UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de São José do Rio Preto

Nícolas Samuel Assis

O problema de corte de estoque bidimensional:
geração de padrões de corte 2-estágios restritos

São José do Rio Preto
2019

Nícolas Samuel Assis

O problema de corte de estoque bidimensional:

geração de padrões de corte 2-estágios restritos

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Matemática, junto ao Programa de Pós-Graduação em Matemática, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Câmpus de São José do Rio Preto.

Financiadora: CAPES – Proc. 88882.434318/2019-01

Orientador: Profª. Drª. Maria do Socorro Nogueira Rangel

São José do Rio Preto
2019

A848p

Assis, Nícolas Samuel

O problema de corte de estoque bidimensional : geração de padrões de corte 2-estágios restritos / Nícolas Samuel Assis. -- São José do Rio Preto, 2019

96 f.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto

Orientadora: Maria do Socorro Nogueira Rangel

1. Matemática aplicada. 2. Problema da mochila limitada. 3. Problema de corte bidimensional guilhotinado 2-estágios restrito. 4. Problema de corte de estoque bidimensional. 5. Programação dinâmica. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Nícolas Samuel Assis

O problema de corte de estoque bidimensional:
geração de padrões de corte 2-estágios restritos

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Matemática, junto ao Programa de Pós-Graduação em Matemática, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Câmpus de São José do Rio Preto.

Financiadora: CAPES – Proc. 88882.434318/2019-01

Comissão Examinadora

Profª. Drª. Maria do Socorro Nogueira Rangel
UNESP – Câmpus de São José do Rio Preto
Orientadora

Prof. Dr. Valeriano Antunes de Oliveira
UNESP – Câmpus de São José do Rio Preto

Prof. Dr. Flávio Molina da Silva
UFTM – Universidade Federal do Triângulo Mineiro

São José do Rio Preto
22 de novembro de 2019

Aos meus pais.

AGRADECIMENTOS

Agradeço aos meus pais por todo apoio que me deram e por me acalmarem nos momentos difíceis.

Agradeço especialmente à minha orientadora Profa. Dra. Socorro Rangel, por ter acreditado no meu trabalho e por me acalmar quando chegava perdido nas reuniões. Agradeço pelos ensinamentos que vão além dos acadêmicos e por ter lutado junto comigo até o fim.

Agradeço ao Prof. Dr. Valeriano Antunes por aceitar compor a banca de defesa e pelas correções feitas ao trabalho.

Agradeço ao Prof. Dr. Flávio Molina por se dispor a sair da sua cidade para compor a banca de defesa e pelas correções feitas ao trabalho.

Agradeço à minha namorada, pela paciência, pelo incentivo e pela ajuda nas etapas finais.

Agradeço aos meus amigos, colegas e professores por todo apoio, ensinamento e experiências compartilhadas.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, à qual agradeço.

RESUMO

Nessa dissertação é feito uma revisão das características gerais dos problemas de corte e empacotamento e apresentam-se duas tipologias encontradas da literatura para classificar os problemas. São estudados em detalhes três problemas: (i) o problema da mochila limitada, (ii) o problema de corte bidimensional guilhotinado 2-estágios restrito, e (iii) o problema do corte de estoque bidimensional. Para o problema (i) é proposto um algoritmo de programação dinâmica adaptado de um algoritmo proposto na literatura. Esse algoritmo é a base para a proposta de duas estratégias para resolver o problema (ii). Os algoritmos desenvolvidos para o problema (ii) são então usados no processo de geração de colunas usado para resolver o problema de corte de estoque exato. Resultados de um estudo computacional realizado para avaliar o desempenho dos algoritmos propostos usando instâncias da literatura são apresentados e analisados.

Palavras–chave: Problema da mochila limitada. Problema de corte bidimensional guilhotinado 2-estágios restrito. Problema de corte de estoque bidimensional. Programação dinâmica. Heurística.

ABSTRACT

In this dissertation a review of the main characteristics of the Cutting and Packing problems are presented together with a summary of two typologies proposed in the literature to classify the problems. Three problems are studied in detail: (i) the Bounded Knapsack problem, (ii) the constraint two-dimensional guillotine 2-stage cutting problem, and (iii) the two-dimensional cutting stock problem. For problem (i) we propose a dynamic programming algorithm adapted from one given in the literature. This algorithm is the basis for the proposal of two strategies to solve the problem (ii). The algorithms developed for problem (ii) are then used in the column generation process employed to solve the exact cutting stock problem. Results of a computational study conducted to evaluate the performance of the proposed algorithms using instances of the literature are presented and analyzed.

Keywords: Bounded knapsack problem. Constraint two-dimensional guillotine 2-stage cutting problem. Two-dimensional cutting stock problem. Dynamic programming. Heuristics.

Lista de Figuras

Lista de Tabelas

Sumário

INTRODUÇÃO

O problema de corte de estoque consiste em cortar objetos grandes em estoque para atender uma demanda de itens menores com o objetivo de utilizar o menor número de objetos. Esse é um problema clássico da Pesquisa Operacional e que está dentro da classe dos problemas de corte e empacotamento. Ele pode ser encontrado em diversos cenários reais como na indústria de papel, vidro e móveis (Cherri e Arenales 2005). Na indústria moveleira, esse problema é resolvido por um responsável da produção (gerente e/ou engenheiro) antes do processo de produção. Utilizando a solução do problema, o operário pode cortar as placas de madeira grandes em placas menores para fazer a produção de mesas, armários etc. (Rangel e Figueiredo 2008). Nesse exemplo, o objeto cortado é tridimensional, no entanto, para o problema somente o comprimento e largura são relevantes. Com isso, esse problema pode ter o caso uni, bi e tridimensional. Além disso, a forma com que um objeto é cortado chama-se padrão de corte. Por mais que esse seja um problema simples de entender, ele possui uma natureza combinatória, apresentando inúmeras soluções possíveis (Arenales et al. 2015). O desafio é como encontrar a solução ótima dentro desse grande espaço de busca.

Uma primeira pergunta a se fazer antes de buscar formas para resolver o problema é: qual é sua relevância? Para o dono de uma indústria a resposta seria simples, quanto menos objetos são cortados para atendar a demanda, menores são os gastos, e por sua vez, maiores são os lucros. Com isso, trabalhos como o de Morabito e Arenales 2000 e Rangel e Figueiredo 2008 fazem um estudo do problema de corte de estoque voltado à indústria moveleira, de forma a melhorar o processo de produção da fábrica. Há também trabalhos que auxiliam nos imprevistos presentes na fábrica, como objetos com defeitos (Martin et al. 2019) e sobras não utilizáveis (Cherri e Arenales 2005, Andrade, Birgin e Morabito 2016).

Dado esse panorama, é evidente que existem empresas e grupos de consultoria que desenvolvem ferramentas e/ou dão soluções para que as insdútrias possam resolver seus problemas (e.g. CorteCerto 2017). Um programa que resolve o problema de corte de estoque bidimensional é o sitema CorteBiFur. Esse programa foi

desenvolvido por Perin e Rangel 1989 e durante os anos contou com diversos colaboradores para melhorar o programa (Lemos 2005, Rangel et al. 2019). O sistema CorteBiFur foi um dos motivadores dessa pesquisa.

Com isso, essa pesquisa tem como objetivo resolver o problema de corte de estoque bidimensional, obtendo padrões de corte bidimensionais guilhotinados (ortogonal) 2-estágios restrito. O corte guilhotinado ortogonal é um tipo de corte que é feito paralelo a algum lado do objeto e vai de ponta a ponta. O 2-estágios está relacionado em quantas rotações tem que ser feitas no objeto para se obter todos os itens, e o restrito, limita a quantidade desses itens no padrão de corte. Para alcançar esse objetivo, foi feito o estudo de mais dois problemas de corte e empacotamento: o problema de corte bidimensional guilhotinado 2-estágios restrito (PC) e o problema da mochila limitada (PML). Durante essa trajetória, a pesquisa teve como contribuição o desenvolvimento de um algoritmo de programação dinâmica (PML^{PD}) que resolve o problema da mochila limitada e dois algoritmos (PC^{PD}_{CPLEX} e PC^{PD}_{H}) que resolvem o problema PC. Também, esses dois algoritmos foram utilizados para resolver o problema de corte de estoque por meio do processo de geração de colunas de Gilmore e Gomory 1961.

Essa dissertação tem a seguinte organização. No Capítulo 1 é feito um estudo dos aspectos gerais dos problemas de corte e empacotamento e é visto duas tipologias para essa classe de problemas. No Capítulo 2 estuda-se o problema da mochila. Inicialmente vendo variações do problema e como a programação dinâmica pode ser utilizada para resolvê-lo. Em seguida, é feito uma adaptação do algoritmo de Perin e Rangel 1989 para o problema da mochila limitada propondo o algoritmo PML^{PD}. No Capítulo 3 é visto o problema de corte bidimensional, são definidas restrições do equipamento de corte e apresenta-se como a programação dinâmica pode ser utilizada nesse problema. Após isso, é dado o desenvolvimento dos algoritmos PC^{PD}_{CPLEX} e PC^{PD}_{H} para gerar padrões de corte bidimensionais 2-estágios restritos. No Capítulo 4 é feito uma breve revisão dos problemas de corte de estoque e explica-se como os algoritmos PC^{PD}_{CPLEX} e PC^{PD}_{H} são utilizados no processo de geração de colunas para resolver o problema de corte de estoque. No Capítulo 5 é realizado um estudo computacional dos três problemas separadamente. Por último, são feitas algumas considerações finais a respeito da pesquisa e são apresentadas propostas futuras.

Captulo 1

CARACTERÍSTICAS GERAIS DOS PROBLEMAS DE CORTE E EMPACOTAMENTO

Nesse capítulo apresentam-se aspectos gerais dos problemas de corte e empacotamento (PCE). Para isso, será introduzido de forma bem ampla alguns parâmetros para que seja possível caracterizar a essência dos problemas de corte e empacotamento conforme proposto em Scheithauer 2017. Assim, considere $O_i \subset \mathbb{R}^d_+, i \in \mathscr{I}$ como sendo um conjunto de itens e $S \subset \mathbb{R}^d_+$ uma região do espaço euclidiano d-dimensional. Também, considere que S é fechado e limitado, assim como os itens O_i para todo $i \in \mathscr{I}$. Além disso, o fecho do interior do item O_i é igual à ele mesmo ($\partial(int(O_i)) = O_i$). Isso passa a ideia do item ser totalmente preenchido, ou seja, não é vazio por dentro. O problema PCE consiste em alocar (ou cortar) os itens na região S de acordo com algum critério pré-determinado. Para formular matematicamente o PCE, considere que cada item O_i está em uma posição normalizada que é a origem $O_i(0)$. Para alocar (ou cortar) os itens na região S é necessário a translação desses itens. Podemos enxergar essa translação como sendo a retirada do item O_i da origem e reposicionando-o na posição u_i ((1.1)). Para alguns problemas é permitido a rotação dos itens, então em (1.2) é dado a rotação do item $O_i(u_i)$, com D_{θ_i} uma rotação qualquer de ângulo θ_i (Scheithauer, 2017).

$$O_i(u_i) := u_i + O_i = \left\{ v \in \mathbb{R}^d \mid v = u_i + u, u \in O_i(0) \right\} \tag{1.1}$$

$$D(O_i(u_i), \theta_i) := \left\{ v \in \mathbb{R}^d \mid v = u_i + D_{\theta_i} u, u \in O_i(0) \right\} \tag{1.2}$$

Duas condições básicas devem ser satisfeitas na solução dos problemas de corte e empacotamento: o objeto estar inteiramente contido na região S ((1.3)) e a de não-sobreposição dos itens ((1.4)). Sendo $\mathscr{I}^*$ o conjunto de itens escolhidos para cortar ou empacotar em S, em (1.3) garante-se que dado qualquer rotação, pelo ângulo de θ_i, do item O_i posicionado em u_i, este item estará totalmente contido na região S. Em (1.4) exige-se que o interior de qualquer item (rotacionado e deslocado) não pode ocupar a mesma região que outro item diferente. Perceba que essa última permite que os itens tenham uma região de fronteira comum.

$$D(O_i(u_i),\theta_i) \subseteq S,\ \forall i \in \mathscr{I}^* \tag{1.3}$$

$$int(D(O_i(u_i),\theta_i)) \cap D(O_k(u_k),\theta_k) = \emptyset,\ \forall i \neq k,\ i,k \in \mathscr{I}^* \tag{1.4}$$

Com essas duas restrições, é possível estabelecer dois problemas teóricos e seus modelos, que caracterizam problemas clássicos de corte e empacotamento, sendo três deles abordados nessa pesquisa. O primeiro problema teórico pode ser enxergado como maximizar a utilização da região S, também chamada de objeto. Então, dado um valor v_i associado a cada item, a ideia é escolher um subconjunto de itens $\mathscr{I}^*$ ((1.8)), de forma a maximizar o valor total obtido por essa escolha ((1.5)), satisfazendo as restrições básicas de contenção e de não-sobreposição ((1.6) - (1.7)). O segundo problema pode ser visto como minimizar o uso de regiões. Dessa forma, se tem um conjunto $\mathscr{J}$ de regiões e um custo c_j associado a cada uma delas. O objetivo é alocar ou cortar todos os itens ((1.12)), de forma a minimizar o custo total ((1.9)) do uso das regiões, dentre as regiões existentes ((1.13)), satisfazendo as restrições básicas em cada uma das regiões utilizadas ((1.10) - (1.11)).

Modelo teórico de maximização da utilização do material.

$$Max \sum_{i \in \mathscr{I}^*} v_i \tag{1.5}$$

$$S.a:\ D(O_i(u_i),\theta_i) \subseteq S,\ \forall i \in \mathscr{I}^* \tag{1.6}$$

$$int(D(O_i(u_i),\theta_i)) \cap D(O_k(u_k),\theta_k) = \emptyset,\ \forall i \neq k,\ i,k \in \mathscr{I}^* \tag{1.7}$$

$$\mathscr{I}^* \subseteq \mathscr{I} \tag{1.8}$$

Modelo teórico de minimização do uso de material.

$$Min \sum_{j \in \mathscr{J}^*} c_j \tag{1.9}$$

$$S.a: \ D(O_i(u_i), \theta_i) \subseteq S_j, \ \forall i \in \mathscr{I}_j \ \forall j \in \mathscr{J}^* \tag{1.10}$$

$$int(D(O_i(u_i), \theta_i)) \cap D(O_k(u_k), \theta_k) = \emptyset, \ \forall i \neq k, \ i,k \in \mathscr{I}_j, \ \forall j \in \mathscr{J}^* \tag{1.11}$$

$$\bigcup_{i \in \mathscr{J}^*} \mathscr{I}_j = \mathscr{I} \tag{1.12}$$

$$\mathscr{I}_j \subseteq \mathscr{I} \ \forall j \in \mathscr{J}^*, \ \ \mathscr{J}^* \subseteq \mathscr{J} \tag{1.13}$$

A partir desses dois modelos teóricos, é possível formular inúmeros problemas de corte e empacotamento, sendo eles, muito ou pouco distintos. Com isso, existem trabalhos que têm como objetivo classificar os diferentes problemas de corte e empacotamento, para facilitar a comunicação entre a comunidade científica, e também, para organizar os resultados teóricos para cada um desses problemas. Nessa pesquisa, apresenta-se na Seção 1.1 dois trabalhos desse tipo, o primeiro foi proposto por Dyckhoff 1990 e o segundo por Wäscher, Haußner e Schumann 2007.

1.1 Tipologia

Para fazer a organização e padronização dos problema de corte e empacotamento são apresentadas duas tipologias, a de Dyckhoff 1990 e a de Wäscher, Haußner e Schumann 2007. Essas tipologias são apresentadas em paralelo para facilitar a comparação e também verificar as diferenças entre as duas. Na tipologia de Dyckhoff 1990 o autor utiliza de quatro critérios para classificar os problemas de corte e empacotamento: *dimensionalidade, tipo de tarefa, variedade dos objetos e variedade dos itens*. Esses critérios foram obtidos pela sistematização das principais características dos problemas encontrados na literatura da época. Por sua vez, cada critério possui uma subdivisão que é utilizada para classificar os problemas.

Na tipologia de Dyckhoff 1990 é possível classificar os problemas de forma geral, contemplando mais os clássicos e não sendo possível classificar suas variações, tornando ela parcialmente ineficiente e imprecisa (Wäscher, Haußner e Schumann 2007). Então, utilizando a tipologia de Dyckhoff 1990, a tipologia de Wäscher, Haußner e Schumann 2007 foi proposta como forma de completar a primeira e classificar novos problema que surgiram nos anos seguintes. Na tipologia de Wäscher, Haußner e Schumann 2007, os autores utilizam cinco critérios para organizar os problemas

de corte e empacotamento: *dimensionalidade, tipo de tarefa, variedade dos objetos, variedade dos itens e forma dos itens*. Assim como na primeira tipologia, cada critério de classificação possui suas subdivisões. Após explicar todos os critérios de classificação, mostra-se como é feito a classificação dos problemas de corte e empacotamento segundo a tipologia de Wäscher, Haußner e Schumann 2007.

Na sequência é apresentado cada um dos critérios de classificação de ambas as tipologias e é feito a discussão das diferenças entre as duas. Sempre na esquerda estará a tipologia de Dyckhoff 1990 e na direita a tipologia de Wäscher, Haußner e Schumann 2007.

O critério de *dimensionalidade*, se refere às dimensões relevantes para o problema: unidimensional, bidimensional etc. Na tipologia de Dyckhoff 1990 é considerado até os problemas N-dimensionais com $N > 3$. Já na tipologia de Wäscher, Haußner e Schumann 2007, os autores consideram os problemas N-dimensionais com $N > 3$ como outra variante, e então, não especificado em seu trabalho.

- Dimensionalidade:
  - (1) Unidimensional;
  - (2) Bidimensional;
  - (3) Tridimensional;
  - (N) N-dimensional com $N > 3$.

I Dimensionalidade:

- Unidimensional;
- Bidimensional;
- Tridimensional.

O critério *tipo de tarefa* está associado ao objetivo do problema. Considerando os dois problemas teóricos tratados até agora e olhando para a tipologia de Dyckhoff 1990, o problema (1.5)-(1.8) seria classificado como (B), pois procura maximizar a utilização do objeto por meio de uma seleção de itens, e o problema (1.9)-(1.13) como (V), porque minimiza a quantidade de objetos para obter/alocar todos os itens. Na tipologia de Wäscher, Haußner e Schumann 2007 a classificação é a mesma, mudando apenas as palavras utilizadas, sendo a maximização dos resultados igual a (B) e a minimização das entradas igual a (V). É importante dizer que esse critério não está diretamente associado à maximizar ou minimizar, mas sim à natureza da tarefa.

- Tipo de tarefa:
  - (B) Todos os objetos e uma seleção de itens;
  - (V) Uma seleção de objetos e todos os itens.

II Tipo de tarefa:

- Maximização dos resultados;
- Minimização das entradas.

No critério *variedade dos objetos*, a tipologia de Dyckhoff 1990 diferencia os objetos em apenas um no problema (O), objetos idênticos (I) e objetos diferentes (D). Na tipologia de Wäscher, Haußner e Schumann 2007 os autores abrem a classificação de um objeto em dois casos: todas as dimensões fixadas e uma ou mais dimensões variáveis. O segundo caso já mostra uma extensão da classificação dos problemas de corte empacotamento, sendo possível classificar os problemas de empacotamento em faixas por exemplo. Quando é considerado mais de um objeto na tipologia de Wäscher, Haußner e Schumann 2007, os autores diferenciam eles em objetos idênticos, fracamente heterogêneos e fortemente heterogêneos. Então, a tipologia de Wäscher, Haußner e Schumann 2007 abre os objetos diferentes (D) da tipologia de Dyckhoff 1990 nos casos fracamente e fortemente heterogêneos, diferenciando ainda mais os problemas.

III Variedade dos objetos:

- Um objeto:
  - Todas as dimensões fixadas;
  - Uma ou mais dimensões variáveis.
- Vários objetos (Todas as dimensões fixadas):
  - Objetos idênticos;
  - Objetos fracamente heterogêneos;
  - Objetos fortemente heterogêneos.

- Variedade dos objetos:
  - (O) Um objeto;
  - (I) Objetos idênticos;
  - (D) Objetos diferentes.

O critério de *variedade dos itens* na tipologia de Dyckhoff 1990 há uma mistura da quantidade dos itens com a forma desses itens. Na tipologia de Wäscher, Haußner e Schumann 2007, não é considerado a quantidade dos itens, somente a característica do conjunto de itens, sendo idênticos, fracamente ou fortemente heterogêneos. Na tipologia de Dyckhoff 1990, as classificações (F) e (M) não transmitem totalmente a característica do conjunto de itens, pois não dá para saber se eles são muito ou pouco distintos. Nesse sentido a tipologia de Wäscher, Haußner e Schumann 2007 organiza melhor esse critério. Além disso, na tipologia de Wäscher, Haußner e Schumann 2007 houve o acréscimo do critério *forma dos itens* que é utilizado quando os problemas possuem dimensão maior do que um. Esse critério serve para organizar ainda mais os problemas de corte e empacotamento, porque a forma dos itens interferem diretamente em quantos parâmetros serão necessários para representar cada item do problema.

- Variedade dos itens:
  - (F) Poucos itens (de formas diferentes);
  - (M) Muitos itens de muitas formas diferentes;
  - (R) Muitos itens de relativamente poucas formas diferentes (não congruentes);
  - (C) Itens congruentes.

IV Variedade dos itens:
- Itens idênticos;
- Itens fracamente heterogêneos;
- Itens fortemente heterogêneos.

V Forma dos itens:
- Regular:
  - Retangular;
  - Circular;
  - ⋮
- Irregular

Com isso, para fazer a classificação de um problema de corte e empacotamento segundo a tipologia de Dyckhoff 1990, basta utilizar a abreviação de cada uma das subdivisões dos critérios de classificação. Por exemplo o problema da mochila inteira, que será discutido nessa pesquisa, classifica-se como 1/B/O/F ou 1/B/O/M pois o problema pode ter pouco ou muitos itens. Segundo a tipologia de Wäscher, Haußner

e Schumann 2007, esse problema é classificado de apenas uma forma: 1-dimensional problema de posicionamento de objeto único. A forma de classificar o problema segundo essa tipologia será explicado a seguir.

Segundo os critérios de classificação listados, Wäscher, Haußner e Schumann 2007 caracterizam os problemas de corte e empacotamento em três tipos: básicos (II e IV), intermediários (III) e refinados (I e V); que são combinados para gerar a classificação final dos problemas de corte e empacotamento. Nas Figuras 1.1 e 1.2 apresentam-se a nomenclatura dos problemas intermediários, que são dados ao combinar os tipos básicos ao critério (III). Para obter a classificação final dos problemas, basta juntar os critérios (I e V) e seguir a regra: (1, 2, 3)-*dimensional* (*retangular, circular*, ..., irregular) (*Tipo de problema intermediário*).

Figura 1.1: Tipos de problemas intermediários: maximização dos resultados.

characteristics of the large objects \ assortment of the small items		identical	weakly heterogeneous	strongly heterogeneous
all dimensions fixed	one large object	Identical Item Packing Problem IIPP	Single Large Object Placement Problem SLOPP	Single Knapsack Problem SKP
	identical		Multiple Identical Large Object Placement Problem MILOPP	Multiple Identical Knapsack Problem MIKP
	heterogeneous		Multiple Heterogeneous Large Object Placement Problem MHLOPP	Multiple Heterogeneous Knapsack Problem MHKP

Fonte: adaptada de Wäscher, Haußner e Schumann 2007.

Figura 1.2: Tipos de problemas intermediários: minimização das entradas.

characteristics of large objects \ assortment of small items		weakly heterogeneous	strongly heterogeneous
all dimensions fixed	identical	Single Stock Size Cutting Stock Problem SSSCSP	Single Bin Size Bin Packing Problem SBSBPP
	weakly heterogeneous	Multiple Stock Size Cutting Stock Problem MSSCSP	Multiple Bin Size Bin Packing Problem MBSBPP
	strongly heterogeneous	Residual Cutting StockProblem RCSP	Residual Bin Packing Problem RBPP
one large object variable dimension(s)		Open Dimension Problem ODP	

Fonte: adaptada de Wäscher, Haußner e Schumann 2007.

Nessa pesquisa, será utilizada a classificação proposta por Wäscher, Haußner e Schumann 2007, por ser mais completa, no entanto, mesmo com as modificações propostas essa tipologia não é suficiente para caracterizar completamente os problemas abordados na pesquisa, então os problemas serão classificados como (1, 2, 3)-*dimensional* (*retangular, circular*, …, irregular) (*Tipo de problema intermediário*) (*Complementos*), no qual, os complementos são as características não previstas na tipologia (*e. g.* Martin et al. 2019).

Dentre os problemas de corte e empacotamento, existem três que são mais pertinentes à essa pesquisa: o problema da mochila limitada, o problema de corte

bidimensional guilhotinado 2-estágios restrito e o problema de corte de estoque bidimensional. Eles são classificados respectivamente como *1-dimensional problema de posicionamento de objeto único (limitado), 2-dimensional retangular problema de posicionamento de objeto único (2-estágios restrito)* e *2-dimensional retangular problema de corte de estoque de tamanho único*. Esses problemas são detalhados nos capítulos seguintes.

Captulo 2

O PROBLEMA DA MOCHILA

O problema da mochila (PM) pode ser definido da seguinte maneira. Considere um conjunto de itens $\mathscr{I}$, em que cada item possui um valor v_i e peso p_i, e uma mochila de capacidade C. O PM consiste em alocar um subconjunto de itens na mochila ($\mathscr{I}^*$) considerando a maximização do valor total dos itens alocados. Supondo que os itens serão alocados um em seguida do outro, o modelo geral (1.5)-(1.8) pode ser adaptado para representar o PM. Nessa formulação, a região em que os itens serão alocados é a mochila e seu parâmetro relevante para o problema é sua capacidade C. Para os itens, o parâmetro relevante em relação as restrições é o peso p_i de cada item e para a função objetivo o valor v_i. Dessa forma, o modelo geral pode ser modificado para (2.1)-(2.3).

$$Max \sum_{i \in \mathscr{I}^*} v_i \tag{2.1}$$

$$S.a: \sum_{i \in \mathscr{I}^*} p_i \leq C \tag{2.2}$$

$$\mathscr{I}^* \subseteq \mathscr{I} \tag{2.3}$$

Em (2.2) está incorporado as duas restrições básicas dos problemas de corte e empacotamento, pois a alocação dos itens respeitam os limites da região sem sobreposição. Essa formulação é equivalente à conhecida como problema da mochila binário ou $0-1$, pois a decisão tomada é colocar ou não um item na mochila. Segundo a tipologia de Wäscher, Haußner e Schumann 2007, o problema classifica-se como *1-dimensional problema da mochila única*. A formulação que será explorada nessa pesquisa é o do problema da mochila inteira (PMI) que pode ser formulado de acordo com ((2.4)-(2.7)). O PMI caracteriza-se como limitada (PML) ou ilimitada (PMI-

I). Nesse caso é necessário incluir a variável de decisão y_i que determina a quantidade de cada item i que será alocado na mochila e as restrições (2.6)-(2.7) que determinam o intervalo de variação da variável y_i e seu domínio inteiro não-negativo. Pela restrição (2.6), se para algum $i = 1 \ldots m$, $b_i < \lfloor \frac{C}{p_i} \rfloor$, então o problema é limitado, caso contrário, ele caracteriza-se como ilimitado. Segundo a tipologia de Wäscher, Haußner e Schumann 2007, esse problema se classifica como *1-dimensional problema de posicionamento de objeto único (limitado)*.

$$Max \sum_{i=1}^{m} v_i \cdot y_i \tag{2.4}$$

$$S.a: \sum_{i=1}^{m} p_i \cdot y_i \leq C \tag{2.5}$$

$$0 \leq y_i \leq b_i, \; i = 1 \ldots m \tag{2.6}$$

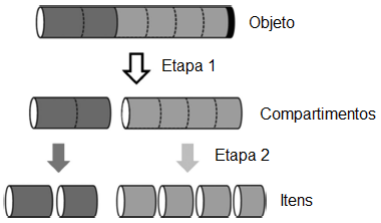
$$y_i \in \mathbb{Z}_+, i = 1 \ldots m \tag{2.7}$$

Esse problema é de simples enunciado e modelagem matemática, porém, ele é de grande importância por ter diversas aplicações práticas como: corte de bobinas de papel, corte de cano de aço etc., e por aparecer em subproblemas de outros problemas, como será tratado nessa pesquisa. Na literatura são abordados algumas de suas variações (Inarejos, Hoto e Maculan 2019, Quiroga-Orozco, Carvalho e Hoto 2019, Malaguti et al. 2019 e Zhou et al. 2019).

2.1 Variação do Problema da Mochila

No trabalho de Inarejos, Hoto e Maculan 2019 e Quiroga-Orozco, Carvalho e Hoto 2019 encontram-se o caso da mochila compartimentada que é uma variação da mochila inteira e define-se da seguinte forma: é dado um conjunto de itens que é classificado em conjuntos disjuntos por algum critério e o objetivo é inserir os itens na mochila sem que as classes se misturem. Ou seja, o item i só poderá estar ao lado do item j se forem da mesma classe ou estiverem na fronteira de seus compartimentos. Esse problema surgiu na indústria metalúrgica no corte de bobinas de aço em duas etapas, na primeira a bobina maior é subdividida de forma a isolar cada compartimento, que é determinado pela espessura da bobina final, e na segunda etapa cada compartimento é cortado conforme a espessura da classe obtendo os itens, conforme é ilustrado na Figura 2.1.

Figura 2.1: Corte de bobina de aço em duas etapas.



Fonte: adaptada de Inarejos, Hoto e Maculan 2019.

Outro tipo de problema da mochila é visto no trabalho de Malaguti et al. 2019 que trata de otimização inteira com penalização de valores fracionários, abordando o caso da mochila binária. Nessa proposta, a variável de decisão é real e limitada entre zero e um, $0 \leq y_i \leq 1, y_i \in \mathbb{R}, i = 1 \ldots m$, e a função objetivo passa a ser o somatório de $F_i(y_i)$ uma função definida por partes, no qual, $F_i(y_i) = 0$ se $y_i = 0$, $F_i(y_i) = v_i$ se $y_i = 1$ e $F_i(y_i) = v_i - f_i(y_i)$ se $0 < y_i < 1, i = 1 \ldots m$ com f_i sendo a função de penalização, caso seja usado uma fração do item. No trabalho, os autores dizem que a relaxação da mochila binária é uma particularização da proposta feita por eles quando $f_i(y_i) = 0$ para qualquer valor dos y_i.

Nesses três trabalhos, é abordado o problema da mochila unidimensional, mas também existem trabalhos que tratam o PM em mais dimensões, como o caso encontrado no trabalho de Zhou et al. 2019. Os autores tratam da mochila bidimensional em setores, todos os itens são retangulares e são alocados em setores. Esse é um problema similar à mochila compartimentada, pois os itens são classificados e sua alocação só pode ser feita na região (setores disjuntos) de sua classe, podendo ela ser não retangular. A diferença entre esse problema e o da mochila compartimentada é que nesse, os setores são pré-determinados, enquanto na compartimentada os compartimentos são definidos quando os itens são alocados, ou seja, durante o processo. Esse foi um problema que surgiu de uma empresa agrícola que precisa alocar experimentos com sementes num campo de teste para avaliar o desempenho das sementes.

Nas pesquisas descritas, é possível ter uma visão geral da importância do problema da mochila, na atualidade, e como ele vem se reinventando e aparecendo em diferentes cenários. Além desses problemas recentes, existem trabalhos tidos como clássicos, um deles é o trabalho de Martello e Toth 1990. Os autores fizeram um livro que possui resultados do problema da mochila $0-1$, limitada e outras variações. Nele apresenta-se métodos de resolução exata como *branch-and-bound* e algoritmos

de programação dinâmica, assim como técnicas que diminuem o espaço de busca do problema da mochila (relações de dominância).

2.2 Soluções via Programação Dinâmica

A programação dinâmica é uma técnica utilizada para a resolução de problemas, suas principais características são: estágios, estados e decisões. Dado um problema de otimização (original), ele será decomposto/formulado em estágios e cada estágio possui estados. Cada combinação estágio/estado, a grosso modo, é um problema menor e mais simples. Formulado o problema, sua solução é obtida através de uma sequência de decisões ótimas que irão resolver os estados, um a um, até a tomada de decisão final, no último estado do último estágio (Arenales et al. 2015).

Considere a forma compacta do problema da mochila inteiro ((2.8)) no qual, v e p são os vetores m-dimensionais do valor e peso dos itens, respectivamente, e y é a variável de decisão também m-dimensional. Uma possível formulação do PMI seria utilizar os estágios (k) como o número de itens considerados ($1 \leq k \leq m$) e os estados (c) a capacidade considerada da mochila ($0 \leq c \leq C$) ((2.9)). Uma política ótima a ser tomada pode ser, partindo de $k = 1 \ldots m$ e para cada k, $c = 0 \ldots C$, decidir se é melhor (obtém valor total máximo) inserir o item k na mochila de capacidade c ou carregar seu estado presente para o próximo estado. Perceba que com isso, um problema da mochila se tornou $m \cdot (C+1)$ problemas da mochila, que são resolvidos recursivamente. A solução e valor do PMI_k^c é dado por $y_k(c)$ e $z_k(c)$ respectivamente, ou seja, $y_k(c)$ informa a solução do estágio k no estado c e $z_k(c)$ o valor total associado.

$$PMI = Max\left\{v \cdot y \mid p \cdot y \leq C,\ y \in \mathbb{Z}_+^m\right\} \tag{2.8}$$

$$PMI_k^c = Max\left\{v \cdot y \mid p \cdot y \leq c,\ y \in \mathbb{Z}_+^k\right\} \tag{2.9}$$

Com isso, a resolução de cada estágio e estado é dado pela fórmula recursiva (2.10) para $k = 2 \ldots m$ e $c = 0 \ldots C$. Nessa fórmula, o parâmetro j representa a quantidade de vezes que o item k será inserido na mochila, e nela avalia-se qual é a melhor decisão a ser tomada entre as $(\lfloor c/p_k \rfloor + 1)$ possíveis. São elas, colocar o item k, $j = 0$ vezes na mochila e carregar o estado $(c - 0 \cdot p_k)$ do estágio anterior $(k-1)$, ou colocar o item k, $j = 1$ vezes na mochila e carregar o estado $(c - 1 \cdot p_k)$ do estágio anterior, e

assim sucessivamente até $j = \lfloor c/p_k \rfloor$. Para utilizar a fórmula é necessário a resolução prévia do primeiro estágio, que é dado por (2.11).

$$z_k(c) = Max\{j \cdot v_k + z_{k-1}(c - j \cdot p_k),\ j = 0 \ldots \lfloor c/p_k \rfloor,\ j \in \mathbb{N}\} \tag{2.10}$$

$$\begin{aligned} z_1(c) &= \begin{cases} 0, & se \quad c = 0 \ldots p_1 - 1 \\ v_1, & se \quad c = p_1 \ldots 2p_1 - 1 \\ \vdots & \vdots \quad\quad \vdots \\ \lfloor C/p_1 \rfloor \cdot v_1, & se \quad c = \lfloor C/p_1 \rfloor \cdot p_1 \ldots C \end{cases} \\ y_1(c) &= \begin{cases} 0, & se \quad c = 0 \ldots p_1 - 1 \\ 1, & se \quad c = p_1 \ldots 2p_1 - 1 \\ \vdots & \vdots \quad\quad \vdots \\ \lfloor C/p_1 \rfloor, & se \quad c = \lfloor C/p_1 \rfloor \cdot p_1 \ldots C \end{cases} \end{aligned} \tag{2.11}$$

Perceba que $y_k(c)$ é um escalar que somente informa a quantidade do item k no estado c do estágio k, sendo necessário um processo de recuperação de dados para obter as soluções anteriores. Se (2.10)-(2.11) fossem implementados para resolver o problema da mochila, seria necessário o armazenamento das soluções e valores de todos os estados de cada estágio, o que se torna computacionalmente custoso para problemas grandes, sendo necessário um tempo computacional maior para resolver o problema.

Pisinger 2000 aborda o problema da mochila limitada, propondo uma nova formulação de programação dinâmica. Ele apresenta um modo de resolver o PML baseado na estratégia vista por Martello e Toth 1990 de transformar o PML em uma mochila binário. Dessa forma, utilizar técnicas de resolução específicas do problema da mochila 0-1. No entanto, ele enumera algumas complicações que esse processo de transformação causa, sendo as principais, o número elevado de variáveis que o problema passa a ter e a característica das instâncias (fracamente relacionados), que são difíceis de serem resolvidas. Essa característica está relacionada com o peso do item não ser necessarimante proporcional ao seu valor. Para evitar esses problemas, o autor primeiro resolve o problema da mochila limitada, relaxando a restrição $y \in \mathbb{Z}^m$ para $y \in \mathbb{R}^m$ (LBKP) e usa um algoritmo que determina a solução ótima a partir da relaxada, pois em geral, poucas alterações são necessárias de uma solução para a outra. Durante o algoritmo, o autor utiliza a programação dinâmica para resolver um subproblema da mochila gerado pelo algoritmo, de forma a modificar a solução do

LBKP. Essa formulação utilizada é diferente da apresentada na Seção 2.4, pois ela usa limites superiores obtidos da solução do LBKP.

As técnicas de pré-processamento são muito importantes para todo tipo de problema. No PMI, as relações de dominância ajudam na redução do espaço de busca de algoritmos. Com isso, existem artigos que exploram novas relações de dominância. Andonov, Poirriez e Rajopadhye 2000 propôs a *threshold dominance* (dominância limiar), que pode ser vista como uma generalização de outras relações de dominância (simples, múltipla e coletiva). Um conjunto de itens do tipo $\mathscr{I}$ domina limiarmente o item j se $\sum_{i\in\mathscr{I}} p_i \cdot y_i \leq p_j \cdot y_j$ e $\sum_{i\in\mathscr{I}} v_i \cdot y_i \geq v_j \cdot y_j$ com $y_i, y_j \in \mathbb{Z}_+$. A dominância simples é dada se $\mathscr{I}$ é conjunto unitário e $y_i = y_j = 1$, a dominância múltipla considera $\mathscr{I}$ unitário e $y_j = 1$, e a dominância coletiva, $y_j = 1$.

As relações de dominância também são úteis para diminuir o espaço de estado ou até mesmo eliminar estágios no método de programação dinâmica. Uma abordagem como essa pode ser encontrada no trabalho de Perin e Rangel 1989. Os autores propuseram um algoritmo lexicográfico para resolver o problema da mochila inteira, no qual somente se considera estados não dominados da programação dinâmica. Nessa pesquisa será feito uma adaptação do algoritmo proposto pelos autores para o caso da mochila limitada.

2.3 Algoritmo de Perin e Rangel (1989)

Após fazer uma pequena busca na literatura sobre programação dinâmica (Seção 2.2 e Seção 3.3), não foi encontrado uma adaptação do algoritmo de programação dinâmica de Horowitz e Sahni 1974 para o problema da mochila binária para resolver o PML, conforme proposto em Perin e Rangel 1989. Para facilitar a compreensão da modificação feita para o caso limitado descrevemos inicialmente o algoritmo de Perin e Rangel 1989.

Considere o terno $(c, z_k(c), y_k(c))$ como sendo um vetor que possui a capacidade da mochila no estado c, valor do estágio k no estado c ($z_k(c)$) e solução do estágio k no estado c ($y_k(c)$). Cada estágio $k = 1 \ldots m$ é identificado como uma sequência S^k desses vetores, ordenados de forma crescente em relação à primeira coordenada. As sequências S^k são não dominadas, definidas de acordo com o critério de dominância simples: $(c, z_k(c), y_k(c))$ domina simplesmente o terno $(\bar{c}, z_k(\bar{c}), y_k(\bar{c}))$, se $c \leq \bar{c}$ e $z_k(c) \geq z_k(\bar{c})$. A sequência S^1 ((2.12)) é obtida de acordo com (2.11). Perceba que os ternos

com $j \cdot p_1 < c < (j+1)p_1$ para $j = 0 \ldots \lfloor C/p_i \rfloor - 1$ não aparecem na sequência, pois o valor deles se mantêm iguais, ou seja, são simplesmente dominados pelo de menor capacidade c.

$$S^1 = \{(0,0,0),(p_1,v_1,1),(2p_1,2v_1,2),\ldots,(\lfloor C/p_1 \rfloor \cdot p_1, \lfloor C/p_1 \rfloor \cdot v_1, \lfloor C/p_1 \rfloor)\} \tag{2.12}$$

A sequência S^k, com $k = 2 \ldots m$ é obtida a partir da sequência S^{k-1} de acordo com o seguinte. Para cada $j = 0 \ldots \lfloor C/p_k \rfloor$ inteiro, são construídas sequências S_j^k intermediárias. Elas são obtidas pela soma de $(j \cdot p_k, j \cdot v_k, j)$ com cada terno de S^{k-1}. Essa soma é a usual do $\mathbb{R}^3$ e a $S_0^k = S^{k-1}$. Em seguida, é feito o processo de *merge*, que é a união não dominada das S_j^k respeitando a ordem da primeira coordenada, obtendo-se a S^k, ou seja, $S^k = \dot{\bigcup}_j S_j^k$. É importante dizer que as S_j^k são sequências crescentes em relação a primeira coordenada, não dominadas e viáveis (primeira coordenada do terno é menor ou igual a C). Com isso, após o processo de *merge*, a S^k também será uma sequência crescente, não dominada e viável de ternos.

O valor ótimo para o problema é encontrado no último terno da S^m, ou seja, em $z_m(c') = z^*$ com c' sendo a capacidade da mochila utilizada. Para obter a solução referente à esse valor, necessita-se de um processo de recuperação de solução iterativo, que é dado da seguinte forma: tome o último terno de S^m, e recupera-se $y_m(c') = y_m^*$ e $c' = C$, depois faz iterativamente $C = C - p_k \cdot y_k^*$ e $y_{k-1}(C) = y_{k-1}^*$ para $k = m \ldots 2$. A solução ótima será dada pelo vetor m-dimensional y^*.

Uma diferença entre esse algoritmo e a implementação de (2.10) e (2.11) é que não são armazenados todos os estados de cada estágio, o que melhora o desempenho do algoritmo, além disso, somente as S^k são armazenadas, não sendo necessário as S_j^k. Observe que nesse algoritmo o único fator que limita a quantidade do item k na mochila é a própria capacidade da mochila $y_k \leq \lfloor C/p_k \rfloor$. No entando, para o caso limitado, muitas vezes a quantidade do item k é limitada por $y_k \leq b_k < \lfloor C/p_k \rfloor$, e portanto é necessário adaptar esse algoritmo para o caso limitado.

2.4 Adaptação do algoritmo para o caso limitado

No algoritmo PR de Perin e Rangel 1989, o número de vezes que o item k é colocado na mochila é determinado pelo número de S_j^k. Para fazer a adaptação é

necessário fazer a limitação da frequência da S_j^k, ou seja, para o item k de limite b_k são construídas $j = 0 \ldots \min\{b_k, \lfloor C/p_k \rfloor\}$ inteiro, sequências S_j^k. Essa adaptação será chamada de PML^{PD}.

O pseudocódigo do algoritmo é apresentado no Algoritmo 1. Os dados de entrada são os pesos (p), valores (v) e limites (b) de cada item e a capacidade C da mochila. Como saída é fornecido um vetor S m-dimensional, no qual, cada coordenada é uma matriz de três colunas que representa a sequência S^k, ou seja, na coordenada k encontra-se a S^k, com cada linha da matriz representando um terno da sequência. Essa estrutura de dados pode ser vista em (2.13). Na linha 1 do pseudocódigo, inicia-se o vetor de armazenamento S com o terno nulo das S^k e nas linhas 2-15 elas são totalmente construídas iterativamente. Nas linhas 3 a 6 recupera-se a S^{k-1}, nas linhas 7 a 8 as matrizes M e M_{op} recebem a S^k. Nas linhas 9 a 12 são obtidas as S_j^k por meio da matriz M_{op} e são armazenadas em M de forma iterativa, na linha 9 está o fator que limita a quantidade do item k na mochila. Na linha 13 é feito o processo de *Merge* em M, obtendo-se a S^k. O processo *Merge* será melhor explicado no Algoritmo 2, pois se difere do *merge* feito no algoritmo PR. Na linha 14 armazena-se a S^k em S. É importante dizer que, assim como no algoritmo PR, somente são armazenadas as S^k presentes em S.

Algoritmo 1: Adaptação de Rangel e Perin (1989)

Entrada: p, v, b, C

Saída: S vetor de matrizes de resultados

1 $S = \{[0\ \ 0\ \ 0], [0\ \ 0\ \ 0], \ldots, [0\ \ 0\ \ 0]\} \rightarrow$ vetor que recebe iterativamente a solução do PM com os k primeiros itens;
2 **para** $k = 1 \ldots m$ **faça**
3 **se** $k > 1$ **então**
4 Faça a coordenada k em S igual à coordenada $k-1$ de S;
5 Faça a terceira coluna da coordenada k de S igual a 0
6 **fim**
7 $M = S^k$;
8 $M_{op} = S^k$;
9 **para** $j = 1 \ldots min\{b_k, \lfloor C/p_k \rfloor\}$ **faça**
10 $M_{op} = M_{op} \ .+ [p_k\ \ v_k\ \ 1] \rightarrow (.+)$ soma de $[p_k\ v_k\ 1]$ aos elementos de cada linha de M_{op};
11 Insira as linhas de M_{op} no final de M
12 **fim**
13 *Merge* em $M \rightarrow$ ordenação e eliminação das linhas dominadas ou inviáveis;
14 Faça S^k igual à M
15 **fim**

$$S = \left\{ \begin{bmatrix} 0 & 0 & 0 \\ p_1 & v_1 & 1 \\ 2p_1 & 2v_1 & 2 \\ \vdots & \vdots & \vdots \\ \lfloor C/p_1 \rfloor p_1 & \lfloor C/p_1 \rfloor v_1 & \lfloor C/p_1 \rfloor \end{bmatrix}, \cdots, \begin{bmatrix} 0 & 0 & 0 \\ s^m_{21} & s^m_{22} & s^m_{23} \\ s^m_{31} & s^m_{32} & s^m_{33} \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ s^m_{n1} & s^m_{n2} & s^m_{n3} \end{bmatrix} \right\} \tag{2.13}$$

Ao observar a descrição feita do algoritmo PR na Seção 2.3 e o pseudocódigo descrito no Algoritmo 1, a principal diferença é que na descrição, as S^k_j são sequências crescentes de ternos não dominados e viáveis, enquanto que no Algoritmo 1, na linha 12, M_{op} que faz o papel das S^k_j, possui ternos inviáveis e M possui ternos inviáveis e dominados. Somente na linha 13 é que M passa a representar a S^k como sequência crescente de ternos não dominados e viáveis. Esse processo é feito através da função

Merge que teve de ser contruída e seu pseudocódigo é dado no Algoritmo 2.

As entradas do Algoritmo 2 são as colunas da matriz M, sendo P a dos pesos, V dos valores e Y das quantidades e a capacidade da mochila C. No pseudocódigo, o termo $\#P$ significa o tamanho de P, que é igual ao número de linhas da matriz M, e $P(x)$ representa a coordenada x de P, ou seja, a linha x da coluna P da matriz M. Nas linhas 1 a 10 é feito a eliminação dos ternos inviáveis, de 11 a 17 os ternos são ordenados de forma crescente em relação a primeira variável e nas linhas 18 a 27 são eliminados os ternos simplesmente dominados. Na implementação do algoritmo PR feito por Perin e Rangel 1989, a sequência S^k de ternos ordenados e não dominados é construída usando as funções nexti, nextj e *push* (e.g. Lemos 2005).

Algoritmo 2: Função *Merge*

Entrada: P, V, Y, C

Saída: M como sequência crescente de ternos não dominados e viáveis

```
r = 1;
enquanto r ≤ #P faça
    enquanto P(r) > C faça
        Elimine o terno da linha r;
        se r > #P então
            PARE
        fim
    fim
    r = r + 1
fim
para i = 1...#P faça
    para j = 1...#P faça
        se P(i) > P(j) e i < j então
            Coloque o terno da linha j na linha i
        fim
    fim
fim
u = 2;
enquanto u ≤ #P faça
    enquanto P(u) ≥ P(u−1) e V(u) ≤ V(u−1) faça
        Elimine o terno da linha u;
        se u > #P então
            PARE
        fim
    fim
    u = u + 1
fim
```

Após utilizar o Algoritmo PML^{PD} e resolver o problema da mochila limitada, o processo de recuperação da solução é feito através do Algoritmo 3. Na mesma ideia do algoritmo anterior, $\#S^k$ significa o número de linhas da coordenada k do vetor S e $S^k(i, j)$ o valor encontrado na linha i da coluna j da coordenada k do vetor S. Os dados de entrada necessários são o peso dos itens (p) e o vetor S, o de saída é o vetor m-

dimensional y^* contendo a solução do problema. Nas linhas 1 e 2 recupera-se quantas vezes o item m é inserido na mochila e a capacidade utilizada, respectivamente. Nas linhas 3 a 7 recupera-se iterativamente as outras soluções, sendo que na linha 4 é feita a atualização da capacidade da mochila, na linha 5 a identificação do terno, e na linha 6 é feito o armazenamento da solução em y^*.

Algoritmo 3: Recuperação da solução

Entrada: p, S

Saída: y^* vetor de solução

1 $y^*_m = S^m(\#S^m, 3) \rightarrow$ linha $\#S^m$ da terceira coluna da matriz S^m ;
2 $C = S^m(\#S^m, 1) \rightarrow$ linha $\#S^m$ da primeira coluna da matriz S^m;
3 **para** $k = m \ldots 2$ **faça**
4 $\quad C = C - p_k y^*_k$;
5 $\quad$ recupere a linha i de S^{k-1} tal que $S^{k-1}(i,1) = C$;
6 $\quad y^*_{k-1} = S^{k-1}(i,3)$
7 **fim**

2.5 Um exemplo para o Problema da Mochila Limitada

Considere o problema no qual a mochila tem capacidade $C = 165$ e os itens têm parâmetros $(p_i \times v_i) = \{(30 \times 138), (45 \times 405), (70 \times 784)\}$ e demanda $b_i = \{3, 2, 1\}$. O objetivo é utilizar o algoritmo PML^{PD} para resolver o problema da mochila e comparar a solução caso não houvesse limite de demanda. Em (2.14) apresenta-se o vetor S contendo as sequências S^k e o valor da solução da mochila é dado na última linha da coluna 2 da S^3 ($z^* = 1394$). É interessante perceber que conforme os estágios progridem, nem todos os ternos são mantidos, isso é causado pelo processo de *Merge* que elimina os ternos dominados, por exemplo, o terceiro terno de S^1 é dominado pelo terceiro terno de S^2, o quarto terno de S^1 é dominado pelo quarto terno de S^2 etc. Para recuperar a solução, $y_m(c') = y^*_3 = 1$ e $c' = C = 160$, então $C = 160 - 70 \cdot 1 = 90$ e $y_2(90) = y^*_2 = 2$, por fim, $C = 90 - 45 \cdot 2 = 0$ e $y_1(0) = y^*_1 = 0$. Portanto, a solução para o problema é $y^* = [0\ 2\ 1]^T$.

$$S = \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 30 & 138 & 1 \\ 60 & 276 & 2 \\ 90 & 414 & 3 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 30 & 138 & 0 \\ 45 & 405 & 1 \\ 75 & 443 & 1 \\ 90 & 610 & 2 \\ 120 & 748 & 2 \\ 150 & 886 & 2 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 30 & 138 & 0 \\ 45 & 405 & 0 \\ 70 & 784 & 1 \\ 100 & 922 & 1 \\ 115 & 1089 & 1 \\ 145 & 1227 & 1 \\ 160 & 1394 & 1 \end{bmatrix} \right\} \tag{2.14}$$

Caso o problema fosse resolvido sem considerar os limitantes, o vetor S obtido seria o apresentado em (2.15). Perceba que de S^2 para S^3 foi reduzido o número de ternos. Isso ocorreu porque ao construir as S_j^3 e fazer o processo de *Merge* na matriz M obtendo a sequência S^3, muitos ternos foram dominados. Além disso, o valor da solução obtida é outro, $z^* = 1568$ e com o uso de capacidade menor $c' = 140$. Isso foi obtido porque o terceiro item é muito valioso e utiliza pouca capacidade da mochila. Para recuperar a solução basta fazer o mesmo processo para o caso limitado, resultando em $y^* = [0\ 0\ 2]$.

$$S = \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 30 & 138 & 1 \\ 60 & 276 & 2 \\ 90 & 414 & 3 \\ 120 & 552 & 4 \\ 150 & 690 & 5 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 30 & 138 & 0 \\ 45 & 405 & 1 \\ 75 & 443 & 1 \\ 90 & 610 & 2 \\ 120 & 748 & 2 \\ 135 & 915 & 3 \\ 165 & 1053 & 3 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 30 & 138 & 0 \\ 45 & 405 & 0 \\ 70 & 784 & 1 \\ 100 & 922 & 1 \\ 115 & 1089 & 1 \\ 140 & 1568 & 2 \end{bmatrix} \right\} \tag{2.15}$$

Com esse exemplo, pode-se ver que o algoritmo proposto realmente cumpre seu propósito de resolver o problema da mochila limitada. No Capítulo 5 são apresentados resultados de testes computacionais realizados para verificar a eficiência do algoritmo, e também, encontrar suas possíveis limitações.

Captulo 3

O PROBLEMA DE CORTE

O problema de corte (PC) unidimensional pode ser definido da mesma forma que foi feito a definição do problema da mochila no Capítulo 2. Isso ocorre pois o PC também é formulado através do modelo (1.5)-(1.8). No caso bidimensional o problema é definido da seguinte forma. Considere um conjunto de itens retangulares, em que cada item possui comprimeto l_i e largura w_i, $i = 1 \ldots m$, e um objeto retangular de comprimento L e largura W. O PC consiste em cortar um subconjunto de itens do objeto considerando a maximização da área total cortada. O corte considerado nessa pesquisa é o corte ortogonal que é feito ortogonalmente aos lados do objeto. Sob essas condições, o problema pode ser classificado como *2-dimensional retangular problema de posicionamento de objeto único*.

A modelagem do caso bidimensional não é trivial, outros fatores como posicionamento e rotação devem ser considerados na formulação matemática do problema. A forma geométrica dos itens e a forma geométrica do objeto também devem ser considerados na formulação, no entanto, nessa pesquisa todos os itens e o objeto são retangulares. É importante também considerar outras características que são referentes a como os itens serão cortados do objeto. Essas características são discutidas na Seção 3.1 de restrições do equipamento de corte.

3.1 Restrição do Equipamento de Corte

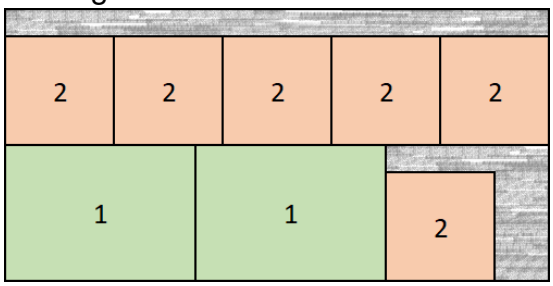
Para tratar das restrições do equipamento de corte, primeiro apresenta-se a definição de padrão de corte bidimensional (Definição 3.1.1).

Definição 3.1.1. *Um padrão de corte bidimensional é qualquer solução factível do*

problema de corte bidimensional, ou seja, é uma representação de como os itens estão posicionados (alocados) no objeto antes de serem cortados.

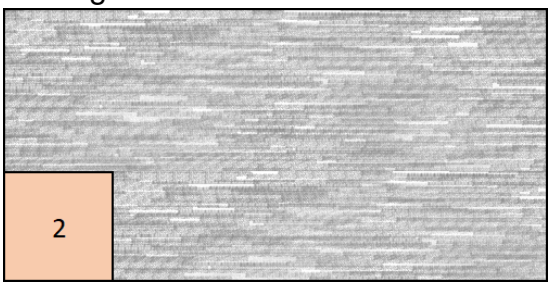
A ilustração dessa definição é dado pelas Figuras 3.1 e 3.2, perceba que contendo apenas um item no objeto (Figura 3.2), ainda caracteriza-se um padrão de corte, e se esse item estivesse um pouco mais a direita, seria outro padrão diferente dos apresentados. Os retângulos hachurados em cinza nos padrões de corte representam as sobras do objeto, pois não fazem parte do conjunto de itens demandados.

Figura 3.1: Padrão de corte 1.



Fonte: autor.

Figura 3.2: Padrão de corte 2.



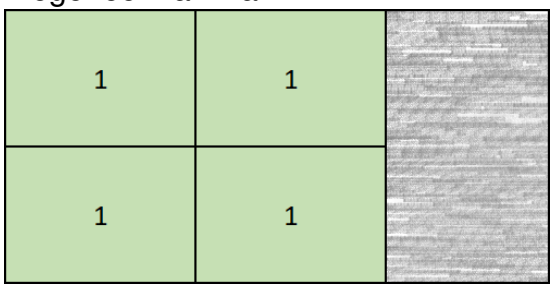
Fonte: autor.

Um ponto importante sobre os padrões de corte é a relação entre os tipos de itens presentes no padrão e o seu aproveitamento. Na Definição 3.1.2 são apresentados alguns desses aspectos, sendo eles, (não)homogêneo, (não)maximal e a combinação desses aspectos.

Definição 3.1.2. *(i) Um padrão de corte é dito homogêneo se há apenas um tipo de item nele. (ii) Um padrão é dito maximal, se não é possível haver o arranjo de mais nenhum item nesse padrão de corte. (iii) Um padrão é dito homogêneo maximal se (i) e (ii) ocorrem simultaneamente.*

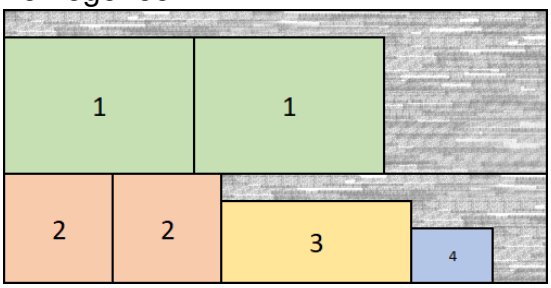
Nas Figuras 3.3 e 3.2 são apresentados padrões homogêneos maximal e não-maximal, respectivamente, e nas Figuras 3.4 e 3.1 padrões não-homogêneos não-maximal e maximal, respectivamente.

Figura 3.3: Padrão de corte homogêneo maximal.



Fonte: autor.

Figura 3.4: Padrão de corte não-homogêneo.



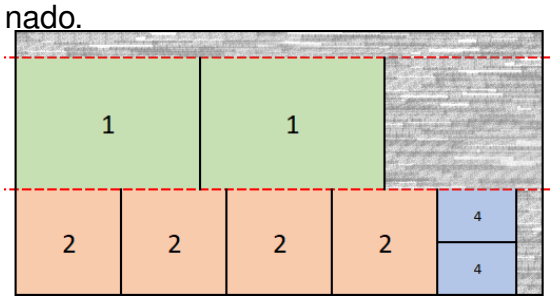
Fonte: autor.

Outro fator relevante na solução do problema de corte bidimensional diz respeito à restrição do equipamento. Em algumas indústrias, a máquina de corte só realiza cortes guilhotinados que podem ser ortogonais ou não-ortogonais, nessa pesquisa será sempre considerado o caso ortogonal (Definição 3.1.3).

Definição 3.1.3. *(i) Um corte guilhotinado ortogonal é um corte paralelo a um dos lados do objeto atravessando-o de ponta a ponta, obtendo-se dois objetos menores. (ii) Um padrão de corte é tido como guilhotinado se, e somente se, para obter todos os itens são feitos somente cortes guilhotinados. (iii) Caso contrário, o padrão é não guilhotinado.*

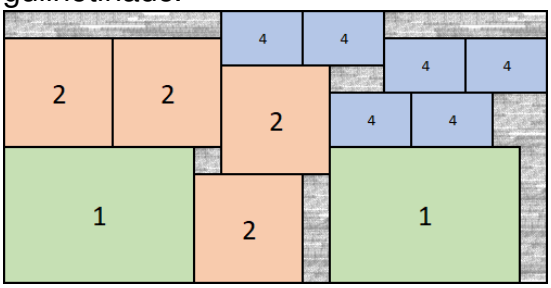
Na Figura 3.5 apresenta-se um corte guilhotinado, no qual as linhas vermelhas representam esses cortes guilhotinados, enquanto na Figura 3.6 não é possível fazer um corte desse tipo em nenhum ponto, pois caso fosse feito, danificaria algum item. O corte tratado na pesquisa será o guilhotinado ortogonal.

Figura 3.5: Padrão de corte guilhotinado.



Fonte: autor.

Figura 3.6: Padrão de corte não-guilhotinado.



Fonte: autor.

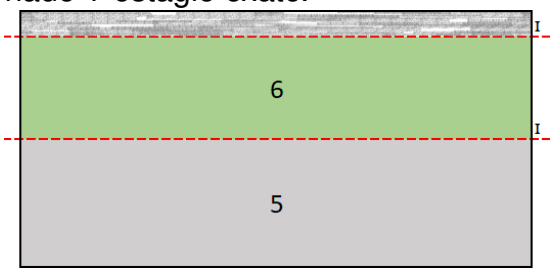
Para padrões de corte guilhotinados, é possível definir quantos estágios são necessários para obter todos os itens, e também, se ele é exato ou não-exato. Os

padrões de corte não-guilhotinados não possuem estágios. A determinação do número de estágios $(k+1)$ no padrão de corte está associado à quantidade (k) de rotações de 90 graus necessárias no objeto para obter todos os itens (Definição 3.1.4).

Definição 3.1.4. *(i) Um padrão de corte guilhotinado é k-estágios se são feitos k-1 rotações de 90 graus para obter todos os itens do padrão de corte. (ii) O padrão de corte é exato se ao final do último estágio todos os itens foram obtidos sem precisar de aparas. (iii) Caso contrário o padrão de corte é não exato.*

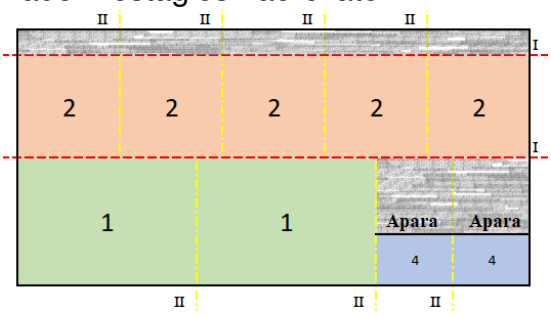
Na Figura 3.7, apresenta-se um padrão de corte 1-estágio exato, pois não foi necessário nenhuma rotação e ao final do primeiro estágio todos os itens foram obtidos. Perceba que o fato de ter sobras no padrão, não está relacionado com ele ser exato ou não. Na Figura 3.8 é dado um padrão 2-estágios não exato, pois após os primeiros cortes guilhotinado (estágio 1), mostrados por I, o objeto é rotacionado e seguem-se os cortes do segundo estágio, mostrados por II. Ao final do segundo estágio, para obter cada um dos itens 4 é necessário um processo de apara, com isso o padrão é não exato. Cabe dizer que o processo de apara pode ser considerado como um terceiro estágio ou não, isso irá depender das condições da indústria, pois algumas fazem as aparas em máquinas diferentes e de manejo rápido, enquanto algumas fazem as aparas nas próprias máquinas de corte que necessitam ajustes mais demorados.

Figura 3.7: Padrão de corte guilhotinado 1-estágio exato.



Fonte: autor.

Figura 3.8: Padrão de corte guilhotinado 2-estágios não-exato.



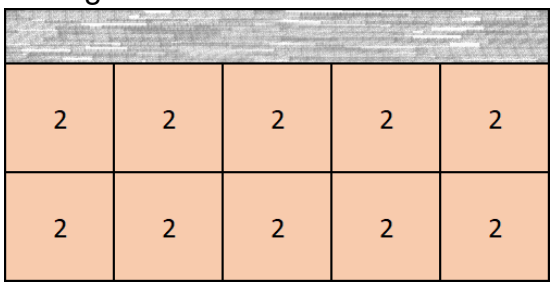
Fonte: autor.

Há também padrões de corte que possuem uma organização melhor dos itens e que permitem o corte simultâneo de faixas em um estágio posterior chamados de padrões de corte n-grupo (Definição 3.1.5). Esses padrões de corte possuem um tempo de processamento baixo porque são necessários poucos ajustes nas máquinas de corte para obtê-los (e. g. Morabito e Arenales 2000).

Definição 3.1.5. *(i) Um padrão de corte* 1*-grupo é um caso especial do padrão de corte 2-estágios, no qual, no segundo estágio todas as faixas obtidas pelo primeiro podem ser cortadas simultaneamente para obter os itens. (ii) Um padrão de corte* n*-grupo com* $n > 1$ *é um padrão composto por* n *padrões* 1*-grupo.*

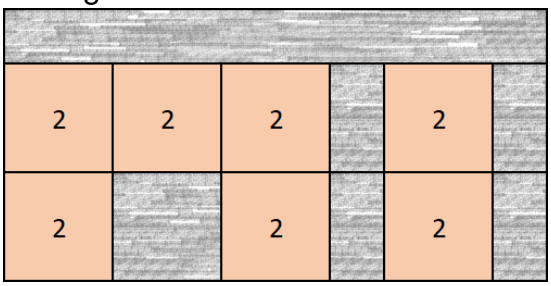
Nas Figuras 3.9-3.11 são ilustrados padrões de corte 1-grupo, veja que as sobras não influenciam na questão do padrão ser 1-grupo ou não. As faixas obtidas no primeiro estágio são iguais e podem ser ou não maximais e/ou não-exatos. Na Figura 3.12 é ilustrado um padrão 2-grupos, perceba que após o primeiro corte vertical (pontilhado), são obtidos dois padrões 1-grupo.

Figura 3.9: Padrão de corte 1-grupo homogêneo maximal.



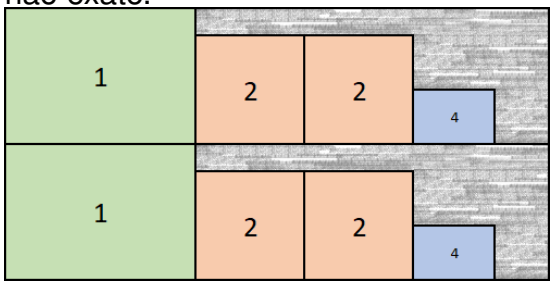
Fonte: autor.

Figura 3.10: Padrão de corte 1-grupo homogêneo não-maximal.



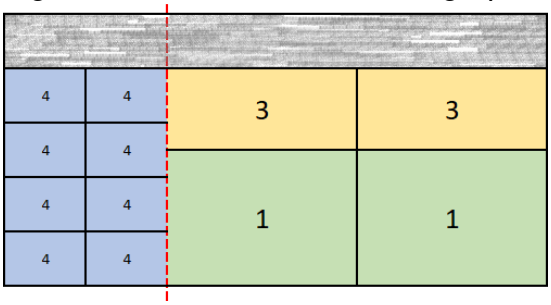
Fonte: autor.

Figura 3.11: Padrão de corte 1-grupo não exato.



Fonte: autor.

Figura 3.12: Padrão de corte 2-grupos.



Fonte: autor.

Outra característica necessária para descrever o problema de corte tratado nessa pesquisa, está relacionado à quantidade de itens no padrão de corte. Assim como no problema da mochila inteira, em que há o caso limitado, no problema de corte também existe essa variante e recebe o nome de restrito.

Definição 3.1.6. *Um padrão de corte é restrito se a quantidade do item* i *no padrão de corte não ultrapassar um limite máximo* b_i*. Caso contrário, ele é irrestrito.*

Um outro critério que pode ser utilizado para resolver o PC é a minimização da perda total do objeto (Definição 3.1.7 (i)). Em situações reais, é necessário diferenciar a perda total do objeto pois há uma perda que é inerente ao processo de corte e que está relacionada com a espessura da serra (δ) de corte (Definição 3.1.7 (ii)) (Rangel e Figueiredo 2008). Para avaliar o desempenho de métodos de resolução nesses casos, é utilizado a perda por sobra de material que considera a espessura da serra no cálculo da área dos itens, ou seja, a espessura da serra não é considerada uma perda (Definição 3.1.7 (iii)) (Rangel e Figueiredo 2008). A Definição 3.1.7 é feita para o caso bidimensional, no entanto, é facilmente simplificada para o caso unidimensional ou estendida para o tridimensional adaptando o cálculo feito em (3.1).

Definição 3.1.7. *(i) A perda de matéria prima é qualquer parte do objeto que não é aproveitada após o processo de corte. (ii) A perda por desgaste da serra é uma perda inerente ao processo de corte e que está incluída na perda total da matéria prima. (iii) A perda por sobra de material não considera a espessura da serra como uma perda e é calculada por (3.1) com a_i representando o número de vezes que o item i é cortado do padrão de corte e δ a espessura da serra.*

$$Ps = \text{Máximo}\left\{0, L \cdot W - \sum_{i=1}^{m} (a_i \cdot (l_i + \delta) \cdot (w_i + \delta))\right\} \tag{3.1}$$

O problema de corte tratado nessa pesquisa considerou a obtenção de um padrão de corte bidimensional guilhotinado 2-estágios restrito, classificado como *2-dimensional retangular problema de posicionamento de objeto único (2-estágios restrito)*.

3.2 Modelagem do problema de corte bidimensional

Visto que o problema de corte bidimensional possui diversas variantes, e para cada uma, a natureza do problema pode sofrer alterações, é muito difícil de se fazer uma adaptação direta do modelo teórico (1.5)-(1.8) para esse caso. Então, para representar o problema será utilizado o modelo não-linear descrito em Yanasse e Morabito 2013. O modelo (3.2)-(3.6) representa o problema de se obter um padrão de corte bidimensional 2-estágios restrito. São considerados os seguintes parâmetros para o modelo:

- $v_i =$ área do item i para $i = 1 \ldots m$;
- $l_i =$ comprimento do item i para $i = 1 \ldots m$;
- $w_i =$ largura do item i para $i = 1 \ldots m$;
- $b_i =$ limite do item i para $i = 1 \ldots m$;
- $L =$ comprimento do objeto;
- $W =$ largura do objeto;
- $K =$ número de larguras diferentes (dos itens);
- $P_k =$ número máximo de faixas $L \times w_k$;
- $\mathscr{I}_k =$ conjunto de índices tal que $w_i \leq w_k$.

As variáveis de decisões para o modelo são:

- $\gamma_{kp}^{i} =$ número de vezes que o item i é cortado do p-ésimo padrão da faixa $L \times w_k$;
- $\beta_{kp} =$ número de vezes que o p-ésimo padrão da faixa $L \times w_k$ é cortado do padrão de corte.

A ideia usada na formulação é construir faixas de comprimento L e diferentes larguras $w_k, k = 1 \ldots K$, tal que na faixa $L \times w_k$ só poderão ser alocados itens i com $w_i \leq w_k$. Para alocar os itens nas faixas, é suposto que eles serão alocados em sequência, assim como no problema da mochila. Após ter obtido as faixas, elas são combinadas na largura W do padrão de corte (seguindo a mesma suposição do problema da mochila). Assim, em (3.2) é dado o critério de maximização da área de ocupação dos itens, veja que essa é uma função não-linear. As restrições (3.3) e (3.4) são de contenção dos itens no objeto, sendo as primeiras $P_k, k = 1 \ldots K$ para o comprimento do objeto e a última para a largura do objeto. A restrição (3.5) garante que a quantidade de cada item não ultrapasse seu limite b, e é também uma restrição não-linear. O domínio das variáveis é definido por (3.6). Perceba que a restrição (1.6) do modelo geral tornou-se $\sum_{k=1}^{K} P_k + 1$ restrições da mochila, que garatem a contenção dos itens.

$$Max \sum_{k=1}^{K} \sum_{p=1}^{P_k} \sum_{i \in \mathscr{I}_k} v_i \cdot \gamma_{kp}^{i} \cdot \beta_{kp} \tag{3.2}$$

$$S.\,a: \sum_{i \in \mathscr{I}_k} l_i \cdot \gamma_{kp}^{i} \leq L, \;\; k = 1 \ldots K, \; p \in P_k \tag{3.3}$$

$$\sum_{k=1}^{K} \sum_{p=1}^{P_k} w_k \cdot \beta_{kp} \leq W \tag{3.4}$$

$$\sum_{k=1}^{K} \sum_{p=1}^{P_k} \gamma_{kp}^{i} \cdot \beta_{kp} \leq b_i, \;\; i = 1 \ldots m \tag{3.5}$$

$$\gamma_{kp}^{i}, \beta_{kp} \geq 0, \; inteiro, \; i = 1 \ldots m; \;\; k = 1 \ldots K; \;\; p = 1 \ldots P_k \tag{3.6}$$

Essa formulação trata do problema de corte bidimensional como problemas de corte unidimensionais. A natureza do problema de corte unidimensional é a mesma do problema da mochila inteira. Ao cortar os itens das faixas de comprimento L, isso é o mesmo do que alocar os itens na mochila de capacidade L. Assim como, combinar as faixas no padrão de corte de largura W é o mesmo de alocar as faixas na mochila de capacidade W. Por isso as restrições (3.3) e (3.4) são iguais a do problema da mochila e é necessário a suposição dos itens serem cortados em sequência.

Yanasse e Morabito 2013 observam que na literatura há poucos modelos que representam o problema de corte bidimensional como múltiplos problemas da mochila unidimensional, e também, a importância desses modelos pois é uma maneira de se desenvolver novos métodos de resolução que aproveitem a estrutura do modelo e podem ser usados para avaliar o desempenho de heurísticas. Eles propõem três modelos lineares. O primeiro obtido através da linearização de um modelo não linear da literatura, o segundo de uma extensão de um modelo que gera padrão 1-grupo e o terceiro uma adaptação do segundo modelo, utilizando variáveis binárias. Eles também observam que tratar o problema de corte bidimensional como múltiplos problemas da mochila unidimensional é uma ideia que já existe desde o trabalho de Gilmore e Gomory 1965. A abordagem de Gilmore e Gomory 1965 será vista na Seção 3.4.

3.3 A Programação Dinâmica no Problema de Corte Bidimensional

Gilmore e Gomory 1965 propõem uma abordagem que resolve o problema de

corte bidimensional guilhotinado 2-estágios. A ideia da abordagem é a mesma do modelo (3.2)-(3.6), no entanto, ela não limita a quantidade de itens no padrão de corte (caso irrestrito). Para fazer a geração das faixas, são resolvidos m problemas da mochila e o arranjo delas é feito em outro problema da mochila. Outra diferença é que a resolução do problema não é feita em apenas um modelo, mas sim em $m+1$ modelos. Para resolvê-los Gilmore e Gomory 1965 utilizam de um algoritmo de programação dinâmica para o problema da mochila que possibilita a resolução dos m primeiros problemas de forma simultânea. A fórmula recursiva utilizada por Gilmore e Gomory 1965 é a mesma apresentada em (2.10), ou seja, é uma formulação para o problema de corte unidimensional. Essa abordagem será explicada em detalhes na Seção 3.4. Nos trabalhos seguintes, é utilizado a formulação de programação dinâmica para o problema de corte bidimensional, inicialmente tratada por Gilmore e Gomory 1966.

Gilmore e Gomory 1966 estendem a programação dinâmica da mochila unidimensional para mais dimensões. São definidas funções recursivas que resolvem o problema de corte bidimensional estagiado e não-estagiado. No caso não-estagiado, não há um limite no número de estágios necessários para cortar todos os itens do padrão de corte. Com o passar do tempo, pesquisadores perceberam que as funções recursivas propostas por Gilmore e Gomory 1965 poderiam ser melhoradas. Os primeiros artigos apresentando modificações nas funções recursivas foram os de Herz 1972 e Beasley 1985.

Herz 1972 mostrou que as funções recursivas propostas por Gilmore e Gomory 1966 do caso não-estagiado não eliminavam simetrias. Esse fato causa o aumento desnecessário do número de estados da programação dinâmica. Essa alteração se deu na geração de um conjunto de discretização do objeto. Beasley 1985 percebe um erro nas recursões de programação dinâmica para o caso estagiado. Esse erro está no fato de Gilmore e Gomory 1966 não terem desenvolvido diferentes funções para as diferentes direções do primeiro corte. Então, Beasley 1985 propõe essas funções e utiliza do conjunto de discretização definido por Herz 1972 para evitar simetrias no padrão.

Cintra et al. 2008 utilizam dos resultados de Herz 1972 e Beasley 1985 para propor três algoritmos. Os dois primeiros algoritmos são para fazer a geração do conjunto de discretização do objeto citado por Herz 1972. Em um deles o conjunto é gerado por enumeração explícita e o outro por programação dinâmica. O terceiro

algoritmo que também é de programação dinâmica, resolve a fórmula recursiva do problema de corte não-estagiado proposto por Beasley 1985. Além disso, Cintra et al. 2008 utilizam desse último algoritmo para resolver o problema de corte bidimensional estagiado com e sem a rotação dos itens.

Nesses artigos descritos, é tratado o caso irrestrito do problema de corte bidimensional. Christofides e Whitlock 1977 e Christofides e Hadjiconstantinou 1995 abordam o caso restrito. Christofides e Whitlock 1977 resolvem o problema de corte bidimensional por um algoritmo de busca em árvore. Assim como nos trabalhos anteriores, são utilizadas técnicas para evitar padrões simétricos e diminuir o espaço de busca. Christofides e Whitlock 1977 utilizam da programação dinâmica para obter limites superiores para serem utilizados no algoritmo de busca em árvore. Esses limites são gerados através da resolução do problema de corte bidimensional irrestrito.

Christofides e Hadjiconstantinou 1995 propõem um novo algoritmo de busca em árvore que é uma melhoria do algoritmo apresentado por Christofides e Whitlock 1977. A diferença entre os dois algoritmos está nos limites superiores usados. Para o algoritmo melhorado, primeiro eles desenvolvem uma formulação de programação dinâmica para o problema de corte bidimensional restrito. Como essa formulação apresenta um número muito elevado de estados, eles utilizam da relaxação do espaço de estado para obter o limite superior que é utilizado no algoritmo de busca em árvore.

Velasco 2017 também trabalha com a formulação de programação dinâmica proposta por Christofides e Hadjiconstantinou 1995 para o caso restrito. Ele segue a mesma abordagem de Christofides e Hadjiconstantinou 1995 para obter limites superiores ao problema (relaxação do espaço de estado). A relaxação do espaço de estado é feita associando pesos a cada um dos itens e a qualidade dos limites depende diretamente dos pesos associados. Então, Velasco 2017 calcula os pesos utilizando um problema de programação linear inteiro que leva em consideração todas as soluções ótimas da PD dos estados anteriores e não somente a do estado presente, como é feito na proposta de Christofides e Hadjiconstantinou 1995.

Russo et al. 2019 fazem uma revisão dos algoritmos de programação dinâmica usados para obter limites superiores. Eles falam que esses algoritmos podem ser implementados segundo duas estratégias: *bottom-up* e *top-down*. Na primeira, o valor de cada estágio e estado é computado apenas uma vez e no decorrer do algoritmo eles são recuperados. Esse é o caso da formulação do PM apresentada no Capítulo 2 e que também vale para o algoritmo PML^{PD}. Na segunda estratégia, o valor de cada

estado e estágio é propagado e atualizado cada vez que um valor melhor é encontrado. Com isso, eles são computados mais de uma vez. O nome dessas estratégias ganham mais sentido para os métodos de busca em árvore. Na *bottom-up*, os retangulos menores são mesclados de forma a obter retângulos maiores (baixo para cima). Na *top-down* o objeto é recursivamente cortado em retângulos menores de forma a obter a melhor disposição dos itens (cima para baixo). Russo et al. 2019 também apontam que algumas técnicas para diminuir o espaço de estado podem causar imprecisões nos métodos. Essa afirmação é feita quando o problema é tratado no caso restrito.

3.4 Método de duas etapas de Gilmore e Gomory (1965)

Para resolver o problema de corte bidimensional guilhotinado, Gilmore e Gomory 1965 propõem uma abordagem que obtem padrões de corte 2-estágios em duas etapas. Essa proposta decompõe o problema de corte bidimensional em $m+1$ problemas da mochila unidimensionais, sendo os primeiros m problemas da mochila referentes a primeira etapa e um referente à segunda. O processo é feito da seguinte forma. Primeiro ordena-se os itens de forma crescente em relação à largura ($w_i \leq w_{i+1}, i = 1 \dots m$), em seguida, são construídas faixas de largura $w_k, k = 1 \dots m$ e comprimento L, considerando para a faixa k somente os itens i com $i \leq k$. Assim, os m problemas da mochila são caracterizados por esse processo de construção das faixas e os m modelos matemáticos são dados em (3.7)-(3.9). Após construir as faixas, o problema $m+1$ é caracterizado pelo arranjo das faixas no padrão de corte, ou seja, decidir quais são as melhores faixas a serem inseridas na mochila de capacidade W, esse problema é representado por (3.10)-(3.12).

Em (3.7)-(3.9), π_i é o valor de utilidade dos itens, a variável γ_{ki} é a variável de decisão da mochila, que nesse contexto, indica a quantidade do item i na faixa k com $i \leq k$. A restrição (3.8) é a mesma do problema da mochila, no qual, l_i é o comprimento do item i, que faz o papel do peso p_i e a restrição exige que a soma dos comprimentos dos itens não ultrapassem o comprimento L do objeto (faixa). Em (3.9) indica-se o domínio das varáveis. No modelo (3.10)-(3.12), π_k^* é o valor de utilidade de cada faixa obtido resolvendo o problema da mochila k e β_k indica a quantidade da faixa k no padrão de corte. Na restrição (3.11), garante-se que a soma da largura das faixas utilizadas (w_k) é menor ou igual à largura W do objeto e em (3.12) é definido o

domínio das variáveis.

$$\pi^*_{k=1\ldots m} = Max \sum_{i=1}^{k} \pi_i \cdot \gamma_{ki} \tag{3.7}$$

$$S.a : \sum_{i=1}^{k} l_i \cdot \gamma_{ki} \leq L \tag{3.8}$$

$$\gamma_{ki} \in \mathbb{Z}_+, i = 1 \ldots k \tag{3.9}$$

$$Max \sum_{k=1}^{m} \pi^*_k \cdot \beta_k \tag{3.10}$$

$$S.a : \sum_{k=1}^{m} w_k \cdot \beta_k \leq W \tag{3.11}$$

$$\beta_k \in \mathbb{Z}_+, k = 1 \ldots m \tag{3.12}$$

Perceba que na primeira etapa permite-se que na faixa k sejam alocados itens com $w_i < w_k$, ou seja, é possível ter um processo de apara ao final do segundo estágio de corte, caracterizando um padrão de corte não exato. Caso queira evitar aparas, tornando o pradrão de corte exato, basta exigir que na faixa k somente sejam considerados os itens i com $i \leq k$ e $w_i = w_k$. O padrão de corte é representado por um vetor m-dimensional em que cada coordenada é dada por $\sum_{k=1}^{m} \gamma_{ki}\beta_k, i = 1 \ldots m$.

Essa abordagem é muito semelhante ao modelo encontrado no trabalho de Yanasse e Morabito 2013, no qual, os padrões para as faixas $L \times w_k$ podem ser enxergados como as faixas construídas nos primeiros m problemas da mochila. Ademais, como o modelo é para o caso restrito, na adaptação da abordagem de Gilmore e Gomory 1965 aparecerá uma restrição que possui o mesmo objetivo da restrição (3.5) e ela é a motivadora para a heurística proposta na Seção 3.6.

3.5 Adaptação do método para o caso restrito

A adaptação da proposta de Gilmore e Gomory 1965 para o caso restrito é feita incluindo novas restrições aos $m+1$ problemas da mochila. Para os m primeiros ((3.13)-(3.16)) são adicionados limites superiores aos itens, $\gamma_{ki} \leq b_i,\ i,k = 1 \ldots m$ e $i \leq k$, ou seja, a quantidade do item i na faixa k é limitada por um parâmetro b_i ((3.15)). Ao

problema $m+1$ ((3.17)-(3.20)) é necessário acrescentar um conjunto de restrições que formam um sistema triangular superior ((3.19)), tornando o problema da mochila em um problema de programação linear inteira, pois a estrutura da mochila é perdida (e. g. Costa 2016). A forma desse conjunto de restrições é dada pela matriz (3.21), que pode ser construída a partir das soluções dos primeiros m problemas da mochila, no qual, as linhas representam as faixas e as colunas os itens. Dessa forma, ao fazer a transposta de γ e multiplicar pelo vetor coluna β (solução do problema $m+1$), obtem-se exatamente a quantidade do item i no padrão de corte. Com isso, a restrição (3.19) consegue limitar o número de cada item no padrão de corte utilizando a demanda b.

$$\pi^*_{k=1\ldots m} = Max \sum_{i=1}^{k} \pi_i \cdot \gamma_{ki} \tag{3.13}$$

$$S.a : \sum_{i=1}^{k} l_i \cdot \gamma_{ki} \leq L \tag{3.14}$$

$$\gamma_{ki} \leq b_i,\ i = 1 \ldots k \tag{3.15}$$

$$\gamma_{ki} \in \mathbb{Z}_+, i = 1 \ldots k \tag{3.16}$$

$$Max \sum_{k=1}^{m} \pi^*_k \cdot \beta_k \tag{3.17}$$

$$S.a : \sum_{k=1}^{m} w_k \cdot \beta_k \leq W \tag{3.18}$$

$$\gamma^T \cdot \beta \leq b \tag{3.19}$$

$$\beta_k \in \mathbb{Z}_+, k = 1 \ldots m \tag{3.20}$$

$$\gamma = \begin{bmatrix} \gamma_{11} & 0 & 0 & \cdots & 0 \\ \gamma_{21} & \gamma_{22} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_{m1} & \gamma_{m2} & \gamma_{m3} & \cdots & \gamma_{mm} \end{bmatrix} \tag{3.21}$$

Para resolver os m problemas da mochila é possível utilizar o algoritmo PML^{PD} proposto na Seção 2.4 e resolver todos eles de forma simultânea, considerando que $p = l$, $v = \pi$ e $C = L$. A diferença é que nessa abordagem, é necessário recuperar m

soluções, ao invés de apenas uma como foi feito na mochila limitada. O Algoritmo 3 deve ser modificado para recuperar as m soluções, de forma a obter todos os elementos γ_{ki} da matriz γ. Essa modificação é exibida no Algoritmo 4 em que seu *loop* em k na linha 2 percorre as linhas da matriz γ e $i \geq 1$ define suas colunas. Assim, recupera-se primeiro o elemento da diagonal principal γ_{kk} e depois os elementos das colunas anteriores γ_{ki} para $i = k-1 \ldots 1$. O índice i define a i-ésima coordenada de S em que será buscado o valor de γ_{ki} (linha 9 do Algoritmo 4).

Algoritmo 4: Recuperação de m soluções

Entrada: l, S

Saída: γ_{ki} com $k, i = 1 \ldots m$ e $i \leq k$

1 $\gamma_{11} = S^1(\#S^1, 3)$;
2 **para** $k = 2 \ldots m$ **faça**
3 $\quad i = k-1$;
4 $\quad \gamma_{kk} = S^k(\#S^k, 3)$;
5 $\quad C = S^k(\#S^k, 1)$;
6 $\quad$ **enquanto** $i \geq 1$ **faça**
7 $\quad\quad C = C - l_{i+1}\gamma_{k(i+1)}$;
8 $\quad\quad$ recupere a linha j de S^i tal que $S^i(j,1) = C$;
9 $\quad\quad \gamma_{ki} = S^i(j,3)$;
10 $\quad\quad i = i-1$
11 $\quad$ **fim**
12 **fim**

Para resolver o problema $m+1$ não é possível utilizar o mesmo método desenvolvido pois (3.17)-(3.20) deixou de ser um problema da mochila. Então, nessa pesquisa consideram-se duas alternativas, a primeira é a utilização do solver CPLEX (ILOG 2009) para resolver o problema linear inteiro (algoritmo PC^{PD}_{CPLEX}) e a segunda é utilizar a heurística proposta na Seção 3.6 (algoritmo PC^{PD}_{H}), possibilitando recuperar m soluções. Com isso, o problema de corte bidimensional guilhotinado 2-estágios restrito é resolvido de duas maneiras na pesquisa, sendo elas diferencidas apenas na resolução da segunda etapa do método.

3.6 Heurística para resolver o Problema de Corte Bidimensional 2-estágios Restrito

A heurística desenvolvida (H_{PD}) faz uso do Algoritmo PML^{PD}, utilizando novos limitantes para as sequências S_j^k, e também, inclui um processo de factibilização final para que o padrão de corte seja restrito. Cabe enfatizar que agora os itens são as faixas, então as S_j^k representam a alocação da faixa k com a frequência j no objeto respeitando a largura W. Em (3.22) apresenta-se a variação de j_k (frequência da faixa k) no algoritmo e seu limite que é dado pelo mínimo entre a demanda b_k, o número máximo de vezes que a faixa k cabe no padrão ($\lfloor W/w_k \rfloor$) e uma aproximação para o número de itens i no padrão ($\lfloor b_i/\gamma_{ki} \rfloor, i = 1 \ldots k$). A aproximação para o número de itens no padrão foi motivada pelas desigualdes presentes em (3.23) (restrição (3.19)). Ela é obtida considerando que para cada inequação $i = 1 \ldots k$ somente o escalar γ_{ki} é diferente de zero, conforme é ilustrado na passagem de (3.23) para (3.24). Com isso, o número de vezes que a faixa k pode ser colocada no padrão não deve exceder o $min\{\lfloor b_i/\gamma_{ki} \rfloor, i = 1 \ldots k\}$, pois caso contrário o limite de algum dos itens i para $i = 1 \ldots k$ é excedido ((3.25)).

$$j_k = 0 \ldots min\{b_k, \lfloor W/w_k \rfloor, \lfloor b_i/\gamma_{ki} \rfloor, i = 1 \ldots k\},\ j_k \in \mathbb{Z}\ k = 1 \ldots m \tag{3.22}$$

$$\begin{array}{ccccccccccc}
\gamma_{11}\beta_1 & + & \gamma_{21}\beta_2 & + & \cdots & + & \gamma_{k1}\beta_k & + & \cdots & + & \gamma_{m1}\beta_m \leq b_1 \\
0\beta_1 & + & \gamma_{22}\beta_2 & + & \cdots & + & \gamma_{k2}\beta_k & + & \cdots & + & \gamma_{m2}\beta_m \leq b_2 \\
 & & & & & & \vdots & & & & \\
0\beta_1 & + & 0\beta_2 & + & \cdots & + & \gamma_{kk}\beta_k & + & \cdots & + & \gamma_{mk}\beta_m \leq b_k \\
 & & & & & & \vdots & & & & \\
0\beta_1 & + & 0\beta_2 & + & \cdots & + & 0\beta_k & + & \cdots & + & \gamma_{mm}\beta_m \leq b_m
\end{array} \tag{3.23}$$

$$\begin{array}{c}
\gamma_{k1}\beta_k \leq b_1 \\
\gamma_{k2}\beta_k \leq b_2 \\
\vdots \\
\gamma_{kk}\beta_k \leq b_k
\end{array} \tag{3.24}$$

$$\begin{array}{l} \beta_k \leq b_1/\gamma_{k1} \\ \beta_k \leq b_2/\gamma_{k2} \quad \Rightarrow \quad \beta_k = min\{\lfloor b_i/\gamma_{ki} \rfloor, i = 1 \ldots k\} \\ \quad \vdots \\ \beta_k \leq b_k/\gamma_{kk} \end{array} \tag{3.25}$$

Para utilizar o método da Seção 2.4 foi preciso relaxar o problema (3.17)-(3.20), retirando o conjunto de restrições dados em (3.19), de modo a recuperar a estrutura da mochila. Então, após resolver (3.17),(3.18) e (3.20) por programação dinâmica pelo algoritmo PML^{PD} com os limites dados por (3.22), é possível fazer a recuperação de m soluções pelo Algoritmo 4 obtendo uma matriz β com mesma estrutura de (3.21), ou seja, ao utilizar a programação dinâmica como forma de resolução, gera-se m padrões de corte $A_j, j = 1 \ldots m$ simultaneamente que são dados por $\sum_{k=1}^{j} \gamma_{ki}\beta_{jk}$ para todo $j, i = 1 \ldots m,\ i \leq j$. Com isso, o padrão A_1 terá somente o item 1, o A_2 poderá ter o 1 e 2, até o padrão A_m que poderá ter todos os itens.

No entanto, não há garantia que os padrões gerados sejam restritos. É necessário um processo de factibilização que consiste em retirar os itens em excesso do padrão de corte. Isso é feito iterativamente para todos os padrões verificando se a condição $a_{ij} \leq b_i, j, i = 1 \ldots m,\ i \leq j$ é satisfeita, caso contrário, $a_{ij} = a_{ij} - 1$ até satisfazer a condição, a_{ij} é o elemento linha i no padrão A_j. O pseudocódigo da heurística está descrito no Algoritmo 5. As entradas do algoritmo são os parâmetros para utilizar o algoritmo PML^{PD} e a matriz γ, para obter os padrões de corte, e sua saída é a matriz β e os m padrões de corte. Nas linhas 4-10 é feito a factibilzação dos m padrões obtidos.

Algoritmo 5: Heurística H_{PD}

Entrada: w, π^*, b, W, γ

Saída: β_{jk} com $j,k = 1 \ldots m$ e $k \leq j$ e A_j para $j = 1 \ldots m$

1 Resolva (3.17),(3.18) e (3.20) pelo Algoritmo 1 com os limites dados por (3.22);
2 Recupere as m soluções do problema resolvido na linha 1 pelo Algoritmo 4;
3 Obtenha os m padrões de corte A_j por $\sum_{k=1}^{j} \gamma_{ki}\beta_{jk}$ para todo $j, i = 1 \ldots m,\ i \leq j$;
4 **para** $j = 1 \ldots m$ **faça**
5 **para** $i = 1 \ldots j$ **faça**
6 **enquanto** $a_{ij} > b_i$ **faça**
7 $a_{ij} = a_{ij} - 1$
8 **fim**
9 **fim**
10 **fim**

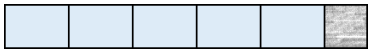
Uma consideração importante de se fazer é que pela estrutura da programação dinâmica, o último padrão de corte seria o de maior valor de utilidade. No entanto, para essa heurística isso não ocorre, pois como o algoritmo não limita totalmente a quantidade dos itens no padrão de corte, ao tornar ele factível para o caso restrito, é possível que se retire muitos itens diminuindo o valor de utilidade do padrão, enquanto que em outros padrões podem ser retirados menos itens, e possivelmente, serem melhores.

3.7 Um exemplo para o Problema de Corte Bidimensional

Considere o problema no qual o objeto tem dimensões $L \times W = 165 \times 70$ e os itens são de dimensões $(l_i \times w_i) = \{(30 \times 23), (45 \times 45), (70 \times 56)\}$ e demanda $b_i = \{5, 6, 2\}$. O objetivo desse exemplo é gerar um padrão de corte restrito, para isso, será utilizado os dois métodos propostos. Para resolver os primeiros $m = 3$ problemas da mochila, utiliza-se do algoritmo PML^{PD} para ambos os métodos, obtendo-se a matriz γ dada em (3.26), as faixas são ilustradas pelas Figuras 3.13-3.15.

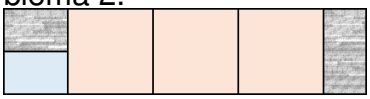
$$\gamma = \begin{bmatrix} 5 & 0 & 0 \\ 1 & 3 & 0 \\ 0 & 2 & 1 \end{bmatrix} \tag{3.26}$$

Figura 3.13: Faixa de valor 3450 gerada pelo problema 1.



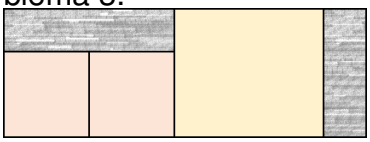
Fonte: autor.

Figura 3.14: Faixa de valor 6765 gerada pelo problema 2.



Fonte: autor.

Figura 3.15: Faixa de valor 7970 gerada pelo problema 3.



Fonte: autor.

A segunda etapa é representada pelo problema (3.27)-(3.32). Veja que as restrições (3.29)-(3.31) formam um sistema triangular superior e a restrição (3.27) é a mesma do PMI. Também, nesse exemplo o valor de utilidade dos itens são suas áreas, então $\pi_1^* = 3450, \pi_2^* = 6765$ e $\pi_3^* = 7970$ em (3.27) representam a área aproveitada de cada faixa. Para resolver o problema, primeiro será utilizado o *solver* CPLEX (ILOG 2009) e em seguida a heurística.

$$Max\ 3450\beta_1 + 6765\beta_2 + 7970\beta_3 \tag{3.27}$$

$$S.\ a:\ 23\beta_1 + 45\beta_2 + 56\beta_3 \leq 70 \tag{3.28}$$

$$5\beta_1 + 1\beta_2 + 0\beta_3 \leq 5 \tag{3.29}$$

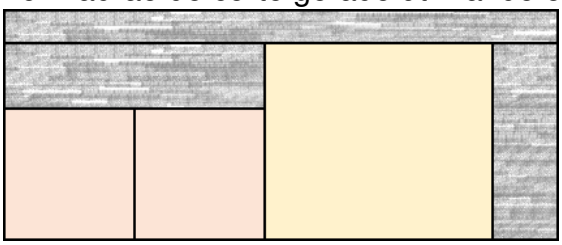
$$0\beta_1 + 3\beta_2 + 2\beta_3 \leq 6 \tag{3.30}$$

$$0\beta_1 + 0\beta_2 + 1\beta_3 \leq 2 \tag{3.31}$$

$$\beta \in \mathbb{Z}_+^3 \tag{3.32}$$

O padrão de corte gerado utilizando o CPLEX na segunda etapa é exibido na Figura 3.16. Nessa solução, foi utilizado uma unidade da faixa 3, ou seja, $\beta = [0\ 0\ 1]^T$. O valor total da solução é 7970 com um aproveitamento percentual de 69,00% da área do objeto.

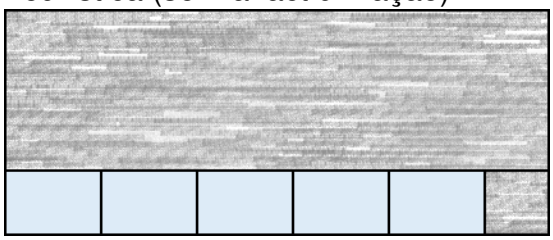
Figura 3.16: Padrão de corte gerado utilizando o CPLEX.



Fonte: autor.

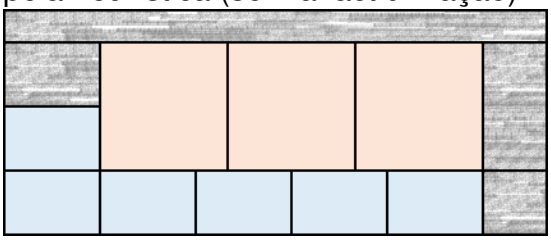
Utilizando a heurística sem a parte de factibilização, são gerados três padrões, sendo os dois últimos iguais e infactíveis para o problema restrito, ilustrados nas Figuras 3.17 e 3.18. Então os valores de β^T são dados em (3.33), no qual, as colunas representam os padrões de corte e as linhas as faixas. Os padrões A_2 e A_3, estão com excesso do item 1, então o *loop* de factibilização é rodado uma vez para deixar os padrões restritos, ilustrados na Figura 3.19. O valor total da solução para A_1 é de 3450 e para A_2 e A_3 é 9525, com aproveitamentos percentuais de 29,87% e 82,47% da área do objeto, respectivamente.

Figura 3.17: Padrão A_1 gerado pela heurística (sem a factibilização).



Fonte: autor.

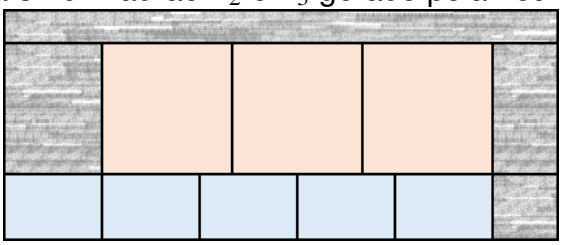
Figura 3.18: Padrão A_2 e A_3 gerado pela heurística (sem a factibilização).



Fonte: autor.

$$\beta^T = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.33}$$

Figura 3.19: Padrão A_2 e A_3 gerado pela heurística.



Fonte: autor.

Perceba que a heurística H_{PD} gera $m = 3$ padrões, mas não necessariamente distintos, isso ocorre devido à programação dinâmica, pois nesse caso, inserir a faixa 3 não implicaria na melhoria da solução presente, então ela se manteve do estágio 2 para o estágio 3. É interessante dizer que a heurística H_{PD} obteve uma solução melhor do que a do CPLEX, o motivo disso é que resolvendo pelo CPLEX, não há a possibilidade de combinar as faixas 1 e 2, pois provoca uma extrapolação da demanda do item 1. Então suas possíveis soluções seriam utilizar a faixa 1 ou a 2 ou a 3, como a terceira é de maior valor, foi ela a escolhida. Por sua vez, a heurística H_{PD} possibilita essa combinação, pois não consegue limitar totalmente o uso das faixas no padrão, carregando então uma solução infactível que foi factibilizada posteriormente gerando uma solução melhor. É importante dizer que nem sempre essa possibilidade de carregar soluções infactíveis irá gerar resultados melhores do que o CPLEX. Além disso, se o CPLEX tivesse disponível as faixas utilizadas na solução final da heurística H_{PD}, os padrões de corte gerados seriam iguais. Isso não ocorre porque o CPLEX só tem as faixas ótimas dos primeiros problemas para combinar e elas nem sempre são as ótimas para o problema original.

Com isso, os algoritmos PC^{PD}_{CPLEX} e PC^{PD}_{H} conseguiram gerar um padrão de corte 2-estágios restrito, e a heurística apresentou um resultado promissor. Dessa forma, no Capítulo 5 são realizados testes computacionais para verificar a qualidade desses métodos em diferentes instâncias, e possivelmente, descobrir suas limitações.

Captulo 4

O PROBLEMA DE CORTE DE ESTOQUE

O problema de corte de estoque (PCE) consiste em atender uma demanda de itens retangulares menores a partir do corte de objetos retangulares maiores em estoque. O critério usual é minimizar o custo total de produção. O modelo geral (1.10)-(1.12) pode ser adaptado da seguinte forma para representar o PCE. O parâmetro a_{ij} representa os itens i que são cortados do objeto j disponível no estoque $\mathscr{J}$ e cada objeto possui um custo c_j. Pelo critério estabelecido, o objetivo é minimizar o custo total de produção ((4.1)), de forma a atender toda a demanda b_i de cada item ((4.2)) utilizando objetos disponíveis em estoque ((4.3)). No parâmetro a_{ij} são garantidas as restrições básicas de contenção do objeto e de não-sobreposição do itens. Nessa pesquisa é tratado o PCE bidimensional, classificado pela tipologia de Wäscher, Haußner e Schumann 2007 como *2-dimensional retangular problema de corte de estoque de tamanho único*.

$$Min \sum_{j \in \mathscr{J}^*} c_j \tag{4.1}$$

$$S.a: \sum_{j \in \mathscr{J}^*} a_{ij} = b_i, \ \forall i \in \mathscr{I} \tag{4.2}$$

$$\mathscr{J}^* \subseteq \mathscr{J} \tag{4.3}$$

O problema de corte de estoque é de grande importância para indústrias em que objetos menores são obtidos de um objeto maior, são exemplos: indústria de vidro, de papel, de confecção e a moveleira (Cherri e Arenales 2005). Essa importância é em parte devido ao custo da matéria prima e de como ela chega na indústria. Na

indústria moveleira o custo da matéria prima (painéis de MDF) é alto (Rangel e Figueiredo 2008), então o desenvolvimento de métodos de resolução que visam minimizar a sobra por perda de material são de grande importância. Como forma de aproveitar ao máximo o uso da matéria prima, tambem há trabalhos que desenvolvem modelos que utilizam de sobras que seriam descartáveis (e.g. Andrade, Birgin e Morabito 2016) e também trabalhos que resolvem o problema considerando defeitos no objeto (e.g. Martin et al. 2019), uma ocorrência que pode acontecer nas fábricas.

4.1 Breve Revisão da Literatura

Nessa pesquisa, o modelo utilizado para o problema de corte de estoque é dado em (4.4)-(4.6), para o atendimento exato da demanda proposto por Gilmore e Gomory (1961, 1963). Para permitir o corte de itens em excesso, trocamos o sinal de " $=$ " em (4.5) por " $\geq$ ". Esse modelo é uma adaptação de (4.1)-(4.3) obtido pela inclusão da variável y_j para representar o número de vezes que o padrão de corte j é cortado. Ao invés de usar o parâmetro a_{ij}, é usado o vetor $A_j \in \mathbb{Z}^m$ que representa melhor a ideia do padrão de corte. Nesse vetor está encapsulado as restrições de contenção e não-sobreposição dos itens. Além disso, como cada objeto a ser cortado possui o mesmo custo, a função objetivo (4.4) representa a minimização da quantidade de objetos cortados. Nesse modelo supõe-se que todos os padrões de corte foram gerados *a priori*.

$$Min \sum_{j=1}^{n} y_j \tag{4.4}$$

$$S.a: \sum_{j=1}^{n} A_j \cdot y_j = b \tag{4.5}$$

$$y_j \geq 0, \ inteiro, \ j = 1 \ldots n \tag{4.6}$$

Na literatura há outras propostas de formulação para o problema do corte de estoque. Carvalho 1999 propõe um modelo de fluxo em arco para resolver o caso unidimensional. Mais recentemente esse modelo foi estendido por Macedo, Alves e Carvalho 2010 para o caso bidimensional, considerando padrões de corte 2-estágios. Enquanto o modelo de Carvalho 1999 utiliza um grafo para representar o padrão de corte, nessa extensão os autores utilizam de dois grafos. Ao fazer a listagem de to-

das as larguras diferentes dos itens, o primeiro grafo determina as faixas no padrão de corte e o segundo quais itens serão cortados dessas faixas. Assim como no caso unidimensional, no caso bidimensional há um excesso de arcos nos grafos. Para fazer uma redução do número de arcos são utilizados três critérios: (1) somente padrões maximais, (2) somente faixas que contenham pelo menos um item de sua largura, e (3) as faixas e os itens são ordenados em forma decrescente de valor. Outro problema encontrado foram as simetrias nos padrões de corte, para evitar isso, além do terceiro critério, que reduz parcialmente as simetrias, foram inseridas duas restrições ao modelo que forçam a ideia de ordenação.

Diferente da proposta do modelo de Gilmore e Gomory 1963, que necessita dos padrões de corte *a priori*, na proposta de Carvalho 1999 e Macedo, Alves e Carvalho 2010, as variáveis de decisão do modelo fornecem os padrões de corte e a respectiva frequência, ou seja as decisões são tomadas simultaneamente. Outro modelo que constrói o padrão e fornece a respectiva frequência é o de Mrad, Meftahi e Haouari 2013. Para o caso bidimensional, os autores propõem um modelo de programação linear inteira para o problema de corte de estoque que gera padrões dois estágios. O modelo também foi baseado na abordagem de Gilmore e Gomory 1965, no entanto, diferente dela, os autores consideram w-padrões e l-padrões. O w-padrão é obtido pela composição de faixas de larguras diferentes e comprimento L e o l-padrão é obtido pela composição de faixas de comprimentos diferentes e largura W. Como esse modelo possui um número muito grande de variáveis, os autores propuseram um método de solução por geração de colunas. Para fazer a geração de coluna, os autores resolveram quatro subproblemas da mochila, dois para gerar as faixas horizontais e verticais e dois para os respectivos padrões. Além disso, para uma melhor convergência da geração de coluna, como os autores resolvem os subproblemas da mochila por programação dinâmica, eles geram mais de duas colunas ao longo de cada iteração. Outra contribuição dos autores foi um algoritmo *branch-and-price*, que utiliza a resolução da relaxação linear e por meio de um grafo de fluxo de arcos, eles adicionam restrições nos subproblemas para as variáveis não inteiras. A ideia dos fluxos em arco é semelhante à de Macedo, Alves e Carvalho 2010.

No problema de corte de estoque há uma questão que chega a ser tão importante quanto o desenvolvimento de novos modelos e técnicas de resolução, que são os fatores humanos e mecânicos de uma indústria que influenciam muito na hora de utilizar as soluções propostas pelos sistemas de solução. Foi pensando nisso que Rangel e Figueiredo 2008 tratam do problema de corte de estoque bidimensional com foco

em indústrias moveleiras de pequeno e médio porte. Os autores fazem uma análise de toda linha de produção da fábrica e exploram os padrões de corte utilizados por ela. A resolução do problema é baseada na abordagem de Gilmore e Gomory 1965, e também, os autores propuseram uma heurística que gera padrões tabuleiros compostos (TC). Esse tipo de padrão de corte é baseado no padrão 1-grupo e também nos padrões cortado pela fábrica. Para obter o padrão TC inicialmente é gerado um padrão 1-grupo maximal, em seguida são retirados faixas do padrão e as áreas de sobras resultantes são preenchidas com outros tipos de faixas. A principal motivação para o desenvolvimento de padrões desse tipo foi atender as necessidades da fábrica no processo de corte, pois padrões 1-grupo tem um custo operacional muito baixo, mas podem gerar perdas altas (e. g. Morabito e Arenales 2000). Ao usar o padrão tabuleiro composto, é possível manter o custo operacional baixo e ainda atender os níveis de perda aceitáveis pela fábrica. No Capítulo 5 será feito uma comparação da Heurística TC com os métodos de resolução propostos para resolver o problema de corte de estoque bidimensional.

Outro trabalho que tem como foco a indústria moveleira é o de Morabito e Arenales 2000. Após os autores fazerem discussões sobre a geração de padrões 2-estágios, 3-estágios e 1-grupo, utilizando modelos e métodos heurísticos, os autores tratam da importância de ter como recurso diferentes tipos de parões de corte e como isso pode influenciar no processo de produção da fábrica. Por exemplo, um padrão de corte 2-estágios é mais rápido para o operador obter os itens, no entanto, é passível de gerar mais perda, enquanto o 3-estágios aproveita melhor o objeto, mas envolve um número maior de preparos da máquina de corte. Essa análise é chamada de *trade-off* entre velocidade de processamento e o aproveitamento do objeto. A partir dos testes realizados, os autores concluíram que o padrão de corte 3-estágios possui a menor perda total, em seguida o 2-estágios não exato e com a maior perda total o 2-estágios exato. Os autores citam que cabe ao tomador de decisão escolher qual estratégia utilizar, dado o período que a indústria se encontra, se há uma baixa procura, então padrões com mais estágios serão mais vantajosos pela menor perda total, caso haja uma alta procura, padrões com um tempo de processamento menor são mais vantajosos, para não ocorrer atrasos e/ou cancelamento de pedidos (Morabito e Arenales 2000).

4.2 Método simplex e a geração de coluna

Para iniciar o estudo do método simplex, primeiro é dado um problema de programação linear genérico em (4.7)-(4.9), no qual, $v \in \mathbb{R}^n$ é um vetor de custos, $b \in \mathbb{R}^m$ um vetor de recursos, $A \in \mathbb{M}[\mathbb{R}]_{m \times n}$ uma matriz de coeficientes e $y \in \mathbb{R}^n_+$ as variáveis do problema. Para tratar o problema, a matriz A pode ser decomposta em duas outras matrizes, sendo uma quadrada inversível (B) chamada matriz básica e a outra não necessariamente quadrada e/ou inversível (N) chamada de não-básica. Fazendo a renomeação dos parâmetros e variáreis de acordo com essa decomposição, o modelo (4.7)-(4.9) pode ser escrito como (4.10)-(4.12).

$$Min \;\; v^T \cdot y \qquad (4.7)$$
$$S.a: \; A \cdot y = b \qquad (4.8)$$
$$y \geq 0 \qquad (4.9)$$

$$Min \;\; v_B^T \cdot y_B + v_N^T \cdot y_N \qquad (4.10)$$
$$S.a: \; B \cdot y_B + N \cdot y_N = b \qquad (4.11)$$
$$y_B, y_N \geq 0 \qquad (4.12)$$

O método simplex basicamente trata de encontrar uma matriz B, de forma inteligente, no qual a solução básica associada seja ótima, quando isso ocorre, B é chamada de base ótima do problema. Para encontrar essa base ótima, o método utiliza um processo iterativo de busca direcionada apresentado no Algoritmo 6.

Nas linhas 1 e 2, são feitos os cálculos da solução básica corrente e do multiplicador simplex, respectivamente. A solução básica e o multiplicador simplex também são chamados de solução primal e dual respectivamente. Uma discussão aprofundada sobre o assunto pode ser vista em Arenales et al. 2015. Na linha 3 calcula-se os custos reduzidos cr (vetor $n-m$-dimensional) e na linha 4 determina-se a variável a entrar na base. Na linha 5 é feito o teste de otimalidade, e caso ele não seja satisfeito, na linha 8 é calculado a direção simplex. Caso a direção simplex tenha todas as entradas negativas, então o problema não possui solução finita. Caso contrário, nas linhas 12 e 13 determina-se a variável a entrar na base. A demonstração desses resultados e métodos para obter uma matriz B inicial que satisfaça (4.11) (base viável) podem ser encontrados em Arenales et al. 2015.

Algoritmo 6: Algoritmo simplex

Entrada: $B, N, y_B, y_N, v_B, v_N, b$

Saída: y^*

1 Calcule a solução básica corrente ($y_N = 0$ e $y_B = B^{-1} \cdot b$);
2 Calcule o multiplicador simplex π ($\pi^T = v_B^T \cdot B^{-1}$);
3 Calcule os custos reduzidos cr ($cr = v_B - \pi^T \cdot N$);
4 Determine a variável a entrar na base (y_{N_j}) $\rightarrow$ variável de menor custo reduzido;
5 Teste a otimalidade da solução básica, **se** $cr \geq 0$ **então**
6 | PARE, $y^* = [y_B^T \;\; y_N^T]^T$ é solução ótima
7 **senão**
8 | Calcule a direção simplex ds ($ds = B^{-1} \cdot N_j$);
9 | **se** $ds \leq 0$ **então**
10 | | PARE, o problema não tem solução ótima finita
11 | **senão**
12 | | Determine o passo θ ($\theta = min\{y_{B_i}/ds_i$, tal que $ds_i > 0, i = 1 \ldots m\}$);
13 | | Determine a variável a sair da base (y_{B_i}) $\rightarrow$ variável referente ao passo mínimo;
14 | | Atualize a solução básica e volte à linha 1
15 | **fim**
16 **fim**

Em alguns casos, o número de colunas da matriz A é muito maior do que o número de linhas ($n \gg m$). Isso pode inviabilizar o método simplex, pois haverá um número muito grande de colunas não-básicas para fazer a escolha da melhor a ser inserida na base. Dessa forma, o método de geração de coluna surgiu como alternativa para contornar esse problema, ao invés de trabalhar com o problema original, que possui muitas colunas, trabalha-se com o problema mestre restrito (PMR). Para se ter uma vizualização melhor do método, o problema (4.7)-(4.9) será reescrito como (4.13)-(4.15) e considerando apenas um subconjunto $r \geq m$ de colunas de A (Problema Mestre Restrito).

$$Min \sum_{j=1}^{r} v_j \cdot y_j \tag{4.13}$$

$$S.a: \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} \cdot y_1 + \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} \cdot y_2 + \cdots + \begin{bmatrix} a_{1r} \\ a_{2r} \\ \vdots \\ a_{mr} \end{bmatrix} \cdot y_r = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{4.14}$$

$$y_j \geq 0, j = 1 \ldots r \tag{4.15}$$

Para o processo de geração de colunas, é feito uma nova partição na matriz A. Ela é decomposta em três matrizes B, N' e N'', sendo as duas primeiras conhecidas (básica e não-básica) e a última não conhecida (não-básica) no qual serão gerados novas colunas. O processo de geração de colunas é dado no Algoritmo 7, veja que as entradas e saída desse algoritmo em relação ao simplex, são as mesmas, isso porque na linha 1, é utilizado o método simplex para resolver o PMR, então nela está encapsulado o Algoritmo 6. Na linha 5 encontra-se o subproblema que deve ser resolvido para gerar uma coluna. Esse subproblema, chamado de subproblema *pricing*, não é dado de forma explícita porque depende da aplicação. Nessa pesquisa, será visto que o subproblema *pricing* para o problema de corte de estoque bidimensional é um problema de corte bidimensional igual ao tratado no Capítulo 3. A verificação para saber se a coluna gerada entrará no PMR é dado nas linhas 6-10 e utiliza do custo reduzido associado a essa coluna para fazer a verificação. As demonstrações desses resultados e melhores discussões encontram-se em Arenales et al. 2015.

Algoritmo 7: Algoritmo de geração de coluna

Entrada: $B, N', y_B, y_{N'}, v_B, v_{N'}, b$

Saída: $y^* \in \mathbb{R}$

1 Resolva o problema mestre restrito até otimalidade e calcule y_r^* e π_r^* pelo Algoritmo 6;
2 **se** *Se PMR não tem solução ótima* **então**
3 | PARE, problema mestre não tem solução ótima
4 **senão**
5 | Resolva o subproblema $Min\left\{cr(A_j) = v(A_j) - \pi_r^* \cdot A_j \mid A_j \text{ é coluna de } N''\right\}$ e obtenha A_j^*;
6 | **se** $cr(A_j^*) \geq 0$ **então**
7 | | PARE, a solução y_r^* é ótima
8 | **senão**
9 | | $r = r + 1$;
10 | | Insira a coluna A_j^* no PMR e volte à linha 1
11 | **fim**
12 **fim**

Um ponto para se dizer do algoritmo de geração de colunas é a importância do subproblema *pricing*, pois quase toda a estrutura é igual ao algoritmo simplex. Então o que definirá a qualidade e velocidade do método de geração de colunas é a rapidez em que o subproblema é resolvido.

A seguir, será explicado como é feito a geração de colunas no problema de corte de estoque utilizando o modelo de Gilmore e Gomory (1961, 1963), e também, explica-se como o subproblema *pricing* presente na geração de colunas é resolvido utilizando os métodos do Capítulo 3.

4.3 Geração de colunas no problema de corte de estoque bidimensional

O modelo utilizado nesta pesquisa para resolver o problema de corte de estoque bidimensional é o modelo de Gilmore e Gomory (1961, 1963) apresentado em (4.4)-(4.6). No processo de geração de colunas apresentado na Seção 4.2, ele é tratado todo em cima de um problema de programação linear. Como o modelo de Gilmore e Gomory (1961, 1963) possui variáveis inteiras, é necessário utilizar a relaxação li-

near desse modelo que é obtida fazendo o domínio das variáveis de decisão y_j serem iguais aos reais ($y_j \in \mathbb{R}, j = 1 \ldots n$). Dessa forma, é possível utilizar o Algoritmo 7.

O subproblema *pricing* (linha 5 do Algoritmo 7) da geração de colunas no problema de corte de estoque é alterado para (4.16). O parâmetro $v(A_j)$ no problema de corte de estoque é sempre 1, pois o custo do objeto é 1. Após isso, o subproblema pode ser manipulado de forma a obter (4.17). Com isso, o subproblema *pricing* passou a ser um problema de corte que procura gerar um padrão com maior valor de utilidade. O valor de utilidade de cada item é dado pelas entradas do vetor multiplicador simplex.

$$Min\left\{cr(A_j) = 1 - \pi^* \cdot A_j \mid A_j \text{ é um padrão de corte}\right\} \tag{4.16}$$

$$Min\left\{cr(A_j) = 1 - \pi^* \cdot A_j\right\} = 1 - Max\left\{\pi^* \cdot A_j \mid A_j \text{ é um padrão de corte}\right\} \tag{4.17}$$

Dado essas modificações no subproblema, ele somente apresentará uma forma totalmente explícita após determinar qual será o tipo de padrão de corte utilizado. Então, se o objetivo é gerar um padrão 2-estágios, ele terá uma formulação, caso queira um 3-estágios, será outra e bastará alterar as restrições do subproblema *pricing*. Como o objetivo da pesquisa é gerar padrões de corte bidimensionais guilhotinados 2-estágios restritos, a formulação utilizada é a apresentada na Seção 3.5 que utiliza a abordagem em duas etapas pelos modelos (3.13)-(3.16) e (3.17)-(3.20).

Como base inicial para o Algoritmo 7, será utilizado a matriz de identidade, ou seja, m padrões de corte diferentes no qual cada um possui apenas uma unidade e um tipo de item. Com isso, o problema mestre restrito é dado por (4.18)-(4.20).

$$Min \sum_{j=1}^{m} y_j \tag{4.18}$$

$$S.a: \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{4.19}$$

$$y_j \geq 0, j = 1 \ldots m \tag{4.20}$$

A resolução desse primeiro PMR é trivial, obtendo solução igual à demanda e multiplicador simplex com entradas de valor 1. Para gerar uma nova coluna, utiliza-se o algoritmo PML^{PD} para resolver os m subproblemas da mochila dados por (3.13)-(3.16) e para resolver o subproblema (3.17)-(3.20) usa-se o CPLEX (algoritmo PC^{PD}_{CPLEX}) para resolvê-lo de forma exata ou a heurística H_{PD} (algoritmo PC^{PD}_{H}) possibilitando gerar m padrões para serem inseridos no PMR. É importante dizer que ao gerar mais de uma coluna simultaneamente, apenas as que possuem valor de utilidade maior do que 1 são inseridas no PMR.

Note que a saída do Algoritmo 7, em geral não é solução para o problema modelado por (4.4)-(4.6), porque o algoritmo obtêm solução real. Para obter a solução inteira é necessário empregar o método *Branch-and-Price* ou arredondar a solução para baixo e aplicar um método que resolva o problema residual (e. g. Chaves, Rangel e Poldi 2019). Nessa pesquisa, o problema (4.4)-(4.6) será resolvido com folga de itens ("$\geq$"), então após obter a solução real do Algoritmo 7, as variáveis não-inteiras serão arrendondadas para cima, garantindo que a demanda dos itens seja atendida.

4.4 Um exemplo para o Problema de Corte de Estoque Bidimensional

Considere o mesmo problema do Capítulo 3, no qual o objeto tem dimensões $L \times W = 165 \times 70$ e os itens são de dimensões $(l_i \times w_i) = \{(30 \times 23), (45 \times 45), (70 \times 56)\}$ e demanda $b_i = \{5, 6, 2\}$. Agora, o objetivo é atender toda a demanda dos itens de forma a utilizar o menor número de objetos. O subproblema *pricing* será resolvido de duas formas, a primeira utilizando o algoritmo PC^{PD}_{CPLEX} e a segunda o algoritmo PC^{PD}_{H}. Então, o PMR para esse problema é dado por (4.21)-(4.23) e ao resolvê-lo, obtêm-se $y^* = [5\ 6\ 2]^T$ e $\pi^* = [1\ 1\ 1]^T$.

$$Min\ y_1 + y_2 + y_3 \tag{4.21}$$

$$S.a: \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 5 \\ 6 \\ 2 \end{bmatrix} \tag{4.22}$$

$$y_1, y_2, y_3 \geq 0 \tag{4.23}$$

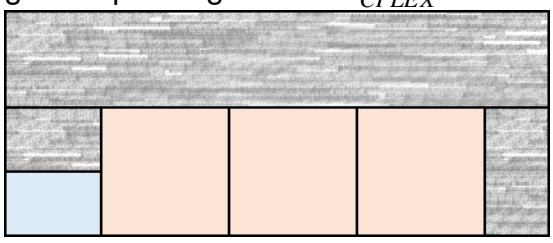
Utilizando o algoritmo PC^{PD}_{CPLEX}, o PMR que resultou na solução ótima para o problema é dado em (4.24)-(4.26), com solução $y^* = \left[0\ 0\ 0\ 0\ 1.4\bar{5}\ 0\ 1.1\bar{8}\ 0.8\bar{1}\right]^T$, ou seja, foram geradas cinco colunas. Como o problema de corte de estoque é inteiro, é preciso arredondar $y_5 = 1.4\bar{5}$ para $y_5 = 2$, $y_7 = 1.1\bar{8}$ para $y_7 = 2$ e $y_8 = 0.8\bar{1}$ para $y_8 = 1$, então foram necessários cortar 5 objetos para atender toda a demanda. Perceba que o padrão de corte obtido no Capítulo 3 pelo CPLEX, foi o último gerado pelo processo de geração de coluna, isso ocorre porque o valor de utilidade dos itens é dado pelo método simplex e não está diretamente relacionado com sua área aproveitada do padrão de corte, sendo possível até mesmo não gerar uma coluna igual à solução do Capítulo 3. Nas Figuras 3.17, 4.1 - 4.3 e 3.16 são ilustrados, respectivamente, os padrões de corte gerados utilizando o CPLEX.

$$Min \sum_{j=1}^{8} y_j \tag{4.24}$$

$$S.a: \begin{bmatrix} 1 & 0 & 0 & 5 & 1 & 0 & 3 & 0 \\ 0 & 1 & 0 & 0 & 3 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 0 & 2 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_8 \end{bmatrix} = \begin{bmatrix} 5 \\ 6 \\ 2 \end{bmatrix} \tag{4.25}$$

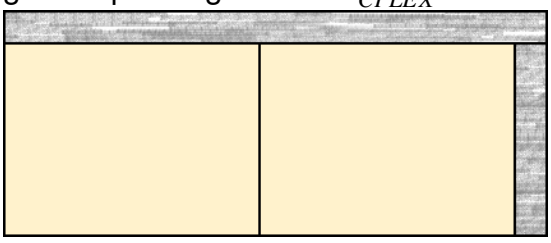
$$y_j \geq 0, j = 1 \ldots 8 \tag{4.26}$$

Figura 4.1: Segundo padrão de corte gerado pelo algoritmo PC^{PD}_{CPLEX}.



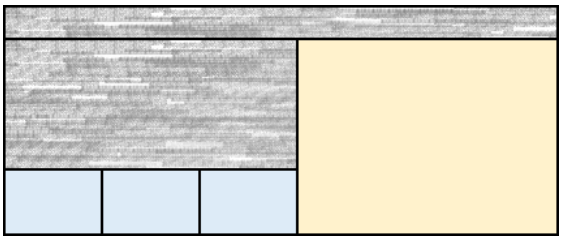
Fonte: autor.

Figura 4.2: Terceiro padrão de corte gerado pelo algoritmo PC^{PD}_{CPLEX}.



Fonte: autor.

Figura 4.3: Quarto padrão de corte gerado pelo algoritmo PC^{PD}_{CPLEX}.



Fonte: autor.

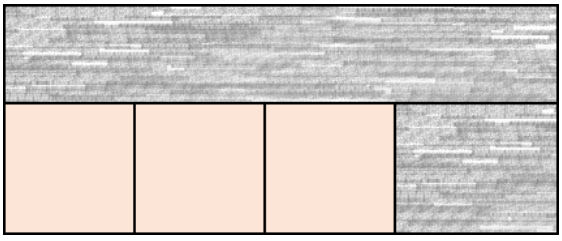
Usando o algoritmo PC^{PD}_{H}, o PMR que resultou na solução ótima para o problema é dado em (4.27)-(4.29), com solução $y^* = [0\ 0\ 0\ 0\ 1\ 0\ 0.25\ 1.5]^T$, então foram geradas cinco colunas. É necessário arredondar as variáveis não-inteiras, dessa forma, $y_7 = 1$ e $y_8 = 2$, ou seja, foram cortados 4 objetos para atender toda a demanda. Perceba que tanto a solução da relaxação do PMR quanto a solução arredondada, na heurística os valores são menores, $y^H_{RL} = 2.75 < y^S_{RL} = 3.4\bar{5}$ e $y^H_I = 4 < y^S_I = 5$. Essa diferença foi motivada pela geração de um padrão de corte mais proveitoso na segunda iteração do método, ilustrado na Figura 3.19 (mesma solução do exemplo para o problema de corte). Outro padrão que foi gerado, diferente dos anteriores é dado pela Figura 4.4.

$$Min \sum_{j=1}^{8} y_j \tag{4.27}$$

$$S.a: \begin{bmatrix} 1 & 0 & 0 & 5 & 5 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 3 & 3 & 0 & 2 \\ 0 & 0 & 1 & 0 & 0 & 0 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_8 \end{bmatrix} = \begin{bmatrix} 5 \\ 6 \\ 2 \end{bmatrix} \tag{4.28}$$

$$y_j \geq 0, j = 1 \ldots 8 \tag{4.29}$$

Figura 4.4: Terceiro padrão de corte gerado pelo algoritmo PC^{PD}_{H}.



Fonte: autor.

O algoritmo PML^{PD} desenvolvido no Capítulo 2 e o algoritmo PC_H^{PD} do Capítulo 3 mostraram-se promissores para a geração de coluna do problema de corte de estoque. É importante enfatizar que o PMR está sendo resolvido de forma exata, no entanto, a solução ótima é arredondada para cima atendendo a demanda com sobras. No Capítulo 5 serão realizados testes computacionais para verificar a qualidade desse método em diferentes instâncias, e possivelmente, descobrir suas limitações.

Captulo 5

ESTUDO COMPUTACIONAL

Neste capítulo são apresentados os resultados do estudo computacional realizado para avaliar o desempenho dos algoritmos PML^{PD}, PC^{PD}_{CPLEX} e PC^{PD}_{H}.

O estudo foi dividido em três partes. Na primeira parte, como forma de validar o algoritmo PML^{PD}, usou-se o solver CPLEX versão 12.1 (ILOG 2009) para resolver instâncias do modelo (2.4)-(2.7) nomeado como PML_{CPLEX}, comparando o valor das soluções e tempo computacional. Outro objetivo desse estudo foi analisar como o algoritmo se comporta dentro do cenário de geração de faixas e alocação de faixas para a geração de um padrão de corte. Com isso, verificar pelo número de estados da programação dinâmica se esses cenários são desafiadores. Para isso, foram adaptadas de instâncias do problema de corte e do corte de estoque da literatura. As instâncias usadas nessa parte do estudo estão descritas na Seção 5.1.1 e os resultados na Seção 5.2.

Na segunda parte do estudo, o objetivo foi fazer uma comparação entre o desempenho dos algoritmos PC^{PD}_{CPLEX} e o PC^{PD}_{H} para resolver o problema de corte bidimensional 2-estágios restrito. Para tal, são analisados o valor das soluções obtidas e o tempo de processamento. Para esse estudo foram utilizadas 18 instâncias da literatura que possuem soluções ótimas conhecidas. As instâncias estão descritas na Seção 5.1.2 e os resultados associados na Seção 5.3.

Na terceira parte do estudo, o objetivo foi verificar se os algoritmos PC^{PD}_{CPLEX} e PC^{PD}_{H} são eficientes no processo de geração de colunas usado para resolver o PCE bidimensional. Os resultados obtidos também são comparados com o sistema CorteBiFur e com a Heurística TC (Rangel e Figueiredo 2008). Para esse estudo, foram utilizadas instâncias baseadas em dados reais de uma fábrica de móveis (Rangel e

Figueiredo 2008). As instâncias usadas são descritas na Seção 5.1.3 e os resultados na Seção 5.4.

5.1 Instâncias

O estudo computacional foi realizado usando um computador Windows 10 com processador Core i7 de CPU 3.20 GHz, memória RAM de 16 GB e sistema operacional de 64 bits. Os algoritmos $PML^{PD}, PML_{CPLEX}, PC^{PD}_{CPLEX}$ e PC^{PD}_{H} foram codificados na linguagem Júlia e na linguagem de modelagem JuMP. O solver utilizado foi o CPLEX versão 12.1.

Nesta seção são apresentadas as instâncias utilizadas no estudo computacional divididas por tipo de problema: PM (Seção 5.1.1), PC (Seção 5.1.2) e PCE (Seção 5.1.3). Para as instâncias que possuem um número elevado de itens (> 10) são apresentados o valor mínimo Mn e o valor máximo Mx de cada parâmetro dentro de um intervalo. As instâncias foram retiradas dos repositórios OR-Benchmark, OR-Library 1990, Wang 1983 e Rangel e Figueiredo 2008.

5.1.1 Instâncias para o Problema da mochila limitada

As 62 instâncias usadas para avaliar os algoritmos propostos para resolver o problema da mochila limitada foram obtidas a partir de instâncias para o problema de corte obtidos na $OR-Library, OR-Benchmark$, Wang 1983 e Rangel e Figueiredo 2008. Para cada instância do problema de corte e do corte de estoque foram geradas duas instâncias para a mochila limitada. As instâncias l(nome da instância) são referentes ao comprimento dos itens e objeto dos problema bidimensionais, enquanto as instâncias w(nome da instância) são referentes à largura. O valor de cada item é dado pela sua área. Por exemplo, a instância OF1 do problema de corte possui 10 itens, objeto de dimensão (70×40) e o primeiro item possui dimensões e demanda igual a $(29 \times 5 \times 1)$. Com isso, gerou-se a instância lOF1 com 10 itens, capacidade da mochila $C = 70$ e o primeiro item recebe valor, peso e limite iguais a $(145 \times 29 \times 1)$. A instância wOF1 tem 10 itens, capacidade da mochila $C = 40$ e o primeiro item possui valor, peso e limite iguais a $(145 \times 5 \times 1)$. Na adaptação das instâncias do problema de corte de estoque, além de fazer esse processo, foram gerados aleatoriamente o limite dos itens dentro do itervalo $[5, 10]$. Para a fase de validação do algoritmo PML^{PD},

foram selecionadas um subconjunto das instâncias.

Nas Tabelas 5.1 a 5.5 são descritas nas colunas 1-4 o nome, a capacidade da mochila (C), o número de itens distintos (m) e o total de itens (n) para cada instância, respectivamente. Na coluna 5 apresenta-se o valor, peso e limite para os itens ou a variação mínima e máxima desses parâmetros para cada instância. Na Tabela 5.1, apresenta-se os dados adaptados das instâncias OF, cgcut e wang20 (OR-Library 1990, OR-Benchmark, Wang 1983). Na Tabela 5.2 e 5.3 exibi-se os dados referentes às instâncias lgcut e wgcut adaptadas de (OR-Library 1990). Nas Tabelas 5.4 e 5.5 é apresentado os dados das instâncias lRF e wRF adaptadas das instâncias do problema do corte de estoque presente em Rangel e Figueiredo 2008.

Tabela 5.1: Instâncias (l e w)OF, (l e w)cgcut e (l e w)wang20 para o problema da mochila limitada (adaptadas de OR-Library 1990, OR-Benchmark, Wang 1983).

Nome	C	m	n	$(v_i \times p_i \times b_i)$
lOF1	70	10	23	$(145\times29\times1),(351\times9\times4),(495\times55\times1),(465\times31\times1),$ $(176\times11\times2),(483\times23\times3),(406\times29\times4),(304\times16\times3),$ $(324\times9\times2),(88\times22\times2)$
lOF2	70	10	24	$(396\times22\times2),(400\times40\times1),(351\times13\times3),(414\times23\times2)$ $(232\times29\times4),(64\times16\times1),(423\times47\times1),(361\times19\times4),$ $(208\times13\times2),(576\times36\times4)$
wOF1	40	10	23	$(145\times5\times1),(351\times39\times4),(495\times9\times1),(465\times15\times1),$ $(176\times16\times2),(483\times21\times3),(406\times14\times4),(304\times19\times3),$ $(324\times36\times2),(88\times4\times2)$
wOF2	40	10	24	$(396\times18\times20,(400\times10\times1),(351\times27\times3),(414\times18\times2)$ $(232\times8\times4),(64\times4\times1),(423\times9\times1),(361\times19\times4),$ $(208\times16\times2),(576\times16\times4)$
lcgcut1	15	7	16	$(66\times8\times2),(35\times3\times1),(24\times8\times3),(17\times3\times5),$ $(11\times3\times2),(8\times3\times2),(2\times2\times1)$
lcgcut2	40	10	23	$(582\times21\times1),(403\times31\times1),(315\times9\times3),(216\times9\times3),$ $(210\times30\times2),(143\times11\times3),(140\times10\times1),(110\times14\times3),$ $(94\times12\times3),(90\times13\times3)$
lcgcut3	40	20	62	$v_i \in [140,500]$, $p_i \in [9,33]$, $b_i \in [1,4]$
wcgcut1	10	7	16	$(66\times4),(35\times7\times1),(24\times2\times3),(17\times4\times5),$ $(11\times3\times2),(8\times2\times2),(2\times1\times1)$
wcgcut2	70	10	23	$(582\times22\times1),(403\times13\times1),(315\times35\times3),(216\times24\times3),$ $(210\times7\times2),(143\times13\times3),(140\times14\times1),(110\times8\times3),$ $(94\times8\times3),(90\times7\times3)$
wcgcut3	70	20	62	$v_i \in [140,500]$, $p_i \in [11,43]$, $b_i \in [1,4]$
lwang20	70	20	42	$v_i \in [153,1333]$, $p_i \in [11,43]$, $b_i \in [1,4]$
wwang20	40	20	42	$v_i \in [153,1333]$, $p_i \in [9,33]$, $b_i \in [1,4]$

Tabela 5.2: Instâncias lgcut para o problema da mochila limitada (adaptada de OR-Library 1990).

Nome	C	m	$(v_i \times p_i \times b_i)$
lgcut1	250	10	$(6020 \times 70 \times 1), (13452 \times 114 \times 1), (11620 \times 83 \times 1),$ $(12267 \times 87 \times 1), (9768 \times 66 \times 1), (25384 \times 167 \times 1),$ $(19200 \times 120 \times 1), (11385 \times 69 \times 1), (23738 \times 143 \times 1),$ $(30728 \times 167 \times 1)$
lgcut2	250	20	$v_i \in [5610, 25110]$, $p_i \in [66, 168]$, $b_i = 1$
lgcut3	250	30	$v_i \in [5041, 24776]$, $p_i \in [63, 176]$, $b_i = 1$
lgcut4	250	50	$v_i \in [4554, 31325]$, $p_i \in [62, 184]$, $b_i = 1$
lgcut5	500	10	$(51040 \times 352 \times 1), (39984 \times 272 \times 1), (51642 \times 342 \times 1),$ $(27745 \times 179 \times 1), (22968 \times 132 \times 1), (40590 \times 198 \times 1),$ $(85904 \times 364 \times 1), (84035 \times 343 \times 1), (41000 \times 164 \times 1),$ $(100392 \times 282 \times 1)$
lgcut6	500	20	$v_i \in [24948, 95465]$, $p_i \in [132, 355]$, $b_i = 1$
lgcut7	500	30	$v_i \in [18963, 136510]$, $p_i \in [129, 365]$, $b_i = 1$
lgcut8	500	50	$v_i \in [18864, 135388]$, $p_i \in [131, 362]$, $b_i = 1$
lgcut9	1000	10	$(116963 \times 343 \times 1), (132060 \times 310 \times 1), (132860 \times 292 \times 1),$ $(197238 \times 426 \times 1), (314964 \times 673 \times 1), (177742 \times 362 \times 1),$ $(161850 \times 325 \times 1), (299700 \times 555 \times 1), (278613 \times 459 \times 1),$ $(209840 \times 305 \times 1)$
lgcut10	1000	20	$v_i \in [125589, 537992]$, $p_i \in [381, 545]$, $b_i = 1$
lgcut11	1000	30	$v_i \in [90706, 403045]$, $p_i \in [266, 674]$, $b_i = 1$
lgcut12	1000	50	$v_i \in [94336, 527067]$, $p_i \in [254, 746]$, $b_i = 1$

Tabela 5.3: Instâncias wgcut para o problema da mochila limitada (adaptada de OR-Library 1990).

Nome	C	m	$(v_i \times p_i \times b_i)$
wgcut1	250	10	$(6020 \times 86 \times 1), (13452 \times 118 \times 1), (11620 \times 140 \times 1),$ $(12267 \times 141 \times 1), (9768 \times 148 \times 1), (25384 \times 152 \times 1),$ $(19200 \times 160 \times 1), (11385 \times 165 \times 1), (23738 \times 166 \times 1),$ $(30728 \times 184 \times 1)$
wgcut2	250	20	$v_i \in [5610, 25110]$, $p_i \in [68, 186]$, $b_i = 1$
wgcut3	250	30	$v_i \in [5041, 25110]$, $p_i \in [68, 186]$, $b_i = 1$
wgcut4	250	50	$v_i \in [4554, 31325]$, $p_i \in [63, 179]$, $b_i = 1$
wgcut5	500	10	$(51040 \times 145 \times 1), (39984 \times 147 \times 1), (51642 \times 151 \times 1),$ $(27745 \times 155 \times 1), (22968 \times 174 \times 1), (40590 \times 205 \times 1),$ $(85904 \times 236 \times 1), (84035 \times 245 \times 1), (41000 \times 250 \times 1),$ $(100392 \times 356 \times 1)$
wgcut6	500	20	$v_i \in [24948, 95465]$, $p_i \in [134, 372]$, $b_i = 1$
wgcut7	500	30	$v_i \in [18963, 136510]$, $p_i \in [147, 374]$, $b_i = 1$
wgcut8	500	50	$v_i \in [18864, 135388]$, $p_i \in [127, 374]$, $b_i = 1$
wgcut9	1000	10	$(116963 \times 341 \times 1), (132060 \times 426 \times 1), (132860 \times 455 \times 1),$ $(197238 \times 463 \times 1), (314964 \times 468 \times 1), (177742 \times 491 \times 1),$ $(161850 \times 498 \times 1), (299700 \times 540 \times 1), (278613 \times 607 \times 1),$ $(209840 \times 688 \times 1)$
wgcut10	1000	20	$v_i \in [125580, 537992]$, $p_i \in [260, 742]$, $b_i = 1$
wgcut11	1000	30	$v_i \in [90706, 403045]$, $p_i \in [274, 745]$, $b_i = 1$
wgcut12	1000	50	$v_i \in [94336, 527067]$, $p_i \in [269, 723]$, $b_i = 1$

Tabela 5.4: Instâncias IRF para o problema da mochila limitada (adaptadas de Rangel e Figueiredo 2008).

Nome	C	m	$(v_i \times p_i \times b_i)$
IRF1	2750	3	$(94785 \times 445 \times 9), (157030 \times 410 \times 7), (146276 \times 388 \times 5)$
IRF2	2750	2	$(42900 \times 390 \times 6), (40700 \times 370 \times 9)$
IRF3	2750	2	$(264000 \times 600 \times 8), (59400 \times 450 \times 8)$
IRF4	2750	3	$(379850 \times 710 \times 5), (562860 \times 1062 \times 5), (293091 \times 647 \times 9)$
IRF5	2750	3	$(31500 \times 630 \times 7), (21650 \times 433 \times 9), (14750 \times 295 \times 8)$
IRF6	2750	6	$(28600 \times 440 \times 6), (31750 \times 635 \times 6), (81720 \times 454 \times 6)$ $(114300 \times 635 \times 5), (61290 \times 454 \times 9), (85725 \times 635 \times 7)$
IRF7	2750	7	$(552900 \times 970 \times 6), (52500 \times 700 \times 8), (148400 \times 700 \times 9),$ $(114100 \times 700 \times 9), (240000 \times 600 \times 5), (25800 \times 430 \times 8),$ $(30000 \times 500 \times 8)$
IRF8	2750	8	$(1412500 \times 2500 \times 8), (293091 \times 647 \times 6), (322340 \times 710 \times 8),$ $(206116 \times 454 \times 5), (490320 \times 1080 \times 7), (240620 \times 530 \times 7),$ $(525000 \times 1050 \times 9), (103845 \times 483 \times 6)$
IRF9	2750	4	$(229500 \times 510 \times 8), (31500 \times 630 \times 5), (21650 \times 433 \times 6),$ $(14750 \times 295 \times 8)$
IRF10	2750	3	$(81720 \times 454 \times 9), (114300 \times 635 \times 6), (61290 \times 454 \times 9)$
IRF11	2750	9	$(474148 \times 1049 \times 9), (225548 \times 499 \times 9), (193908 \times 452 \times 6),$ $(561750 \times 1050 \times 5), (267500 \times 535 \times 8), (230050 \times 535 \times 5),$ $(148400 \times 700 \times 8), (25800 \times 430 \times 5), (30000 \times 500 \times 7)$
IRF12	2750	9	$(150000 \times 2500 \times 7), (26700 \times 445 \times 8), (17800 \times 445 \times 9),$ $(29400 \times 490 \times 8), (30000 \times 500 \times 6), (63000 \times 1050 \times 9)$ $(25800 \times 430 \times 5), (26400 \times 440 \times 9), (63600 \times 1060 \times 5)$
IRF13	2750	3	$(25800 \times 430 \times 7), (30000 \times 500 \times 5), (63000 \times 1050 \times 7)$

Tabela 5.5: Instâncias wRF para o problema da mochila limitada (adaptadas de Rangel e Figueiredo 2008).

Nome	C	m	$(v_i \times p_i \times b_i)$
wRF1	1830	3	$(94785 \times 213 \times 9), (157030 \times 383 \times 9), (146276 \times 377 \times 5)$
wRF2	1830	2	$(42900 \times 110 \times 7), (40700 \times 110 \times 5)$
wRF3	1850	2	$(264000 \times 440 \times 5), (59400 \times 132 \times 8)$
wRF4	1830	3	$(379850 \times 535 \times 5), (562860 \times 530 \times 9), (293091 \times 453 \times 7)$
wRF5	1830	3	$(31500 \times 50 \times 6), (21650 \times 50 \times 7), (14750 \times 50 \times 9)$
wRF6	1830	6	$(28600 \times 65 \times 5), (31750 \times 50 \times 6), (81720 \times 180 \times 9),$ $(114300 \times 180 \times 5), (61290 \times 135 \times 9), (85725 \times 135 \times 8)$
wRF7	1850	7	$(552900 \times 570 \times 8), (52500 \times 75 \times 6), (148400 \times 212 \times 8),$ $(114100 \times 163 \times 8),$ $(240000 \times 400 \times 5), (25800 \times 60 \times 9), (30000 \times 60 \times 9)$
wRF8	1830	8	$(1412500 \times 565 \times 6), (293091 \times 453 \times 9), (322340 \times 454 \times 7),$ $(206116 \times 454 \times 5), (490320 \times 454 \times 7), (240620 \times 454 \times 6),$ $(525000 \times 500 \times 5), (103845 \times 215 \times 5)$
wRF9	1830	4	$(229500 \times 450 \times 7), (31500 \times 50 \times 7), (21650 \times 50 \times 5),$ $(14750 \times 50 \times 6)$
wRF10	1830	3	$(81720 \times 180 \times 9), (114300 \times 180 \times 5), (61290 \times 135 \times 6)$
wRF11	1850	9	$(474148 \times 452 \times 8), (225548 \times 452 \times 5), (193908 \times 429 \times 6),$ $(561750 \times 535 \times 7), (267500 \times 500 \times 8), (230050 \times 430 \times 9),$ $(148400 \times 212 \times 9), (25800 \times 60 \times 9), (30000 \times 60 \times 8)$
wRF12	1830	9	$(150000 \times 60 \times 7), (26700 \times 60 \times 8), (17800 \times 40 \times 8),$ $(29400 \times 60 \times 6), (30000 \times 60 \times 5), (63000 \times 60 \times 8),$ $(25800 \times 60 \times 6), (26400 \times 60 \times 9), (63600 \times 60 \times 5)$
wRF13	1830	3	$(25800 \times 60 \times 5), (30000 \times 60 \times 6), (63000 \times 60 \times 5)$

5.1.2 Instâncias para o Problema de corte bidimensional 2-estágios restrito

As 18 instâncias para o problema de corte bidimensional 2-estágios restrito foram obtidas nos repositórios *OR-Brenchmark*, Wang 1983 e *OR-Library* e são nomeadas como: OF1, OF2, cgcut1, cgcut2, cgcut3, wang20 e gcut1-gcut12. É importante dizer que as instâncias gcut1-gcut12 foram propostas inicialmente para o problema irrestrito, e adaptadas acrescentando a demanda dos itens igual a 1. No entanto, conforme a tipologia de Wäscher, Haußner e Schumann 2007, elas deixam de ser instâncias para o *2-dimensional retangular problema de posicionamento de objeto único* e passam a ser instâncias para o *2-dimensional retangular problema da mochila única*, porque caracterizam um problema da mochila binária bidimensional.

Na Tabela 5.6 apresenta-se as características das instâncias utilizadas. Nas colunas 1-3, é apresentado o nome das instâncias e o comprimento e largura do objeto, respectivamente. Na coluna 4 são dados o número de itens distintos (m) e na coluna 5 o total de itens para cada instância (n).

Tabela 5.6: Instâncias para o problema de corte bidimensional 2-estágios restrito.

Nome	L	W	m	n
OF1	70	40	10	23
OF2	70	40	10	24
cgcut1	15	10	7	16
cgcut2	40	70	10	23
cgcut3	40	70	20	62
wang20	70	40	20	42
gcut1	250	250	10	10
gcut2	250	250	20	20
gcut3	250	250	30	30
gcut4	250	250	50	50
gcut5	500	500	10	10
gcut6	500	500	20	20
gcut7	500	500	30	30
gcut8	500	500	50	50
gcut9	1000	1000	10	10
gcut10	1000	1000	20	20
gcut11	1000	1000	30	30
gcut12	1000	1000	50	50

5.1.3 Instâncias para o Problema do corte de estoque bidimensional

As 13 instâncias usadas para avaliar o método de geração de colunas para resolver o problema do corte de estoque bidimensional foram obtidas no trabalho de Rangel e Figueiredo 2008. Embora essas instâncias não sejam pertinentes ao quesito geração de padrões restritos porque a demanda dos itens é muito alta, elas ainda são relevantes para avaliar o comportamento dos algoritmos PC^{PD}_{CPLEX} e PC^{PD}_{H} dentro de um cenário real. Na Tabela 5.7 exibi-se os dados das 13 instâncias. Nas colunas 1-4 apresentam-se o nome das instâncias, o número de itens distintos, comprimento e largura do objeto, respectivamente. Na coluna 5 é dado as dimensões dos itens e a demanda.

Tabela 5.7: Instâncias para o problema de corte de estoque bidimensional.

Nome	m	L	W	(l_i, w_i, b_i)
RF1	3	2750	1830	(445, 213, 1200), (410, 383, 600), (388, 377, 900)
RF2	2	2750	1830	(390, 110, 1800), (370, 110, 900)
RF3	2	2750	1850	(600, 440, 600), (450, 132, 900)
RF4	3	2750	1830	(710, 535, 320), (1062, 530, 320), (647, 453, 960)
RF5	3	2750	1830	(630, 50, 480), (433, 50, 320), (295, 50, 480)
RF6	6	2750	1830	(440, 65, 320), (635, 50, 160), (454, 180, 960) (635, 180, 480), (454, 135, 640), (635, 135, 320)
RF7	7	2750	1850	(970, 570, 320), (700, 75, 160), (700, 212, 480), (700, 163, 320) (600, 400, 0), (430, 60, 0), (500, 60, 0)
RF8	8	2750	1830	(2500, 565, 240), (647, 453, 160), (710, 454, 80), (454, 454, 80) (1080, 454, 200), (530, 454, 200), (1050, 500, 40), (483, 215, 80)
RF9	4	2750	1830	(510, 450, 40), (630, 50, 0), (433, 50, 0), (295, 50, 0)
RF10	3	2750	1830	(454, 180, 320), (635, 180, 160), (454, 135, 0)
RF11	9	2750	1850	(1049, 452, 200), (499, 452, 200), (452, 429, 80), (1050, 535, 80), (535, 500, 80), (535, 430, 80), (700, 212, 160), (430, 60, 0), (500, 60, 0)
RF12	9	2750	1830	(2500, 60, 480), (445, 60, 480), (445, 40, 1040), (490, 60, 40), (500, 60, 120), (1050, 60, 200), (430, 60, 120), (440, 60, 160), (1060, 60, 80)
RF13	3	2750	1830	(430, 60, 800), (500, 60, 160), (1050, 60, 160)

5.2 Resultados para o Problema da Mochila Limitada

Na Tabela 5.8 apresenta-se os resultados das 10 instâncias selecionadas para a parte I do estudo que consiste em avaliar os métodos de solução PML^{PD} e PML_{CPLEX}.

Na primeira coluna é exibido o nome das instâncias. Nas colunas 2 e 4 é exibido o valor ótimo e tempo computacional em segundos obtido pelo PML_{CPLEX} em cada instância. Nas colunas 3 e 5 é exibido o valor ótimo e tempo computacional em segundos do PML^{PD} em cada instância.

Veja que o valor obtido entre os dois é o mesmo em todas as instâncias. Então, valida-se o algoritmo PML^{PD} assim como sua codificação. Além disso, o tempo computacional do algoritmo de programação dinâmica é menor do que o utilizado pelo CPLEX, mostrando que sua utilização é mais vantajosa do que a simples implementação do modelo (2.4)-(2.7) e a utilização do solver para resolver as instâncias.

Tabela 5.8: Comparação entre PML_{CPLEX} e PML^{PD}.

	Valor		Tempo (s)	
Instância	PML_{CPLEX}	PML^{PD}	PML_{CPLEX}	PML^{PD}
lOF1	2.356	2.356	6,753	2,438
wOF1	1.366	1.366	6,682	2,453
lcgcut1	118	118	6,492	2,387
wcgcut1	156	156	6,616	2,393
lwang20	2.277	2.277	6,832	2,458
wwang20	1.486	1.486	6,751	2,481
lgcut1	42.348	42.348	6,519	2,436
wgcut1	31.404	31.404	6,603	2,513
lRF1	942.180	942.180	7,044	2,524
wRF1	777.231	777.231	6,690	2,402

Na Tabela 5.9 são apresentados o número máximo de estados do algoritmo PML^{PD} em cada uma das 62 instâncias. Nessa tabela, percebe-se que as instâncias não são muito desafiadoras para o algoritmo PML^{PD}, já que o número máximo de estados atingido em todo o conjunto de instâncias foi de 243 (instância wRF6). Isso mostra que para abordagens que consideram os problemas de corte e de corte de estoque como casos particulares do problema da mochila, não é necessário algoritmos elaborados de programação dinâmica para resolvê-los. Lembrando que no PML^{PD} é usada uma relação de dominância simples para eliminar os ternos (ver Seção 2.2).

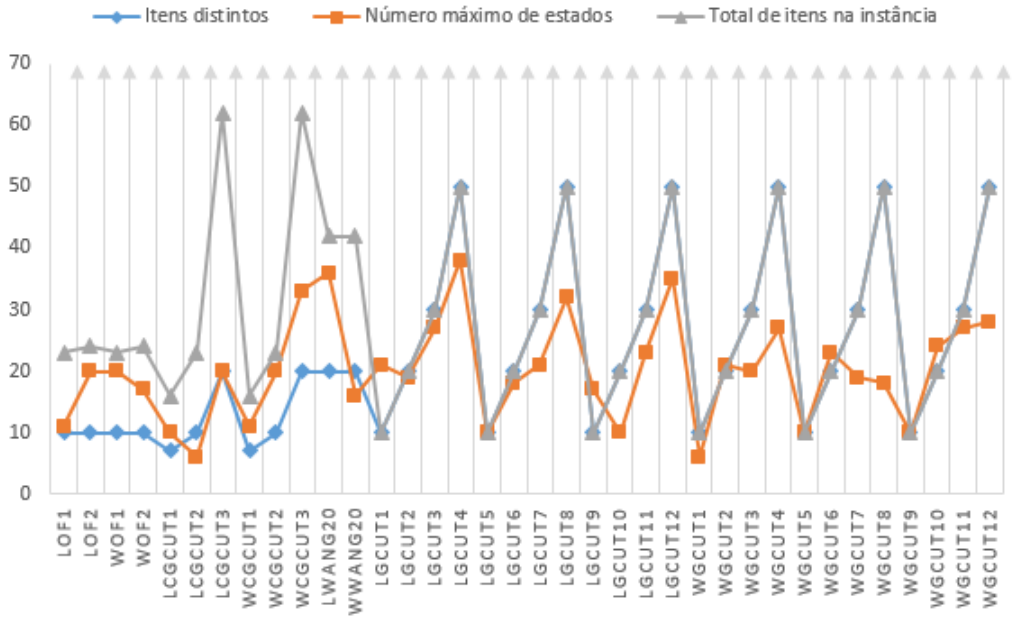
Dado esse panorama geral, na Figura 5.1 é apresentado o gráfico comparando o número de itens distintos, o número total de itens e o número máximo de estados do PML^{PD}. Perceba pelo gráfico que a linha associada ao total de itens e o número máximo de estados apresentam comportamento quase semelhante. Quando uma linha desce, quase sempre a outra faz o mesmo movimento. Isso significa que o total de itens tem influência no número máximo de estados do algoritmo. Outro ponto

Tabela 5.9: Número de Estados do Algoritmo PML^{PD}.

Nome / Número	1	2	3	4	5	6	7	8	9	10	11	12	13
lOF	11	20	-	-	-	-	-	-	-	-	-	-	-
wOF	20	17	-	-	-	-	-	-	-	-	-	-	-
lcgcut	10	6	20	-	-	-	-	-	-	-	-	-	-
wcgcut	11	20	33	-	-	-	-	-	-	-	-	-	-
lwang20	36	-	-	-	-	-	-	-	-	-	-	-	-
wwang20	16	-	-	-	-	-	-	-	-	-	-	-	-
lgcut	21	19	27	38	10	18	21	32	17	10	23	35	-
wgcut	6	21	20	27	10	23	19	18	10	24	27	28	-
lRF	27	35	9	17	80	20	12	81	16	19	31	222	37
wRF	40	13	17	9	23	243	113	19	37	33	46	84	17

que pode ser observado, é que a linha do total de itens se mantém mais elevada do que a dos estados em quase todas as instâncias. Essa observação evidencia que as instâncias do problema de corte não são desafiadoras, porque apresentam muitos ternos dominados, facilitando sua resolução. Uma hipótese que também pode ser levantada e que enfraquece essa última evidência é que nesse conjunto de instâncias, o peso dos itens variam, em geral, de 25% a 75% da capacidade da mochila. Dessa forma, para cada estágio, poucos estados são gerados, fazendo com que o número máximo de estados seja baixo.

Figura 5.1: Comparação entre o número de itens distintos, total de itens e número máximo de estados nas instâncias do problema de corte.

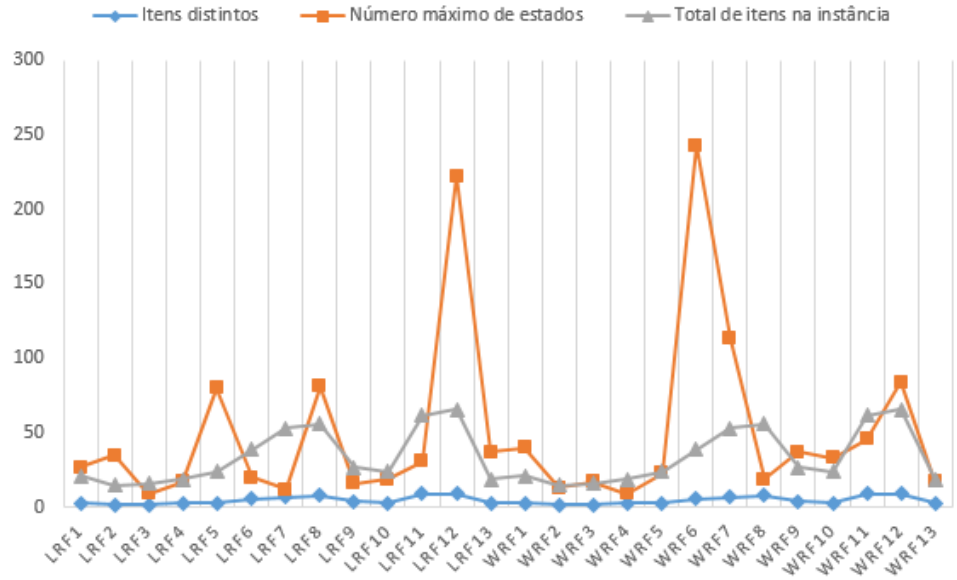


Fonte: autor.

Na Figura 5.2 apresenta-se o gráfico de linha das 26 instâncias RF, adaptadas do problema do corte de estoque. Nesse conjunto de instâncias, o peso dos itens variam, em maioria, de 15% a 30% da capacidade da mochila, além disso o número total de itens se assemelha ao número de itens do conjunto anterior. Então, é possível chegar numa conclusão da hipótese levantada. Como as linhas cinza e laranja ficam alternando, ora uma está acima, ora outra, chega-se a duas conclusões. Quando a linha laranja está acima, significa que em cada estágio foram gerados muitos estados e pouco deles conseguiram ser eliminados, evidenciando uma instância desafiadora. Quando a linha cinza está acima, significa que em cada estágio foram gerados muitos estados, mas a maioria foi eliminada pela relação de dominância, evidenciando uma instância pouco desafiadora. Observando a Figura 5.2, como tiveram apenas dois picos da linha laranja que destoam de todo o conjunto de instâncias, a conclusão mais forte é que os cenários do problema do corte e do corte de estoque são cenários pouco desafiadores para o algoritmo PML^{PD}. Isso confirma o panorama geral exibido na Tabela 5.9.

Já está sendo desenvolvido um estudo com instâncias para o PML geradas aleatoriamente, seu objetivo é comparar o algoritmo PML^{PD} com outros algoritmos de programação dinâmica na literatura (Assis e Rangel 2019). Em resultados preliminares é possível dizer que os cenários do problema de corte e do problema de corte de estoque não são desafiadores porque eles possuem um número total de itens muito baixo (100 itens) em comparação com as instâncias aleatórias para o problema da mochila (1.000 itens para mais). Dessa forma, o número de estados da programação dinâmica não é elevado nesses cenários.

Figura 5.2: Comparação entre o número de itens distintos, total de itens e número máximo de estados nas instâncias do problema do corte de estoque.



Fonte: autor.

5.3 Resultados para o Problema de Corte Bidimensional Guilhotinado 2-estágios Restrito

Na Tabela 5.10 apresentam-se os resultados obtidos pelo PC^{PD}_{CPLEX} e PC^{PD}_{H} usado para resolver as 18 instâncias descritas na Seção 5.1.2. Nas colunas 1 e 2 é exibido o nome das instâncias e sua solução ótima (OPT), nas colunas 3-5 e 6-8 são apresentados a solução (sol), GAP e tempo computacional em segundos (t(s)) para cada instância e método respectivamente. O GAP foi calculado em relação à solução ótima da literatura do seguinte modo $GAP = 100 * (OPT - sol)/OPT$.

Pela Tabela 5.10, o algoritmo PC^{PD}_{CPLEX} obteve a solução ótima em 15 das 18 instâncias. O único motivo para ele não chegar na solução ótima está relacionado com as faixas geradas nos primeiros m problemas da mochila. Isso não quer dizer que o algoritmo PML^{PD} está errado. Uma primeira explicação é que o padrão ótimo possa ser um w-padrão como citado em Mrad, Meftahi e Haouari 2013. Dessa forma, como o método de Gilmore e Gomory 1965 não constrói faixas verticais, ao se resolver o problema $m+1$ não é possível obter a solução ótima para o problema. A segunda possibilidade está relacionada com o fato das faixas geradas não serem exatamente as ótimas para o problema de corte. Isso porque na combinação dos itens nas faixas

(m problemas da mochila), não é levado em conta o problema $m+1$. Com isso, dadas as faixas ótimas para os problemas da mochila, há uma dificuldade em alocar as faixas no padrão, devido o excesso de itens. Esse caso é o mesmo presente no exemplo da Seção 3.7.

Tabela 5.10: Resultados com instâncias da literatura para o Problema de Corte Bidimensional 2-estágios Restrito.

	OPT	PC^{PD}_{CPLEX}			PC^{PD}_{H}		
Instância	sol	sol	GAP	t(s)	sol	GAP	t(s)
OF1	2.713	2.713	0	5,387	2.713	0	0,184
OF2	2.515	2.515	0	5,395	2.515	0	0,187
cgcut1	240	240	0	5,3681	240	0	0,183
cgcut2	2.535	2.467	2,68%	5,473	2.208	12,90%	0,185
cgcut3	1.720	1.660	3,48%	5,366	1.620	5,81%	0,187
wang20	2.623	2.623	0	5,403	2.623	0	0,186
gcut1	43.024	43.024	0	5,369	43.024	0	0,182
gcut2	57.996	57.996	0	5,454	57.996	0	0,185
gcut3	59.895	59.895	0	5,510	59.895	0	0,186
gcut4	60.504	60.504	0	5,657	60.504	0	0,197
gcut5	193.379	193.379	0	5,556	170.411	11,88%	0,183
gcut6	224.399	224.399	0	5,555	224.399	0	0,187
gcut7	238.974	238.974	0	5,472	238.974	0	0,188
gcut8	245.758	245.758	0	5,525	245.758	0	0,199
gcut9	919.476	878.821	4,42%	5,509	696.847	24,21%	0,183
gcut10	856.445	856.445	0	5,416	664.464	22,42%	0,187
gcut11	942.219	942.219	0	5,514	687.428	27,04%	0,187
gcut12	970.744	970.744	0	5,429	970.744	0	0,196

A solução ótima foi obtida em 12 das 18 instâncias pelo algoritmo PC^{PD}_{H}. As possibilidades para o algoritmo não ter chego na solução ótima são três. A primeira e segunda são as mesmas do algoritmo PD^{PC}_{CPLEX}. A terceira possibilidade está relacionada com o fato da heurística H_{PD} não limitar totalmente o número de itens no padrão ao fazer a combinação das faixas. Essa possibilidade é dada dentro do seguinte cenário. CN: *Suponha que as faixas ótimas para o problema de corte foram geradas em algum dos m problemas da mochila. Além disso, existe pelo menos uma faixa (ótima ou não) muito atrativa.* Dessa forma, ao fazer a combinação das faixas no padrão, essa faixa atrativa será colocada no padrão com uma frequência maior do que a frequência ótima. Com isso, os itens presentes nela irão ultrapassar seu limite, necessitando do processo de factibilização para retira-los do padrão. As instâncias que o algoritmo PC^{PD}_{H} não resolveu de forma ótima, esse cenário CN aconteceu nas instâncias gcut5, gcut10 e gcut11. Note que o algoritmo PC^{PD}_{CPLEX} chegou na solução

ótima, ou seja, as faixas ótimas para o problema de corte estavam disponíveis.

Essa última possibilidade foi descrita dentro de um cenário que as faixas ótimas estão disponíveis. No entanto, dentro do cenário das duas primeiras possibilidades, esse processo pode gerar resultados melhores do que a resolução do problema $m+1$ pelo CPLEX. Isso acontece no exemplo da Seção 3.7. Além do algoritmo PC_H^{PD} chegar em muitas soluções ótimas, outra vantagem dele é o seu tempo de processamento que não ultrapassou 0,2s em todas as instâncias (5s abaixo do CPLEX). Outra vantagem, e possivelmente a melhor delas, é não precisar do uso de nenhum *solver* comercial. O maior problema do algoritmo PC_H^{PD} é quando ele se encontra dentro do cenário CN e quanto mais atrativa a faixa for, maior a chance de obter soluções piores. Perceba que nas instâncias que há a certeza disso ter ocorrido, os GAP's são bem grandes, chegando até 27,04%. Uma possível estratégia que possa ajudar o cenário CN não acontecer é fazer o uso da relaxação do espaço de estado utilizado por Christofides e Hadjiconstantinou 1995. Dessa forma, possivelmente a frequência das faixas utilizadas no padrão de corte tenha um controle melhor. Além disso, outra forma de melhorar os algoritmos PC_{CPLEX}^{PD} e PC_H^{PD} é modificar o algoritmo PML^{PD} para que seja considerada a rotação dos itens, possibilitando a geração de novas faixas a serem utilizadas na segunda etapa do método de Gilmore e Gomory 1965.

5.4 Resultados para o Problema de Corte de Estoque Bidimensional Exato

Na Tabela 5.11 apresenta-se os resultados obtidos para o PCE considerando as 13 instâncias descritas na Seção 5.1.3. São apresentados os resultados para a Heurística TC e pelo sistema CorteBiFur retirados de Rangel e Figueiredo 2008 e para o método de geração de colunas usando o algoritmo PC_{CPLEX}^{PD} e o algoritmo PC_H^{PD} para resolver o subproblema *pricing*. Na primeira coluna é dado o nome das instâncias. Nas colunas seguintes, para cada método de resolução, mostra-se o número de padrões (NP), número de objetos utilizados (NO) e sobras máxima (S_{mx}), média (S_{md}) e mínima (S_{mn}) dos padrões de corte. Nas colunas 2-6 é exibido os resultados obtidos pela Heurística TC, 7-11 pelo CorteBiFur, 12-16 pelo algoritmo PC_{CPLEX}^{PD} e 17-21 pelo algoritmo PC_H^{PD}. Os resultados dos métodos são calculados fazendo o arredondamento da solução para cima gerando sobras de itens, com exceção da Heurística TC que resolve o problema inteiro pelo método *branch-and-cut* (e. g. Arenales et al. 2015).

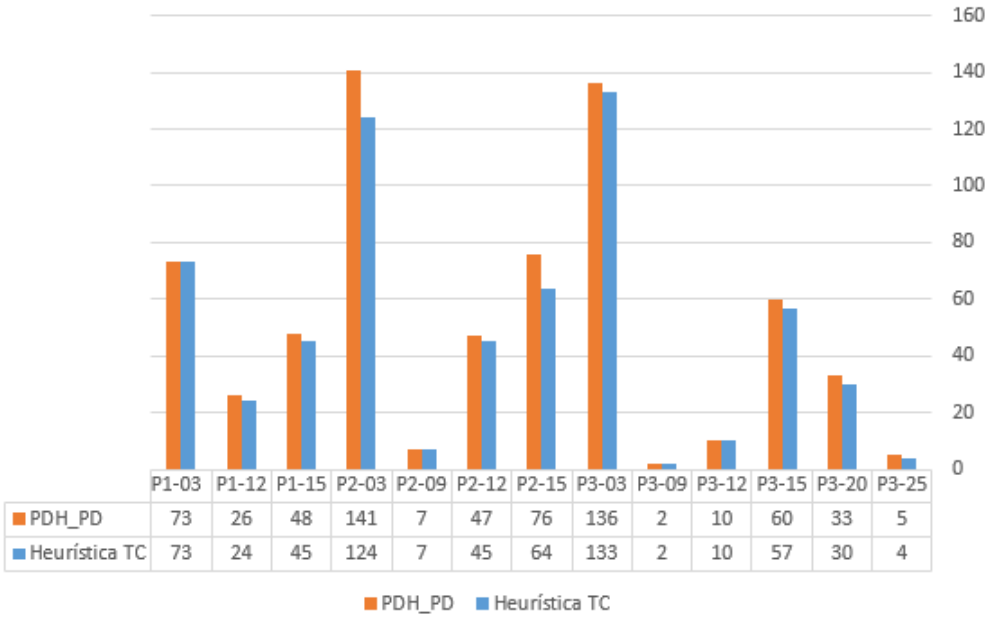
Pela tabela, a primeira análise feita é em relação aos métodos PC^{PD}_{CPLEX} e PC^{PD}_{H}. Em 8 instâncias das 13 os dois algoritmos tiveram as mesmas soluções. Na instância RF6 o algoritmo PC^{PD}_{H} gerou uma coluna a menos e teve sobra média inferior, ou seja, conseguiu gerar padrões melhores do que o PC^{PD}_{CPLEX}. Na instância RF8, o algoritmo PC^{PD}_{H} gerou uma coluna a mais, no entanto, cortou um objeto a menos. Essa coluna gerada a mais possui sobra próxima a média. Nas instâncias RF11 e RF12, foram gerados padrões com sobras percentual maior, no entanto essa diferença não é muito grande. Na instância RF13 foram gerados padrões melhores do que o algoritmo PC^{PD}_{CPLEX}. Como tiveram instâncias que ora um algoritmo foi melhor, ora o outro, isso confirma as possibilidades ditas nos testes para o problema de corte. Num panorama geral, os dois algoritmos não apresentaram grandes diferenças. Era esperado que o tempo computacional do algoritmo PC^{PD}_{H} fosse menor do que o algoritmo PC^{PD}_{CPLEX} conforme os resultados da Seção 5.3. No entanto, ambos tiveram o tempo computacional em torno de 10s, não apresentando diferença significativa entre eles. Por esse fato, os resultados referentes ao tempo computacional de cada algoritmo foi omitido. As próximas análises serão feitas em relação ao algoritmo PC^{PD}_{H} e os métodos Heurística TC e CorteBiFur.

Tabela 5.11: Resultados com instâncias para o problema de corte de estoque bidimensional.

	Heurística TC					CorteBiFur					PC^{PD}_{CPLEX}					PC^{PD}_{H}				
Nome	NP	NO	S_{mx}	S_{md}	S_{mn}	NP	NO	S_{mx}	S_{md}	S_{mn}	NP	NO	S_{mx}	S_{md}	S_{mn}	NP	NO	S_{mx}	S_{md}	S_{mn}
RF1	3	73	10,9	5,1	2,8	3	72	4,8	1,5	1,3	7	73	12,0	5,6	0,2	7	73	12,0	5,6	0,2
RF2	3	24	1,7	1,0	0,8	2	25	0,8	0,7	0,7	2	26	5,1	2,6	0,0	2	26	5,1	2,6	0,0
RF3	2	45	5,1	3,4	2,8	2	45	5,1	2,8	2,4	3	48	15,7	7,8	0,0	3	48	15,7	7,8	0,0
RF4	2	124	5,6	5,5	5,4	2	124	5,6	5,5	5,4	4	141	32,1	22,2	5,4	4	141	32,1	22,2	5,4
RF5	3	7	1,4	1,2	0,0	3	7	2,4	1,2	0,0	5	7	2,0	0,4	0,0	5	7	2,0	0,4	0,0
RF6	8	45	1,4	0,3	0,0	6	48	2,2	0,3	0,0	13	47	6,5	1,5	0,0	12	47	6,5	1,3	0,0
RF7	3	64	4,7	3,9	3,4	4	63	5,4	4,4	4,1	10	76	28,3	21,8	19,3	10	76	28,3	21,8	19,3
RF8	9	133	8,7	5,7	0,0	7	135	8,7	5,9	0,0	12	137	15,2	5,6	0,0	13	136	15,2	5,6	0,0
RF9	1	2	1,6	1,6	1,6	1	2	1,4	1,4	1,4	1	2	7,3	7,3	7,3	1	2	7,3	7,3	7,3
RF10	2	10	1,8	1,8	1,2	2	11	4,4	4,2	3,6	2	10	6,5	3,3	0,0	2	10	6,5	3,3	0,0
RF11	9	57	5,0	3,6	1,2	7	60	8,1	5,0	3,2	17	60	22,9	9,2	2,5	17	60	28,3	9,5	2,5
RF12	10	30	6,0	2,7	0,1	9	33	2,0	1,5	0,0	20	33	4,8	1,2	0,0	20	33	8,6	1,6	0,0
RF13	2	4	2,0	1,6	0,6	3	5	2,4	2,1	1,9	4	5	56,9	16,0	0,3	4	5	55,8	15,8	0,3

Na Figura 5.3 apresenta-se o gráfico de barras do número de objetos utilizados por cada uma das 13 instâncias. Pelo gráfico, as barras em laranja representam a quantidade de objetos cortados na solução obtida pelo algoritmo PC_H^{PD} e em azul pela Heurística TC. Então, em 8 das 13 instâncias a solução obtida pelo algoritmo PC_H^{PD} cortou mais objetos do que a solução da Heurística TC, chegando a cortar até 17 objetos a mais na instância RF4. Não houve nenhuma instância em que o algoritmo cortou menos objetos.

Figura 5.3: Comparação do número de objetos cortados entre o algoritmo PC_H^{PD} e Heurística TC.



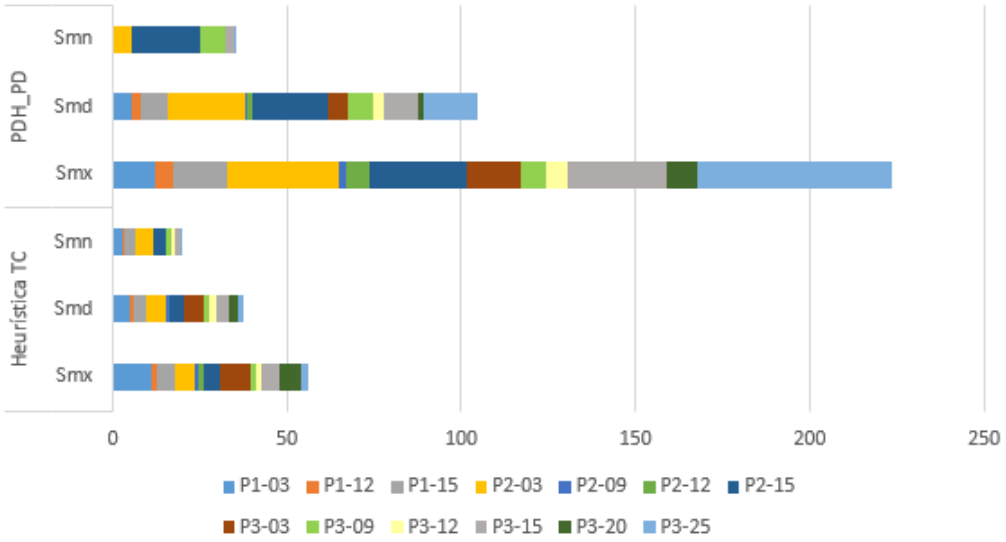
	P1-03	P1-12	P1-15	P2-03	P2-09	P2-12	P2-15	P3-03	P3-09	P3-12	P3-15	P3-20	P3-25
PDH_PD	73	26	48	141	7	47	76	136	2	10	60	33	5
Heurística TC	73	24	45	124	7	45	64	133	2	10	57	30	4

Fonte: autor.

Outra análise que pode ser feita é da soma total das sobras de material. Na Figura 5.4 ilustra-se o comparativo do acúmulo de perda por sobra de material. Observando a Tabela 5.11, é possível ver que a solução obtida pelo algoritmo PC_H^{PD} apresenta mais instâncias com aproveitamento máximo do objeto. No entanto, pela Figura 5.3 é possível ver que o acúmulo das sobras mínimas das soluções do algoritmo PC_H^{PD} é maior do que as soluções da Heurística TC. Além disso, as sobras mínimas das soluções do algoritmo PC_H^{PD} se assemelham às médias das soluções da Heurística TC. Uma outra forma de enxergar o gráfico, é olhando o tamanho da barra de cada instância. Na Heurística TC, suas barras não variam muito de tamanho. Enquanto que no algoritmo PC_H^{PD} elas são bem heterogêneas. Isso mostra que em algumas instâncias o algoritmo PC_H^{PD} possui um desempenho ruim. Isso é causado porque o

algoritmo PML^{PD} não faz a rotação dos itens, então há um leque menor de padrões de corte possíveis a serem gerados.

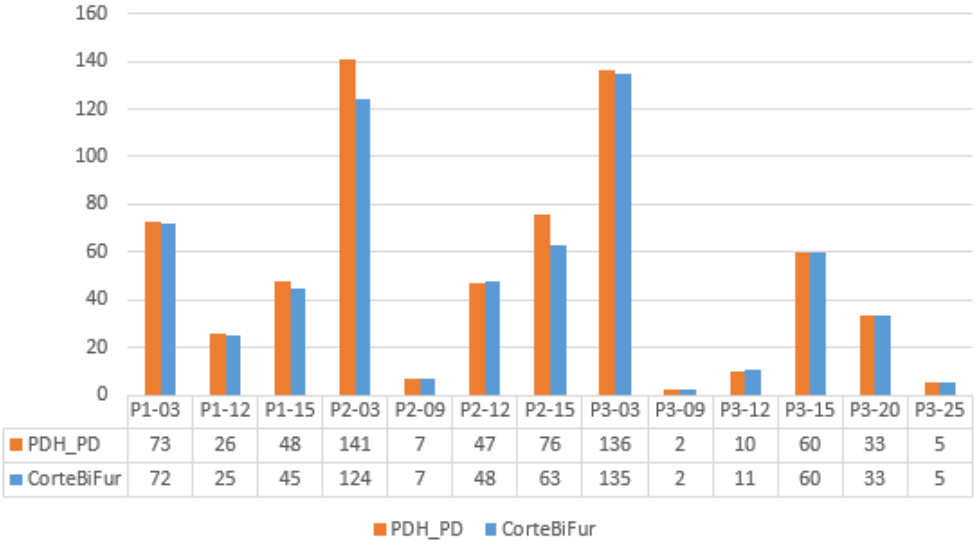
Figura 5.4: Comparação do acúmulo de perda por sobra de material entre o algoritmo PC_H^{PD} e Heurística TC.



Fonte: autor.

Na Figura 5.5, as barras em laranja representam o número de objetos cortados nas soluções obtidas pelo algoritmo PC_H^{PD} e as barras em azul o número de objetos cortados nas soluções do sistema CorteBiFur. Para 6 instâcias das 13, foram cortados mais objetos na solução obtida com o algoritmo PC_H^{PD} do que a solução obtida com o sistema CorteBiFur, chegando a ser até 17 objetos a mais na instância RF4. Também, o algoritmo PC_H^{PD} conseguiu solução com uma unidade a menos do que o sistema CorteBiFur em duas instâncias RF6 e RF10. Dentro dessa análise, houve uma certa igualdade no uso de objetos pelos dois métodos, somente sendo discrepante nas instâncias RF4 e RF7.

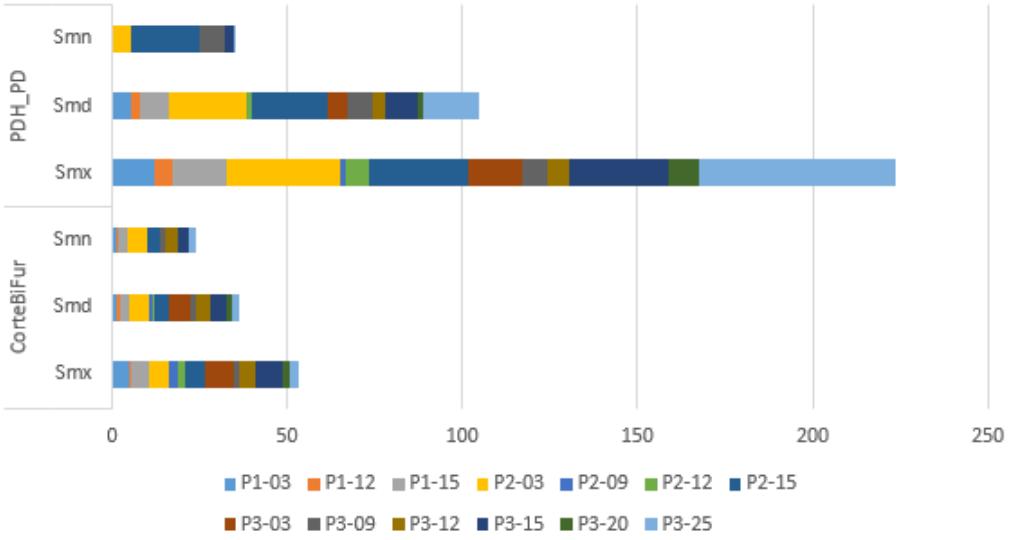
Figura 5.5: Comparação do número de objetos cortados entre o algoritmo PC_H^{PD} e a Heurística TC.



Fonte: autor.

Observando a Figura 5.6, a análise feita da Figura 5.4 entre o algoritmo PC_H^{PD} e a Heurística TC pode ser estendida para a comparação entre o algoritmo PC_H^{PD} e o sistema CorteBiFur.

Figura 5.6: Comparação do acúmulo de perda por sobra de material entre o algoritmo PC_H^{PD} e o sistema CorteBiFur.



Fonte: autor.

Com esse estudo pode ser dito que o algoritmo PC_H^{PD} e o algoritmo PC_{CPLEX}^{PD} tiveram praticamente o mesmo desempenho, não mostrando diferenças significativas

entre os dois. Então o uso do algoritmo PC_H^{PD} mostra-se mais vantajoso, pois qualquer usuário poderia utilizá-lo não sendo necessário qualquer tipo de licença de *solver*. Claro que nessas instâncias foram utilizados o CPLEX por ambos os algoritmos para resolver o problema mestre restrito, mas caso o usuário fizesse uma implementação caseira do método simplex, ele estaria livre de qualquer dependência de *solver*.

Em relação à Heurística TC, o algoritmo PC_H^{PD} teve um pior desempenho tanto no número de objetos cortados quanto no acúmulo de perda por sobra de material. Em relação ao sistema CorteBiFur, o algoritmo PC_H^{PD} mostrou uma certa igualdade quando comparado ao número de objetos cortados em suas soluções. Esse equilíbrio era esperado, já que o algoritmo PML^{PD} foi baseado no algoritmo presente no sistema CorteBiFur. Mas pela discrepância em duas instâncias, evidencia-se que o algoritmo PC_H^{PD} precisa ser melhorado. Um ponto interessante é em cenários no qual o padrão de corte gerado precisa ser restrito, neles o algoritmo PC_H^{PD} pode ser bem vantajoso.

É importante enfatizar que a Heurística TC e o sistema CorteBiFur consideram a rotação dos itens, possibilitando a geração de padrões de corte mais proveitosos. No algoritmo PML^{PD} ainda não é feita a rotação dos itens, então isso é possivelmente um causador dos algoritmos PC_{CPLEX}^{PD} e PC_H^{PD} terem um pior desempenho no processo de geração de colunas.

CONSIDERAÇÕES FINAIS

Nessa dissertação foram abordados três problemas de corte e empacotamento, o problema da mochila limitada, o problema de corte bidimensional guilhotinado 2-estágios restrito e o problema de corte de estoque. No Capítulo 1 fez-se um estudo das características gerais dos problemas de corte e empacotamento. Considerando-se as tipologias propostas por Dyckhoff 1990 e Wäscher, Haußner e Schumann 2007 percebeu-se que a classificação dos problemas de corte e empacotamento são difíceis devido ao grande número de variações possíveis. No entanto, essa classificação é de grande importância para a busca e divulgação de artigos científicos.

No Capítulo 2 foi estudado o problema da mochila. Inicialmente se viu variações do problema e como a programação dinâmica é utilizada para resolvê-lo. Ao tratar do problema da mochila limitada, como contribuição, foi apresentado um algoritmo de programação dinâmica PML^{PD} para resolver o problema e que teve o seu desenvolvimento baseado no algoritmo PR proposto por Perin e Rangel 1989 que resolve PMI.

No Capítulo 3, foi visto o problema de corte bidimensional. Foram definidas restrições do equipamento de corte que são necessárias para caracterizar um padrão de corte. Também apresentou-se como a programação dinâmica pode ser utilizada para resolver o problema e também gerar limites superiores para outros métodos de resolução. Como contribuição, desenvolveu-se a heurística H_{PD} para resolver a segunda etapa da geração de padrões de corte 2-estágios restrito, uma adaptação do método de Gilmore e Gomory 1965. Além disso, houve a combinação do algoritmo PML^{PD} com a heurística H_{PD} (PC_H^{PD}) para resolver o problema de corte bidimensional guilhotinado 2-estágios restrito. Como forma de comparar a heurística desenvolvida, usou-se o algoritmo PML^{PD} junto com o *solver* CPLEX (PC_{CPLEX}^{PD}) para resolver esse problema.

No Capítulo 4 apresentou-se uma breve revisão da literatura sobre o problema de corte de estoque. Em seguida foi explicado o processo de geração de colunas e utilizou-se os algoritmos PC_H^{PD} e PC_{CPLEX}^{PD} para resolver o subproblema *pricing* associado. Com isso, o problema de corte de estoque é resolvido gerando padrões de corte

guilhotinados 2-estágios restritos.

No Capítulo 5 apresentam-se resultados do estudo computacional associado aos três problemas tratados na pesquisa. Na primeira parte do estudo, referente ao problema da mochila limitada, concluiu-se que os cenários do problema de corte e corte de estoque presentes na literatura são pouco desafiadores para o algoritmo PML^{PD}. Na segunda parte, tratando do problema de corte, concluiu-se que o algoritmo PC_H^{PD} apresenta bons resultados num tempo computacional bem pequeno. No entanto, quando ele se depara com o cenário CN: *"Suponha que as faixas ótimas para o problema de corte foram geradas em algum dos m problemas da mochila. Além disso, existe pelo menos uma faixa (ótima ou não) muito atrativa."*, seu desempenho pode ser bem prejudicado. Na terceira parte, referente ao problema de corte de estoque, percebeu-se que o uso do algoritmo PC_H^{PD} na geração de colunas, quase não apresentou diferença quando comparado ao algoritmo PC_{CPLEX}^{PD}. Além disso, ao comparar o algoritmo PC_H^{PD} com o sistema CorteBiFur, somente em duas instâncias o algoritmo teve um desempenho melhor do que o sistema, mostrando uma igualdade entre os métodos na maioria dos casos. O único problema visto é a inconstância do algoritmo, gerando de padrões muito bons a muito ruins, podendo ser causado pelo cenário CN e/ou pelo algoritmo PC_H^{PD} não fazer a rotação dos itens.

Após ter uma visão geral dos testes computacionais, algumas considerações podem ser feitas. Ao utilizar o método de Gilmore e Gomory 1965 para gerar padrões de corte 2-estágios, não há necessidade do uso de algoritmos de programação dinâmica muito sofisticados para resolver os $m+1$ problemas da mochila, pois as instâncias são pouco desafiadoras. Dessa forma, olhando para os problemas de corte, é perceptível que o algoritmo PML^{PD} não tem influência sobre o padrão de corte gerado. O que realmente determinante se um padrão de corte é o ótimo para o problema é a abordagem utilizada para gerar esse padrão. Como a abordagem de Gilmore e Gomory 1965 não prevê w-padrões, em alguns cenários a solução ótima para o problema não aparece em todo o espaço de busca dos algoritmos PC_H^{PD} e PC_{CPLEX}^{PD}. Então, como proposta futura propõe-se adaptar a abordagem de Gilmore e Gomory 1965 de forma que w-padrões também possam ser obtidos durante o método de resolução. Além disso, o algoritmo PML^{PD} não permite a rotação dos itens, essa seria outra possibilidade para obter soluções melhores nos algoritmos PC_H^{PD} e PC_{CPLEX}^{PD}. Voltado ao caso restrito, talvez a utilização da relaxação do espaço de estado proposta por Christofides e Hadjiconstantinou 1995 combinada ao limite superior usado na heurística H_{PD}, possa gerar limitantes melhores para os itens.

As propostas feitas para o PML e para o PC podem ser uma forma de se obter padrões de corte que possuem uma perda por sobra de material menor durante o processo de geração de colunas.

Como proposta futura, utilizar os algoritmo PC_H^{PD} e PC_{CPLEX}^{PD} para fazer o empacotamento bidimensionais em faixas em níveis. Esse também é um problema da classe dos problemas de corte e empacotamento e que é classificado segundo a tipologia de Wäscher, Haußner e Schumann 2007 como *2-dimensional rectangular Open Dimension Problem (em níveis)*. Esse problema possui aplicação em gondolas de supermercado (e.g. Bezerra 2017). No método de duas estapas de Gilmore e Gomory 1965, a construção das faixas representam os níveis das prateleiras das gondolas e o padrão de corte representa todos os itens que serão colocados nas prateleiras das gondolas. Dessa forma, é possível que os algoritmos PC_H^{PD} e PC_{CPLEX}^{PD} resolvam o problema.

REFERÊNCIAS

ANDONOV, R.; POIRRIEZ, V.; RAJOPADHYE, S. Unbounded knapsack problem: Dynamic programming revisited. *European Journal of Operational Research*, Elsevier, v. 123, n. 2, p. 394–407, 2000.

ANDRADE, R.; BIRGIN, E. G.; MORABITO, R. Two-stage two-dimensional guillotine cutting stock problems with usable leftover. *International Transactions in Operational Research*, Wiley Online Library, v. 23, n. 1-2, p. 121–145, 2016.

ARENALES, M. et al. *Pesquisa operacional: para cursos de engenharia*. [S.l.]: Elsevier Brasil, 2015.

ASSIS, N.; RANGEL, S. Um estudo dos algoritmos de programação dinâmica para o problema da mochila limitada. In: *(Em processo)*. São José do Rio Preto - SP: [s.n.], 2019.

BEASLEY, J. Algorithms for unconstrained two-dimensional guillotine cutting. *Journal of the Operational Research Society*, Taylor & Francis, v. 36, n. 4, p. 297–306, 1985.

BEZERRA, V. M. *Problemas de empacotamento bidimensional em níveis : estratégias baseadas em modelagem matemática*. Tese (Doutorado) — Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos, 2017.

CARVALHO, J. Valério de. Exact solution of bin-packing problems using column generation and branch-and-bound. *Annals of Operations Research*, Springer, v. 86, p. 629–659, 1999.

CHAVES, C.; RANGEL, S.; POLDI, K. C. Um estudo sobre a influência da ordenação dos padrões de corte em heurísticas residuais - caso bidimensional. In: 2019, A. eletrônicos... C. G. (Ed.). *Simpósio Brasileiro de Pesquisa Operacional*. Limeira: url: https://proceedings.science/sbpo-2019/papers/um-estudo-sobre-a-influencia-da-ordenacao-dos-padroes-de-corte-em–heuristicas-residuais—caso-bidimensional?lang=pt-br, 2019.

CHERRI, A. C.; ARENALES, M. N. O problema de corte de estoque com reaproveitamento das sobras de material–heurística ffd modificada. *Anais do XXXVII Simpósio Brasileiro de Pesquisa Operacional, Gramado*, p. 1700–1711, 2005.

CHRISTOFIDES, N.; HADJICONSTANTINOU, E. An exact algorithm for orthogonal 2-d cutting problems using guillotine cuts. *European Journal of Operational Research*, Elsevier, v. 83, n. 1, p. 21–38, 1995.

CHRISTOFIDES, N.; WHITLOCK, C. An algorithm for two-dimensional cutting problems. *Operations Research*, INFORMS, v. 25, n. 1, p. 30–44, 1977.

CINTRA, G. et al. Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation. *European Journal of Operational Research*, Elsevier, v. 191, n. 1, p. 61–85, 2008.

CORTECERTO. *Corte certo: planos de corte e de lucros. url: https://cortecerto.com/. Acesso em: 11 de novembro de 2019*. [S.l.], 2017.

COSTA, L. L. S. *Um estudo sobre o problema de corte de estoque bidimensional 2-estágios*. 103 p. Dissertação (Mestrado) — IMECC - UNICAMP, Campinas - SP, 2016.

DYCKHOFF, H. A typology of cutting and packing problems. *European Journal of Operational Research*, Elsevier, v. 44, n. 2, p. 145–159, 1990.

GILMORE, P.; GOMORY, R. E. Multistage cutting stock problems of two and more dimensions. *Operations research*, INFORMS, v. 13, n. 1, p. 94–120, 1965.

GILMORE, P.; GOMORY, R. E. The theory and computation of knapsack functions. *Operations Research*, INFORMS, v. 14, n. 6, p. 1045–1074, 1966.

GILMORE, P. C.; GOMORY, R. E. A linear programming approach to the cutting-stock problem. *Operations research*, INFORMS, v. 9, n. 6, p. 849–859, 1961.

GILMORE, P. C.; GOMORY, R. E. A linear programming approach to the cutting stock problem—part ii. *Operations research*, INFORMS, v. 11, n. 6, p. 863–888, 1963.

HERZ, J. Recursive computational procedure for two-dimensional stock cutting. *IBM Journal of Research and Development*, IBM, v. 16, n. 5, p. 462–469, 1972.

HOROWITZ, E.; SAHNI, S. Computing partitions with applications to the knapsack problem. *Journal of the ACM (JACM)*, ACM, v. 21, n. 2, p. 277–292, 1974.

ILOG. *CPLEX Reference Manual. url: https://www.ibm.com/us-en/?lnk=m. Acesso em: 4 de novembro de 2019*. 12.1. ed. [S.l.], 2009.

INAREJOS, O.; HOTO, R.; MACULAN, N. An integer linear optimization model to the compartmentalized knapsack problem. *International Transactions in Operational Research*, Wiley Online Library, v. 26, n. 5, p. 1698–1717, 2019.

LEMOS, R. B. *Reengenharia de software do sistema CorteBi com a inclusão de novas funcionalidades*. 2005. Trabalho de Conclusão de Curso (Graduação em Bacharelado em Ciências da Computação) - Universidade Estadual Paulista Júlio de Mesquita Filho.

MACEDO, R.; ALVES, C.; CARVALHO, J. V. D. Arc-flow model for the two-dimensional guillotine cutting stock problem. *Computers & Operations Research*, Elsevier, v. 37, n. 6, p. 991–1001, 2010.

MALAGUTI, E. et al. Integer optimization with penalized fractional values: The knapsack case. *European Journal of Operational Research*, Elsevier, v. 273, n. 3, p. 874–888, 2019.

MARTELLO, S.; TOTH, P. *Knapsack problems: algorithms and computer implementations*. [S.l.]: John Wiley & Sons, Inc., 1990.

MARTIN, M. et al. The constrained two-dimensional guillotine cutting problem with defects: an ilp formulation, a benders decomposition and a cp-based algorithm. *International Journal of Production Research*, Taylor & Francis, p. 1–18, 2019.

MORABITO, R.; ARENALES, M. Optimizing the cutting of stock plates in a furniture company. *International Journal of Production Research*, Taylor & Francis, v. 38, n. 12, p. 2725–2742, 2000.

MRAD, M.; MEFTAHI, I.; HAOUARI, M. A branch-and-price algorithm for the two-stage guillotine cutting stock problem. *Journal of the Operational Research Society*, Taylor & Francis, v. 64, n. 5, p. 629–637, 2013.

OR-BENCHMARK. *url: ftp://cermsem.univ-paris1.fr/pub/CERMSEM/hifi/OR-Benchmark.html*. Acesso em: 4 de novembro de 2019.

OR-LIBRARY. *Copyright (c) (2010) (J E Beasley)*. 1990. Url: http://people.brunel.ac.uk/ mastjjb/jeb/info.html. Acesso em: 4 de novembro de 2019.

PERIN, C.; RANGEL, S. O problema do corte bidimensional. In: *Anais do Congresso Nacional de Matemática Aplicada e Computacional*. São José do Rio Preto - SP: [s.n.], 1989. p. 131–134.

PISINGER, D. A minimal algorithm for the bounded knapsack problem. *INFORMS Journal on Computing*, INFORMS, v. 12, n. 1, p. 75–82, 2000.

QUIROGA-OROZCO, J. J.; CARVALHO, J. Valério de; HOTO, R. S. V. A strong integer linear optimization model to the compartmentalized knapsack problem. *International Transactions in Operational Research*, Wiley Online Library, v. 26, n. 5, p. 1633–1654, 2019.

RANGEL, S.; FIGUEIREDO, A. G. d. O problema de corte de estoque em indústrias de móveis de pequeno e médio portes. *Pesquisa Operacional*, SciELO Brasil, v. 28, n. 3, p. 451–472, 2008.

RANGEL, S. et al. Cortebifur - corte bidimensional retangular em dois estágios. *Manual do Programa de Computador registrado - UNESP - SJRP. Departamento de Matemática Aplicada*, 2019.

RUSSO, M. et al. Constrained two-dimensional guillotine cutting problem: upper-bound review and categorization. *International Transactions in Operational Research*, Wiley Online Library, 2019.

SCHEITHAUER, G. *Introduction to Cutting and Packing Optimization: Problems, Modeling Approaches, Solution Methods*. [S.l.]: Springer, 2017.

VELASCO, A. S. *Algoritmos híbridos para o problema de corte bidimensional*. Tese (Doutorado) — UFF, Niterói, 2017.

WANG, P. Two algorithms for constrained two-dimensional cutting stock problems. *Operations Research*, INFORMS, v. 31, n. 3, p. 573–586, 1983.

WÄSCHER, G.; HAUSSNER, H.; SCHUMANN, H. An improved typology of cutting and packing problems. *European journal of operational research*, Elsevier, v. 183, n. 3, p. 1109–1130, 2007.

YANASSE, H. H.; MORABITO, R. Modelos lineares e não lineares inteiros para problemas da mochila bidimensional restrita a 2 estágios. *Production*, SciELO Brasil, v. 23, n. 4, p. 887–896, 2013.

ZHOU, S. et al. Two-dimensional knapsack-block packing problem. *Applied Mathematical Modelling*, Elsevier, v. 73, p. 1–18, 2019.