N° D'ORDRE : 53/2024-D/MT

République Algérienne Démocratique et populaire
Ministère D'enseignements Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie Houari Boumediene
Faculté de mathématiques

THÈSE DE DOCTORAT EN SCIENCES

Présentée pour l'obtention du grade de docteur

En : MATHEMATIQUES

Spécialité : Recherche opérationnelle

Par : SLIMI BOUALEM

Sujet

Contribution à la résolution du Problème de Découpe Bi-objectif

Soutenue publiquement, le 11/11/2023, Devant le jury composé de :

M.	Mustapha Moulai professeur	à USTHB	Présidant.
M.	Moncef Abbas professeur	à USTHB	Directeur de thèse.
M.	Mouaouia Cherif Bouzid Maitre de Conférence /A	à ENSTA, Alger	Examinateur
Mme.	Isma Dahmani Maitre de Conférence /A	à USTHB	Examinatrice
M.	Nourddine Ikhlef Eschouf professeur	à UYF, Médéa	Examinateur
M.	Mohamed Bendraouche professeur	à USD, Blida 1	Examinateur.
M.	Yazid Benkanoun Maitre de Conférence / B	à USTHB	Invité.

Date de soutenance : 11 / 11 / 2023

REMERCIEMENTS

Je remercie Dieu Le Tout Puissant généreux de m'avoir aidé et guidé tout au long de ce parcours.

Je tiens à exprimer en tout premier lieu ma profonde gratitude à mon directeur de thèse de Doctorat, le Professeur ABBAS Moncef, de l'Université des Sciences et Technologie Houari Boumedienne, qui a bien voulu me proposer un sujet et d'avoir suivi le travail jusqu'à son terme. Ses remarques, ses conseils fructueux, m'ont beaucoup aidé à réaliser ce travail.

J'adresse également mes sincères remerciements au Professeur MOULAÏ Mustapha, d'avoir accepté de présider ma soutenance, malgré ses nombreuses occupations.

J'exprime mes sincères reconnaissances aux membres de jury, Monsieur BENDRAOUCHE Mohamed, Professeur à l'Université Saad Dahlab, Blida1, Monsieur IKHLEF ESCHOUF Noureddine, Professeur à l'Université Yahia Fares, Médéa, Monsieur BOUZID Mouaouia Cherif, Maître de Conférences Classe «A», à l'École nationale supérieure de technologie, Alger, Madame DAHMANI Isma, Maître de Conférences Classe «A», à l'Université des Sciences et Technologie Houari Boumedienne, Monsieur BENKANOUN Yazid, Maître de Conférences Classe «B», à l'Université des Sciences et Technologie Houari Boumedienne, pour le temps et l'attention qu'ils ont consacré à la lecture et à la critique de ce travail.

Qu'il me soit aussi permis de remercier tous ceux qui, par leur présence et leur amitié, ont contribué à ce travail.

RÉSUMÉ

En réponse au développement technologique moderne, dans la réalisation du processus de découpe, plusieurs extensions sont en effet apparues dans les problèmes de découpe, notamment le coût d'installation, qui survient dans les opérations de découpe industrielle, c'est ce qui a poussé de nombreux auteurs à intensifier les recherches dans ce sens, en utilisant des nouvelles méthodes d'optimisations comme les méthodes d'optimisations multi-objectifs, car de nombreuses études ont prouvé que ce type de problème ne se limitent pas à un seul objectif, mais le dépassent dans certains cas à un certain nombre d'objectifs. Par exemple, la minimisation simultanée de la perte totale de la matière première et le nombre d'objets à découper est un problème de découpe à deux objectifs. Tandis que la minimisation de coût total du panneau, la maximisation de profit des restes et le profit des restes provenant des opérations de découpe passées est décrit comme un problème de découpe à trois objectifs.

Parmi les problèmes de découpe à deux objectifs qui sont apparus dans la littérature, le problème de découpe avec coût d'installations, où l'optimisation exprime deux objectifs contradictoires, à savoir, la perte totale de matière première et le nombre d'installation, ce qui rend l'utilisation de la technique multi-objectif pour résoudre le problème plus précise et plus appropriée à la réalité. Selon ce point de vue, nous proposons dans cette thèse deux approches de résolution d'un problème de découpe unidimensionnelle et bidimensionnelle bi-objectif, où l'on cherche à minimiser à la fois la perte totale de la matière première et le nombre d'installations à effectuer. Ces techniques sont constituées de deux étapes dont la première consiste à générer tous les patrons de découpe réalisables et la seconde permet de construire des plans de découpes, satisfaisant les demandes, grâce à un sous ensemble de ces patrons. Ces différents plans de découpe représentent l'ensemble des solutions réalisables dont chacun est caractérisé par un nombre d'installations et la quantité totale de chutes.

Mots clés : perte, installation, patron de découpe réalisables, plan de découpe, problème de découpe bi-objectif.

ABSTRACT

In response to modern technological demands in the realization of the cutting process, several extensions have indeed appeared in the cutting stock problems, including the setup cost that arises in industrial cutting operations. This is what has prompted many authors to intensify research in this direction using new methods of resolution such as multi-objective optimization, because many studies have proven that this type of problem is not limited to a single objective, but in some cases exceeds it to a number of objectives, such as the simultaneous the minimization of the total trim loss of the raw material and the number of objects to be cut is cutting stock problems with two objectives, while the minimization of total panel cost, the maximization of remnant profit and the remnant profit from past cutting operations is described as a three-objective cutting stock problem.

Among the cutting stock problems with two objectives which are appear in the literature, the cutting stock problems with cost of setups, where the optimization expresses two contradictory objectives, namely, the total trim loss of raw material and the number of setups, which makes the use of the multi-objective technique to solve the problem more precise and more appropriate to reality. According to this point of view, we propose in this thesis two approaches for solving a one-dimensional and two-dimensional bi-objective cutting stock problem with setup cost, where we seek to minimize both the total trim loss of raw material and the number of setups to be performed. This technique consists of two steps, the first of which consists in generating all the achievable cutting patterns and the second allows the construction of cutting plans, satisfying the requests, thanks to a subset of these patterns. These different cutting plans represent all the achievable solutions, each of which is characterized by a number of setups and the total quantity of falls.

Keywords: Trim loss, setups, feasible cutting pattern, cutting plan, bi-objective cutting stock problem.

ملخص

استجابة للتطور التكنولوجي الحاصل في إجراء عملية القطع، ظهرت بالفعل عدة امتدادات في مسائل القطع، بما في ذلك تكلفة التركيب التي تنشأ في عمليات القطع الصناعية، وهذا ما دفع العديد من المؤلفين، إلى تكثيف البحث في هذا الاتجاه، باستخدام طرق جديدة للحل، مثل التحسين متعدد الأهداف، لأن العديد من الدراسات أثبتت أن هذا النوع من المسائل لا يقتصر على هدف واحد، بل يتجاوزه في بعض الحالات إلى عدد من الأهداف. فعلى سبيل المثال، التقليل المتزامن للخسارة الكلية للمواد الخام وعدد الأشياء المراد قطعها وكذلك تقليل الخسارة الكلية للمواد الخام وعدد التركيبات هي مسائل قطع ثنائية الاهداف في حين أن تقليل التكلفة الإجمالية للوحة، وتعظيم الربح المتبقي (الطول غير المستخدم للوحة)، والربح المتبقي من عمليات القطع السابقة هي مسألة قطع ثلاثية الأهداف. من بين مسائل القطع ذات الهدفين التي تظهر في الأدبيات، مسألة القطع مع تكلفة التركيب، حيث يعبر التحسين في هذه الحالة عن هدفين متناقضين، وهما الخسارة الكلية للمواد الخام وعدد التركيبات، مما يجعل استخدام الأسلوب المتعدد الأهداف لحل المسألة أكثر دقة وأكثر ملائمة للواقع. فوفقًا لوجهة النظر هذه، نقترح في هذه الأطروحة مقاربتان لحل مسائلة قطع أحادية وثنائية الأبعاد بهدفين، حيث نسعى لتقليل كل من الخسارة الكلية للمواد الخام وعدد التركيبات التي يتم تنفيذها. تتكون هتان التقنيتان من خطوتين، الأولى تتمثل في إنشاء جميع أنماط القطع القابلة للتحقيق، والثانية تسمح ببناء مخططات القطع، وتلبية الطلبات، وذلك بفضل مجموعة فرعية من هذه الأنماط. تمثل خطط القطع المختلفة هذه جميع الحلول القابلة للتحقيق، ويتميز كل منها بعدد من التركيبات والكمية الإجمالية من المواد الضائعة

الكلمات المفتاحية: الخسارة، التركيب، نمط القطع، خطة القطع، مسألة القطع ثنائية الأهداف

TABLE DES MATIERS

CHAPITRE 3 : COMPLEXITÉ ALGORITHMIQUE ET ABÉCÉDAIRE SUR LE PROBLEME DE DECOUPE

CHAPITRE 4 : POSITION DU PROBLÈME ET MODÈLES MATHÉMATIQUES

LISTES DES ILLUSTRATIONS, GRAPHIQUES ET TABLEAUX

INTRODUCTION GÉNÉRALE

Les problèmes de découpe (CSP), représentent un domaine fertile de recherche aussi bien au niveau de ses applications qu'au niveau de sa résolution. Cependant, le poteniel du problème est encore plus important au niveau de ses extensions. Le problème de découpe a été posé par Kantorovich [55], afin de trouver une modélisation cohérente des problèmes reliant l'organisation et la théorie de la production, mais il est resté confiné au cadre académique jusqu'au début des années soixante, les universitaires Gilmore et Gomory [33], qui ont souligné l'importance économique du problème, ce qui a poussé plusieurs chercheurs à entreprendre des recherches portant sur ce problème. Récemment, certains d'entre eux ont identifié un grand nombre d'articles traitant des problèmes de découpe. Ces travaux, en premier lieu, ont été concentrés sur des problèmes standards, alors que les aspects pratiques sont moins fréquemment traités, notamment, sur le nombre d'installations présents dans les industries de découpe. D'autre part, le problème de découpe avec coût d'installation, connait un intérêt croissant depuis le milieu des années 70, avec l'apparition d'heuristiques séquentielles pour ce type de problème.

Le problème de découpe apparaît dans une large variété de secteurs, tels ceux de l'industrie du papier, de l'aluminium et d'autres métaux, du verre, etc. El présente plusieurs variantes. De façon générale, il s'agit d'affecter un ensemble de pièces à un ensemble de grands objets tout en satisfaisant certaines contraintes. Cette idée fut d'abord à un objet ayant seulement une dimension à partir duquel on devait découper plusieurs pièces de différentes longueurs, on obtient un problème de découpe que l'on appelle problème de découpe à une dimension. Alors que s'il s'agit d'un objet à deux dimensions, on obtient un problème de découpe bidimensionnelle. Selon Dyckhoff [27] le problème de découpe est lié à quatre caractéristiques : la dimension, le type d'affectation, les caractéristiques des grands objets, et celles de l'ensemble des pièces. Le premier paramètre compte le nombre de dimensions suivant lesquelles le découpage est effectué. Il y a principalement deux types : la découpe unidimensionnelle, la découpe bidimensionnelle.

L'affectation peut être faite de deux manières : déterminer un sous-ensemble de pièces à découper sur tous les grands objets ou sélectionner un sous-ensemble de grands objets pour

Découper toutes les pièces. Trois cas de figures sont possibles pour les types de grands objets : un seul grand objet, plusieurs grands objets identiques et plusieurs grands objets différents. Enfin, la dernière caractéristique définit la distribution des différentes longueurs des pièces.

Alors que Wäscher et al. [76] ont noté qu'en plus de ces caractéristiques, le problème de découpe est également lié aux critères de définition des catégories de problèmes homogènes et des types de problèmes dont la solution contient des informations sur l'ensemble des patrons de découpe et la valeur correspondante à la fonction objectif. Par la suite, ces auteurs ont proposé une extension aux problèmes de découpe, représentés par le coût d'installation, où ils ont prouvé que l'inclusion de ce coût dans l'étude des problèmes de découpe, réduit le coût total de production et rapproche le modèle mathématique de la réalité. Ils ont également démontré que ce type de problème a un caractère multi-objectif.

Dans ce travail nous nous appuyons principalement sur cette dernière propriété afin de donner une nouvelle vision, de contribuer à résoudre le problème de découpe bi-objectif avec coût d'installation.

Les méthodes de résolution que l'on met en œuvre pour résoudre les problèmes d'optimisation combinatoire tels que le problème de découpe, doit prendre en considération deux facteurs : la qualité des solutions et le temps de résolution. Bien que ces deux facteurs soient généralement liés, parfois il est nécessaire de faire le choix entre trouver une solution (des solutions) optimale(s) ou de se contenter d'une solution (des solutions) approchée(s). Souvent, ce choix est influencé par la nature du problème traité. Toutefois, l'efficacité d'une méthode de résolution dépend de sa complexité temporaire et spatiale. Si la complexité temporaire est bornée par une fonction polynomiale, d'un paramètre caractérisant la taille du problème, alors la méthode est dite efficace ou polynomiale. Dans la nomenclature de la complexité algorithmique, les problèmes d'optimisation combinatoire pour lesquels on ne connait pas d'algorithmes de résolution polynomiaux, sont dits NP-difficiles.

Ces méthodes peuvent être classées en deux catégories : les méthodes exactes qui garantissent l'optimalité de la solution et les méthodes approchées qui perdent en optimalité pour gagner en efficacité. Le principe de base d'un algorithme exact consiste en général à énumérer l'ensemble des solutions de l'espace de recherche de façon implicite. Cependant, ce type d'algorithme ne permet de résoudre que des problèmes de taille modérée. Autrement, le temps de calcul risque d'augmenter exponentiellement avec la taille du problème. Les méthodes approchées constituent une alternative intéressante pour la résolution des problèmes

d'optimisation de grande taille si l'optimalité n'est pas primordiale. Parmi ces méthodes nous citons les méthodes gloutonnes, les heuristiques et les métas heuristiques représentés essentiellement par les méthodes de voisinage et les algorithmes évolutifs tels que les algorithmes génétiques.

Notons qu'il existe d'autres méthodes de résolution combinant des algorithmes approchés et des méthodes exactes. Ces méthodes, appelées méthodes hybrides, représentent un outil assez puissant pour résoudre les problèmes combinatoires. Dans cette thèse nous nous intéressons à la conception d'algorithmes approchés et exacts pour la résolution d'un problème de découpe bi-objectif qu'elle soit unidimensionnelle ou bidimensionnelle. Il s'agit de découper un certain nombre de pièces dont les tailles sont déterminées par les commandes des clients dans un ensemble d'objets de matière première identiques. Celles-ci sont disponibles en quantités illimitées. Alors que le modèle mathématique qui exprime le problème est un programme linéaire en nombres entiers. Pour y parvenir, nous proposons deux méthodes en deux étapes. Dans la première, nous développons deux heuristiques qui nous permettent de générer tous les patrons de découpe possibles. Dans la seconde nous proposons deux méthodes exactes pour construire des plans de découpe et satisfaire toutes les demandes et trouver également les meilleurs plans de découpe en tenant comptent des deux objectifs.

Cette thèse se compose de deux parties principales. La première est consacrée à la définition et à la présentation de problème posé dans cette étude, qui est le problème de découpe. Tandis que la deuxième partie s'intéresse à l'introduction de nouvelles méthodes de résolution et à l'étude d'autres méthodes qui ont été largement utilisées, dans la résolution de ce type de problème qui se pose dans les opérations de découpe industrielle

Plus précisément, nous nous intéressons dans le premier chapitre, à une revue des articles les plus importants publié dans la littérature et qui ont étudié les problèmes de découpe et leur extension. Ces travaux ont été concentrés au premier temps sur des problèmes standards, puis ont été développées pour répondre aux exigences économiques et à la croissance des opérations de découpe industrielle. Par conséquent, nous aborderons dans un premier lieu les travaux liés aux problèmes de découpe monocritère avec coût d'installation. Ensuite, nous passons à d'autres travaux sur les problèmes de découpe multicritère. Et enfin, nous terminons par une conclusion.

Le deuxième chapitre est consacré à l'optimisation combinatoire multi-objectif, où nous présentons d'abord, quelques notions de bases de l'optimisation combinatoires, puis parlons

Ensuite de ce qu'est l'optimisation multi-objectifs, de ses caractéristiques les plus importantes et des méthodes de résolution.

Dans le troisième chapitre, nous nous intéresserons initialement à l'étude de la complexité des problèmes combinatoires. Nous discuterons également leur importance pour proposer des algorithmes permettant de résoudre ce génère de problème. Alors que dans la section suivante, nous allons introduire, le concept de théorie de la complexité pour les problèmes de découpes. Passons par la suite, à la classification des problèmes de découpe qui a été proposée par certains auteurs, Puis nous présentons quelques des difficultés que nous rencontrons dans l'étude des problèmes de coupe.

Le quatrième chapitre est consacré à la présentation du problème étudié, où nous aborderons dans un premier temps quelques notions liées à la découpe unidimensionnelle, nous introduirons ensuite le modèle mathématique classique de découpe, qui était souvent concentré sur la recherche d'une solution pour découper un objet principal en plusieurs parties communément appelées pièces, afin de minimiser le nombre d'objets en stock. De nombreuses études se sont penchées sur ce type de problème. Cependant, dans les applications du monde réel, il existe généralement des contraintes qui rendent la forme du problème différente de la version classique et la rendent plus difficile à résoudre ce qu'on appelle l'extension du modèle de base, c'est ce qui nous a incités à introduire la définition et la notation du problème découpe unidimensionnelle avec coût d'installation. Nous présenterons ensuite un problème de découpe qui sera étudiée dans ce travail représenté par le problème de découpe bidimensionnelle bi-objectif avec cout d'installation, où le but de ce problème est donc de chercher un patron de découpe qui minimise l'aire totale inoccupée par l'ensemble des rectangles du patron et le nombre d'installations.

Dans le cinquième chapitre, nous nous intéressons à présenter les méthodes les plus courantes apparaissant dans la littérature pour résoudre le problème de découpe, qu'elle soit unidimensionnelle ou bidimensionnelle, où ces méthodes sont divisées en : méthodes exactes, heuristiques ou hybrides. Quant aux méthodes exactes qui ont occupé une grande place dans les travaux de nombreux chercheurs, elles sont représentées par la méthode de génération de colonne, méthode par programmation dynamique et par séparation et évaluation qui sont appuyé principalement sur l'exploration de l'espace de recherche pour trouver une solution exacte. Alors que les méthodes heuristiques et hybrides exploitent au mieux la structure du problème considéré dans le but de trouver une solution approchée.

Le sixième chapitre est consacré à la contribution proposée dans cette étude, qui est représentée en deux méthodes hybrides pour résoudre le problème de découpe unidimensionnelle bi-objectif (CSP1DB) et le problème de découpe bidimensionnelle bi-objectifs (CSP2DB). Pour y parvenir, nous élaborons d'abord deux techniques heuristiques qui consiste à générer tous les patrons de découpe réalisables. Puis, nous développons également deux techniques exactes pour trouver l'ensemble de solutions efficaces des problèmes de découpe unidimensionnelle et bidimensionnelle respectivement. La première repose sur le codage des patrons de découpe réalisables pour construire des plans dits de découpe. La seconde est basée sur la division des patrons de découpes sur la base que les patrons appartenant au même ensemble ne traitent pas le même type de pièces. Pour vérifier l'efficacité de ces méthodes, elles ont été testées sur plusieurs exemples trouvés dans la littérature et d'autres sélectionnés dans le domaine industriel tels que Société japonaise des fibres chimiques et Société de l'industrie cinématographique. Une étude comparative a également été menée avec certaines méthodes de la littérature. Dans le cas mono-objectif y compris une étude comparative avec méthode de programmation linéaire en nombres entiers de Gilmore et Gomory (ILP) [33], méthode heuristique séquentielle de Haessler (SHH) [39], méthode hybride de Pazand et Mohammadi (EHH) [63]. Quant au cas des deux objectifs, une comparaison a été faite avec les travaux de Wu et Yang, [79], et Cui et Zhao [11] et l'algorithme génétique proposé par Araujo et al. [2]. Notre première méthode proposée s'appelle, méthode de construction des plans de découpes pour résoudre le problème de découpe unidimensionnelle bi-objectifs étudié dans ce travail, notée par CCPM, Alors que la deuxième méthode vise à résoudre le problème de découpe bidimensionnelle bi-objectifs, notée par CPBM. Où ces méthodes ont fourni des résultats très encourageants et ont montré une concurrence à celles de la littérature. Enfin, nous terminons ce travail par une conclusion et travaux futurs.

Chapitre 1

REVUE DE LA LITTÉRATURE

1.1 Introduction

Le problème de découpe avec leurs différentes dimensions, est un véritable défi pour les chercheurs en recherche opérationnelle. Afin d'améliorer ces problèmes, plusieurs auteurs ont proposé de découper un ensemble de formes géométriques de différentes dimensions, à partir d'un patron ayant des dimensions fixes, dans le cas à une dimension le problème considéré consiste donc à découper sur un objet de taille standard, des petites pièces dont les tailles sont déterminées par les commandes des clients. L'objectif de l'optimisation est souvent de minimiser la perte totale de la matière première. Où la perte est la partie qui reste après avoir découpé toutes les pièces qui serviront à satisfaire les demandes. Alors que dans la forme géométrique à deux dimensions, on parle d'un problème de découpe bidimensionnelle, le but de cette découpe est de maximiser l'aire totale à découpée ce qui peut aussi être considéré comme une minimisation de l'espace non découpé que l'on qualifier alors de perte.

Dans la suite de deux paragraphes suivants, nous aborderons les travaux les plus importants qui ont accompagné les extensions du problème de découpe. Dans la section 1.2, nous passons en revue les travaux qui ont été effectués sur les problèmes de découpe monocritères, et dans la section 1.3 nous discuterons les articles les plus importants sur des problèmes de découpe multicritère, Enfin, terminons par une conclusion.

1.2 Problèmes de découpe monocritère

Le problème de découpe a été posé par Kantorovich [55] afin de trouver une modélisation cohérente des problèmes reliant l'organisation et la théorie de la production.

Il fut repris par Gilmore et Gomory [33] pour la résolution du problème de découpe à une et deux dimensions en utilisant des méthodes de la programmation linéaire (PL) pour résoudre un problème de découpe industrielle en s'appuyant sur la technique de génération de colonnes, cette méthode connue des modifications qui ont été proposées par Haessler [38].

Ses modifications reposent sur le contrôle de la génération des schémas de découpe en utilisant une formulation plus restrictive. Farley [29], ont également proposé des améliorations de cette approche visant à l'adapter à des situations plus pratiques. Cette adaptation se situe au niveau de la génération d'une solution initiale. Les mêmes auteurs [33, 34, 35] l'ont généralisé aux problèmes de découpe à deux et plusieurs dimensions s'appuyant sur d'autres méthodes de résolution exacte. Ces deux auteurs montraient l'intérêt commercial et industriel porté sur ces méthodes.

Peu après, ce problème a été utilisé par Codd [17] pour l'étude des systèmes multiprogrammes et par Garey et Graham [32] pour l'étude des systèmes multiprocesseurs. Par la suite, une généralisation du problème a été proposée pour l'interprétation d'un ensemble de variétés de problèmes industriels (papier, verre, cuir, ...etc.).

Nous mentionnons par exemple le travail d'Umetani et al. [74] où ils se sont intéressés à l'étude du problème de découpe unidimensionnelle avec un nombre donné d'installations, pour minimiser le nombre de rouleaux en stock tout en limitant le nombre de patrons de découpe différents dans une limite donnée par les utilisateurs. En ce qui concerne la résolution de ce problème, ils ont proposé un algorithme de recherche locale qui utilise alternativement deux types de processus de recherche locale avec le voisinage 1-add et le voisinage de décalage, respectivement. Alors que les modèles mathématiques exacts et les algorithmes développés au cours des cinquante dernières années ont été étudiés pour l'un des problèmes d'optimisation combinatoire les plus connus, le problème de découpe, par Delorme et al. [22] où ils ont discuté les principales approches proposées dans la littérature, et ont fourni une évaluation expérimentale des logiciels disponibles sur différentes classes de benchmarks. Martinovic et al, [57] ont abordé le problème de découpe unidimensionnelle consistant à déterminer le nombre minimum de gros rouleaux de stock qui doivent être coupés pour satisfaire les demandes de certaines longueurs d'articles plus petites. Ils ont comparé les formules proposées avec celles de la littérature d'un point de vue théorique et numérique.

D'autre part, de nombreux chercheurs se sont intéressés au problème de découpe bidimensionnelle, où l'on trouve parmi les travaux parus dans la littérature, ce qui a été fait par Cintra et Wakabayashi [10], où ils ont proposé un algorithme basé sur la programmation dynamique et la génération de colonnes pour résoudre un problème du découpe et d'emballage de bandes et le travail de Mellouli et Damak [59], ils ont développé une méthode heuristique

Qui se compose de trois étapes et basée principalement sur une procédure de génération des patrons de découpe. Et afin de contribuer à résoudre les problèmes de découpe bidimensionnelle de petites tailles, Rodrigo et al. [65] ont développé un algorithme porte sur l'utilisation de la méthode de séparation et d'évaluation pour générer des patrons de découpe réalisables. Par la suite, Furini et al. [30] ont proposé un algorithme heuristique utilisant l'idée de génération de colonne à deux étapes, pour la résolution d'un problème de découpe bidimensionnelle, où les résultats expérimentaux ont montré que cet algorithme est meilleur que beaucoup de ceux de la littérature.

Henn et Wäscher [48], ont posé le problème de minimisation des coûts d'installations qui se posent dans les opérations de découpe industrielle lors du démarrage d'un nouveau patron de découpe différent du précédent, il nécessite l'installation d'un nouvel équipement de découpe pour répondre aux exigences technologiques de patron installer. Sur le même sujet Oliveira et al. [62] ont proposé une série des heuristiques liées à la résolution du problème d'emballage de bandes rectangulaires, alors que Cui et al. [12] ont présenté un algorithme qui repose sur la combinaison de la génération de colonnes et un algorithme heuristique qui résolvait les problèmes résiduels à plusieurs reprises jusqu'à ce que toutes les demandes soient satisfaites à l'aide des patrons de découpe en deux étapes. Une autre étude du problème de découpe bidimensionnelle a été réalisée par Hifi et Roucairol [44]. Il s'agit de proposer deux algorithmes, le premier est un algorithme approché qu'est basé sur l'utilisation d'une procédure de programmation dynamique afin de résoudre plusieurs problèmes de sacs à dos bornés et un algorithme exact basé sur la technique de séparation et évaluation. D'autre part une approche heuristique a été développé par Wu et Yang [79], pour résoudre un problème bidimensionnel, tenant compte de l'équilibre entre l'utilisation du matériau et la complexité de découpe, porte sur l'idée que les patrons de découpes par phases sont nécessaires pendant le processus de découpe lors de la séparation d'un ensemble d'éléments rectangulaires de plaques rectangulaires dans les industries manufacturières. En outre, le problème de découpe avec plusieurs tailles de stock a été discuté par Ayasandir et Azizoglu [3].

1.3 Problèmes de découpe multicritère

Les problèmes de découpes multicritère sont apparus dans littérateur au début de la première décennie du XXIème siècle, où les premiers qui ont écrire sur ce sujet fut les scientifiques Kolen et Spieksma [54]. Par la suite vinrent les travaux de nombreux chercheurs dans ce domaine. Parmi ceux qui apparaissent dans la littérature, le problème de découpe bidimensionnelle

Etudiée par Mellouli et al. [60] qui est représenté par la minimisation de la perte totale de la matière première et le nombre d'installations. L'approche proposée à cet égard, combine un algorithme génétique avec un modèle de programmation linéaire pour estimer le front de Pareto optimal de ces deux objectifs. Ainsi que Wuttke et Heese [80], motivés par une entreprise de l'industrie textile, ils ont étudié un problème de découpe bidimensionnelle avec des coûts d'installations dépendant de la séquence et des tolérances admissibles. Ils ont fourni une heuristique séquentielle avec boucle de rétroaction basée sur l'approche de Gilmore et Gomory [33] et ont formulé le problème de séquençage sous la forme d'un programme mixte en nombres entiers.

Vanderbeck [77] a présenté une méthode pour améliorer le placement d'un ensemble de pièces rectangulaires sur une feuille de papier rectangulaire, il a étudié deux types de découpe, le premier étant la découpe bord à bord (coupe guillotine) et le second utilisant une coupe non guillotine où les panneaux à découper placés côte à côte ou les uns sur les autres. L'objectif de l'optimisation est de minimiser la surface totale perdue de la feuille principale utilisé, ainsi que maximisé le nombre total de pièces requis. Alors que Alvarez-Valdés et al. [1] ont développé plusieurs heuristiques de résolution du problème de découpe bidimensionnelle basées sur l'utilisation de la technique de génération de colonnes de Gilmore–Gomory. Ainsi que Tanir et al. [73] ont étudié le problème de découpe unidimensionnelle avec des éléments divisibles, qui se pose dans les industries sidérurgiques. Lors de la planification des opérations de découpe de l'acier, chaque élément peut être divisé en plus petits morceaux, puis ils peuvent être recombinés par soudage. L'objectif est de minimiser à la fois la perte totale de matière première et le nombre de soudures. Pour y parvenir, ils ont proposé un algorithme heuristique basé sur la programmation dynamique.

Belov et Scheithauer [5], se sont intéressés à la résolution d'un problème de découpe bidimensionnelle en deux étapes, en appliquant un algorithme combinant, les procédures de séparation et évaluation et la technique de génération de colonnes. Cintra et Yakabayashi. [10] s'est concentrée sur la résolution d'un problème de découpe bidimensionnelle, en combinant la programmation dynamique et la technique de génération de colonnes. Ainsi que dans une étude du problème de découpe bidimensionnelle, Cui et al. [12] ont proposé un algorithme exact, basé sur la technique de séparation et évaluation, où la borne inférieure initiale est obtenue par un algorithme heuristique. Cui et Huang [15] se sont focalisés sur un problème de découpe bidimensionnelle avec restriction de découpe guillotine en trois étapes.

L'objectif de l'optimisation est de minimiser le coût total des panneaux utilisés et le coût de découpe. Ce problème a été résolu par la génération des colonnes classiques, où les nouveaux patrons de découpe ont été générés par une procédure heuristique, basé sur la création d'un arbre de recherche. Dans une autre étude du problème de découpe bidimensionnelle multi-objectif, Cui et Yang [13] ont proposé une procédure heuristique séquentielle améliorée, appelée heuristique de regroupement séquentiel pour minimiser simultanément deux objectifs, les plaques principales de matière première à découper et les patrons des découpes générer.

Par la suite et dans une étude pour résoudre le problème de découpe bidimensionnelle par des méthodes exactes et approchées. Silva et al. [68] ont introduit certaines contraintes au problème étudié, telles que la longueur de coupe et les valeurs résiduelles de la plaque à découpé, puis ils ont résolu le modèle proposé par un solveur commercial Cplex. La conclusion prétendait qu'il faut beaucoup de temps de calcul, pour obtenir une solution optimale.

Gonzalez et al. [36] ont étudié un problème de découpe tridimensionnelle avec deux objectifs : le volume total perdu et le poids des éléments placés. Ils ont appliqué des algorithmes évolutionnaires et ont développé une heuristique de remplissage à plusieurs niveaux pour obtenir tous les vecteurs objectifs non dominés.

D'Armas et al. [21] ont considéré deux objectifs : le bénéfice total et le nombre de coupes. Pour générer l'ensemble des vecteurs objectifs non dominés, ils ont proposé un algorithme génétique de tri non dominé et algorithme évolutionnaire de front de Pareto et un algorithme évolutionnaire basé sur des indicateurs. Ils ont montré la supériorité de leurs algorithmes sur les simples procédures heuristiques.

1.4 Conclusion

Dans les deux sections précédentes 1.2 et 1.3, nous avons passé en revue les travaux les plus importants liés à l'étude des problèmes de découpe unidimensionnelle et bidimensionnelle, mono et multicritère qui ont accompagné le développement industriel dans ce domaine, plusieurs algorithmes de résolution ont été construits. Plus récemment encore, d'autres auteurs, ont continué de développer des heuristiques, afin de résoudre ce génère de problème et ont présenté des idées innovantes, que ce soit au niveau de la modélisation mathématique ou au niveau des méthodes de résolution, alors que l'optimisation multi-objectifs, a été l'un des moyens les plus pratiques qui nous permet de fournir les solutions les plus appropriées à des problèmes réels. Il ressort également de ces articles les larges utilisations des problèmes de

Découpe dans divers domaines industriels. C'est ce qui nous a motivés dans cette thèse à contribuer à résoudre le problème de la découpe avec cout d'installation, comme nous le verrons dans les chapitres suivants.

Chapitre 2

OPTIMISATION MULTI-OBJECTIF

2.1 Introduction

Dans la littérature, il existe deux types d'approches d'optimisation, la première est l'optimisation mono-objectif, qui se base sur la minimisation (ou la maximisation) d'une seule fonction objectif, où le but est de trouver la meilleure solution, appelée solution optimale. D'autre part, l'optimisation multi-objectif optimise simultanément plusieurs fonctions objectifs, qui sont souvent contradictoires, on cherche à trouver la meilleure solution suivant un ensemble de performance du problème. Le résultat d'un problème d'optimisation multi-objectif est généralement un assortiment de solutions, qui se distinguent par différents compromis réalisés entre les objectifs. Cet assortiment est connu par Pareto-optimal. Donc le but de l'optimisation multi-objectif est d'obtenir les solutions de Pareto, par conséquent, à connaître l'ensemble des compromis possibles entre les objectifs.

Dans ce chapitre, nous introduisons quelques concepts de base sur l'approche multi-objectif. Tout d'abord, nous discuterons de certaines définitions de base de l'optimisation combinatoire, telles que : fonction objectif, espace de recherche, espace réalisable, nous présentons par la suite l'approche multi-objectif avec quelques méthodes de résolution.

2.2 Principaux concepts en optimisation

Fonction objectif 2.1 : Une fonction objectif est une fonction qui modélise le but à atteindre dans le problème d'optimisation sur l'ensemble des critères. Il s'agit de la fonction qui doit être optimisée, elle est notée F(x) de manière générale F(x) est un vecteur : $F(x) = [f_1(x), f_2(x) \ldots f_k(x)]$ Elle est aussi appelée : critère d'optimisation, fonction coût, fonction d'adaptation, ou encore performance.

Paramètres 2.2: Un paramètre du problème d'optimisation, est une variable qui exprime une donnée quantitative ou qualitative sur une dimension du problème : coût, temps, taux d'erreurs...etc. Ces paramètres correspondent aux variables de la fonction objectif. Ils sont ajustés pendant le processus d'optimisation, pour obtenir les solutions optimales. On les appelle aussi variables d'optimisation, variables de conception ou de projet.

Vecteur de décision 2.3: Un vecteur de décision est un vecteur correspondant à l'ensemble des variables du problème, il est noté : $x = [x_1, x_2, ..., x_n]^T$ avec : n le nombre de variables ou dimension du problème et x_k la variable sur la dimension Ƙ.

Critère de décision 2.4: est un critère sur lequel sont jugés les vecteurs de décision pour déterminer le meilleur vecteur. Un critère peut être une variable du problème ou une combinaison de variables.

Contraintes 2.5: Une contrainte du problème est une condition que doivent respecter les vecteurs de décision du problème. Une contrainte est notée : $g_i(x)$ avec i = 1, …, m.

Espace de recherche 2.6: représentant l'ensemble des valeurs pouvant être prises par les variables.

Espace réalisable 2.7: représentant l'ensemble des valeurs des variables satisfaisant les contraintes. Dans la figure 2.1, on a : $\underline{x_i}$ et $\overline{x_i}$ représentent la borne inférieure et supérieure pour x_i .

C : représente l'espace de recherche égal au produit cartésien des domaines des variables.

X : représente l'espace réalisable délimité par les contraintes. Cette figure illustre en dimension deux ($C = \Re^2$) le fait que : $X = \{x/g_i(x) \leq 0, x \in C\}$

L'espace réalisable X (délimité par des contraintes) est un sous-ensemble de C, l'espace de recherche.

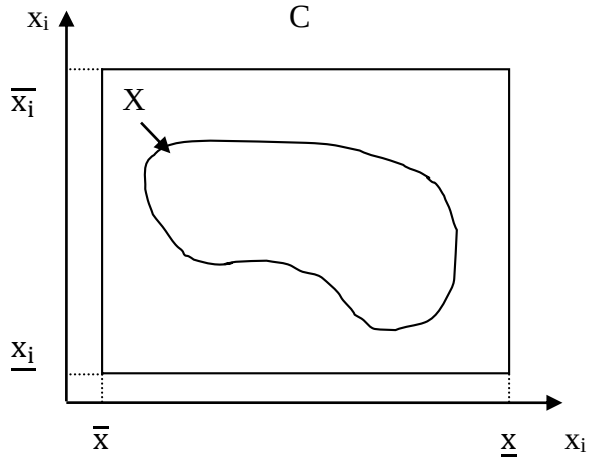

Figure 2.1 : espace de recherche et espace réalisable

Espace des objectifs 2.8: ensemble d'image de l'espace de recherche, déterminé par toutes les valeurs possibles des fonctions objectifs.

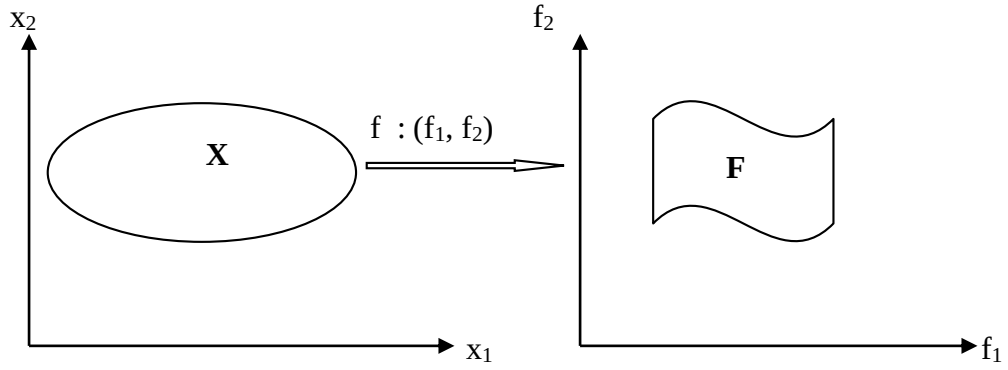

Figure 2.2 : Définition de X, F et f

2.3 Problème d'optimisation multi-objectif

Les problèmes qui surgissent dans des secteurs très divers. Peut fréquemment être exprimé sous la forme générale d'un problème d'optimisation mono- objectif. C'est-dire que la fonction objectif est définie pour maximiser ou minimiser un unique critère. On peut par exemple minimiser le coût d'un produit dans le cadre d'une production industrielle, ou bien maximiser les profits faits par une entreprise. Cependant, les problèmes réels ne sont pas toujours aussi "simples". En effet, il arrive parfois que l'on ait besoin de définir, pour un même problème, plusieurs objectifs à optimiser. Par exemple, on peut avoir besoin de minimiser le coût d'un produit tout en maximisant sa qualité. Dans ce cas, on a deux objectifs à optimiser en même temps, sachant qu'ils peuvent être contradictoires. Effectivement, dans notre exemple, le fait de minimiser le coût du produit aura un impact sur sa qualité, et inversement.

L'optimisation multi-objectif fait partie des familles de méthodes d'optimisation combinatoire [19]. Elle trouve ses origines au cours du XIXème siècle dans les travaux en économie d'Edgeworth [28] et de Pareto [64].

On peut formaliser un problème d'optimisation multi-objectif par la minimisation (ou maximisation) d'une solution $x = (x_1, ..., x_n)$ par m fonctions objectifs.

Chacune optimisant un critère du problème qui peut être défini par :

$$\begin{aligned} &\text{Optimize } \big(F(x)\big) = \big(f_1(x), f_2(x), \dots, f_k(x)\big) \\ &\text{sous } x \in D \end{aligned} \tag{2.1}$$

On dit alors que $F(x)$ est le vecteur de fonctions objectifs et f_i les objectifs optimisé (ou critères de décision) avec $k \geq 2$ le nombre d'objectifs. Sans perte de généralité nous supposerons par la suite que nous considérons des problèmes de minimisation. On peut d'ailleurs ajouter au problème un ensemble de contraintes à satisfaire, de la même manière que pour les problèmes d'optimisation linéaire. On doit alors distinguer la notion d'espace de recherche, représentant l'ensemble des valeurs pouvant être prises par les variables, et la notion d'espace réalisable, sous-espace des variables satisfaisant les contraintes, ici représente par D. L'image de l'espace de recherche par la fonction objectif F est appelé espace des objectifs (ou espace des critères).

2.4 Notions de dominances et d'optimalité

Une solution réalisable $x^* \in D$ est Pareto (ou efficace) si et seulement s'il n'existe pas de solution $x \in D$ telle que x domine x*.

On dit d'une solution $y = (y_1, y_2, \dots, y_k)$ qu'elle domine une solution $z = (z_1, z_2, \dots, z_k)$, dans le cas d'une minimisation d'objectifs, ssi $\forall i \in \{1 \dots n\}$, $f_i(y) \leq f_i(z)$ et $\exists i \in \{1 \dots n\}$ tel que $f_i(y) < f_i(z)$.

Ainsi, toute solution de l'ensemble Pareto peut être considérée comme meilleur compromis, puisque aucune amélioration ne peut être faite sur un objectif sans dégrader la valeur relative à un autre objectif. Ces solutions forment le front Pareto.

Dans le cas d'un problème bi-objectif (deux objectifs à minimiser par exemple), les solutions efficaces peuvent être identifiées visuellement dans l'espace objectif, comme étant celles pour lesquelles le rectangle inférieur gauche formé par la solution et le point (0, 0) est vide de solution réalisable (voir Figure 2.3).

Attachée à la notion de voisinage, la notion de Pareto localement optimale qualifie une solution qui n'est dominée par aucune solution de son voisinage.

Définition 2.1 [voisinage] : le voisinage d'une solution x est composé des solutions pouvant être atteintes depuis x à l'aide d'une transformation élémentaire, alors appelée opérateur de voisinage.

2.5 Points particuliers

En vue d'avoir certains points de références permettant de discuter de l'intérêt des solutions trouvées, des points particuliers ont été définis dans l'espace objectif. Ces points peuvent représenter des solutions réalisables ou non.

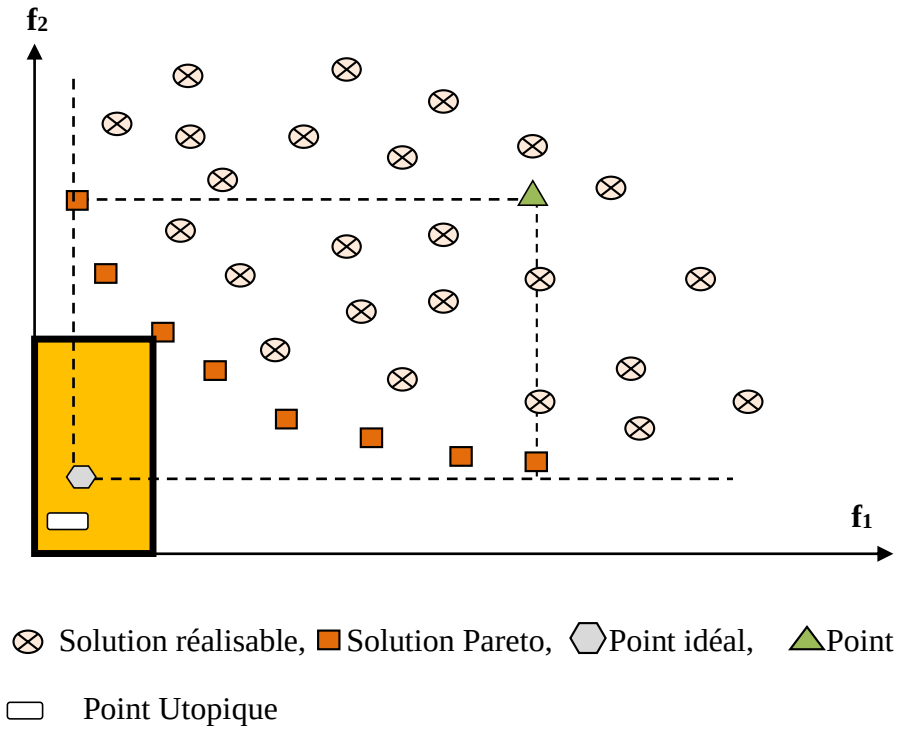

Figure 2.3 : Illustration des différentes solutions

1. Tout d'abord, le point idéal z^I est le point qui a comme valeur pour chaque objectif la valeur optimale de l'objectif considéré.

$$Z^I \text{ tel que } \forall\, i \in \{1... n\},\ f_i(z^I) = \text{opt}_{x \in D} f_i(x) \qquad (2.2)$$

Ce point ne correspond pas à une solution réalisable car si c'était le cas, cela sous-entendrait que les objectifs ne sont pas contradictoires et qu'une solution optimisant un objectif

Optimise simultanément tous les autres, ce qui ramènerait le problème à un problème ayant une seule solution Pareto optimale.

2. De ce point idéal peut être défini le point utopique z^U de la façon suivante :

$$z^U = z^I - \theta^U \tag{2.3}$$

Où $\theta > 0$ et U est le vecteur unitaire ($U = (1, \ldots, 1) \in R^n$). Il est clair, de par sa définition, que ce point n'est pas réalisable.

3. Enfin le point Nadir qui est défini en bi-objectif par :

$$z^N \text{ tel que } \forall i \in \{1...2\},\ f_i(z^N) = \text{opt}_{x \in D / f_i(x) = f_j(z^I)} f_j(x) \text{ avec } j \neq i \tag{2.4}$$

Cela revient donc à affecter pour chaque objectif du point Nadir la meilleure valeur possible parmi les solutions optimisant l'autre objectif.

2.6 Structure du front Pareto

L'objectif est donc de fournir aux décideurs un ensemble (le plus complet possible) de solutions Pareto, afin qu'ils puissent ensuite choisir les solutions qui les intéressent le plus. Une question se pose donc sur la nature de ces solutions Pareto et la nécessité de les obtenir toutes. Une étude de la frontière Pareto doit donc être réalisée.

2.6.1 Front minimal et front maximal complet

La définition de front se réfère à l'espace des objectifs. Une solution appartient au front si elle n'est dominée par aucune autre solution réalisable.

2.6.2 Solutions supportées et non supportées

Sur le front Pareto, deux types de solutions peuvent être différenciées : les solutions supportées et les solutions non supportées. Les premières sont celles situées sur l'enveloppe convexe de l'ensemble des solutions et peuvent donc être trouvées à l'aide d'une agrégation linéaire des objectifs. Elles sont donc plus simples à obtenir que les solutions non supportées.

D'ailleurs, les premiers travaux en optimisation combinatoire multi-objectif se sont pour la plupart focalisés sur la recherche de ces solutions supportées en optimisant des combinaisons linéaires des objectifs utilisant différents vecteurs de poids.

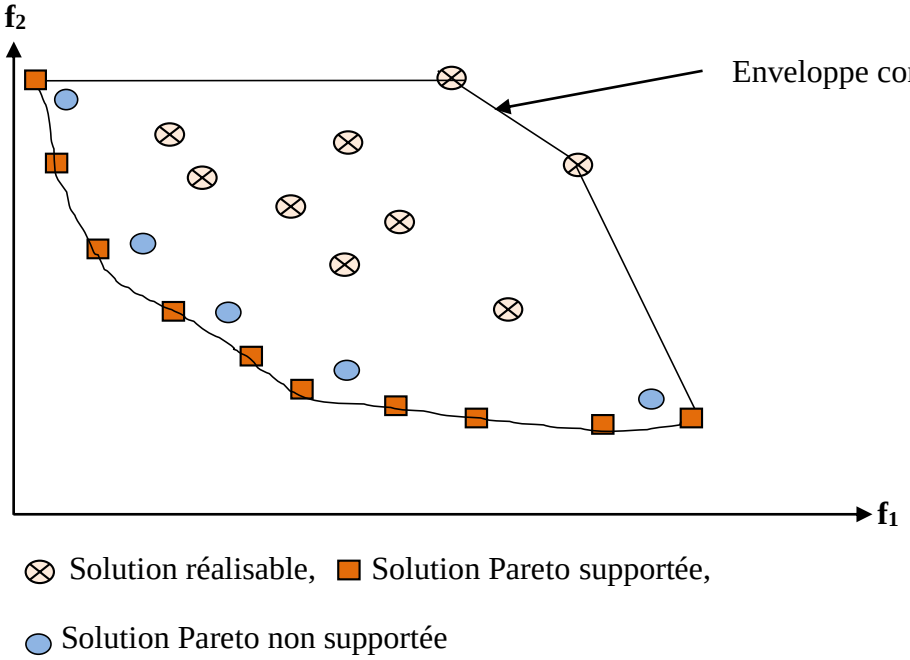

Figure 2.4 : Illustration des différentes solutions

2.7 Quelques méthodes de résolutions

Il est bien connu que l'optimisation multi-objectifs ne possède pas une seule solution optimale, mais potentiellement plusieurs, en fonction des objectifs que l'on suppose plus ou moins importants. Il est alors assez difficile d'établir une définition de ce que doit être l'optimum. On fait alors intervenir un décideur, qui va choisir, parmi les solutions possibles, celle qui lui semble la meilleure.

Le décideur peut intervenir dans différentes phases de l'optimisation. Les méthodes que nous allons présenter peuvent alors être classées en trois catégories :

- Méthodes a priori : Le décideur intervient en amont de la méthode d'optimisation. Il peut par exemple pondérer les objectifs afin de décrire leurs importances.
- Méthodes a posteriori : Le décideur intervient en aval de l'optimisation. Il dispose alors d'un ensemble de solutions et peut choisir celle qui lui semble la plus intéressante.
- Méthodes interactives : Tout au long de la recherche, le décideur peut intervenir pour carterou favoriser des solutions.

2.7.1 Méthodes agrégées

Ces méthodes reposent sur la connaissance du problème par le décideur. Il sait comment attribuer différentes importances aux critères à optimiser. On dit alors que le décideur optimise une fonction d'utilité $f = f(f_1, f_2, \ldots f_k)$.

Les méthodes d'agrégation les plus simples utilisent de mise à l'échelle de chaque critère, afin de pouvoir les additions (modèle additif) ou bien le multiplier (modèle multiplicatif). Les critères peuvent agrégés à optimiser peuvent alors être regroupés dans une unique fonction objectif.

Une autre technique très utilisée est la moyenne pondérer : elle consiste à additionner tous les objectifs en leurs affectant un coefficient. Cette pondération représente l'importance relative que le décideur attribue à chaque objectif :

$$f(x) = w_1 \times f_1(x) + w_2 \times f_2(x) +, \ldots, w_n \times f_k(x) = \sum_{i=1}^{k} w_i \times f_i(x) \qquad (2.5)$$

Cette méthode a cependant plusieurs inconvénients. Tout d'abord, les poids doivent être attribués le décideur avant la phase de l'optimisation. Il ne sait donc pas à l'avance le type de résultat qu'il peut obtenir, ce qui rend cette pondération difficile. De plus, dans le cas où le front de Pareto est non convexe. Certaines solutions peuvent ne pas être accessible en utilisant cette méthode.

L'avantage des méthodes agrégées est leur simplicité de l'utilisation. En effet, l'optimisation multi-objectif est transformée en optimisation mono-objectif, que l'on sait résoudre par des méthodes classiques. Leur inconvénient principal est que le décideur doit avoir connaissances approfondies du problème pour pouvoir paramétrer correctement la fonction objectif. De plus, il est souvent difficile, pour un décideur, de comprendre le sens de chacun de paramètres. Ce genre de méthodes requiert donc un nombre de test important pour déterminer l'influence de chaque objectif.

2.7.2 Méthodes basées sur l'équilibre de Pareto

Ces méthodes sont les plus utilisées en optimisations multi-objectif. Elles reposent sur le postulat de Pareto ″ Il existe un équilibre tel que l'on ne peut pas améliorer un critère sans détériorer au moins un des autres critères″. En effet, on ne peut pas définir la valeur optimale d'un problème d'optimisation multi-objectif. Il existe donc un ensemble de valeurs optimales, que l'on appelle ″Pareto optimale″, selon le critère de dominance au sens de Pareto.

Définition 2.2 [Dominance] : soient $\{f_i, i \in 1 \dots m\}$ un ensemble de critères à minimiser et x et x′ deux solutions de l'espace réalisable. On dit que x domine x′ au sens de Pareto si, $\forall i \in 1 \dots m$ $f_i(x) \leq f_i(x')$. Avec au moins une inégalité stricte.

Ce critère de dominance est la base des méthodes dites de Pareto. Une solution dite Pareto optimale est une solution non dominée, c'est-à-dire qu'aucune autre solution de l'espace réalisable ne domine cette solution, selon le critère de dominance. On obtient ainsi un ensemble de solutions qui forment une frontière d'optimalité, que l'on nomme front de Pareto ou surface de compromis.

Les solutions placées sur le front de Pareto ne peuvent pas être comparées aucune n'étant systématiquement meilleur que les autres sur tous les objectifs. C'est le décideur qui aura pour rôle de choisir la solution à retenir.

De nombreux algorithmes ont été proposés pour résoudre ce type de problème, les plus populaires sont issus des algorithmes évolutionnaires. En effet, au début des années 90, un nombre important d'algorithmes évolutionnaires ont été suggérées pour résoudre des problèmes d'optimisation multi-objectif. On peut citer, par exemple, les algorithmes *Multi Objectif Genetic Algorithm*, dans le domaine de la découpe on a: *A genetic algorithm for the one-dimensional cutting stock problem with setups, A Genetic Symbiotic Algorithm Applied to Cutting Stock Problem with Multiple objective*.

2.7.3 Méthodes non agrégées non Pareto

D'autre méthodes n'utilisent aucun des deux principes énoncés précédemment, possèdent un processus de recherche qui traite séparément les objectifs.

2.8 Conclusion

Dans ce chapitre, nous avons évoqué quelques concepts de base et quelques méthodes de résolution pour l'optimisation multi-objectifs afin de clarifier le domaine mathématique dans lequel nous construisons notre modèle mathématique. À travers ce que nous avons mentionné dans ce chapitre, il devient clair que l'optimisation multi-objectif optimisation diffère de l'optimisation à mono-objectif du concept et de solutions obtenues. Cela rend l'optimisation multi-objectifs plus adaptée à l'étude de certaines des problèmes soulevés dans la littérature.

Chapitre 3

COMPLEXITÉ ALGORITHMIQUE ET ABÉCÉDAIRE SUR LE PROBLEME DE DECOUPE

3.1 Introduction

Les problèmes de découpes sont généralement des problèmes considérés comme difficiles et souvent de grande dimension. Ils sont par conséquent des sujets de recherche de premier ordre, où l'on retrouve beaucoup de méthodes abordées dans d'autres domaines. Bon nombre d'articles ont donc déjà été publiés dans la littérature, explorant de nombreux aspects possibles non seulement des problèmes théoriques existants, mais surtout des applications industrielles mises en œuvre. Ces problèmes sont caractérisés par le fait que les problèmes ayant essentiellement la même structure logique apparaissent sous différents noms dans la littérature, parmi lesquels nous citons : problème d'emballage des bacs (bin packing problem), problème d'emballage en bande (strip packing problem), problème de perte de finition, problème d'assortiment, problèmes de sac à dos (backpack problems), problème de découpe du stock et perte de coupe (cutting stock and trim loss problems). Ce dernier fait partie des problèmes découpe les plus connues qui comprend des problèmes de découpe unidimensionnelles, bidimensionnelles et même tridimensionnelles. Ces problèmes ont été largement étudiés, notamment aux problèmes standards, quant à ses extensions, elle fait l'objet de nombreuses études à l'heure actuelle, parmi eux se trouve cette étude.

Par conséquent, afin de mieux étudier ce type de problème, certains aspects qui s'y rapportent doivent être abordés tels que la complexité de problème combinatoire, qui exprime une mesure de la difficulté d'un problème donné, et c'est ce que nous allons parler dans le premier paragraphe de ce chapitre. Tandis que dans la deuxième section nous nous intéresserons à la théorie de la complexité des problèmes de découpe, alors que la troisième section est consacrée à la classification de ces problèmes. Ensuite, nous verrons certains types de problèmes de découpe. Enfin, nous présentons quelques difficultés que nous rencontrons dans les problèmes de découpe.

3.2 Complexité d'un problème combinatoire

Afin de mesurer la difficulté d'un problème donné et la comparer avec celles d'autres problèmes, pour pouvoir dire qu'un tel problème est plus facile à résoudre que l'autre, nous pouvons calculer la complexité algorithmique de chacun d'entre eux. La complexité d'un problème donné est discutée sur deux côtés : coté temporel (complexité temporelle) et coté spatial (complexité spatiale). La complexité temporelle consiste à évaluer le temps de calcul nécessaire pour résoudre un problème donné. Tandis que, la complexité spatiale permet d'estimer les besoins en mémoire (l'espace mémoire requis) pour la résolution d'un problème donné. La complexité d'un problème donné est estimée en fonction du nombre d'instructions permettant d'aboutir à la solution du problème posé. Elle est influencée par la taille du problème en question. En effet, elle exprime un rapport entre la taille du problème, le temps de calcul nécessaire et l'espace mémoire requis.

3.3 Théorie de la complexité

La théorie de la complexité s'intéresse à l'évaluation de la difficulté des problèmes via l'étude de la complexité de solutions algorithmiques proposées. Elle associe une fonction de complexité à chaque algorithme résolvant un problème donné et mesure les ressources nécessaires pour la résolution d'un problème posé.

Théoriquement, tout problème d'optimisation peut être résolu par une énumération complète de toutes les solutions possibles de l'espace de recherche. De telles méthodes de résolution sont d'ailleurs appelée méthodes énumératives [70]. Cependant, sur certains problèmes, une énumération exhaustive amène des algorithmes de complexité exponentielle (c'est-dire en $O(e^n)$), inutilisables en pratique.

Même si, pour certains problèmes, on connait des algorithmes efficaces de complexité polynomiale (c'est-dire en $O(n^p)$), ce n'est pas généralisable à l'ensemble des problèmes. Nous pouvons donc conjecturer que certains problèmes sont par nature plus difficiles que d'autres, et parconséquent que leur résolution requiert des algorithmes de complexité plus élève. Il est alors intéressant de définir une classification permettant de regrouper les problèmes ayant un même niveau de difficulté.

Il existe de nombreuses classes de problèmes dans la littérature, nous ne mentionnerons ici queles plus représentatives : les classes P et NP [31, 69]. On dit qu'un problème est de classe P

s'il peut être résolu, de manière exacte, par un algorithme de complexité polynomiale. On peut citer par exemples les problèmes bien connus de tri de tableaux, de recherche d'un sous-graphe connexe. La classe NP, quant elle, regroupe les problèmes dont on peut vérifier que n'importe quelle proposition est bien une solution du problème, en un temps polynomial. On peut donc facilement en déduire que $P \subseteq NP$. La question qui fait encore débat aujourd'hui est de savoir si $NP \subseteq P$. Ceci prouverait qu'il existe des algorithmes polynomiaux permettant de résoudre n'importe quel type de problèmes. Or, la conjecture actuelle est plutôt de dire que $P \neq NP$. Ce qui impliquerait qu'il n'existe pas d'algorithmes polynomiaux pour résoudre les problèmes NP. De tels problèmes sont alors catégorisés comme difficiles et leur résolution requiert des algorithmes à complexité exponentielle.

Une dernière classe de problèmes existe, et est particulièrement importante, il s'agit des problèmes NP-complets [50, 51]. Ce sont les problèmes NP les plus difficiles, leur particularité est que tout problème NP peut être transformé en un problème NP-complet en un temps polynomial. Ces problèmes constituent donc le "noyau dur" des problèmes NP, si bien que si l'on trouvait un algorithme polynomial permettant de résoudre un seul problème NP-complet, on pourrait en déduire un autre pour tout problème NP. Parmi les problèmes NP-complets les plus connus, on peut citer : la clique maximum, le problème du sac dos ou bien le problème de découpe.

3.4 La théorie de la complexité de problème de découpe

Les problèmes de découpe sont des problèmes combinatoires, ils sont classés dans la catégorie des problèmes NP-complets et par conséquent les méthodes exactes qui consistent à énumérer de façon implicite toutes les solutions réalisables pour trouver la meilleure solution, sont efficaces que lorsqu'il s'agit de problèmes de taille modérée. Néanmoins, les méthodes approchées constituent une alternative intéressante pour la résolution des problèmes de grande taille.

3.5 Diversité des problèmes de découpe

Les industries sont très souvent confrontées des problèmes dans la découpe, notamment lorsqu'il s'agit de découper des matières premières précieuses dont les pertes sont difficiles à récupérer, leur objectif est de réduire les chutes dans les découpes à une ou deux dimensions.

Ou de placer au mieux les éléments dans une carte électronique ou de remplir au mieux des containers avec des objets. Ainsi, les problèmes de découpe recouvrent différents domaines, pour lesquels une certaine classification est toujours essentielle. L'enjeu économique de ces problèmes est certain. Le gain sur la chute générée, ou sur le temps nécessaire pour réaliser cette découpe, répercutes sur le prix de revient du produit fabriqué.

3.5.1 Classifications des problèmes de découpes et d'emballage

Une typologie donne une vision concise des objets et, ainsi, prépare le terrain pour une recherche orientée vers la pratique. De plus, il aide à unifier les définitions et les notations et ce faisant, facilite la communication entre les chercheurs dans le domaine. Enfin, si les publications sont catégorisées selon une typologie existante, ils offrent un accès plus rapide à la littérature pertinente. Une typologie des problèmes de recherche opérationnelle, en particulier, fournit la base d'une analyse structurelle des types de problèmes sous-jacents, l'identification et la définition de problèmes standards, le développement de modèles et d'algorithmes, générateurs de problèmes.

3.5.1.1 Classification de Dyckhoff

Dyckhoff [25], a proposé une classification pour les différents types de problèmes de découpe. Il a utilisé quatre caractéristiques : la dimension, le type d'affectation, les caractéristiques des grands objets, et celles de l'ensemble des pièces. Le premier paramètre compte le nombre de dimensions suivant lesquelles le découpage est effectué.

1. **Les dimensions** : 1, 2, 3 ou n-dimensions avec n > 3
2. **Le type d'allocation** : L'affectation peut être faite de deux manières : déterminer un sous-ensemble de pièces à découper sur tous les grands objets ou sélectionner un sous-ensemble de grands objets pour découper toutes les pièces.

a) **B** [Belade problem] : placer un nombre donné d'objets dans un nombre non limité de boîtes.

b) **V** [Verladc problem] : remplir un nombre donné de boîtes avec un nombre non limité d'objets.

3. Les types des boites

Trois cas de figures sont possibles pour les types de boites : une boîte, des boîtes de forme identiques et des boîtes de forme différentes.

a) **O** [One] : une boîte,

b) **I** [Idcntical] : des boîtes de forme identiques,

c) **D** [Dwrent] : des boîtes de formes différentes,

4. Les types d'objets

a) **F** [Few] : peu d'objets de formes différentes,

b) **M** [Many] : beaucoup d'objets avec beaucoup de formes différentes,

c) **R** [Relatively] : beaucoup d'objets avec peu de formes différentes,

d) **C** [Congrucnt] : objets de forme similaire,

D'autres paramètres, tels que les restrictions sur les couteaux et la réutilisation de certaines pièces non utilisées peuvent fournir des contraintes additionnelles au problème.

3.5.1.2 Classification de Wäscher

La typologie des problèmes de découpe et d'emballage introduite par Dyckhoff [27] a initialement fourni un excellent instrument pour l'organisation et la catégorisation de la littérature existante et nouvelle. Cependant, au fil des années, certaines lacunes de cette typologie sont également et devenues évident, ce qui a créé des problèmes pour faire face aux développements récents et l'empêché d'être accepté plus généralement, c'est ce qui a poussé Wäscher et al [76], à présenter une typologie améliorée, qui est partiellement basée sur les idées originales de Dyckhoff [27], mais introduit de nouveaux critères de catégorisation. Par ailleurs, un nouveau système de noms cohérent est suggéré pour ces catégories de problèmes. Ils ont utilisé cinq critères, à savoir, dimensionnalité, type d'affectation, assortiment de gros objets, assortiment de petits objets, et forme des petits objets.

Dans une première étape, les deux critères, le type d'affectation et assortiment de petits objets, sont utilisés pour définir les types de problèmes de base. L'application supplémentaire de troisième critère, l'assortiment de gros objets, conduit aux types de problèmes intermédiaires. Enfin, l'application des critères de dimensionnalité et forme de petits objets fournit les types de problèmes raffinés.

Cette classification est résumée par le schéma suivant :

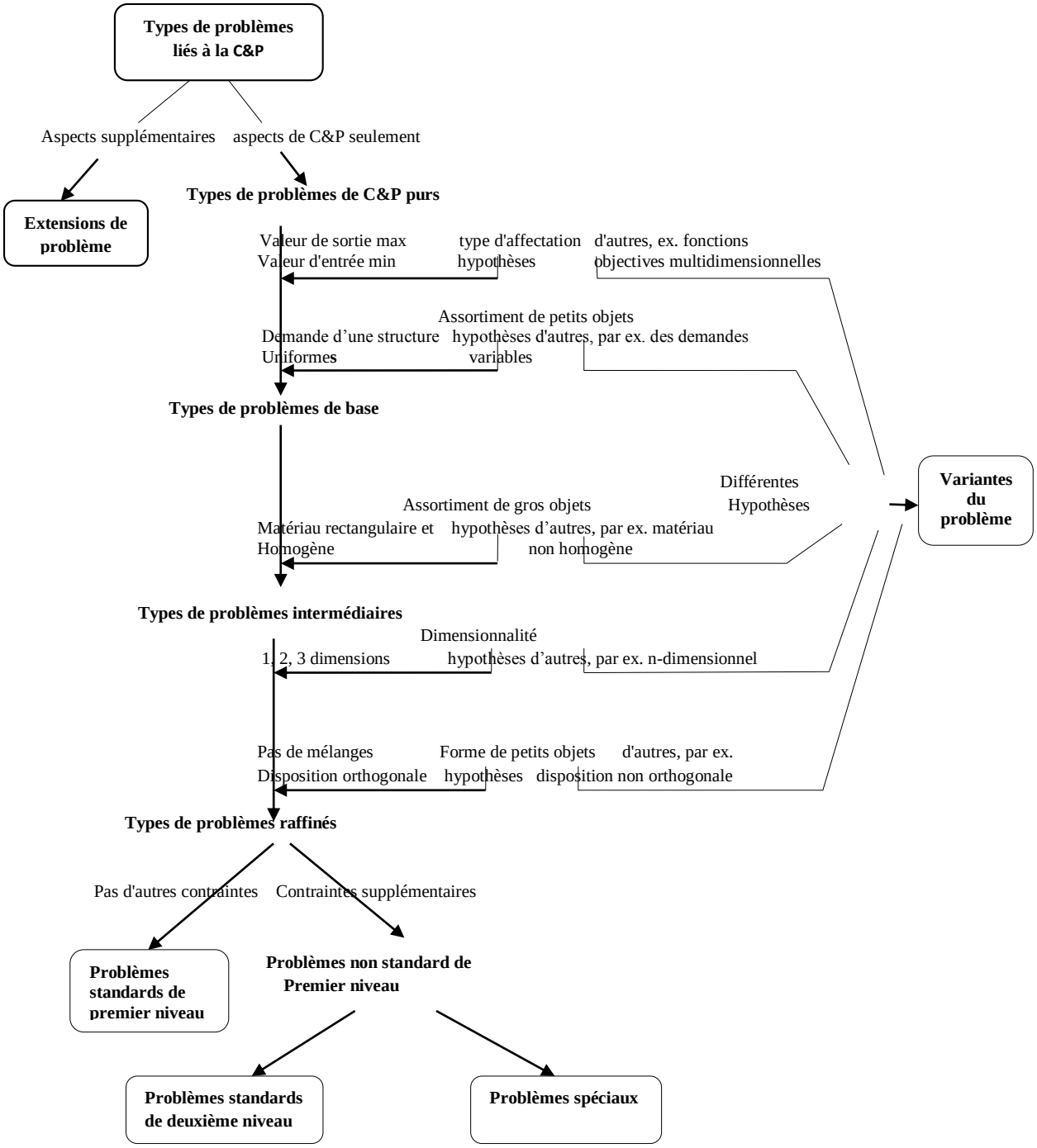

Figure 3.1 : Aperçu des types de problèmes liés aux problèmes de C&P.

3.5.1.3 Extension de Henn et Wäscher [48]

Les processus des installations dont nous avons besoin lors des opérations de découpe, dans le but de minimiser le coût total de la découpe, est un changement de patron de découpe, où à chaque fois qu'un nouveau patron de découpe est lancé (c'est-à-dire un patron différent du précédent) l'équipement de découpe doit être prêt à répondre aux exigences technologiques du nouveau patron installé. Par conséquent les installations de ce type comprennent la perte d'énergie, le temps de production et la consommation de ressources. À la suite de cela, les effets de ce genre doivent être explicitement pris en compte pour la planification et le contrôle des processus de découpe. Cela résulte dans des extensions aux modèles traditionnels. Tandis que cette extension, est une augmentation du problème de découpe de taille de stock unique (Single Stock Size Cutting Stock Problem (SSSCSP)), qu'il a été défini dans la typologie de Henn and Wäscher [48]. L'approche d'intégration des installations dans le SSSCSP serait de transformer ce modèle en un modèle de minimisation du coût total qui comprend le coût de perte de découpe et le coût des installations. Donc selon cette extension la détermination des schémas de découpe et les fréquences d'application correspondantes représentent les éléments de base pour résoudre un problème de découpe. Le problème que nous allons aborder dans ce travail se situe dans cette classe de problèmes de découpe de stock de taille unique sous ses formes unidimensionnelle et bidimensionnelle avec coût d'installation.

3.6 Types de problèmes de découpe

Les problèmes de découpe peuvent concerner différents domaines. Nous en donnons quelques exemples :

3.6.1 Problème de perte de finition

Le problème de perte de finition concerne l'allocation de la liste de commande sur les feuilles de stock données. La liste de commandes décrit l'ensemble des pièces qui doivent être découpé. L'objectif est de minimiser le coût total des feuilles de stock nécessaires pour exécuter la commande.

3.6.2 Problème d'assortiment

Ce problème implique la détermination des tailles de stock nécessaires pour remplir la liste de commandes. La liste de commande doit être affectée à une fourniture de feuilles de stock de sorte que la meilleure sélection de feuilles soit utilisée.

3.6.3 Problème d'emballage du bac (Bin packing problem)

La découpe des bacs peut être trouvée lorsque le matériau de base est disponible sous forme de feuilles plutôt que de rouleaux. La découpe de la bande n'est pas limitée au boîtier rectangulaire. Dans les applications industrielles, ce problème est appelé découpe de stock, par exemple dans l'industrie du verre, du bois et des métaux.

L'objectif est généralement de trouver l'ensemble des feuilles pour accommoder toutes les parties de la liste de commande tout en minimisant le total des matériaux utilisés. Selon l'application, les feuilles peuvent être identiques ou avoir des dimensions différentes.

3.6.4 Problème d'emballage en bande (strip packing problem)

L'industrie du papier est principalement concernée par le problème de l'emballage des bandes, car la matière première est disponible sous forme de rouleaux. D'où le processus d'emballage vise à réduire la hauteur de contenant.

3.7 Quelques difficultés rencontrées dans les problèmes de découpe

Les problèmes de découpe sont classés comme des problèmes difficiles, dont la solution nécessite de surmonter de nombreuses difficultés. Il y a plusieurs étapes entre la visualisation du problème et sa résolution. La découpe de matière n'est, le plus souvent, qu'une partie du problème posé. S'y ajoutent plusieurs autres problèmes que nous résumons ci-dessous :

1. Le problème de disposition des produits sur les aires de stockage. Ce problème d'agencement des stocks est un problème statistique dont l'étude se basera sur les historiques d'utilisation des stocks.

2. Le problème de décisions de découpe qui consiste, connaissant les commandes à réaliser (en fonction des dates de livraison chez les clients), les stocks (disponible se approvisionnement futurs), les moyens disponibles en machines, en ouvriers et en moyens de transport, à décider de la manière dont la découpe doit être effectuée de façon à minimiser un coût donné.
3. Le problème de gestion de commandes qui consiste à décider de la manière dont les décisions de découpe sera appliquée (décision de découpe par commande décision de découpe sur des regroupements des commandes).
4. Le problème de gestion des découpes, de regroupement de découpe de façon à augmenter la productivité, notons que ce problème est très lié au problème précédent (gestion des commandes).
5. Le problème de gestion des réapprovisionnements en tenant compte, bien entendu, des demandes et des stocks, mais aussi des contrariantes des fournisseurs.

Les formes, les dimensions et les propriétés (fragilité, rigidité) des objets et des pièces restreignent souvent le nombre de patrons de découpes admissibles de découpe.

3.8 Conclusion

Dans ce chapitre, nous avons abordé quelques notions de base liées aux problèmes de découpe, afin de prendre en compte tous les facteurs qui influent sur l'étude de problème ou qui interviendraient dans la recherche de solutions à ce problème. Où en premier lieu nous avons mentionné les types de problèmes de découpe et leurs classifications, nous avons également présenté les problèmes de découpes et d'emballage les plus importants qui sont apparus dans la littérature, ainsi qu'à certaines des difficultés qui pouvaient être rencontrées, que ce soit dans leur étude ou dans la réalité lors de la réalisation d'opérations de découpe. Nous sommes également arrivés à la conclusion que les problèmes de découpe sont des problèmes difficiles dont la solution nécessite de surmonter de nombreuses difficultés. Il y a plusieurs étapes entre la visualisation du problème et sa résolution. De plus, la découpe du matériau fait souvent partie du problème à résoudre.

Chapitre 4

POSITION DU PROBLÈME ET MODÈLISATION

4.1 Introduction

Dans ce chapitre, nous nous intéressons à l'étude du problème de découpe dans ses cas unidimensionnels et bidimensionnels où l'on minimise à la fois le nombre d'installations et la perte totale de matière première. Alors que dans le cas unidimensionnel, il s'agit à découper un objet principal en plusieurs parties communément appelées pièces, où les objets en stock sont identiques et disponibles en nombre illimité, tandis que dans le cas à deux dimensions la feuille ainsi que les formes géométriques utilisées sont des rectangles. Où, dans les premières sections, nous donnons quelques définitions du problème de découpe à une dimension et à deux dimensions. Compte tenu du coût d'installation et des différents objectifs et les autres vocabulaires de base utilisé dans la littérature, ce qui nous permettra de mieux décrire le problème de découpe, Nous abordons également le modèle mathématique du problème découpe avec coût d'installation à deux objectifs.

4.2 Problèmes de découpe

De nombreuses entreprises industrielles acquièrent des stocks de matériaux en tailles standard puis les transforment en lots de plus petites dimensions selon la demande. Cette transformation, qui réduit un objet de taille standard à un certain nombre de pièces de tailles commandées, s'appelle découpe ou découpage. Découper selon le dictionnaire Robert, est coupé régulièrement suivant un contour ou un tracé. La forme de ce tracé détermine la nature de la découpe, la découpe est dite unidimensionnelle si le tracé est linéaire, bidimensionnelle si le tracé est selon un plan, et tridimensionnelle si le tracé est selon un espace.

4.2.1 La découpe unidimensionnelle

Certains matériaux, tels que le papier, la Cellophane et les feuilles de métaux, se présentent souvent sous forme de grands rouleaux (objets) qu'on est amené à découper en plus petits objets, appelées en générale pièces, de dimensions variant selon les besoins des clients.

Définition 4.1 [la découpe unidimensionnelle] : Une telle découpe est dite unidimensionnelle dans la mesure où c'est uniquement une seule dimension d'objet principal qui est découpé. Les pièces obtenues sont de tailles plus petites, comme illustre la figure 4.1 suivante :

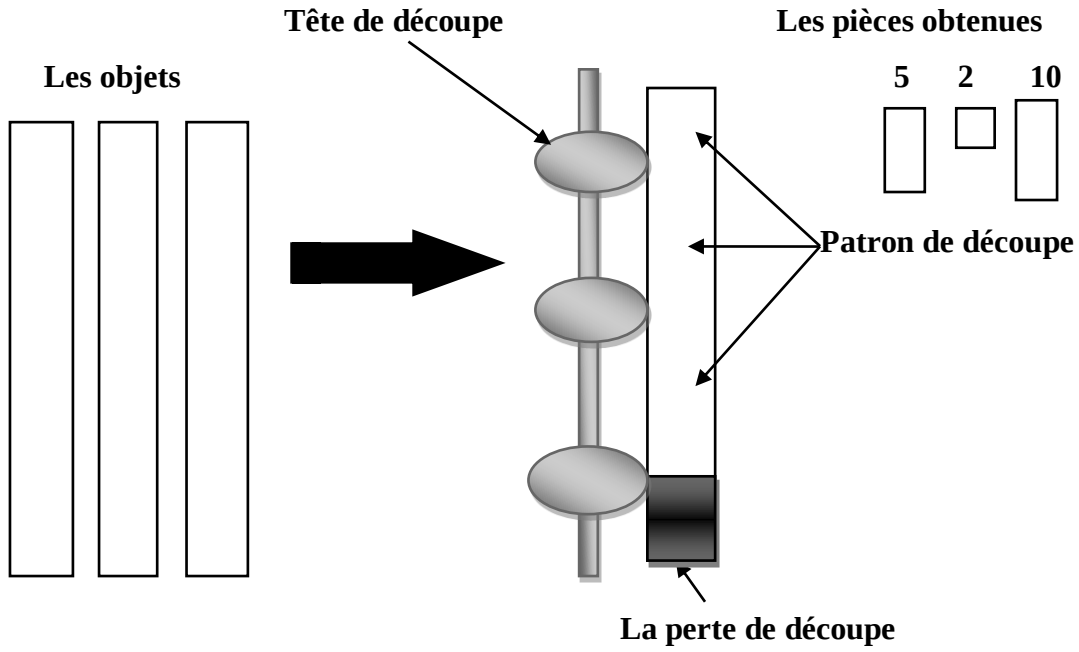

Figure 4.1 : Découpe unidimensionnelle

Définition 4.2 [Patron de découpe] : On appelle un patron de découpe réalisable, un ensemble de pièces découpées dans un objet, et la position de chacune (de celle-ci) dans l'objet.

Définition 4.3 [Plan de découpe] : On appelle plan de découpe réalisable un ensemble de patrons de découpe réalisables pour satisfaire les différents types de demande.

4.2.2 Description de modèle de découpe classique

Le modèle de découpe classique s'intéresse surtout à la découpe des bobines (rouleaux) de papier ou de cellophane, ce cas peut être considéré comme étant le cas le plus général et le plus soulevé dans la littérature traitant du problème unidimensionnel, comme le montre la figure 4.2 suivante :

Figure 4.2 : stock des objets (bobines) de différentes tailles

Nous supposons que la commande comprend n pièces des tailles différentes que dénotons par d_i, i variant de 1 à n. (d = (d_1, d_2, …,d_n) vecteur de commande), Dénotons par W la taille de l'objet principal à découpe et par w_i tailles des pièces découpées dans l'objet principal, en pratique la somme des termes $w_i \times d_i$, pour i variant de 1 à n, est très supérieur de la taille W de l'objet, ce qui impose de découper plusieurs objets, avec chacune contribuant à la satisfaction d'une partie de la commande. Nous supposons que les objets en stock sont disponibles en n'importe quelles quantités.

La découpe de chaque objet peut se faire selon plusieurs façons dont chacune s'appelle patron de découpe, en effet il y a autant de patron de découpe que de solutions (p_1, p_2, …, p_n) au système suivant (où $\mathbb{N}$ l'ensemble des entiers naturels et p_i la quantité produite de chaque type de pièces engendrées par un patron de découpe) :

$$\sum_{i=1}^{n} w_i \, p_i \leq W \qquad (4.1)$$

$$p_i \in \mathbb{N}, \; i = 1, \dots, n \qquad (4.2)$$

La contrainte (4.1) garantit que la somme du produit des tailles de pièces requises multipliée par la quantité produite de chaque type de pièces engendrées par un patron de découpe est

inférieure ou égale à la taille de l'objet principal à découper. Tandis que la contrainte (4.2) indique que les quantités produites de chaque type de pièces engendrées par un patron de découpe sont des entiers naturels.

Et si nous voulons assurer que la quantité produite de chaque type de pièces engendrées par un patron de découpe ne dépasse pas la quantité commandée, nous pouvons ajouter au système la contrainte : $p_i \leq d_i$, i = 1, …, n.

La découpe d'un objet suivant le patron de découpe P_j où j variant de 1 à T, (où T le nombre de patrons utilisent) produit p_{ij} (où p_{ij} est la fréquence de la pièce i dans le patron de découpe j) pièces de chacune des n tailles commandées. Où, ce processus de découpe peut être illustré par la figure suivante :

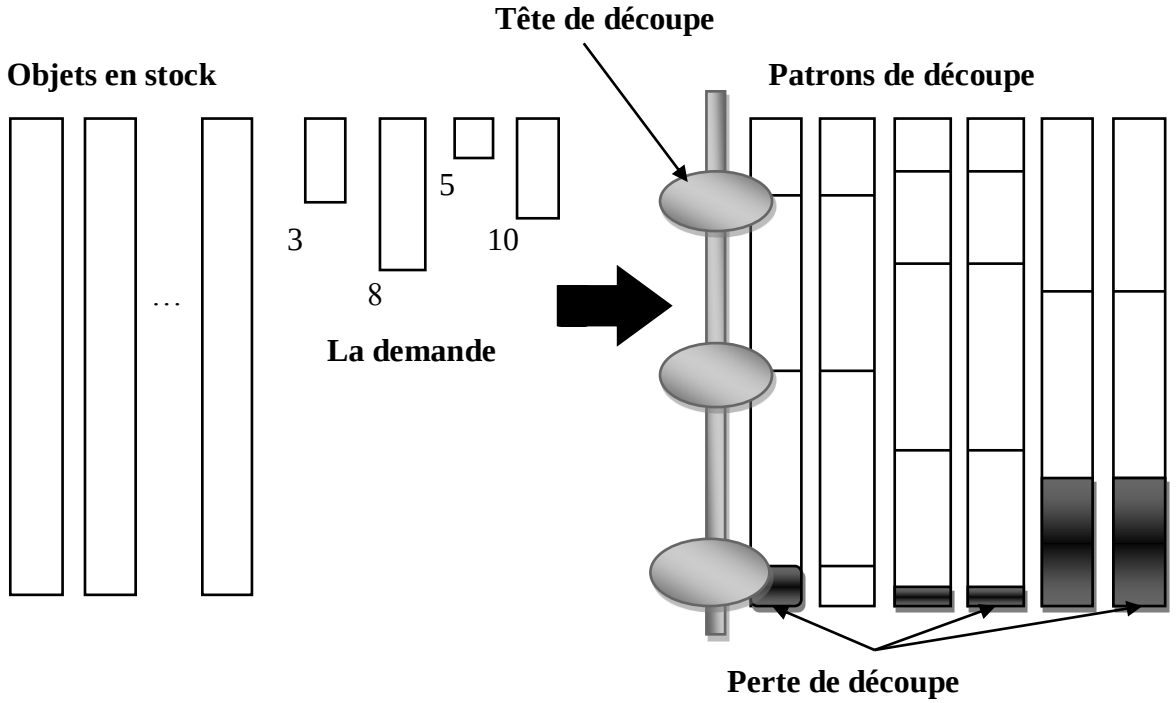

Figure 4.3 : Processus de découpe unidimensionnelle

Modèle de base de découpe 4.1: Dans le modèle de base de découpe, parmi les patrons de découpe trouvés, certains s'avèrent suffisamment commodes pour être répétés plus d'une fois (c'est-à-dire qu'on les appliquer sur plus d'un objet). On note par x_j le nombre d'objet à découper selon le $j^{ème}$ patron P_j (x_j ne peut prendre que des valeurs entières).

En se fixant l'objectif de minimiser la perte totale de la matière première :

$$\min\left(\sum_{j=1}^{T} c_j x_j\right) \qquad (4.3)$$

Où c_j : le coût de perte de l'utilisation de patron P_j, et T le nombre de patrons utilisent.

Et en imposant une production au moins égale à la commande :

$$\sum_{j=1}^{T} p_{ij}\, x_j \geq d_i \qquad i = 1 \ldots n \tag{4.4}$$

Où p_{ij} est la fréquence de la pièce i dans le patron de découpe j

Le problème se ramène à trouver les valeurs des variables entières x_j :

$$x_j \in \mathbb{N}, \quad j = 1, \ldots, T \tag{4.5}$$

Qui optimisent (4.3) sous les contraintes (4.4). Le modèle de base donc être défini comme étant le programme linéaire avec variables entières (4.3) à (4.5).

Extension de modèle de base 4.2 : La formulation de problème de découpe unidimensionnelle, comme ci-dessus, se limite au cas de base, une extension de ce modèle pour s'adapter à des situations plus réelles consiste à minimiser les objets en stocke à découper : dans ce cas la fonction objectif s'écrit simplement :

$$\mathrm{Min}\left(\sum_{j=1}^{T} x_j\right) \tag{4.6}$$

Pour montrer la différence entre les deux fonctions objectifs, la perte totale de matière première et le nombre d'objets en stocke à découper, supposons qu'on a un objet de taille W = 100 mm, et trois types de pièces des tailles respectivement w_1 = 50 mm, w_2 = 30 mm et w_1 = 20 mm, avec des demandes de chaque type $d_3 = d_1 = 1$ et $d_2 = 4$. Avec la première fonction objectif (perte totale de matière première), on satisfait la commande avec deux objets découpés selon les deux patrons de découpe $(1, 1, 1)^t$, et $(0, 3, 0)^t$ (où t dénote le transposer) entrainant une perte de 10 mm. Avec la deuxième fonction objectif (nombre d'objets à découper), par contre, préfère quatre objets toutes découpés selon le même patron de découpe $(1, 1, 1)^t$ pour avoir une perte nulle, mais avec une production significativement supérieure à la commande.

4.3 Formule de Gilmore et Gomory de problème de découpe unidimensionnelle

Gilmore et Gomory [31] ont traité le problème de découpe à une dimension comme un problème de sac dos dans lequel, on associe à chaque élément i d'un ensemble de n éléments et une ressource disponible en quantité limitée, b. Une variable booléenne x_i, égale à 1 si i est sélectionné, égale à 0 sinon. Pour i = 1 à n, on note c_i le profit associé à la sélection de l'élément i et on note l_i la quantité de ressource que nécessite l'élément i, s'il est sélectionné. Les coefficients c_i et l_i prennent des valeurs positives pour tout i = 1 à n. Le profit total obtenu peut

alors s'écrire comme la somme : $\sum_{i=1}^{n} c_i x_i$ et la quantité totale de ressource utilisée comme la somme : $\sum_{i=1}^{n} l_i x_i$. La fonction utilisée par les deux auteurs prend le nom de la fonction knapsack définie par :

$$KP(c) \equiv \begin{cases} \text{Max}(F(x) = \sum_{i=1}^{n} c_i x_i \\ s.c \quad \sum_{i=1}^{n} l_i x_i \leq b \\ \quad x_i \in \{0,1\} \end{cases} \tag{4.7}$$

La contrainte de ressource est appelée "contrainte de sac-à-dos".

4.4 Définitions et notations du CSP1D avec coût d'installation

Tel que le changement de patron de découpe dans le processus de production nécessite une préparation de l'équipement de découpe pour répondre aux exigences technologiques de chaque nouveau patron de découpe installé, les installations de cette sorte comportent la perte de temps de production et la consommation de ressources qui peuvent être évitées ou, au moins, être réduites. Dans la présente section, nous décrivons ce type de problème c'est-à dire le problème de découpe avec coût d'installation.

Donc, comme nous avons cité dans le modèle de base au début de ce chapitre, nous considérons un problème de découpe qui consiste à découper un objet de taille W en plusieurs parties appelées pièces de tailles $w_i < W$ où i = 1,…, n, pour satisfaire n tailles différentes que dénotons par d_i, i variant de 1 à n. (d = (d_1, d_2, …,d_n) vecteur de commande), et à chaque fois qu'on passe d'un patron de découpe à un autre, on change la position des scies dans la découpeuse, cette fixation de la découpeuse engendre souvent des coûts importants (appeler installation), que nous dénotons par c_2 (dus essentiellement à l'arrêt de la production pendant l'opération de fixation). Le nombre de fois où on subit le coût c_2 est égal au nombre de patron de découpe utilisée, soit au nombre de variables $x_j > 0$. Nous introduisons une variable booléenne $\delta(x_j)$ égale à 1 lorsque $x_j > 0$ est égale à 0 lorsque $x_j = 0$, où j = 1, …T. donc en ajoutant le terme : $c_2 \sum_{j=1}^{T} \delta(x_j)$ au coût total à minimiser (c_2 étant le coût d'installation).

Donc, dans l'optimisation mono-objectif le modèle mathématique à minimiser le coût de perte et le coût d'installation est donné par :

$$\begin{cases} \text{Minimizes} \quad c_1\left(W \times \sum_{j=1}^{T} x_j - \sum_{i=1}^{n} w_i d_i\right) + c_2 \sum_{j=1}^{T} \delta(x_j) & (4.8) \\ \text{s.t.} \sum_{j=1}^{T} p_{ij} x_j \geq d_i \quad i = 1, \dots, n & (4.9) \\ x_j \in \mathbb{N} \quad j = 1, \dots, T & (4.10) \end{cases}$$

Où c_1 le coût de perte et c_2 le coût d'installation, où p_{ij} est la fréquence de la pièce i dans le patron de découpe j, x_j est le nombre de fois que le $j^{ième}$ patron de découpe est utilisé et $\delta(x_j) = \begin{cases} 1 \text{ si } x_j > 0 \\ 0 \text{ si } x_j = 0 \end{cases}$. Est une variable booléenne est égale à 1 lorsque le patron de découpe est utilisé, est égale à 0 sinon.

L'ensemble de contraintes (4.9) garantit que la demande pour toute pièce de commande doit être satisfaite.

Alors que, dans le cas du problème de découpe unidimensionnelle bi-objectifs, le modèle mathématique est le suivant :

$$\begin{cases} \text{Min}\left(f_1(x)\right) = \text{Min}\left(W \times \sum_{j=1}^{T} x_j - \sum_{i=1}^{n} w_i d_i\right) \quad \text{(perte)} & (4.11) \\ \text{Min}\left(f_2(x)\right) = \text{Min}\left(\sum_{j=1}^{T} \delta(x_j)\right) \quad \text{(installations)} & (4.12) \\ \sum_{j=1}^{T} p_{ij} x_j \geq d_i \quad i = 1, \dots, n & (4.13) \\ x_j \in \mathbb{N} \quad j = 1, \dots, T & \\ \delta(x_j) = \begin{cases} 1 \text{ if } x_j > 0 \\ 0 \text{ if } x_j = 0 \end{cases} & (4.14) \end{cases}$$

4.5 La découpe bidimensionnelle

Le processus de découpe dans le cas à deux dimensions dépend de deux types de découpe, à savoir, guillotine et non guillotine, pour réaliser le plan de découpe. La découpe de type guillotine qui produit un ensemble de pièces rectangulaires dans une plaque rectangulaire, est la découpe effectuée en coupant d'un côté à son côté opposé parallèlement aux deux autres côtés. Par contre la découpe du type non-guillotine consiste à utiliser le même procédé que dans la découpe guillotine et de plus, elle peut être effectuée tout en marquant des arrêts avant d'atteindre le côté opposé du rectangle à découper.

4.6 Problème de découpe à deux dimensions (bidimensionnelle)

Le problème de découpe à deux dimensions ou problème de découpe bidimensionnelle consiste à découper un ensemble de pièces à deux dimensions contenant différents types ou formats à partir d'une feuille ayant deux dimensions fixes. L'objectif du problème est donc de chercher un patron de découpe qui maximise l'aire totale occupée par l'ensemble des rectangles faisant partie du patron. D'un autre point de vue, cela consiste à chercher un patron de découpe qui minimise l'aire totale inoccupée par l'ensemble des rectangles du patron, surface que l'on appelle perte.

4.7 Problème de découpe bidimensionnelle avec coût d'installation

Le problème de découpe bidimensionnelle avec coût d'installation consiste à découper un ensemble de pièces à deux dimensions contenant différents types ou formats à partir d'une feuille ayant de dimensions fixes. L'objectif de problème est donc de chercher un plan de découpe qui minimise la surface totale perdue et le nombre d'installations à effectuer.

4.8 Problème de découpe bidimensionnelle bi-objectif.

Le problème de découpe bidimensionnelle bi-objectif avec coût d'installation consiste à découper un ensemble de pièces à deux dimensions contenant différents types ou formats à partir d'une feuille ayant de dimensions fixes. Les objectifs étaient de minimiser à la fois le nombre d'installations de la machine et le gaspillage de matériel.

Dans le cadre de ce travail et dans le cas à deux dimensions, nous porterons notre attention au problème de découpe avec coût d'installation bidimensionnelle bi-objectif, où la feuille et les pièces utilisées sont de formes rectangulaires. Ce type de problème est celui qui est le plus fréquemment rencontré dans la littérature.

Définition 4.3 [Patron de découpe guillotine] : Un patron de découpe est dit guillotine s'il est formé successivement par des coupes guillotines, ce qui a pour effet de créer des sous-patrons distincts les uns des autres comme l'illustre la figure 4.4.

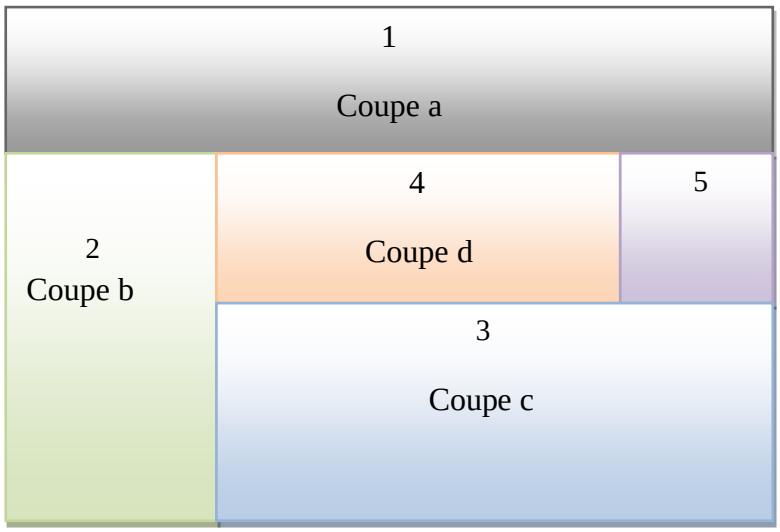

Figure 4.4 : Patron de découpe guillotine d'une feuille

Dans la figure 4.4, la coupe *a* découpe la feuille en deux parties, soient le rectangle 1 et le sous-patron S_1 contenant les rectangles 2, 3, 4 et 5. Ensuite, la coupe b découpe le sous-patron S_1 pour séparer le rectangle 2 et le sous-patron S_2 contenant les rectangles 3, 4 et 5. De même, la coupe c découpe le sous-patron S_2 en deux parties, soient le rectangle 3 et le sous-patron S_3 contenant les rectangles 4 et 5. Finalement, la coupe d sépare les deux rectangles formant le sous-patron S_4.

Définition 4.4 [Patron de découpe non guillotine] : Un patron de découpe non guillotine est un patron formé par des coupes qui ne sont pas nécessairement de type guillotine au sens où certaines de celles-ci ne séparent pas la feuille (ou un sous-patron) en deux parties distinctes.

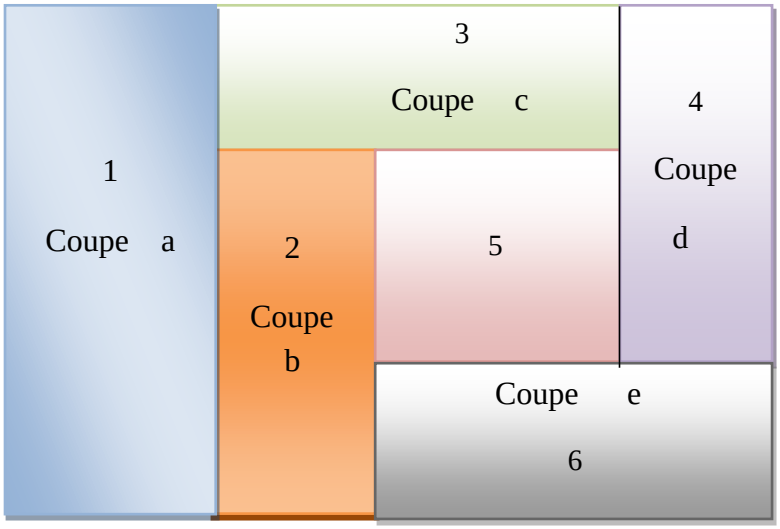

Figure 4.5 : Patron de découpe non guillotine d'une feuille

On remarque dans la figure 4.5 que la coupe *a* découpe la feuille en deux parties soient le rectangle 1 et le sous-patron S_1 et par conséquent forme une coupe guillotine. Par contre, chacune des coupes b, c, d et e ne peut séparer le sous-patron S_1 en deux parties distinctes et donc elles ne sont pas des coupes guillotines. Ainsi, le patron est formé par une coupe guillotine de la feuille et quatre coupes non guillotines rendant ainsi le patron de découpe non guillotine.

4.9 Définitions et notations du CSP2DB

Nous considérons un problème de découpe qui consiste à découper un certain nombre de pièces rectangulaires de longueur l_i et de largeur w_i et donc de surface $s_i = l_i \times w_i$, dans un ensemble de plaques de matière première de longueur L et largeur W et donc de surface S = L × W, elles-mêmes identiques. Celles-ci sont disponibles en quantités illimitées, pour satisfaire un vecteur de commande $d = (d_1, d_2, \ldots, d_n)$ et à chaque fois qu'on passe d'un patron de découpe à un autre, on change la position des scies dans la découpeuse, cette fixation de la découpeuse engendre souvent des coûts importants (appeler installation).

Que nous dénotons par c_2 (dus essentiellement à l'arrêt de la production pendant l'opération de fixation). Le nombre de fois où on subit le coût c_2 est égal au nombre de patron de découpe utilisée, soit au nombre de variables $x_j > 0$. Nous introduisons une variable booléenne $\delta(x_j)$ égale à 1 lorsque $x_j > 0$ est égale à 0 lorsque $x_j = 0$, où j = 1, …T. donc en ajoutant le terme : $c_2 \sum_{j=1}^{T} \delta(x_j)$ au coût total à minimiser (c_2 étant le coût d'installation), la fonction coût à optimiser se ramène à :

$$\text{Minimize} \quad c_1 \left(S \times \sum_{j=1}^{T} x_j - \sum_{i=1}^{n} s_i d_i \right) + c_2 \sum_{j=1}^{T} \delta(x_j) \qquad (4.15)$$

Où c_1 le coût de perte et c_2 le coût d'installation, S est l'aire de la feuille principale, s_i est la surface des pièces requises, x_j est le nombre de fois que le $j^{ième}$ patron de découpe est utilisé

$$\delta(x_j) = \begin{cases} 1 & \text{si } x_j > 0 \\ 0 & \text{si } x_j = 0 \end{cases}$$

Où nous pouvons représenter le processus de découpe à travers le schéma de découpe suivant :

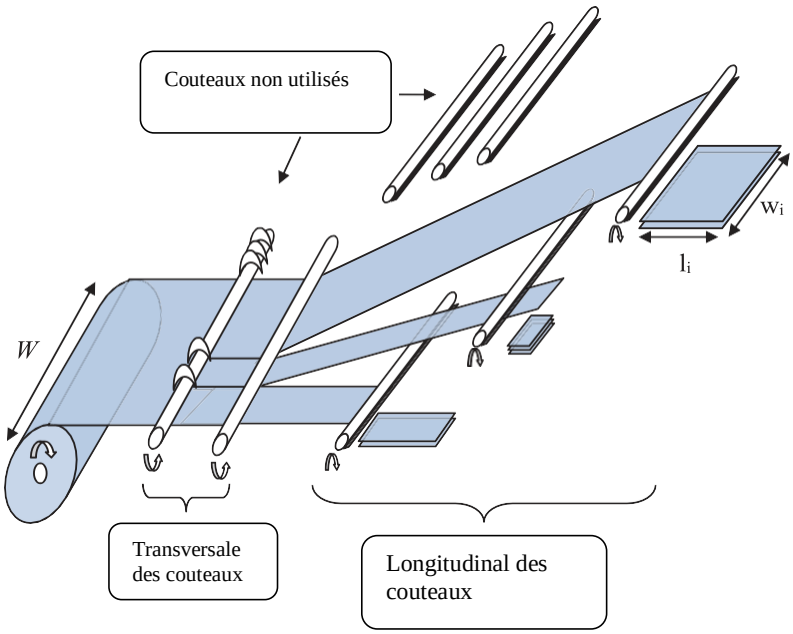

Figure 4.6 : Représentation schématique du processus de découpe bidimensionnelle

Pour formuler le modèle mathématique de ce problème, on introduit les notations suivantes.

Paramètres

- $S = L \times W$ est l'aire de la feuille principale,
- $s_i = l_i \times w_i$ est la surface des pièces requises,
- p_{ij} est le nombre d'occurrences de la $i^{ème}$ pièce dans le $j^{ième}$ modèle,
- d_i est les différents types de demandes,
- T est le nombre de patrons de découpe,

Variables de décision

- x_j est le nombre de fois que le $j^{ème}$ patron de découpe est utilisé,

Fonctions objectifs

Nous avons deux fonctions objectifs :

- La surface totale perdue :

$$\text{Min}\left(f_1(x)\right) = \text{Min}\left(S \times \sum_{j=1}^{T} x_j - \sum_{i=1}^{n} s_i d_i\right)$$

- nombre d'installations

$$\text{Min}\left(f_2(x)\right) = \text{Min}\left(\sum_{j=1}^{T} \delta(x_j)\right)$$

Dans le cadre de ces notifications, nous pouvons formuler un modèle bi-objectif pour le problème de découpe bidimensionnelle avec cout d'installation décrit ci-dessus, sous la forme suivante :

$$\begin{cases} \text{Min}\left(f_1(x)\right) = \text{Min}\left(S \times \sum_{j=1}^{T} x_j - \sum_{i=1}^{n} s_i d_i\right) & (4.16) \\ \text{Min}\left(f_2(x)\right) = \text{Min}\left(\sum_{j=1}^{T} \delta(x_j)\right) & (4.17) \\ \sum_{j=1}^{T} p_{ij} x_j \geq d_i \quad i = 1, \dots, n & (4.18) \\ x_j \in \mathbb{N} \quad j = 1, \dots, T & (4.19) \\ \delta(x_j) = \begin{cases} 1 \text{ if } x_j > 0 \\ 0 \text{ if } x_j = 0 \end{cases} & (4.20) \end{cases}$$

L'ensemble de contraintes (4.18) garantit que la demande pour toute pièce de commande doit être satisfaite.

4.10 Conclusion

Dans ce chapitre, nous avons décrit les deux types de problèmes de découpe étudiés dans ce travail, qui sont le problème de découpe unidimensionnelle et le problème de découpe bidimensionnelle. En commençant par la notion de la découpe unidimensionnelle et bidimensionnelle en passant par la description du problème de découpe de base et ses extensions jusqu'à le problème bi-objectifs avec cout d'installations, ainsi on mettre en lumière l'évolution de la problématique du découpage et de ses prolongements. Où les deux variantes étudiées ici sont considérés comme essentiels dans les problèmes de découpe, comme ils sont les premiers à être étudiés dans ce domaine. De plus, Gilmore et Gomory ont été les premiers à proposer de découper ces formes géométriques et de les modélisés comme une programmation linéaire en nombre entier. Actuellement, ils servent de point de départ à de nombreuses extensions de problèmes de découpe, notamment des problèmes de découpe avec coût d'installation.

Chapitre 5

QUELQUES MÉTHODES DE RÉSOLUTION POUR LES CSP MONO–ET MULTI -OBJECTIF

5.1 Introduction

Dans la littérature, les approches de résolution de problèmes d'optimisation combinatoire sont généralement divisées en deux catégories: les méthodes exactes et les méthodes approchées, où le terme de méthodes exactes regroupe l'ensemble des méthodes permettant d'obtenir la solution d'un problème, en temps raisonnable, alors que les méthodes approchées, comprend les méthodes qui exploitent au mieux la structure du problème étudié afin de trouver une solution approchée, de qualité raisonnable, en un minimum de temps.

Dans le cas du problème de découpe, il est difficile de développer un algorithme polynomial pour le résoudre, vu que ce dernier appartient à la classe NP-complets. Malgré cette difficulté, plusieurs instances de grande taille peuvent être résolues dans des délais raisonnables. Ce résultat impressionnant est la récolte de plusieurs décennies de recherche qui ont exposé les différentes propriétés du problème de découpe qui ont rendu sa résolution moins difficile.

Nous présentons dans ce chapitre quelques méthodes qui ont été largement utilisées dans la résolution de problèmes de découpe, qu'elles soient unidimensionnelles ou bidimensionnelles, dans le cas de l'optimisation mono- et multi-objectifs. Alors que ces méthodes sont exprimées soit par des méthodes exactes, telles que la méthode de génération de colonnes, la méthode par programmation dynamique, par séparation et évaluation, soit par des méthodes heuristiques telles que l'heuristique séquentielle, l'heuristique séquentielle modifié et procédé de génération des patrons de découpe ou méthodes évolutionnaires comme les algorithmes génétiques.

5.2 Méthodes exactes

Les méthodes exactes parcourent, souvent implicitement, la totalité de l'espace de recherche. Ainsi, elles ont l'avantage de produire une solution optimale lorsqu'aucune contrainte de temps n'est donnée. Cependant, le temps de calcul nécessaire pour atteindre une solution optimale peut devenir vite prohibitif, ce malgré les diverses techniques et heuristiques qui ont été développées pour accélérer l'énumération des solutions. Ces méthodes sont se caractérisent par le fait qu'elles permettent d'obtenir une ou plusieurs solutions dont l'optimalité est garantie [51]. Généralement, pour des problèmes des petites tailles ou des tailles moyennes. Malgré les progrès réalisés, le temps de calcul nécessaire pour trouver une solution risque d'augmenter exponentiellement avec la taille du problème.

5.2.1 Méthode de génération de colonnes

Parmi les méthodes exactes utilisées pour résoudre le problème de découpe industrielle, la technique de génération de colonnes qui a été introduite par Gilmore et Gomory entre 1961 et 1965, [33, 34, 35]. Dans leurs travaux, l'algorithme générateur consistait à résoudre un problème du sac-à-dos qui représentait des patrons de découpe.

L'idée centrale de cette technique vient du fait que lors de la résolution d'un programme linéaire (PL) par la méthode de simplexe, plusieurs variables sont hors base et, très souvent, la plupart d'entre elles sont nulles à l'optimum, et peuvent donc être négligées. Reposant sur cette particularité, la génération de colonnes consiste à ne manipuler, à la fois, qu'un sous ensemble de variables (colonnes) de petite taille et à identifier les variables entrant en base au cours de la résolution sans les énumérer explicitement. En effet, le mécanisme de génération de colonnes, s'effectue au sein d'un algorithme, appelé algorithme générateur qui résout généralement un ensemble de sous problèmes indépendants afin de filtrer les meilleures colonnes hors formulation actuelle du programme linéaire, c'est-à-dire celles qui sont susceptibles d'améliorer la solution courante. Ces colonnes correspondent aux variables dont les coûts réduits sont négatifs (problème de minimisation). Autrement dit, la génération de colonnes est basée sur la décomposition du problème original (PL) en un problème maître et un sous-problème.

Le problème maître contient seulement un sous-ensemble de variables et le sous-problème est résolu pour déterminer si des colonnes peuvent être rajoutées au problème maître ou alors

Affirmer le contraire. Pour illustrer les étapes de cet algorithme, nous allons inclure un exemple illustratif comme indiqué dans l'annexe A.

La méthode de génération des colonnes était et est toujours considérée comme un point de départ pour de nombreux travaux, qui sont parus dans la littérature. Citons par exemple les travaux de Juttijudata et Sudjarittham [52] où ont développé un cadre de travail pour résoudre les problèmes de découpe bidimensionnelles, à l'aide d'une méthode de génération de colonne modifiée. Prise en compte des patrons de découpe en deux étapes. La relation entre les patrons de découpe dans la première et deuxième étape conduit à des contraintes supplémentaires. Par conséquent, la méthode de génération de colonne a été modifiée pour gérer ces contraintes. Pour ce faire, ils considèrent un modèle mathématique définit de la façon suivante :

$$\begin{cases} \min(Z) = \sum_{p\geq 1} x_p & (5.1) \\ \sum_{p\geq 1} a_{ip} x_p - \sum_{q=1}^{Q} y_q \geq 0, \forall\, i & (5.2) \\ \sum_{p\geq 1} a_{ijq}\, y_q \geq d_{ij}, \forall\, i, j & (5.3) \\ x_p, y_q \geq 0 & (5.4) \\ x_p, y_q \in I & (5.5) \\ \sum_{i=1}^{I} w_i a_{ip} \leq W, \forall\, p \text{ et } \sum_{j=1}^{I} l_j b_{ijq} \leq L, \forall\, i, q & (5.6) \end{cases}$$

Où x_p est le nombre de feuilles standard découpées avec le patron p à l'étape 1, a_{ip} est le nombre de bandes découpées sur la largeur w_i de la taille standard, i est l'indice des largeurs de bandes allant de i = 1, …, I, où I représente le nombre total de largeurs de bandes, b_{ijq} est le nombre de bandes à la direction verticale le long des w_i avec q patron de découpe, j est l'indice des longueurs de pièces allant de j =1,…,J. où J représente le nombre total de longueurs de bandes, L et W sont la longueur et la largeur de la feuille standard respectivement, l_i et w_i sont la longueur et la largeur de la bonde respectivement, y_q est le nombre de feuilles standard découpées avec le patron q à l'étape 2. La contrainte (5.4) garantit que la somme du produit du nombre de bandes découpées sur la largeur w_i de la taille standard par le nombre de feuilles standards découpées avec le patron p à l'étape 1 est supérieure ou égale à la somme du nombre de feuilles standards découpées avec le patron q à l'étape 2. La contrainte (5.2) garantit que la demande pour toute pièce de commande doit être satisfaite. La contrainte (5.6) garantit que la somme du produit des largeurs des tailles de pièces par la quantité produite de chaque type de pièces engendrées par un patron de découpe p est inférieure ou égale à la largeur de l'objet standard et la somme du produit des longueurs des tailles de pièces par la quantité produite de

chaque type de pièces engendrées par un patron de découpe p est inférieure ou égale à la longueur de l'objet standard.

En ce qui concerne le problème maître intégré dans cette étude, Les auteurs ont formulé un problème maître contraint, avec le moins de variables de décision possible, et ont introduit de nouvelles variables de décision et de nouveaux patrons de découpe dans la base, selon les besoins, d'une manière similaire à la méthode du simplexe. Alors que dans le cas de la résolution du problème de découpe unidimensionnelle. La formulation du sous-problème était la suivante :

$$\begin{cases} \min \mathrm{RCC} = \min\left(1 - \sum_{i=1}^{I} \pi_i a_{ip}\right) = 1 - \max\left(\sum_{i=1}^{I} \pi_i a_{ip}\right) & (5.7) \\ \text{s.c} \ \sum_{i=1}^{I} w_i a_{ip} \leq W, \forall P & (5.8) \\ a_{ip} \in I, \forall\, i, p & (5.9) \end{cases}$$

Où RCC est le coefficient de coût réduit et π_i le prix fictif du problème maître restreint. Cependant, dans le cas bidimensionnel, la contrainte Guillotine et la relation entre x_p et y_q imposent certaines contraintes sur les colonnes a_{ip} et b_{ijq} du problème maître restreint. Ceci a obligé les auteurs à proposer deux modifications. Premièrement, comme les structures des colonnes de la matrice A (matrice de contraintes définie dans l'équation 5.1) ont des structures différentes, le sous-problème pour chaque structure doit être formulé séparément. La deuxième modification vient du fait qu'il y aura trois colonnes générées à chaque étape, plutôt qu'une seule colonne en tant que bases au problème à la fin de chaque itération.

En conclusion, les auteurs ont montré que la solution au problème maître et au sous-problème continue jusqu'à ce que la solution optimale soit lorsque RCC ≥ 0, ou qu'aucune nouvelle colonne indépendante n'est générée. Ils ont également souligné que pour obtenir le prix fictif du problème du sac à dos, le problème maître (contraint) doit être assoupli en problème PL.

Un autre travail de modélisation par la programmation linéaire en nombre entier est donné par Costa [16] pour la découpe de panneaux de bois. L'algorithme doit parcourir toutes les configurations de coupe et retenir les quatre qui minimise la chute, et qui respectent au mieux les proportions de chaque type de rectangles commandés. Il ne reste qu'à déterminer à l'aide d'une méthode de résolution de problèmes de programmation linaire en nombres entiers, le nombre de panneaux découpés suivant chaque configuration de coupe pour réaliser les commandes.

5.2.1.1 Algorithme de génération de colonnes

Algorithme 5.1 : Algorithme de génération de colonnes

Entrée : Une instance du problème (PL),

Sortie : Une solution optimale de (PL),

1. **Initiation** : B_0 : = base réalisable initiale ; Itération k = 0 ; pricing = vrai.
2. **Tant que** (pricing = vrai) **faire**

(a) Calculer la solution de base $\overline{x} = B_k^{-1}.b$,

(b) Calculer les variables duales $\alpha = cB_k.B_k^{-1}$,

(c) Résolution du problème de "pricing" :

Pour ($j \in \mathbb{N}$)

i. **Si** $\widehat{c_j} = c_j - \alpha A_j \leq 0$ **alors**

Pricing = faux ;

ii. **Sinon**

Ajouter la variable j avec $\widehat{c_j} > 0$,

Ajouter la colonne A_j à A' ,

pricing = vrai ;

(d) k := k + 1,

Fin Tant que

Sortir avec une solution optimale de (PL).

5.2.2 Méthode par 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 dans [33] 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 [7] pour répondre au problème de découpe guillotine.

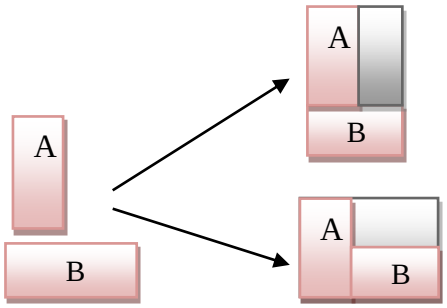

Figure 5.1 : Constructions horizontales et verticales.

Par la suite, Hifi [43, 45] a proposé une amélioration de l'algorithme exact de Viswanathan et Bagchi's [78] pour le problème de découpe guillotine à deux dimensions. Peu après Hifi et Roucairol [44] ont proposé un algorithme exact pour le même problème basé sur une stratégie de construction avec un retour arrière. Alors que, Dusberger et Raidl [26] ont présenté un algorithme basé sur la programmation dynamique composé de 3 étapes. Ils ont considéré un problème de découpe qui consiste à découper un certain nombre de pièces rectangulaires de hauteur h_i et de largeur w_i, dans un ensemble de plaques de matière première de hauteur H et de largeur W elles-mêmes identiques. Celles-ci sont disponibles en quantités illimitées, pour satisfaire un vecteur de commande d = (d_1, d_2, …, d_n). L'objectif est de trouver une meilleure disposition des éléments sur les feuilles de stock sans recouvrement, où ont supposé que X et Y désignent les points raster de la hauteur et de la largeur, respectivement et V_k^v (x, y) et V_k^v (x, y) les valeurs du patron de découpe optimal avec au plus k étapes de découpe appliquées au rectangle (x, y), où la première étape consiste en des coupes verticales ou horizontales, respectivement. Ces valeurs calculées récursivement par :

$$\begin{cases} V_k^v(x,y) = \max\{V_{k-1}^h(x,y), \max_{y'<y,\, y'\in Y}\{V_k^v(x,y') + V_{k-1}^h(x-x',y)\}\} & (5.10) \\ V_k^h(x,y) = \max\{V_{k-1}^v(x,y), \max_{x'<x,\, x'\in X}\{V_k^h(x',y) + V_{k-1}^v(x,y'-y)\}\} & (5.11) \\ \text{Pour chaque } k > 0, x \in X \text{ et } y \in Y \text{ et} & \\ V_0^h(x,y) = V_0^v(x,y)\max\{0, \max_{i\in E}\{h_i w_i \mid h_i \le y, w_i \le x\}\} & (5.12) \end{cases}$$

Les restrictions imposées par les types de demandes, conduit au problème que les solutions optimales, basées sur des solutions partielles sous-optimales. D'autre part, pour une instance

composée d'un seul type d'élément, la récursivité donnera clairement la valeur optimale. Pour minimiser le nombre d'éléments en excès que l'algorithme considère dans la récursivité, l'algorithme en question stocke pour chaque valeur calculée v les éléments utilisés pour obtenir cette valeur. Cela permet de s'assurer que la récursivité ne considère pas les combinaisons de deux sous-patrons dépassant la demande résiduelle d'un ou plusieurs types d'éléments. Lors de la construction du patron de découpe réel basé sur les valeurs optimales, un excès de certains types d'éléments peut encore être choisi, qui est finalement supprimé.

5.2.2.1 Stratégie de construction

- Un plan de découpe forme une construction horizontale si la combinaison des deux rectangles (x_1, y_1) et (x_2, y_2) engendre un rectangle de dimensions $(x_1 + x_2, \max(y_1, y_2)) \leq (L, W)$.
- Un plan de découpe forme une construction verticale si la combinaison des deux rectangles (x_1, y_1) et (x_2, y_2) engendre un rectangle de dimensions $(\max(x_1, x_2), y_1 + y_2) \leq (L, W)$.

A chaque rectangle guillotine R, l'algorithme fait correspondre respectivement une fonction intermédiaire g(R) et une fonction complémentaire h(R). La fonction g(R) représente la valeur de la somme des profits des pièces constituant R et h(R) est l'estimation du reste de la surface du support, noté P = surface (P) / surface (R). C'est aussi la valeur maximale de la région P composée par un sous-ensemble de pièces (en respectant les contraintes sur les pièces). Finalement, le but est de construire le rectangle guillotine de valeur maximale f(.) = g(.) + h(.). Généralement, il n'est pas évident de calculer la valeur de f(.) en un temps raisonnable. Pour trouver une estimation pour la fonction h(R). L'algorithme utilise une borne supérieure pour le problème initial $f_0(R)$.

5.2.2.2 Principe de l'algorithme de programmation dynamique

Le principe de l'approche s'appuie essentiellement sur l'utilisation de deux listes. La première liste, notée Open, comporte au départ une copie de chaque type de pièces. Elle représente la liste globale de l'algorithme. La deuxième liste, notée Clist, est initialement vide. Elle représente la liste intermédiaire qui sert à stocker toutes les constructions susceptibles d'améliorer la solution en cours. A chaque étape de l'algorithme un élément R, de valeur f_0

maximum, est choisi dans la liste Open. Cet élément est un rectangle guillotine qui est immédiatement transféré vers la liste Clist. Par la suite, tous les éléments de la liste Clist (y compris R) sont combinés avec le rectangle choisi R. Ces combinaisons sont réalisées en appliquant les constructions horizontales et verticales.

Chaque rectangle guillotine R_0, issu d'une des constructions considérées, est introduit dans la liste Open s'il vérifie les conditions suivantes :

1. Les dimensions de R_0 ne dépassent pas les dimensions du support rectangulaire.
2. Le contenu de R ne viole pas les contraintes sur les demandes.

L'algorithme s'arrête soit lorsque le rectangle R, choisi dans la liste Open, réalise la valeur $h_0(R) = 0$, soit lorsque, la borne supérieure $f_0(R)$ est inférieure ou égale à la valeur de la meilleure solution en cours.

5.2.3 Méthode par séparation et évaluation

Les algorithmes de séparation et évaluation sont des algorithmes de résolution exacte qui explorent l'espace des solutions en construisant un arbre de recherche. Ils utilisent la stratégie divisée pour régner en partitionnant l'espace de recherche des solutions en sous-problèmes pour les traiter de manière individuelle. Un sous-problème peut lui aussi être divisé en sous-problèmes de sorte que ce processus peut être vu comme la construction d'un arbre ou comme une exploration arborescente. L'efficacité de la méthode par séparation et évaluation dépend essentiellement de la stratégie de développement qui peut être en largeur, en profondeur ou meilleur d'abord. De plus, la méthode du branchement employée et la nature de la fonction d'évaluation utilisée influencent fortement cette méthode.

Dans le cas de problèmes de découpe, cette méthode est souvent utilisée comme une technique de résolution d'un problème sous forme d'un programme linéaire en nombre entier (PLNE). L'idée est de relâcher les contraintes d'intégrité et de résoudre à l'optimal le problème en nombres réels. Le problème initial relâché constitue la racine de l'arbre. Chaque nœud de l'arbre est un programme linaire réel et chaque branchement ajoute une (ou des) contraintes, qui portent généralement sur les variables, et qui tendent à rendre la solution entière.

De cette manière le programme linéaire d'un nœud fils est constitué à partir du programme linéaire du père auquel on a ajouté une ou des contraintes. Les feuilles de l'arbre sont associées

aux solutions entières du problème initial. La meilleure solution entière trouvée pendant l'exploration de l'arbre est la solution optimale.

Un exemple de la résolution par méthode de séparation et évaluation est donné par Rodrigo et al. [65] Pour la découpe de feuilles principales de forme rectangulaire avec des tailles fixes, aux des pièces rectangulaires avec des dimensions et un emplacement connu de chaque patron de découpe réalisable à l'intérieur de la feuille principale est donnée dans le plan de coordonnées cartésiennes. L'algorithme de séparation et évaluation est utilisé pour générer des patrons de découpe réalisables et pour définir l'emplacement de chaque élément dans chaque patron dans le plan de coordonnées cartésiennes. Alors que le problème est modélisé comme un programme linéaire en nombre entier.

5.2.4 Méthode combinant la génération de colonnes et séparation et évaluation

La génération de colonnes appliquée à un problème maître en nombres entiers fournit une solution qui n'est pas toujours entière. Pour obtenir une solution entière, La deuxième approche est une approche exacte appelée Branch and Price dont l'idée de base est de combiner la méthode de génération de colonnes avec la méthode par séparation et évaluation (Branch and Bound). Cette combinaison comprend l'adaptation des principes du branch and bound au contexte de la génération de colonnes en permettant d'expliciter de nouvelles colonnes à chaque nœud de l'arbre, c'est-à-dire en utilisant la méthode de génération de colonnes comme procédure d'évaluation des nœuds.
Soit x la solution calculée par la technique de génération de colonnes en un nœud donné. Deux cas se présentent alors :

- x est une solution entière, alors le nœud est élagué et x sera utilisé comme borne supérieure (problème de minimisation).
- x est une solution non entière, alors un branchement sera effectué.

La recherche arborescente par branch and price permet donc théoriquement de trouver la solution optimale d'un PLNE. Elle peut être également utilisée dans le cadre d'une approche heuristique pour obtenir des solutions de bonne qualité.

5.3 Méthodes heuristiques

Dans la résolution de problèmes d'optimisation combinatoires, l'utilisation de méthodes exactes n'est pas toujours possible, par exemple à cause de temps de calcul trop important ou bien d'une séparation du problème impossible. Dans ces cas, nous utiliserons des méthodes approchées, appelés heuristiques. Il convient néanmoins de souligner qu'une méthode heuristique peut être déterministe ou stochastique.

Le mot heuristique vient du grec ancien eurisko et qualifie tout ce qui sert à la découverte, l'invention et la recherche. Ces méthodes exploitent au mieux la structure du problème considéré dans le but de trouver une solution approchée, de qualité raisonnable, en un temps aussi faible que possible. Typiquement, elles trouvent une solution approchée à un problème NP en temps polynomial. On peut citer des heuristiques très simples commeles algorithmes gloutons [11, 18] ou les approches par amélioration itérative [4]. Le principe des méthodes gloutonnes est de faire une succession de choix optimaux localement, jusqu' ce que l'on ne puisse plus améliorer la solution, et ce, sans retour en arrière possible. Ce principe, assez générique, doit être adapté en fonction de la structure du problème. Les heuristiques sont guidées par des spécificités liées au problème posé.

5.3.1 Heuristique séquentielle

La méthode séquentielle proposée par Haessler pour l'industrie de papier pour résoudre un problème de découpe unidimensionnelle avec coût d'installation est présenté dans diverse articles [37, 38, 39, 40, 41 et 42], et probablement la méthode heuristique la mieux connue dans la littérature, elle est définie par la formule mathématique suivante :

$$\begin{cases} \text{Min}\left(c_r \sum_{j\in J} T_j\, x_j + c_s \sum_{j\in J} \delta(x_j)\right) & (5.13) \\ R_{l_i} \le \sum_{j\in J} a_{ij}\, x_j \le R_{u_i}\ i = 1, \dots m & (5.14) \\ x_j \in \mathbb{N} & (5.15) \end{cases}$$

Où c_r : coût de perte, c_s : coût de changement de patron de découpe lors de la découpe, R_l et R_u les bornes supérieurs et inférieurs de l'objet que le client est prêt à acheter pour que l'usine ait moins de pertes possibles. x_j : le nombre de fois que le $j^{ième}$ patron est utilisé, a_{ij} : le nombre de

pièces de largeurs w_i produite à chaque fois que le patron j est utilisé, T_j : la perte associe au patron j.

La contrainte (5.14) garantit que la somme du produit de nombre de pièces de largeurs w_i produite à chaque fois que le patron j est utilisé, par le nombre de fois que le $j^{ième}$ patron est utilisé, est limitée entre les bornes supérieurs et inférieurs de l'objet que le client est prêt à acheter pour que l'usine ait moins de pertes possibles.

$$\delta_{ij} = \begin{cases} 1 & \text{si } x_j > 0 \\ 0 & \text{sinon} \end{cases}$$

En effet, la procédure de résolution commence par analyser les commandes non encore satisfaites pour fournir deux valeurs descriptives de la situation courante. La première valeur est une estimation du nombre d'objets principaux qui nécessite la satisfaction du reste de commande, elle est égale $\sum_{i=1}^{n} \frac{d_i \times w_i}{W}$, la deuxième valeur est une estimation de nombre moyen de pièces à obtenir d'objets principaux elle est égale à la quantité totale des commandes à satisfaire devisée par la première valeur descriptive, soit $(\sum_{i=1}^{n} d_i W)/(\sum_{i=1}^{n} d_i w_i)$, à partir de ces deux valeurs, on fixe trois paramètres utilisés pour chercher un patron de découpe satisfait trois caractéristiques. Le patron de découpe trouvé est alors utilisé pour découper le maximum d'objets principaux, entraînant un problème avec des nouvelles valeurs pour d_i (réduites par les commandes satisfaites par ce patron de découpe trouvé) de nouvelles valeurs descriptives, de nouveaux paramètres et une nouvelle recherche. La procédure continue jusqu'à l'annulation de toutes les valeurs de d_i. Cependant, dans le cas mono-objectif cette heuristique n'est pas d'attendre la solution optimale, mais d'avoir une bonne solution avec un grand nombre de patrons de coupe à utiliser.

5.3.2 Heuristique séquentielle modifié

La procédure heuristique séquentielle modifiée, est une extension de la technique séquentielle de Hessler [37] est l'une des méthodes de résolution de problème de découpe unidimensionnelle avec coût d'installation. Il s'agit d'un algorithme glouton qui génère des patrons de découpe séquentiels jusqu'à satisfaire toutes les commandes. De nombreux chercheurs ont tenté de développer la proposition originale de Hessler [37]. Parmi eux, Cui et al. [15] qui ont développé un algorithme de génération des patrons de découpes à travers un problème de sac à dos avec un petit ensemble des types de pièces restants. Cet ensemble est

contrôlé selon des seuils spécifiques pour minimiser la perte totale de matière première en ne considérant que certains types de pièces afin de maximiser la fréquence des patrons de découpes. Ainsi que, les mêmes auteurs ont élaboré un algorithme robuste qui génère les patrons de découpe à l'aide d'une heuristique séquentielle de Hessler [37], puis ont utilisé un modèle Linéaire en nombre entier qui minimise la perte totale de matière première et le nombre d'installation. Alors que Mobasher et Ekici [61] ont proposé un modèle linéaire mixte en nombre entier du problème. Ils ont également développé deux algorithmes de recherche locale (pour un cas particulier du problème) et un algorithme heuristique basé sur la génération de colonnes. Mais la modification la plus importante est celui apporté par De-salles Neto et al [25] où ils ont proposé la formule mathématique suivante :

$$\begin{cases} \text{Min}\left(c_r \sum_{j=1}^{n} x_j + c_s \sum_{j=1}^{n} \epsilon_j\right) & (5.16) \\ \sum_{j=1}^{n} p_{ij}\, x_j \geq d_i \qquad i \in \{1,2 \dots m\} & (5.17) \\ x_j \leq M_j \epsilon_j, \qquad j \in \{1,2 \dots n\} & (5.18) \\ x_j \in \mathbb{N}, \qquad j \in \{1,2 \dots n\} & (5.19) \\ \epsilon_j \in \{0,1\} & (5.20) \end{cases}$$

où ϵ_j est une variable binaire dont la valeur est 1 si le patron j est utilisé, zéro sinon et selon les contraintes $x_j \leq M_j \epsilon_j$ le paramètre M_j est limité par la fréquence maximum possible du patron j, c'est-à-dire, $\text{Max}_{\forall\, i/p_{ij}>0}\left\{\frac{d_i}{p_{ij}}\right\}$, où d_i est la demande de différentes tailles de pièce et p_{ij} est la fréquence de la pièce i dans le patron de coupe j, les contraintes $\sum_{j=1}^{n} p_{ij}\, x_j \geq d_i$ garantit que la demande pour toute pièce de commande doit être satisfaite.

Haessler dans son algorithme a utilisé les bornes pour satisfaire la demande, Alors que De-Salles Neto et al, [25] aient noté que l'utilisation de bornes pour satisfaire la demande est une faille dans cette méthode, car elle conduit à la génération d'un grand nombre de patrons de découpe pour fournir une solution approchée, et donc la modification proposée se concentrait principalement sur la réduction du nombre de patrons de découpe qui basés sur le fait que la demande vérifie la fréquence maximale du patron de découpe sans occurrence de surplus.

5.3.3 Procédé de génération des patrons de découpe

Le procédé de génération des patrons de découpe réalisable est une technique heuristique, a été proposé, par Suliman [71]. L'idée centrale de ce procédé, consiste à générer d'une manière indépendante les colonnes de la matrice de contraintes du problème d'optimisation combinatoire de Sac-à-dos.

La génération des patrons découpe réalisables est réalisée par un arbre de recherche. Les niveaux de l'arbre représentent les largeurs demandées qui sont arrangées dans un ordre décroissant avec la plus grande taille au premier niveau tandis que la plus petite taille occupant Le niveau le plus bas de l'arbre. La racine de l'arbre désigne la taille de l'objet disponible pour générer les patrons des découpes. Les arcs du niveau i indiquent le nombre de fois où la pièce i est appliquée dans le patron j. La construction de l'arbre commence de la racine puis de gauche à droite et continue à se déplace vers le bas en ajoutant les tailles additionnelles aux combinaisons déjà indiquées par les arcs précédents. Le chemin de la racine de l'arbre à un nœud terminal représente un patron de découpe réalisable dont a_{ij} composants. Voici l'algorithme proposé.

Algorithme 2 : Algorithme de Génération des patrons découpe réalisables

Entrée : La taille de l'objet principale et la taille des pièces requises,

Sortie : Patrons découpe réalisables et la perte de matière première,

1. Arranger les largeurs requises w_i(i = 1, …, n) dans l'ordre décroissant.
2. Appliquer a_{kj} = le plus petit Int $[(W - \sum_{z=1}^{i-1} a_{zj} w_z) / w_i]$, k = 1,..n
3. Calculer la perte $c_j = W - \sum_{i=1}^{n} a_{ij} w_i$, résulte du patron de découpe j.
4. Placer l'index au niveau i à n -1.
5. Vérifier le nœud courant du niveau i, c.-à-d. le nœud (i, j). Si ce nœud a une Valeur égale à zéro (c.-à-d., $a_{ij} = 0$), alors aller à l'étape 7,

   Sinon, générer une nouvelle colonne j = j +1 et calculer les éléments suivants :

   a. $a_{zi} = a_{z,i-1}\ (z = 1, \dots, i-1)$ Pour remplir les nœuds qui précèdent le nœud courant i, j,

   b. $a_{ij} = a_{ij-1} - 1$ Pour remplir le nœud courant i, j,

   c. Remplir les nœuds restants de la nouvelle colonne j, a_{i+1j}, a_{i+2j}, a_{i+3j}, …, a_{n_j} Utiliser a_{ij} = le plus petit Int $[(W - \sum_{z=1}^{i-1} a_{zj} w_z) / w_i]$.
6. Calculer la perte de découpe $c_j = W - \sum_{i=1}^{n} a_{ij} w_i$

   En résulte du patron de découpe j et aller à l'étape 4,
7. Poser i = i – 1, si i > 0, répéter l'étape 5. sinon, arrêter.

5.3.4 L'algorithme génétique

L'algorithme génétique représente une célèbre méta-heuristique évolutionnaire. Il a été proposé par Jhon Holland en 1975, [49]. L'algorithme génétique s'inspire des mécanismes biologiques tels que les lois de Mendel et la théorie de l'évolution proposée par Charles Darwin, [25]. Son processus de recherche de solutions à un problème donné imite celui des êtres vivants dans leur évolution. Il utilise le même vocabulaire que celui de la biologie et la génétique classique, on parle donc de : gène, chromosome, individu, population et génération.
Un gène : est un ensemble de symboles représentant la valeur d'une variable. Dans la plupart des cas, un gène est représenté par un seul symbole (un bit, un entier, un réel ou un caractère).

Un chromosome : est un ensemble de gènes, présentés dans un ordre donné de manière qui prend en considération les contraintes du problème à traiter.

Un individu : est composé d'un ou de plusieurs chromosomes. Il représente une solution possible au problème traité.

Une population : est représentée par un ensemble d'individus (i.e. l'ensemble des solutions du problème).

Une génération : est une succession d'itérations composées d'un ensemble d'opérations permettant le passage d'une population à une autre.

L'algorithme génétique fait évoluer une population composée d'un ensemble d'individus pendant un ensemble de génération jusqu'à ce qu'un critère d'arrêt soit vérifié. Le passage d'une population à une autre est réalisé grâce à des opérations d'évaluation, de sélection, de reproduction (croisement et mutation) et de remplacement.

L'algorithme commence la recherche avec un ensemble d'individus. A chaque itération de la procédure de recherche, les meilleurs individus sont sélectionnés pour survivre et se reproduire. La sélection des individus est fondée sur leurs qualités qui sont mesurées à partir d'une fonction appelée « fonction objectif ou fonction fitness ». Ensuite, les individus (appelés parents) sont sélectionnés pour subir des opérateurs de croisement et de mutation permettant la génération d'une autre population d'individus (appelés enfants). Les individus de la nouvelle population seront évalués pour remplacer une partie des individus de la population courante.

Le processus de recherche de l'algorithme génétique est fondé sur les opérateurs suivants :
Un opérateur de codage des individus : Il permet la représentation des chromosomes représentant les individus.

Un opérateur d'initialisation de la population : Il permet la production des individus de la population initiale. Malgré que cet opérateur ne s'intervienne qu'une seule fois et au début de la recherche, mais il joue un rôle non négligeable dans la convergence vers l'optimum global. En fait, le choix de la population initiale peut rendre la recherche de la solution optimale du problème traité plus facile et plus rapide.
Un opérateur de sélection : Il permet de favoriser la reproduction des individus qui ont les meilleures fitness (i.e. les meilleures qualités).

Un opérateur de croisement : Il permet l'échange des gènes entre parents (deux parents en général), pour créer 1 ou deux enfants en essayant de combiner les bonnes caractéristiques des parents. Le but de cet opérateur est la création de nouveaux individus en exploitant l'espace de recherche.

Un opérateur de mutation : Il consiste à modifier quelques gènes des chromosomes des individus, dans le but d'intégrer plus de diversité au sein du processus de la recherche.
Un opérateur d'évaluation : Il permet de valoriser la qualité des individus, en se basant sur la fonction « objectif » (la fonction fitness) qui permet de calculer la qualité de chaque individu.
En outre des différents opérateurs permettant de guider la recherche par l'algorithme génétique, ce dernier nécessite un certain nombre de paramètres de base, sur lesquels dépendent les différents opérateurs cités en dessus. Ces paramètres doivent être fixés à l'avance, ils jouent un rôle très important dans la performance de l'algorithme. On parle de : la taille de la population, la probabilité de croisement, la probabilité de mutation et le nombre maximum de génération.

La taille de la population : Elle représente le nombre d'individus de la population. S'il est trop grand, le processus de recherche demande un coût de recherche élevé, que se soit en termes d'espace mémoire ou du temps de calcul nécessaires. Cependant, s'il est trop petit, l'algorithme risque d'être tombé dans le cas de la convergence prématurée à cause du manque de la diversité au sein de la population. Il est préférable donc de choisir une taille moyenne en prenant en considération l'instance du problème à traiter.

La probabilité de croisement : Elle représente la probabilité d'échange de patrimoine (i.e. les gènes) entre deux individus (ou plus). Plus elle est grande, plus elle permet la génération de nouveaux enfants qui peuvent être meilleurs que leurs parents.

La probabilité de mutation : Elle est en général faible, dans le but d'échapper aux possibilités de modifications radicales des solutions, particulièrement, des solutions de bonnes qualités qui ne nécessitent que peu d'amélioration pour passer aux solutions optimales.

Le nombre maximum de génération : Ce paramètre peut jouer le rôle d'un critère d'arrêt. Il peut construire un obstacle pour l'algorithme. En fait, il peut empêcher les différents opérateurs d'aboutir à la meilleure solution s'il est trop petit. Comme il peut engendrer un temps de calcul prohibitif dans le cas où il est trop grand. Ainsi, le choix de sa valeur peut se baser sur des tests préliminaires.

L'algorithme génétique a été utilisé pour la résolution des problèmes multi-objectifs. Son avantage principal est qu'il permet une bonne combinaison entre l'exploitation de solutions et l'exploration de l'espace de recherche. Cela est établi en fonction des opérateurs de croisement et de mutation respectivement. Cependant, son inconvénient réside dans deux points : un temps de calcul assez important pour pouvoir converger vers la solution optimale. Et le nombre de paramètres importants (taille de la population, paramètres de sélection, paramètres de croisement, paramètres de mutation, critère d'arrêt…). Pour remédier à ce problème plusieurs chercheurs ont proposé l'algorithme symbiotique génétique.

Symbiose : le terme symbiose décrit la situation où différentes organisations vivent ensemble dans une association étroite, [23]. Et embrasse tous les types de mutualisme.

5.4 Conclusion

Dans ce chapitre, nous avons présenté un certain nombre des méthodes exactes les plus courantes pour résoudre le problème de découpe, qui se caractérisent par une énumération complète de l'espace de recherche et fournissent des solutions exactes. Cependant, dans les applications du monde réel, il existe généralement certaines contraintes qui rendent le problème de découpe,

Qui se caractérisent par une énumération complète de l'espace de recherche et fournissent des solutions exactes. Cependant, dans les applications du monde réel, il existe généralement certaines contraintes qui rendent le problème plus complexe et plus difficile à résoudre. Nous citons parmi ces difficultés, l'extension de modèle de découpe pour inclure de nouveaux coûts et de nouvelles contraintes, entraver l'efficacité de la méthode de résolution. L'augmentation exponentielle de la taille du problème n'a permis aux logiciels standards que résoudre des problèmes de tailles nettement inférieures à celles rencontrées en pratique. Cela fait de la réflexion sur l'utilisation de méthodes heuristiques ou de méthodes hybrides la conception la plus proche de la résolution ce génère du problème.

Chapitre 6

DEUX NOUVELLES TECHNIQUES DE RÈSOLUTION POUR LES CSP UNI –ET BIDIMENSIONNELLE BI-OBJECTIF

6.1 Introduction

La résolution des problèmes de découpe s'est toujours avérée très difficile, voire souvent impossible, compte tenu de la taille du problème qui augmente exponentiellement d'un logiciel standard de programmation linéaire n'a permis que la résolution de petits problèmes de tailles nettement inférieures à celles rencontrées en pratique. Si le problème de découpe à susciter tant d'intérêt pour les chercheurs c'est surtout à cause des difficultés qu'on continue à rencontrer dans sa résolution et ce malgré tous les progrès enregistrés en matière d'algorithmes, de logiciels et d'ordinateur. Il n'y a pas encore une procédure qui domine les autres, il y a plutôt des procédures plus efficaces dans certaines situations, mais moins efficaces dans d'autres.

Dans ce chapitre, nous présentons les techniques que nous avons proposées pour résoudre les problèmes de découpe à deux objectifs avec coût d'installation, que ce soit sous sa forme unidimensionnelle ou bidimensionnelle. Ces méthodes reposent d'abord sur la génération de patrons de découpe par un algorithme heuristique pour des problèmes de découpe unidimensionnelle, qui a été modifié pour générer des patrons de découpe pour les problèmes bidimensionnels.

On sait que l'approche de la programmation linéaire (génération de colonne) qu'été proposé par, Gilmore et Gomory, [33, 34] reste toujours représente l'inconvénient de nécessiter, en pratique, un temps de calcul aberrant. En plus, le modèle qu'essaient de résoudre les méthodes de cette approche risque de ne pas être une présentation fiable de la réalité. Par exemple, cette méthode conçue pour résoudre le modèle de base. Mais l'extension de ce modèle pour inclure d'autres coûts et d'autres contraintes peut entraver l'efficacité de la méthode. Pour remédier à ce problème, nous proposons une heuristique qui se caractérise Par la génération rapide des patrons de découpe, dont l'idée de base est de créer une procédure qui calcul directement le nombre de fois qu'une pièce peut être coupée de l'objet principal.

.6.2 Description de l'approche de résolution du CSP1DB.

Dans la deuxième étape de la solution, nous définissons ce que l'on appelle, plan de découpe, qui est un ensemble de patrons de découpe réalisables, permettant de satisfaire les différents types de demande. Pour y parvenir, nous nous sommes appuyés pour construire des plans de découpe, sur le codage des patrons de découpe puis nous avons créés une méthode arborescente, après avoir modélisé le problème sous forme d'un graphe orienté et qui consiste, à résoudre le problème de découpe unidimensionnelle bi-objectif avec coût d'installation. Dans le cas de problème de découpe bidimensionnelle, nous utilisons une autre stratégie, pour construire les plans de découpe, qui consiste à placer les patrons de découpe, qui ne traite pas la même taille de pièce, dans un même ensemble, les patrons appartenant à différents ensembles constituer des plans de découpe.

Dans les deux paragraphes suivants, nous présentons la méthode de construction des plans de découpes (CCPM), proposée dans cette étude pour résoudre le problème de découpe unidimensionnelle, bi-objectifs. Où nous expliquons les étapes de cette méthode avec une description des algorithmes fournis.

6.2.1 Génération des patrons de découpe réalisables

L'idée de cette heuristique, est qu'après avoir arrangé les longueurs des types de pièces requis par ordre décroissant, de la plus grande longueur à la plus petite, ($w_1 > w_2 > \ldots > w_n$). Nous commençons, à calculer d'abord le nombre de fois, que la longueur du premier type de pièces, peut être découpée de la longueur de l'objet principal, où nous notons par p_{11}, et nous calculons également, le nombre de fois que les longueurs des types de pièces restants, peuvent être coupées du reste de la longueur de l'objet principal et toujours pour ce premier type de pièces, nous diminuons p_{11} par 1, (c'est-à-dire qu'on met $p_{21}= p_{11} -1$) et nous calculons le nombre de fois que les longueurs des types de pièces restants, peuvent être coupées du reste de la longueur de l'objet principal. Où nous continuons à rétrécir p_{11} à chaque fois par 1, jusqu'à ce qu'il n'y ait plus de p_{j1} où $j \geq 1$ et dans chacun nous calculons le nombre de fois que les longueurs des types de pièces restants, peuvent être coupées du reste de la longueur de l'objet principal. Puis, pour passer à la découpe du deuxième type de pièces, nous fixons p_{j1} dans chacune des valeurs obtenues, et pour chaque valeur de p_{j1}, on diminue le nombre de fois.
Que la longueur du deuxième type de pièces ait été coupée de la longueur de l'objet principal, (ce qui veut dire qu'on met p_{j1} dans sa première valeur et nous commençons à abaisser la valeur

de p_{12} à chaque fois par 1, où $j \geq 1$) et en calculant le nombre de fois qui les types de pièces restants peuvent être coupés de la longueur de l'objet principal, de la même manière que la première étape. Ce processus, est répété pour chaque longueur de type de pièce, jusqu'à ce que i = n-1. Pratiquement, on procède comme suit :

1. On calcul $p_{11} = \left\lfloor \frac{W}{w_1} \right\rfloor$ où $\lfloor\ \rfloor$ représente la partie entière inférieure,
2. Pour j = 1 et i variant de 2 à n, $p_{1i} = \left\lfloor \frac{W - \sum_{z=1}^{i-1} p_{1z} \times w_z}{w_i} \right\rfloor$, [71],
3. On diminuer p_{11} par 1 (c'est-à-dire que nous mettons $p_{21} = p_{11} - 1$),
4. Pour j = 2 et i variant de 2 à n, faire $p_{2i} = \left\lfloor \frac{W - \sum_{z=1}^{i-1} p_{2z} \times w_z}{w_i} \right\rfloor$,
5. Continuer à diminuer p_{11} à chaque fois de 1 et à calculer p_{ji} jusqu'à l'annulation de p_{11} où j > 2 et i variant de j à n,
6. L'algorithme est mis à jour en fixant p_{ji} où $j \geq 1$ aux valeurs trouvées précédemment et les mêmes opérations sont répétées avec la longueur du deuxième type de pièces,
7. Répéter le même processus pour chaque longueur de type de pièce jusqu'à ce que i = n-1.

En fin, nous aurons une matrice de taille m × n où chacune de ses lignes est un patron de découpe que nous notons P et nous l'appelons la matrice de découpe.

Algorithme 1. Algorithme de génération des patrons de découpes

Entrée : longueur standard, longueurs des types de pièces,

Sortie : Patrons de découpes réalisables,

1. Arranger les longueurs des pièces demandées, w_i, i = 1, 2, …, n par ordre décroissant, c'est à dire. ($w_1 > w_2 > \ldots > w_n$), où n est le nombre de type de pièces demandées,
2. Calculer $p_{11} = \left\lfloor \frac{W}{w_1} \right\rfloor$, poser j = 1, pour i = 2 to à n faire $P_{1i} = \left\lfloor \frac{W - \sum_{z=1}^{i-1} p_{1z} \times w_z}{w_i} \right\rfloor$,
3. Poser i = 1, h = 1, k = 1,
   a) Poser j = 1, d = 1,
   b) Si $p_{ij} > 0$ alors j = j + 1, l = 1 et aller à (c) sinon aller à 4,
   c) h = h + 1,
   d) Si i = 1 alors $p_{hi} = p_{j-1,i} - 1$,

      Pour i := k + 1 à n faire $p_{hi} = \left\lfloor \frac{W - \sum_{z=1}^{i-1} p_{hz} \times w_z}{w_i} \right\rfloor$,
   e) Sinon pour z := 1 à i-1 faire $p_{hz} = p_{j-1,z}$,

      Pour i := k à $p_{hi} = p_{j-1,i} - 1$,

      Pour i: = k + 1 à n faire $p_{hi} = \left\lfloor \frac{W - \sum_{z=1}^{i-1} p_{hz} \times w_z}{w_i} \right\rfloor$,
   f) Si $p_{hi} > 0$ alors poser l = l + 1, d = d + 1 et aller à (c) sinon aller à (g),
   g) d = d + 1,
   h) Si $d < m$,(où m le nombre de patrons de découpe jusqu'à l'étape en cours), alors aller à (b) sinon si $i < n - 1$ alors poser i = i + 1, k = k + 1 et aller à (a), sinon stop,
4. Poser j = j + 1, d = d + 1 et aller à (b).

6.2.2 Stratégie de construction des plans de découpe

Dans cette section, nous allons montrer comment construire des plans de découpe et pour justifier cela nous présentons la proposition suivante.

Proposition 6.1 : Soit P une m×n-matrice booléenne, w un n-vecteur colonne à coordonnées entières strictement positives et V un m-vecteur colonne donnée par P × w. Soient r, i, j, k et s des entiers positives, $P'_s = P_i + P_j + \cdots + P_k$ une somme de vecteurs lignes dans la matrice P (respectivement P_r un vecteur ligne dans la matrice P) et $v'_s = v_i + v_j + \cdots + v_k$, ($\forall\ i \neq j \neq k$) une somme de composantes dans le vecteur V correspondante au vecteur P'_s (respectivement v_r une composante dans le vecteur V correspondante au vecteur P_r), on a, la proposition suivante: $v'_s \geq \sum_{r=1}^{n} w_r \Leftrightarrow P'_s \geq (1, \dots, 1)$, [66].

Preuve :

(i) Si $v'_s \geq \sum_{r=1}^{n} w_r$ alors $P'_s \geq (1, \dots, 1)$, par contraposée on a : si $P'_s < (1, \dots, 1)$ alors $v'_s < \sum_{r=1}^{n} w_r$. On multiplie à droite l'inégalité $P'_s < (1, \dots, 1$ par le vecteur colonne w, donc : $P'_s w < (1, \dots, 1)w \Leftrightarrow P'_s w < (w_1 + w_2 + \cdots + w_n) \Leftrightarrow v'_s < \sum_{r=1}^{n} w_r$.

(ii) Si $P'_s \geq (1, \dots, 1)$, alors $v'_s \geq \sum_{r=1}^{n} w_r$. On a: $P'_s \geq (1, \dots, 1) \Leftrightarrow P'_{s1} \geq 1, P'_{s2} \geq 1, \dots, P'_{sn} \geq 1$. On multiplie à droite chaque inégalité de $P'_{si} \geq 1$ par w_i où $w_i \in \mathbb{N}^*$ et $1 \leq i \leq n$, d'où : $P'_{s1}w_1 \geq w_1, \dots, P'_{sn}w_n \geq w_n \Leftrightarrow P'_{s1}w_1 +, \dots, + P'_{sn} w_n \geq w_1 +, \dots, + w_n \Leftrightarrow P'_s w \geq \sum_{r=1}^{n} w_r \Leftrightarrow v'_s \geq \sum_{r=1}^{n} w_r$.

Le processus de construction des plans de découpe dépend principalement des patrons de découpe qui ont été générés par l'algorithme précédent, car chaque sous-ensemble de ces patrons satisfaisant les demandes est considéré comme un plan de découpe et pour former ces plans on code chaque patron de découpe réalisable et pour y parvenir, on affecte le numéro 1 à chaque composante p_{ij} différente de 0, et on affecte le numéro 0 à chaque composante p_{ij} égale à 0. On forme donc un patron de découpe codé, chacune de ses valeurs est soit 0 soit 1. Ainsi, une matrice codée est formée notée P'. Où on multiplie cette matrice par le vecteur de tailles des pièces requises w_i, le vecteur colonne résultant est noté $V = \sum_{r=1}^{m} \sum_{j=1}^{n} p_{rj} w_j$ et selon la proposition précédente, on a pour chaque somme des composantes $v'_s = v_i + v_j + \cdots + v_k$, supérieur ou égale à $\sum_{r=1}^{n} w_r$ et $P'_s = P_i + P_j + \cdots + P_k \geq (1, \dots, 1)$, l'ensemble des patrons de

découpe P_i, P_j, P_k, corresponds aux ces composantes, forme un plan de découpe (qu'est noté pd_i) et ainsi, s'il existe dans le vecteur V, une composante v_r, supérieur ou égale à $\sum_{r=1}^{n} w_r$, et $P_r \geq (1, \dots, 1)$ alors le patron de découpe P_r, correspond à cette composante forme un plan de découpe et par conséquent chaque plan de découpe de plus petite perte et un nombre d'installations fixe, est une solution efficace.

6.2.3 Résolution de problème

La résolution de problèmes passe par deux étapes, l'étape de modélisation et l'étape de résolution.

Étape 1. À ce stade, nous modélisons comment calculer les sommes des composantes du vecteur V, c'est-à-dire que nous modélisons v_s', où $v_s' = v_i + v_j + \cdots + v_k$, par un graphe G = (X, U, C), où X : l'ensemble des sommets, U: l'ensemble des arcs, C: l'évaluation des arcs, dans laquelle, chaque composante du vecteur V représente par un sommet x de G et chaque somme de composantes $x_i + x_j$ décrit par un arc u relie le sommet x_i par le sommet x_j, on ajoute aussi un sommet source r joignant les sommets x de G qui sont situés devant le sommet source r.

Ensuite, nous poursuivons le processus de modélisation comme suit :

1. Pour j = 1 et i variant de 1 à m, on pose $x_{ij} = x_i$,
2. Pour j variant de 1 à m-2 et i variant de m - j + 1 à 2 on pose $x_{i-1,j+1} = x_{ij}$,
3. j variant de 1 à m-1 et i variant de 1 à m - j on relie les sommets x_{ij} par les sommets $x_{i+1,j}$,
4. Pour j variant de 1 à m-2, i variant de 1 à m - j – 1 et z variant de i + 1 à m on relie les sommets x_{ij} par les sommets $x_{z,j+1}$.

Étape 2. Dans cette étape, nous cherchons la solution de problème, qui consiste à trouver l'ensemble des solutions efficaces. Dans le graphe qui s'est formé à partir de la phase de modélisation, on commence d'abord à initialiser $E = \{r\}$, $E_{1j} = \emptyset$, (j = 1, 2…m) et pour chaque arc u à une extrémité initiale le sommet r, on fixe $c(I(u)) = 0$. Nous déterminons ensuite, les sommets x_{ij} dans X - E dont les prédécesseurs sont dans E, et on pose $E_{ij} = E_{ij} \cup \{x_{ij}\}$ et on calcule $S(T(u)) = c(I(u)) + c(u)$. Puis on fait le test, si $S(T(u)) \geq \sum_{r=1}^{n} w_r$ et $P_s' = P_i' + P_j' + \cdots + P_k' \geq (1, \dots, 1)$, où P_i', P_j', P_k' les patrons de découpe correspondent à v_i, v_j, v_k. Alors le plan de découpe pd_i correspond à $c(I(u)) = S(T(u))$ est une solution réalisable. Dans l'étape

suivante, on pose $c(I(u)) = S(T(u))$ pour les arcs qui sont adjacents au même sommet déjà défini comme prédécesseur. Et également, nous déterminons les sommets x_{ij} dans X - E dont les prédécesseurs sont en E et pour continuer les calculs, nous répétons les mêmes étapes précédentes. Où la procédure de calcul se poursuit de la même manière jusqu'à ce que nous nous rapprochions ou que nous atteignions la valeur théorique $\sum_{j=1}^{T} x_j = \left\lceil \frac{\sum_{i=1}^{n} w_i d_i}{W} \right\rceil$, où $\lceil\ \rceil$ la partie entière supérieure.

Algorithme 2. Algorithme de construction du plan de découpe

Entrée : patrons de découpe réalisables,

Résultat : des solutions efficaces,

1. Appliquer l'algorithme 1,
2. Calculer la valeur théorique : $\sum_{j=1}^{T} X_j = \left\lceil \frac{\sum_{i=1}^{n} w_i \times d_i}{W} \right\rceil$,
   a) S'il y a des patrons de découpe P_r où $P_r \geq (1, ..., 1)$, puis déterminer les plans de découpe correspondants pd_i et aller en (b), sinon aller en (3),
   b) Calculer $x_j = Max_i \left(\left\lceil \frac{d_i}{p_{ij}} \right\rceil \right)$, où $\left\lceil \frac{d_i}{p_{ij}} \right\rceil$, la partie entière supérieure de $\frac{d_i}{p_{ij}}$ et $P_{ij} \neq 0$,
   c) Calculer la perte totale de découpe $pt_i = W \times \sum_{j=1}^{T} x_j - \sum_{r=1}^{n} w_r d_r$,
   d) Calculer Pt = Min (pt_i) et aller en (3), i = 1... p (où p le nombre de plans de découpe obtenus),
3. Coder les patrons de découpe réalisables P_i comme suit : $P'_{ij} = \begin{cases} 1 \text{ si } p_{ij} \neq 0 \\ 0 \text{ si } p_{ij} = 0 \end{cases}$
4. Calculer $V = \sum_{r=1}^{m} \sum_{j=1}^{n} p'_{rj} w_j$,
5. Poser j = 1, pour i := 1 à m faire $x_{ij} = x_i$,
6. Pour j := 1 à m-2 faire
   Pour i : = m - j + 1 à 2 faire $x_{i-1,j+1} = x_{ij}$,
7. Pour j := 1 à m-1 faire,
   Pour i : = 1 à m - j relier le sommet x_{ij} par le sommet $x_{i+1,j}$,
8. Pour j := 1 à m - 2 faire
   Pour i := 1 à m - j - 1 faire
   Pour z : = i + 1 à m - j rejoindre x_{ij} par $x_{z,j+1}$,
9. Poser E = {r}, $E_{1j} = \emptyset$, j = 1, 2…, m et c(I(u)) = 0 pour chaque arc u à une extrémité initiale le sommet r, et aller en (10),

10. Déterminer les sommets x_{ij} dans X - E dont les prédécesseurs sont dans E et poser $E_{ij} = E_{ij} \cup \{x_{ij}\}$,
11. Calculer $S(T(u)) = c(I(u)) + c(u)$, où u un arc d'extrémité terminal (T) est le Sommet x_{ij},
    a) Si $S(T(u)) \geq \sum_{r=1}^{n} w_r$ alors déterminer les patrons de découpe correspondant aux sommets $S(T(u))$ et aller en (b), sinon aller en (g),
    b) Si $P_s' = P_i' + P_j' + \cdots + P_k' \geq (1, \ldots, 1)$, alors déterminer le plan de découpe pd_i correspondant aux patrons décodés P_i, P_j, P_k, et aller à l'étape (c), (d), (e) et (f) sinon aller en (g),
    c) Calculer $b_{ij} = \sum_{i \geq 1} p_{ij}$, $x_j = Max_i \left(\left\lceil \frac{d_i}{b_{ij}} \right\rceil \right)$ où $p_{ij} \neq 0$, et calculer $x_{pd_i} = \sum_{j=1}^{T} x_j$,
    d) S'il existe des plans de découpe de même nombre d'installations et de même x_{pd_i}, éliminer les plans redondants et aller en (e), sinon aller en (e),
    e) Calculer la perte de découpe : $pt_i = W \times \sum_{j=1}^{T} x_j - \sum_{r=1}^{n} w_r d_r$,
    f) Calculer pt = Min (pt_i), i = 1... p (où p le nombre de plans de découpe obtenus),
    g) Poser $c(I(u)) = S(T(u))$ pour les arcs adjacents au même sommet déjà défini comme prédécesseur,
    h) Si le nombre d'arcs entrant en un T(u) est supérieur à 1 alors éclater cette extrémité au nombre d'arcs entrant en T(u), alors soit $c(I(u')) = S(T(u'))$, $c(I(u'')) = S(T(u''))$, …, et aller en (i), sinon aller en (i),
    i) Répéter $E = E \cup \{x_{ij}\}$, et aller en (10) jusqu'à atteindre ou s'approcher à la valeur théorique $\sum_{j=1}^{T} X_j = \left\lceil \frac{\sum_{i=1}^{n} w_i \times d_i}{W} \right\rceil$,

6.2.4 Finitude des algorithmes

On vérifie dans ce paragraphe que le nombre d'itérations est fini et que l'algorithme ne boucle pas :

(i) Algorithme de génération de patron de découpe : L'idée dans cet algorithme consiste à calculer la fréquence p_{11} de la première taille de pièce w_1 sur l'objet principal W et à chaque fois diminué cette fréquence de 1 jusqu'à l'annulation de p_{11} et calculer les autres fréquences par $p_{1i} = \left\lfloor \frac{W - \sum_{z=1}^{i-1} p_{1z} \times w_z}{w_i} \right\rfloor$, l'algorithme réitéré pour chaque pièce jusqu'à i = n-

1. En effet une pièce considérée n'est pas revisitée une seconde fois, donc l'algorithme ne boucle pas et comme le nombre de pièces est fini alors le nombre d'itérations est fini.

(ii) Construction des plans de découpe : dans cette étape l'algorithme consiste à additionner les évaluations des arcs de tous les chemins élémentaires possibles dans le graphe G = (X, U, C), proches à proches, du sommet source à où on ne peut pas développer la solution, en fait, seul un sommet considéré n'est visité qu'une seule fois, et comme G est sans circuit, donc on ne boucle pas dans l'algorithme, et comme G avec un nombre fini de sommets et d'arcs alors le nombre d'itérations est fini.

6.2.5 Résultats et discussion

Dans cette section, nous présentons une étude empirique et analytique, où nous discutons les résultats obtenus par notre méthode proposée CCPM [66] et les comparons avec certains résultats trouvés dans la littérature.

Dans la première partie, nous le comparons d'abord avec une technique de résolution d'un problème de découpe unidimensionnelle mono-objectif, Où nous choisissons pour cette comparaison l'étude réalisée par Pazand et Mohammadi [63] qui est une extension de l'algorithme qui a été proposé par Hessler [38] pour résoudre le problème de découpe unidimensionnelle avec coût d'installations, (nous l'avons noté par EHH), où ils ont traité des exemples réels dans l'industrie cinématographique par trois méthodes différentes :

- Méthode de programmation linéaire en nombres entiers de Gilmore et Gomory (ILP),
- Méthode heuristique séquentielle de Haessler (SHH),
- Méthode hybride de Pazand et Mohammadi (EHH).

Où, nous présentons les données du premier exemple dans le tableau 6.1, pour les résultats obtenus par les trois méthodes dans le tableau 6.2, et l'ensemble des solutions efficaces obtenues à l'aide de CCPM sont présentés dans le tableau 6.3. Quant au deuxième exemple, nous présentons les données dans le tableau 6.4. En ce qui concerne les résultats obtenus par les trois méthodes indiquées dans le tableau 6.5 et l'ensemble des solutions efficaces obtenues à l'aide de CCPM sont présentés dans le tableau 6.6.

Dans la deuxième partie de cette étude, nous comparerons la technique CCPM à celle de l'algorithme génétique proposé par Araujo et al. [2] pour résoudre un problème de découpe unidimensionnelle bi-objectif avec coût d'installation, où nous sélectionnons 30 instances sur

40 disponibles dans Umetani et al. [75], tandis que ces exemples sont tirés d'une entreprise de fibres chimiques au Japon. Selon les données suivantes :

- Le nombre de types de pièces requises : n varie de 4 à 20,
- La longueur de l'objet principale : W = 5180 pour les 15 premières instances et W = 9080 pour les 15 instances restantes,
- Longueur de pièces requises w_i : ont été générés aléatoirement dans l'intervalle [500, 2000],
- Demande : d_i ont été générés aléatoirement dans l'intervalle [2, 264].

Les résultats obtenus par la méthode proposée par Araujo et al. [2] nommé MOGA et notre méthode nommée CCPM [66], sont présentés dans le tableau 6.8 pour la longueur d'objet W = 5180 et le tableau 6.9 pour la longueur d'objet W = 9080. Notons enfin que nous prenant les notations suivantes :

- F_{solu} : les solutions réalisables
- $N°_{effs}$: les solutions efficaces
- $P_{\text{per découpe}}$: le pourcentage (%) de perte de matière première
- $P_{\text{surf perd}}$: le pourcentage (%) de surface perdue
- $N_{installs}$: le nombre d'installations (Setup)
- x_j : le nombre de fois où le modèle j est utilisé
- T_v : la valeur théorique : $\sum_{j=1}^{T} X_j = \left\lceil \frac{\sum_{i=1}^{n} s_i \times d_i}{S} \right\rceil$

n =13		W = 6480 mm			
i	w_i (mm)	d_i (mm)	i	w_i (mm)	d_i (mm)
1	1200	14	8	720	62
2	1100	8	9	680	24
3	1020	22	10	600	72
4	960	14	11	560	8
5	900	8	12	510	100
6	850	8	13	400	40
7	760				

Tableau 6.1 : Longueurs de pièces requises et demandes

Les résultats obtenus par ILP, SHH, EHH sont présentés dans le tableau suivant :

Méthode	$P_{\text{per découpe}}$	N_{installs}
ILP	0	1
SHH	18.4	1
EHH	5.3	6

Tableau 6.2 : Résultats obtenus par ILP, SHH, EHH

$N°_{\text{effs}}$	$P_{\text{per découpe}}$	N_{installs}
1	2.13	13
2	2.86	12
3	3.67	11
4	4.52	10
5	4.61	9
6	5.24	8
7	6.73	7
8	7.31	6
9	8.46	5

Tableau 6.3 : Résultat obtenu par CCPM

Le tableau 6.3 représente un vecteur de solutions efficaces pour une instance de taille n = 13, pour différentes longueurs de type de pièce et différentes tailles de commande. La précision des résultats obtenus, varie de 8,46 % de perte totale de matière première et 5 installations à 2,13 % de perte totale de matière première et 13 installations, avec un temps d'exécution estimé à moins d'une minute. Cependant, par rapport aux résultats du tableau 6.2, il existe des techniques utilisées qui ont donné un meilleur résultat.

	n =13	W = 6480 mm			
i	w_i (mm)	d_i (mm)	i	w_i (mm)	d_i (mm)
1	510	54	8	12	950
2	600	18	9	32	1020
3	680	61	10	30	1100
4	720	17	11	22	1140
5	730	33	12	32	1200
6	760	14	13	54	1356
7	900	12			

Tableau 6.4 : Longueurs de découpe requises et demandes

Les résultats obtenus par ILP, SHH, EHH sont présentés dans le tableau suivant :

Méthode	$P_{\text{per découpe}}$	N_{installs}
ILP	0.36	13 (one of them infeasible)
SHH	6.62	8
EHH	9.98	5

Tableau 6.5 : Résultats obtenus par ILP, SHH, EHH.

Les résultats obtenus par CCPM sont présentés dans le tableau suivant :

$N°_{\text{effs}}$	$P_{\text{per découpe}}$	N_{installs}
1	1.51	13
2	1.90	12
3	2.07	11
4	3.61	10
5	3.89	9
6	4.02	8
7	4.21	7
8	4.72	6
9	5.01	5
10	5.20	4

Tableau 6.6 : Résultat obtenu par le CCPM

Les résultats obtenus par CCPM dans cet exemple représentent une gamme de solutions efficaces allant de 5,20 % de perte de matière première avec 4 nombres d'installations à 1,51 % de perte totale de matière première avec 13 nombres d'installations. La comparaison de ces résultats avec ceux donnés dans le tableau 6.5 montre que CCPM domine EHH, car cette dernière fournit une solution optimale de 9,98% de matières premières gaspillées avec 5 nombres d'installations, tandis que CCPM dans 5 installations, la perte totale de matière première est 5,01%. Et il en va de même pour SHH, où CCPM fournit une solution efficace de 4,02 % de matières premières gaspillées avec 8 installations, alors que SHH fournit une solution optimale de 6,62 % de matières premières gaspillées avec le même nombre d'installations. Quant à ILP, il était meilleur que CCPM car il fournit une solution à 0,36 % de matières premières gaspillées avec 13 installations, alors que CCPM avec le même nombre d'installations a un gaspillage 1,51 %.

La comparaison des résultats obtenus par CCPM avec ceux obtenus par MOGA ils sont présentés dans les tableaux 6.7 et 6.8 suivants :

Setup	MOGA	CCPM
Instance Fiber06 5180		
7		1.413
6	2.09	1.829
5	8.27	2.557
4		3.365
3		5.632
Instance Fiber 07 5180		
10		1.263
9		1.784
8		2.152
7		2.836
6		3.062
5		3.475
4	4.98	4.264
3	8.16	6.113
Instance Fiber08 5180		
10		2.045
9		2.468
8		2.731
7		4.012
6		4.472
5	3.26	5.469
4	4.46	7.582
3	6.86	9.381
2		10.563
Instance Fiber09 5180		
9		4.231
8		7.756
7	9.36	9.067
6		9.978
5	11.28	10.043
4		10.648

Setup	MOGA	CCPM
Instance Fiber11 5180		
12		1.575
11		2.123
10		3.857
9		4.379
8		4.937
7	2.98	5.075
6	9.13	6.654
5	12.20	7.475
4	14.67	8.536
Instance Fiber13a 5180		
14		1.854
13		2.631
12		2.753
11		3.214
10		3.879
9	4.24	5.625
8	6.06	7.541
7	10.05	10.638
6		11.057
5		12.631
4		13.542
3		16.564
2		21.076
Instance Fiber13b 5180		
9		1.112
8		1.921
7		2.027
6	2.92	2.671
5	10.26	3.758
4		6.531

Setup	MOGA	CCPM
Instance Fiber19 5180		
25		0.879
24		1.065
23		1.211
22		1.645
21		1.936
20		2.043
19		2.227
18		2.566
17		2.876
16		3.210
15		3.346
14		3.951
13		4.323
12		4.774
11		5.120
10	5.77	6.431
9	8.92	9.057
8		10.546
7		13.472
6		22.551
5		29.475
4		36.328
3		41.547
Instance Fiber28b 5180		
22	2.86	1.751
21	4.49	3.094
20	7.76	3.762
19	28.90	4.105
18		4.845
17		5.312

3		11.054
2		11.672
Instance Fiber10 5180		
14		1.543
13		1.879
1		2.432
11		2.874
10		3.061
9		3.453
8		3.976
7		4.120
6	4.20	4.206
5	7.17	7.132
4	47.31	10.056
3		15.216
2		20.542
Instance Fiber14 5180		
11		1.054
10		1.768
9		3.546
8		4.672
7		6.453
6	5.45	6.872
5		8.590
4		9.441
3		11.632
2		15.211

3		8.042
2		11.463
Instance Fiber 18 5180		
14	2.97	2.113
13	4.03	3.734
12	47.56	5.316
11		7.472
10		8.521
9		10.346
8		10.965
7		11.678
6		14.054
5		14.732
4		17.125
3		17.543
Instance Fiber20 5180		
16	3.96	2.075
15	7.11	3.326
14	32.31	5.647
13	99.81	5.874
12		6.366
11		8.529
10		10.321
9		11.654
8		13.426
7		16.563
5		17.454
4		19.677

16		5.982
15		6.836
1		7.128
13		7.471
12		8.326
11		8.852
10		9.120
9		10.541
8		11.043
7		11.453
6		13.683
5		14.543
Instance Fiber17 5180		
10	3.01	2.322
9	5.30	4.231
8	9.63	8.766
7	44.62	9.775
6		10.111
5		13.012
4		15.636
Instance Fiber23 5180		
16	2.83	1.021
15	4.26	2.349
14		2.744
13		4.272
12		5.763
11		7.264
10		8.063
9		10.211
8		10.782
7		12.134
6		14.652
5		16.212
4		19.580

Tableau 6.7 : Installations et pourcentage de perte obtenus par CCPM et MOGA (W = 5180)

Setup	MOGA	CCPM	Setup	MOGA	CCPM	Setup	MOGA	CCPM
Instance Fiber07 9080			Instance Fiber15, 9080			Instance Fiber28b, 9080		
4	5.95	1.763	6	1.31	1.765	19	2.42	2.202
3		4.262	5	7.64	3.878	18	5.23	3.743
2	11.52	5.647	4	13.97	4.555	17	23.50	5.372
1		7.551	3		6.653	16	32.54	6.865
Instance Fiber08 9080			2		8.677	15		7.428
			1		10.543	14		7.881
4	1.03	1.121	Instance Fiber16, 9080			13		8.455
3	3.13	2.054	13	2.95	1.574	12		10.463
2	19.97	5.533	12	7.24	2.745	11		11.054
1		10.324	11	93.04	3.231	10		11.658
Instance Fiber09 9080			10		5.162	9		13.124
5	2.86	1.768	9		5.874	8		13.461
4		2.976	8		6.213	7		15.547
3		4.421	7		7.532	6		16.934
2		5.043	6		9.041	5		18.725
1		7.665	5		9. 543	Instance Fiber26, 9080		
Instance Fiber10 9080			4		11.112	15	1.00	1.074
5	1.76	1.421	3		15.653	14		1.762
4	12.20	2.443	2		20.434	13	1.94	2.875
3		3.775	Instance Fiber17, 9080			12	17.89	3.764
2		5.543	12		2.041	11		5.547
1		8.654	11	3.18	2.767	10		9.056
Instance Fiber11 9080			10		4.422	9		10.423
5	2.38	1.057	9	16.07	6.321	8		13.522
4	5.07	2.648	8	71.96	9.032	7		14.053
3	11.90	2.975	7		9.544	6		14.346
2		4.073	6		11.999	5		16.871
1		7.751	5		12.323	4		18.346
Instance Fiber13a 9080			4		14.655	Instance Fiber29, 9080		
6	2.25	1.843	3		17.543	13	3.03	2.054
5	44.25	3.612	2		20.765	12		3.021

4		4.221
3		5.475
2		7.452
1		10.761
Instance Fiber13b 9080		
5	3.08	2.371
4	28.85	3.542
3		7.683
2		12.631
1		38.654
Instance Fiber14b 9080		
7	5.62	1.852
6	16.94	3.237
5		3.875
4		4.534
3		6.352
2		5.471
Instance Fiber18, 9080		
9	2.34	1.058
8	4.20	2.673
7	6.06	4.012
6	6.86	4.831
5		5.063
4		6.561
3		7.842
11	5.89	3.213
10		4.890
9		6.743
8		7.047
7		7.852
6		9.362
5		10.541
4		11.752
3		16.445
2		25.322

Tableau 6.8: Installation et pourcentage de perte obtenus par CCPM et MOGA, (W = 9080)

A la lecture des résultats présentés dans le tableau 6.7, nous constatons que notre méthode proposée a fourni des solutions efficaces pour chacun des instances étudiées, de sorte que chaque solution est caractérisée par un nombre d'installations et une quantité de matières premières perdues dans un temps d'exécution raisonnable. Où il ne dépasse pas une minute dans la plupart des instances.

En comparant ces résultats avec ceux obtenus par MOGA, où l'on note que CCPM est dominant dans la plupart des instances, par exemple, dans les instances : Fibre 06, Fibre 07, Fibre 09, Fibre 10, Fibre 13b, Fibre 17, Fibre 28, et autres. CCPM domine MOGA, tandis que cette dernière domine CCPM dans les instances : Fibre 08, Fibre 13a et Fibre 19.

Dans le tableau 6.8, la longueur de l'objet principal a été augmentée alors que les longueurs des types de pièces requises sont restées les mêmes, nous avons observé que le CCPM est resté

stable pour ce changement, ce qui signifie qu'il a fourni des solutions pour toutes les instances étudiées avec la même efficacité. Quant à la comparaison des résultats avec ceux obtenus par MOGA, ils sont similaires à ceux vus dans le tableau 6.7, où dans de nombreuses instances CCPM domine encore MOGA.

Enfin, à la suite de cette étude pilote, nous pouvons dire que nous avons observé que le CCPM est une méthode compétitive car elle fournit un ensemble de solutions qui sont généralement meilleures que les méthodes monocritères. De plus, elle domine souvent la méthode bi-objective MOGA.

6.3 Description de l'approche de résolution du CSP2DB

Dans cette section, nous présenterons la méthode de construction des plans de découpes proposée pour résoudre les problèmes de découpe bidimensionnelle bi-objectif avec cout d'installation (CPBM).

Où nous abordons d'abord les modifications qui ont été apportés à la méthode de génération des patrons de découpe qui ont été développés dans le cas du problème de découpe unidimensionnelle, nous discutons dans la suite la stratégie de construction des plans de découpe dans le cas du problème de découpe à deux dimensions à deux objectifs, nous présentons également une étude numérique dans laquelle nous mettons en évidence les résultats obtenus à partir de notre méthode proposée et donnons les résultats de l'étude comparative qui a été réalisée avec d'autres méthodes de la littérature.

6.3.1 Technique modifiée de génération des patrons de découpe réalisables

L'idée de l'heuristique de génération des patrons de découpes est consistée à générer une matrice de découpe (notée P de taille m × n), dans laquelle chaque ligne de P représente un patron de découpe. Pour adapter cette heuristique afin de générer des patrons de découpes pour le problème bidimensionnel nous apportons les modifications suivantes :

- Nous calculons d'abord l'aire totale de l'objet principale que nous désignons par S = L × W où L et W la langueur et la largeur de l'objet principal respectivement,
- Calculons les l'aires des pièces requises que nous désignons par $s_i = l_i \times w_i$,
- Arrangeons les aires s_i par ordre décroissant ($s_1 \geq s_2 \geq \ldots \geq s_n$),
- Appliquons la restriction non guillotine pour découper les aires s_i de l'objet principal.
- En se référant aux étapes visées au paragraphe 6.2.1 et à l'algorithme 1, dérivons l'algorithme suivant :

Algorithme 3. Algorithme de génération des patrons de découpe réalisables modifie

Entrée : l'aire standard, l'aire des types de pièces,

Sortie : Patrons de découpes réalisables,

1. Déterminer la première ligne de la matrice P par :

a) Calculer $p_{11} = \left\lfloor \frac{S}{s_1} \right\rfloor$,

b) Poser j = 1, pour i = 2 to à n faire $p_{1i} = \left\lfloor \frac{S - \sum_{z=1}^{i-1} p_{1z} \times s_z}{s_i} \right\rfloor$,

2. Poser i = 1, h = 1, k = 1

i) Poser j = 1, d = 1,

j) Si $p_{ji} > 0$ alors j = j + 1, l = 1,

k) h = h + 1,

l) Si i = 1 alors $p_{hi} = p_{j-1,i} - 1$,

Pour i := k + 1 à n faire $p_{hi} = \left\lfloor \frac{S - \sum_{z=1}^{i-1} p_{hz} \times s_z}{s_i} \right\rfloor$,

Sinon pour z := 1 à i-1 faire $p_{hz} = p_{j-1,z}$,

Pour i := k à $p_{hi} = p_{j-1,i} - 1$,

Pour i: = k + 1 à n faire $p_{hi} = \left\lfloor \frac{S - \sum_{z=1}^{i-1} p_{hz} \times s_z}{s_i} \right\rfloor$,

m) Si $p_{hi} > 0$ alors poser l = l + 1, d = d + 1 et aller à (c) sinon aller à (g),

n) d = d + 1,

o) Si $d < m$, (où m le nombre de patrons de découpe jusqu'à l'étape en cours), alors aller à (b) sinon si $i < n - 1$ alors poser i = i + 1, k = k + 1 et aller à (a), sinon stop,

3. Sinon poser j = j + 1, d = d + 1 et aller à (b).

6.3.2 Stratégie de construction des plans de découpe

La construction de plans de découpe basée sur la génération de tous les patrons de découpe réalisables qui permet de construire des plans de découpe, satisfaisant les demandes, grâce à un sous-ensemble de ces patrons. Où cette stratégie est constituée de deux étapes :

La première consiste à regrouper tous les patrons de découpe qui ne découpe pas le premier type de pièces et on le met dans un ensemble B_1, (pratiquement on cherche toutes les lignes qui ont des zéros dans la même colonne de la matrice P et on le mettre dans un ensemble B_1), puis on cherche tous les patrons de découpe qui ne coupe pas le deuxième type de pièces et on le met dans un ensemble B_2, cette procédure est poursuivie jusqu'à la recherche de tous les patrons de découpe qui ne traite pas le $n^{ième}$ type de pièces et on le met dans un ensemble B_n. Enfin, nous avons un certain nombre d'ensembles, où les éléments de chaque ensemble sont des patrons de découpes qui ne traitent pas le même type de pièces.

Tandis que la seconde se justifie par la proposition suivante :

Proposition 6.2 : Soit $\Omega = B_1 \cup B_2 \cup ... \cup B_n$ un sous-ensemble de $\mathbb{N}$ et $\{P_i\}$ inclus dans au moins un sous-ensemble B_k où $k \geq 1$, s'il existe au moins un sous-ensemble $B_d \subset \overline{B_i \cup B_j \cup ... \cup B_h}$ (complément de $B_i \cup B_j \cup \ldots \cup B_h$) alors $\{P_i\} \cap B_d = \emptyset$, [67].

Preuve : par contraposée, nous avons $\{P_i\} \cap B_d \neq \emptyset \Rightarrow \forall B_d \subset \Omega, B_d \not\subset \overline{B_i \cup B_j \cup ... \cup B_h}$, Supposons que nous avons : $\{P_i\}$ inclus dans au moins un sous-ensemble $B_k \Rightarrow \{P_i\} \subset B_k \vee \{P_i\} \subset B_k \wedge B_j \vee \{P_i\} \subset B_k \wedge B_j \ldots \wedge B_h \ldots \vee \{P_i\} \subset B_i \wedge B_j \ldots \wedge B_h \ldots \vee \{P_i\} \subset B_k \wedge B_j \ldots \wedge B_h \ldots \wedge B_n$ tellement comme $\{P_i\} \subset B_i \wedge B_j \ldots \wedge B_h$ alors $\{P_i\} \subset B_i \cap B_j \ldots \cap B_h$ et un autre par $\{P_i\} \cap B_d \neq \emptyset \Rightarrow \{P_i\} \subset B_d \Rightarrow \{P_i\} \subset B_d \cap B_i \cap B_j \ldots \cap B_h \Rightarrow B_d \subset B_i \cup B_j \ldots \cup B_h \Rightarrow B_d \not\subset \overline{B_i \cup B_j \cup ... \cup B_h}$.

Selon cette proposition, nous procédons comme suit :

1. On arrange les éléments des ensembles B_k dans l'ordre non décroissant P_1 à P_m où m le nombre des éléments de tous les ensembles puis pour i variant de 1 à m-1 on détermine P_i appartient aux ensembles B_k, où k = 1, 2 …n et les sous-ensembles de (l -1) éléments dans lesquels P_i et les (l -1) éléments n'appartiennent pas au même ensemble, ainsi que P_i et les (l -2) éléments ne forment pas un plan de découpe dans l'étape précédente où l = 2, 3 ...m.

2. On ajoute aux meilleurs plans de découpe sur les deux fonctions objectifs de l'étape précédente un patron de découpe pour minimiser les différents types de demande. La construction les plans de découpe est continue, jusqu'à ce que soit $\sum_{j=1}^{T} X_j$ attient la valeur théorique $\sum_{j=1}^{T} X_j = \left\lceil \frac{\sum_{i=1}^{n} s_i \times d_i}{S} \right\rceil$ ou plus proche de cette valeur.

Algorithme 4. Algorithme de stratégie de construction des plans de découpe

Entrée : Patrons de découpes réalisables

Sortie : Plans de découpes réalisables,

1. Calculer la valeur théorique : $\sum_{j=1}^{T} X_j = \left\lceil \frac{\sum_{i=1}^{n} s_i \times d_i}{S} \right\rceil$, où $\lceil\ \rceil$ est la partie entière supérieure,
2. S'il existe une ligne $P_i \in P$, ne contient pas des zéros, alors déterminer le plan de découpe $pd_k = P_i$, où i, k sont des entiers positifs et aller en (a), sinon aller en (3),
   a) Calculer $x_j = Max_i \left(\left\lceil \frac{d_i}{P_{ij}} \right\rceil \right)$, où $P_{ij} \neq 0$,
   b) Calculer l'aire totale perdue : $pt_i = S \times x_j - \sum_{r=1}^{n} s_r\, d_r$,
   c) Calculer $Pt = Max_i\ (Pt_i)$, i = 1... au nombre de plans de découpe
3. Pour j = 1 à n faire
   Pour i = 1 à m faire
   a) Si $P_{ij} = 0$, alors $A_{ij} = \{P_i\}$, sinon $A_{ij} = Ø$,
   b) Pour i = 1 à m faire $B_j = \bigcup_{i=1}^{m} A_{ij}$,
4. Arranger les patrons de découpe dans un ordre non décroissant P_1 à P_m,
5. Poser l = 2, t =1,
   a) S'il existe des plans de découpe avec l - 1 nombre d'installations, alors ajouter à chaque plan de découpe à l - 1 nombre d'installations, un patron de découpe de manière à minimiser les valeurs les plus grandes de $\left\lceil \frac{d_i}{b_{ij}} \right\rceil$ où $b_{ij} = \sum_{i \geq 1} P_{ij}$ et aller en (b) sinon aller en (b),
   b) Pour i = t à m-1 faire
6. Déterminer P_i appartient à des ensembles B_k, k = 1, 2…
7. S'il existe des sous-ensembles de (l - 1) éléments dans lesquels P_i et les (l - 1) éléments n'appartiennent pas au même ensemble, de même, P_i et les (l - 2) éléments ne forment pas des plans de découpe à l'étape précédente, alors

déterminer le plan découpe $Pd_h = \begin{cases} P_i \\ (l-1) \text{ éléments'} \end{cases}$, h = 1, 2…, poser t := t + 1 et aller en (b) sinon poser t: = t + 1 et aller en (b)

c) Calculer $b_{ij} = \sum_{i\geq 1} P_{ij}$, $x_j = Max_i \left(\left[\frac{d_i}{b_{ij}} \right] \right)$, $x_{pd_i} = \sum_{j=1}^{T} x_j$, et Min (x_{pd_i}),

d) S'il existe x_{pd} redondants et $x_{pd} >$ Min (x_{pd_i}) alors éliminer x_{pd} redondant et aller en (e) sinon aller en (e),

e) Calculer l'aire totale perdue : $pt = S \times \text{Min}\left(x_{pd_i}\right) - \sum_{r=1}^{n} s_r d_r$,

f) Si $\sum_{j=1}^{T} X_j$ atteint ou n'y a pas d'amélioration des solutions, arrêter, sinon poser l : = l + 1 et aller en (a).

6.3.3 Finitude de l'algorithme

On vérifie dans cette sous-section que le nombre d'itérations est fini et que l'algorithme ne boucle pas :

Construction des plans de coupe : l'algorithme proposé consiste à chercher les lignes qui contiennent des zéros de la première colonne, jusqu'aux lignes qui contiennent des zéros de la $n^{ième}$ colonne, dans une matrice de taille m × n, cela signifie que l'algorithme part de la première colonne et arrive à la $n^{ième}$ colonne, donc, l'algorithme ne boucle pas et comme la matrice est de taille finie alors le nombre d'itérations est fini. D'autre part, les ensembles B_j que construire dans la première étape, l'algorithme cherche à chaque itération les sous-ensembles de l'éléments dans lesquels chaque l-1 où l = 2, 3…, éléments appartiennent à des ensembles différents, jusqu'à ce que le test d'arrêt soit atteint, donc, l'algorithme est terminé et ne boucle pas.

6.3.4 Résultats et discussion

Dans la première partie de cette section, nous présentons les résultats de l'application de la méthode de construction des plans de découpe pour résoudre les problèmes de découpe bidimensionnelle bi-objectif avec cout d'installation (CPBM) [67], sur deux exemples tirés de http://or.dei.unibo.it/library/two-dimensional-bin-packing-problemas, tandis que la troisième exemple est dans Wu et Yang,[79] où chaque exemple contient le nombre total de tous les types de pièces demandés, les tailles des plaques de matières premières en millimètres (mm), les surfaces de pièces requises en millimètres (mm) et la quantité de commande pour chaque type de pièce, ces données sont résumées dans les tableaux 6.9, 6.11 et 6.13. Dans la deuxième partie,

nous comparons ces résultats avec ceux publiés dans la littérature dans les travaux de [79], et [11].

Les résultats obtenus sont présentés dans les tableaux 6.10, 6.12 et 6.14. Quant pour les résultats du tableau 6.14 sont les solutions obtenues par notre méthode pour le tableau 6.9 dans [16], de sorte que chaque tableau contient l'ensemble de solutions efficace, dont chacune d'elles est caractérisée par la surface totale perdue (en pourcentage) et le nombre d'installations à effectuer.

n = 59	S = L × W = 824648										
i	s_i	d_i	i	s_i	d_i	i	s_i	d_i	i	s_i	d_i
1	3984	605	16	3920	93	31	4214	144	45	3266	149
2	1887	44	17	3192	570	31	1190	40	46	705	77
3	2970	95	18	3726	55	32	3248	256	47	4277	801
4	4576	104	19	2665	500	33	3740	422	48	3024	400
5	3520	49	20	3150	430	34	3572	50	49	3936	132
6	2115	51	21	4056	48	35	1978	406,	50	1716	477
7	2925	72	22	4452	154	36	4960	55	51	3403	230
8	3339	45	23	4000	408	37	3120	143	52	4824	225
9	3626	505	24	1394	146	38	2079	403	53	4331	448
10	986	433	25	1435	513	39	5094	141	54	3300	81
11	2550	41	26	1512	53	40	3536	240	55	4020	135
12	2009	409	27	1920	560	41	2660	160	56	3872	230
13	2160	148	28	1280	245	42	1960	507	57	1482	156
14	5016	43	29	3220	505	43	2760	344	58	2112	155
15	656	257	30	1580	58	44	689	500	59	1980	96

Tableau 6.9 : Surfaces de pièces requises et demande

N°_{effs}	$P_{surf\ perd}$	$N_{installs}$	N°_{effs}	$P_{surf\ perd}$	$N_{installs}$	N°_{effs}	$P_{surf\ perd}$	$N_{installs}$
1	0.028	32	10	1.543	24	19	3.586	15
2	0.052	31	11	1.764	23	20	4.034	14
3	0.075	30	12	1.895	22	21	4.483	13
4	0.201	29	13	2.004	21	22	4.154	12
5	0.422	28	14	2.075	20	23	5.269	11
6	0.832	27	15	2.164	19	24	5.578	10
7	1.043	26	16	2.253	18	25	7.043	9
8	1.075	25	17	2.765	17	26	11.216	8
9	1.164	25	18	3.012	16	27	21.475	7

Tableau 6.10 : Résultats des calculs du tableau 6.9

Le tableau 6.10 représente un vecteur de solutions efficaces d'une instance de taille n = 59, de types de pièces très différents ainsi que de tailles de commandes différentes aussi. Alors que la précision des résultats obtenus varie entre 21,475% de surface totale perdue et 5 installations à 0,028% de surface totale perdue et 44 installations, avec un temps d'exécution estimé à 266s. Au vu de ces résultats obtenus, on constate que la qualité de la solution est bonne, pour un temps d'exécution raisonnable.

n =100	S = L × W = 400791										
i	s_i	d_i	i	s_i	d_i	i	s_i	d_i	i	s_i	d_i
1	4469	44	26	5141	62	51	697	108	76	4998	41
2	5152	34	27	4140	106	52	4800	28	77	1681	79
3	4900	21	28	4047	96	53	3780	87	78	550	16
4	4747	10	29	470	15	54	928	98	79	1829	64
5	9870	98	30	3680	41	55	3172	25	80	5340	82
6	5670	88	31	1643	105	56	4212	76	81	1189	404
7	5568	97	32	2346	95	57	4048	39	82	1218	57
8	4224	87	33	2982	86	58	3588	66	83	4545	54
9	1472	77	34	4180	76	59	430	76	84	3564	169
10	4558	67	35	3990	78	60	4510	59	85	4118	53
11	5060	37	36	4356	64	61	4092	22	86	3978	57
12	4494	27	37	5130	54	62	583	12	87	3738	63
13	5520	107	38	560	44	63	4128	80	88	924	83
14	4800	79	39	2793	90	64	540	63	89	5103	109
15	4095	54	40	832	67	65	3913	16	90	5520	154
16	539	44	41	4074	70	66	3477	91	91	2303	53
17	470	84	42	5369	47	67	1488	26	92	2009	147
18	2850	74	43	650	96	68	5040	51	93	4136	86
19	5760	75	44	720	38	69	1960	49	94	3760	71
20	4324	65	45	4698	86	70	1872	39	95	1083	151
21	3442	58	46	5035	46	71	2623	92	96	5202	56
22	5022	48	47	3760	36	72	4182	29	97	2704	133
23	616	61	48	798	97	73	5100	63	98	600	25
24	4770	51	49	1426	44	74	4488	72	99	1600	305
25	645	35	50	828	34	75	1450	11	100	2240	132

Tableau 6.11 : Surfaces de pièces requises et demande

N°_{effs}	$P_{surf\ perd}$	$N_{installs}$	N°_{effs}	$P_{surf\ perd}$	$N_{insttals}$	N°_{effs}	$P_{surf\ perd}$	$N_{insttals}$
1	0.002	44	13	32	0.426	25	2.456	20
2	0.004	43	14	31	0.583	26	2.733	19
3	0.006	42	15	30	0.734	27	3.982	18
4	0.009	41	16	29	0.982	28	4.172	17
5	0.012	40	17	28	1.012	29	5.276	16
6	0.023	39	18	27	1.171	30	6.694	15
7	0.075	38	19	26	1.205	32	6.785	14
8	0.097	37	20	25	1.392	33	7.128	13
9	0.101	36	21	24	1.594	34	8.423	12
10	0.125	35	22	23	1.731	35	10.879	11
11	0.193	34	23	22	1.942	36	14.237	10
12	0.257	33	24	21	2.184	37	20.436	9

Tableau 6. 12 : Résultats des calculs du tableau 6.11

A la lecture du tableau 6.12, nous remarquons que les données de ce tableau sont similaires aux données du tableau 6.9, avec une différence dans la taille de l'instance, où n = 100 dans l'exemple 6.9, alors que n = 59 dans le tableau 6.12 et aussi que les résultats des deux instances sont similaires à l'exception du temps d'exécution, où il a été estimé à 465s, tandis que dans l'instance 6.9 est de 266s, la précision de la méthode reste la même dans les deux instances.

n = 38	S = L × W = 140904										
i	s_i	d_i	i	s_i	d_i	i	s_i	d_i	i	s_i	d_i
1	12012	6	11	3096	6	21	7930	6	31	7200	5
2	2114	5	12	11914	4	22	11970	5	32	6244	5
3	1771	5	13	11448	6	23	5742	6	33	990	6
4	6384	5	14	3213	5	24	6467	5	34	2868	6
5	7992	5	15	8877	5	25	720	4	35	716	6
6	2546	6	16	2916	4	26	10224	4	36	4960	4
7	1612	5	17	5577	4	27	4976	4	37	816	4
8	4914	4	18	2752	5	28	5980	5	38	2613	6
9	7632	6	19	3864	6	29	8670	5			
10	11520	4	20	3180	6	30	10542	5			

Tableau 6. 13 : Surfaces de pièces requises et demande

N°_{effs}	$P_{surf\ perd}$	$N_{installs}$	N°_{effs}	$P_{surf\ perd}$	$N_{installs}$	N°_{effs}	$P_{surf\ perd}$	$N_{installs}$
1	0.012	15	5	1.035	11	9	3.056	7
2	0.026	14	6	1.148	10	10	5.376	6
3	0.157	13	7	2.102	9	11	7.114	5
4	0.346	12	8	2.445	8	12	11.346	4

Tableau 6.14 : Résultats des calculs du tableau 6.13

Dans le troisième cas de cette analyse où les instances des tableaux 6.9 et 6.13 sont de tailles non éloignées l'une de l'autre n = 59 et n = 38, alors que les quantités de commande de chaque type de pièces de chaque instance sont très différentes. Cependant, la méthode conserve la même précision dans les deux exemples.

n =10	S = 625 × 102			n = 10	S = 25 × 104			n = 10	S = 25 × 104		
ID	N°_{eff}	$P_{surf\ perd}$	$N_{instals}$	ID	N°_{effs}	$P_{surf\ perd}$	$N_{installs}$	ID	N°_{effs}	$P_{surf\ perd}$	$N_{installs}$
Gcut1dr	1	0.112	6	Gcut2dr	1	0.022	6	Gcut3dr	1	0.513	6
	2	0.526	5		2	1.526	5		2	1.507	5
	3	1.257	4		3	5.154	4		3	6.621	4
	4	3.465	3		4	9.377	3		4	10.28	3
Gcut4dr	n = 20			Gcut5dr	n = 20			Gcut6dr	n = 20		
	1	0.178	11		1	0.152	11		1	0.122	11
	2	0.463	10		2	0.321	10		2	0.254	10
	3	0.748	9		3	0.549	9		3	0.638	9
	4	0.952	8		4	0.723	8		4	1.501	8
	5	1.236	7		5	1.154	7		5	2.353	7
	6	2.362	6		6	2.661	6		6	4.311	6
	7	9.475	5		7	5.475	5		7	7.525	5
	8	15.564	4		8	14.564	4		8	11.61	4
Gcut7dr	n = 30				9	18.423	3		9	25.02	3
	1	0.113	15	Gcu8dr	n = 30			Gcut9dr	n = 30		
	2	0.762	14		1	0.121	15		1	0.081	15
	3	1.236	13		2	0.502	14		2	0.413	14
	4	1.975	12		3	1.317	13		3	1.117	13
	5	3.123	11		4	1.836	12		4	1.532	12
	6	4.857	10		5	3.101	11		5	2.001	11
	7	6.379	9		6	3.652	10		6	2.453	10
	8	8.937	8		7	5.721	9		7	4.131	9
	9	9.075	7		8	7.134	8		8	4.538	8
	10	11.473	6		9	8.436	7		9	5.422	7
	11	14.575	5		10	10.55	6		10	7.273	6
	12	25.436	4		11	15.64	5		11	11.43	5
					12	27.45	4		12	14.26	4
									13	20.57	3

	n = 50				n = 50				n = 50		
	1	0.076	20		1	0.102	19		1	0.064	19
	2	0.117	19		2	0.212	18		2	0.112	18
	3	0.506	18		3	0.416	17		3	0.334	17
	4	0.673	17		4	0.673	16		4	0.537	16
	5	1.546	16		5	1.124	15		5	1.012	15
	6	1.895	15		6	1.478	14		6	1.352	14
	7	2.783	14		7	2.047	13		7	1.836	13
Gcut10dr	8	3.274	13	Gcut11dr	8	2.548	12	Gcut12	8	2.711	12
	9	3.815	12		9	4.181	11	dr	9	3.572	11
	10	4.063	11		10	4.665	10		10	4.345	10
	11	5.943	10		11	6.763	9		11	7.161	9
	12	08.56	9		12	9.362	8		12	10.45	8
	13	10.05	8		13	13.74	7		13	15.28	7
	14	13.65	7		14	20.45	6		14	21.34	6
	15	19.67	6		15	28.06	5				

Tableau 6.15 : Résultats des données de Cui et Zhao, par CPBM [11]

Dans cette comparaison, nous utilisons l'instance APT30 dans [79] qui représente l'exemple 6.13 de notre travail et qui a été résolu par les heuristiques ISHP, RCCG et SVCH, et les résultats ont été qu'APT30 atteint sa limite inférieure dans l'ISHP heuristique, mais n'atteint pas sa borne inférieure dans les autres heuristiques, alors que notre méthode proposée [63] est meilleure que l'heuristique précédente en termes de qualité de solution car elle fournit un ensemble de solutions efficaces comme indiqué dans le tableau 6.14 en un temps de 135s et elle atteint sa limite inférieure à partir de 12 itérations.

A la lecture des résultats présentés dans le tableau 6.15 des travaux de Cui et Zhao, [11] on constate que les heuristiques utilisées, chacune d'elles était adaptée à la résolution de certaines instances et non adaptée à d'autres. Par exemple, la solution RCCG est meilleure dans 1 instance (Gcut3dr), la même dans 9 instances, et moins bonne dans 2 instances (Gcut4dr et Gcut8dr), avec un temps de calcul moyen d'une instance de 0,073 seconde (cas non orienté). Alors que notre technique, elle résout les 12 instances avec la même efficacité comme indiqué dans le tableau 6.15, sauf que l'approche RCCG est meilleure au temps de l'exécution car notre méthode

exécute ces instances avec un temps variant entre 55 secondes et 465 secondes selon la taille de l'instance.

6.3 Conclusion

Dans ce chapitre, nous avons présenté notre contribution en résolvant deux variantes de problèmes de découpe, la première pour résoudre le problème de découpe uni-dimensionnelle mono-objectifs, et la seconde pour résoudre le problème de découpe bidimensionnel bi-objectifs. Les deux méthodes proposées ont été testées sur plusieurs exemples de tailles différentes et comparées à plusieurs méthodes précédentes utilisées pour résoudre les mêmes problèmes. Un ordinateur présentant les caractéristiques suivantes a été utilisé : Xeon® Silver 4110 machine (2,1 GHz/8 cœurs/11,00 Mo/85 W) Ram 16 Go, DDR4 2400T HDD 1 To Sata DVD-RW Windows 10 Pro 64 bits.

La comparaison est faite entre les valeurs obtenues à partir de la perte de matière première et le nombre d'installation pour chaque méthode. La meilleure méthode est celle qui donne le moins de perte de matière première avec un plus petit nombre d'installations. Les deux méthodes proposées ont prouvé leur valeur dans la résolution des deux problèmes étudiés dans ce travail et leur supériorité dans de nombreux cas par rapport aux autres méthodes.

CONCLUSION GÉNÉRAL ET PERSPECTIVES

Le fil conducteur des travaux présentés dans cette thèse, s'appuie principalement sur l'utilisation de l'approche multi-objectif pour résoudre deux variantes du problème de découpe, Il s'agit du problème de découpe unidimensionnelle avec coût d'installations et du problème de découpe bidimensionnelle avec coût d'installation. Les deux problèmes étudiés portent sur la minimisation de deux fonctions objectif : la perte totale de matière première et le nombre d'installations. Sous la contrainte de satisfaire la demande, cette approche est différente de l'approche d'optimisation combinatoire mono-objectif de différentes manières, par exemple la notion d'une solution optimale résultant d'une évaluation unique de plusieurs coûts d'une fonction objectif peut ne pas toujours la meilleure façon pour résoudre un problème combinatoire surtout s'il est possible d'évaluer chaque coût séparément, comme dans le cas des problèmes de découpe lorsqu'il s'agit de minimiser la perte totale de matière première et le nombre d'installations. Où l'évaluation indépendante de chaque objectif permet de calculer un ensemble de solutions de compromis pour lesquelles il n'existe pas d'autre solution meilleure sur chacun des objectifs. Comme ces solutions sont appelées solutions efficaces, elles rendront l'étude plus réaliste. Pour cela nous avons introduit deux nouvelles contributions basées sur la génération des patrons de découpe et la constriction des plans de découpe afin de trouver des solutions efficaces pour ces deux types de problèmes de découpe. En premier lieu, notre étude s'est concentrée principalement sur la recherche de l'ensemble des solutions efficaces du problème unidimensionnel, Puis, l'accent a été mis sur la recherche des solutions efficaces au problème bidimensionnel.

Afin de tester les performances des méthodes proposées, celles-ci ont été testées sur de nombreux exemples de la littérature, et afin de prouver leur efficacité, elles ont été comparées à de nombreuses méthodes apparues dans des travaux antérieurs. Et à travers l'étude expérimentale, qui, comme mentionné ci-dessus, il s'avère la capacité de ces méthodes à apporter des solutions appropriées au problème étudié, comme il est devenu clair grâce à l'étude comparative précédente à quel point ces méthodes sont compétitives. Au terme de cette conclusion, on peut évoquer quelques axes de recherche qui ont été ouverts :

1. Une voie de recherche intéressante consiste à trouver les solutions en deux étapes peut améliorer notre approche et accroître son efficacité.

2. Une autre voie de recherche pourra également être considérée est celle qui consiste à augmenter le nombre des objectifs contradictoires à optimiser dans le problème de découpe, comme les dates dues.
3. Un axe de recherche qui peut être étudié dans les futurs travaux de recherche est l'hybridation de l'heuristique séquentielle avec notre approche proposée.

REFERENCES

[1] R. Alvarez-Valdes, A. Parajon, M. J. Tamarit. A computational study of lp-based heuristic algorithms for two-dimensional guillotine cutting stock problems. *OR Spectrum*, 24 (2), URL http://dx.doi.org/10.1007/s00291-002-0093-3, pp 179–192, 2002.

[2] S. Araujo, K. C. Poldi, and J. Smith. A genetic algorithm for the one-dimensional cutting stock problem with Setups. *Pesquisa Operacional*, 34(2). Doi :10.1590/0101-743-8. 2014.034.020165, pp 165-187, 2014.

[3] U. Ayasandır, M. Azizoğlu. Two-dimensional cutting stock problems with multiple stock sizes. To appear, in *International Journal of Planning and Scheduling*, pp1-61, 2021.

[4] V. R. Basili, A. J. Turner. Iterative enhancement: A practical technique for software Development. *IEEE Transactions on Software Engineering*, vol. 1, no. 4, pp 390, 396, 1975.

[5] G. Belov, G. Scheithauer, A Branch-and-cut-and-Price Algorithm for one- dimension-Al stock cutting and two-dimensional two-stage cutting. *European Journal of Operational Research*, Volume 171, Issue 1. pp 85-106, 16 May 2006.

[6] J. E. Beasley. An algorithm for the two-dimensional assortment problem. *European Journal of Operational Research*, Vol.19: pp 253 - 261, 1985.

[7] J. E. Beasley. Algorithms for unconstrained two-dimensional guillotine cutting. *Journal Operational Research and Society*, Vol.36, pp 297-306, 1985.

[8] M. Biro, E. Boros. Network ows and non-guillotine cutting patterns. *European Journal of Operational Research*, Vol.16, pp 215 - 221, 1984.

[9] V. Chvàtal, Linear Programming, *Verlag: W.H. Freeman and Company*, ISBN 10: 07-16715872 ISBN 13: 9780716715870, 1983.

[10] G. Cintra, W. Yakabayashi. Dynamic programming and column generation based apiproaches for two-dimensional guillotine cutting problems. In experimental and efficent algorithms, *volume 309 of lecture Notes in Computer Science*, pp175–190, 2004.

[11] Y. Cui, Z. Zhao, Heuristic for the rectangular two-dimensional single stock size cutting stock problem with two-staged patterns. *European Journal of Operational Research* 231 (2): pp 288-298, 2013.

[12] Y. Cui, L. Yang, Z. Zhao, T. Tang, and M. Yin. Sequential grouping heuristic for the Two-dimensional cutting stock problem with pattern reduction. *International Journal of production economics*, 144(2), pp 432-439, 2013.

[13] Y. Cui, Y. Yang. A heuristic for the one-dimensional cutting stock problem with usa-

Ble Leftover. *European Journal of Operational Research*, 204(2), pp 245–250, 2010

[14] Y. Cui, C. Zhong, Y. Yao, Pattern-set generation algorithm for the one-dimensional cutting stock problem with setup cost, *European Journal of Operational Research*, 243 (2). Doi : 10.1016/j. ejor. 2014.12.01, pp 540 - 546, 2015.

[15] Y. Cui, B. Huang. Heuristic for constrained T-shape cutting patterns of rectangular pieces. *Comput Oper Res.* 39(12): pp 3031-3039, 2012.

[16] M. C. Costa, Formalisation et résolution des problèmes de découpe linéaires, *R.A.I.R O Recherche Opérationnelle*, 16, pp 65-82, 1982.

[17] C.G. Codd, Multiprogram sheduling. *Comm. of the ACM*, Vol.3 (6): pp 347-350, 1960

[18] T. H. Cormen, C. E. Leiserson and R. L. Rivest. Introduction to algorithms, chapter 1 6 greedy algorithms. MIT Press and McGraw-Hill, 1st edition, pp 1- 68, 1990.

[19] Y. Collette, P. Siarry. Optimisation multiobjectif. *Eyrolles*, 2002.

[20] C. Darwin. On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life. *John Murray*, 1859.

[21] J. De Armas, G. Miranda, C. León, A multi-objective approach for the 2D guillotine Cutting stock problem. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6692 LNCS (PART 2), pp 292–299, 2011.

[22] M. Delorme, M. Iori, and S. Martello. Bin packing and cutting stock problems, Mat-Hematical models and exact algorithms. *European Journal of Operational Research*, 255 (1), pp 1- 20, 2016.

[23] R. A. DeVore, V. N. Temlyakov. Some remarks on greedy algorithms. *Advances in Computational Mathematics*, vol. 5, nº. 1, pp 173-187, 1996.

[24] P. DeCani. A note on the two-dimensional rectangular cutting-stock problem. *Europ-Ean Journal of Operational Research*, Vol.29: pp.703-706, 1978.

[25] L. L. De Salles Neto, A. R. Lizana, M. A. Jimenez, and G. Ruiz-Garzon Weak efficiency in cutting stock problem. *XXXIII Congresso Nacinal de Mathmatica Aplicada E Computacial*, 2010. pp 187-193, 2010.

[26] F. Dusberger, G. R. Raidl, Solving the 3-staged 2-dimensional cutting stock problem by Dynamic programming and variable neighborhood search. *Electronic Notes in discrete Mathematics* 47URL http://dx.doi.org/10.1016/j.endm. 2014. 11.018 , pp 133 – 140, 2015.

[27] H. Dyckhoff. A typology of cutting and packing problems. *European Journal of Operation Research*, 444, pp 145-159, 1990.

[28] F.Y. Edgeworth. Mathematical Physics. P. Keagan, London, 1881.

[29] A. A. Farley. Mathematical programming models for cutting-stock problems in the clothing industry. *Journal of the Operational Research Society*, 39(1), pp 41–53. 1988.

[30] F. Furini, , E. Malaguti. Models for the two-dimensional two-stage cutting stock problem with multiple stock size. *Computers and Operations Research*, *40*(8), pp 1953–1962, 2013.

[31] L. Fortnow. The status of the P versus NP problem. *Communications of the ACM*, vol. 52, no. 9, pp 78_86, 2009.

[32] M. R. Garey, R. L. Graham, Bounds on multiprocessing scheduling with resource constraints. Siam, Jour. Compute, Vol.4(2), pp187-200, 1975.

[33] P. C. Gilmore, R. E. Gomory, A Linear programming approach to the cutting stock problem. *Operations Research*, 9, pp 849 -859, 1961.

[34] P. C. Gilmore, R. E. Gomory, A linear programming approach to the cutting stock Problem Part II. *Operations Research*, 11 (6), pp 863–888, 1963.

[35] P. C. Gilmore, R .E. Gomory, Multistage cutting stock problems of two and more dimensions. *Operations Research*, 13 (1), pp 94–120, 1965.

[36] Y. Gonzalez, G. Miranda, and C. Leon, multi-objective multi-level filling evolutionary algorithm for the 3D cutting stock problem, *Procedia Computer Science*, 96, pp 355 - 364, 2016.

[37] R.W. Haessler. A Heuristic programming approach to a nonlinear cutting stock problem. *Management science*, 17, pp B793 – B802, 1971.

[38] R. W. Haessler. Controlling cutting pattern changes in one-dimensional trim problems, *Operations Research*. 23, pp 483 - 493, 1975.

[39] R. W. A Haessler, Note on computational modifications to the Gilmore and Gomory Cutting stock problem, *Operations Research*, 28, pp 1001 - 1005, 1980.

[40] R. W. Haessler, Doveloping an industrial-grade heuristic problem-solving procedure, *Interface*, 13, pp 62-71, 1983.

[41] R.W. Haessler. Selection and design of heuristic procedures for solving roll trim loss Problems, *Management, Science* 34 (12), pp 1460–1471, 1988.

[42] R.W. Haessler, P.E. Sweeney, cutting stock problem and solution procedure europeaN *Journal of Operational Research*. 54, pp. 141 – 150, 1991.

[43] M. Hifi. Dynamic programming and hill-climbing techniques for constrained two-dimensional cutting stock problems. *Journal of Combinatorial Optimization*, 8 (1), pp 65 –84, 2004.

[44] M. Hifi, C. Roucairol, Approximate and exact algorithms for constrained (un) weighted two-dimensional two-staged cutting stock problems. *Journal of Combinatorial Optimization*, 5 (4), pp 465– 494, 2001.

[45] M. Hifi, Approximate algorithms for the container loading problem. *International Transactions in Operational Research*, Vol.9, pp.747-774, 2002.

[46] M. Hifi, T. Saadi. A cooperative algorithm for constrained two-staged 2d cutting problems. *IEEE Service Systems and Service Management international Conference, Troy-France*, Vol. 2, pp 928-933, 2006.

[47] M. Hifi, T. Saadi. Using bounded knapsack problems for two-staged two-dimensional Cutting stock problems. In international conference on met heuristics and nature inspired computing, *META, Tunis*, Tunisie, pp 212-221, 2006.

[48] S. Henn, G. Wäscher. Extensions of cutting problems, setups. *Pesquisa Operational*, Vol 33(2). pp 133-162, 2013.

[49] J. Holland. Adaptation in natural and artificial systems, Mit. Press, Cambridge, Mass, 1975.

[50] D. S. Johnson. The NP-completeness column, an ongoing guide. *Journal of Algorithms*, vol. 6, no. 3, pp 434-451, 1985.

[51] F.J. Johnston, T. Pethica, Optimization methods in the paper industry, *The Paper Maker and British Paper Trade Journal*, 137, pp 68 -70, 1959.

[52] S. Juttijudata, P. Sudjarittham, two-dimensional cutting stock problems with a modified column generation method, *Current Applied Science and Technology*, Vol. 20 N^{o} 2, May-August 2020. pp 217- 235, 2020.

[53] R. M. Karp. Reducibility among combinatorial problems, *Complexity of Computer Computations*, pp 85-104, 1972.

[54] A. W. J. Kolen, F. C. R. Spieksma, solving a bi-criterion cutting stock problem with Open-ended demand: a case study. *Journal of The Operational Research Society*, 51, pp 1238-1247, 2000.

[55] L. V. Kantorovich, Mathematical methods of organizing and planning production. *Management Sci.* 6, pp 366-422, 1960.

[56] B. Mans, C. Roucairol, Performances of parallel branch and bound algorithms with Bestirs search. *Discrete Applied Mathematics*, Vol.66: pp.57-76, 1996.

[57] J. Martinovic, G. Scheithauer, and J. M. Valério de Carvalho. A comparative study of the arc flow model and the one-cut model for one-dimensional cutting stock problem *European Journal of Operational Research*, 266 (2), pp 458-471, 2018.

[58] M. Monaci, A. Lodi, Integer linear programming models for 2-staged two-dimensional knapsack problems. *Mathematical Programming*, Vol. 94: pp 257-278, 2003.

[59] A. Mellouli, A. Dammak. an algorithm for the two-dimensional cutting stock problem based on a pattern generation procedure, *International Journal of Information and Management Sciences*, 19 (2): pp 201-218, 2008.

[60] A. Mellouli, R. Mellouli, and F. Masmoudi. An innovative genetic algorithm for a multi-objective optimization of two-dimensional cutting-stock problem. *Applied Artificial Intelligence*, 33(6), pp 531–547, 2019.

[61] A. Mobasher, A. Ekici. Solution approaches for the cutting stock problem with setup Cost. *Computers & Operations Research*, 40(1), http://dx.doi.org/10.1016/j.cor.2012.06.007, pp 225-235, 2013.

[62] F. Oliveira, A, J. Neuenfeldt, E. Silva, and M. A. Carravilla, A survey on heuristics For the two-dimensional rectangular strip packing problem, *Pesquisa Operational* 36 (2): version ISSN0101-7438/On line version ISSN 1678-5142 www.scielo.br/pope doi: 10.1590/0101-7438.2016. 036.02.0197, pp 1- 31, 2016.

[63] K. Pazand, A. Mohammadi. Extended haessler heuristic algorithm for cutting stock problem: a case study in film Industry. *Australian Journal of Basic and Applied Sciences*, 3(4): ISSN 1991-8178, pp 3944-3953, 2009.

[64] V. Pareto. Cours d'économie politique. Rouge, Lausanne, 1896.

[65] W. N. P. Rodrigo, W.B. Daundasekera and A. A. I. Perera. Pattern Generation for two Dimensional cutting stock problem. *International Journal of Mathematics Trends and Technology* (IJMTT), Vol. 3 Issue2, pp 54- 62, 2012.

[66] B. Slimi, M. Abbas. Contribution to solving a one-dimensional cutting stock problem With two objectives based on the generation of cutting patterns. *Revista Electrónica de Comunicaciones y Trabajos de ASEPUMA. Rect@.* Volumen 23(2022). doi : 10. 24309 /recta.2022. 22.2.04. pp 1-22. 2022.

[67] B. Slimi, M. Abbas. Contribution to solving a two-dimensional cutting stock problem With two objectives. *Economic Computation and Economic Cybernetics Studies and Research*, Issue 2/2022; Vol. 56. doi 1024818/18423264/56.2.22.12. pp 179-194, 2022

[68] E. Silva, F. Alvelos, J. M. Valério de Carvalho. An integer programming model for two-and three-stage two-dimensional cutting stock problems. *European Journal of Operational Research* 205 (3), URL http://dx.doi.org/10.1016/ j. ejor. 2010.01.039, pp 699–708, 2010.

[69] M. Sipser. The history and status of the P versus NP question. In Proceedings of the 2

4th annual ACM Symposium on Theory of Computing (STOC'92), Victoria, BC, Cana da, May 4-6, pp 603- 618,1992.

[70] K. Spielberg. Enumerative methods in integer programming. *Annals of Discrete Mathematics*, vol. 5, pp139-183, 1979.

[71] S. M. Suliman, Pattern generating procedure for the cutting stock problem. *International Journal of Production Economics*, 74(1-3), doi: 10.1016/S0925-5273(01) 00134-7, pp 293–30, 2001.

[72] S. Voßs, A. Fink, and C. Duin. Looking ahead with the pilot method. Annals of *Operations Research*, 136(1), pp 285–302, 2005.

[73] D. Tanir, O. Ugurlu, A. Guler, and U. Nuriyev. One-dimensional cutting stock proble M with divisible items: a case study in steel industry, *TWMS J. App. Eng. Math*, V.9, N.3473-484, pp 1-12, 2019.

[74] S. Umetani, M. Yagiura, and T. Ibaraki. An LP-based local search to the one-dimensional cutting stock problem using a given number of cutting patterns. *IEICE Transactions on fundamentals of electronics, communications and computer Sciences*, E 86- A pp 1093–1102, 2003.

[75] S. Umetani, M. Yagiura, and T. Ibaraki. One dimensional cutting stock problem with a given number of setups: A hybrid approach of met-heuristics and linear programming, *Journal of Mathematical Modelling and Algorithms* 5, pp 43-64, 2006.

[76] G. Wäscher, H. Haußner, H. Schumann. An improved typology of cutting and Packing problems. *European Journal of Operational Research*. 183 (3), URL http://dx.doi. . org/10.1016/j.ejor.2005.12.047, pp 1109 -1130, 2007.

[77] F. Vanderbeck. Exact algorithm for minimising the number of setups in the one dimensional cutting stock problem, *Oper. Res*. 48, pp 915-926. 2000.

[78] K. V. Viswanathan, A. Bagchi, Best-first search methods for constrained two-dimens ional cutting stock problems. *Operations Research* 41(4), pp 768-776, 1993.

[79] D. Wu, G. Yang. A Heuristic approach for two-dimensional rectangular cutting stock Problem considering Balance for Material Utilization and Cutting Complexity, *Advances in Materials Science and Engineering,* Volume 2021, Article ID 3732720,15 htt ps://doi.org/10. 1155 /2021 /3732720, pp 1-15, 2021.

[80] D. A. Wuttke, H. S. Heese. Two-dimensional cutting stock problem with sequence dependent setup times, *European Journal of Operational Research*. 265(1). Doi :10. 101 6/j.ejor.2017.07.036, pp 303-315, 2017.

Annexe A

Exemple A .1 : Afin d'illustrer les étapes d'application de l'algorithme de génération de colonnes, nous considérons le problème de découpe en variables continue où nous minimisons la somme des objets à couper sous la contrainte de satisfaire exactement la commande :

$$\begin{cases} \min\left(\sum_{j=1}^{T} x_j\right) \\ \text{s.c. } \sum_{j=1}^{T} p_{ij}\, x_j = d_i \quad i = 1, \dots, n \\ x_j \in \mathbb{N}, \qquad j = 1, \dots, T \end{cases} \tag{A.1}$$

Alors que la taille de l'objet principal (W) et les tailles des pièces requises (w_i). Ainsi, les quantités de demandes (d_i) sont résumées dans le tableau suivant :

n = 4	**W = 91(cm)**	
i	w_i (cm)	d_i (cm)
1	25,5	78
2	22,5	40
3	20	30
4	15	30

Tableau A.1: Longueurs de pièces requises et demandes

Variables de décision : x_j : Le nombre de pièces découpés de le patron j, $x_j \geq 0$, (j = 1, …, T).

Fonction objectif : Minimiser le nombre de objets à découpés : $\min\left(\sum_{j=1}^{T} x_j\right)$

Contraintes :

$$\begin{cases} \sum_{j=1}^{T} p_{1j}\, x_j = 78 \\ \sum_{j=1}^{T} p_{2j}\, x_j = 40 \\ \sum_{j=1}^{T} p_{3j}\, x_j = 30 \\ \sum_{j=1}^{T} p_{4j}\, x_j = 30 \end{cases}$$

Forme matricielle : $\begin{cases} \min c^T x \\ Ax = b \\ x \geq 0 \end{cases}$, $\quad b = \begin{bmatrix} 78 \\ 40 \\ 30 \\ 30 \end{bmatrix}$, $\quad c^T = [1,\ 1, 1, 1, 1, \dots, 1]$,

$$P = \begin{pmatrix} 3 & 2 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 3 & 2 & 2 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 4 & 3 & 0 & \cdots & 0 \\ 0 & 0 & 2 & 1 & 1 & 0 & 0 & 1 & 0 & 2 & 1 & 0 & 3 & 2 & 3 & 2 & 1 & 0 & 1 & 3 & \cdots & 0 \\ 1 & 1 & 0 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 1 & 2 & 0 & 1 & 0 & 1 & 3 & 0 & 0 & 2 & \cdots & 6 \end{pmatrix}$$

La matrice P, contient tous les vecteurs-colonnes (P_1, P_2, P_3, P_4) tels que :

$$25{,}5P_1 + 22{,}5P_2 + 20P_3 + 15P_4 \leq 91$$

Initialisation du simplexe révisé : $B = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 6 \end{bmatrix}$ avec b_{11} = E [91/25.5] = 3

b_{22}= E [91/22.5] = 4, b_{33} = E [91/20] = 4, b_{44} = E [91/15] = 6, où E est la partie entière.

$$\widetilde{x_b} = \begin{bmatrix} 26 \\ 10 \\ 7{,}5 \\ 5 \end{bmatrix}$$

$x_1 = b_{11}/b_1 = 78/3 = 26$, $x_2 = b_{22}/b_2 = 40/4 = 10$, $x_3 = b_{33}/b_3 = 30/4 = 7.5$, $x_4 = b_{44}/b_4 = 20/4 = 5$.

1ère itération

1) Génération de la colonne entrante

a) Calcul de y^T tel que $y^T B = c_B^T$ (où $B^T y = c_B$) $\Rightarrow y^T = \left[\frac{1}{3}, \frac{1}{4}, \frac{1}{4}, \frac{1}{6}\right]$

b) Générer colonne *a* telle que $y^T a > 1$

$$\begin{cases} \frac{1}{3}P_1 + \frac{1}{4}P_2 + \frac{1}{4}P_3 + \frac{1}{6}P_4 > 1 \\ 25{,}5P_1 + 22{,}5P_2 + 20P_3 + 15P_4 \leq 91 \\ P_1, P_2, P_3, P_4 \in \mathbb{N} \end{cases}$$

Problème de sac-à-dos : $\max(y^T P)$ avec $P = \begin{bmatrix} 2 \\ 0 \\ 2 \\ 0 \end{bmatrix}$ et $y^T P = \frac{7}{6} > 1$,

2) Détermination de la colonne sortante

a) Calcul de d tel que $Bd = P \Rightarrow d = \begin{bmatrix} 2/3 \\ 0 \\ 1/2 \\ 0 \end{bmatrix}$

b) Trouver max(t) tel que $\widetilde{x_B} - td \geq 0$

$$\begin{cases} 26 - \frac{2}{3}t \geq 0 \\ 7.5 - \frac{1}{2}t \geq 0 \end{cases} \Rightarrow t = 15 \Rightarrow \text{Suppression de la } 3^{\text{ème}} \text{ colonne}$$

3) Mise à jour de $\widetilde{x_B}$: $\begin{bmatrix}26\\10\\7,5\\5\end{bmatrix}$ - 15 $\begin{bmatrix}2/3\\0\\1/2\\0\end{bmatrix}$ = $\begin{bmatrix}16\\10\\0\\5\end{bmatrix}$, $B = \begin{bmatrix}3&0&2&0\\0&4&0&0\\0&0&2&0\\0&0&0&6\end{bmatrix}$, $\widetilde{x_b} = \begin{bmatrix}16\\10\\15\\5\end{bmatrix}$

2ème itération

1) Génération de la colonne entrante

a) Calcul de y^T tel que $y^T B = c_B^T$ (où $B^T y = c_B$) $\Rightarrow y^T = \left[\frac{1}{3}, \frac{1}{4}, \frac{1}{6}, \frac{1}{6}\right]$

b) Générer colonne a telle que $y^T P > 1$,

$$\begin{cases} \frac{1}{3}P_1 + \frac{1}{4}P_2 + \frac{1}{6}P_3 + \frac{1}{6}P_4 > 1 \\ 25{,}5P_1 + 22{,}5P_2 + 20P_3 + 15P_4 \leq 91 \\ P_1, P_2, P_3, P_4 \in \mathbb{N} \end{cases}$$

Problème de sac-à-dos : $\max(y^T P)$ avec $P = \begin{bmatrix}2\\1\\0\\1\end{bmatrix}$ et $y^T P = \frac{13}{12} > 1$,

2) Détermination de la colonne sortante

a) Calcul de d tel que $Bd = P \Rightarrow d = \begin{bmatrix}2/3\\1/4\\0\\1/6\end{bmatrix}$,

b) Trouver max(t) tel que $\widetilde{x_B} - td \geq 0$, t = 24 $\Rightarrow$ Suppression de la 1ème colonne

3) Mise à jour de $\widetilde{x_B}$: $\begin{bmatrix}26\\10\\15\\5\end{bmatrix}$ - 24 $\begin{bmatrix}2/3\\1/4\\0\\1/6\end{bmatrix}$ = $\begin{bmatrix}0\\4\\15\\1\end{bmatrix}$, $B = \begin{bmatrix}2&0&2&0\\1&4&0&0\\0&0&2&0\\1&0&0&6\end{bmatrix}$, $\widetilde{x_b} = \begin{bmatrix}24\\4\\15\\1\end{bmatrix}$

3ème itération

1) Génération de la colonne entrante

a) Calcul de y^T tel que $y^T B = c_B^T$ (où $B^T y = c_B$) $\Rightarrow y^T = \left[\frac{7}{24}, \frac{1}{4}, \frac{5}{24}, \frac{1}{6}\right]$

b) Générer colonne a telle que $y^T a > 1$,

$$\begin{cases} \frac{7}{24}P_1 + \frac{1}{4}P_2 + \frac{5}{24}P_3 + \frac{1}{6}P_4 > 1 \\ 25{,}5P_1 + 22{,}5P_2 + 20P_3 + 15P_4 \leq 91 \\ P_1, P_2, P_3, P_4 \in \mathbb{N} \end{cases}$$

Problème de sac-à-dos : $\max(y^T P)$ avec $P = \begin{bmatrix} 2 \\ 1 \\ 1 \\ 0 \end{bmatrix}$ et $y^T P = 1 \not> 1$,

La solution courante est optimale : $B = \begin{bmatrix} 2 & 0 & 2 & 0 \\ 1 & 4 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 1 & 0 & 0 & 6 \end{bmatrix}$, $\widetilde{x_b} = \begin{bmatrix} 24 \\ 4 \\ 15 \\ 1 \end{bmatrix}$,

À partir de rouleaux de 91cm, on doit procéder aux découpages suivants :

Donc les nombres optimaux des objets utilisés sont 44 objets, avec une perte totale est 65 cm.

Annexe B

Afin de clarifier le fonctionnement de l'algorithme proposé pour résoudre le problème de découpe unidimensionnelle à deux objectifs avec coût d'installation, nous l'exécutons dans l'exemple suivant :

Exemple B.1

• W = 80, n = 3,

• w = (w_1, w_2, w_3) = (50, 40, 30),

• d = (d_1, d_2, d_3) = (4, 6, 11),

1. Calculer $p_{11} = \left\lfloor \frac{80}{50} \right\rfloor = 1$, poser j = 1, pour i = 2 à 3 faire $p_{12} = \left\lfloor \frac{W - \sum_{z=1}^{1} p_{1z}}{w_2} \right\rfloor = \left\lfloor \frac{80 - 1 \times 50}{40} \right\rfloor = 0$ $p_{13} = \left\lfloor \frac{80 - 1 \times 50 - 0 \times 40}{30} \right\rfloor = 1$,

   P = (1 0 1)

2. Poser i = 1, h =1, k =1,

a) Poser j =1, d =1,

b) $p_{11} = 1 > 0$ alors j := j +1, l =1,

c) h = h + 1= 1 + 1 = 2,

d) i = 1 alors $p_{21} = p_{11} - 1 = 1 - 1 = 0$,

   Pour i = 2 à 3 faire $p_{22} = \left\lfloor \frac{80 - 0 \times 50}{40} \right\rfloor = 2$, $p_{23} = \left\lfloor \frac{80 - 0 \times 50 - 2 \times 40}{30} \right\rfloor = 0$,

   P = $\begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \end{pmatrix}$

e) p_{21} = 0 alors aller en (f),

f) d = d + 1 = 1 + 1 = 2,

g) d = m = 2

h) i = 1< n -1 = 2 alors poser i = i + 1= 2, k = k +1= 2, aller en (a),

a) Poser j = 1, d = 1,

b) p_{12}= 0,

i) Poser j = j +1= 2, d = d +1=2,

   b) p_{22} = 2 > 0 alors j = j + 1= 3, l = 1,

   c) h = h + 1= 2 + 1= 3,

   d) Pour z = 1 à 1 faire $p_{31} = p_{21}$ = 0,

      Pour i = k = 2 faire $p_{32} = p_{22} - 1 = 2 - 1 = 1$,

Pour i = 3 à 3 faire $p_{33} = \left\lfloor \frac{80 - 0 \times 50 - 1 \times 40}{30} \right\rfloor = 1$,

$$P = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \end{pmatrix},$$

e) $P_{32} = 1 > 0$, alors poser l = l + 1= 2, d = d + 1 = 3 aller en (c),

c) h = h + 1 = 4,

d) Pour z = 1 à 1 faire $p_{41} = p_{31} = 0$,

Pour i = k = 2 faire $p_{42} = p_{22}$ -2= 2 – 2= 0,

Pour i = 3 à 3 faire $p_{43} = \left\lfloor \frac{80 - 0 \times 50 - 0 \times 40}{30} \right\rfloor = 2$,

$$P = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix},$$

e) $P_{42} = 0$, d = d + 1= 4, i = 2 = n -1, stop.

∄ un patron de découpe P_r, (r = 1, …, 4) où $P_{rj} \neq 0$, (j = 1, …, 3), alors aller en (4),

3. La valeur théorique est $\sum_{j=1}^{T} X_j = \left\lceil \frac{770}{80} \right\rceil = 10$,

4. Les patrons de découpe réalisables codés sont indiqués dans la matrice suivante :

$$P' = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

5. Le vecteur $V = \sum_{r=1}^{m} \sum_{j=1}^{n} p'_{ij} w_j$, présenté par le vecteur suivant : $V = \begin{pmatrix} 80 \\ 40 \\ 70 \\ 30 \end{pmatrix}$,

$X = (x_1, x_2, x_3, x_4)$, $U = (u_1, u_2, u_3 \ ...)$, C = (80, 40, 70, 30) et m = cardinale (V) = 4.

j = 1, pour i = 1 à 4 faire $x_{11} = x_1$, $x_{21} = x_2$, $x_{31} = x_3$, $x_{41} = x_4$,

1. Pour j = 1 à 2, pour i = 4 – j + 1 à 2 faire $x_{i-1,j+1} = x_{ij}$,

j = 1, pour i = 4 à 2 faire $x_{32} = x_{41}$, $x_{22} = x_{31}$, $x_{12} = x_{21}$,

j = 2, pour i = 3 à 2 faire $x_{23} = x_{32}$, $x_{13} = x_{23}$,

6. Pour j = 1 à 3, pour i = 1 à 4 – j connecter le sommet x_{ij} par le sommet $x_{i+1,j}$,

Pour j = 1, pour i = 1 à 3, connecter x_{11}par x_{21}, connecter x_{21} par x_{31}, connecter x_{31} par x_{41},

Pour j = 2, pour i = 1 à 2, connecter x_{12} par x_{22}, connecter x_{22} par x_{32},

Pour j = 3, pour i = 1, connecter x_{13} par x_{23},

7. Pour j = 1 à 2, pour i = 1 à 4 – j – 1, pour z = i + 1 à 4 - j rejoindre x_{ij} par $x_{z,j+1}$,

   j = 1, i = 1, pour z = 2 à 3, rejoindre x_{11} par x_{22}, rejoindre x_{11} par x_{32},

   j = 1, i = 2, pour z = 3, rejoindre x_{21} par x_{32},

   j = 2, i = 1, pour z = 2, rejoindre x_{12} par x_{23},

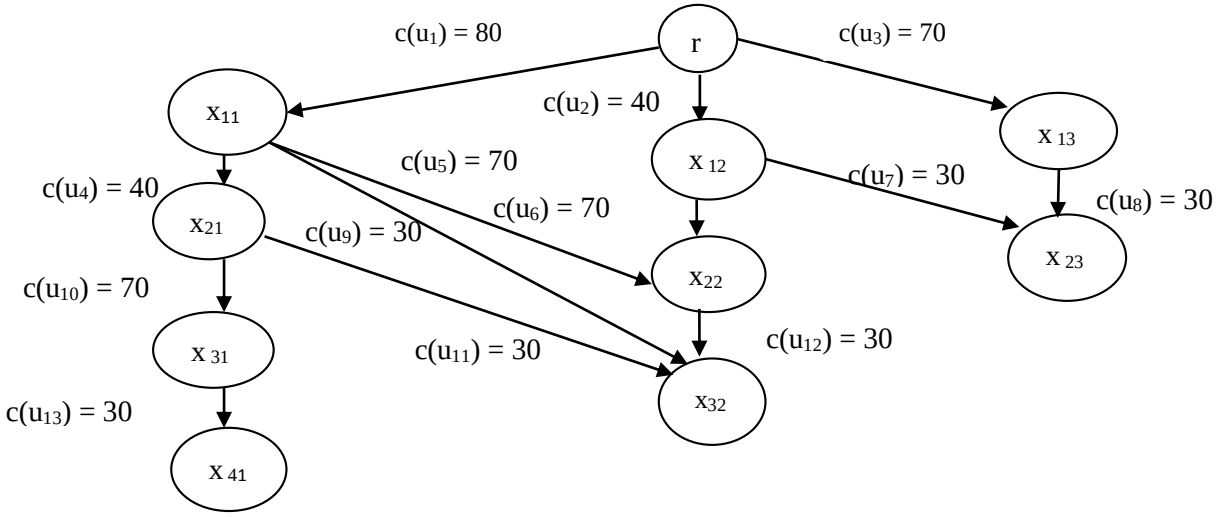

Figure B.1 : graphe de somme des composants du vecteur V

E = {r}, $E_{1j} = \emptyset$, (j = 1… 3), $c(I(u_1)) = 0$, $c(I(u_2)) = 0$, $c(I(u_3)) = 0$,

1ère itération

10. Les sommets x_{11}, x_{12}, x_{13} sont des sommets dans X – E dont les prédécesseurs sont en E, poser $E_{11} = \{x_{11}\}$, $E_{12} = \{x_{12}\}$, $E_{13} = \{x_{13}\}$,
11. $S(T(u_1)) = c(I(u_1)) + c(u_1)) = 0 + 80 = 80 < \sum_{r=1}^{3} w_r$, $S(T(u_2)) = c(I(u_2)) + c(u_2) = 0 + 40 < \sum_{r=1}^{3} w_r$, $S(T(u_3)) = c(I(u_3)) + c(u_3) = 0 + 30 = 30 < \sum_{r=1}^{3} w_r$, aller en (c).

c) Poser $c(I(u_4)) = S(T(u_1))$, $c(I(u_5)) = S(T(u_1))$, $c(I(u_9)) = S(T(u_1))$, $c(I(u_6)) = S(T(u_2))$, $c(I(u_7)) = S(T(u_2))$, $c(I(u_8)) = S(T(u_3))$, et aller en (d),

d) Poser E = {r} ∪ $\{x_{11}, x_{12}, x_{13}\}$ = $\{r, x_{11}, x_{12}, x_{13}\}$ et aller en (10),

10. Les sommets x_{21}, x_{22}, x_{23} sont des sommets dans X – E dont les prédécesseurs sont dans E, poser $E_{21} = \{x_{11}, x_{21}\}$, $E_{22} = \{x_{12}, x_{22}\}$, $E_{23} = \{x_{13}, x_{23}\}$,
11. $S(T(u_4)) = c(I(u_4)) + c(u_4) = 80 + 40 = 120$, $S(T(u_5)) = c(I(u_5)) + c(u_5) = 80 + 70 = 150$, $S(T(u_6)) = c(I(u_6)) + c(u_6) = 40 + 70 = 110$,

$S(T(u_7)) = c(I(u_7)) + c(u_7) = 40 + 30 = 70$, $S(T(u_8)) = c(I(u_8)) + c(u_8) = 70 + 30 = 100$.

a) $S(T(u_4)) \geq \sum_{r=1}^{3} w_r = 120$ donc les patrons de découpe $P_1' = (1, 0, 1)$ et $P_2' = (0, 1, 0)$ Correspond à v_1, v_2, peut être formé un plan de découpe, aller à (b),

b) $P_1'' = P_1' + P_2' = (1, 1, 1) \geq (1, 1, 1)$ alors les patrons de découpe décodés P_1, P_2 forme un plan de découpe noté $pd_1 = \begin{cases} P_1 = (1, 0, 1) \\ P_2 = (0, 2, 0) \end{cases}$,

a) $S(T(u_5)) \geq \sum_{r=1}^{3} w_r = 120$ donc les patrons de découpe $P_1' = (1, 0, 1)$ et $P_3' = (0, 1, 1)$, correspond à v_1, v_3, peut être formé un plan de découpe, aller à (b),

b) $P_2'' = P_1' + P_3' = (1, 1, 2) \geq (1, 1, 1)$ alors les patrons de découpe décodés P_1, P_3 forme un plan de découpe noté $Pd_2 = \begin{cases} P_1 = (1, 0, 1) \\ P_3 = (0, 1, 1) \end{cases}$, et aller en (c).

c) $x_{pd_1} = \begin{cases} \text{Max}\left[\frac{4}{1}, \frac{11}{1}\right] = 11 \\ \text{Max}\left[\frac{6}{2}\right] = 3 \end{cases}$, $x_{pd_2} = \begin{cases} \text{Max}\left[\frac{4}{1}, \frac{11}{2}\right] = 6 \\ \text{Max}\left[\frac{6}{1}, \frac{11}{2}\right] = 6 \end{cases}$,

$pt_{pd_1} = 80 \times (11 + 3) - (4 \times 50 + 6 \times 40 + 11 \times 30) = 3, 5\ \%$,

$pt_{pd_2} = 80 \times (6 + 6) - (4 \times 50 + 6 \times 40 + 11 \times 30) = 1, 9\ \%$,

$Pt = \text{Min}\ (pt_{pd_1}, pt_{pd_2}) = 1.\ 9\%$,

d) Poser $c(I(u_{10})) = S(T(u_4))$, $c(I(u_9)) = S(T(u_1))$, $c(I(u_{11})) = S(T(u_4))$, $c(I(u_{12}')) = S(T(u_5))$, $c(I(u_{12}'')) = S(T(u_6))$ and go to (e),

e) Poser $E = \{x_{11}, x_{12}, x_{13}\} \cup \{x_{21}, x_{22}, x_{23}\} = \{r, x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}\}$ et aller en (10),

2ème itération

10. Les sommets x_{31}, x_{32} sont des sommets dans X – E dont les prédécesseurs sont dans E, $E_{31} = \{x_{11}, x_{21}, x_{31}\}$, $E_{32} = \{x_{11}\ x_{21}, x_{32}\}$, $E_{32}' = \{x_{11}, x_{22}, x_{32}'\}$, $E_{32}'' = \{x_{12}, x_{22}, x_{32}''\}$,
11. $S(T(u_{10})) = c(I(u_{10})) + c(u_{10}) = 120 + 70 = 190$, $S(T(u_9)) = c(I(u_9)) + c(u_9) = 80 + 30 = 110$, $S(T(u_{11})) = c(I(u_{11})) + c(u_{11}) = 120 + 30 = 150$, $S(T(u_{12}')) = c(I(u_{12}')) + c(u_{12}') = 150 + 40 = 190$, $S(T(u_{12}'')) = c(I(u_{12}'')) + c(u_{12}'') = 110 + 30 = 140$.

a) $S(T(u_{10})) \geq \sum_{r=1}^{3} w_r = 120$ donc les patrons de coupe $P_1' = (1, 0, 1)$, $P_2' = (0, 1, 0)$ et $P_3' = (0, 1, 1)$, correspond à v_1, v_2, v_3, peut être formé un plan de découpe, aller à (b),

b) $P_3'' = P_1' + P_2' + P_3' = (1, 2, 2) \geq (1, 1, 1)$ alors les patrons de découpe décodés P_1, P_2 et P_3, forme un plan de découpe noté $Pd_3 = \begin{cases} P_1 = (1, 0, 1) \\ P_2 = (0, 2, 0), \\ P_3 = (0, 1, 1) \end{cases}$

a) $S(T(u_{11})) \geq \sum_{r=1}^{3} w_r = 150$, alors $P_1' = (1, 0, 1)$, $P_2' = (0, 1, 0)$ et $P_4' = (0, 0, 1)$, correspond à v_1, v_2, v_4, peut être formé un plan de découpe, aller à (b),

b) $P_4'' = P_1' + P_2' + P_4' = (1, 1, 2) \geq (1, 1, 1)$ alors les patrons de découpe décodés P_1, P_2 et P_4, forme un plan de découpe noté $Pd_4 = \begin{cases} P_1 = (1, 0, 1) \\ P_2 = (0, 2, 0) , \\ P_4 = (0, 0, 2) \end{cases}$

a) $S(T(u_{12}')) \geq \sum_{r=1}^{3} w_r = 120$ alors les patrons de découpe $P_1' = (1, 0, 1)$, $P_3' = (0, 1, 1)$ et $P_4' = (0, 0, 1)$, correspond à v_1, v_3, v_4, peut être formé un plan de découpe, aller à (b),

b) $P_5'' = P_1' + P_3' + P_4' = (1, 1, 3) \geq (1, 1, 1)$ alors les patrons de découpe décodés P_1, P_3 et P_4, forme un plan de découpe noté $Pd_5 = \begin{cases} P_1 = (1, 0, 1) \\ P_3 = (0, 1, 1), \\ P_4 = (0, 0, 2) \end{cases}$

a) $S(T(u_{12}'')) \geq \sum_{r=1}^{3} w_r = 120$ alors les patrons de découpe $P_2' = (0, 1, 0)$, $P_3' = (0, 1, 1)$ et $P_4' = (0, 0, 1)$ correspond respectivement à v_2, v_3, et v_4, peut être formé un plan de découpe, aller à (b),

b) $P_6'' = P_2' + P_3' + P_4' = (P_{21}' + P_{31}' + P_{41}', P_{22}' + P_{32}' + P_{42}', P_{23}' + P_{33}' + P_{43}') = (0, 2, 1)$ $\exists$ $P_{21}' + P_{31}' + P_{41}' = 0$, alors $\nexists$ pd_i, aller en (c),

c) $x_{pd_3} = \begin{cases} \text{Max}\left[\frac{4}{1}, \frac{11}{2}\right] = 6 \\ \text{Max}\left[\frac{6}{3}\right] = 2 \\ \text{Max}\left[\frac{6}{3}, \frac{11}{2}\right] = 6 \end{cases}$, $x_{pd_4} = \begin{cases} \text{Max}\left[\frac{4}{1}, \frac{11}{3}\right] = 4 \\ \text{Max}\left[\frac{6}{2}\right] = 3 \\ \text{Max}\left[\frac{11}{3}\right] = 4 \end{cases}$, $x_{pd_5} = \begin{cases} \text{Max}\left[\frac{4}{1}, \frac{11}{4}\right] = 4 \\ \text{Max}\left[\frac{6}{1}, \frac{11}{4}\right] = 6 , \\ \text{Max}\left[\frac{11}{4}\right] = 3 \end{cases}$

d) $pt_{pd_3} = 80 \times (6 + 6 + 2) - (7 \times 50 + 11 \times 40 + 4 \times 30) = 3, 5$ %,

$pt_{pd_4} = 80 \times (4 + 3 + 4) - (4 \times 50 + 6 \times 40 + 11 \times 30) = 1, 1$ %,

$pt_{pd_5} = 80 \times (4 + 6 + 3) - (7 \times 50 + 11 \times 40 + 4 \times 30) = 2, 7$ %,

$Pt = \text{Min}(pt_{pd_3}, pt_{pd_4}, pt_{pd_5}) = 1, 1$ % .

Arrêter parce que la valeur de $\sum_{j=1}^{3} X_j = 4 + 3 + 4 = 11$, est très proche de la valeur théorique au sens du nombre entier $\sum_{j=1}^{T} X_j = 10$, n'est donc pas une amélioration de la solution.

Alors l'ensemble des solutions efficaces présenté dans le tableau suivant :

N^o	$N°_{effs}$	$P_{per\ découpe}$	$N_{installats}$
1	Pd_4	1.1	3
2	Pd_1	1.9	2

Tableau B.1 : Ensemble de solutions efficaces.

Annexe C

L'exemple suivant représente le fonctionnement de notre méthode proposée. En résolvant le problème de découpe bidimensionnelle bi- objectifs avec coût d'installation :

Exemple C.1

• $S = L \times W = 60 \times 40 = 2400$,

• $n = 3$,

• $s_i = (l_1 \times w_1, l_2 \times w_2, l_3 \times w_3) = (50 \times 25, 25 \times 20, 20 \times 12) = (1250, 500, 400)$,

• $d = (d_1, d_2, d_3) = ((2, 3, 1)$,

1. Classer par ordre décroissant, $1250 > 500 > 400$,
2. Déterminer la première ligne de la matrice P : $p_{11} = \left\lfloor \frac{2400}{1250} \right\rfloor = 1$

a) Poser j = 1, pour i = 2 à 3 faire $p_{12} = \left\lfloor \frac{2400 - 1250 \times 1}{500} \right\rfloor = 2$, $P_{13} = \left\lfloor \frac{2400 - 1250 \times 1 - 500 \times 2}{400} \right\rfloor = 0$,

P = (1 2 0)

3. poser i = 1, h = 1, k = 1,
   a) poser j =1, d =1,
   b) $P_{11} = 1 > 0$ alors j = j + 1 = 2, l = 1,
   c) h := h + 1= 2,
   d) i = 1, $p_{21} = p_{11} - 1 = 0$,

      Pour i = 2 à 3 faire $p_{22} = \left\lfloor \frac{2400 - 1250 \times 0}{500} \right\rfloor = 4$, $p_{23} = \left\lfloor \frac{4000 - 1250 \times 2 - 3 \times 500}{400} \right\rfloor = 0$,

      $P = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 4 & 0 \end{pmatrix}$,
   e) $P_{21} = 0$, aller en (g),
   g) d := d +1= 2,
   h) i = 1< n -1 alors i: = i + 1= 2, k := k + 1 := 1 + 1= :2 et aller en (a),
   a) poser j = 1, d = 1,
   b) $P_{12} = 2 > 0$ alors j: = j + 1 = 2, l = 1,
   c) h := h + 1 = 3
   d) i > 1 alors pour z =1 à 1 faire $p_{31} = p_{11} = 1$,

      pour i = k = 2 faire $p_{32} = p_{12} - 1 = 2 - 1 = 1$,

pour i = 3 à 3 faire $p_{33} = \left\lfloor \frac{2400 - 1250\times1 - 1\times500}{400} \right\rfloor = 1$,

$$P = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 4 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

Et nous continuons à générer les patrons de coupe de cette manière comme le résultat est montré dans la matrice suivante :

$$P = \begin{pmatrix} 1 & 2 & 0 \\ 0 & 4 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 2 \\ 0 & 3 & 2 \\ 0 & 2 & 3 \\ 0 & 1 & 4 \\ 0 & 0 & 6 \end{pmatrix}$$

Tableau C.1 : Matrice des patrons de découpes réalisables

1. $\sum_{j=1}^{T} X_j = \left\lceil \frac{\sum_{i=1}^{n} s_i \times d_i}{S} \right\rceil = \left\lceil \frac{4400}{2400} \right\rceil = 2$,
2. $Pd_1 = (1, 1, 1)$ est un plan de découpe, $X_{pd_1} = \text{Max}\left(\frac{2}{1}, \frac{3}{1}, \frac{1}{1}\right) = 3$

   $Pt_1 = 2400 \times 3 - (1250 \times 2 + 500 \times 3 + 400 \times 1) = 28\%$,

3. **1ère itération :** Pour j = 1, pour i = 1 à 8 faire
   a) $P_{11} = 1$, alors $A_{11} = Ø$, $P_{21} = 0$, alors $A_{21} = \{P_2\}$, $P_{31} = 1$, alors $A_{31} = Ø$, $P_{41} = 1$, alors $A_{41} = Ø$, $P_{51} = 0$, alors $A_{51} = \{P_5\}$, $P_{61} = 0$, alors $A_{61} = \{P_6\}$, $P_{71} = 0$, alors $A_{71} = \{P_7\}$, $P_{81} = 0$, alors $A_{81} = \{P_8\}$, aller en (b),
   b) Pour i = 8, poser $B_1 = \bigcup_{i=1}^{8} A_{i1}$, $= A_{11} \cup A_{21} \cup A_{31} \cup A_{41} \cup A_{51} \cup A_{61} \cup A_{71} \cup A_{81} = Ø \cup \{P_2\} \cup Ø \cup Ø \cup \{P_5\} \cup \{P_6\} \cup \{P_7\} \cup \{P_8\} = \{P_2, P_5, P_6, P_7, P_8\}$,

2ème itération : Pour j = 2, pour i = 1 à 8 faire

   a) $P_{12} = 2$, alors $A_{12} = Ø$, $P_{22} = 4$, alors $A_{22} = Ø$, $P_{32} = 1$, alors $A_{32} = Ø$, $P_{42} = 0$, alors $A_{42} = \{P_4\}$, $P_{52} = 3$, alors $A_{52} = Ø$, $P_{62} = 2$, alors $A_{62} = Ø$, $P_{72} = 1$, alors $A_{72} = Ø$, $P_{82} = 0$, alors $A_{82} = \{P_8\}$et aller en (b),
   b) Pour i = 8, poser $B_2 = \bigcup_{i=1}^{8} A_{i2}$, $= A_{12} \cup A_{22} \cup A_{32} \cup A_{42} \cup A_{52} \cup A_{62} \cup A_{72} \cup A_{82} = Ø \cup Ø \cup \{P_4\} \cup Ø \cup Ø \cup Ø \cup Ø \cup \{P_8\} = \{P_4, P_8\}$,

3ème itération : Pour j = 3, pour i = 1 à 8 faire

a) $P_{13} = 0$, $A_{13} = \{P_1\}$, $P_{23} = 0$, $A_{23} = \{P_2\}$, $P_{33} = 1$, $A_{33} = Ø$, $P_{43} = 2$, $A_{43} = Ø$, $P_{53} = 2$, $A_{53} = Ø$, $P_{63} = 3$, $A_{63} = Ø$, $P_{73} = 4$, $A_{73} = Ø$, $P_{83} = 6$, $A_{83} = Ø$ aller en (b),

b) Pour i = 8, poser $B_3 = \bigcup_{i=1}^{8} A_{i3}$, $= A_{13} \cup A_{23} \cup A_{33} \cup A_{43} \cup A_{53} \cup A_{63} \cup A_{73} \cup A_{83} = \{P_1\} \cup \{P_2\} \cup Ø \cup Ø \cup Ø \cup Ø \cup Ø \cup Ø = \{P_1, P_2\}$,

Nous avons: $B_1 = \{P_2, P_5, P_6, P_7, P_8\}$, $B_2 = \{P_4, P_7\}$, $B_3 = \{P_1, P_2\}$,

4 ème itération : poser l = 2,

∃ $Pd_1 = (1, 1, 1)$ un plan de découpe avec 1 nombre d'installation, nous ajoutons à Pd_1, P_1 de manière à réduire $\left\lceil \frac{d_1}{P_{11}} \right\rceil$ et $\left\lceil \frac{d_2}{P_{22}} \right\rceil$. Alors $Pd_2 = (P_1, P_4)$ un plan de coupe avec 2 nombre d'installation $Pd_2 = \begin{cases} P_1 = (1, 2, 0) \\ P_3 = (1, 1, 1) \end{cases}$, nous calculons: $x_{pd_2} = \begin{cases} \text{Max} \left\lceil \frac{2}{2}, \frac{3}{3}, 0 \right\rceil = 1 \\ \text{Max} \left\lceil \frac{2}{2}, \frac{3}{3}, \frac{1}{1} \right\rceil = 1 \end{cases}$,

a) Arranger les patrons de découpe dans un ordre non décroissant P_1 à P_8

b) Pour i = 1 à 8 faire

1. Déterminer P_i appartient à des ensembles B_k, où k = 1, 2, 3
2. Pour $P_1 \in B_3$, ∃ $\{P_4\}, \{P_5\}, \{P_6\}, \{P_7\}, \{P_8\}$ sous-ensembles, dans lesquels P_4, P_5, P_6, P_7 et P_8, n'appartient pas à B_3, Ainsi, le P_4, P_5, P_6, P_7 et P_8 ne formes pas de plans de découpe à l'étape précédente $Pd_3 = \begin{cases} P_1 \\ P_4 \end{cases}$, $Pd_4 = \begin{cases} P_1 \\ P_5 \end{cases}$, $Pd_5 = \begin{cases} P_1 \\ P_6 \end{cases}$, $Pd_6 = \begin{cases} P_1 \\ P_7 \end{cases}$, $Pd_7 = \begin{cases} P_1 \\ P_8 \end{cases}$, calculer: $x_{pd_3} = \begin{cases} \text{Max} \left\lceil \frac{2}{2}, \frac{3}{2}, 0 \right\rceil = 2 \\ \text{Max} \left\lceil \frac{2}{2}, 0, \frac{1}{2} \right\rceil = 1 \end{cases}$, $x_{pd_4} = \begin{cases} \text{Max} \left\lceil \frac{2}{1}, \frac{3}{5}, 0 \right\rceil = 2 \\ \text{Max} \left\lceil 0, \frac{3}{5}, \frac{1}{2} \right\rceil = 1 \end{cases}$ x_{pd_6}, x_{pd_7},

Nous représentons toutes les solutions réalisables avec deux nombres d'installations dans le tableau suivant :

N°	F_{solu}	x_j	$P_{surf\ perd}$	$N_{insttals}$	N°	F_{solu}	x_j	$P_{surf\ perd}$	$N_{installs}$
1	$P_1 = (1, 2, 0)$ $P_3 = (1, 1, 1)$	$\begin{cases} 1 \\ 1 \end{cases}$	4.00	2	3	$P_4 = (1, 0, 2)$ $P_6 = (0, 2, 3)$	$\begin{cases} 2 \\ 2 \end{cases}$	52.00	2
2	$P_1 = (1, 2, 0)$ $P_4 = (1, 0, 2)$	$\begin{cases} 2 \\ 1 \end{cases}$	28.00	2	4	$P_4 = (1, 0, 2)$ $P_7 = (0, 1, 4)$	$\begin{cases} 2 \\ 3 \end{cases}$	76.00	2

Tableau C.2 : Ensemble de solutions réalisables pour deux nombres d'installations.

$\text{Min}(x_{pd}) = \text{Min}(x_{pd_2}, x_{pd_3}, x_{pd_4}, x_{pd_5}, x_{pd_6}, x_{pd_7}, x_{pd_8}, x_{pd_9}, x_{pd_{10}}) = x_{pd_2}$.

Arrêtez, car la valeur théorique de $\left\lceil \sum_{j=1}^{T} X_j \right\rceil = \left\lceil \frac{4400}{2400} \right\rceil = 2$, alors $\text{Min}(x_{pd}) = x_{pd_2} = 2$,

c) $Pt = 2400 \times (1 + 1) - 4400 = 4\ \%$,

Ainsi toutes les solutions efficaces sont présentées dans le tableau suivant :

N°	$N°_{effs}$	$P_{surf\ perd}$	$N_{insttals}$
1	Pd_2	4.00	2
2	Pd_1	28.00	1

Tableau C.3: Ensemble de solutions efficaces