{\min}$ (respectivement, $W - w_{\min}$).

L'algorithme a été construit de tel façon que le modèle optimale peut être facilement obtenu. Pour cela, l'algorithme enregistre dans une matrice, pour chaque rectangle de longueur $p_i \in P_1$ et la hauteur $q_j \in Q_1$, qui represente de la première coupe guillotine (horizontale ou verticale) qui doit être faite dans ce rectangle. Dans le cas où aucune coupe ne peut étre faite dans le rectangle, l'algorithme enregistre le pièce qui correspond à ce rectangle.

L'algorithme permet de résoudre de manière constructive à la formule de récurrence (4.1). La première étape de l'algorithme consiste à stocker pour chaque rectangle de largeur $p_i \in P_1$ et de hauteur $q_j \in Q_1$, la pièce de plus grande valeur qui peut être découpé (lignes $1-5$). Aux (lignes $6-17$), l'algorithme considère un rectangle de dimensions (p_i, q_j) et trouve une solution optimale pour ce rectangle de la manière suivante: pour chaque point de possible p_x où une coupe verticale peut être faite, l'algorithme test si la meilleure solution connue est pire que celle pour laquelle une coupe verticale se fait dans le sens vertical au point p_x (lignes $9-12$). Aux (lignes $14-17$) l'algorithme teste une coupe horizontale. L'algorithme commence à trouver une solution avec le plus petit rectangle possible et augmente ensuite de manière itérative la taille du rectangle (lignes $6-7$), en utilisant les meilleures solutions connues de petits rectangles de déterminer la meilleure solution pour le rectangle considéré dans l'itération.

Algorithme DP

Entrée: une instance $I = (L, W, l, w, v)$ du problème de RK .

Sortie : Une solution optimale pour I.

Soit $p_1 \leq ... \leq p_r$ être les points de l'ensemble P_1 .

Soit $q_1 \leq ... \leq q_s$ être les points de l'ensemble Q_1 .

1 pour $i = 1$ à r

2 Pour $j = 1$ à s

3 $V(i,j) = max(\{v_k | 1 \leq k \leq m, l_k \leq p_i \text{ et } w_k \leq q_j\} \cup \{0\})$.

4 élément $(i,j) = max(\{k | 1 \leq k \leq m, l_k \leq p_i, w_k \leq q_j \text{ et } v_k = V(i,j)\} \cup \{0\})$.

5 guillotine $(i,j) = nil$.

6 Pour $i = 2$ à r

7 Pour $j = 2$ à s

8 $n = max(k | 1 \leq k \leq i \text{ et } p_k < \lfloor \frac{P_i}{2} \rfloor)$.

9 Pour $x = 1$ à n

10 $t = max(k | 1 \leq k \leq r \text{ et } p_k \leq p_i - p_x)$.

11 Si $V(i,j) \leq V(x,j) + V(t,j)$ puis

12 $V(i,j) = V(x,j) + V(t,j)$, la position $(i,j) = p_x$ et la guillotine $(i,j) =' V'$.

13 $n = max(k | 1 \leq k \leq j \text{ et } q_k \leq \lfloor \frac{q_j}{2} \rfloor)$.

14 Pour $y = 1$à n

15 $t = max(k | 1 \leq k \leq s \text{ et } q_k \leq q_j - q_y)$.

16 Si $V(i,j) < V(i,y) + V(i,t)$, puis

17 $V(i,j) = V(i,y) + V(i,t)$, la position $(i,j) = q_y$ et guillotine $(i,j) =' W'$.

Lorsque l'algorithme DP s'arrête, pour chaque rectangle de dimensions (p_i, q_j), nous avons que $V(i,j)$ qui contient la valeur optimale que l'on peut obtenir pour ce rectangle, guillotine (i,j) indique la direction de la première coupe à guillotine, et la position (i,j) est la position (dans l'axe des abscisses ou à l'axe des y) lorsque la première coupe à guillotine est effectuée.

Si guillotine $(i, j) = nil$, alors aucune coupe ne doit être faite dans ce rectangle. Dans ce cas, le point (i, j) (si non-zéro) indique l'élément correspond à ce rectangle. La valeur de la solution optimale sera dans $V(r, s)$.

Nous pouvons utiliser un vecteur X (resp. Y), de la taille L (resp. W), et soit X_i (resp. Y_j) contenir $p(i)$ (resp. $q(j)$). Une fois que les points de discrétisation sont calculées, il faut du temps $O(L + W)$ pour déterminer les valeurs dans les vecteurs X et Y.

4.4 Les résultats des calculs pour le problème de RK

La performances de l'algorithme DP a été testé sur les instances de RK disponibles dans le OR-Library. 13 instance de RK ont été éxaminé, appelés $gcut1, ..., gcut13$ disponible dans cette bibliothèque. Pour tous ces cas, à l'exception de l'instance $gcut13$, des solutions optimales avaient déjà été trouvé [1]. En utilisant l'algorithme DP nous avons trouvé une solution optimale pour l'instance $gcut13$ en 22 secondes. Pour tous ces instance, la valeur de chaque pièce est égale à sa superficie. Nous notons que Caprara et Monaci et Fekete et Schepers ne pouvaient pas trouver une solution optimale pour cette instance en 1800 seconde (un Pentium III 800 MHz et $2,8GHz$ Pentium IV avec $1GB$ de mémoire, respectivement).

Nous rappelons que leurs approches sont pour le cadre plus général dans lequel les coupes ne doivent pas être guillotine (Dans ce cas général, l'approche DP peut être utilisée pour obtenir une borne inférieure).

Étant donné que l'algorithme a résolu tous ces cas en quelques secondes, nous avons construit quatre autres cas ($gcut14 - gcut17$) combinant les instances disponibles dans le OR-BIBLIOTHÈQUE. Ces nouvelles instances ont été obtenus par la mise ensemble les éléments de

l'instance *gcut*13 avec les éléments de chacune des instances *gcut* 9, *gcut* 10, et *gcut*11, *gcut*12. Pour ces nouveaux cas nous avons considéré que $R = (3500, 3500)$ [4] .

Tableau

Performance de l'algorithme DP

Instance	Quantity of items	Dimensions of the bin	r	s	Optimal Solution	Waste (%)	Time (sec)
gcut1	10	(250, 250)	28	9	56,460	9.664	0
gcut2	20	(250, 250)	39	52	60,536	3.142	0
gcut3	30	(250, 250)	81	42	61,036	2.342	0
gcut4	50	(250, 250)	85	84	61,698	1.283	0
gcut5	10	(500, 500)	19	27	246,000	1.600	0
gcut6	20	(500, 500)	34	42	238,998	4.401	0
gcut7	30	(500, 500)	66	33	242,567	2.973	0
gcut8	50	(500, 500)	97	136	246,633	1.347	0
gcut9	10	(1000, 1000)	31	11	971,100	2.890	0
gcut10	20	(1000, 1000)	29	55	982,025	1.798	0
gcut11	30	(1000, 1000)	69	109	980,096	1.990	0
gcut12	50	(1000, 1000)	155	124	979,986	2.001	0
gcut13	32	(3000, 3000)	1457	2310	8,997,780	0.025	22.03
gcut14	42	(3500, 3500)	2390	2861	12,245,410	0.037	118.45
gcut15	52	(3500, 3500)	2422	2933	12,246,032	0.032	120.86
gcut16	62	(3500, 3500)	2559	2943	12,248,836	0.010	161.47
gcut17	82	(3500, 3500)	2676	2953	12,248,892	0.009	212.66

Dans ce tableau, nous montrons les instances résolues et les résultats de calcul.

La colonne "Waste" montre - pour chaque solution trouvée- le pourcentage de la surface de qui ne correspond à aucune pièce.

La colonne "Time" indique le temps nécessaire pour résoudre l'instance, l'entrée 0 indique que le temps nécessaire est inférieur à $0.000,001$ secondes. Notez que les nouvelles instances *gcut*14, . . ., *gcut*17 avéré beaucoup plus difficile à résoudre: quelques minutes ont été nécessaires pour trouver une solution optimale.

Gcut 32

Il format de ces fichiers de données est: nombre de pièce (m), longueur, hauteur pour le rectangle de stock pour chaque pièce de la solution optimale pour les fichiers de données *gcut*1 . . . *gcut*12

est donnée dans le document ci-dessus.Plus d'informations sur la solution pour these13 problèmes d'essai peut être trouvée ici. 32

3000	3000	
365	185	67525
378	200	75600
410	165	67650
425	148	62900
425	296	125800
439	116	50924
464	1006	466784
520	205	106600
520	350	182000
540	530	286000
549	1413	775737
549	1882	1033218
553	496	274288
555	755	419025
555	496	275280
555	659	365745
567	473	268191
572	592	338624
572	975	557700
572	1175	672100
572	1575	900900
572	1390	795080
572	1490	852280
572	1590	909480

949 445 422305

949 478 453622

970 436 449110

4.5 Conclusion

Dans ce chapitre, nous avons présenté des algorithmes pour le problème RK, nous avons présenté l'algorithme pseudo-polynomiale DP, Cette algorithme peuvent soit utiliser l'algorithme DEE ou DDP pour générer les points de discrétisation. Nous avons également montré que ces algorithmes peuvent être mis en œuvre afin de fonctionner en temps polynomial.

Conclusion Générale

Parmi la grande famille des problèmes de découpe on a choisi de traiter l'une de ses variantes qui est le problème de découpe guillotine à deux dimensions non contraint (DGDD-NC). Il existe plusieurs méthodes de résolution pour ce genre de problème. Nous avons étudié quelques une d'entre elles.

Nous avons par la suite étudié problème RK nous avons présenté l'algorithme pseudo-polynomiale DP.

Cet algorithme peut soit utiliser l'algorithme DEE ou DDP pour générer les points de discrétisation. Nous avons également montré que ces algorithmes peuvent être exécutés en temps polynomial lorsque les éléments ne sont pas très petite par rapport à la taille du rectangle initiale.

REFERENCES

1 **Beasley, J. E**. (1985): Algorithms for unconstrained two-dimensional guillotine cutting. Journal of the Operational Research Society 36, 297-306.

2 **Christofides, N.; Whitlock, C**. (1977): An algorithm for two-dimensional cutting problems. Operations Research 25, 30-44.

3 **Herz, J. C**. (1972): Recursive computational Procedure for two-dimensional stock cutting. IBM Journal of Research Development 16, 462-469.

4 **G.F. Cintra a, F.K. Miyazawab , Y. Wakabayashic,*, E.C. Xavierd** : Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation, European Journal of Operational Research 191 (2008) 61–85.

5 **Gilmore, P. C.; Gomory, R. E.** (1961): A linear programming approach to the cutting-stock problem part I. Operations Research 9, 849-859.

6 **Gilmore, P. C.; Gomory, R. E**. (1963): A linear programming approach to the cutting-stock problem part II. Operations Research 11, 864-888.

7 **Gilmore, P. C.; Gomory, R. E**. (1965): Multistage cutting stock problems of two and more dimensions. Operations Research13, 94-120.

8 **Gilmore, P.C, and Gomory, R. E** (1966): The theory and computational of knapsack functions, Operations Research 14 , 1045- 1074.

9 **G. Wäscher, H. Haussner, et H. Schumann**: An improved typology of cutting and packing problems. European Journal of Operational Research, to appear,2006.

10 **Kantorovich. L.V**.(1960):"Mathematical methods of organizing and planning production," Management Science, vol. 6, pp. 363–422.

11 **Lodi, A. and Monaci, M.** (2003) : "Integer linear programming models for 2-staged two-dimensional knapsack problems", Mathematical Programming, Series B, vol. 94, pp. 257-278

12 **Morabito. R. and Arenales**. (1996): Staged and constrained two-dimensional guillotine cutting problems: An and-or graph approach," European Journal of Operational Research, vol. 94, no. 3, pp. 548–560.

13 **Sidi Mohamed Douiri, Souad Elbernoussi, Halima Lakhbab**: Cours des Méthodes de Résolution Exactes Heuristiques et Méta heuristiques .mémoire on ligne .

14 **Scheithauer, G., 2002**: On a two-dimensional guillotine cutting problem. Presented at IFORS 2002, Edinburgh, UK.

15 **Toufik Saadi**,Résolution séquentielles et parallèles des problèmes de découpe/placement, Modeling and Simulation. Université Panthéon-Sorbonne-Paris I, 2008. French.<tel-00354737>.