Universidade Estadual de Londrina

JOHN JAIRO QUIROGA OROZCO

UM MÉTODO BRANCH AND BOUND PARA O PROBLEMA DA COMPARTIMENTAÇÃO DAS MOCHILAS

Londrina
2018

JOHN JAIRO QUIROGA OROZCO

UM MÉTODO BRANCH AND BOUND PARA O PROBLEMA DA COMPARTIMENTAÇÃO DAS MOCHILAS

Dissertação de mestrado apresentada ao Departamento de Matemática da Universidade Estadualde Londrina, como requisito parcial para a obtenção do Título de MESTRE em Matemática Aplicada e Computacional.

Orientador: Prof. Dr. Robinson Samuel Vieira Hoto.
Co-orientador:Prof. Dr. José Manuel Vasconcelos Valériode Carvalho.

Londrina
2018

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

Orozco, John Jairo Quiroga.
UM MÉTODO BRANCH AND BOUND PARA O PROBLEMA DA COMPARTIMENTAÇÃO DAS MOCHILAS / John Jairo Quiroga Orozco. - Londrina, 2018.
165 f.

Orientador: Robinson Samuel Vieira Hoto.
Coorientador: José Manuel Vasconcelos Valério de Carvalho.
Dissertação (Mestrado em Matemática Aplicada e Computacional) - Universidade Estadual de Londrina, Centro de Ciências Exatas, , 2018.
Inclui bibliografia.

1. Problema da Mochila Compartimentada - Tese. 2. Modelo Linear Forte - Tese. 3. Branch and Bound - Tese. 4. Programação Inteira - Tese. I. Vieira Hoto, Robinson Samuel. II. Valério de Carvalho, José Manuel Vasconcelos. III. Universidade Estadual de Londrina. Centro de Ciências Exatas. . IV. Título.

JOHN JAIRO QUIROGA OROZCO

UM MÉTODO BRANCH AND BOUND PARA O PROBLEMA DA COMPARTIMENTAÇÃO DAS MOCHILAS

Dissertação de mestrado apresentada ao Departamento de Matemática da Universidade Estadualde Londrina, como requisito parcial para a obtenção do Título de MESTRE em Matemática Aplicada e Computacional.

BANCA EXAMINADORA

_____

Orientador: Prof. Dr. Robinson Samuel Vieira Hoto
Universidade Estadual de Londrina - UEL

_____

Co-orientador: Prof. Dr. José M. V. Valério de Carvalho
Universidade do Minho - UMinho

_____

Prof. Dr. Alireza Mohebi Ashtiani
Universidade Tecnológica Federal do Paraná - UTFPR

_____

Profa. Dra. Michele de Oliveira Alves
Universidade Estadual de Londrina - UEL

Londrina, 21 de fevereiro de 2018.

Dedico este trabalho a minha família: meu pai, Jairo Quiroga Aponte (in memoriam), que navega como pairo pelo imenso mar de nossas vidas; minha mãe, Susana Orozco Prada, que sua força e amor infinto faz que lute nobremente por tudos meus sonhos e, meu irmão, Giovanni Agustin, que sua tenacidade, valentia e decisão é o espírito que preenche meu coração frente toda adversidade e faz que nunca baixe minha olhada.

AGRADECIMENTOS

Aos meus pais, Susana Orozco, Jairo Quiroga (*in memoriam*) e meu irmão Giovanni Agustin, por sempre estar juntos, lutar e sair adiante nas mais difíceis situações, por acreditar em meus sonhos, por sua paciência e por fazer da pessoa que hoje sou.

Às famílias Quiroga, Orozco, Onatra, Vargas, Imues, Abril, Morera Riveros, Palenque, Cardenas, Guerrero, Salazar e Mayta pelo carinho, apoio incondicional e as vozes de ânimo para concluir com sucesso este mestrado.

À Betty S. M., por preencher de luz e ser guia em meu caminho, por me levantar de minhas quedas, como também de testemunhar o sacrifício que fiz para alcançar este sonho.

Ao meu orientador Prof. Doc. Robinson Samuel Hoto Vieira, pela guia, confiança em minhas capacidades, contribuição e acompanhamento nas fases do desenvolvimento de este projeto de dissertação. Além de seu apoio enorme nos momentos críticos que afrontei no transcurso de meus estudos.

Ao meu coorientador Prof. Doc. José Manuel Vasconcelos Valério de Carvalho, por me receber em Braga e na Universidade do Minho, pela importante contribuição no projeto de dissertação e seu acompanhamento permanente no meu trabalho.

Ao Osvaldo Inarejos e o Matheus Pimenta, pelo acompanhamento, dedicação, contribuição e discussão dos componentes científicos deste trabalho.

Ao Prof. Doc. e grande amigo Vladimir Angulo Castillo, por criar em mim o sonho de chegar até o Brasil, como também de ter sua disposição e amizade sempre que eu precisei.

Aos meus colegas, amigos e professores do PGMAC, que fizeram importantes aportes em meu desenvolvimento como pesquisador e profissional matemático, como além de ter seu apoio continuo para levar adiante este mestrado, especialmente de: Anderson Salata, Juniormar Organista, Kariston Stevan, Gustavo Naozuka, Arthur Henrique, Elias Borges, Lucas Cavalcante, Weberty Domingos, Camila Hiromi Tamura, Saulo Medrado, Guilherme de Martini, Tadasi Matsubara, Alex Issamu Moriya, Paulo Liboni, José Henrique Rodrigues, Michele Alves, Adriano de Azevedo e Marcio Antonio da Silva.

Aos meus grandes amigos Georgina Palenque, Angélica Colmenares, Alejandra Robles, Helda Johanna Rueda, Amalia Rincon, Alirio Cardenas (*in memoriam*), Angel Labrador, Marjorie Gomez, Amanda Mendes, Oscar Arias, Yuly Gomez, Cristina Martinez e Ingrid Blanco que sempre me acompanharam antes e durante este enorme desafio, pela ajuda desinteressada que sempre me ofereceram, seu apoio, força e sua muito valiosa amizade.

Á Organização dos Estados Americanos (O.E.A.), o Grupo Coimbra de Universidade Brasileiras (G.C.U.B.) e a CAPES pela bolsa concedida para desenvolver meus estudos de Pós-graduação no Brasil.

QUIROGA OROZCO, John Jairo. **UM MÉTODO *BRANCH AND BOUND* PARA O PROBLEMA DA COMPARTIMENTAÇÃO DAS MOCHILAS** . 2018. 165. Dissertação (Mestrado em Matemática Aplicada e Computacional) – Universidade Estadual de Londrina, Londrina, 2018.

RESUMO

O Problema de Mochila Compartimentada (PMC) é um tipo de problema relativamente novo, com uma ampla aplicação de processos industriais, como é o caso da indústria metalúrgica no corte de bobinas de aço em duas fases, onde foi seu surgimento. Atualmente, tem-se duas formas de tratar este tipo de problema seguindo suas formulações matemáticas: pela sua formulação clássica que é referida a um problema de otimização não linear sendo resolvido com heurísticas de decomposição, e em segundo lugar, pela sua formulação linear, usando-se métodos de solução exata.

Este trabalho de dissertação tem como finalidade apresentar o estudo de novos métodos de solução exata ao PMC, aproveitando-se da linearidade do problema. Duas abordagens foram feitas para desenvolver os novos métodos: primeiro, por meio do fortalecimento do modelo linear, e segunda, pela elaboração de um algoritmo especializado de *Branch and Bound* como uma alternativa de solução.

Para conseguir os novos métodos, neste trabalho de dissertação se fez um estudo teórico dos componentes matemáticos do Problema da Mochila Compartimentada, do método especializado do *Branch and Bound* para problemas de Programação Inteira, junto ao estudo e uso da teoria de programação inteira para fortalecer as informações associadas ao Problema da Mochila Compartimenta. Para a criação do Algoritmo *Branch and Bound* foi estudado um processo de ordenação e criação da árvore de enumeração de soluções factíveis junto ao estudo de limitantes superiores tipo *Backtracking* e via Relaxações Lagrangeanas.

Os principais resultados obtidos neste trabalho foi a definição de um domínio reduzido para a relaxação linear do PMC em sua versão restrita como a formulação de um Modelo Linear Forte, definição de limitantes superiores via Relaxação Lagrangeana, um algoritmo de enumeração de soluções factíveis e um algoritmo *Branch and Bound* para o PMC.

Palavras-chave: Problema da Mochila Compartimentada; Modelo Linear Forte; Relaxação Lagrangeana; *Branch and Bound*; Programação Inteira.

QUIROGA OROZCO, John Jairo. **A BRANCH AND BOUND METHOD FOR THE COMPARTMENTALIZED KNAPSACK PROBLEM**. 2018. 165. Dissertation (Master of Science in Applied Matematics and Computacional's Matematics) – Universidade Estadual de Londrina, Londrina, 2018.

ABSTRACT

The Compartmentalized Knapsack Problem (CKP) is a relatively new type of problem, with a wide application of industrial processes, such as in the case of the metallurgical industry in the cutting of steel coils in two phases, where it emerged. Currently, there are two ways to treat this type of problem by following its mathematical formulations: by its classical formulation which is referred to a nonlinear optimization problem being solved with decomposition heuristics, and second, by its linear formulation, using exact solution methods.

This thesis aims to present the study of new methods of exact solution to the CKP, taking advantage of the linearity of the problem. Two approaches were developed to develop the new methods: one, by means of the strengthening of the linear model, and another, by the elaboration of a specialized algorithm of Branch and Bound as a solution alternative.

To achieve the new methods in the work of dissertation made a theoretical study of the mathematical components of Compartmentalized Knapsack Problem, the specialized method of Branch and Bound to Integer Programming problems with the study and use of programming theory to strengthen the information associated with the CKP. For the creation of the Branch and Bound Algorithm a process of ordering and creation of the feasible solutions tree was studied along with the study of generation of upper boundaries via the Backtracking type and by means of Lagrangian Relaxations.

The main results obtained in this work were the definition of a reduced domain for the linear relaxation of PMC in its restricted version as the formulation of a Strong Linear Model, definition of superior limitants via Lagrangian Relaxation, an algorithm of enumeration of feasible solutions and one Branch and Bound algorithm for the PMC.

Keywords: Compartmentalized Knackpack Problem; Linear Strong Model; Lagrangian Relaxation; Branch and Bound; Discrete Optimization.

QUIROGA OROZCO, John Jairo. **UN MÉTODO DE RAMIFICACIÓN Y PODA PARA EL PROBLEMA DE LA COMPARTIMENTACIÓN DE MOCHILAS**. 2018. 165. Disertación (Maestría en Matemática Aplicada y Computacional) – Universidade Estadual de Londrina, Londrina, 2018.

RESUMEN

El problema de la Mochila Compartimentada (PMC) es un tipo de problema relativamente nuevo, con una amplia aplicación en procesos industriales, como es el caso de la indústria metalúrgica en el corte de bobinas de acero en dos fases, siendo este donde ha surgido. En la actualidad, se tiene dos formas de tratar este tipo de problema siguiendo sus formulaciones matemáticas: del modelo clásico, que se refiere a un problema de optimización no linear, con soluciones enfocadas a uso de heurísticas de descomposición, y en segundo lugar, por su formulación linear, usando métodos de solución exacta.

Este trabajo de disertación tiene como finalidad presentar el estudio de nuevos métodos de solución exacta al PMC, aprovechándose de la linealidad del problema. Dos enfoques fueron hechos para desarrollar los nuevos métodos: una, por medio del fortalecimiento del modelo linear, y otra, por la elaboración de un algoritmo especializado de *Branch and Bound* como una alternativa de solución.

Para lograr los nuevos métodos, en el trabajo de disertación se hizo un estudio teórico de los componentes matemáticos del Problema de la Mochila Compartimentada, del método especializado del *Branch and Bound* para problemas de programación entera, junto al estudio y uso de la teoría de programación entera para fortalecer las informaciones asociadas al PMC. Para la creación del algoritmo del *Branch and Bound* se estudió un proceso de ordenación de las informaciones del problema, la creación de un árbol de enumeración de las soluciones factibles junto al estudio de la generación de limitantes superiores vía tipo *Backtracking* y por medio de relajaciones Lagrangeanas a las restricciones del problema.

Los principales resultados obtenidos en este trabajo fueron la definición de un dominio reducido para la relajación linear del PMC en su versión restringida como la formulación de un modelo linear fuerte para o PMC, definición de limitantes superiores a través de la relajación Lagrangeana, un algoritmo de enumeración de soluciones factibles y un algoritmo *Branch and Bound* para el PMC.

Palabras claves: Problema de la Mochila Compartimentada; Modelo Lineal Fuerte; Relajación Lagrangeana; *Branch and Bound*; Programación Entera.

LISTA DE ABREVIATURAS E SIGLAS

PGMAC Pós-Graduação em Matemática Aplicada e Computacional

B&B Algoritmo *Branch and Bound*

PPL Problema de Programação Linear

PPI Problema de Programação Inteira

PM Problema da Mochila

PMI Problema da Mochila Irrestrita

PMR Problema da Mochila Restrita

PMC Problema da Mochila Compartimentada

PMCR Problema da Mochila Compartimentada Restrito

RLPMCR Relaxação Linear do Problema da Mochila Compartimentada Restrito

PCBA Problemas de Corte de Bobinas de Aço

ML Modelo Linear para o PMCR de [20]

MLF Modelo Linear Forte para o PMCR

RLAGP$_\lambda$ Relaxação Lagrangeana ao problema P para certo λ

RLAGP Dual da Relaxação Lagrangeana ao problema P

RLAGPMCR$_\lambda$ Relaxação Lagrangeana do Problema de Mochila Compartimentada Restrita para certo λ.

RLAGPMCR Dual da Relaxação Lagrangeana para o Problema da Mochila Compartimentada Restrita.

MSFC Método do Subgradiente para Funções Convexas

V(RLAGP$_\lambda$) Solução ótima á RLAGP$_\lambda$

V(RLAGPMCR$_\lambda$) Solução ótima á RLAGPMCR$_\lambda$

V(RLAGPMCR) Solução ótima ao problema Dual Lagrangeano.

LISTA DE ILUSTRAÇÕES

ILUSTRAÇÃO **Página**

LISTA DE TABELAS

TABELA **Página**

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

Nesta dissertação se faz apresentação da pesquisa desenvolvida sobre um Método *Branch and Bound* para o Problema da Mochila Compartimentada.

Trata-se de um problema de otimização combinatória inteira relativamente novo (modelado, formalmente, em [18]), com ampla aplicação em Problemas de Corte e Empacotamento, especialmente nos problemas de corte de duas fases, como é o caso do corte de bobinas de aço na indústria metalúrgica. A Mochila Compartimentada encaixa-se em dois momentos no desenvolvimento dos componentes teóricos do problema: uma primeira etapa, pela formalização e modelação do Problema da Mochila Compartimentada (como versão irrestrita) como um Problema de Programação não linear [18], junto a este, integrando-se a modelagem do problema em seu caso restrito em [27] e [35], tendo como motivação o problema do corte em duas fases de bobinas de aço, bem como o estudo de sua resolução por meio de a heurísticas como em [22]. Uma segunda etapa do problema decorre do estudo de sua linearidade, em [15], com tentativa de reformulação em [6] e em seguida em [20] que apresenta um modelo linear equivalente ao modelo não-linear.

Nesta segunda etapa, o modelo linear do Problema da Mochila Compartimentada oferece a oportunidade de estudar novos métodos de solução, como métodos heurísticos e métodos exatos, por exemplo, o método do *Branch and Bound*, na qual é o objeto de estudo da presente dissertação.

A pesquisa de um método *Branch and Bound* para o Problema da Mochila Compartimentada contempla quatro grandes momentos: o primeiro consiste em reformular o modelo linear num modelo linear "forte", qual visa obter melhores valores na relaxações lineares do problema, procurando eliminar regiões fracionarias no conjunto de soluções factíveis e simetrias do problema, assim também, bem como melhorar os tempos de obtenção da solução de um exemplo. O segundo momento consiste em estabelecer o algoritmo que cria a árvore de enumeração das soluções factíveis para o PMCR. No terceiro momento, foi estudado limitantes superiores, principalmente gerados via relaxação Lagrangeana. O último momento foi o laboratório computacional de avaliação dos algoritmos e reformulações para o problema. Como principais contribuições deste trabalho destaca-se a obtenção de um modelo linear forte para o Problema da Mochila Compartimentada, a obtenção de limitantes superiores e um algoritmo *Branch and Bound* que permite sua resolução exata.

1.1 OBJETIVOS DA PESQUISA

Criar um algoritmo *Branch and Bound* como método de solução exata do Problema da Mochila Compartimentada, usando como base o Modelo Linear do PMCR de [20]. Para atingir o objetivo, foi preciso desenvolver:

- O fortalecimento do Modelo Linear do PMCR de [20], visando obter uma melhor eficiência computacional, em termos de tempos de execução.
- Um algoritmo de *Branching* para criação da árvore de enumeração das soluções factíveis.
- Obter limitantes superiores como critério de *bounding* na árvore de enumeração. Tem-se dois sub-objetivos para o estudo dos limitantes superiores:
  - Criar limitantes superiores tipo *Backtracking* como do modelo Restrito - Irrestrito de [5].
  - Obter limitantes superiores via Relaxação Lagrangeana do problema.

1.2 ORGANIZAÇÃO DO TEXTO

Esta dissertação está composta por seis capítulos junto ao corpo de referências usados de suporte da pesquisa. A seguir, segue uma breve apresentação dos conteúdos dos capítulos.

Capítulo 1. Capítulo introdutório ao trabalho de dissertação.

Capítulo 2. Faz-se a apresentação do Problema da Mochila Compartimentada. Inicia-se com uma introdução do surgimento do problema a partir do Problema de Corte de Bobinas de Aço na indústria metalúrgica. Em seguida, apresenta-se a modelagem matemática do problema desde sua formulação original não-linear, seguindo até a formulação linear. Finaliza-se o capítulo, revisando alguns métodos de resolução do problema.

Capítulo 3. Introduza-se o método especializado *Branch and Bound* para problemas de programação inteira (PPI) e, em particular para o Problema da Mochila Irrestrito, Restrito e $0-1$, fixando as ferramentas e aportes teóricos para à aplicação do método ao PMC.

Capítulo 4. Faz-se estudos da teoria de programação inteira para fortalecer os dados associados ao problema da compartimentação de mochilas, procurando diminuir a região viável do PMCR na busca de obter melhores tempos de execução. Ainda, se faz um melhoramento do modelo linear do PMCR de [20], apresentando um modelo linear forte para o PMCR.

Capítulo 5. Define-se o algoritmo para construir a árvore de enumeração do PMC e estuda-se limitantes superiores para o problema. No caso dos limitantes superiores, foi estudado um limitante similar ao *Branch and Bound* para o PMR, definindo sua qualidade como critério de poda de ramos inviáveis e que não fornecem melhoria no objetivo do problema. Também foi estudado a Relaxação Lagrangeana do PMCR, usando como método de solução o Método do Sub-gradiente para funções convexas, como estratégia de obtenção de limitantes superiores "fortes" para o problema.

Capítulo 6. Neste capítulo apresenta-se experimentos computacionais visados na avaliação da modelagem forte do PMCR e do algoritmo Branch and Bound desenvolvido.

Capítulo 7 Apresenta-se os resultados obtidos na pesquisa de um algoritmo *Branch and Bound* para o Problema de Mochilas Compartimentadas. Ainda, apresenta-se trabalhos futuros para outros estudos do PMCR.

1.3 Considerações Iniciais

Como um preparatório na abordagem da presente dissertação, se faz uma breve introdução aos Problemas de Mochila. Os Problemas de Mochila são problemas de Programação Linear Inteira e são classificados no campo da complexidade computacional (segundo a sua resolução) como problemas *NP-difíceis* (para uma maior compreensão [29] e [20]).

A seguir, é dada uma definição intuitiva e clássica do **Problema da Mochila**: suponha que se tem uma *mochila* com capacidade de C e, suponha também, que se tem à disposição diversos itens que podem serem carregados na mochila. Os itens tem duas características importantes a serem considerados, o "peso" p (que por exemplo, se x itens de peso p são incluídos na mochila, sua capacidade passará a ser $C - px$) e a "utilidade" u (representa um ganho por ter selecionado o item).

e sua utilidade u (define-se um valor real positivo a cada item, de forma que a mair valor em $\mathbb{R}^+$, maior e sua utilidade). O **Problema da Mochila** consiste na escolha de itens a inserir na mochila (suponha-se neste ponto, que se tem itens de al forma que a soma de seus pesos ultrapassa a capacidade)de forma que ofereçam a maior **utilidade** que pode-se levar nela, respeitando sempre a capacidade da mochila (que vão-se denominar como restrição física do problema).

Provalvelmente, a primeira citação sobre o Problema da Mochila pode-se encontrar em Dantzig (1957 apud Cruz, 2010, pág. 3). Na Figura 1.1 pode-se ver uma representação do Problema Clássico da Mochila.

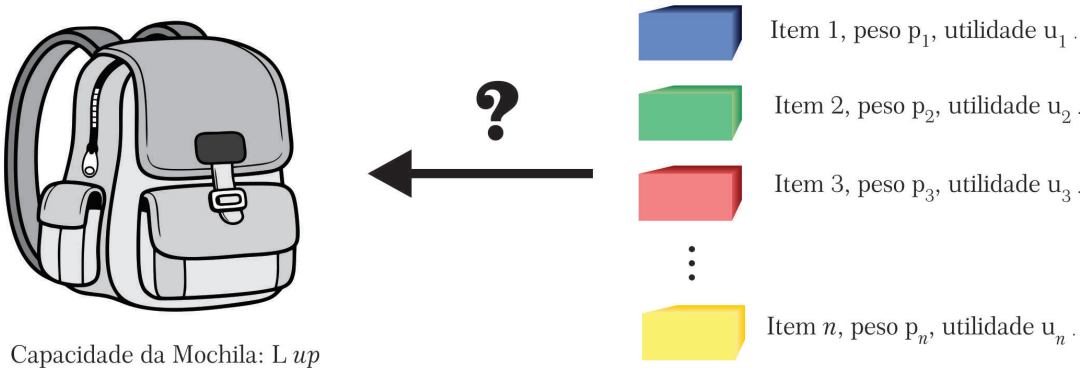

Figura 1.1: Representação do Problema da Mochila
Fonte: Próprio autor.

Voltando na definição clássica da mochila, descreve-se a formula matemática do problema com diversas variações.

Considere n itens , sendo l_i o peso e u_i a utilidade do $i-$ésimo item e considere-se a *variável de decisão* x_i com valor de 1, se o item i é selecionado para integrar a mochila, e 0, caso contrário. O modelo do *Problema da Mochila* $0-1$ está expressado em (1.1)-(1.3):

$$\text{Maximizar: } \sum_{j=1}^{n} u_j x_j \tag{1.1}$$

$$\text{sujeito a: } \sum_{j=1}^{n} l_j x_j \leq L \tag{1.2}$$

$$x_j \in \{0, 1\} \tag{1.3}$$

A equação (1.1) faz referência ao objetivo de maximizar as utilidades dos itens que irão compor a mochila, a restrição (1.2) refere-se à limitação física da mochila, garantindo que a capacidade da mochila não seja violada, isto é, a soma dos pesos dos itens que irão compor a mochila não ultrapassarão a capacidade da mochila, neste caso denominada L. A restrição (1.3) faz referência ao domínio do problema, ou seja, condição de que as variáveis objetivos são apenas variáveis binarias.

Algumas observações a respeito do problema considerado nesta dissertação. Se a soma de todos os pesos dos itens é menor ou igual a capacidade da mochila, a solução (trivial) do problema é $x_i = 1$ para todo $i = 1, ..., n$. Assim, suponha-se que $\sum_{j=1}^{n} l_j \geq L$ e de forma análoga, suponha também que o peso de cada item não ultrapassa L. Agora, note uma generalização ao Problema da Mochila $0 - 1$. Suponha-se que para cada item $i = 1, ..., n$ pode-se escolher mais de uma vez para que serem preenchidos da mochila até certa quantidade disponível (a demanda do item). Com esta consideração, modifica-se a variável objetivo binária a uma variável inteira limitada por uma demanda, definindo assim o *Problema da Mochila Restrita* (PMR) com as formulações (1.4)-(1.7):

$$\text{Maximizar: } \sum_{j=1}^{n} u_j x_j \tag{1.4}$$

$$\text{sujeito a: } \sum_{j=1}^{n} l_j x_j \leq L \tag{1.5}$$

$$0 \leq x_j \leq d_j, \text{ para } j = 1, ..., n \tag{1.6}$$

$$x_j \text{ e inteiro, para } j = 1, ..., n \tag{1.7}$$

No caso de não limitar a quantidade de item que podem serem inseridos ($d_j = \infty$), define-se assim o *Problema da Mochila Irrestrita* (PMI) como segue:

$$\text{Maximizar: } \sum_{j=1}^{n} u_j x_j \tag{1.8}$$

$$\text{sujeito a:} \sum_{j=1}^{n} l_j x_j \leq L \tag{1.9}$$

$$x_j \geq 0 \text{ e inteiro, para } j = 1, ..., n \tag{1.10}$$

Um caso particular do Problema da Mochila $0-1$, é o caso quado $u_i = l_i$ para todo $i = 1, ..., n$, qual, é chamado o *Problema da Soma de Subconjuntos*, conhecido na literatura anglo-saxónica como *Subset Sum Problem* [29]. Este problema será aprofundado no Capítulo 4, o qual será de interesse no processo de modelar problemas de programação inteira mais fortes.

$$\text{Maximizar: } \sum_{j=1}^{n} l_j x_j \tag{1.11}$$

$$\text{sujeito a:} \sum_{j=1}^{n} l_j x_j \leq L \tag{1.12}$$

$$\tag{1.13}$$

Outro tipo de generalização ao problema de Mochila é considerar várias mochilas para inserir os itens. Neste caso, suponha-se que se tem m mochilas de diversas capacidades L_i, para cada $i = 1, ..., m$ e n itens com larguras (pesos) l_j e utilidade u_j para todo $j = 1, ..., n$. Então, define-se o *Problema de Múltiplas Mochilas* $0-1$ como o problema de maximizar a soma das utilidades dos itens a serem preenchidos nas mochilas respeitando suas capacidades. Define-se como as variáveis de decisão x_{ij} para $i = 1, ..., m$ $j = 1, ..., n$ onde, $x_{ij} = 1$ se o item i é preenchido na mochila j, e 0 no caso contrário. O modelo do problema é apresentado abaixo:

$$\text{Maximizar: } \sum_{i=1}^{m} \sum_{j=1}^{n} u_j x_{ij} \tag{1.14}$$

$$\text{sujeito a:} \sum_{j=1}^{n} l_j x_{ij} \leq L_i, \text{ para } i = 1, ..., m \tag{1.15}$$

$$\sum_{i=1}^{m} x_{ij} \leq 1, \text{ para } j = 1, ..., n \tag{1.16}$$

$$x_i \in \{0, 1\}, \text{ para } i = 1, ..., m,\ j = 1, ..., n \tag{1.17}$$

Outros tipos de Problemas de Mochila, por exemplo, *O Problema da Designação Generalizada* e o *Bin-packing*, pode-se estudar em [29], [20] e [18], na qual não serão apresentados neste trabalho por não serem os objetos principais de estudo desta dissertação.

Um problema adicional neste trabalho, é o *Problema da Mochila Encapsulada*, que possui

semelhanças com o *Problema da Mochila Compartimentada*. Na literatura especializada, este tipo de problema é nomeado como *Nested Knapsack Problem*, na qual, este problema consiste em construir cápsulas de tamanhos predefinidos nas quais terão os itens de interesse do problema. Além de cada cápsula pode-se criar novas cápsulas, com tamanhos fixos novamente e assim sucessivamente. As semelhanças entre o *Nested Knapsack Problem* e o PMC (capítulo 2) é do fato de como são consideradas as cápsulas (para o *Nested Knapsack Problem*) e os compartimentos (para PMC), que são fixas e flexíveis, respectivamente.

Segue um exemplo de [35] para ilustrar o problema: precisa-se selecionar itens que devem serem levados até um ponto de difícil acesso (por exemplo uma região montanhosa). Inicialmente os itens são transportados num vagão de trem até certo ponto que é possível o transporte rodoviário, em seguida,os itens são transportados por meio de pequenos furgões até outro ponto em que o transporte só pode ser feito por carregadores. A carga de cada carregador deve ser proveniente de somente um dos forgões. Estes três tipos de transportes existe uma capacidade de carga e custo operacional associado. O problema consiste em selecionar itens que serão carregados em cada etapa do transporte, de tal forma que, seja maximizada a utilidade dos itens e minimizado os custos de transporte.

Para solução deste problema é considerado um modelo do *Nested Knapsack Problem* de Johnston e Khan (1995 apud Hoto *et al.*, 2003, pág. 58) assim sejam as variáveis de decisão: x_{ij}^s, onde $x_{ij}^s = 1$ se o item i for designado para a mochila j na etapa de transporte s, e 0 no caso contrário, e y_j^s, onde $y_j^s = 1$ se a mochila j é usada na etapa de transporte s, e igual a 0 no caso contrário. Tem-se também os parâmetros γ^s e c^s que representam, respectivamente, o custo e a capacidade da mochilas nas etapas de transporte $s = 1, ..., k$. O modelo é dado por:

$$\text{Maximizar: } \sum_{i=1}^{m} u_i x_{i1}^1 - \sum_{j=1}^{m}\sum_{s=2}^{k} \gamma^s y_j^s \tag{1.18}$$

$$\text{sujeito a: } \sum_{j=1}^{m} p_i x_{i1}^1 \leq c \tag{1.19}$$

$$\sum_{j=1}^{m} p_i x_{ij}^s \leq c^s y_j^s, \text{ para } j = 1, ..., m, \; s = 2, ..., k \tag{1.20}$$

$$x_{i1}^1 \leq \sum_{j=1}^{m} x_{ij}^s, \text{ para } i = 1, ..., m, \; s = 1, ..., k \tag{1.21}$$

$$\max_j(x_{pj}^s + x_{qj}^s) \leq \max_j(x_{pj}^{s-1} + x_{qj}^{s-1}), \tag{1.22}$$

$$\text{para } s = 3, ..., k \text{ e } p, q = 1, ..., m, \; p \neq q$$

$$x_{ij}^s, y_j^s \in \{0, 1\}, \; i, j = 1, ..., m, \; s = 1, ..., k. \tag{1.23}$$

2 PROBLEMA DA MOCHILA COMPARTIMENTADA

O Problema da Mochila Compartimentada é o problema de concentração da presente dissertação. Neste capítulo será apresentado o surgimento do problema e sua formalização sob o ênfase na criação de padrões de corte de bobinas de aço em duas fases para a indústria Metalúrgica. Em seguida apresenta-se sua formulação matemática desde sua concepção não linear [18] até sua modelagem linear [20]. Finalmente, mostra-se algumas abordagens de solução ao problema visado em heurísticas e a motivação do estudo da resolução do problema através de um algoritmo de *Branch and Bound* que será exposto nos seguintes capítulos.

Tem-se uma mochila de capacidade L e n tipos de itens indexados em um conjunto N (o item $i = 1, ..., n$, onde $i \in N$), que podem serem preenchidos na mochila. Cada item $i \in N$ tem associado um "peso" (p_i) e uma "utilidade" (u_i). O problema de mochila clássico consiste em determinar quais itens e quantos deles (dependendo da demanda disponíveis de cada item) podem serem preenchidos na mochila afim de obter a máxima utilidade deles respeitando a capacidade da mochila (veja-se o Capítulo 1).

Uma variação deste problema clássico é: classifique os itens sob algum critério e faça uma divisão destes em q agrupações disjuntas, ou seja, faça uma partição do conjunto N em $N = \bigcup_{k=1}^{q} N_k$ com $\bigcap_{k=1}^{q} N_k = \varnothing$. Inserir os itens $i \in N$ terá uma condição: itens de diferentes classes não podem misturar-se na mochila, ou seja, os itens $i \in N_s$ e $i \in N_g$ não podem serem inseridos "juntos" na mochila se $s, g \in C = \{1, 2, ..., q\}$, onde C é o conjunto das classes, tem-se que $s \neq g$.

Para conseguir inserir itens de diversas classes na mochila é necessário *construir* dentro da mochila **compartimentos** onde serão alocados itens da mesma afinidade (da mesma classe). Estes compartimentos não estão previamente determinados, o qual serão definidos pelos itens escolhidos para preencher estes compartimentos.

O ***Problema da Mochila Compartimentada*** (PMC) consiste em determinar adequadamente as capacidades dos *compartimentos* que podem serem criados para cada classe $k = 1, ..., q$ na busca de obter a *máxima* utilidade possível deles.

O problema da Mochila Compartimentada tem seu surgimento na prática industrial, especialmente no processo de corte de bobinas de aço em duas fases (ver. [18]), onde são derivados diversos problemas de corte de bobinas na procura do maior aproveitamento e benefício que pode-se ter destas, sendo categorizados como Problemas de Corte de Bobinas de Aço (PCBA). Neste trabalho de dissertação apresenta-se de forma concisa o PCBA (com fines teóricos) com a intenção de introduzir e apresentar formalmente o Problema da Mochila Compartimentada.

2.1 HISTÓRICO DO PROBLEMA

O Problema da Mochila Compartimentada foi tratado de forma implícita (em seu caso restrito) em trabalhos como [13], [8], [33], [37] e [38]. Os primeiros elementos matemáticos na abordagem do Problema da Mochila Compartimentada são encontrados em [17] para em seguida ser definido de forma concisa em um modelo matemático do Problema da Mochila Compartimentada Irrestrito (PMC) em [18]. Além da apresentação do modelo para o PMC, [18] faz um estudo detalhado da aplicação do PMC ao problema de corte de bobinas de aço (PCBA), mostra um algoritmo de compartimentação exata chamado *COMPEX* visado a solucionar o Problema, uma heurística de compartimentação (na busca de acelerar o processo de compartimentação) nomeado *COMPMT* (onde utiliza limitantes de [29]) com um desempenho favorável em comparação com o Algoritmo de Decomposição *COMPEX*. Também faz definição do Problema da Mochila Compartimentada $0-1$ com um método *Branch and Bound* para sua resolução.

[27] faz um estudo aprofundado ao caso restrito do Problema da Mochila Compartimentada (PMCR) apresentado um modelo para o problema junto a heurísticas visado a sua resolução, como são Heurística de Decomposição, Heurística do Melhor Compartimento e Heurística das "z" Melhores Compartimentos. Em [28] foi apresentado a Heurística das "W" Capacidades, na qual forneceu melhores comportamentos computacionais com as heurísticas expostas em [27]. Além do anterior, no mesmo trabalho, foi feita uma relaxação linear do PMC na qual é obtido um limitante superior para o problema.

Em [16] faz uso da decomposição da compartimentação das mochilas para formular uma heurística com geração de colunas chamado de Heurística de Compartimentação Restrita com Geração de Colunas, na qual se afirma ter maior desempenho em comparação da Heurística das "W" capacidades de [28], especialmente para exemplares com número pequeno de agrupamento de itens.

Uma primeira abordagem do PMCR como um problema linear (PMCRL) foi exposto em [15], com uma nova formulação da função objetivo e suas restrições, junto à apresentação de heurísticas de retro-alimentação ($retro$), melhor compartimento para W capacidades com retro-alimentação ($WRetro$) e heurística do melhor compartimento para W capacidades W vezes (WW). [6] dá continuidade das abordagens lineares ao PMCR com uma nova modificação da função objetivo e das restrições do problema apresentado em [15], com uso das heurísticas expostas anteriormente. Em [20] apresenta a formalização de um modelo linear para o PMCR baseado em [6], onde faz a demostração de sua linearidade e também apresenta um Algoritmo de Descomposição Exaustiva para resolver de forma exata o PMCR.

Em [22], tem-se um novo tratamento linear para o PMCR junto a uma heurística para resolve-lo chamada de *Hybrid Heuristic* onde faz fusão das "z" Melhores Compartimentos e as "W" Capacidades para o problema. Também apresenta um método para resolver o problema mestre restrito do PMCR usando o método de geração de colunas, chamado

de *Dynamic Column Generation Method*. Em [23], faz o tratamento do Problema de Mochila Compartimentado como um problema bidimensional, formulando diversos modelos matemáticos para o problema bidimensional, sendo considerado para sua formulação estágio de processos de cortes guilhotinados em três etapas.

O modelo linear feito em [20] foi aceito para publicação em revista científica aparecendo em [19]. Além do modelo linear, apresenta no mesmo trabalho, uma nova heurística para o PMCR, usando o modelo clássico do problema, chamado de *The Decreasing Classes Heuristic*.

Seguindo os resultados obtidos no trabalho de [20] e [19] com o modelo linear para o PMCR, será utilizada a nova abordagem da mochila compartimentada para explorar novos métodos de solução exata ao problema, tendo como base e objetivo o estudo de um algoritmo de enumeração sistemático chamado Algoritmo Especializado *Branch and Bound*, na qual, será abordado no Capítulo 5.

2.2 Problema de Corte de Bobinas de Aço

Traslada-se a ideia intuitiva apresentada no inicio da seção introdutória deste capítulo para ser representada como um Problema de Corte de Bobinas de Aço, mais especificamente, ao PCBA em duas fases, fazendo assim uma introdução mais formal ao problema. O PCBA terá um breve tratamento neste trabalho, do fato que terá só fins teóricos, maiores informações sobre os PCBA pode ser consultadas em [18].

Inicia-se apresentando os elementos involucrados em um PCBA em duas fases: bobinas mestre de aço, bobinas intermediárias e fitas. Define-se como *bobinas mestres de aço*, as bobinas em estoque que serão submetidos aos processos de corte. Pode-se fazer a comparação análoga das bobinas mestres com as mochilas a serem preenchidas. As bobinas mestres de aço são sometidas a um primeiro processo de corte, obtendo *sub-bobinas de aço*, chamadas de *bobinas de aço intermediárias*. Compara-se as bobinas intermediárias com os compartimentos a serem criados em uma mochila. Das bobinas intermediárias são submetidas um novo processo de corte obtendo destas as *fitas*, sendo estas o correspondente aos itens que vão compor a mochila. Elementos técnicos derivados no tratamento de ditas bobinas de aço pode-se ver em [18] e [8]. Na Figura 2.3 podes-se ver a representação descrita acima.

O PBCA em duas fases, consiste na forma da cortar *eficientemente* uma bobina mestre de aço para obter as fitas. As fitas são os insumos para produzir *tubos de aço*, que são requeridos na indústria metalúrgica para diversas aplicações, como tubos para caldeiras, para a indústria automobilística, para a construção civil, para a confecção de bicicletas, etc (veja-se [18]). De acordo com o requerimento industrial, as fitas apresentam diversas dimensões referidas a largura e espessura (em [18] tem-se que as bobinas mestres tem pesos que variam $1200\ Kg$ a $13500\ Kg$; suas largura varia de $1100\ mm$ a $1200\ mm$ e com variação das espessuras do aço entre $0,90\ mm$ a $5,10\ mm$), sendo assim necessário criar etapas no processamento das bobinas de aço. Usualmente, as bobinas mestre de aço sofrem um primeiro processo de corte gerando

bobinas intermediárias que apresentam afinidades em questão de espessura, que em seguida são novamente cortadas já com as larguras definidas na procura industrial para assim obter as fitas. A produção das fita é obtida através de um processo industrial chamado de *corte* e *laminação* das bobinas de aço. O processo de *laminação* consiste na diminuição da espessura da lâmina de aço das bobinas intermediárias por meio do submetimento de um sistema de pressão (Ver Figura 2.1). As máquinas de corte das bobinas intermediárias e fitas são diferentes, tendo assim limitantes técnicos, fazendo que este processo seja feito nos dois processos de corte descritos.

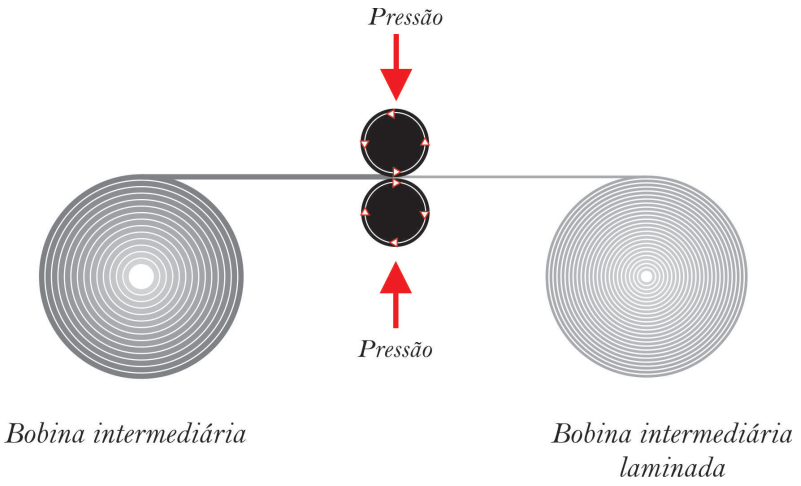

Figura 2.1: Representação do Processo de laminação das bobinas intermediárias de aço.
Fonte: Próprio autor.

A linha de produção inicia com o processo de corte das bobinas mestres (primeira fase) seguindo as afinidades de espessura das fitas que será criados (na qual é relacionado como os compartimentos a serem criados), assim obtendo as bobinas intermediárias (cada bobina intermediara pode-se ver como uma combinação finita de fitas). Cada bobina intermediária é submetida ou não ao processo de laminação dependendo dos requisitos de cada fita. Em seguida da laminação, o resultante passa pelo processo do *encruador* com a finalidade de corrigir as imperfeições sofridas no processo anterior. Seguidamente faz-se um segundo processo de corte que produz as fitas (segunda fase). Finalmente as fitas passam por um processo de *recozimento* para corrigir imperfeições físicas adquiridos em toda a linha de produção. A laminação, encruador e recozimento não tem influências significativas no estágios de corte. Na Figura 2.2 pode-se ver a representação da linha de produção descrita.

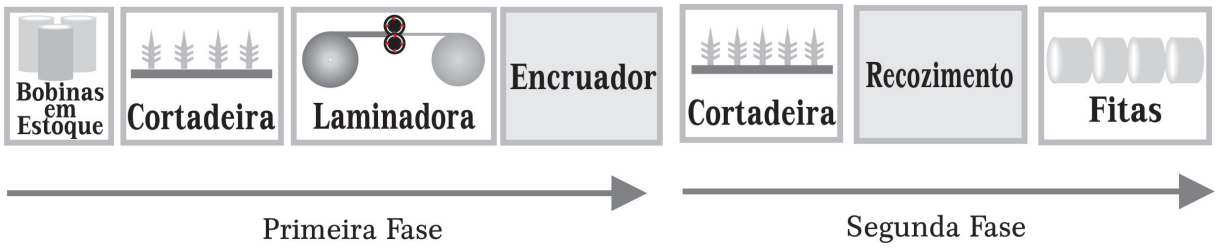

Figura 2.2: Linha de produção para a obtenção das *fitas* de aço.
Fonte: Próprio autor.

O processo descrito anteriormente induz a necessidade de criar padrões de corte nas bobinas mestres para cada fase de corte (no momento nada evidentes), onde, procura-se o maior aproveitamento desta (ou seja, obter a menor quantidade de *sobras* das bobinas e um tratamento adequado dos *refugos*, que são as perdas inevitáveis aos momentos de corte) e a maior utilidade que pode-se fornecer com a criação das fitas. Todo o processo finalizado pode-se ver na Figura 2.3.

Com uma referência do processo de cortes de bobinas de aço a duas fases e uma ilustração dos PCBA, na Seção 2.3 estuda-se as formulações matemáticas para o problema.

2.3 Modelagem Matemática para o Problema da Mochila Compartimentada

Antes de apresentar o modelo matemático do Problema da Mochila Compartimentada (PMC), segue um exemplo que permite ilustrar e ligar a definição intuitiva do problema junto ao processo de corte de duas fases, para assim, facilitar a compreensão do modelo.

Exemplo 1. Considere uma mochila com capacidade (L) de 25 uc (uc unidades de comprimento). Os itens disponíveis para preencher a mochila são os que se apresentam na Tabela 2.1. Tem-se dois grupos (classes) de itens que não podem misturar (no sentido dos problemas de corte de bobinas de aço, por questão dos corte das bobinas intermediárias na primeira e das fitas na segunda fase). Seja $N = \{1, 2, 3, 4\}$, conjunto dos índices dos itens, onde, $N_1 = \{1, 2\}$ é o conjunto dos índices dos itens da classe 1 e $N_2 = \{3, 4\}$ é o conjunto dos índices dos itens da classe 2, tal que $N = N_1 \cup N_2$ e $N_1 \cap N_2 = \varnothing$. O que quer dizer, por exemplo, que não se pode inserir em um compartimento da mochila o item 1 da classe 1 com algum item da classe 2.

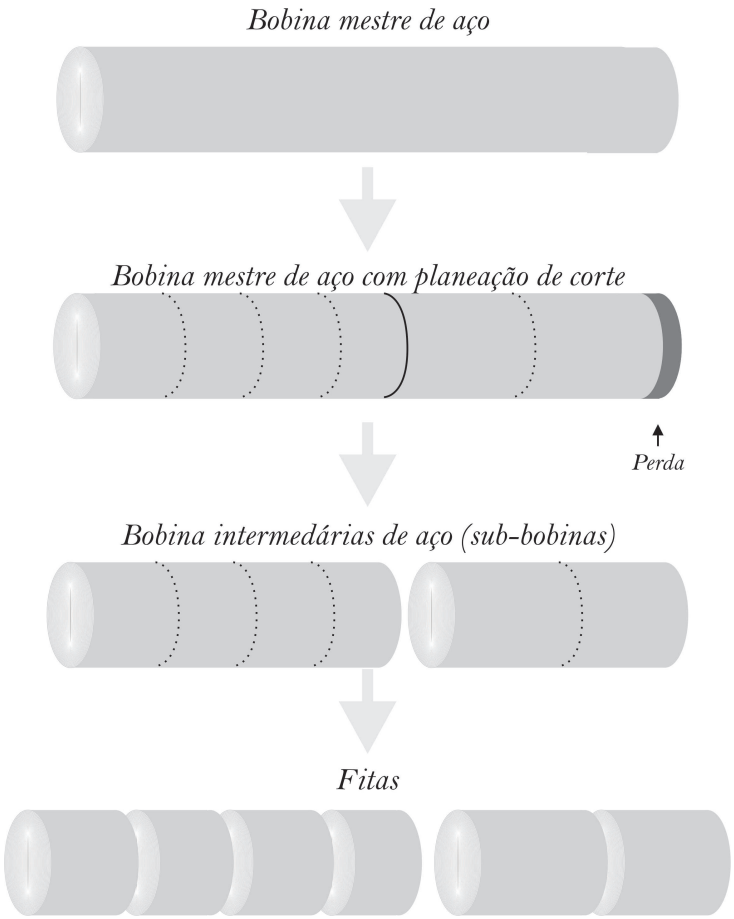

Figura 2.3: Representação das bobinas mestres, bobinas intermediárias (sub-bobinas) e as fitas de aço.

Fonte: Próprio autor.

Classe (k)	**Item (i)**	**Largura ("peso",** l_i**)**	**Utilidade** (u_i**)**
1	1	6	8
	2	8	13
2	3	9	11
	4	7	15

Tabela 2.1: Informações do Exemplar 1 de um PMC.

Precisa-se criar compartimentos dentro da mochila para inserir os itens sob alguns parâmetros. No caso do PMC, define-se limitantes dos compartimentos que podem ser criados na mochila (lembrando-se que estes limitantes estão referidos com os limitantes técnicos das máquinas nos processo de corte a duas fases das bobinas de aço). Neste exemplo, define-se como limitantes o seguinte: as combinações de itens a escolher da classe 1, que a soma de suas capacidades esteja entre 8 uc (L^1_{min}) e 16 uc (L^1_{max}). Para os itens da classe 2 a soma de suas capacidades esteja entre 9 uc (L^2_{min}) e 15 uc (L^2_{max}). Um resumo dos limitantes e capacidade da mochila, pode-se ver na Tabela 2.2.

Classe (k)	L^k_{min}	L^k_{max}	L
1	8	16	25
2	9	15	

Tabela 2.2: Limitantes das capacidades dos compartimentos do Exemplar 1.

Com as informações do problema, procura-se agora enumerar os possíveis compartimentos (bobinas intermediárias) que pode-se criar (compartimentos viáveis). Observe-se que para a classe 1, pode-se criar um compartimento que tenha só o item 2, já que satisfaz que $8 \leq l_2 \leq 16$. Outra possibilidade é criar um compartimento com dois itens 2 da classe 1, onde satisfaz que $8 \leq 2 \cdot l_2 \leq 16$. Outra possibilidade também pode ser um compartimento com os itens 1 e 2 tal que $8 \leq l_1 + l_2 \leq 16$.

Seguindo o raciocínio anterior, define-se $a_j = (a_{1j}, a_{2j})$, como o compartimento de índice j da classe 1 que satisfaz a relação $8 \leq 6a_{1j} + 8a_{2j} \leq 16$. Com a notação definida para os compartimentos, representa-se o dito anteriormente por, respectivamente, como $a_1 = (0, 1)$, $a_2 = (0, 2)$, $a_3 = (1, 1)$. Outros compartimentos que pode-se criar, e que não foram enumerados em relação aos itens da classe 1, é $a_4 = (2, 0)$. Os compartimentos a_1, a_2, a_3 e a_4 são os compartimentos viáveis para a classe 1. De forma análoga para a classe 2, define-se o compartimento $a_j = (a_{3j}, a_{4j})$, como o compartimento de índice j da classe 2 que satisfaz a relação $9 \leq 9a_{3j} + 7a_{4j} \leq 15$. Os compartimentos possíveis de serem construídos para a classe 2 são $a_5 = (1, 0)$ e $a_6 = (0, 2)$,

Com os compartimentos viáveis para as classes 1 e 2, define-se os conjuntos V_1 com os índices de todos os compartimentos viáveis criados para a classe 1, ou seja, $V_1 = \{1, 2, 3, 4\}$ e V_2 com os índices de todos os compartimentos viáveis criados para a classe 2 como $V_2 = \{5, 6\}$. Tem-se o conjunto $V = V_1 + V_2 = \{1, 2, 3, 4, 5, 6\}$ como o conjunto de índices de todos os compartimentos viáveis para a mochila em questão.

Agora, define-se o critério para escolher os compartimentos que será parte da mochila. Lembrando que a capacidade da mochila mochila é de $L = 25uc$, junto ao peso associado a cada um dos itens, então, escolha-se para preencher a mochila os compartimentos que cumprem com $6a_{1j}+8a_{2j}+9a_{3j}+7a_{4j} \leq 25$, que é, a restrição da mochila. Observe-se que, uma possível *compartimentação* da mochila poderia ser com os compartimentos a_3 ($3 \in V_1$) e a_5 ($5 \in V_2$), que do fato cumprem com a restrição da mochila, ou seja, tem-se que $6 \cdot a_{13} + 8 \cdot a_{23} + 9 \cdot a_{35} + 7 \cdot a_{45} = 6 \cdot 1 + 8 \cdot 1 + 9 \cdot 1 + 7 \cdot 0 = 23 \leq 25$. Pelo anterior, esta compartimentação feita tem com utilidade associado $u_1 \cdot a_{13} + u_2 \cdot a_{23} + u_3 \cdot a_{35} + u_4 \cdot a_{45} = 8 \cdot a_{13} + 13 \cdot a_{23} + 11 \cdot a_{35} + 15 \cdot a_{45} = 32$. Observe-se também que é possível criar um só compartimento (com a combinação de compartimentos de uma mesma classe), por exemplo, com duas vezes o compartimento a_4, tem-se uma utilidade também de $32\ uu$ (uu unidade de utilidade).

Em seguida, considere-se a mochila do Exemplo 1 como uma bobina de aço e os itens descritos na Tabela 2.1 as fitas a produzir. Na Figura 2.4 pode-se uma representação das informações da Tabela 2.1.

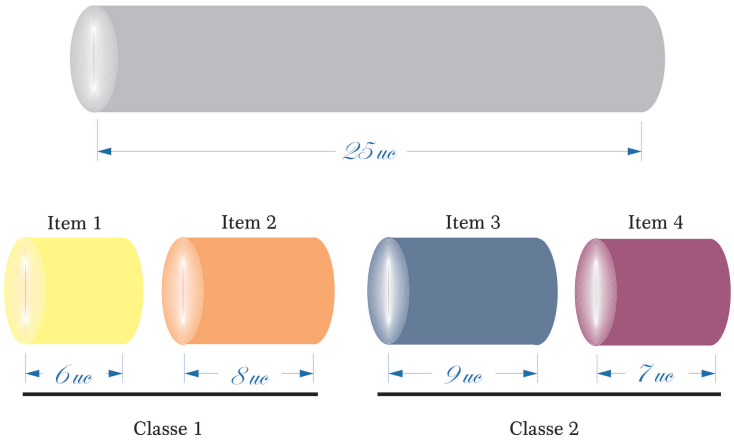

Figura 2.4: Representação das informações da Tabela 2.1 do Exemplo 1 de compartimentação de uma mochila.

Fonte: Próprio autor.

Os possíveis compartimentos que podem ser criados para cada classe $V_1 = \{1, 2, 3, 4\}$ e $V_2 = \{5, 6\}$, neste caso corresponderiam as bobinas intermediárias, são mostrados na Figura 2.5.

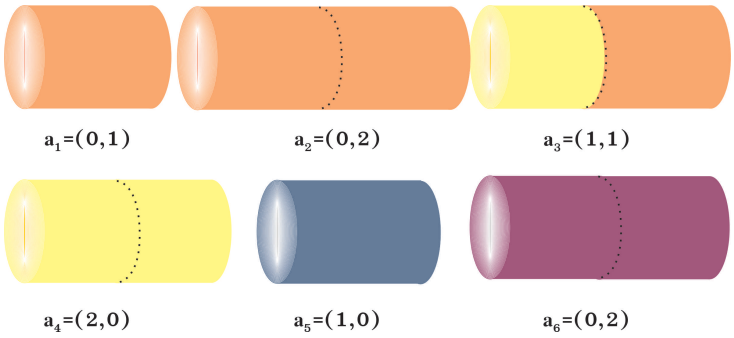

Figura 2.5: Compartimentos factíveis para o Exemplar 1.

Fonte: Próprio autor.

A duas compartimentações da mochila com a obtenção de 32 uu apresentado acima pode-se ver como dois *padrões de corte* para a bobina mestre de aço, como se ilustra na Figura 2.6. Com este resultado, decidir qual padrão é melhor depende do tipo de objetivo inicial da elaboração das fitas e decisões operativas da indústria. Por exemplo, define-se um custo pela utilização de um compartimento (devido a seu processamento operativo), suponhamos um valor de 5 uv (onde uv representa as unidades do custo de produção) para cada compartimento da classe 1 e um valor de 3 uv para os compartimentos da classe 2, tem-se como custos de 10 uv para o primeiro padrão e 7 uv para o segundo padrão, obtendo assim, como melhor opção pelo menor custo na geração das bobinas intermediárias o padrão 2.

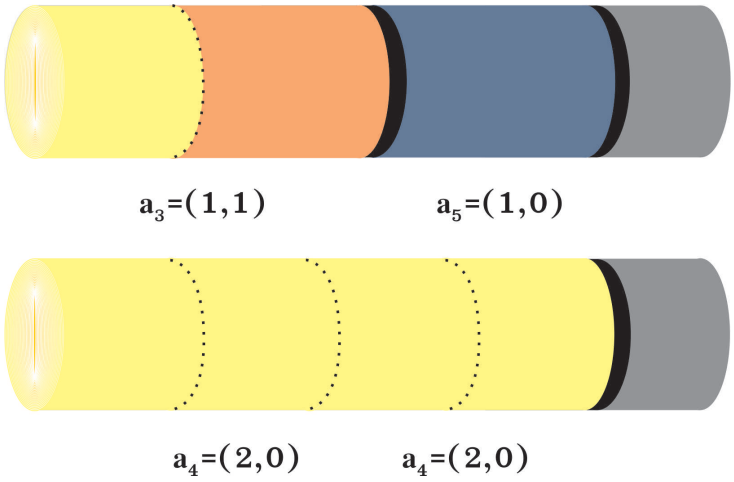

Figura 2.6: Dois padrões de corte para a bobina mestre de aço do Exemplo 1.
Fonte: Próprio autor.

Com a intenção de fornecer uma compartimentação da mochila e inserir itens de tal forma de obter a maior utilidade com as informações das tabelas 2.1 e 2.2, tem-se como padrão de corte, que define a solução ótima ao problema, o qual que é formado por os compartimentos a_1 e a_6, onde vão-se criar um compartimento de a_1 e dois de a_6, obtendo assim, uma utilidade associada de $43\ uu$. Na Figura 2.7 pode-se ver o resultado da compartimentação.

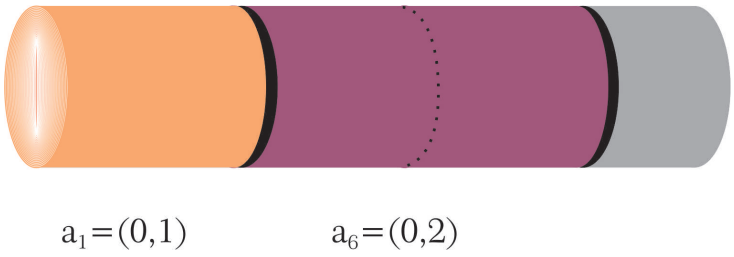

Figura 2.7: Melhor padrão de corte do Exemplo 1 com a obtenção da máxima utilidade.
Fonte: Próprio autor.

Após da apresentação intuitiva do problema e a mostra de um exemplo de mochila com compartimentação (Exemplo 1), tem-se a continuação a formulação matemática ao PMC apresentado por [18].

Considere-se uma mochila com capacidade L. Indexe-se os itens globalmente (item 1, item 2,..., item s,...) e define-se o conjunto N como conjunto de índices dos itens. Suponha-se que o conjunto N esteja particionado em q classes, de tal forma que para cada classe tem-se um subconjunto de índices onde são agrupados, ou seja N_k, $k = 1, ..., q$. Então, verifique-se que $N = \bigcup_{k=1}^{q} N_k$ e $N_p \cap N_t = \varnothing$, para todo $p, t \in \{1, 2, ..., q\}$ e $p \neq t$. Na mochila precisa-se construir compartimentos onde será inseridos os itens de $N_k, k = 1, ..., q$ (um compartimento só pode ser preenchido com itens de uma mesma classe), definindo a variável de decisão a_{ij}, qual indicará a quantidade de itens i no compartimento j. Em seguida para cada classe são definidos

restrições de capacidades (limitantes), entre um valor inteiro mínimo L^k_{min}, $k = 1, ..., q$ e um valor inteiro máximo L^k_{max}, $k = 1, ..., q$. Seguindo as capacidades anteriores e com informações dos compartimentos que podem compor a mochila (compartimentos viáveis), crie-se o conjunto V_k, conjunto formado pelo índices dos compartimentos viáveis gerados por itens de índices em N_k. Observe-se que, para cada N_k com $k = 1, ..., q$, tem-se associado um único V_k tal que $V = \bigcup_{k=1}^{q} V_k$ e $V_p \cap V_t = \varnothing$, para todo $p, t \in \{1, 2, ..., q\}$ e $p \neq t$.

Define-se u_i a utilidade e l_i o peso do item $i \in N$, a variável de decisão a_{ij} que é quantidade de itens de índice i no compartimento j, y_j definirá a quantidade de compartimentos j poderá compor a mochila e γ_j é o custo por construir o compartimento j.

Num problema de cortes de bobinas de aço em duas fases, veja-se uma *classe* como o conjunto de itens que serão sometidos a um mesmo processo de laminação. Em dito processo, os itens devem ser laminados juntos onde devem ser respeitados as restrições técnicas da máquina laminadora, ou seja, devem ser especificados uma largura máxima e mínima dos itens que podem-se tratar nela. Neste caso o peso de cada item corresponderá à largura da fita.

A formulação matemática apresentada em [18] consiste em:

$$\text{Maximizar } \sum_{j \in V_1} \left(\sum_{i \in N_1} u_i a_{ij} - \gamma_j \right) y_j + \cdots + \sum_{j \in V_q} \left(\sum_{i \in N_q} u_i a_{ij} - \gamma_j \right) y_j \tag{2.1}$$

$$\text{sujeito a: } \sum_{j \in V_1} \left(\sum_{i \in N_1} l_i a_{ij} \right) y_j + \cdots + \sum_{j \in V_q} \left(\sum_{i \in N_q} l_i a_{ij} \right) y_j \leq L \tag{2.2}$$

$$L^k_{min} \leq \sum_{i \in N_k} l_i a_{ij} \leq L^k_{max}, \text{ para } j \in V_k, k = 1, ..., q \tag{2.3}$$

$$a_{ij}, y_j \geq 0 \text{ e inteiros, com } i \in N = \bigcup_{k=1}^{q} N_k \text{ e } j \in V = \bigcup_{k=1}^{q} V_k \tag{2.4}$$

Tem-se na formulação com função objetivo do PMC o descrito em (2.1). A restrição (2.2) estabelece que a soma das capacidades dos compartimentos criados e com itens inseridos não ultrapassa a capacidade da mochila. A restrição (2.3) refere-se à limitação da capacidade que tem cada compartimento. A restrição (2.4) determina a integralidade das variáveis de decisão. Para evitar soluções triviais ao PMC admita-se que $l_i \leq L^k_{max} \leq L$, para todo $i \in N_k$, $k = 1, .., q$. Pode-se observar que o PMC sob a formulação descrita acima é um problema *não linear*.

Caso de que cada compartimento só seja utilizado uma vez ou não na mochila, restringe-se a variável y_j a tomar valores entre $\{0, 1\}$ (1 se é utilizado, 2 se não é utilizado), definindo assim o *Problema da Mochila Compartimentada 0-1* (PMC 0-1). Veja a formulação apresentada por [18]

$$\text{Maximizar } \sum_{j\in V_1}\left(\sum_{i\in N_1} u_i a_{ij} - \gamma_j\right) y_j + \cdots + \sum_{j\in V_q}\left(\sum_{i\in N_q} u_i a_{ij} - \gamma_j\right) y_j \tag{2.5}$$

$$\text{sujeito a: } \sum_{j\in V_1}\left(\sum_{i\in N_1} l_i a_{ij}\right) y_j + \cdots + \sum_{j\in V_q}\left(\sum_{i\in N_q} l_i a_{ij}\right) y_j \leq L \tag{2.6}$$

$$L^k_{min} \leq \sum_{i\in N_k} l_i a_{ij} \leq L^k_{max}, \text{ para } j \in V_k,\ k = 1, ..., q \tag{2.7}$$

$$y_j \in \{0,1\},\ a_{ij} \geq 0 \text{ e inteiros, com } i \in N = \bigcup_{k=1}^{q} N_k \text{ e } j \in V = \bigcup_{k=1}^{q} V_k \tag{2.8}$$

Observe-se que a ideia base na formulação do modelo matemático para PMC irrestrito foi em supor a construção de todos os compartimentos viáveis (bobinas intermediárias) para em seguida ser escolhido aqueles que deverão ser parte da compartimentação. Caso que o conjunto V fosse previamente determinado, tem-se que os valores $L_j = \sum\limits_{i\in N_k} l_i a_{ij}$ e $U_j = \sum\limits_{i\in N_k} u_i a_{ij}$, onde representam respectivamente a largura e a utilidade do $j \in V_k$, estariam bem determinados também. Pelo fato do que L_j e U_j estão bem determinados com $j \in V_k$, tem-se que a condição (2.3) seria satisfeita. Assim, reformulando (2.1)-(2.4) produz um Problema de Mochila Irrestrito (desconsiderando o custo da criação do compartimento):

$$\text{Maximizar } \sum_{j\in V_1} U_j y_j + \cdots + \sum_{j\in V_q} U_j y_j \tag{2.9}$$

$$\text{sujeito a: } \sum_{j\in V_1} L_j y_j + \cdots + \sum_{j\in V_q} L_j y_j \leq L \tag{2.10}$$

$$y_j \geq 0 \text{ e inteiros, com } j \in V = \bigcup_{k=1}^{q} V_k \tag{2.11}$$

Com a formulação anterior pensa-se em uma tentativa de gerar todo o conjunto V para o PMC , mas computacionalmente é inviável para grandes exemplares. Segue-se que para cada classe o conjunto de compartimentos factíveis os elementos deles são candidatos a serem possibilidades de solução factível para o problema de mochila. Examinar no conjunto dos compartimentos no pior dos casos é necessário verificar no conjunto das partes de V ($P(V)$) qual conjunto possui a maior utilidade ao problema, gerando assim a necessidade de examinar, pelo menos, $2^n - 2$ possibilidades, deixando qualquer tentativa de resolução exaustiva para grandes exemplares inviável. Como se comentou no capítulo introdutório deste trabalho (veja-se o Capítulo 1), O Problema da Mochila Irrestrito está classificado sob a complexidade computacional como *NP-difícil*, podendo afirmar que o PMC "não é mais difícil" do que o PMI, por tanto PMC é *NP-difícil* (veja-se [20]).

2.4 PROBLEMA DA MOCHILA COMPARTIMENTADA RESTRITO

Como se observou na Figura 2.2 (linha de produção das fitas a partir das bobinas mestres de aço), no processo de corte de estoque em duas fases, tem-se a considerar outras restrições que aparecem na linha de produção, produto novamente das limitações técnicas das máquinas involucradas no processo de corte das bobinas mestre de aço.

Considere-se como d_i, a demanda exigida a produzir para o item (fita) i. Assim, na busca de garantir que não sejam elaborados itens em excesso, realiza-se a restrição seguinte:

$$\sum_{k=1}^{q} \sum_{j \in V_k} a_{ij} y_j \leq d_i, \, i \in N = \bigcup_{k=1}^{q} N_k \tag{2.12}$$

Veja-se que a soma das quantidades de um determinado item i para cada compartimento que o produz é limitada por sua demanda d_i.

Agora considere-se os processos de corte na primeira e segunda fase na linha de produção. Dito processo de corte, pode-se ver a forma com que os objetos são alocados na mochila. Tem-se uma série de facas (finitas) que cortam a bobina mestre de aço e também as bobinas intermediárias (conjunto de fitas que sofreram o mesmo processo de laminação) para a produção das fitas, onde o dito processo é limitado (restrição técnica das cortadeiras). Define-se uma constante inteira F_1 indicando o número de facas disponíveis no primeiro processo de corte. Enquanto ao corte da segunda fase, define-se com a constante inteira F_2 o número de facas disponíveis para o processo. Com o anterior, tem-se duas novas restrições, F_1 vão limitar o número de compartimentos que podem ser criados e F_2 vão limitar o número de itens que podem ser inseridos. A formulação da limitação do F_1 é:

$$\sum_{k=1}^{q} \sum_{j \in V_k} y_j \leq F_1 \tag{2.13}$$

A formulação da limitação imposta para o segundo processo de corte, F_2, é como segue a continuação:

$$\sum_{i \in N_k} a_{ij} \leq F_2 \text{ para } j \in V = \bigcup_{k=1}^{q} V_k, \, k = 1, ..., q \tag{2.14}$$

Com as observações descritas e as formulações (2.12),(2.13) e (2.14) obtém a formulação ao Problema da Mochila Compartimentada Restrito:

$$\text{Maximizar } \sum_{j\in V_1}\left(\sum_{i\in N_1} u_i a_{ij} - \gamma_j\right) y_j + \cdots + \sum_{j\in V_q}\left(\sum_{i\in N_q} u_i a_{ij} - \gamma_j\right) y_j \tag{2.15}$$

$$\text{sujeito a: } \sum_{j\in V_1}\left(\sum_{i\in N_1} l_i a_{ij}\right) y_j + \cdots + \sum_{j\in V_q}\left(\sum_{i\in N_q} l_i a_{ij}\right) y_j \leq L \tag{2.16}$$

$$\delta_j L^k_{min} \leq \sum_{i\in N_k} l_i a_{ij} \leq \delta_j L^k_{max}, \text{ para } j \in V_k,\ k = 1, ..., q \tag{2.17}$$

$$\sum_{k=1}^{q}\sum_{j\in V_k} a_{ij} y_j \leq d_i, \text{ com } i \in N = \bigcup_{k=1}^{q} N_k \tag{2.18}$$

$$\sum_{k=1}^{q}\sum_{j\in V_k} y_j \leq F_1 \tag{2.19}$$

$$\sum_{i\in N_k} a_{ij} \leq F_2, \text{ para } j \in V, k = 1, ..., q \tag{2.20}$$

$$\delta_j \in \{0, 1\} \text{ e } a_{ij}, y_j \geq 0 \text{ e inteiros, com } i \in N = \bigcup_{k=1}^{q} N_k \text{ e } j \in V = \bigcup_{k=1}^{q} V_k \tag{2.21}$$

A modelagem matemática referida a (2.15)-(2.21) apresenta as condições mais realistas ao Problema de Corte de Bobinas de Aço em duas fases. A solução do PMCR por meio de (2.15)-(2.21) vão definir padrões de corte nas bobinas mestres de aço na obtenção de fitas. Observe-se que as formulações (2.15)-(2.18) são as mesmas referidas no PMC irrestrito, sendo assim o PMCR um problema de *mochila não linear*.

Observação: Pelo estudo teórico do problema, no tratamento do PMC neste trabalho de dissertação não vão-se considerar o custo da criação dos compartimentos para cada classe $k = 1, ..., q$.

2.5 Um modelo linear para o problema da Mochila Compartimentada

Em [15] faz uma nova abordagem do PMC visado em um tratamento linear do problema, que em seguida em [6] da continuidade, aprofundando na abordagem e efetuando testes computacionais que favoreceu em fortalezar a conjetura da linearidade do PMC. No trabalho apresentado por [20], aproveitando-se dos avanços anteriores, mostra e prova um modelo linear para o PMC, em efeito para PMCR, declarando a legalidade da conjetura da linearidade como verdadeira, e, formalizando assim, uma nova modelagem para o problema. Nesta seção é feito a apresentação do modelo linear descrito por [20], na qual, dito modelo é o ponto de partida para o estudo de um algoritmo *Branch and Bound* para o PMC que será o assunto a discutir nos Capítulos 4 e 5.

Na abordagem clássica, a formulação do PMC parte de considerar todos os

compartimentos viáveis (bem determinados) e em seguida escolher quais e quantos devem ser construídos (inseridos na mochila). Do anterior, lembre-se de 2.1, onde y_j é a variável de decisão que define a quantidade de compartimentos j em V_k podem ser criados, é certamente a variável que produz a não linearidade do problema. Na abordagem linear, vão-se limitar o número de compartimentos que podem ser criados para cada classe, procurando assim, eliminar o tratamento de V como de y_j do problema, da seguinte forma:

Para cada classe $k = 1, ..., q$, a quantidade de compartimentos que podem ser escolhidos não pode ultrapassar de $\left\lfloor L/L^k_{min} \right\rfloor$[1] e, em caso de um PMCR, de F_1 (incluindo-se repetições implícitas). Define-se $p_k = min\left\{F_1, \left\lfloor L/L^k_{min} \right\rfloor\right\}$ o limitador de compartimentos que podem ser criados para cada classe k. Assim pode-se criar compartimentos repetidos sim ultrapassar p_k, podendo eliminar as variáveis y_j. Observe-se que a criação dos compartimentos não possuem determinação previa nas quantidades de cada item que os compõem, então, serão as variáveis de decisão. Tem-se (teoricamente) a construção de $\sum_{k=1}^{q} p_k$ compartimentos, alguns serão nulos (a quantidade de item que será incluído no compartimento será nula) que não será parte, na prática, na compartimentação da mochila e outros serão compartimentos implícitos, ou seja, com mesma quantidade de cada item.

Observação: para facilitar a formulação e tratamento de informações para o PMC, seguindo o feito em [20], efetua-se uma indexação *local* para os compartimentos em relação ao limitante p_k. Fala-se de uma indexação local no sentido de enumerar por classes os itens e os compartimentos, caso contrário como foi feito no Exemplo 1, onde a indexação foi global. Desta forma, para cada classe $k = 1, ..., q$, faz-se uma enumeração dos itens de 1 até $|N_k|$ e com os compartimentos de 1 até p_k. Outra observação importante é que a forma como a indexação dos itens e compartimentos, global ou local, não prejudica na resolução, já que sem importar a forma como seja organizado cada N_k, os itens continuam ligados a suas respectivas classes.

Em seguida assim, seguindo a observação anterior, vão-se modificar a notação usada no Capítulo 2 sobre a modelagem do PMC (formulado em (2.1)-(2.4)), onde, a variável de decisão a_{ij} será ajustada, tendo em consideração a indexação local, a ser identificado de forma explicita onde o item i pertencem. Então, nota-se a variável de decisão como a^k_{ij}, que vão representar o número de itens i da classe k que está preenchido no compartimento j da mochila. Para maior claridade, apresenta-se um exemplo com o tratamento das indexações locais com efeito da limitação dos compartimentos por classes:

Exemplo 2. Considere 3 classes e 9 itens, sendo 2 itens para a primeira classe, 4 itens para a segunda classe e 3 para a última classe. Então, tem-se, seguindo as notações do problema como: número de classes $q = 3$, o conjunto dos itens $N = \{1^1, 2^1, 1^2, 2^2, 3^2, 4^2, 1^3, 2^3, 3^3\}$ (para este exemplo, entende-se i^k, como o elemento i da classe k, onde, tem-se para cada classe os itens $N_1 = \{1, 2\}$, $N_2 = \{1, 2, 3, 4\}$, $N_3 = \{1, 2, 3\}$. Agora, define-se como os limitantes (capacidade) dos compartimentos como: para a classe 1 como $L^1_{min} = 8\ uc$ e

[1] A expressão $\lfloor z \rfloor$ denota ao inteiro obtido por arrendondamento por abaixo de z.

$L^1_{max} = 10\ uc$ (uc unidades de cumprimento); para a classe 2 como $L^2_{min} = 7\ uc$ e $L^2_{max} = 12\ uc$, e, para a classe 3 como $L^3_{min} = 10\ uc$ e $L^3_{max} = 14\ uc$. Considere-se a capacidade da mochila de $L = 23\ uc$ e $F_1 = 4$ (não foi considerado o F_2 do fato que não tem implicações relevantes no presente exemplo). Com o anterior, tem-se que o limite de compartimentos que se tem por classe e o seguinte: $p_1 = min\{4, \lfloor 23/8 \rfloor\} = 2$, $p_2 = min\{4, \lfloor 23/7 \rfloor\} = 3$ e $p_3 = min\{4, \lfloor 23/10 \rfloor\} = 2$. O anterior indica que no máximo 2 compartimentos da classe N_1, 3 compartimentos da classe N_2 e compartimentos da classe N_3 podem ser inseridos na mochila. Assim, pode-se construir 7 compartimentos (podendo ter alguns compartimentos nulos, dependendo das outras informações como as utilidades e os pesos referidos aos itens de cada classe) onde, os índices dos compartimentos criados com itens referidos a N_1 pertencem a $\{1, 2\}$, para os compartimentos com itens referidos a N_2 pertencem a $\{1, 2, 3\}$ e para os compartimentos com itens referidos N_3 pertencem a $\{1, 2\}$. Na Tabela 2.3 tem-se um resumo da informações do exemplo.

Classe (k)	L^k_{min}	L^k_{max}	p_k	**L**	F_1	F_2
1	8	10	2	23	4	*
2	7	12	3			
3	10	14	2			

Tabela 2.3: Resumo informações do Exemplar 2 com limitação de compartimentos para cada classe. * Não precisa o exemplo desta informação.

Para o modelo linear foi introduzido uma variável binaria de controle de criação de compartimentos δ^k_j, onde, para todo $k = 1, ..., q$ e $j = 1, .., p_k$, tem-se que δ^k_j obtém o valor de 1 quando o compartimento de índice j referente à classe k é não nulo (em outras palavras, na solução do problema, é um compartimento que é viável e pode-se construir), e o valor de 0 no caso quando é nulo (o compartimento não é criado).

Finalmente, com a limitação dos compartimentos e eliminação da variável y_j da formulação clássica, junto a uma indexação local dos itens e compartimentos, além da nova notação para a variável de decisão do problema, apresenta-se o modelo linear para o PMCR feito em [20] como:

$$\text{Maximizar: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} u^k_i a^k_{ij} \tag{2.22}$$

$$\text{sujeito a: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} l^k_i a^k_{ij} \leq L \tag{2.23}$$

$$\delta^k_j L^k_{min} \leq \sum_{i \in N_k} l^k_i a^k_{ij} \leq \delta^k_j L^k_{max} \text{ com } j = 1, ..., p_k, k = 1, ..., q \tag{2.24}$$

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k, i \in N_k, k = 1, ..q \tag{2.25}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta_j^k \leq F_1 \tag{2.26}$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2, j = 1, ..., p_k, k = 1, ..., q \tag{2.27}$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ e inteiros, } i \in N_k, j = 1, ..., p_k, k = 1, ..., q \tag{2.28}$$

Da formulação feita em (2.22)-(2.28) tem-se em consideração o seguinte: a função objetivo apresentada em (2.22) representa a soma das utilidades de todos os itens escolhidos em cada compartimentos. A restrição (2.23) garante que a soma das larguras dos itens escolhidos é limitada pela capacidade da mochila. Observe-se que a variável a_{ij}^k serão definidas apenas se o item i e o compartimento j são referentes à mesma classe N_k (com isto não é necessário incluir o conjunto V e os subconjuntos V_k para o modelo linear). A restrição (2.24) limita a capacidade dos compartimentos não nulos e anula os coeficientes dos compartimentos que não serão utilizados com a variável de controle δ_j^k ($\delta_j^k = 1$ se o compartimento j da classe k é não nulo e $\delta_j^k = 0$ no caso contrário). A restrição (2.25) limita a quantidade de cada item $i \in N$ por sua respectiva demanda. A restrição (2.26) limita a quantidade de compartimentos não nulos que podem ser inseridos na mochila e em (2.27) a quantidade de itens que podem ser inseridos em cada compartimento. Na restrição (2.28) estabelece o dominio das variáveis.

2.6 Abordagens de Solução para o PMC

Nesta seção vão-se apresentar algumas abordagens de solução para o PMCR usando como estratégias heurísticas e um método de solução exata por meio de decomposição exaustiva para o PMCR.

2.6.1 Algumas Heurísticas

Inicia-se a apresentação das heurísticas das z Melhores Compartimentos e as w Capacidades do trabalho de [28], na qual, esta última será amplamente usada no Capítulo 5.

Para formular o modelo original do PMC (2.15)-(2.21) foi usado a ideia de considerar todos os compartimentos viáveis que podem ser construídos na mochila, para em seguida ser escolhidos os compartimentos que deverão fazer parte da compartimentação. Obter todos os compartimentos é o caso ideal para resolver de forma exata o problema, mas, como já foi discutido, fazer este processo para exemplares grandes é um processo computacional inviável. Então, se em lugar de considerar todos os compartimentos só se faz consideração de uma parcela, tem-se assim uma forma de relaxar o PMCR. Observe que para o modelo (2.15)-(2.21) um compartimento é viável (controlado por a_{ij}) se é satisfeito, independentemente de y_j, para todo $j \in V_k$, $k = 1, ..., q$ as seguintes restrições:

$$\sum_{i \in N_k} l_i a_{ij} \leq L \tag{2.29}$$

$$L^k_{min} \leq \sum_{i \in N_k} l_i a_{ij} \leq L^k_{max} \text{ ou } a_{ij} = 0, \text{ para todo } i \in N \tag{2.30}$$

$$a_{ij} \leq d_i, \text{ para todo } i \in N \tag{2.31}$$

$$\sum_{i \in N_k} a_{ij} \leq F_2 \tag{2.32}$$

$$a_{ij} \geq 0 \text{ e inteiro}, i \in N \tag{2.33}$$

Agora observe o seguinte: como $L \geq L^k_{max}$ para todo $k = 1, ..., q$ a restrição (2.29) é satisfeita se é assumido as condições (2.30)-(2.33). Observe também que se $a_{ij} = 0$ para todo $i \in N$, nenhum item será somado na função objetivo e por tanto, não é necessário a criação de compartimentos.

Então, com $L^k_{min} \leq \sum_{i \in N_k} l_i a_{ij} \leq L^k_{max}$ e as restrições (2.31)-(2.33) definem todos os compartimentos viáveis. Com todos os compartimentos viáveis definidos, ou seja, com todos os valores de a_{ij} estabelecidos, pode-se ver cada classe como se fosse só um grande item, onde pode-se definir sua utilidade e largura da seguinte forma:

$$U_j = \sum_{i \in N_k} u_i a_{ij} \tag{2.34}$$

$$L_j = \sum_{i \in N_k} l_i a_{ij} \tag{2.35}$$

Em seguida, com a definição da utilidade geral de cada compartimento como (2.34) da largura geral de cada compartimento como (2.35), o modelo clássico do PMCR (2.15)-(2.21) pode ser re-escrito como:

$$\text{Maximizar: } \sum_{j \in V_1} U_j y_j + \cdots + \sum_{j \in V_q} U_j y_j \tag{2.36}$$

$$\text{sujeito a: } \sum_{j \in V_1} L_j y_j + \cdots + \sum_{j \in V_q} L_j y_j \leq L \tag{2.37}$$

$$\sum_{k=1}^{q} \sum_{j \in V_k} a_{ij} y_j \leq d_i, i \in N \tag{2.38}$$

$$\sum_{k=1}^{q} \sum_{j \in V_k} y_j \leq F_1 \tag{2.39}$$

$$y_j \geq 0 \text{ e inteiro}, j \in V \tag{2.40}$$

O modelo (2.36)-(2.40) será chamado de *Problema Mestre* para o PMCR. Note que as restrições (2.17)-(2.20) são satisfeitas em (2.30)-(2.33), por tanto não se tem restrições análogas em (2.36)-(2.40).

Com o Problema Mestre definido, procura-se estratégias para determinar compartimentos convenientes que serão usados para alimentar o modelo modelo (2.36)-(2.40), daqui surgem heurísticas para o PMCR chamadas de *heurísticas de decomposição*. As heurísticas de decomposição consistem em criar alguns compartimentos viáveis para o PMCR para em seguida ser usados no Problema Mestre (2.36)-(2.40) e obter uma solução aproximada para o PMCR.

Como ponto de partida para formular as heurísticas é estabelecer como calcular o melhor compartimento de cada classe. Então, usa-se o seguinte problema de Mochila Restrito para gerar o "melhor" compartimento $j \in V_k$ de capacidade máxima $L_{cap} \leq L^k_{max}$ para cada classe $k = 1, ..., q$:

$$\text{Maximizar: } \sum_{i \in N_k} u_i a_{ij} \tag{2.41}$$

$$\text{sujeito a: } L^k_{min} \leq \sum_{i \in N_k} u_i a_{ij} \leq L_{cap} \tag{2.42}$$

$$a_{ij} \leq d_i, \, i \in N \tag{2.43}$$

$$\sum_{i \in N_k} a_{ij} \leq F_2 \tag{2.44}$$

$$a_{ij} \geq 0 \text{ e inteiro}, \, i \in N \tag{2.45}$$

Agora, define-se as heurísticas de decomposição apresentadas em [28]: a *Heurística da Decomposição* consiste em, para cada classe de itens k, gerar o "melhor compartimento", onde é usado do PMR descrito acima (2.41)-(2.45) e em seguida resolvendo o Problema Mestre (2.36)-(2.40). A *Heurística dos "z Melhores Compartimentos"*, onde de forma análoga à heurística anterior, para cada classe k dos itens é gerado os "z melhores compartimentos", utilizando o PMR formulado em (2.41)-(2.45) e em seguida resolvendo o Problema Mestre (2.36)-(2.40). Pode-se ver esta heurística no Algoritmo 1. Observe-se que se é definido $z = 1$ na Heurística dos "z Melhores Compartimentos', esta vão corresponder com a Heurística da Decomposição.

A *Heurística do Melhor Compartimento* busca eliminar a resolução por meio do Problema mestre, da seguinte forma: cria-se o melhor compartimento de cada classe $k + 1, ..., q$ usando o problema de mochila restrito (2.41)-(2.45). Em seguida usando algum critério de seleção, por exemplo usando a eficiência associada com os grandes U_j e L_j para cada classe, inicia-se o preenchimento da mochila com o melhor dos compartimentos criados. Com este primeiro compartimento, atualiza-se a demanda disponível e proceda-se a escolher pelo critério fixado o seguinte melhor compartimento e preencha-se a Mochila. O processo continua iterativamente

até que o padrão de corte esteja definido. A *Heurística das "w Capacidades"* (Algoritmo 2) permite o cálculo para cada classe k o melhor compartimento para w diferentes capacidades, ou seja, varia-se a largura máxima L_{cap} para obter compartimentos de larguras diferentes, obtendo assim $q \times w$ compartimentos que irão compor a compartimentação da mochila. Observe-se para o caso de $w = 1$, tem-se de novo a Heurística da Decomposição.

Pode-se encontrar outras heurísticas para abordar na solução do PMCR, como o caso de heurísticas de retro-alimentação (uma estrutura similar á Heurística do Melhor Compartimento), recomenda-se [15], [6], uma heurística de decomposição onde é feita ordenação de classes com geração de mais compartimentos nas melhores classes chamado de The Decreasing Classes heuristic em [19] e uma heurística que usa p_k para criar os compartimentos usando teoria de fortalecimento, *The Heuristic of p_k Strong Capacities*, em [34].

Algoritmo 1: Heurística dos z Melhores Compartimentos

Entrada: N, u_i, l_i, d_i, L, L^k_{max}, L^k_{min}, z
Saída: a_{ij}, y_j
Inicialização: $V = \varnothing$, $comp = 1$
para todo $k = 1, ..., q$ **faça**
  $L_{cap} = L^k_{max}$; Calcule-se as z melhores soluções do (2.41)-(2.45);
  para todo $contador = 1, ..., z$ **faça**
    $j = comp + contador - 1$; $j \in V_k$;
    Salve a_{ij}, U_j e L_j de acordo com a $contador - $ *ésima* melhor solução de (2.41)-(2.44); $comp = comp + z$
  fim
  Resolva (2.36)-(2.40)
fim

Fonte: [28]

Algoritmo 2: Heurística das w Capacidades

Entrada: N, u_i, l_i, d_i, L, L^k_{max}, L^k_{min}, w
Saída: a_{ij}, y_j
Inicialização: $V = \varnothing$, $comp = 1$
para todo $k = 1, ..., q$ **faça**
  $L_{cap} = L^k_{max}$;
  para todo $j = comp, ..., comp + w - 1$ **faça**
    Resolva (2.41)-(2.44); $j \in V_k$;
    Salve a_{ij}, U_j e L_j ; $L_{cap} = L_j - 1$; $comp = comp + w$
  fim
  Resolva (2.36)-(2.40)
fim

Fonte: [28]

2.6.2 Algoritmo de Decomposição Exaustiva

Nesta subseção vão-se expor uma maneira de resolver de forma exata o PMCR por meio da geração de todos os compartimentos viáveis do problema chamado de Algoritmo de decomposição Exaustiva apresentado por [20]. Estes compartimentos são criados usando um algoritmo de enumeração sistemática dos compartimentos viáveis chamado Gerador de Compartimentos, onde usa um critério similar ao método de *Branching and Bound* (este método será estudado no Capítulo 3), onde para cada classe $k = 1, ..., q$ do PMCR é gerada uma "árvore" que enumera todos compartimentos viáveis para assim formar o conjunto V.

A busca dos compartimentos é feita a profundidade, conforme vão-se modificando os valores das variáveis de decisão a_{ij}, $i \in N_k$, onde $a_j = (a_{ij})_{i \in N_k}$ criará um compartimento se satisfaz (2.42)-(2.45).

Para criar á árvore, precisa-se fixar certos valores de referencia onde possa-se efetuar a busca, ou seja, definir os nós da árvore. Neste caso, os valores de referencia para cada nó vai corresponder aos itens com índices indexados em N. O autor recomenda realizar a indexação dos itens por meio de uma ordenação das variáveis por ordem decrescente das larguras dos itens, dado que assim pode-se ter um melhor rendimento do algoritmo [20].

A geração de cada ramo (compartimento que pode pertencer a V_k), precisa-se considerar um limitante superior para a capacidade de um compartimento, a demanda do item e a quantidade máxima de itens que é aceito no compartimento. Então, para criar um compartimento V_k, o máximo valor que pode-se atribuir a a_{ij} é:

$$\hat{a_{ij}} = min \left\{ \left\lfloor \left(L^k_{max} - \sum_{n \in N_k, n \neq i} l_i \hat{a}_{nj} \right) / l_i \right\rfloor , d_i, \bar{F} \right\} \tag{2.46}$$

Observe que aparece a expressão $\bar{F}$, que é a quantidade de itens que ainda podem serem inseridos no compartimento, dada por $\bar{F} = F_2 - \sum_{n \in N_k, n \neq i} l_i \hat{a}_{nj}$. O ramo desenvolvido deve ter uma capacidade (a soma das capacidades dos compartimentos criados) maior ou igual do que L^k_{min} para que seja viável. Ao garantir o anterior, a solução obtida pelo ramo desenvolvido é salvada, ou seja, é criado o compartimento. Caso contrário não é salvada a solução. Em seguida, continua-se a construção da árvore. A partir do último nó volte-se até o nó que seja $\hat{a}_i \neq 0$, diminua-se em uma unidade o valor de dito nó e faz de novo a construção de um novo ramo e efetua-se novamente o teste de factibilidade. Repita-se até que a solução seja infactível ou as possibilidade para a última variável estejam esgotados. Quando o ramo produz um compartimento infactível, faz-se a volta nos nós. A continuação mostra-se o algoritmo descrito como *Algoritmo Gerador de Compartimentos*.

Após a geração de todos os compartimentos viáveis, o Algoritmo da Decomposição Exaustiva consiste na resolução do Problema Mestre (2.36)-(2.40) e assim obter a solução exata do PMCR. Este tipo de solução só é viável, computacionalmente, quando os exemplares

a implementar são pequenos, do fato que apresenta tempos práticos não muito elevados para desenvolver estudos comparativos com outras formas de solução ao problema (como o caso do Modelo Linear já apresentado). No Capítulo 6 pode-se ver o comportamento do algoritmo para quantidades fixadas de itens e de classes em exemplares relacionados com problemas de corte de bobinas de aço.

Algoritmo 3: Gerador de Compartimentos

Entrada: N_k, L^k_{min}, L^k_{max}, l_i, d_i, F_2
Saída: V, a_{ij}
Inicialização: faça $j = 1$
para todo $k = 1, ..., q$ **faça**
  Ordene os itens em ordem decrescente de largura. Se $\mid N_k \mid = m$, então, considere os índices dos itens variando entre 1 e m, com $\hat{l}_i \geq \hat{l}_{i+1}$, $i = 1, ..., m-1$; $r = 0$; $\bar{F} = F_2$
fim
Passo 1 *(desenvolvimento de um ramo)*:
para todo $i = r+1, .., m$ **faça**
  $\hat{a}_i = min\left\{\left\lfloor\left(L^k_{max} - \sum_{i=1}^{j-1} l_i\hat{x}_i\right)/l_j\right\rfloor, \hat{d}_i, \bar{F}\right\}$; $\bar{F} = \bar{F} - \hat{a}_{ij}$; $r = m$; **Passo 2**
fim
Passo 2 *(salvar solução corrente)*:
se $\sum_{n\in N_k} \hat{l}_n\hat{a}_n \geq L^k_{min}$ **então**
  salve o compartimento ($j \in V_k$ e $a_{ij} = \hat{a}_i$, onde o item de índice i corresponde ao índice i' na ordenação); $j = j + 1$;
senão
  Passo 4
fim
Passo 3 *(ramifica)*:
se $\hat{a}_r > 0$ **então**
  $\hat{a}_r = \hat{a}_r - 1$; $\bar{F} = \bar{F} + 1$;
  se $r = m$ **então**
    Passo 2
  senão
    Passo 1
  fim
  Passo 4
fim
Passo 4 *(Critério de Poda)*:
se $r = 1$ **então**
  pare!
senão
  $r = r - 1$; $\bar{F} = \bar{F} + \hat{a}_r$;
  Passo 3
fim

Fonte: (Inarejos 2015, [20])

3 O MÉTODO ESPECIALIZADO *BRANCH AND BOUND*

Neste capítulo é feito o estudo da técnica de enumeração implícita de soluções factíveis para Problemas de Programação Linear chamado *Branch and Bound*. Pretende-se introduzir ao leitor na técnica junto à dinâmica teórica para o desenvolvimento de uma. Esta técnica vai-se utilizar para formular um algoritmo que permita a resolução exata ao Problema da Mochila Compartimentada, onde é estudada no Capítulo 5.

3.1 PRINCIPIO DO *Branch and Bound*

É possível que o trabalho que deu origem á pesquisa de métodos *Branch and Bound* foi em [21] para a solução de diversos problemas de Programação Linear (PPL), seguindo os comentários de [36]. O termo certamente foi adjudicado em [24], trabalho onde foi estudado formas de solucionar o problema do Caixero Viajante, tema que é amplamente estudado em [2]. Neste trabalho pode-se encontrar em alguns parágrafos o algoritmo de *Branch and Bound* abreviado por B&B para comodidade do leitor.

A técnica do B&B esta baseada na ideia de desenvolver uma enumeração sistemática das soluções candidatas (soluções viáveis) na região viável de um PPL na busca da solução ótima do problema. Em um primeiro momento o algoritmo faz a partição da região viável do PPL onde ocorreram as explorações. Para cada partição gera-se sub-problemas de tal forma que possibilite a exploração sistemática do algoritmo encontrando soluções (candidatas) que resolvem o problema e outras que não o favorecem (usualmente com soluções não viáveis), assim, ao momento de terminar todas as explorações tem-se entre as soluções candidatas a solução ótima à PPL. Este tipo de procedimento tem a capacidade de resolver de forma exata PPL mas com custo computacional usualmente alto (ou seja, com tempos execução muito altos). Considere, por exemplo, uma enumeração exaustiva para PPL's que contém uma grande quantidade de variáveis de decisão, então, qualquer tentativa computacional para sua exploração torna-se inviável. Para evitar um processo computacional custoso, procura-se um critério de decisão que avalie se certa partição da região viável pode serem uma candidata a ter a solução ótima ao problema, e assim, a busca da solução se faz de forma estratégica e mais rápida.

Com a busca exaustiva de soluções viáveis do problema com a procura feita anteriormente, estas soluções pode-se representar por meio de um ramo de uma árvore de soluções. Estes ramos são chamados de *Branch* de nós (onde os nós são os subproblemas gerados). Em seguida, a técnica do B&B consiste em definir um algoritmo que crie uma árvore de enumeração (enumeração de soluções candidatas) e segundo um critério que defina a conveniência da exploração de certas partições da região viável do PPL.

3.2 CONCEITOS BÁSICOS

Os conceitos apresentados nesta seção foram seguindo os trabalhos de [36], [2] e [12]. Consideremos um problema P estabelecido da seguinte forma:

$$\begin{aligned} &\text{(P) Maximizar: } f(x) \\ &\text{s.a.: } x \in X \end{aligned} \tag{3.1}$$

Onde f é a função objetivo do problema P com região viável X discreto. Com base a (3.1) tem-se a seguinte série de definições:

Definição 3.1 (Problema Relaxado)**.** Seja o problema RP como segue:

$$\begin{aligned} &\text{(RP) Maximizar: } g(x) \\ &\text{s.a.: } x \in Y \end{aligned} \tag{3.2}$$

Onde g é a função objetivo do problema RP com região viável Y discreto. Definamos RP como *Problema Relaxado* do P aquele que satisfaz os seguintes critérios:

- $X \subseteq Y$ e
- $x \in X$ tem-se que $g(x) \geq f(x)$

Definição 3.2 (Branching)**.** O processo de *Branching* é definido como um processo de criação de sub-problemas (relaxados ou restritos segundo o problema) a partir de P, e em seguida criação de novos sub-problemas a partir dos anteriores definidos (dependendo do critério de busca). No caso para Problemas de Programação Inteira (PPI) define-se o *Branching* como segue: define-se P_0 com região viável X_0 como problema relaxado de P. Em seguida, com relação a P_0 gera-se sub-problemas restritos $P_1, P_2, ..., P_n$ onde para cada um deles está definido uma partição da região viável X_0. Novamente a partir de cada P_i, $i \in \{1, 2, ..., n\}$ gera-se novos sub-problemas que conterão partições de cada subconjunto obtido anteriormente até agotar as possibilidades de separação. No processo de *Branching* para PPI tem-se as seguintes observações:

- No desenvolvimento do algoritmo não se elimina soluções viáveis que sejam candidatas de serem soluções ótimas para o problema P.
- Cada solução dos novos P_i deverá nos aproximar á solução do problema P.
- O número de Sub-problemas criados deve ser polinomial (com respeito a alguma medida).

- Os sub-domínios dos Sub-problemas criados devem, em princípio, ser mutuamente exclusivos.

Por meio da fixação das variáveis que compõe o problema P (variáveis de decisão) pode-se definir outra forma de particionar a região viável do P. Considere a variável de decisão $x_i, i \in \{1, 2, ...\}$ do P cujos valores possíveis são $1, ..., n$ (ou seja, o número de itens i que podem serem usados). Pode-se partir P em n subproblemas, $P_1, P_2, ..., P_n$, tal que $\bigcup_{j=1}^{n} P_j = P$. P_j corresponde o j-ésimo valor possível da variável fixada x_i.

Definição 3.3 (Nó da árvore de enumeração)**.** Cada sub-problema restrito gerado a partir de um problema relaxado de um PPI é nomeado como nó da árvore de enumeração. Caso da fixação das variáveis de decisão do problema P, os nós vai corresponder a ditas variáveis.

Definição 3.4 (Bounding)**.** Chama-se de *Bounding* o processo de avaliação para cada sub-problema gerado do P na contribuição que pode fornecer na solução ótima do problema. Quando um sub-problema gerado não contribui em favorecer na solução ótima ao problema, o processo de *Brenching* nesse nó é interrompido. Fala-se que dito nó é podado ou foi deito um teste de sondagem (TS). Mais informação em [1].

Em um PPI, o *Branching* é interrompido quando uma das seguintes condições é satisfeita.

- TS1 ou poda por infactibilidade: o sub-problema restrito criado é infactível.
- TS2 ou poda por optimalidade: a solução do problema relaxado é inteira.
- TS3 ou poda por qualidade: O valor de qualquer solução factível do problema relaxado é pior que o valor da melhor solução factível atual. Para dos no nó n_i.

Quando é feito a fixação de variáveis, a avaliação ocorre por meio do cálculo de Limitantes Superiores, processo que será descrito nas seções seguintes.

Definição 3.5 (árvore de enumeração B&B para o problema P)**.** O processo de *Braching* (separação) e *Bounding* (avaliação) pode-se observar como uma árvore B&B para o problema P, onde P_0 é estabelecido como raiz da árvore como o nó n_0. Os seguintes nós representam cada P_i gerado sucessivamente a partir de P_0, como se pode ver na Figura 3.1 (representado no caso da geração de sub-problemas restritos).

No caso da fixação recursiva das variáveis de decisão de P, nós internos representaram todas a soluções que podem serem obtidas. Cada *branch* ou *folia* representaram uma solução viável do problema.

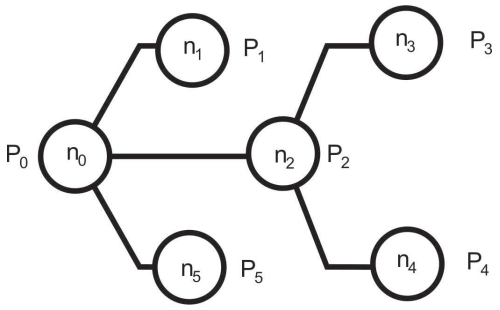

Figura 3.1: Árvore B&B para um problema P com geração de sub-problemas restritos.
Fonte:Próprio autor.

Definição 3.6 (Ramificação da árvore)**.** Para um nó n_j, onde a partir dele é gerado dois o mais novos nós em função do *branching* (separação) do problema P_j, fala-se que o nó n_j foi *ramificado* (Figura 3.2).

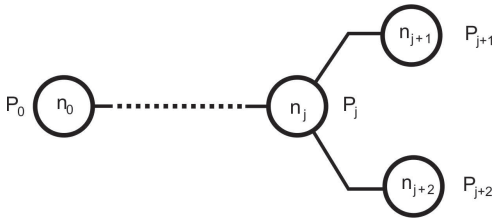

Figura 3.2: Ramificação de um nó n_j
Fonte: Próprio autor.

Definição 3.7 (Nós abertos e fechados)**.** Chama-se **nós abertos** aqueles nós que ainda não foram ramificados. Estes corresponderam as *folhas* da árvore dos problemas que não foram sometidos a um processo de *branching*. Os outros nós serem chamados de **nós fechados**.

3.3 O ALGORITMO BRANCH AND BOUND EM UM PROBLEMA PPI

Apresenta-se a continuação um exemplo de um Problema de Programação Inteira onde é criado um algoritmo *B&B* para sua resolução, onde é relaxado o problema e em seguida são gerados de forma sucessiva sub-problemas restritos. Utiliza-se este exemplo para comparar o tratamento dos problemas no caso de limitar os nós que terá a árvore por meio da fixação das variáveis de decisão, como é caso dos Problemas de Mochila, em especial o Problema de Mochila Compartimentada.

Exemplo 3 (*Branch and Bound* num problema PPI)**.** Seja o seguinte Problema de Programação Inteira:

$$\begin{aligned} \text{Maximizar: } & 21x_1 + 11x_2 \\ \text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\ & x_1, x_2 \geq 0 \text{ e inteiros} \end{aligned}$$

Inicia-se com a relaxação do PPI, que vai configurar o nó raiz da árvore (n_0). No PPI desconsidere a restrição de ser inteiros das variáveis x_1 e x_2 e define-se o problema P_0 como segue:

$$\begin{aligned} (P_0)\ \text{Maximizar: } & 21x_1 + 11x_2 \\ \text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\ & x_1, x_2 \geq 0 \end{aligned}$$

A solução do P_0 é $x_1 = \frac{39}{21} \approx 1.86$ e $x_2 = 0$ com valor da função objetivo de $z_0 = 39$. Em seguida, procede-se a avaliar as soluções ao problema (*Bounding*). Como o valor de x_1 não é inteiro, pode-se iniciar o *Branching* ao ramificar o nó n_0, onde vai-se considerar as regiões com valor de $x_1 \geq 2$ e $x_1 \leq 1$ para o problema P_0, definindo os sub-problemas restritos P_1 e P_2 como segue:

$$\begin{aligned} (P_1)\ \text{Maximizar: } & 21x_1 + 11x_2 \\ \text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\ & x_1 \geq 2 \\ & x_1, x_2 \geq 0 \end{aligned} \qquad \begin{aligned} (P_2)\ \text{Maximizar: } & 21x_1 + 11x_2 \\ \text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\ & x_1 \leq 1 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Para o problema P_1 se aplica o critério de poda TS1 no fato que não se tem soluções factíveis. No caso de P_2 tem-se como solução solução corrente $z_2 = 37.5$ com valores de $x_1 = 1$ e $x_2 = 1.5$. Como o valor de x_2 não é inteiro, se ramifica o nó n_2, com os problemas P_3 para quando o valor de $x_2 \leq 1$ e $x_2 \geq 2$:

$$\begin{aligned} (P_3)\ \text{Maximizar: } & 21x_1 + 11x_2 \\ \text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\ & x_2 \leq 1;\ x_1 \leq 1 \\ & x_1, x_2 \geq 0 \end{aligned} \qquad \begin{aligned} (P_4)\ \text{Maximizar: } & 21x_1 + 11x_2 \\ \text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\ & x_2 \geq 2;\ x_1 \leq 1 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Tem-se para P_3 como solução corrente $z_3 = 32$ com valores para as variáveis $x_1 = 1$ e $x_1 = 1$ onde se aplica o critério de poda TS2 onde é primer candidato a serem solução ótima do PPI. Para P_4 tem-se $z_4 = 37$ com $x_1 = 0.71$ e $x_2 = 2$. Faz-se una nova ramificação ao nó n_4 com a criação dos subproblemas P_5 para $x_1 \leq 0$ e P_6 com $x_1 \geq 1$:

$$
\begin{aligned}
(P_5)\ \text{Maximizar: } & 21x_1 + 11x_2 \\
\text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\
& x_2 \geq 2 \\
& x_1 \leq 1 \\
& x_1 \leq 0 \\
& x_1, x_2 \geq 0
\end{aligned}
\qquad
\begin{aligned}
(P_6)\ \text{Maximizar: } & 21x_1 + 11x_2 \\
\text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\
& x_2 \geq 2 \\
& x_1 \leq 1 \\
& x_1 \geq 1 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

Observe que o problema P_6 é não factível então aplique-se o criterio de poda TS1 (fecha-se o nó n_6). O problema P_5 tem-se $z_5 = 35.75$ com $x_1 = 0$ e $x_2 = 3.25$ então, ramifica-se o nó n_5 nos problemas P_7 para $x_2 \leq 3$ e P_8 com $x_2 \geq 4$:

$$
\begin{aligned}
(P_7)\ \text{Maximizar: } & 21x_1 + 11x_2 \\
\text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\
& x_2 \geq 2 \\
& x_1 \leq 1 \\
& x_1 \leq 0 \\
& x_2 \leq 3 \\
& x_1, x_2 \geq 0
\end{aligned}
\qquad
\begin{aligned}
(P_8)\ \text{Maximizar: } & 21x_1 + 11x_2 \\
\text{s.a.: } & 7x_1 + 4x_2 \leq 13 \\
& x_2 \geq 2 \\
& x_1 \leq 1 \\
& x_1 \leq 0 \\
& x_2 \geq 4 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

Tem-se para o problema P_7 a solução corrente $z_7 = 33$ com $x_1 = 0$ e $x_2 = 3$ em seguida aplique-se o critério de poda TS2 (feche-se o nó n_7. O problema $P8$ não tem solução factível, então, aplique-se o critério de poda TS1 (feche-se o nó n_8).

O *B&B* terminou a exploração dado que todos os nós estão fechados na iteração 8 do algoritmo. Observe-se que se tem dois candidatos para solução ótima ao PPI. Tem-se que $z_3 < z_7$, então, a solução ótima para o PPI corresponde aos valores encontrados na iteração 7, ou seja, no nó n_7 com $z_7 = 33$, $x_1 = 0$ e $x_2 = 3$. A exploração feita pelo *B&B* pode-se ver na Figura 3.3.

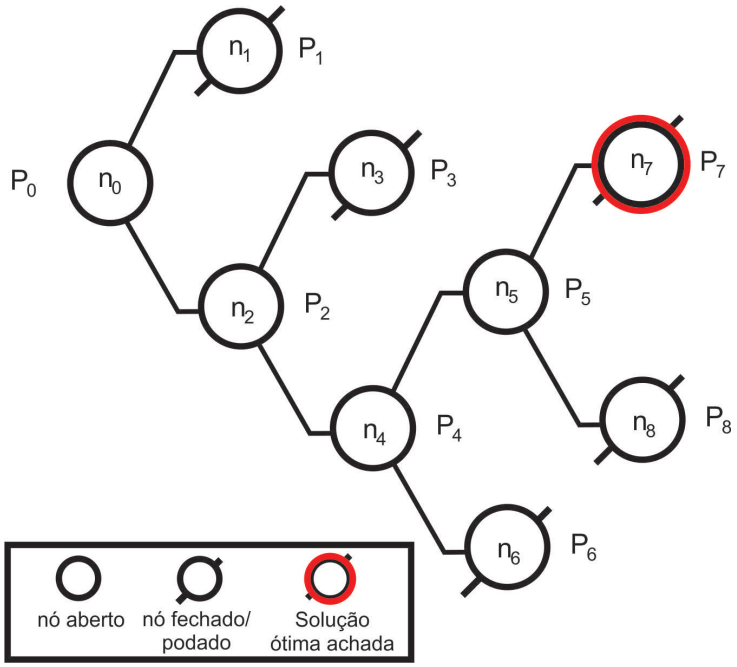

Figura 3.3: BB num problema PMI
Fonte: Próprio autor.

3.4 Branch and Bound para o problema da Mochila Irrestrito

Nesta subseção será descrito um algoritmo *B&B* para resolver o problema da Mochila Restrito apresentado no Capítulo 1. Para este problema vai-se trabalhar a estratégia de fixação das variáveis de decisão. Em [5] define um algoritmo para o problema chamado de *Backtracking* onde for usados as regras *a-priori* e *adaptativas* descritas na seção anterior.

Para a apresentação do algoritmo precisa-se das seguintes definições e sirva-se de um exemplo extraído de (CHVÁTAL, 1983, p. 201):

Proposição 3.8. *Seja* a^* *uma solução ótima para* (1.8)-(1.10)*, então,* $L - \sum_{i=1}^{n} l_i a_i^* < l_r$ *para qualquer* $r \in \{1, ..., n\}$.

Demonstração. Seja a^* uma solução ótima para (1.8)-(1.10) onde se tem $\sum_{1}^{n} l_i a_i^* < L$, assim pode-se acrescentar em uma unidade qualquer $a_r, r \in \{1, ..., n\}$ então, $l_1 a_1^* + l_2 a_2^* + ... + l_r(a_r^* + 1) + ... + l_n a_n^* > L$ logo $L - \sum_{1}^{n} l_i a_i^* < l_r$ para qualquer $r \in \{1, ..., n\}$. □

Definição 3.9 (Soluções sensíveis)**.** Uma solução x é denominada *sensivel* se $L - \sum_{1}^{n} l_i x_i < l_r$ para qualquer $r \in \{1, ..., n\}$.

Definição 3.10 (Eficiência da variável x_j)**.** Define-se como *eficiência* da variável x_j que corresponde ao item j do (1.8), com utilidade u_j e peso l_j à expressão $\frac{u_j}{l_j}$.

Exemplo 4.

$$\text{Maximizar: } 4x_1 + 5x_2 + 5x_3 + 2x_4 \tag{3.3}$$

$$\text{sujeito a: } 33x_1 + 49x_2 + 51x_3 + 22x_4 \leq 120 \tag{3.4}$$

$$x_1, x_2, x_3, x_4 \geq 0 \text{ e inteiro} \tag{3.5}$$

O algoritmo *B&B Backtracking* tem com procedimentos fundamentais, primeiro a ordenação das variáveis x_j por ordem decrescente de suas eficiências ($\frac{u_1}{l_1} > \frac{u_2}{l_2} > ... > \frac{u_j}{l_j} > ... \frac{u_n}{l_n}$) e a enumeração de todas as soluções sensíveis.

Observe que no exemplo já se tem dita ordenação, onde $\frac{4}{33} > \frac{5}{49} > \frac{5}{51} > \frac{2}{22}$. Na Figura 3.4 pode-se verificar que o problema (3.3) tem 13 soluções sensíveis, onde faz novamente um diagrama de árvore. Desde a raiz da árvore, define-se um nó por cada variável x_j do problema, onde vão-ser fixados em direção de esquerda a direita (direção de crescimento de cada folha da árvore). Cada ramo é desenvolvido na ordem que estão de acima para baixo. Para o primeiro ramo, calcule-se x_1 como $\lfloor L/l_1 \rfloor$ [1], ou seja, a máxima quantidade de itens x_1 que podem serem inseridos na mochila. Em seguida, segundo ao espaço disponível na mochila, calcula-se a quantidade de itens x_2 que podem serem adicionados por meio de $x_j = \left\lfloor \left(L - \sum_{i=1}^{j-1} l_i x_i \right) / l_j \right\rfloor$, para todo $j \in \{1, ..., n\}$. O anterior corresponde ao processo de *Branching* do Problema. Para o exemplo tem-se para $x_1 = \lfloor 120/33 \rfloor = 3$, $x_2 = x_3 = x_4 = 0$.

Com o primeiro ramo desenvolvido, salva-se a solução obtida no momento (solução corrente), fazendo a suposição que os anteriores x_j são a solução ótima do problema. A continuação inicia-se o processo de busca. Faça-se a volta (*Bactrack*) do ramo a partir do último nó em direção à raiz até o nó n_k, tal que $x_k \neq 0$.

Em seguida, faz-se $x_k := x_k - 1$ e desenvolva o no ramo com o *Branching* descrito. No exemplo tem-se $k = 1$ e $x_1 := 3 - 1 = 2$, com $x_2 = 1$ e $x_3 = x_4 = 0$. O algoritmo novamente faz o *Backtrack* e *Branching* para criar um novo ramo. O processo de busca a profundidade termina quando $k = 1$ e $x_1 = 0$. Cada novo ramo, salva-se a solução sensível que melhora a função objetivo.

O processo de geração da árvore pode ter tem um custo computacional alto se o problema contem muitas variáveis e com uma capacidade grande da mochila. Para um problema com n itens, no pior dos casos, o algoritmo deve verificar qual conjunto das partes dos itens fornece a maior melhora da função objetivo, procurando-se desenvolver $2^n - 2$ ramos. Resolução exaustiva de este tipo para problemas com n muito grande é praticamente inviável.

[1] A expressão $\lfloor z \rfloor$ denota ao inteiro obtido por arrendondamento por abaixo de z.

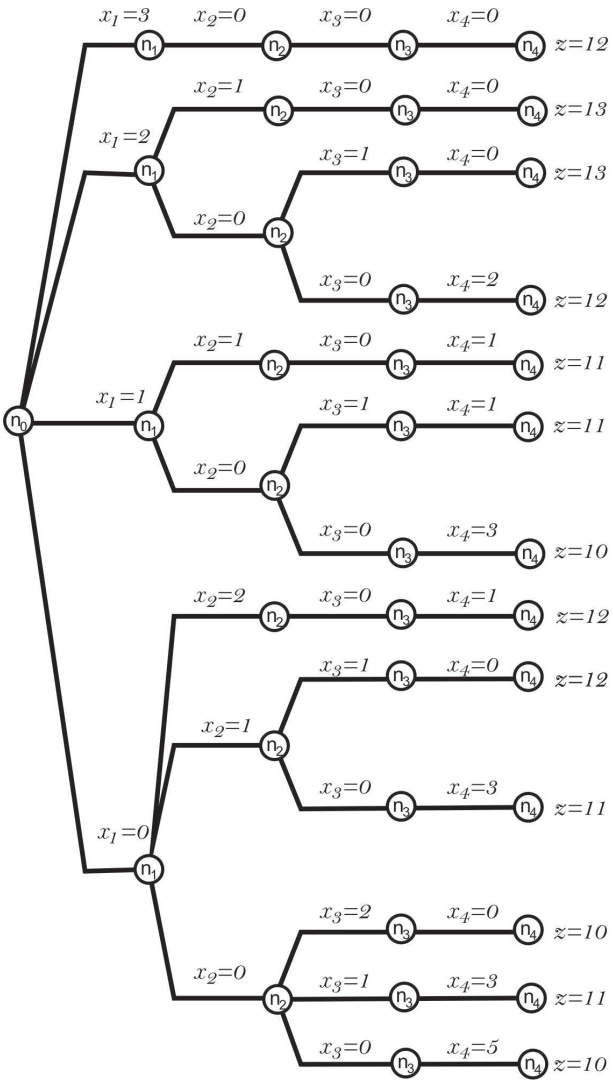

Figura 3.4: Árvore para o exemplo da Mochila Irrestrito sem Podas
Fonte: Próprio autor.

O objetivo seguinte do algoritmo é estabelecer os critérios para Testes de Sondagem ou Podas onde não seja necessário uma busca exaustiva na árvore toda, assim podando aqueles que não apresentem a esperança de serem candidatos a solução ótima ao problema.

Para estabelecer dito critério é preciso estimar o valor que um ramo pode alcançar na função objetivo. Considere um ramo arbitrário $\hat{x}_1, \hat{x}_2, ..., \hat{x}_k, ..., \hat{x}_n$ com $1 \leq k \leq n-1$ é a melhor solução encontrada até o momento ($\hat{z}$). Veja-se a Figura 3.5.

Define-se para $\hat{x}_k := \hat{x}_k - 1$ e faz-se uma nova ramificação do nó n_k e assim, criando um novo ramo com os valores $\hat{x}_1, \hat{x}_2, ..., \hat{x}_k - 1, \bar{x}_{k+1}, ..., \bar{x}_n$. Pretende-se examinar os valores achados para o novo ramo, com solução $\bar{z}$, tem capacidade de melhorar a solução corrente $\hat{z}$, ou seja, verifique-se se $\bar{z} \geq \hat{z}$. Para isso acontecer, é preciso encontrar um limitante superior para $\bar{z}$, denotado por M onde não implique o cálculo de $\bar{x}_{k+1}, ..., \bar{x}_n$. Se com o limitante tem-se $M \leq \hat{z} \leq \bar{z}$ pode-se garantir que não é preciso fazer *Branching* ao nó n_k, fecha-se dito nó e poda-se o ramo, pois os valores $\bar{x}_{k+1}, ..., \bar{x}_n$ não forneceram uma melhora na solução corrente.

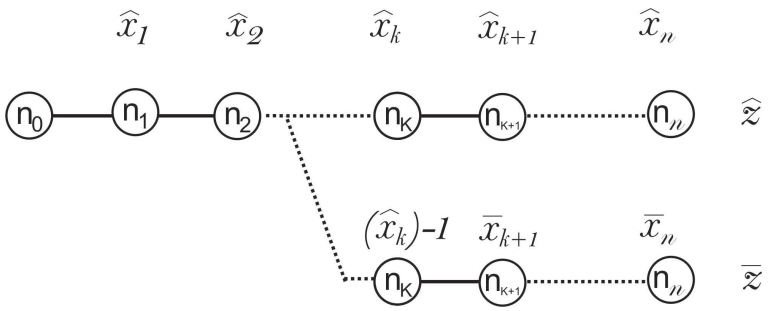

Figura 3.5: Etapa *k* do *Branching* para o exemplo do PMI
Fonte: Próprio autor.

Um limitante para $\bar{z}$ pode ser obtido seguindo as seguintes formulações:

$$\begin{aligned} \bar{z} &= \sum_{i=1}^{n} u_i x_i \\ &= \sum_{i=1}^{k} u_i \hat{x}_i + \sum_{i=k+1}^{n} u_i \bar{x}_i \end{aligned} \tag{3.6}$$

Observe que os valores conhecidos são $\hat{x}_1, \hat{x}_2, ..., \hat{x}_k - 1$ e os desconhecidos $\bar{x}_{k+1}, ..., \bar{x}_n$, onde o valor a serem estimado é $\sum_{i=k+1}^{n} u_i \bar{x}_i$. Como restrição que se tem no *Branching* do nó n_k é a capacidade da mochila (1.9) (além do (1.10)), ou seja, $\sum_{i=1}^{n} l_i \bar{x}_i \leq L$. Usando esta restrição e aplicado a (3.6), tem-se:

$$\begin{aligned} \bar{z} = \sum_{i=1}^{n} l_i x_i &= \sum_{i=1}^{k} l_i \hat{x}_i + \sum_{i=k+1}^{n} l_i \bar{x}_i \\ \Rightarrow \sum_{i=k+1}^{n} l_i \bar{x}_i &\leq L - \sum_{i=1}^{k} l_i \hat{x}_i \\ \Rightarrow \underbrace{\sum_{i=k+1}^{n} \frac{l_i}{u_i} u_i \bar{x}_i}_{\text{Valores Desconhecidos}} &\leq \underbrace{L - \sum_{i=1}^{k} l_i \hat{x}_i}_{\text{Valores Conhecidos}} \end{aligned} \tag{3.7}$$

Para limitar o valor desconhecido de (3.6), $\sum_{i=k+1}^{n} u_i \bar{x}_i$, observe que na expressão (3.7) aparece a relação l_i/u_i onde é preciso extrair para assim obter a limitação desejada. Agora, neste passo da formulação do algoritmo faz-se uso do critério de fixação das variáveis e a justificação da ordenação das variáveis pela eficiência de cada x_j.

Veja-se que as variáveis foram ordenadas seguindo o critério de eficiências das x_j de forma decrescente, ou seja, $\frac{u_1}{l_1} > \frac{u_2}{l_2} > ... > \frac{u_j}{l_j} > ... > \frac{u_n}{l_n}$ onde sua forma equivalente é

$\frac{l_1}{u_1} < \frac{l_2}{u_2} < ... < \frac{l_j}{u_j} < ... < \frac{l_n}{u_n}$. Pela anterior ordenação tem-se que:

$$min\left(\frac{l_i}{u_i}, i \in \{k+1, ..., n\}\right) = \frac{l_{k+1}}{u_{k+1}} \tag{3.8}$$

Assim, com (3.6)-(3.8) segue que:

$$\begin{aligned}
&\sum_{i=k+1}^{n} \frac{l_i}{u_i} u_i \bar{x}_i \leq L - \sum_{i=1}^{k} l_i \hat{x}_i \\
&\Rightarrow \sum_{i=k+1}^{n} \frac{l_i}{u_i} u_i \bar{x}_i = \sum_{i=k+1}^{n} min\left(\frac{l_i}{u_i}, i \in \{k+1, ..., n\}\right) u_i \bar{x}_i \leq L - \sum_{i=1}^{k} l_i \hat{x}_i \\
&\Rightarrow min\left(\frac{l_i}{u_i}, i \in \{k+1, ..., n\}\right) \sum_{i=k+1}^{n} u_i \bar{x}_i \leq L - \sum_{i=1}^{k} l_i \hat{x}_i \\
&\Rightarrow \sum_{i=k+1}^{n} u_i \bar{x}_i \leq max\left(\frac{u_i}{l_i}, i \in \{k+1, ..., n\}\right)\left(L - \sum_{i=1}^{k} l_i \hat{x}_i\right) \\
&\Rightarrow \sum_{i=1}^{k} u_i \hat{x}_i + \sum_{i=k+1}^{n} u_i \bar{x}_i \leq \sum_{i=1}^{k} u_i \hat{x}_i + max\left(\frac{u_i}{l_i}, i \in \{k+1, ..., n\}\right)\left(L - \sum_{i=1}^{k} l_i \hat{x}_i\right) \\
&\Rightarrow \bar{z} = \sum_{i=1}^{n} u_i \hat{x}_i \leq \sum_{i=1}^{k} u_i \hat{x}_i + max\left(\frac{u_i}{l_i}, i \in \{k+1, ..., n\}\right)\left(L - \sum_{i=1}^{k} l_i \hat{x}_i\right)
\end{aligned} \tag{3.9}$$

Pelo fato já falado da ordenação das variáveis, tem-se:

$$max\left(\frac{u_i}{l_i}, i \in \{k+1, ..., n\}\right) = \frac{u_{k+1}}{l_{k+1}} \tag{3.10}$$

Então, define-se o limitante M para $\bar{z}$, obtido de (3.9) e (3.10) como:

$$\bar{z} \leq \sum_{i=1}^{k} u_i \hat{x}_i + \frac{u_{k+1}}{l_{k+1}}\left(L - \sum_{i=1}^{k} l_i \hat{x}_i\right) = M \tag{3.11}$$

Com o limitante M para o PMI o algoritmo *B&B* para o Problema da Mochila Irrestrito se apresenta no Algoritmo 4.

Algoritmo 4: Algoritmo Branch and Bound para o Problema da Mochila Irrestrito.

Entrada: n, u_i, l_i, L
Saída: a_i, z
Inicialização: faça z:=0, k:=0
Passo 1 *(Branching, desenvolvimento de um ramo)*:

Entrada: n, u_i, l_i, L

Saída: a_i, z

Inicialização: faça z:=0, k:=0

Passo 1 *(Branching, desenvolvimento de um ramo)*:

para todo $j = k+1, .., n$ **faça**

$\hat{x_j} = \left\lfloor \left(L - \sum_{i=1}^{j-1} l_i \hat{x_i} \right) / l_j \right\rfloor$; k:=n; **Passo 2**

fim

Passo 2 *(Salvar solução corrente)*:

se $\sum_{i=1}^{n} c_i \hat{a_i} > z$ **então**

$z := \sum_{i=1}^{n} c_i \hat{a_i}$;

para todo $j = 1, ..., n$ **faça**

$a_i = \hat{a_i}$

fim

Passo 3

senão

Passo 3

fim

Passo 3 *(Backtrack, volta atrais)*:

se $\hat{a_k} = 0$ **então**

$k- = 1$

se $k = 1$ **então**

Pare!

fim

senão

$\hat{a_k} - = 1$

Passo 4

fim

Passo 4 *(Critério de Poda)*:

se $\sum_{i=1}^{k} u_i \hat{a_i} + \frac{u_{k+1}}{l_{k+1}} \left(L - \sum_{i=1}^{k} l_i \hat{a_i} \right) > z$ **então**

Passo 1

senão

Passo 3

fim

Fonte: Chvátal (1983, p. 206)

Aplicando o algoritmo *B&B* no problema (3.3)-(3.5) tem-se como solução ótima o ramo com os valores $x_1 = 2$, $x_2 = 1$, $x_3 = 0$, $x_4 = 0$. Na Figura 3.6 pode-se observar a exploração feita e as podas realizadas pelo algoritmo.

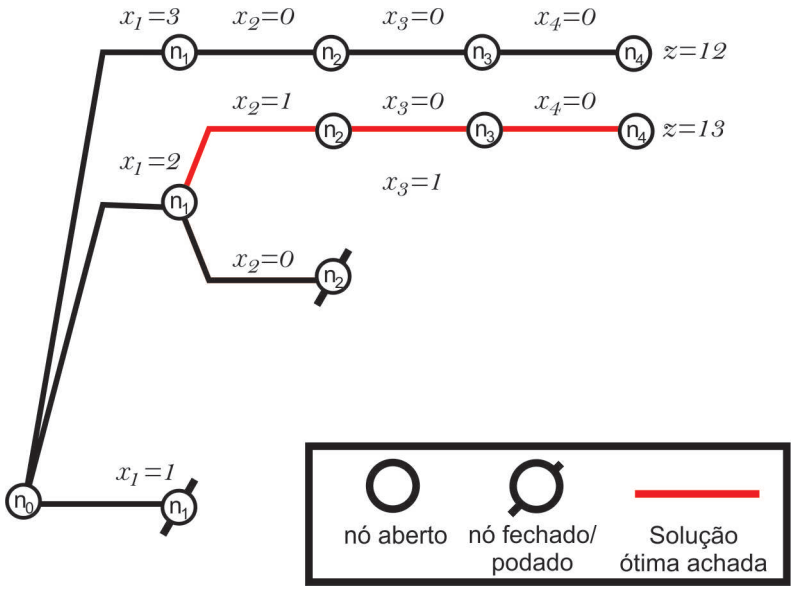

Figura 3.6: Árvore para o exemplo da Mochila Irrestrito com Podas
Fonte: Próprio autor.

3.4.1 Branch and Bound para o Problema da Mochila Restrito e 0-1

Usa-se a estrutura do *B&B* descrito no Algoritmo 4 para o Problema da Mochila Irrestrito. No caso do Problema da Mochila Restrito (1.4-1.7) observe que se tem que garantir que a quantidade de cada item escolhido não exceda a suas respectivas demandas (d_j).

Pela observação da demanda, façã-se modificação do Algoritmo 4 no passo 1 no cálculo de $\hat{x}_j$ por:

$$\hat{x_j} = min\left\{\left\lfloor\left(L - \sum_{i=1}^{j-1} l_i\hat{x_i}\right)/l_j\right\rfloor, d_j\right\}$$

Para o caso do Problema da Mochila 0-1 (1.1-1.3), tem-se que garantir que cada escolha x_j seja 0 ou 1. Para isto, partindo de novo do Algoritmo 4 faz-se a modificação do passo 1 de cada x_j por:

$$\hat{x_j} = min\left\{\left\lfloor\left(L - \sum_{i=1}^{j-1} l_i\hat{x_i}\right)/l_j\right\rfloor, 1\right\}$$

4 O MODELO LINEAR FORTE DO PROBLEMA DA MOCHILA COMPARTIMENTADA

Neste capítulo se apresentam componentes teóricos em Programação Inteira baseados em [32] para o *fortalecimento* dos dados relacionados com o PMCR junto à formulação de um modelo linear *forte*, o *Modelo Linear Forte do Problema da Mochila Compartimentada*.

Define-se um problema de programação inteira (PPI) junto a sua relaxação linear (RLPPI), como segue a continuação:

$$\begin{aligned} &\text{(PPI) Maximizar: } z_{IP} = cx && (4.1) \\ &\text{s.a.:} \quad Ax \leq b \\ &\quad x \geq 0 \text{ e } x \in \mathbb{Z} \end{aligned} \qquad \begin{aligned} &\text{(RLPPI) Maximizar: } z_{LP} = cx && (4.2) \\ &\text{s.a.: } Ax \leq b \\ &\quad x \geq 0 \text{ e } x \in \mathbb{R} \end{aligned}$$

Onde $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, e $x \in \mathbb{Z}_+^n$ é o vetor de variáveis de decisão. Para resolver Problemas de Programação Inteira, como em (4.1), é feito implementações usando software especializado em otimização combinatória. Este tipo de implementações usualmente tem duas etapas para resolver um PPI: primeiro relaxa linearmente o problema, como por exemplo a relaxação linear do P definido em (4.2), e segundo, com a solução da relaxação, faz uso de um método *Branch and Bound* com geração de restrições restritas para assim obter o ótimo do problema (veja-se a explicação do processo de *Branch and Bound* com geração de restrições restritas no Capítulo 3). Além do anterior, o software faz ajustes nas informações do problema (fala-se das informações do como os valores conhecidos que estão associados ao mesmo, no caso de PMC correspondem como informações do problema as utilidades e pesos relacionados aos itens, capacidades dos compartimentos, número de facas dos processo de corte, demanda dos itens, capacidade da mochila, entre outros) com a procura de "reduzir" o domínio das soluções factíveis do problema relaxado (região viável do problema relaxado) sem eliminar nenhuma solução factível do problema inteiro, de forma tal que, sempre é preservado a solução ótimo do mesmo.

A redução do domínio da relaxação linear do PPI favorecerá na eliminação de regiões fraccionarias do conjunto de soluções factíveis da relaxação linear do problema, tendo assim, uma maior limitação da exploração do método *Branch and Bound*, visando obter uma quantidade menor de nós a serem pesquisados.

Defina-se s o conjunto de soluções factíveis do PPI como o conjunto discreto $X_{PPI} = \{x \in \mathbb{Z}_+^n : Ax \leq b\}$. O conjunto de soluções factíveis da relaxação linear do PPI é o conjunto convexo $X_{RLPPI} = \{x \in \mathbb{R}_+^n : Ax \leq b\}$. Observe-se que o conjunto X_{RLPPI} contem todas as soluções inteiras do problema como $X_{PPI} = X_{RLPPI} \cap \mathbb{Z}_+^n$. Se se puder para a RLPPI redefinir

o domínio de soluções factíveis X_{RLPPI} por um domínio menor X'_{RLPPI} ($X'_{RLPPI} \subseteq X_{RLPPI}$), onde este novo conjunto preserve as soluções inteiras factíveis do problema, tem-se que para o RLPPI no processo involucrado para sua resolução será igual ou mais eficiente (em termos do tempo computacional requerido para sua solução) se for resolvido usando o domínio original da relaxação. Considerando o anterior, o RLPPI com domínio X'_{RLPPI}, vão-se dizer que é um *domínio reduzido* em comparação com o domínio original X_{RLPPI}. Do anterior, faça-se a continuação a definição do domínio reduzido de um RLPPI.

Definição 4.1 (Domínio reduzido)**.** Seja um PPI com relaxação linear RLPPI. Tem-se X como a região viável ou domínio de RLPPI com ótimo x^*. Se puder definir um novo conjunto X' com $X' \subseteq X$ e $x^* \in X'$, onde redefinindo o RLPPI com o conjunto X' como região viável, fala-se que X' é um domínio reduzido de X para a relaxação linear do PPI.

Para determinar um conjunto reduzido para a relaxação linear do PPI, é preciso desenvolver ajustes nos valores de $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ no modelo 4.1. Com os valores ajustados A' (proveniente do ajuste de A) e b' (proveniente do ajuste de b) pode-se formular um problema equivalente ao PPI. Denomine-se este problema equivalente como PPI', e pelo anterior, tem-se que $v(\text{PPI}) = v(\text{PPI'})$. Mas, como o novo modelo PPI′, o domínio da RLPPI′ e um domínio reduzido para a RLPPI, implicando que $v(\text{RLPPI'}) \leq v(\text{RLPPI})$. Pelo anterior, dize-se que o problema equivalente formulado PPI′ é um problema **forte** em comparação ao modelo original. Segue a continuação a definição do problema forte de um PPI.

Definição 4.2 (Modelo Forte para o PPI)**.** Considere-se os problemas equivalentes PPI e PPI′, com relaxação linear RLPPI e RLPPI′ respectivamente. Seja $X \subseteq \mathbb{R}^n$ e $X' \subseteq \mathbb{R}^n$ conjuntos convexos de soluções factíveis das RLPPI e RLPPI′, respectivamente, implicando que $X_{PPI} = X_{PPI'} = X \cap \mathbb{Z}^n_+ = X' \cap \mathbb{Z}^n_+$. O PPI' com relaxação linear com o conjunto X' é dito ser mais forte do que o modelo PPI com relaxação linear com o conjunto X, se $X' \subset X$.

Observe que definir o problema forte PPI′ do PPI não é suficiente o intercambiar as informações inicias A e b pelas informações ajustadas A' e b' que produz um domínio reduzido X', na qual tem-se que garantir a equivalência dos modelos. Para garantir o anterior, tem se que redefinir as restrições originais do PPI e estabelecer novas restrições para o PPI' (em caso seja necessário) para que tenham correspondência com os valores ajustados A' e b' e preservar a igualdade do valor ótimo dos problemas equivalentes. As restrições que são redefinidas ou criadas por causa dos valores ajustados A' e b' são chamadas de *restrições fortes válidas* (em inglês *Strong Valid Inequialities*). Para maior compreensão sob restrições válidas fortes para Problemas de Programação Inteira, recomenda-se [32] e [30].

Se para um PPI é definido um PPI', pode-se prever que o desempenho computacional na implementação do PPI' será igual (na pior das hipóteses) ou mais eficiente (respeito ao tempo de execução da implementação) no processo de resolução do problema. Do anterior, no transcurso da pesquisa de um método *Branch and Bound* para o PMCR, surge a motivação de "fortalecer"

o PMCR aproveitando-se ainda do modelo linear de [20] e do método interno de *Branch and Bound*

4.1 DOMÍNIO REDUZIDO PARA O PMCR

Vai-se aproveitar o Problema de Mochila Compartimentada em sua versão linear de [20] para estudar condições para fortalecer as informações do problema, procurando definir um domínio reduzido para a relaxação linear do PMCR. Lembre-se do Capítulo 2 as informações que compõe um PMC, mais especificamente para o PMCR: capacidade da mochila L (neste caso, fazendo à analogia com uma bobina de aço, corresponde a sua largura), conjunto de itens N particionado em q classes que tem associados uma utilidade u_i^k e um peso l_i^k com $i \in N_k$ e $k = 1, ...q$. Junto ao anterior, para cada item $i \in N_k$ com $k = 1, ..., q$ tem-se definido uma procura (ou demanda) d_i^k. Para cada classe $k = 1, ..., q$ se estabelece o número máximo de compartimentos que podem serem criados por p_k, junto as capacidades máximas e mínimas que podem ter os compartimentos de cada classe $k = 1, ..., q$. Finalmente tem-se definido a quantidades de facas disponíveis para o primeiro e segundo processo de corte das bobinas de aço, que são respectivamente F_1 e F_2.

Agora, tem-se como estratégia geral de fortalecimento das informações da RLPMCR manipular algumas das restrições do problema para procurar ter uma re-definição delas como restrições válidas fortes e assim, ir formando o domínio reduzido da RLPMC. As restrições que vão-se manipular são: a restrição da compartimentação das mochilas 2.24, onde tem-se como objetivo "apertar" os valores de L_{min}^k e L_{max}^k para todo $k = 1, ..., q$; a restrição 2.26 que limita o número de compartimentos que podem serem inseridos na mochila, onde tem-se como objetivo "apertar" F_1; a restrição 2.27 que limita o número de itens que podem serem inseridos na mochila, onde tem-se como objetivo "apertar" F_2. Outro tipo de fortalecimento que vão-se usar é o fortalecimento dos pesos (larguras) associados a acada item do problema, mediante o uso de um método *lifting* para ditos pesos.

Para o fortalecimento das informações do PMCR linear de de [20] será usar um exemplo simples de mochila compartimentada para facilitar a compreensão do leitor. Na Tabela 4.1 apresenta-se os dados do exemplo de um problema de mochila compartimentada a fortalecer.

4.1.1 Ajuste das capacidades dos compartimentos

Para cada classe $k = 1, ..., q$ tem-se definido limitantes superiores e inferiores que definem a capacidade dos compartimentos, que são L_{min}^k e L_{max}^k. Procura-se "apertar" cada L_{min}^k e L_{max}^k de cada classe $k = 1, ..q$, ou seja, formular novos $L_{min}^{k^*}$ e $L_{max}^{k^*}$ onde $L_{min}^{k^*} \geq L_{min}^k$ e $L_{max}^{k^*} \leq L_{max}^k$ tal que a relação $L_{min}^{k^*} \leq \sum_{i \in N_k} l_i^k a_{ij}^k \leq L_{max}^{k^*}$ seja uma relação de desigualdades válidas para o PMCR. Em seguida, fazendo comparação da relação de desigualdades formulada com a restrição de compartimentação original faça desta uma restrição redundante, ou seja, pode-se trocar a restrição original pela nova relação de desigualdades sem ter perda do ótimo

Classe (k)	Itens ($i \in N_k$)	u_i^k	l_i^k	d_i^k	L_{min}^k	L_{max}^k	p_k	L	F_1	F_2
1	1	0.82	7	3	9	12	2	23	14	8
	2	0.512	10	2						
	3	0.59	9	1						
2	1	0.283	11	3	5	13	4			
	2	0.975	10	6						
	3	0.594	7	4						
	4	0.286	9	2						
3	1	0.979	8	1	10	14	2			
	2	0.672	6	4						

Tabela 4.1: Informações de um Exemplar de um PMC que será "fortalecido".

do PMCR. Cumprindose o anterior, tem-se que a nova restrição de compartimentação $L_{min}^{k^*} \leq \sum_{i \in N_k} l_i^k a_{ij}^k \leq L_{max}^{k^*}$ é uma restrição válida forte para o PMCR. Então, define-se como a restrição de compartimentação forte para o PMC com $L_{min}^{k^*} \leq L_{min}^k$ (sem ter perda do ótimo) como:

$$L_{min}^{k^*} \leq \sum_{i \in N_k} l_i^k a_{ij}^k \leq L_{max}^{k^*} \tag{4.3}$$

Para calcular os novos $L_{min}^{k^*}$ e $L_{max}^{k^*}$ faça o seguinte: para cada classe $k = 1, ..., q$, avalia-se quantos itens no máximo podem serem inseridos em referencia ao limite superior do compartimento, L_{max}^k, mas preservando a demanda que tem disponível cada item, ou seja, que não ultrapasse a demanda definida para cada item, e também que a quantidade de itens a serem preenchidos da classe não ultrapasse o número de itens que podem serem criados, estabelecido pela faca F_2, e assim cumprindo com as exigência do problema. De forma análoga, para cada classe $k = 1, ..., q$, avalia-se quantos itens no mínimo podem serem inseridos em referencia ao limite inferior do compartimento, L_{min}^k, cumprindo a demanda que possui cada item e não ultrapassando o total de itens que podem serem criados estabelecido pela faca F_2. O anterior define para o PMCR sub-problemas de mochila restrito na qual vão fornecer novos limitantes, e assim, modificar as capacidades dos compartimentos, mais especificamente, reduzindo ditas capacidades. Este tipo de subproblemas de mochila restrito do PMCR para obter novos L_{max}^k e L_{min}^k, correspondem a problemas de mochilas restrito chamado de Problemas da Soma de Subconjuntos Restrito (veja-se a formulação geral do problema em 1.11-1.13 no Capítulo 1).

Então, seguindo os comentários anteriores, calcule-se $L_{max}^{k^*}$ para cada classe $k = 1, ..., q$, resolvendo um problema de mochila com restrições suplementares de cardinalidade:

$$\text{Maximizar: } \sum_{i \in N_k} l_i^k x_i^k \tag{4.4}$$

sujeito a:

$$\sum_{i \in N_k} l_i^k x_i^k \leq L_{max}^k \tag{4.5}$$

$$x_i^k \leq d_i^k, \text{ para cada } i \in N_k \tag{4.6}$$

$$\sum_{i \in N_k} x_i^k \leq F_2, \, i \in N_k \tag{4.7}$$

$$x_i^k \geq 0 \text{ e inteiros, para todo} i \in N_k, \, j = 1, ..., p_k \tag{4.8}$$

Fazendo um procedimento similar para obter o $L_{max}^{k^*}$, tem-se em relação ao L_{min}^k, para cada classe $k = 1, ..., q$ resolva-se um problema de mochila com restrições suplementares de cardinalidade:

$$\text{Minimizar: } \sum_{i \in N_k} l_i^k x_i^k \tag{4.9}$$

sujeito a:

$$\sum_{i \in N_k} l_i^k x_i^k \geq L_{min}^k \tag{4.10}$$

$$x_i^k \leq d_i^k, \text{ para cada } i \in N_k \tag{4.11}$$

$$\sum_{i \in N_k} x_i^k \leq F_2, \, i \in N_k \tag{4.12}$$

$$x_i^k \geq 0 \text{ e inteiros, } i \in N_k, \, j = 1, ..., p_k \tag{4.13}$$

Pode-se observar que o problema definido em (4.9)-(4.13), não define um problema de mochila. Então, faça conversão das restrições (4.10) e (4.12) na forma $\leq$ usando a variável $z_i^k = 1 - x_i^k$ obtendo uma reformulação de (4.9)-(4.13). Então, faça o cálculo $L_{min}^{k^*}$ para cada classe $k = 1, ..., q$, resolvendo o problema de mochila com uma restrição de cardinalidade suplementar:

$$\text{Maximizar: } \sum_{i \in N_k} l_i^k z_i^k - \sum_{i \in N_k} l_i^k \tag{4.14}$$

sujeito a:

$$\sum_{i \in N_k} l_i^k z_i^k \leq \sum_{i \in N_k} l_i^k - L_{min}^k \tag{4.15}$$

$$z_i^k \geq 1 - d_i^k, \text{ para todo } i \in N_k \tag{4.16}$$

$$\sum_{i \in N_k} z_i^k \geq 1 - F_2, \text{ para todo } i \in N_k \tag{4.17}$$

$$z_i^k \geq 0 \text{ e inteiros, } i \in N_k, \, j = 1, ..., p_k \tag{4.18}$$

Uma importante vantagens com o apertamentos de L^k_{min} e L^k_{max} para cada classe $k = 1, ..., q$ é que pode-se atualizar os valores de p_k, por valores mais apertados, obtendo assim maior limitação dos números de compartimentos que pode-se criar. Define-se os novos limitadores de compartimentos por classe apertados p_k como p_{k^*}, onde $p_{k^*} \leq p_k$.

Exemplo 5. Veja-se a Tabela 4.1 com os dados a fortalecer. Tem-se como limite inferior da capacidade de cada classe como $L^1_{min} = 9$, $L^2_{min} = 5$ e $L^3_{min} = 10$. Os limitantes superiores para cada classe são $L^1_{max} = 12$, $L^2_{max} = 13$ e $L^3_{max} = 14$. Agora usa-se os problemas de Soma de Subconjuntos Restritos para cada classe $k = 1, 2, 3$ para obter os novos limitantes, que são: como limites inferiores $L^{1^*}_{min} = 9$, $L^{2^*}_{min} = 7$ e $L^{3^*}_{min} = 12$; como limites superiores $L^{1^*}_{max} = 10$, $L^{2^*}_{max} = 11$ e $L^{3^*}_{max} = 14$. Veja-se que pode-se apertar o limitador do número de compartimentos que pode-se criar na classe 2 como $p_2 = min\{14, \lfloor 23/7 \rfloor\} = 3$ e para a classe 3 como $p_3 = min\{14, \lfloor 23/12 \rfloor\} = 1$. Os valores para a classe 1 de p_1 vão-se manter o mesmo. A Tabela 4.2 resume os dados obtidos.

Classe (k)	L^k_{min}	$L^{k^*}_{min}$	L^k_{max}	$L^{k^*}_{max}$	p_k	p_{k^*}
1	9	9	12	10	2	2
2	5	7	13	11	4	3
3	10	12	14	14	2	1

Tabela 4.2: Novos limitantes fortes $L^{k^*}_{min}$, $L^{k^*}_{max}$ e novos fortes p_k para cada classe $k = 1, 2, 3$ para as capacidades dos compartimentos do exemplo a fortalecer na introdução do Capítulo 4.

4.1.2 Ajuste dos valores das facas F_1 e F_2

De forma análoga no analise feito para "ajustar" os limites superiores é inferiores das capacidades dos compartimentos visto na Sub-seção 4.1.1, vão-se tratar com as facas F_1 e F_2.

Inicia-se com o ajuste da faca F_1, quem limita o número de compartimentos que se podem criar na mochila. Para ajustar o valor da faca F_1, tem-se que considerar a menor capacidade que pode ter os compartimentos de todas as classes, ou seja, com $\underset{k=1,..,q}{min}(L^{k^*}_{min})$, determine-se quantas vezes poderia ser inserido na mochila com capacidade L. Então, seguindo a ideia anterior, a nova faca, F^*_1 (observe que $F^*_1 \leq F_1$) será calculada como segue a continuação:

$$F^*_1 = \left\lfloor L / \underset{k=1,..,q}{min}(L^{k^*}_{min}) \right\rfloor \tag{4.19}$$

Como como feito para ajustar o valor da faca F_1, pode-se também ajustar o valor da faca F_2. Neste caso, tem-se que considerar para cada classe $k = 1, ..., q$ o item que tem associado a menor largura l_i com $i \in N_k$ e em seguida, avaliar quantas vezes pode ser inserido dito item em um compartimento, ou seja, comparando com respeito do limitante superior do compartimento $L^{k^*}_{max}$, mas, respeitando o valor inicial de F_2. Seguindo o razonamento anterior, para cada classe

$k = 1, ..., q$, associa-se uma nova faca F_2, na qual, sendo calculadas como segue:

$$F_2^k = min\left\{\left\lfloor L_{max}^{k^*}/\underset{i\in N_k}{min}(l_{min}^k)\right\rfloor, F_2\right\} \tag{4.20}$$

Exemplo 6. Usando as informações da Tabela 4.1 tem-se que $F_1 = 14$ e $F_2 = 8$. Então, calcule-se o novo F_1^* como $F_1^* = \lfloor 23/7 \rfloor = 3$. Agora, incia-se para a classe 1, $F_2^1 = min\{\lfloor 10/7 \rfloor\} = 1$; para a classe 2, $F_2^2 = min\{\lfloor 11/7 \rfloor\} = 1$ e para a classe 3, $F_2^3 = min\{\lfloor 14/6 \rfloor\} = 2$. Na Tabela 4.3 tem-se o resumo dos valores fortes obtidos:

Classe (k)	F_1	F_1^*	F_2	F_2^k
1	14	3	8	1
2				1
3				2

Tabela 4.3: Novos valores fortes para as facas F_1 e F_2 do exemplo a fortalecer na introdução do Capítulo 4.

4.1.3 *Lifting* (simples) as larguras associados aos itens do problema.

Outro tipo de apertamento que pode-se desenvolver nas informações do problema é ajustar o valor das larguras associadas aos itens de cada classe respeito à capacidade dos compartimentos. Tal pode ser feito nas restrições (2.24), porque é possível decompor esse grupo de restrições num conjunto de restrições independentes, cada uma delas dizendo respeito a valor de k diferente. Note-se que o ajustamento descrito nesta secção não pode ser aplicado aos valores dos coeficientes l_i^k da restrição (2.23) porque esses valores representam o espaço ocupado pelo item no padrão de corte, no qual se combina com outras tiras intermédias.

Seguindo as informações das larguras dos itens do exemplo referido na Tabela 4.1 veja-se uma forma de apertar os valores de ditas larguras.

Exemplo 7. Inicia-se com as larguras da primeira classe $k = 1$. Com os apertamentos dos anteriores exemplos feitos neste capítulo tem-se como restrição de compartimentação o seguinte:

$$L_{min}^{1^*} = 9 \leq l_i^1 a_{ij}^1 \leq L_{max}^{1^*} = 10$$

Observemos a desigualdade a esquerda e reescreva-se de forma explicita as informações ali contidas (como vão-se analisar só a compartimentação, troque-se a_{ij}^1 por x_i^1):

$$7x_1^1 + 10x_2^1 + 9x_3^1 \leq 10$$

Agora, vão-se analisar se for construído os compartimentos para a classe 1, quais itens podem serem inseridos. Para o item 1 da classe 1 veja-se que só poderia ser inserido uma vez só, do fato, que a combinação do item 1 com qualquer dos outros itens da classe 1, não vão cumprir com a restrição da compartimentação. Pelo anterior, a largura do item 1 de cumprimento 7 poderia-se ajustar por o valor máximo da capacidade do compartimento dessa classe 1 por $L^{1^*}_{max} = 10$, ou seja, pode-se ajustar l^1_1 por $l^{1^*}_1 = 10$. Esta ajuste da largura do item 1 vão cumprir a mesma função do seu antigo valor de que pode-se inserir uma vez só. Assim, com o ajuste, também tem-se um ajuste da desigualdade a esquerda da capacidade máxima dos compartimentos da classe 1 como:

$$10x^1_1 + 10x^1_2 + 9x^1_3 \leq 10$$

O item 2 da classe 1, como o valor de sua largura é o mesmo de $L^{1^*}_{max}$, não tem ajustes por fazer. Finalmente, vão-se analisar se pode-se ajustar o valor a largura associada ao item 3 da classe 1. Novamente, temos que, o item 3 pode-se inserir uma vez só, já que uma combinação com qualquer dos outros itens da mesma classe, não vão cumprir com capacidade máxima que pode ter o compartimento. Então, ajuste-se o valor de l^1_3 por $l^{1^*}_3 = L^{1^*}_{max} = 10$. Finalmente, obtemos como nova desigualdade a seguinte:

$$10x^1_1 + 10x^1_2 + 10x^1_3 \leq 10$$

De forma análoga, avalia-se para os itens das outras classes se tem, sob o razonamento feito, possibilidade de ajustar as larguras associadas. Na Tabela 4.4 é apresentado o resumo dos ajustes desenvolvidos.

Pode-se ver as larguras associadas dos itens como os coeficientes das variáveis de decisão em relação da quantidade dos itens que podem serem inseridos sujeito à capacidade de compartimentação da mochila. O processo feito no exemplo, qual consiste em incrementar o valor das larguras, é chamado de *lifting*. Veja-se que o *lifting* aplicado no exemplo só considero as larguras que só podem serem inseridos uma vez, por tal motivo foi catalogado como *lifting* simples. Mas, para exemplares maiores, tem-se a possibilidade de estudar o incremento das larguras para quando para certo item $i \in N_k$ da classe $k \in \{1, 2, ..., q\}$, combinado com outros itens da mesma classe pode-se inserido mais de uma vez e, procurando critério adicionais, possa-se aumentar o valor da largura.

Para este trabalho só foi considero uma estratégia de *lifting* simples, mas tem-se como proposta ampliar o estudo de ajuste das larguras dos itens para melhorar ainda mais o fortalecimento do PMC. Mais informação, pode-se consultar sob o *lifting* em coeficientes de Programação Inteira em [32] e [30].

Classe (k)	**Itens** ($i \in N_k$)	l_i^k	$l_i^{k^*}$
1	1	7	10
	2	10	10
	3	9	10
2	1	11	11
	2	10	11
	3	7	11
	4	9	11
3	1	8	8
	2	6	6

Tabela 4.4: Ajuste das larguras dos itens do exemplo a fortalecer na introdução do Capítulo 4.

O *lifting* simples aplicado para as larguras dos itens para o PMC, pode-se obter da seguinte forma: seja a classe $k \in \{1, 2, ..., q\}$ e o primeiro item $1 \in N_k$, avalia-se quantas vezes poderia ser inserido o primeiro item na mochila, ou seja, qual é o máximo de itens (sem o item 1) da mesma classe que podem serem inseridos sujeito a $\sum_{i \in N_k - \{1\}} l_i^k x_i^k \leq L_{max}^{k^*} - l_1^k$. Em seguida, para o item $2 \in N_k$ com $k \in \{1, 2, ..., q\}$, faça o mesmo razonamento mas agora tendo em consideração o novo valor (se foi ajustado) de $l_1^{k^*}$, ou seja, qual é o máximo de itens (sem o item 2) da mesma classe que podem serem inseridos sujeito a $\sum_{i \in N_k - \{1\}} l_i^k x_i^k \leq L_{max}^{k^*} - l_2^k$. De forma sucessiva se faz até o último item da classe k. Um aspeto importante é que, no momento de avaliar quantas vezes pode-se inserir um item $i \in N_k$ com $k \in \{1, 2, ..., q\}$ não pode ultrapassar o valor de sua demanda.

Como pode-se ver, o *lifting* simples implica resolver um problema de mochila. Então, seja a largura l_i^k com $i \in N_k$ e $k \in \{1, 2, ..., q\}$, onde, o *lifting* simples para l_i^k é resolver o problema de mochila:

$$\text{Maximizar: } \sum_{j \in N_k - \{i\}} l_j^k x_j^k \tag{4.21}$$

sujeito a:

$$\sum_{j \in N_k - \{i\}} l_j^k x_j^k \leq L_{max}^{k^*} - l_i^k \tag{4.22}$$

$$\sum_{j \in N_k - \{i\}} x_j^k \leq d_j^k \tag{4.23}$$

$$x_i^k \geq 0 \text{ e inteiros, } i \in N_k,\ j = 1, ..., p_k,\ k = 1, ..., q \tag{4.24}$$

Para finalizar a seção, na Tabela 4.5 pode-se ver em resumo o fortalecimento do exemplo de um PMCR com os dados referidos em 4.1 que foi apresentado no inicio do capítulo.

Classe (k)	**Itens** ($i \in N_k$)	u_i^k	l_i^k	$l_i^{k^*}$	d_i^k	L_{min}^k	L_{min}^{k*}	L_{max}^k	$L_{max}^{k^*}$	p_k	p_{k^*}	**L**	F_1	F_1^*	F_2	F_2^k
1	1	0.82	7	10	3	9	9	12	10	2	2	23	14	3	8	1
	2	0.512	10	10	2											
	3	0.59	9	10	1											
2	1	0.283	11	11	3	5	7	13	11	4	3					1
	2	0.975	10	11	6											
	3	0.594	7	11	4											
	4	0.286	9	11	2											
3	1	0.979	8	8	1	10	12	14	14	2	1					2
	2	0.672	6	6	4											

Tabela 4.5: Informações de um Exemplar de um PMC que foi “fortalecido”.

4.1.4 Diminuição de Simetrias do MLPMCR

Com os ajustes das inforamções associadas a um problema de mochila compartimentada até o momento, tem-se condições suficientes para fazer definição de um domínio reduzido para a relaxação linear do PMCR. Agora, volte-se à estrutura do Modelo Linear de referencia neste trabalho de dissertação, o modelo Linear de [20], e estude-se um fenômeno que se apresenta no processo de resolução do problema: simetria em soluções viáveis. Para entender o fenômeno de simetrias que acepta o modelo linear, veja-se o seguinte exemplo:

Exemplo 8. Continua-se com o exemplo a fortalecer com dados expostos na Tabela 4.1. Veja-se a mochila como uma bobina de aço e os itens como as fitas a produzir (veja-se o Capítulo 2). Na Figura 4.1 tem-se uma representação gráfica da bobina e das fitas só da classe 2, do fato que são os dados que vão-se tratar em este exemplo.

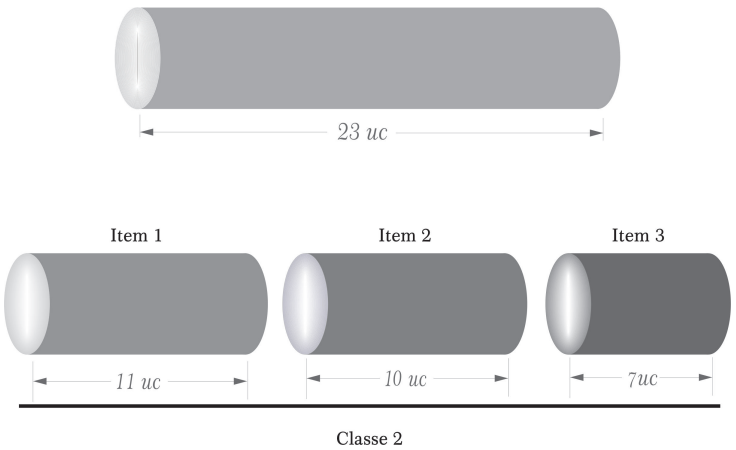

Figura 4.1: Representação dos itens da classe 2 da Tabela 4.1 como fitas a produzir.
Fonte: Próprio autor.

Lembrando da Sub-seção 4.1.1 que foram feitos processos de “ajustamento” de $L_{min}^2 = 5$ e $L_{max}^2 = 13$ por $L_{min}^{2^*} = 7$ e $L_{max}^2 = 11$ e também das larguras com o *lifting* simples aplicado em esses itens , com os novos valores para l_2^2 e l_3^2 por $l_2^2 = l_3^2 = 11$ (veja-se a Tabela 4.5). Do

anterior, tem-se que pode-se construir no máximo 3 compartimentos para a classe 2, ou seja, tem-se que $p_{k^*} = 3$. Com o itens 1 da classe 2, pode-se inserir em um dos compartimentos duas vezes, sendo assim, uma solução factível da compartimentação. A forma de ser inserido o item no compartimento tem três possibilidades:

1. Que seja inserido duas vezes no compartimento 1. Do anterior, tem-se que $\delta_1^2 = 1$. Em seguida, o espaço remanente da mochila vão-se de 1 uc, pelo qual, não pode-se inserir mais itens na mochila, fazendo inviável a construção dos outros compartimentos, desta forma, faça $\delta_2^2 = \delta_3^2 = 0$.

2. Que seja inserido duas vezes no compartimento 2. Do anterior, tem-se que $\delta_2^2 = 1$. Em seguida, como não foi preenchido nenhum item no primeiro compartimento, tem-se o espaço tudo disponível para o segundo compartimento (ou seja, não foi usado o compartimento 1, tem-se que $\delta_1^2 = 0$). Em seguida de inserir o item no compartimento 2, o espaço remanente da mochila vão-se de 1 uc, pelo qual, não pode-se inserir mais intens na mochila, fazendo inviável a construção do outro compartimento, desta forma, faça $\delta_3^2 = 0$.

3. Finalmente, pode-se não usar o primeiro e segundo compartimento, ou seja, faça $\delta_1^2 = \delta_2^2 = 0$, e em seguida, no compartimento três, vão-se inserir o item duas vezes, habilitando o último compartimento com $\delta_3^2 = 1$.

Na Figura 4.2 veja-se uma representação da forma como poderia-se inserir o item 1 da classe 2 na mochila. Observe-se que nas três possibilidades representadas faz referencia à mesma solução, tendo assim, uma simetria de uma solução factível para a compartimentação da mochila.

4.2 Reformulação linear do PMCR por o modelo linear forte do PMCR (MLFPMCR)

Com o ajuste feitos nas seções anteriores das informações associadas a um problema de mochila compartimentada, especificamente das limitações das capacidades para cada classe $k = 1, ..., q$ com $L_{min}^k \leq L_{min}^{k^*}$ e $L_{max}^{k^*} \leq L_{max}^k$, atualizando o número máximo de compartimentos que podem serem construídos por p_{k^*} onde $p_{k^*} \leq p_k$, o ajuste dos valores das facas por F_1^* e F_2^k e fazendo um *lifting* simples para algumas larguras dos itens, produz um *domínio reduzido* para a relaxação linear do PMCR. Do anterior, manipulando o modelo Linear de [20], tem-se uma redefinição das restrições do modelo, como segue a continuação:

Para a restrição da compartimentação do PMCR 2.24, tem-se como nova restrição a seguinte:

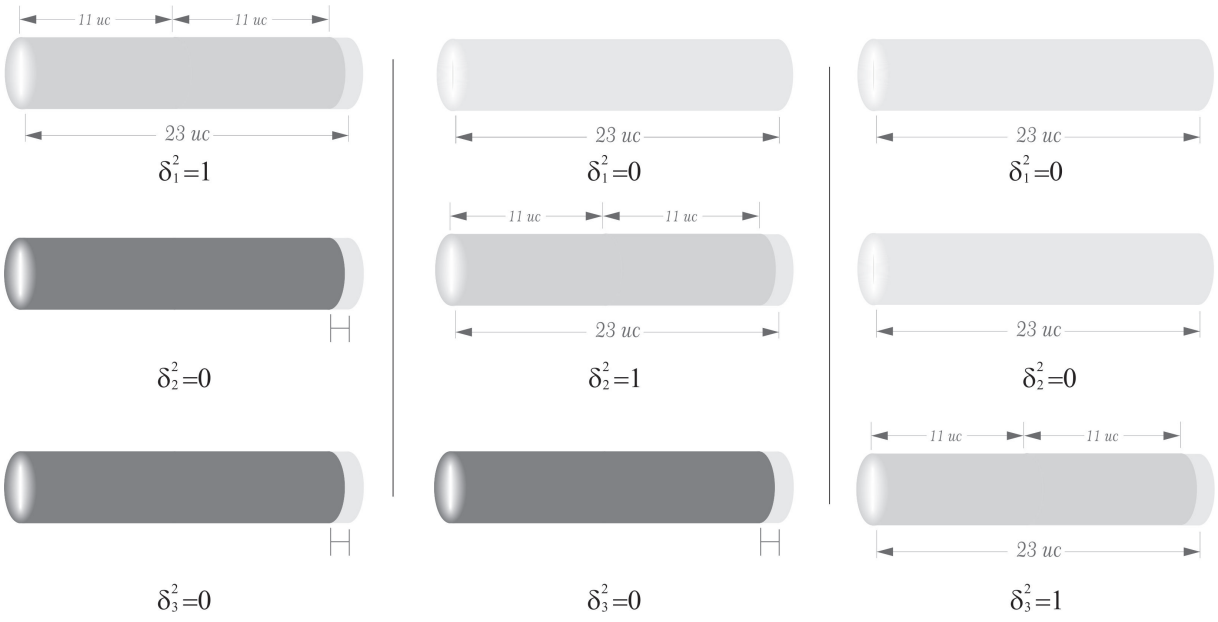

Figura 4.2: Exemplo de uma simetria para uma solução factível do exemplo a fortalecer referente à Tabela 4.1.

Fonte: Próprio autor.

$$\delta_j^k L_{min}^{k^*} \leq \sum_{i \in N_k} l_i^k a_{ij}^k \leq \delta_j^k L_{max}^{k^*} \text{ com } j = 1, ..., p_k, k = 1, ..., q \tag{4.25}$$

Na restrição do número de compartimentos que podem serem inseridos na mochila do PMCR 2.26 vão-se reescrever com o novo valor da faca F_1 como segue:

$$\sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta_j^k \leq F_1^* \tag{4.26}$$

Para a restrição do número de itens que podem serem inseridos na mochila do PMCR 2.27, para toda classe $k = 1, ..., q$, foi definido um novo valor da faca F_2, obtendo assim, a nova restrição:

$$\sum_{i \in N_k} a_{ij}^k \leq F_2^k, j = 1, ..., p_k, k = 1, ..., q \tag{4.27}$$

Da nova restrição (4.27) só vão-se avaliar no caso se o compartimento j da classe k for construído, ou seja, $\delta_j^k = 1$. Pelo anterior, pode-se fazer mais restritivo (4.27), adicionando para cada classe k, que só seja avaliado a quantidade de itens que podem serem inseridos na mochila se o compartimento j dessa classe está habilitado, ou seja:

$$\sum_{i \in N_k} a_{ij}^k \leq F_2^k \delta_j^k,\ j = 1, ..., p_k, k = 1, ..., q \tag{4.28}$$

Como foi observado na subseção 4.1.4, o problema da mochila compartimentada é susceptível a ter simetrias nas soluções factíveis. Então, pode-se adicionar uma restrição que possa diminuir dito efeito. No Exemplo 8, pode-se ver que na simetria, ser preenchido no primeiro compartimento é equivalente a ser preenchido nos outros compartimentos, podendo assim, deixar a primeira compartimentação e eliminar as possibilidades de preenchimentos nos compartimentos seguintes. Seguindo a ideia exposta, adiciona-se uma restrição que faça diminuição do efeito de simetria no PMCR, como segue a continuação:

$$\delta_j^k \geq \delta_{j+1}^k, \text{ para todo } j = 1, \ldots, p_k - 1 \text{ e } k = 1, \ldots, q \tag{4.29}$$

Agora, com a restrição de simetria, pode-se reforçar a restrição da demanda do PMCR 2.25 dos itens fazendo que seja só avaliado a demanda que foi definida no primeiro compartimento, do fato, que os compartimentos que apresentavam simetrias foram descartados, ou seja, se compartimento j da classe $k \in \{1, ..., q\}$ é uma simetria do problema, tem-se que $\delta_j^k = 0$. Assim, redefine-se a restrição da demanda dos itens do PMCR como segue a continuação:

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k \delta_1^k,\ i \in N_k,\ k = 1, ..q \tag{4.30}$$

Finalmente com as novas restrições (4.25)-(4.30) vão definir um novo modelo linear para o PMCR, sendo este um modelo equivalente ao Modelo Linear de [20], na qual será chamado como **O Modelo Linear Forte do Problema da Mochila Compartimentada**, e sendo formulado como segue a continuação:

$$\text{Maximizar: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_{k^*}} u_i^k a_{ij}^k \tag{4.31}$$

$$\text{sujeito a: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_{k^*}} l_i^k a_{ij}^k \leq L \tag{4.32}$$

$$\delta_j^k L_{min}^{k^*} \leq \sum_{i \in N_k} l_i^{k^*} a_{ij}^k \leq \delta_j^k L_{max}^{k^*} \text{ com } j = 1, ..., p_{k^*}, k = 1, ..., q \tag{4.33}$$

$$\delta_j^k \geq \delta_{j+1}^k, \forall j = 1, \ldots, p_k - 1 \text{ e } k = 1, \ldots, q \tag{4.34}$$

$$\sum_{j=1}^{p_{k^*}} a_{ij}^k \leq d_i^k \delta_1^k, i \in N_k, k = 1, ..q \tag{4.35}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{p_{k^*}} \delta_j^k \leq F_1^* \tag{4.36}$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2^k \delta_j^k, j = 1, ..., p_{k^*}, k = 1, ..., q \tag{4.37}$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ e inteiros, } i \in N_k, j = 1, ..., p_{k^*}, k = 1, ..., q \tag{4.38}$$

A função objetivo apresentada em (4.31) representa a soma das utilidades de todos os itens escolhidos em cada compartimentos. A restrição (4.32) garante que a soma das larguras dos itens escolhidos é limitada pela capacidade da mochila. A restrição (4.33) limita a capacidade dos compartimentos não nulos e anula os coeficientes dos compartimentos que não serão utilizados com a variável de controle δ_j^k ($\delta_j^k = 1$ se o compartimento j da classe k é não nulo e $\delta_j^k = 0$ no caso contrário). A restrição (4.34) é uma restrição de redução de simetrias no problema. A restrição (4.35) limita a quantidade de cada item $i \in N$ por sua respectiva demanda mas, com o controle δ_1^k o valor de a_{ij}^k será diferente de 0 só se o compartimento j na classe k for construído. A restrição (4.36) limita a quantidade de compartimentos não nulos que podem serem inseridos na mochila com um valor de faca apertado e em (4.37) a quantidade de itens que podem serem inseridos em cada compartimento onde para cada classe tem um valor associado do valor apertado de F_2. Na restrição (2.28) estabelece o domínio das variáveis do problema.

5 UM ALGORITMO *BRANCH AND BOUND* PARA O PROBLEMA DA MOCHILA COMPARTIMENTADA

Com a modelagem linear do PMC, junto a proposta de limitar o número de compartimentos para cada classe, inspira-se à aplicação de métodos *Branch and Bound* como técnica de solução exata ao problema. Neste capítulo, presenta-se o desenvolvimento de um algoritmo *Branch and Bound* para o PMCR, explorando um processo sistemático de enumeração de soluções viváveis e o estudo de técnicas de geração de limitantes superiores que são usados como critério de poda de ramos na árvore de enumeração.

No Capítulo 3 foi visto como o algoritmo de um *Branch and Bound* pode ser aplicado em Problemas de Programação Inteira (PPI) para fornecer a solução exata do problema. Para criar dito algoritmo, precisa-se de dois componentes essenciais: critérios para formar a árvore de enumeração (o *Branching*), em outra palavras, a enumeração implícita das soluções sensíveis do problema (ou soluções candidatas a ser ótimo do problema) e limitantes superiores (o *Bounding*) que estabeleça o comportamento de exploração com a intenção de fazer o algoritmo viável na prática da busca do ótimo para o PMC. Estes dois componentes para formar um algoritmo *Branch and Bound* para o PMC será explorado nas seções seguintes.

5.1 DEFINIÇÃO DA ÁRVORE DE ENUMERAÇÃO PARA O PMCR

O algoritmo que definirá o processo de *Branching* na criação da árvore de enumeração para o PMCR terá como referência o algoritmo *Branch and Bound* desenvolvido para o Problema da Mochila Restrito por [5] (veja-se o processo no Capítulo 3). Os nós da árvore de enumeração para o PMCR será compostos pelas variáveis de decisão do problema, as quais, vão serem fixados. Uma primeira questão é decidir o critério de fixação das variáveis procurando ter alguma vantagem no momento de desenvolver o processo de busca no *Branch and Bound*. Para isto, lembre-se do processo feito para o Problema da Mochila Restrito, onde as variáveis de decisão que vão serem fixadas são ordenadas em correspondência da ordenação dos itens feita pela eficiência (relação entre a utilidade e a largura), obtendo assim como vantagem redução na criação de ramos e maior eficiência no processo de poda com o limitante Backtrack. Para o caso do PMC, não só tem-se que pensar como ordenar os itens, se não também como ordenar as classes.

Pode-se definir diversos critérios para estabelecer um critério de relação entre as classes, por exemplo, escolher de cada classe a melhor eficiência y fazer de este o representante dela para desenvolver o processo comparativo. Em testes iniciais foi ampliado o critério anterior e escolhido como via de comparação, em lugar de só escolher um representante, faça o cálculo para cada classe da média das eficiências dos seus itens. Com o anterior, poderia-se esperar que as classes com maior média de eficiência seja qual seus itens façam mais vezes parte da

composição dos compartimentos na mochila. Seguindo as ideias anteriores, faça definição da eficiência para uma classe k:

Definição 5.1 (Eficiência das Classes para PMCR)**.** Seja k uma classe onde $k \in C = \{1, ..., q\}$ e N_k o conjunto de itens indexados de k. Define-se a eficiência da classe k como a média das eficiências dos seus itens.

Nota-se a eficiência da classe k por $\psi_k = \sum_{i \in N_k} \frac{u_i}{l_i} / |N_k|$. Com o critério para representar as eficiências das classes, agora define-se a relação das classe para o PMCR, como segue:

Definição 5.2 (Relação das Classes para o PMCR)**.** Seja $C = \{1, ..q\}$ o conjunto das classes do PMCR e $\psi_k \in \mathbb{R}$ a eficiência da classe $k \in C$. Define-se a relação $\preceq$ em $C \times C$ como:

$$\forall k, s \in C, k \preceq s \Leftrightarrow \psi_k \leq \psi_s$$

Com o estabelecimento da relação de ordem das classes tem-se a condições para expor os diferentes algoritmos que vão criar a árvore de enumeração sistemáticas das soluções viável do PMCR, que será estudado nas próximas sub-seções.

Com a fixação das variáveis feita, inicia-se o processo de construção das soluções sensíveis do problema, onde graficamente será representado por um ramo da árvore.

5.1.1 Algoritmo de fixação dos nós para o PMCR

Para cada classe $k = 1, ..., q$ calcule-se sua eficiência ψ_k e ordene-se de forma decrescente. Cada classe $k = 1, ..., q$, ordene-se os itens $i \in N_k$ por eficiência de forma decrescente. Fixa-se as variáveis de decisão segundo a ordem das classes e os itens para os p_k compartimentos. Para facilitar o processo de pesquisa de preenchimento, faça uma indexação global das variáveis de decisão de 1 até M, com $M = \sum_{k=1}^{q} P_k \cdot |N_k|$, que é o total de variáveis a trabalhar. Agora, redefine-se a variável de decisão a_{ij}^k por α_n seguindo a anterior indexação feita, sendo esta quem vão a representar os nós da árvore. Cada α_n tem informações dos valores associados aos itens, compartimentos e classes referida ao nó n. Para compreender este processo, sirva-se como referencia o seguinte exemplo.

Exemplo 9. Suponha-se um PMCR com $q = 2$, refira-se a classe 1 como q_1 e a classe 2 como q_2, o conjunto de índices dos itens do problema $N = \{1^1, 2^1, 1^2, 2^2, 3^2\}$, onde os índices dos itens da classe q_1 são $N_1 = \{1, 2\}$ e para à classe q_2 são $N_2 = \{1, 2, 3\}$. Tem-se para q_1, $p_1 = 3$ compartimentos e para q_2 $p_2 = 2$ compartimentos. Sem ordenar os itens e classes, as variáveis de decisão poderiam serem fixados como segue: a_{11}^1, a_{21}^1, a_{12}^1, a_{22}^1, a_{13}^1, a_{23}^1, a_{11}^2, a_{21}^2, a_{31}^2, a_{12}^2, a_{22}^2 e a_{32}^2. Agora, suponha-se que $q_2 \preceq q_1$ e listando os índices de itens ordenados de maior a menor segundo sua eficiência é para q_1 $\{2^1, 1^1\}$ e para q_2 $\{2^2, 1^2, 3^2\}$. Seguindo a ordenação anterior, as variáveis serão fixadas na seguente ordem: a_{21}^2, a_{11}^2, a_{31}^2, a_{22}^2, a_{12}^2, a_{32}^2, a_{21}^1, a_{11}^1, a_{22}^1, a_{12}^1, a_{23}^1

e a^1_{13}. Indexando as variáveis de decisão e formando assim os nós da árvore como: $\alpha_1 = a^2_{21}$, $\alpha_2 = a^2_{11}$, $\alpha_3 = a^2_{31}$, $\alpha_4 = a^2_{22}$, $\alpha_5 = a^2_{12}$, $\alpha_6 = a^2_{32}$, $\alpha_7 = a^1_{21}$, $\alpha_8 = a^1_{11}$, $\alpha_9 = a^1_{22}$, $\alpha_{10} = a^1_{12}$, $\alpha_{11} = a^1_{23}$ e $\alpha_{12} = a^1_{13}$. Veja-se uma representação do exemplo na Figura 5.1.

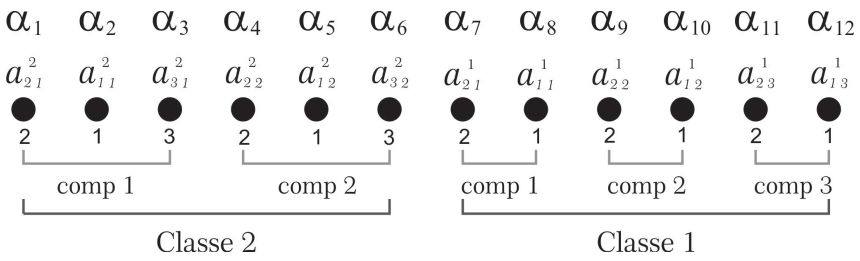

Figura 5.1: Representação da fixação dos nós para o PMC
Fonte: Próprio autor.

Finalmente o Algoritmo 6 apresenta o processo de ordenação e fixação das variáveis para o PMCR.

Algoritmo 5: Ordenação das informações do PMCR.

Inicialização 1 (Ordenação das classes):

- Ordene-se para cada classe $k = 1, ..., q$ seus itens pela eficiência de forma decrescente.
- Para cada $k = 1, ..., q$ calcule-se ψ_k.
- Ordene-se cada classe em ordem decrescente segundo ψ_k.
- Fixa-se as variáveis (a^k_{ij}) seguindo a ordenação feita acima: primeiro pelas classes, em seguida em cada compartimento disponível seguindo p_k ordene-se os itens. Faça uma indexação global das variáveis, de 1 até M, com $M = \sum_{k=1}^{q} p_k \cdot |N_k|$.

Fonte: Próprio autor.

5.1.2 Passo 1: Construção dos Compartimentos

Nesta subseção é descrito o processo de preenchimento dos itens nos compartimentos como o primeiro passo que desenvolve a árvore de enumeração para o PMCR. Para comodidade do leitor, apresenta-se a continuação uma série de notações que são usados no resto de algoritmo para o *Branch and Bound*.

- a^k_{ij} variáveis de decisão do problema.
- α_n representa o nó da árvore, onde estão fixados as variáveis de decisão (cada α_n tem informação do item, compartimento e classe onde ele esteja localizado).

- $m = \sum_{k=1}^{q} p_k$, soma total de compartimentos que podem serem construídos.
- δ_j^k, controle para compartimentos usado ou não usado. Quando o compartimento j da classe k é usado tem-se que $\delta_j^k = 1$ e é zero no caso contrário.
- $F1$ representa a quantidade de compartimentos (de facas disponíveis para o primeiro processo de corte) que ainda podem serem construídos.
- $F2_j^k$, quantidade de itens que ainda podem serem inseridos para cada compartimento (quantidades de facas disponíveis para o segundo processo de corte).
- $Lmoch$, espaço disponível na mochila.
- $Lcom_j^k$, espaço ainda disponível no compartimento.
- D_i, demanda ainda disponível para cada item.
- r, posição do nó que esteja em avaliação ou em preenchimento.
- $Ordem(r)$, leitura das informações (i, j, k) da variável fixa no nó r.
- z, solução corrente.
- $fecha\_comp$, vetor com informações dos nós onde termina cada compartimento.
- $abre\_classe$, vetor com informações dos nós que iniciam cada classe.
- Os comentário nos álgoritmos será precedidos por "!".

Em seguida da ordenação das informações do problema pelo Algoritmo 6, faça a construção dos compartimentos, onde para o nó do indicador r, faça leitura da classe (suponha-se que é a classe s) e o compartimento associado a esse nó (suponha-se que é o compartimento j) em seguida fazer o respetivo preenchimento dos o itens i da classe no compartimento associado segundo a disponibilidade do espaço que se tem na mochila $Lmoch$, a disponibilidade que se tem no compartimento $Lcom_j^k$ como além da demanda disponível do item i D_i e a verificação sim se tem facas disponíveis para seu corte com $F2_j^k$. Os compartimentos podem serem criados sim se tem facas disponíveis para sua criação com $F1$. Se o compartimento j for preenchido com pelo menos um item, o compartimento é aberto com $\delta_j^k = 1$, em caso contrário é fechado com $\delta_j^k = 0$. Veja-se uma representação deste processo na Figura 5.2 e tem-se o respetivo algoritmo de este no Algoritmo 7.

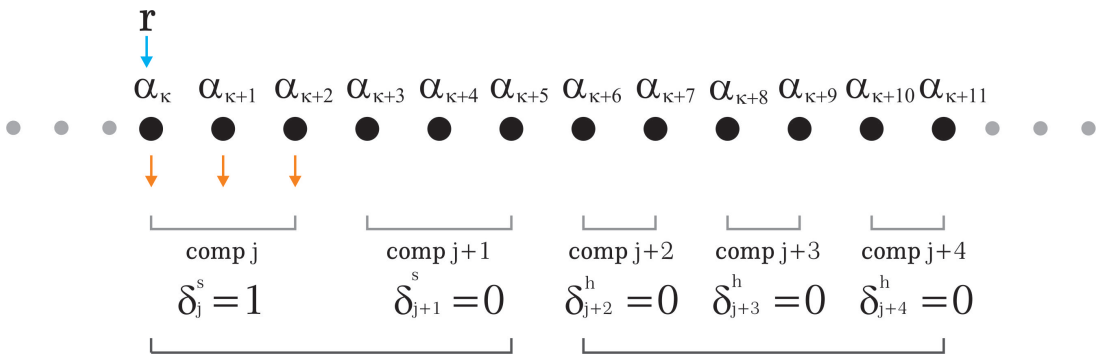

Figura 5.2: Representação do processo de criação dos compartimentos no Passo 1 da árvore de enumeração do PMCR.

Fonte: Próprio autor.

Algoritmo 6: Passo 1, construção dos compartimento para o PMCR.

Entrada: L, L^k_{min}, L^k_{max}, u_i, l_i, d_i,F_1,F_2

Saída: a^k_{ij}, z

Inicialização 2:

Faça $z = 0$; $r = 0$; $F1 = F_1$; $F2^k_j = F_2$ para $k = 1, ..., q$ e $j = 1, ..., p_k$; $\delta^k_j = 0$ para $k = 1, ..., q$ e $j = 1, ..., p_k$; $Lmoch = L$; $Lcomp^k_j = L^k_{max}$ para $k = 1, ..., q$ e $j = 1, ..., p_k$; $comp = 1$; $classe = 1$; $D_i = d_i$, $rr = 0$.

Passo 1 *(Construção dos compartimentos)*:

para todo $n = r + 1, .., fecha\_comp(comp)$ **faça**

  $(i, j, k) = ordem(n)$!leitura da posição r no nó da árvore.

  se $F1 > 0$ *ou* $\delta^k_j = 1$ **então**

    $\alpha_n = min\left\{ \left\lfloor \frac{min\{Lmoch, Lcomp^k_j\}}{l_i} \right\rfloor, D_i, F2^k_j \right\}$;

    $Lmoch- = l_i \cdot \alpha_n$; $Lcomp^k_j- = l_i \cdot \alpha_n$

    $D_i- = \alpha_n$;

    se $\alpha_n > 0$ *e* $\delta^k_j = 0$ **então**

      $\delta^k_j = 1$; $F1- = 1$

    fim

  senão

    $\alpha_n = 0$

  fim

  Passo 2

  $rr = r$

fim

Fonte: Próprio autor.

5.1.3 Passo 2: Verificação de factibilidade dos compartimentos construídos

No Algoritmo 7 tem-se verificação constante das restrições das facas, demandas e o cumprimento de não ultrapassar a capacidade da mochila e dos compartimentos, mas não se tem a verificação da satisfação que o compartimento tenha capacidade igual o maior do que L^k_{min}. Do anterior é definido um algoritmo para tal propósito como o Passo 2 da árvore de enumeração. Ante de enunciar este processo de verificação, define-se alguns procedimentos que vão-se utilizar no algoritmo. Caso que o compartimento não seja factível, faz-se uma devolução dos espaços ocupados na mochila, espaços ocupados no compartimento, devolução da demanda usada dos itens, feche-se o compartimento e zere-se α_n. Este processo de devolução é descrito no Algoritmo 8.

Quando no processo de verificação o compartimento j da classe s é declarado não factível, tem-se a seguentes considerações: se ainda tem-se compartimentos disponíveis para preenchimento na classe s, estes compartimentos não tem-se que someter ao processo de construção, pelo qual é obrigado o algoritmo a iniciar a construção de compartimentos na seguente classe do s; em caso $s = q$, é finalizado o processo de construção. Este processo de verificação da classe é descrito no Algoritmo 9.

Algoritmo 7: Passo 2, Devolução.

$Lmoch+ = l_i \cdot \alpha_n;\ Lcomp^k_j+ = l_i \cdot \alpha_n;\ D_i+ = \alpha_n;$
se $\delta^k_j = 1$ **então**
  $\delta^k_j = 0; F1+ = 1$
senão
  $\alpha_n = 0$
fim

Fonte: Próprio autor.

Algoritmo 8: Passo 2, Verificação da Classe.

se $classe = q$ **então**
  $r = M;\ comp = m;$ **Passo 3**;
senão
  $comp = \sum_{k=1}^{classe} (p_k + 1)$!O primeiro compartimento da próxima classe
  $classe+ = 1;$
  $r = abre\_classe(classe) - 1;$
  Passo 1;
fim

Fonte: Próprio autor.

Caso contrário, se o compartimento é factível, traslada-se ao seguinte compartimento e faz-se sua construção e faça o processo de avaliação novamente. Este processo de

compartimento factível pode-se ver no Algoritmo 10.

Algoritmo 9: Passo 2, Compartimento Factível.

$comp+ = 1$;
se $classe = q$ **então**
  Passo 1;
senão
  se $abre\_classe(classe + 1) = fecha\_comp(comp - 1) + 1$; **então**
    $classe+ = 1$;
    Passo 1;
  fim
fim

Fonte: Próprio autor.

Alguns algoritmo adicionais são introduzidos para melhorar processos em passos que são feitos na árvore de enumeração, produto de avaliações previas do algoritmo por meio de testes de funcionalidade do mesmo, sendo chamados estes como *algoritmos de controle*. Por exemplo, para o passo 1 é feito o Algoritmo 11, um algoritmo de controle que tem como função desenvolver uma exploração mais profunda quando o indicar r é devolvido (por um processo de *backtrack* que será descrito mais adiante) a um nó que tem associado o compartimento 1, evitando que nesse processo de volta o algoritmo faça um pare inadequado do processo iterativo.

Algoritmo 10: Passo 2, processo de controle.

se $comp = 1$ **então**
  $r = rr$; !faça leitura da posição r que inciou o processo de construção do compartimento
  se $\alpha_n = 0 \; \forall n = 1, ..., fecha\_comp(1)$ **então**
    Faça Algoritmo 9 *(Verificação da Classe)*
  fim
  Passo 4;
fim
se $rr <> fecha\_comp(comp)$ **então**
  $r = rr + 1$; **Passo 1**;
senão
  Faça Algoritmo 9 *(Verificação da Classe) (Devolução)*
fim

Fonte: Próprio autor.

Finalmente tem-se o algoritmo de verificação descrito no Algoritmo 12 como o passo 2 da árvore de enumeração do PMCR.

5.1.4 Passo 3 e 4: salvar a solução corrente e processo de volta

Quando o processo da construção e verificação dos compartimentos finaliza, situando pelos algoritmos anteriores o indicador r no nó M, fala-se que o ramo foi construído. Em seguida, com o ramo construído, faça validação do valor obtido na função objetivo. Se o valor destes valores fornece o melhor valor da função objetivo em relação as anteriores iterações, é salvada estes valores como a melhor solução atual (solução corrente) para as variáveis a_{ij}^k e em seguida se dá continuidade para o seguente processo. No Algoritmo 13 faz descrição deste processo.

Algoritmo 11: Passo 2, verificação da factibilidade dos compartimentos construídos.

Passo 2 *(Verificação da Factibilidade)*:
$r = fecha\_comp(comp)$
se $r \neq M$ **então**
  se $\sum_{n=inicio\_comp(comp)}^{fecha\_comp(comp)} \alpha_n \cdot l_{ordem(n)} \geq L_{min}^{ordem(n)}$ **então**
    Faça Algoritmo 10 *(Compartimento Factível)*
  senão
    Faça Algoritmo 11 (Controle)
    para todo $n = inicio\_comp, ..., fecha\_comp(comp)$ **faça**
      Faça Algoritmo 8 *(Devolução)*
    fim
  fim
senão
  se $\sum_{n=fecha\_comp(comp-1)+1}^{M} \alpha_n \cdot l_{ordem(n)} \geq L_{min}^{ordem(n)}$ **então**
    Passo 3
  senão
    para todo $n = fecha\_comp(comp-1)+1, ..., M$ **faça**
      Faça Algoritmo 8 *(Devolução)*
    fim
  fim
fim

Fonte: Próprio autor.

Com o indicador r no nó M é iniciado o processo de retorno em direção à raiz da árvore até o primeiro $\alpha_n \neq 0$, processo chamado de *volta* o em inglês de backtrack. Este processo vão configurar o passo 4 do desenvolvimento da árvore de enumeração para o PMCR. Neste processo não só vai-se ter o processo de retorno, senão também vai-se ter os critérios de parada do algoritmo. Os algoritmos 14 e 15 são os critérios de parada fixados para à árvore de enumeração do PMCR.

Também foram definidos algoritmos de controle para obrigar ao processo iterativo fazer construção dos compartimentos com os novos valores de α_n como a constante correção da

posição do valor de com e $Class$. Estes processos são descritos nos próximos algoritmos (16)-(18).

Algoritmo 12: Passo 3, salvar solução corrente.

Passo 3 *(Salvar Solução Corrente)*:
se $\sum_{n=1}^{M} \alpha_n \cdot u_{ordem(n)} > z$ **então**
  $z = \sum_{n=1}^{M} \alpha_n \cdot u_{ordem(n)}$
  para todo $n = 1, ..., M$ **faça**
    $(i, j, k) = ordem(n)$; !faça leitura das informações do nó n
    $a(i, j, k) = \alpha(n)$;
  fim
  Passo 4;
senão
  Passo 4;
fim

Fonte: Próprio autor.

Algoritmo 13: Passo 4, critério de Pare! 1.

se $classe = q$ *e* $\alpha_n = 0,\ \forall n = 1, .., M$ **então**
  Pare!
fim

Fonte: Próprio autor.

Algoritmo 14: Passo 4, critério de Pare! 2.

se $comp = 1$ *e* $\alpha_n = 0,\ \forall n = 1, .., fecha\_comp(1)$ **então**
  Pare!
fim
se *comp=1* **então**
  Passo 5
fim

Fonte: Próprio autor.

Algoritmo 15: Passo 4, controle para construção de compartimentos 1.

se $classe <> q$ *e* $\alpha_n = 0,\ \forall n = 1, .., fecha\_comp(1)$ **então**
  $classe+ = 1$;
  $r := abre\_classe(classe) - 1$; **Passo 1**
fim

Fonte: Próprio autor.

Algoritmo 16: Passo 4, controle para construção de compartimentos 2.

se $r = 0$ **então**
  $classe+ = 1$;
  $r := abre\_classe(classe) - 1$; **Passo 1**
fim

Fonte: Próprio autor.

Algoritmo 17: Passo 4, controle do nó com mudança "com" e "Class".

Passo 4 *(Volta)*:
se $fecha\_comp(comp - 1) = r$ **então**
  $comp- = 1$
  se $\delta^{ordem(r)}_{ordem(r)} = 1$ **então**
    $\delta^{ordem(r)}_{ordem(r)} = 0; \quad F1+ = 1;$
  fim
  se $r + 1 = abre\_classe(classe)$ **então**
    $classe- = 1$
  fim
  Passo 5
senão
  $(i, j, k) = ordem(r)$
  se $\alpha_n = 0 \; \forall n = fecha\_comp(comp - 1) + 1, .., fecha\_comp(comp)$ ***e*** $\delta^k_j = 1$ **então**
    $\delta^k_j = 0; \; F1+ = 1;$
  fim
  Passo 5
fim

Fonte: Próprio autor.

Finalmente define-se o passo 4 de volta ou *bactrack* para à árvore de enumeração do PMCR como é apresentado no Algoritmo 19.

5.1.5 Passo 5 e 6: ramificação da árvore e poda.

Com a posição do indicador r no nó n como o primeiro nó $\alpha_j \neq 0$ no processo de retorno em direção á raiz da árvore, é feito para este valor uma diminuição em uma unidade do seu valor. Além do anterior, é desenvolvido um processo de devoluções associado com a redução do α_n, como é a devolução de espaço na mochila, de espaço no compartimento associado ao nó n, devolução em uma unidade da demanda disponível, aumento de uma faca para uso no segundo processo de corte e em caso de ficar sem itens o compartimento, este é fechado. Este

processo veja-se descrito no Algoritmo 20.

Algoritmo 18: Passo 4, processo de volta ou *Backtrack*.

Faça o Algoritmo 16; Faça o Algoritmo 14;
se $\alpha_r = 0$ **então**
  $r- = 1$; Faça o Algoritmo 17;
  se $\alpha_r = 0$ **então**
    Faça o Algoritmo 15; Faça o Algoritmo 18
  fim
senão
  se $comp = 1$ **então**
    Passo 5
  fim
  se $fecha\_comp(comp - 1) = r$ **então**
    $comp- = 1$
  fim
  se $r + 1 = abre\_classe(classe)$ **então**
    $classe- = 1$
  fim
  Passo 5
fim

Fonte: Próprio autor.

Algoritmo 19: Passo 5: Ramificação.

Passo 5 *(Poda)*:
se α_0 **então**
  Passo 4
senão
  $(i, j, k) = ordem(r)$; $\alpha_r- = 1$; $Lmoch+ = l_i$
  $Lcomp_j^k+ = l_i$; $D_i+ = 1$; $F2_j^k+ = 1$;
  se $comp > 1$ **então**
    se $\alpha_n = 0\ \forall n = fecha\_comp(comp - 1) + 1, ..., fecha\_comp(comp)$ *e* $\delta_j^k = 1$
    então
      $\delta_j^k = 0$
      $F1+ = 1$
    fim
  fim
  Passo 6
fim

Fonte: Próprio autor.

Como passo final para a construção da árvore de enumeração é avaliar se para o nó em questão, o nó n, é válido desenvolver um novo ramo ou o processo iterativo pode fechar o nó para reiniciar o processo de volta até a raiz da árvore. Esta avaliação para desenvolver a nova ramificação ou poda é feito pelo limitante superior ou *bounding* estabelecido com os valores já desenvolvidos na árvore, na qual, este será o tema de discussão na seguente seção. Então, define-se o Algoritmo 21 como o critério de poda da árvore de enumeração

Algoritmo 20: Passo 6, critério de poda ou *Bounding* para o PMCR

se *Bounding*>z **então**
| **Passo 1**
senão
| **Passo 5**
fim

Fonte: Próprio autor.

Como resultado tem-se que os algoritmos (6)-(21) definem o processo de enumeração sistemática das soluções viáveis do PMCR, gerando assim a árvore de enumeração para o problema.

5.2 DEFINIÇÃO DO *Bounding* OU LIMITANTE SUPERIOR PARA O PMCR.

Já com a proposta do gerador da árvore para o PMCR, procura-se "bons" limitantes superiores que permitam acelerar (aumentar a quantidade de podas de ramos) no algoritmo. Fala-se de que um limitante superior M é bom no sentido de que ao comparar com a solução ótima z^* para o PMCR ele esteja muito próximo ($|M - z^*| \leq \epsilon, \epsilon \in \mathbb{R}$).

Inicia-se dita procura em duas direções: a primeira, vão-se apresentar limitantes superiores seguindo a estrutura do *Branch and Bound* para o Problema da Mochila Restrita-Irrestrita apresentada no Capítulo 3 (veja-se o limitante formulado em 3.11). Em seguida, gera-se limitantes superiores usando como estratégia a técnica da relaxação Lagrangeana do problema por meio da manipulação das restrições consideradas como restrições "complicadora" (aquelas restrições que aumentam a nível de complexidade do problema).

Veja-se a continuação o razonamento detrais da busca dos limitantes superiores para o PMCR. Para certo ramo arbitrário com solução corrente $\hat{z}$, o indicador r está na fase de retorno em direção à raiz da árvore até o primeiro nó com $\hat{\alpha}_r \neq 0$. Suponha-se que o nó que cumpre a anterior condição é o nó k onde, os nós de 1 até o nó k tem valores já calculados como $\hat{\alpha}_1$, $\hat{\alpha}_2$,...,$\hat{\alpha}_{k-1}$ e $\hat{\alpha}_k$. Em seguida, para o valor de α_k faça-se uma diminuição de uma unidade obtendo um novo valor para o nó k como $\hat{\alpha}_k - 1$.

Com o novo valor $\hat{\alpha}_k$, pretende-se avaliar se é conveniente ramificar desde o nó k até o nó M (veja-se uma representação da ramificação do nó k na Figura 5.3). Caso de ser ramificado, o novo ramo vão corresponder com os mesmo valores dos nos anteriores a k com $\hat{\alpha}_1$, $\hat{\alpha}_2$,..., $\hat{\alpha}_{k-1}$

e a partir de $k+1$ com novos valores $\hat{\alpha}_k - 1$, $\bar{\alpha}_{k+1}$, $\bar{\alpha}_{k+2}$, ...,$\bar{\alpha}_{M-1}$, $\bar{\alpha}_M$, com solução corrente (suponha-se conhecidos os valores dos nós seguentes a $k+1$) $\bar{z}_1$. Em seguida, pretende-se estimar o valor de $\bar{z}_1$ usando só os valores conhecidos até o nó k, ou seja $\hat{\alpha}_1$, $\hat{\alpha}_2$,..., $\hat{\alpha}_{k-1}$. Com a estimação de $\bar{z}_1$ pode-se comparar com a solução corrente $\hat{z}$. Em dita comparação vão-se decidir se o nó k será ramificado no caso sim se tem possibilidades de melhorar a solução corrente, ou em caso de ter que não terá melhoria, podar (fechar) o nó.

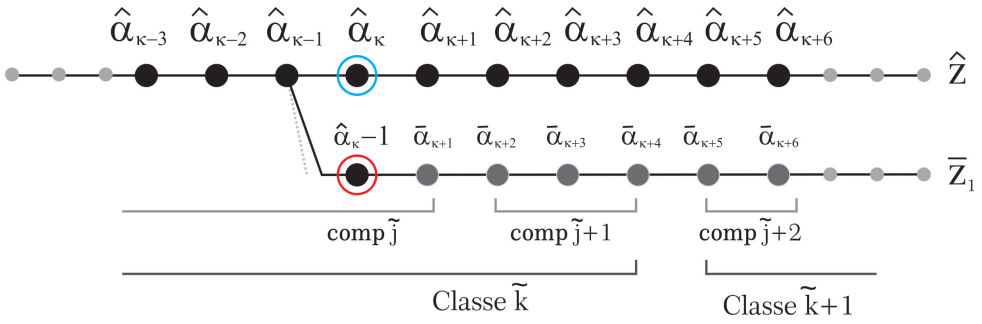

Figura 5.3: Representação da ramificação do nó k na árvore de enumeração do PMCR
Fonte: Próprio autor.

Estimar o valor $\bar{z}_1$ pode não ser uma tarefa muito simples, mas pode-se planejar estrategias para obter valores razoavelmente perto desse valor, principalmente maiores a este (no caso deste problema de maximização) e assim ter como comparar com a solução corrente $\hat{z}$. Então, seguindo o anterior, procura-se um limitantes superior $M_1 \geq \bar{z}_1$, onde se $M_1 \leq \hat{z}$ o nó k é fechado, no caso contrário, se $M_1 \geq \hat{z}$ o nó é ramificado.

Agora suponha-se que o nó k com o valor de $\hat{\alpha}_k - 1$ não foi ramificado por causa que com o limitante superior calculado M_1 foi obtido que $M_1 \geq \bar{z}_1$, pelo anterior tem-se a pretensão de fazer poda do nó mas observe-se o seguente: pode-se continuar estudando a factibilidade do valor do nó k fazendo uma diminuição mais uma novamente no valor atual do nó, ou seja, tendo $\hat{\alpha}_k - 2$, desta forma outros ramos podem serem obtidos sem se continua diminuindo em una unidade o valor do nó k, ou seja, fazendo $\hat{\alpha}_k - 2$, $\hat{\alpha}_k - 3$, ..., $\hat{\alpha}_k - \hat{\alpha}_k$, com solução corrente respectivamente $\bar{z}_2$, $\bar{z}_3$, ..., $\bar{z}_{\hat{k}}$. Pelo anterior também novos limitantes superiores podem serem definidos respectivamente a estes novos valores do nó k com M_2, M_3, ..., $M_{\hat{k}}$. Espera-se que ocorra que entre os limitantes superiores que sejam obtidos para o PMCR tenha a propriedade de que $M_1 \geq M_2 \geq ... \geq M_{\hat{k}}$, assim, com só ter $M_1 \leq \hat{z}$ poda-se fechar o nó completamente independente dos próximo valores que possa ter $\hat{\alpha}_k$ e em seguida iniciar novamente o processo de volta atrais pelo ramo da árvore.

5.2.1 Limitantes Superior tipo *Backtracking* para o PMCR.

Faça-se a continuação um estudo análogo feito ao Problema da Mochila Irrestrita-Restrita na formulação dos limitantes superiores para o PMCR (veja-se o Capítulo 3). Então, seguindo

os comentários do inicio da seção, busca-se estimar em primeira instancia o valor de $\bar{z}_1$. Para isto, observe-se a estrutura do $\bar{z}_1$:

$$\bar{z}_1 = \sum_{i=1}^{M} \tilde{u}_i \alpha_i = \sum_{i=1}^{k} \tilde{u}_i \hat{\alpha}_i + \sum_{i=k+1}^{M} \tilde{u}_i \bar{\alpha}_i = \underbrace{\sum_{i=1}^{k-1} \tilde{u}_i \hat{\alpha}_i + \tilde{u}_k(\hat{\alpha_k} - 1)}_{\text{Valores Conhecidos}} + \underbrace{\sum_{i=k+1}^{M} \tilde{u}_i \bar{\alpha}_i}_{\text{Valores Desconhecidos}} \tag{5.1}$$

Pela ordenação e indexação global das variáveis, tem-se que $\tilde{u}_r$, com $r = 1, ..., M$, é a utilidade correspondente as informações do nó r (posição atual do indicador) que estão associada com α_r (lembre-se que cada α_r tem associado uma variável de decisão a_{ij}^k, veja-se os comentário feitos no inicio do capítulo). O mesmos casos ocorram com as larguras $\tilde{l}_r$. O valor que se quer estimar é $\sum_{i=k+1}^{M} \tilde{u}_i \bar{\alpha}_i$. Para estimar dito valor, precisa-se ter em consideração do problema a restrição da capacidade da mochila, pela qual veja-se esta a continuação:

$$\sum_{i=1}^{k-1} \tilde{l}_i \hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1) + \sum_{i=k+1}^{M} \tilde{l}_i \bar{\alpha}_i \leq L, \text{ restrição da capacidade da mochila.} \tag{5.2}$$

Usa-se a restrição da capacidade da mochila para detalhar os valores involucrados, como segue acontinuação:

$$\begin{aligned} \sum_{i=1}^{k-1} \tilde{l}_i \hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1) + \sum_{i=k+1}^{M} \tilde{l}_i \bar{\alpha}_i \leq L \Rightarrow \sum_{i=k+1}^{M} \tilde{l}_i \bar{\alpha}_i &\leq L - \left(\sum_{i=1}^{k-1} \tilde{l}_i \hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - 1) \right) \\ \Rightarrow \sum_{i=k+1}^{M} \frac{\tilde{l}_i}{\tilde{u}_i} \tilde{u}_i \bar{\alpha}_i &\leq L - \left(\sum_{i=1}^{k-1} \tilde{l}_i \hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1) \right) \end{aligned} \tag{5.3}$$

Na equação 5.3 pretende-se retirar a fração $\frac{\tilde{l}_i}{\tilde{u}_i}$ para assim conseguir estimar a parte desconhecida do novo ramo gerado. Uma forma de realizar o anterior, novamente seguindo as ideias feita no PMR (PMI), é aproveitando-se da ordenação dos itens e das classes como foi apresentada no Algoritmo 6 para o PMCR. Então, calcule-se a menor relação $\frac{\tilde{l}_i}{\tilde{u}_i}$ assim: $min\left(\frac{\tilde{l}_i}{\tilde{u}_i}, i \in \{k+1, ..., M\}\right)$.

Antes de continuar, veja-se a seguinte observação: pela ordenação feita nas classes e nos itens para o PMCR, o compartimento onde está posicionado o nó k, tem-se que $\frac{\tilde{l}_{k+1}}{\tilde{u}_{k+1}} \leq \frac{\tilde{l}_{k+2}}{\tilde{u}_{k+2}} \leq ... \leq \frac{\tilde{l}_{f\_c(k)}}{\tilde{u}_{f\_c(k)}}$ (a expressão $f\_c(k)$ faz referencia ao último nó que faz parte do compartimento associado ao nó k, por exemplo, tem-se da Figura 5.3 que $f\_c(k) = k + 1$). Observe-se que a classe $\tilde{k}$ (classe associada ao nó k) tem um melhor média de eficiências em relação a seguintes classes $\tilde{k} + 1$, $\tilde{k} + 2$, ..., $\tilde{k} + q$, mas esta condição não garante que no último

nó $f\_c(k)$ do compartimento onde pertence k esteja a menor relação $\frac{\tilde{l}_i}{\tilde{u}_i}$ do resto dos itens dos nós $k+1$ até M. Do anterior, tem-se só que $min\left(\frac{\tilde{l}_i}{\tilde{u}_i}, i \in \{k+1, ..., f\_c(k)\}\right) = \frac{\tilde{l}_{k+1}}{\tilde{u}_{k+1}}$. Então, tomando a referencia a posição de k, tome-se das próximas classes $\bar{k}+1, \bar{k}+2, ..., q$ a menor relação $\frac{l_i}{u_i}$, sendo representada respectivamente $\frac{\tilde{\tilde{l}}^{\bar{k}+1}}{\tilde{\tilde{u}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{l}}^{q}}{\tilde{\tilde{u}}^{q}}$ e a continuação redefina-se o mínimo $min\left(\frac{\tilde{l}_i}{\tilde{u}_i}, i \in \{k+1, ..., M\}\right)$ pelo mínimo dessas relações por:

$$min\left(\frac{\tilde{l}_{k+1}}{\tilde{u}_{k+1}}, \frac{\tilde{\tilde{l}}^{\bar{k}+1}}{\tilde{\tilde{u}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{l}}^{q}}{\tilde{\tilde{u}}^{q}}\right) \text{ onde } min\left(\frac{\tilde{l}_{k+1}}{\tilde{u}_{k+1}}, \frac{\tilde{\tilde{l}}^{\bar{k}+1}}{\tilde{\tilde{u}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{l}}^{q}}{\tilde{\tilde{u}}^{q}}\right) \leq \frac{\tilde{l}_{k+1}}{\tilde{u}_{k+1}} \tag{5.4}$$

Em seguida, com a expressão (5.3) e a relação obtida em (5.4) segue que:

$$\begin{aligned}
&\sum_{i=k+1}^{M} \frac{\tilde{l}_i}{\tilde{u}_i}\tilde{u}_i\bar{\alpha}_i \leq L - \left(\sum_{i=1}^{k-1}\tilde{l}_i\hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1)\right) \\
&\Rightarrow \sum_{i=k+1}^{M} min\left(\frac{\tilde{l}_{k+1}}{\tilde{u}_{k+1}}, \frac{\tilde{\tilde{l}}^{\bar{k}+1}}{\tilde{\tilde{u}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{l}}^{q}}{\tilde{\tilde{u}}^{q}}\right)\tilde{u}_i\bar{\alpha}_i \leq \sum_{i=k+1}^{M} \frac{\tilde{l}_i}{\tilde{u}_i}\tilde{u}_i\bar{\alpha}_i \leq L - \left(\sum_{i=1}^{k-1}\tilde{l}_i\hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1)\right) \\
&\Rightarrow min\left(\frac{\tilde{l}_{k+1}}{\tilde{u}_{k+1}}, \frac{\tilde{\tilde{l}}^{\bar{k}+1}}{\tilde{\tilde{u}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{l}}^{q}}{\tilde{\tilde{u}}^{q}}\right)\sum_{i=k+1}^{M}\tilde{u}_i\bar{\alpha}_i \leq L - \left(\sum_{i=1}^{k-1}\tilde{l}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - 1)\right) \\
&\Rightarrow \sum_{i=k+1}^{M}\tilde{u}_i\bar{\alpha}_i \leq max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\bar{k}+1}}{\tilde{\tilde{l}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right)\cdot\left(L - \left(\sum_{i=1}^{k-1}\tilde{l}_i\hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1)\right)\right) \\
&\Rightarrow \bar{z}_1 \leq \sum_{i=1}^{k-1}\tilde{u}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - 1) + max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\bar{k}+1}}{\tilde{\tilde{l}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right)\cdot\left(L - \left(\sum_{i=1}^{k-1}\tilde{l}_i\hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1)\right)\right)
\end{aligned} \tag{5.5}$$

De (5.5) resuma-se a relação obtida como:

$$\bar{z}_1 \leq \sum_{i=1}^{k-1}\tilde{u}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - 1) + max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\bar{k}+1}}{\tilde{\tilde{l}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right)\cdot\left(L - \left(\sum_{i=1}^{k-1}\tilde{l}_i\hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1)\right)\right) \tag{5.6}$$

Veja-se em (5.6) a formulação de um limitante superior, tendo como principal característica que todos os valores tratados no limitantes são valores conhecidos. Desta forma, defina-se como o limitante superior para o valor a estimar $\bar{z}_1$ como $M_1 = \sum_{i=1}^{k-1}\tilde{u}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - 1) + max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\bar{k}+1}}{\tilde{\tilde{l}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right)\cdot\left(L - \left(\sum_{i=1}^{k-1}\tilde{l}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - 1)\right)\right)$. Seguindo a ideia de continuar reduzindo o valor de α_k em $\alpha_k - 2$, $\alpha_k - 3$,...,$\alpha_k - \alpha_k$, tem-se de forma análoga limitantes superiores M_r, com $r = 2, 3, ..., \hat{\alpha}_k$ para limitar $\bar{z}_r$, com $r = 2, 3, ..., \hat{\alpha}_k$, definindo $M_r = \sum_{i=1}^{k-1}\tilde{u}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - r) + max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\bar{k}+1}}{\tilde{\tilde{l}}^{\bar{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right)\cdot\left(L - \left(\sum_{i=1}^{k-1}\tilde{l}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - r)\right)\right)$. Como foi discutido no inicio da seção, verifique-se com os limitantes superiores formulados sim se cumpre a relação $M_1 \geq M_2 \geq ..., M_{\hat{\alpha}_k}$ para ter garantia de ser um limitante eficiente no processo de poda na árvore de numeração. Então, vara verificar o anterior, estude-se a relação que se tem da diferença entre o limitante associado ao nó r e o próximo limitante do nó $r+1$, como segue a continuação:

$$M_r - M_{r+1} =$$
$$\sum_{i=1}^{k-1} \tilde{u}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - r) + max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\tilde{k}+1}}{\tilde{\tilde{l}}^{\tilde{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right) \cdot \left(L - \left(\sum_{i=1}^{k-1} \tilde{l}_i\hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - r)\right)\right) -$$
$$\sum_{i=1}^{k-1} \tilde{u}_i\hat{\alpha}_i - \tilde{u}_k(\hat{\alpha}_k - r - 1) - max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\tilde{k}+1}}{\tilde{\tilde{l}}^{\tilde{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right) \cdot \left(L - \left(\sum_{i=1}^{k-1} \tilde{l}_i\hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - r - 1)\right)\right) =$$
$$\tilde{u}_k - max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\tilde{k}+1}}{\tilde{\tilde{l}}^{\tilde{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right) \cdot \tilde{l}_k$$

Temos que $M_r - M_{r+1} = \tilde{u}_k - max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\tilde{k}+1}}{\tilde{\tilde{l}}^{\tilde{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right) \cdot \tilde{l}_k$, onde $M_r \geq M_{r+1}$ se e somente se $\tilde{u}_k - max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\tilde{k}+1}}{\tilde{\tilde{l}}^{\tilde{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right) \cdot \tilde{l}_k \geq 0$, isto último pode-se ver como $\frac{\tilde{u}_k}{\tilde{l}_k} \geq max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\tilde{k}+1}}{\tilde{\tilde{l}}^{\tilde{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right)$.

Observe-se que não se pode garantir a relação anterior já que existe a possibilidade que em outra classe exista uma melhor eficiência em relação á eficiência associada ao nó k, todo devido à forma como foi ordenado inicialmente as informações do problema. Esta questão faz que o limitante formulado para certos momentos no processo iterativo na árvore não seja eficiente, mas, certamente cumpre com a condição de fazer podas aos ramos.

Aceitando este possível comportamento na árvore, defina-se este como o primeiro limitante superior formulado para o PMCR, na qual será chamado de *limitante superior tipo Backtracking*, sendo formalizado a continuação como:

$$M = \sum_{i=1}^{k-1} \tilde{u}_i\hat{\alpha}_i + \tilde{u}_k(\hat{\alpha}_k - 1) + max\left(\frac{\tilde{u}_{k+1}}{\tilde{l}_{k+1}}, \frac{\tilde{\tilde{u}}^{\tilde{k}+1}}{\tilde{\tilde{l}}^{\tilde{k}+1}}, \ldots, \frac{\tilde{\tilde{u}}^{q}}{\tilde{\tilde{l}}^{q}}\right) \cdot \left(L - \left(\sum_{i=1}^{k-1} \tilde{l}_i\hat{\alpha}_i + \tilde{l}_k(\hat{\alpha}_k - 1)\right)\right) \tag{5.7}$$

Com a expressão (5.7) redefine-se o passo 6 do algoritmo da árvore de enumeração como:

Algoritmo 21: Passo 6: *Bounding* como limitante superior tipo *Bactracking*

Passo 6 *(Poda)*:

se M>z **então**

| **Passo 1** Faça o Algoritmo 7

senão

| **Passo 5** Faça o Algoritmo 20

fim

Fonte: Próprio autor.

5.2.2 Limitantes Superiores via Relaxação Lagrangeana ao PMCR.

Nesta subseção apresenta-se o processo de geração de limitantes superiores por meio da técnica da Relaxação Lagrangeana aplicado ao Problema da Mochila Compartimentada. Este tipo de técnica é usado em diversos trabalhos em Programação Inteira para formular limitantes superiores (ou limitantes inferiores, dependendo do tipo de problema inteiro a resolver) e em

seguida é usado como critério de poda para ramos da árvore de enumeração (veja-sem alguns exemplos em [3], [9] e [12]). A escolha para o estudo das relaxações Lagrangeanas para o PMCR foi seguindo a sugestões do relatório técnico de [14].

Esta subseção vão estar dividida em duas partes: a primeira parte contem uma revisão teórica da relaxação Lagrangeana, passando pela definição do problema Dual Lagrangeano (problema na qual que vão fornecer os limitantes) e revisão do Método do Subgradiente como estratégia para solucionar-lo. Em seguida, na segunda parte, faz-se um estudo técnico da relaxação Lagrangeana para uma das restrições do PMCR, tendo assim, uma estratégia geral para sua abordagem para qualquer restrição do PMCR.

A relaxação Lagrangeana

Os elementos teóricos apresentados a continuação sob a Relaxação Lagrangeana estão baseados nos trabalhos de [9] e [12]. Define-se o problema (P), sem perda de generalidade, como:

$$\begin{aligned} \text{(P) Maximizar: } & f(x) \\ \text{s.a.: } & Ax \leq b \\ & Cx \leq d \\ & x \in X \end{aligned} \tag{5.8}$$

Sendo X a região viável, x é um vetor das variáveis de dimensão $n \times 1$, b é um vetor de dimensão $m \times 1$, d é um vetor de dimensão $k \times 1$ e as matrizes A e C tem dimensões definidas conforme ao problema. Assuma-se que as restrições do (P) foram particionados em dois conjuntos, $Ax \leq b$ e $Cx \leq d$, onde, o problema (P) é relativamente "fácil" de se resolver se o conjunto de restrições $Ax \leq b$ é removido. Para criar o Problema Lagrangeano, primeiro define-se um vetor não negativo de multiplicadores $\lambda \in \mathbb{R}^m$, qual será chamados de *multiplicadores Lagrangeanos*. Define-se uma *relaxação Lagrangeana* ao problema (P) como segue:

Definição 5.3 (Relaxação Lagrangeana)**.** A *relaxação Lagrangeana* do problema (P) sob as restrições $Ax \leq b$, com o não negativo multiplicador Lagrangeano λ, define-se como o problema:

$$\begin{aligned} \text{(RLAGP}_\lambda\text{) Maximizar: } & f(x) + \lambda(b - Ax) \\ & Cx \leq d \\ & x \in X \end{aligned} \tag{5.9}$$

Observe-se que a restrição $Ax \leq b$ foi adicionada na função objetivo com pesos λ e esta restrição foi removida do problema. Fala-se que as restrições $Ax \leq b$ foram *dualizadas*. A continuação algumas notações: seja o problema de otimização (P), o conjunto de soluções factíveis vão-se denotar por SF(P), o conjunto de soluções factíveis do problema (P) como SO(P), o valor ótimo do problema (P) como *v*(P). O Problema (RLAGP$_\lambda$) é uma relaxação do (P) desde que satisfaz o seguinte:

- SF(P) $\subset SF$(RLAGP$_\lambda$) e
- Para todo $x \in$ SF(P), $f(x) + \lambda(b - Ax) \geq f(x)$

Sim se tem que (RLAGP$_\lambda$) é uma relaxação do (P), tem-se que v(RLAGP$_\lambda$) $\geq v$(P), para todo $\lambda \geq 0$. O anterior implica que, para qualquer $\lambda \geq 0$, o valor ótimo de v(RLAGP$_\lambda$) é um limite superior do valor ótimo do problema (P), v(P). Agora, o interesse dos limitantes superiores é achar aquele o "mais" perto do valor ótimo do problema (P), qual define o seguinte problema de otimização:

Definição 5.4 (Problema Dual Lagrangeano)**.** O menor limitante superior Lagrangeano do v(P) e definido pelo problema:

$$(\text{RLAGP}) \underset{\lambda \geq 0}{min}\, v(\text{RLAGP}_\lambda) \tag{5.10}$$

Na Figura 5.4 representa-se o dual Lagrangeano em relação a qualquer vetor de multiplicadores Lagrangeanos $s, t \in \{1, 2, ...\}$.

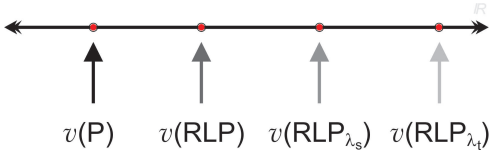

Figura 5.4: Representação do dual Lagrangeano para o poblema P
Fonte: Próprio autor.

O problema (RLAGP) é um problema no espaço dual dos multiplicadores Lagrangeanos, enquanto (LR$_\lambda$) é um problema em x. Seja $x(\lambda)$ a solução ótima ao problema RLAGP$_\lambda$ para certo $\lambda \geq 0$. Chame-se $x(\lambda)$ uma *solução Lagrangeana*. Observe-se que $x(\lambda)$ para certo $\lambda \geq 0$ é uma solução ótima para o problema (P) quando ocorre que $\lambda(b - AX) = 0$, em outras palavras, $f(x(\lambda)) + \lambda(b - AX) = 0 = v$(P) só se v(P) $= f(x(\lambda))$, ou seja $x(\lambda)$ é uma solução ótima do problema (P). Com as observações anteriores a Proposição 5.5 (para maiores detalhes em [12]).

Proposição 5.5. *1. Se $x(\lambda)$ é a solução ótima do problema (RLAGP$_\lambda$) para certo $\lambda \geq 0$, então*

$$f(x(\lambda)) + \lambda(b - Ax(\lambda)) \geq v(P)$$

2. *Se adicionalmente $x(\lambda)$ é uma solução viável para o problema (P), então*

$$f(x(\lambda)) \leq v(P) \leq f(x(\lambda)) + \lambda(b - Ax(\lambda))$$

3. *Se adicionalmente $\lambda(b - Ax(\lambda)) = 0$, então $x(\lambda)$ é a solução do problema (P) e*

$$v(P) = f(x(\lambda))$$

Demonstração. Veja-se em [12]. □

Define-se como *função Lagrangeana* à expressão $z(\lambda) = v(\text{RLAGP}_\lambda) = f(x^*) + \lambda(b - Ax^*)$, onde $z(\lambda)$ é uma função implícita de λ e x^* é o valor ótimo de $v(\text{RLAGP}_\lambda)$ para λ . Seja o conjunto $\{x \in X | Cx \leq d\} = \{x^1, x^2, \ldots, x^K\}$, a expressão (5.9) pode-se reescrever (5.9) como:

$$(\text{RLAGP}_\lambda) = \underset{k=1,\ldots,K}{Max}\{f(x^k) + \lambda(b - Ax^k)\} \tag{5.11}$$

Com a expressão (5.11), cada λ da função Lagrangeana $z(\lambda)$ define uma função linear de λ. Então, a função Lagrangeana $z(\lambda)$ gera um *envelope superior* de famílias de funções lineares de λ, sendo $z(\lambda)$ uma função convexa.

Proposição 5.6. *A função $z(\lambda)$ é uma função convexa.*

Demonstração. Sejam $\lambda, \eta \in \mathbb{R}^m$ arbitrários, onde $\lambda, \eta \geq 0$. Então, para todo $t \in (0, 1)$ tem-se que:

$$\begin{aligned}
&z(t\lambda + (1-t)\eta) = v(\text{RLAGP}_{t\lambda+(1-t)\eta}) = \\
&f(x^*) + (t\lambda + (1-t)\eta)(b - Ax^*) \leq \\
&f(x^*) + t\lambda(b - Ax^*) + f(x^*) + (1-t)\eta(b - Ax^*) = \\
&z(t\lambda) + z((1-t)\eta), \text{ como } z(\lambda) \text{é uma função linear para todo} \lambda \in \mathbb{R}^m \\
&\text{tem-se que } z(t\lambda) + z((1-t)\eta) = tz(\lambda) + (1-t)z(\eta), \text{ então} \\
&z(t\lambda + (1-t)\eta) \leq tz(\lambda) + (1-t)z(\eta)
\end{aligned}$$

□

De forma análoga, com $\{x \in X | Cx \leq d\} = \{x^1, x^2, \ldots, x^K\}$, a expressão (5.10) pode-se reescrever como:

$$
\begin{aligned}
(\text{RLAGP}) \min_{\lambda \geq 0} v(\text{RLAGP}_\lambda) &= \min_{\lambda \geq 0} z(\lambda) = \\
&\min_{\lambda \geq 0} \max_{k=1,\ldots,K} \{f(x^k + \lambda(b - Ax^k)\} = \\
&\min_{\lambda \geq 0, \eta} \{\eta | \eta \geq f(x^k + \lambda(b - Ax^k), k = 1, \ldots, K\}
\end{aligned} \tag{5.12}
$$

Observe-se que a função convexa $z(\lambda)$ define uma função por trechos para todo $\lambda \in \mathbb{R}$. Sendo assim, $v(\text{RLAGP})$ é o mínimo de dita função mas só conhecida de forma implícita. Por ser $z(\lambda)$, uma função definida por trechos de funções lineares, $z(\lambda)$ possui pontos na qual não é diferenciável. A Figura 5.5 representa à função convexa $z(\lambda)$.

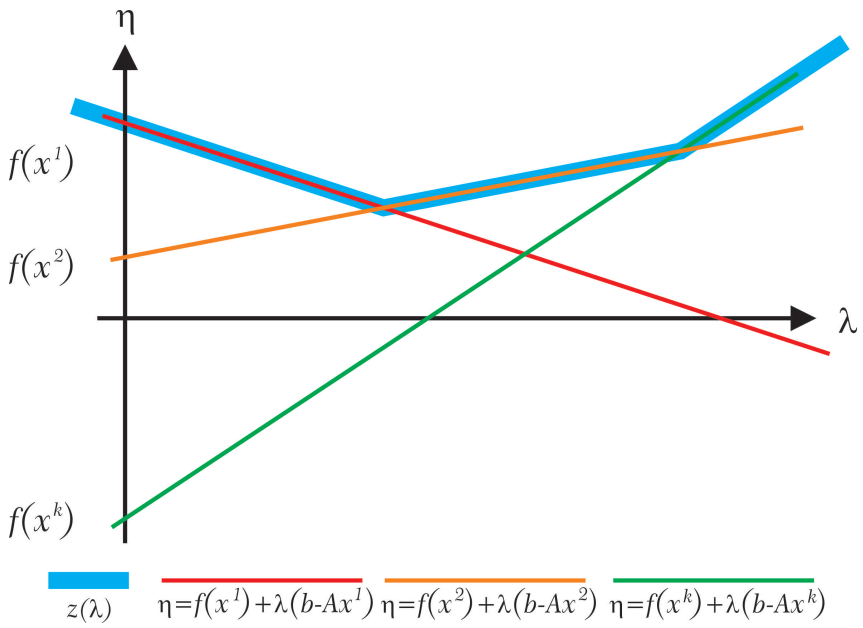

Figura 5.5: Representação da função convexa $z(\lambda)$.
Fonte: Próprio autor com referência em [12].

Em seguida define-se o subgradiente de uma função convexa com o objetivo de procurar uma caracterização de $z(\lambda)$.

Definição 5.7 (Subgradiente de uma função convexa)**.** Seja $f : \mathbb{R}^n \to \mathbb{R}$ uma função convexa e seja o ponto $\bar{x} \in dom\{f\}$. O vetor $y \in \mathbb{R}^n$ é chamado de *subgradiente* da função convexa f no ponto $\bar{x} \in \mathbb{R}^n$ se para todo $x \in \mathbb{R}^n$

$$
f(x) - f(\bar{x}) \geq \langle y, x - \bar{x} \rangle = y(x - \bar{x}) \tag{5.13}
$$

Definição 5.8 (Subdiferencial de uma função)**.** O conjunto de todos os subgradientes da função convexa f no ponto $\bar{x}$ é chamado do *subdiferencial* de f no ponto $\bar{x}$ e, este é denotado por $\partial f(\bar{x})$. Em outras palavras:

$$\partial f(\bar{x}) = \{y \in \mathbb{R}^n | f(x) - f(\bar{x}) \geq y(x - \bar{x}), \text{ para todo } x \in \mathbb{R}^n\} \tag{5.14}$$

Proposição 5.9. *O subdiferencial $\partial f(\bar{x})$ da função convexa f no ponto $\bar{x}$ é um conjunto não vazio, fechado, convexo e limitado.*

Demonstração. Note que $\partial f(\bar{x})$ é convexo, pois sejam $\bar{x} \in dom\{f\}$, $v, w \in \partial f(\bar{x})$ arbitrários e $t \in (0, 1)$. De fato, tem-se que:

$(vt + (1-t)w)(x - \bar{x}) = vt(x - \bar{x}) + (1-t)w(x - \bar{x}) = vt(x - \bar{x}) + w(x - \bar{x}) - wt(x - \bar{x}) \leq$
$t(f(x) - f(\bar{x})) + (f(x) - f(\bar{x})) - t(f(x) - f(\bar{x})) = (f(x) - f(\bar{x}))$.
Então, tem-se que para todo $v, w \in \partial f(\bar{x})$ tem-se $vt + (1-t)w \in \partial f(\bar{x})$

Agora, observe-e para todo $y \in \mathbb{R}^n$ gradientes de f no ponto $\bar{x} \in \mathbb{R}^n$ define semiespaços fechados $D_y = \{y \in \mathbb{R}^n \mid f(x) - f(\bar{x}) \geq y(x - \bar{x})\}$, onde $\partial f(\bar{x}) = \bigcap_{y \in \mathbb{R}^n} D_y$, tal que define a fechadura de $\partial f(\bar{x})$. Em [31] páginas 8-10 pode-se ver a demostração de $\partial f(\bar{x})$ é limitado. □

Observe-se que se, o subdiferencial da função convexa f consiste em um único elemento, este elemento corresponde ao gradiente da função f no ponto $\bar{x}$, denotado por $\nabla f(\bar{x})$. Volte-se á função $z(\lambda)$. Seja $\lambda^* \in \mathbb{R}^m$ quem é o minimizador de $z(\lambda)$ e seja $\eta^* = z(\lambda^*)$. Seja λ^k a estimação corrente de λ^*. Agora seja $\eta^k = z(\lambda^k)$ e define-se o hiperplano que passa através de λ^k como

$$H_k = \{\lambda \mid f(x^k) + \lambda(b - Ax^k) = \eta^k\} \tag{5.15}$$

Tem-se dois casos em consideração em relação da diferenciabilidade da função $z(\lambda)$ no ponto λ^k:

- Se $z(\lambda)$ é diferenciável no ponto λ^k, ou seja, se RL$_\lambda$ tem uma única solução ótima x^k, o vetor gradiente $\nabla z(\lambda^k)$ no ponto λ^k é:

$$\nabla z(\lambda^k) = (b - Ax^k) \tag{5.16}$$

  Observação: o vetor $\nabla z(\lambda^k)$ é ortogonal a H_k.

- Se $z(\lambda)$ não é diferenciável no ponto λ^k, ou seja, RL$_\lambda$ tem mais de uma solução ótima, tem-se que o vetor subgradiente da função $z(\lambda)$ no ponto λ^k é:

$$s_k = (b - Ax^k) \tag{5.17}$$

Observação: o vetor $s_k = (b - Ax^k)$ é ortogonal a H_k.

Proposição 5.10. *O vetor $s_k = (b - Ax^k)$ é o subgradiente da função $z(\lambda)$ no ponto λ^k.*

Demonstração. Sejam $\lambda, \eta \in \mathbb{R}^m$, multiplicadores Lagrangeanos da função z, define-se os problemas Lagrangeanos:

$$(\text{RLAGP}_\lambda) = \underset{x}{Max}\{f(x) + \lambda(b - Ax) \mid Cx \leq d, x \in X\}$$
$$(\text{RLAGP}_\eta) = \underset{x}{Max}\{f(x) + \eta(b - Ax) \mid Cx \leq d, x \in X\}$$
$$\text{Com valores ótimos } (\text{RLAGP}_\lambda) = z(\lambda) \text{ e } (\text{RLAGP}_\eta) = z(\eta).$$

Então, tem-se que: $(b - Ax)(\lambda - \eta) = \lambda(b - Ax) - \eta(b - Ax) + f(x) - f(x) = f(x) + \lambda(b - Ax) - (f(x) + \eta(b - Ax)) \geq z(\lambda) - z(\eta)$.

□

Com a definição do vetor subgradiente s_k para a função $z(\lambda)$ no ponto λ^k tem-se as condições para estudar a resolução do problema Lagrangeano dual para o PMCR, mediando o uso do método do subgradiente, que é descrito a continuação.

Resolução do Problema Dual Lagrangeano: Método do subgradiente

Para a resolução do problema Lagrangeano dual tem-se escolhido o Método do Subgradiente para Funções Convexas (MSFC) seguindo os trabalhos sobre resolução do Problema Dual Lagrangeano para problemas de programação inteira de [9], [3] e [12]. O MSFC é um método interativo na qual tamanhos de passos (quem vão determinar o comportamento da relaxação) são determinados ao longo do sentido negativo do subgradiente da função $z(\lambda)$.

Seja a iteração k com vetor Lagrangeano corrente λ^k e x^k como solução ótima de (RLP_{λ^k}). Tem-se o vetor subgradiente $s_k = (b - Ax^k)$ da função $z(\lambda)$ no ponto λ^k. Considere-se λ^* como solução ótima ao problema dual (RLAGP) com $\eta^* = z(\lambda^*)$, seja $\widetilde{\lambda}^{k+1}$ a projeção de λ^k no hiperplano H^*, definido por:

$$H^* = \{\lambda \mid f(x^k) + \lambda(b - Ax^k) = \eta^*\} \tag{5.18}$$

$H^* = \{\lambda \mid f(x^k) + \lambda(b - Ax^k) = \eta^*\}$. Observe-se que o hiperplano H^* e paralelo ao hiperplano H^k, também o hiperplano H^* passa pelo ponto ótimo λ^* se $f(x^k) + \lambda^*(b - Ax^k) =$

η^*. O vetor s_k é perpendicular aos hiperplanos H^k e H^*, e, pela projeção de λ^*, tem-se que o vetor $\widetilde{\lambda}^{k+1} - \lambda^k$ é um múltiplo negativo de s_k, ou seja:

$$\widetilde{\lambda}^{k+1} - \lambda^k = -\mu s_k \tag{5.19}$$

A Como $\widetilde{\lambda}^{k+1} \in H^*$ tem-se que:

$$f(x^k) + \widetilde{\lambda}^{k+1}(b - Ax^k) = \eta^* \tag{5.20}$$

Assim, por (5.19) e (5.20) tem-se que:

$$f(x^k) + \widetilde{\lambda}^{k+1}(b - Ax^k) = f(x^k) + (-\mu s_k + \lambda^k)(b - Ax^k) = f(x^k) + \lambda^k(b - Ax^k) - \mu s_k s_k \tag{5.21}$$

Agora, pela definição (5.15) e com (5.21) tem-se que $f(x^k) + \lambda^k(b - Ax^k) - \mu s_k s_k = \eta^k - \mu s_k s_k = \eta^k - \mu \cdot \parallel s_k \parallel^2 = \eta^*$. Pela igualdade anterior, define-se μ como:

$$\mu = \frac{(\eta^k - \eta^*)}{\parallel s_k \parallel^2} \tag{5.22}$$

Define-se $t_k = \mu$, com o *tamanho do passo* na iteração k. Com (5.22) em (5.19) tem-se:

$$\widetilde{\lambda}^{k+1} = \lambda^k - \left(\frac{(\eta^k - \eta^*)}{\parallel s_k \parallel^2} \right) s_k \tag{5.23}$$

Finalmente, garantindo que os multiplicadores Lagrangeanos sejam positivos, tem-se a condição de atualização do multiplicador λ^k, ou seja, a projeção do vetor λ^k no plano H^* como:

$$\lambda^{k+1} = max(0, \widetilde{\lambda}^{k+1}) \tag{5.24}$$

Então, com $\lambda^k \in \mathbb{R}^m$, define-se a próxima iteração $\lambda^{k+1} \in \mathbb{R}^m$ como a projeção de $\widetilde{\lambda}^{k+1}$ no ortante[1] não negativo do $\mathbb{R}^m$, do fato que $\lambda \in \mathbb{R}^m \geq 0$.

As formulações (5.18)-(5.24) foram desenvolvidas pela suposição de conhecer a solução ótima λ^* (e η^*) do problema dual (RLAGP), na qual, é um valor desconhecido. Para resolver o anterior, pode-se estimar o valor de λ^* mas com a questão de usar um múltiplo muito pequeno o

[1]Ortante é o equivalente a um quadrante no plano cartesiano de $\mathbb{R}^2$ e um octante para o espaço $\mathbb{R}^3$.

muito grande para o subgradiente $-s^k$. Se é muito pequeno, pode-se estar fazendo um tamanho de passo muito pequeno e o processo de convergência poderia ser muito lento. Se é muito grande, λ^{k+1} poderia estar projetado longe de λ^k e também possivelmente de λ^*. Observando no processo iterativo que o comportamento da função objetivo da relaxação Lagrangeana ao problema P não melhora (ou seja, não diminui), pode-se suspeitar que o valor da estimação feito para η^* foi subestimado. Para reduzir a diferença $\eta^k - \eta^*$ pelo efeito comentado anteriormente, pode-se introduzir em (5.23) um fator positivo de *correção*, $\epsilon_k \in (0, 2]$, obtendo a seguinte expressão:

$$\widetilde{\lambda}^{k+1} = \lambda^k - s_k \cdot \epsilon_k \left(\frac{(\eta^k - \eta^*)}{\| s_k \|^2} \right) \tag{5.25}$$

Com (5.25), vão-se modificando ϵ_k no caso de que a função objetivo não tenha uma melhora para certas p iterações. Seguindo o trabalho de [3], inicia-se o valor do corretor $\epsilon_0 = 2$ e em seguida vão-se diminuindo pela metade se para p iterações a função objetivo não apresenta melhora. Com a observação anterior surge a questão dos critério de parada do método. Uma observação importante é que o método dependendo do problema P, a convergência é praticamente imprevisível. No caso para ϵ_k, poderia-se pensar em estabelecer um ponto de parada quando o valor corretor alcance um valor "muito pequeno". Seguindo os comentários de [12], para certos problemas P, o método fornece uma convergência rápida e com resultados (limitantes) bastantes confiáveis, enquanto que para outros problemas tende a produzir comportamentos erráticos na criação dos multiplicadores ou no valor da função Lagrangeana no processo iterativo. A conveniência do método para a geração dos limitantes superiores para o PMCR será discutido na Capítulo 6 com os resultados do laboratório computacional e um análise mais detalhado no Apêndice A.

Definições das Relaxações Lagrangeanas e do Método do Subgradiente para geração de limitantes superiores para o PMCR

Em seguida da revisão teórica da Relaxação Lagrangeana e do Método do Subgradiente para Funções Convexas, volta-se ao PMCR sob o modelo linear de [20] formulado em (2.22)-(2.28) e exposto novamente a continuação, por comodidade do leitor, para assim, iniciar com o estudo das Relaxações Lagrangeana para o PMC:

$$\text{(PMCR) Maximizar: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} u_i^k a_{ij}^k$$

$$\text{sujeito a: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} l_i^k a_{ij}^k \leq L$$

$$\delta_j^k L_{min}^k \leq \sum_{i \in N_k} l_i^k a_{ij}^k \leq \delta_j^k L_{max}^k \text{ com } j = 1, ..., p_k, k = 1, ..., q$$

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k, i \in N_k, k = 1, ..q$$

$$\sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta_j^k \leq F_1$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2, j = 1, ..., p_k, k = 1, ..., q$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ e inteiros, } i \in N_k, j = 1, ..., p_k, k = 1, ..., q$$

Para facilidade na manipulação das informações do modelo linear do PMCR, define-se uma nomenclatura para a função objetivo e suas restrições da seguinte forma:

- (Função Objetivo): $f(a) = \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} u_i^k a_{ij}^k$, onde $a \in M$, $M = \sum_{k=1}^{q} p_k \cdot |N_k|$.
- Restrição R1: $\sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} l_i^k a_{ij}^k \leq L$.
- Restrição R2: $\delta_j^k L_{min}^k \leq \sum_{i \in N_k} l_i^k a_{ij}^k \leq \delta_j^k L_{max}^k$ com $j = 1, ..., p_k, k = 1, ..., q$. Foi definida como uma só restrição, mas pode-se descompor em duas assim:
  - Restrição R2A: $\delta_j^k L_{min}^k \leq \sum_{i \in N_k} l_i^k a_{ij}^k$ com $j = 1, ..., p_k, k = 1, ..., q$.
  - Restrição R2B: $\sum_{i \in N_k} l_i^k a_{ij}^k \leq \delta_j^k L_{max}^k$ com $j = 1, ..., p_k, k = 1, ..., q$.
- Restrição R3: $\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k, i \in N_k, k = 1, ..q$
- Restrição R4: $\sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta_j^k \leq F_1$
- Restrição R5: $\sum_{i \in N_k} a_{ij}^k \leq F_2, j = 1, ..., p_k, k = 1, ..., q$
- Restrição R6: $\delta_j^k \in \{0, 1\}$ e $a_{ij}^k \geq 0$ e inteiros, $i \in N_k, j = 1, ..., p_k, k = 1, ..., q$

Com o modelo linear do PMCR busca-se relaxar uma ou umas restrições ("eliminar a restrição") do problema e assim definir os problemas Lagrangeanos (como foi discutido na subseção anterior) onde vão-se obter os referidos limitantes superiores. A questão importante agora é escolher a restrição ou restrições de forma que, quando sejam "eliminadas" o problema relaxado criado seja um problema "mais fácil de se resolver" em comparação ao problema primal. Ou seja, pode-se entender "mais fácil de se resolver" no sentido de que tenha um menor esforço - custo computacional. Outra questão que debe-se considerar, é que a escolha

das restrições a relaxar, na resolução do problema Lagrangeano para o PMCR forneça o melhor limitante superior e com tempos de execução aceptáveis. As restrições escolhidas para ser removidas do problema para a formulação do problema Lagrangeano será definido como um penalizador da função objetivo (ver Definição 5.3).

Escolher a restrição ou restrições de forma analítica para criar os problemas relaxados para o PMCR não são muito evidentes. Se for eliminados a priori as restrições R2 até R5, certamente tem-se um problema clássico de mochila irrestrito, mas, uma relaxação deste tipo perde a estrutura da mochila compartimentada podendo ocasionar geração de limitantes superiores muito longes do valor ótimo do problema original, sendo uma prática, por enquanto, inviável. Para realizar dita escolha, tem-se como estratégia o uso de testes computacionais para as relaxações das diversas restrições e segundo seu comportamento com exemplares de controles (exemplares pequenos de fácil visualização do processo) qual tipo de relaxação será usado para gerar os Limitantes Superiores. Dito estudo computacional pode-se ver no Apêndice A.

A modo de **exploração técnica do problema**, inicia-se a geração da primeira relaxação Lagrangeana ao problema pela restrição R2, do fato, que esta restrição é a responsável da compartimentação da mochila, assim, tem-se a *intuição* de ser uma possível boa restrição a que serem relaxada.

Então, para o PMCR (2.22)-(2.28), retira-se a restrição R2 e penaliza-se a função objetivo com esta, definindo o Problema Relaxado Lagrangeanamente na restrição R2 (seguindo a definição 5.3) como:

(RLAGPMCR$_\theta$_R2) Maximizar:

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} u_i^k a_{ij}^k + \sum_{k=1}^{q}\sum_{j=1}^{p_k}\lambda_j^k\left(\sum_{i\in N_k} l_i^k a_{ij}^k - \delta_j^k L_{min}^k\right) + \sum_{k=1}^{q}\sum_{j=1}^{p_k}\eta_j^k\left(\delta_j^k L_{max}^k - \sum_{i\in N_k} l_i^k a_{ij}^k\right) =$$

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k}\left(u_i^k + l_i^k\left(\lambda_j^k - \eta_j^k\right)\right)a_{ij}^k + \sum_{k=1}^{q}\sum_{j=1}^{p_k}\delta_j^k\left(\eta_j^k L_{max}^k - \lambda_j^k L_{min}^k\right) \tag{5.26}$$

sujeito a:

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} l_i^k a_{ij}^k \leq L \tag{5.27}$$

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k,\ i \in N_k,\ k = 1,..q \tag{5.28}$$

$$\sum_{k=1}^{q}\sum_{j=1}^{p_k}\delta_j^k \leq F_1 \tag{5.29}$$

$$\sum_{i\in N_k} a_{ij}^k \leq F_2,\ j = 1,...,p_k,\ k = 1,...,q \tag{5.30}$$

$\delta_j^k \in \{0,1\}$ e $a_{ij}^k \geq 0$ e inteiros, $i \in N_k$, $j = 1, ..., p_k$, $k = 1, ..., q$ (5.31)

Onde, $\theta \in \mathbb{R}^{2m}$ e $m = \sum_{k=1}^{q} p_k$ é o vetor de multiplicadores Lagrangeanos, com $\theta = (\lambda\ \eta) \geq 0$, $\lambda \in \mathbb{R}^m$ e $\eta \in \mathbb{R}^m$. Pode-se observar que efetivamente RLAGPMCR$_\theta$_R2 e uma relaxação ao PMCR. Agora, define-se o Problema Dual Lagrangeano, RLAGPMCR para R2, como:

$$(\text{RLAGPMCR\_R2})\ \min_{\theta \geq 0} v(\text{RLAGPMCR}_\theta\text{\_R2}) \tag{5.32}$$

Define-se como Z_{lsup} a solução do RLAGPMCR_R2, ou seja, $Z_{lsup} = v(\text{RLAGPMCR}_\theta\text{\_R2})$. Para resolver o Problema Dual Lagrangeano RLAGPMCR_R2, usa-se o Método do Subgradiente para Funções Convexas (MSFC). Para isso, define-se o vetor subgradiente e o tamanho do paso para o RLAGPMCR_R2 como segue a continuação:

Define-se o sub-gradiente G como $G = \begin{pmatrix} \overset{\lambda}{G}\ \overset{\eta}{G} \end{pmatrix}$ onde:

$$\overset{\lambda}{G}{}_j^k = \sum_{i \in N_k} l_i^k X_{ij}^k - DEL_j^k L_{min}^k \tag{5.33}$$

$$\overset{\eta}{G}{}_j^k = DEL_j^k L_{max}^k - \sum_{i \in N_k} l_i^k X_{ij}^k \tag{5.34}$$

Os valores que compõem o vetor subgradiente X_{ij}^k e DEL_j^k são as soluções das variaveis respectivamente a_{ij}^k e δ_j^k para $k = 1, ..., q$, $i \in N_k$ e $j = 1, .., p_k$, correspondente do Problema Lagrangeano Dual (5.32). Para evitar confusões, define-se $\bar{k}$ como a iteração "k" no fato do processo iterativo do Método do Subgradiente. Com o anterior, define-se o tamanho do passo $t_{\bar{x}}$ como:

$$t_{\bar{k}} = \frac{\epsilon_{\bar{k}} \cdot (Z_{lsup} - Z_{linf})}{\sum_{k=1}^{q} \sum_{j=1}^{p_k} \left(\left(\overset{\lambda}{G}{}_j^k \right)^2 + \left(\overset{\eta}{G}{}_j^k \right)^2 \right)} \tag{5.35}$$

Agora, usa-se o MSFC para resolver o RLAGPMCR_R2. Vai-se usar processos iterativos na atualização do vetor dos multiplicadores Lagrangeanos, neste caso de θ, usando como referencia os trabalhos de [12], [3] e [9], como segue:

- **Passo 1**:

- Escolhe-se valor inicial de ϵ_0 (corretor das iterações para o cálculo do passo t_k [12]), onde $0 < \epsilon_{\bar{k}} \leq 2$. Por [9] e [3], inicia-se com $\epsilon_0 = 2$. Ajuste-se $\epsilon_{\bar{k}}$ no caso que a partir de $\bar{p}$ iterações a função objetivo não apresenta melhoria. Escolhe-se $\bar{p} = 10$ iterações, a modo de ter um parâmetro de iteração de controle. Para um modo geral, seja a variável computacional cc_ϵ o que vão tomar o rol de $\bar{p}$ iterações infrutuosas.
- Procura-se calcular um bom estimativo para η^*, na qual este valor corresponderá a um limitante inferior ao PMCR. Define-se este valor como Z_{linf}. Foi escolhido por ter bons resultados na literatura para estimar Z_{linf} a heurística das W capacidades para o PMCR [28].
- Define-se o conjunto inicial do vetor multiplicador Lagrangeano θ. Em [9] propõe iniciar com $\theta = 0$, mas no trabalho de [3] foi calculado um vetor inicial com componentes emtre 0 e 1. Por conveniência, inicia-se com o vetor $\theta = 0$.

- **Passo 2**: Resolva o (RLAGPMCR$_\theta$_R2) e atualize Z_{lsup}.
- **Passo 3**: Calcule o vetor subgradiente G.
- **Passo 4**: Define-se o tamanho do passo $t_{\bar{k}}$. Lembre-se $\bar{k}$ corresponde á iteração em execução.
- **Passo 5**: Atualize o vetor θ por $tetha = max\{0, \theta - t_k \cdot G\}$ e volte ao **Passo 2**. Determine-se um critério de parada para o processo iterativo. O critério de parada de controle vão-se fixar quando o valor de correção atinga a $\epsilon_{\bar{k}} \leq 0,005$.

Como foi estudado na seção anterior, o método do Subgradiente não proporciona um critério genérico de parada do processo iterativo. Pelo tanto, pode-se definir um algoritmo que provê um critério de parada para resolver o RLAGPMCR_R2, que é como segue: seja a iteração $\bar{p}$ no método do Subgradiente, onde, tem-se para dita iteração o valor corrente de $Z_{lsup_{\bar{p}}}$ (para evitar confusões, será representar como $Z_{lsupIter}$) e o melhor limitante achado nas $\bar{p}$ iterações como Z_{lsup}, onde, com esta informação avalia-se, primeiro, se foi obtido o valor ótimo na iteração $\bar{p}$, onde, tem-se que cumprir que $Z_{lsup} = Z_{linf}$. Segundo, onde para $Zlsup \neq ZlsupIter$, se o valor corretor ϵ diminui em um valor menor a $0,005$ (parâmetro definido como menor valor de ϵ aceptável), o processo iterativo finaliza. No Algoritmo 23 pode-se ver o critério de parada definido.

Algoritmo 22: Algoritmo Critério de parada Método do Subgradiente

Entrada: $Zlsup$

Saída: Pare o continuação de execução do algoritmo

se $Zlsup = Zlin$ **então**
| Pare! Valor ótimo achado

fim

se $ZlsupIter - Zlsup <> 0$ *!É um controle no processo iterativo* **então**
| **se** $\epsilon_{\bar{p}} \leq 0,005$ **então**
| | Pare! Valor de ϵ_k atingido
| **fim**

fim

Fonte: Próprio autor

Também é preciso fazer um controle da mudança de cc_ϵ, qual vão fazer a variação do valor de ϵ_k para o caso de não ter para $\bar{s}$ iterações um melhor valor da função objetivo da relaxação Lagrangeana. Em seguida, define-se uma variável computacional que vão "contar" o numero de iterações de não melhoria na função objetivo, define-se como *contador2*. Também define-se a variável cc_ϵ, como o número de iterações permitidas para antes mudar o valor de ϵ_k. Então, tem-se com algoritmo:

Algoritmo 23: Algoritmo mudança ϵ_k

Entrada: $contador2$, cc_ϵ

Saída: $\epsilon_{\bar{p}}$

se $contador2 = cc_\epsilon$ **então**
| $\epsilon_{\bar{p}} := \frac{\epsilon_{\bar{p}}}{2}$;
| $contador2 = 0$

fim

Fonte: Próprio autor

Outro parâmetro a definir no método, é a quantidade de iterações que vão-se realizar, procurando ter uma convergência no processo. Define-se $iterac$ o número de iterações que vão desenvolver no Método do Subgradiente, onde, a modo de controle, vão-se iniciar com $iterac = 100$.

Em testes prévios foi necessário definir um algoritmo de controle para o processo iterativo em relação de obter limitantes superiores válidos, do fato que algumas relaxações Lagrangeanas podem ter qualidade (em sentido de aproximação do ótimo) não adequadas. Dito algoritmo é como sege a continuação:

Algoritmo 24: Algoritmo de Controle para atualizar $ZlsupIter$

se $Z_{lsup} \leq Z_{lin}$ *!Controle da primeira iteração* **então**
  Calcule-se o Subgradiente G (5.33)-(5.34);
  Calcule-se $t_{\bar{k}}$ (5.35);
  $\theta = max\{0, \theta - t_k \cdot G\}$;
  !Observação: este controle tem como objetivo pular o resultado até obter limitantes válidos para o problema;
senão
  se $contador1 = 1$ **então**
    $ZlsupIter := Zlsup$
  senão
    se $Zlsup < ZlupIter$ **então**
      $ZlsupIter := Zlsup$
    fim
  fim
fim

Fonte: Próprio autor

Faça um Algoritmo que resume os algoritmo que definem os passos 3, 4 e 5 do Método do Subgradiente, como segue a continuação:

Algoritmo 25: Passo 3, 4 e 5 do Método do Subgradiente

Faça:
Algoritmo 24 !Critério de mudança do ϵ_k;
Algoritmo 23 !Critério de Pare! do método;
Calcule-se o Subgradiente G (5.33)-(5.34)!Passo 3;
Calcule-se $t_{\bar{k}}$ (5.35) !Passo 4;
$\theta = max\{0, \theta - t_k \cdot G\}$!Passo 5;

Fonte: Próprio autor

Pela ordenação feita no inicio do problema, seja r a posição do nó onde está-se pesquisando a necessidade de desenvolver o ramo ou fazer a poda do mesmo, com valores conhecidos $\hat{\alpha}_1, \hat{\alpha}_2, ..., \hat{\alpha}_r$, onde cada $\hat{\alpha}_t$ para $t = 1, ..., r$ representa o nó da árvore com as informações do item, compartimento e classe onde ele esteja localizado (veja-se os comentários iniciais do presente capítulo sobre ordenação das variáveis de decisão). Com os valores conhecidos $\hat{\alpha}_1, \hat{\alpha}_2, ..., \hat{\alpha}_r$ vão-se aplicar o Método do Subgradiente para calcular o limitante do ramo em questão. Finalmente, tem-se o Algoritmo do Método do Subgradiente para Funções

Convexas (MSFC) para resolver RLAGPMCR_R2 como segue a continuação:

Algoritmo 26: Método do Subgradiente para Funções Convexas (MSFC) para RLAGPMCR_R2

Entrada: r , $\hat{\alpha}_1$, $\hat{\alpha}_2$,..., $\hat{\alpha}_r$

Saída: Z_{lsup}

Inicialização: faça leitura de $\hat{\alpha}_1, \hat{\alpha}_2, ..., \hat{\alpha}_r$

Passo 1: faça $\epsilon_0 := 2$, $contador1 := 0$, $contador2 := 0$, $Z_{lsup} := 0$, $Z_{lin} :=$ (execução Algoritmo 2), $iterac := 100$, cc_ϵ;

enquanto $contador1 < iterac$ **faça**

- $contador1 += 1$;
- **Passo 2 MSFC** *(Cálculo do melhor Limitante Superior)*:
- Faça solução de RLAGPMCR$_\theta$_R2;
- Faça o Algoritmo 25;
- **se** $ZlsupIter - Zlsup \leq 0$ **então**
  - $contador2 += 1$
  - Faça o Algoritmo 26;
- **senão**
  - $contador2 = 0$;
  - Faça o Algoritmo 25;
- **fim**

fim

Fonte: Próprio autor

Com a obtenção de Z_{lsup}, redefine-se o Passo 6 da arvore de exploração do *Branch and Bound* para o PMCR como segue no Algoritmo 28.

A restrição R2 foi usada para estudar o processo para formular a Relaxação Lagrangeana ao PMC como foi descrito em (5.26)-(5.31), assim como definir o problema Lagrangeano Dual (5.32) junto à definição dos Subgradientes (5.33)-(5.34) e o tamanho do passo (5.35).

Algoritmo 27: Passo 6: com limitante superior via Relaxação Lagrangeana para RLAGPMCR_R2

Passo 6 *(Poda)*:

se $Z_{lsup} > z$ **então**

- **Passo 1** Ver Algoritmo 7

senão

- **Passo 4** Ver Algoritmo 19

fim

Fonte: Próprio Autor

Também foi usada para apresentar os algoritmos que definem o Método do Subgradiente para resolver o problema Lagrangeano Dual RLAGPMCR_R2 com o Algoritmo 27. Então, pode-se generalizar a relaxação Lagrangeana para qualquer restrição ou conjunto delas para o PMCR, definindo um vetor multiplicador Lagrangeano $\lambda \geq 0$ sob as restrições que vão penalizar a função objetivo, na qual, será do interesse em atualizar para calcular o Problema Lagrangeano Dual de qualquer relaxação feita. Nesse sentido, vão-se definir de forma geral o Método do Subgradiente para Funções Convexas, e assim, ter o aproveitamento do algoritmo para estudar e avaliar diversas relaxações Lagrangeanas (veja-se o Capítulo 6 e Apêndice A). Qual seja a restrição a relaxar, sempre vão-se definir o vetor multiplicador Lagrangeano $\lambda \neq 0$, o problema Lagrangeano RLAGPMCR$_\lambda$, o vetor subgradiente G e o tamanho do passo t_k. Dessa forma, pode-se definir um algoritmo de controle geral para atualizar $Z_{lsupIter}$ como é apresentado no Algoritmo 29:

Algoritmo 28: Algoritmo Geral de Controle para atualizar $ZlsupIter$

se $Z_{lsup} \leq Z_{lin}$ *!Controle da primeira iteração* **então**
  Calcule-se o vetor Subgradiente G;
  Calcule-se o tamanho do passo $t_{\bar{k}}$;
  $\lambda = max\{0, \lambda - t_k \cdot G\}$;
senão
  se $contador1 = 1$ **então**
    $ZlsupIter := Zlsup$
  senão
    se $Zlsup < ZlupIter$ **então**
      $ZlsupIter := Zlsup$
    fim
  fim
fim

Fonte: Próprio autor

Já com o Algoritmo 29 pode-se definir o algoritmo geral do Método do Subgradiente para qualquer relaxação Lagrangeana do PMC como segue o Algoritmo 31. Com a obtenção de Z_{sup} para qualquer Problema Dual Lagrangeano obtido do PMCR, redefine-se o Passo 6 da árvore de enumeração do *Branch and Bound* para o PMCR como é apresentado no Algoritmo 30.

Algoritmo 29: Passo 6 (Poda): com limitante superior via Relaxação Lagrangeana Geral

Passo 6 *(Poda)*:

se Z_{lsup}>z **então**

| **Passo 1** Ver Algoritmo 7

senão

| **Passo 4** Ver Algoritmo 19

fim

Fonte: Próprio Autor

Algoritmo 30: Método do Subgradiente Geral para para o PMCR

Entrada: r , $\hat{\alpha}_1$, $\hat{\alpha}_2$,..., $\hat{\alpha}_r$

Saída: Z_{lsup}

Inicialização: faça leitura de $\hat{\alpha}_1, \hat{\alpha}_2, ..., \hat{\alpha}_r$

Passo 1: faça $\epsilon_0 := 2$, $contador1 := 0$, $contador2 := 0$, $Z_{lsup} := 0$, $Z_{lin} :=$ (execução Algoritmo 2), $iterac := 100$, cc_ϵ

enquanto $contador1 < iterac$ **faça**

| $contador1+ = 1$

| **Passo 2** *(Cálculo do melhor Limitante Superior)*:

| Faça solução de RLAGPMCR$_\lambda$;

| Algoritmo 29;

| **se** $ZlsupIter - Zlsup \leq 0$ **então**

| | $contador2+ = 1$;

| | Algoritmo 24 !Critério de mudança do ϵ_k; Algoritmo 23 !Critério de Pare! do método; Calcule-se o Subgradiente G !Passo 3;

| | Calcule-se o tamanho do passo $t_{\bar{k}}$!Passo 4; $\lambda = max\{0, \lambda - t_k \cdot G\}$!Passo 5;

| **senão**

| | $contador2 = 0$;

| | Algoritmo 24 !Critério de mudança do ϵ_k; Algoritmo 23 !Critério de Pare! do método; Calcule-se o Subgradiente G !Passo 3;

| | Calcule-se o tamanho do passo $t_{\bar{k}}$!Passo 4;

| | $\lambda = max\{0, \lambda - t_k \cdot G\}$!Passo 5;

| **fim**

fim

Fonte: Próprio autor

6 EXPERIMENTOS COMPUTACIONAIS

Dois grandes momentos vão configurar o Capítulo de Experimentos Computacionais. O primeiro vão apresentar os testes computacionais referentes ao Capítulo 4, do Modelo Linear Forte do Problema da Mochila Compartimentada. O Segundo será dedicado ao estudo experimental do método *Branch and Bound* definido no Capítulo 5.

Antes de abordar os experimentos computacionais feitos neste trabalho de dissertação, observe-se os componentes técnicos que foram usados. As implementações, simulações e ensaios numéricos foram feitos com o uso da suíte de FICO® Xpress para arquitetura 64 bits com licença completa, com os seguentes componentes: interfase gráfica FICO® Xpress IVE versão $1.24.20$, linguagem FICO® Xpress Mosel versão $4.8.1$ e pacotes de otimização com FICO® Xpress *Optimizer* versão $32.01.05$. Foi usado o equipamento informático do laboratório de simulação SIMULAB do departamento de matemáticas da Universidade Estadual de Londrina, em uma máquina com especificações: processador Intel® Inside™ Xeon® CPU W3520, quatro núcleos com frequência baseada no processador de $2,67$ GHz, cache de 8 MB e memória R.A.M. de 8 GB sob sistema operativo Microsoft Windows® 10.

6.1 EXPERIMENTOS COMPUTACIONAIS DO MODELO LINEAR FORTE DO PMCR

Nesta seção vão-se apresentar a avaliação da qualidade do Modelo Linear Forte do Problema da Mochila Compartimentada que foi estudado no Capítulo 4.

Para desenvolver a avaliação, foram organizadas 4 categorias de exemplares definidas pelos tamanhos das classes com $q = 5, 10, 30, 40$, e, para cada categoria foram organizados 5 sub-categorias definidos pelo número de itens que terá cada classe, sendo representado por n, com $n = 5, 10, 20, 30, 40$. Desta forma, represente-se cada sub-categoria por q/n, onde este vão indicar exemplares com q classes e com n itens em cada uma delas. Veja-se na Tabela 6.1 o resumo desta organização.

	Categorias dos exemplares				
	q	5	10	30	40
Sub-categorias	q/n	5/5	10/5	30/5	40/5
		5/10	10/10	30/10	40/10
		5/20	10/20	30/20	40/20
		5/30	10/30	30/30	40/30
		5/40	10/40	30/40	40/40

Tabela 6.1: Categorias e sub-categorias definidas para os exemplares.

Para criar os exemplares em cada sub-categoria definida acima, foi fixado os seguente

parâmetros de simulação seguindo valores realísticos de problemas de corte de bobinas de aço em duas etapas de [18]: capacidade da mochila $L = 1100\ mm$; valores das facas $F_1 = 8$ e $F_2 = 12$, capacidades dos compartimentos para cada classe $k = 1, ..., q$ por $L^k_{min} = 154\ mm$ e $L^k_{max} = 456$, as larguras dos itens vão estar definidos entre os valores $53\ mm$ e $230\ mm$ e as utilidades dos itens vão estar compreendidos entre os valores 0 e 1. Para a designação da demanda de cada item foi definido para cada classe $k = 1, ..., q$ um valor chamado de *demanda máxima da classe* na qual é dividido em quantidades inteiras (diferentes) para logo atribuí-los como as demandas dos itens. Esta demanda máxima vão corresponder três vezes o número total dos itens da classe.

Com este parâmetros foi criado um gerador aleatório de exemplares com referencia em [10], onde para um exemplo de uma categoria q/n é designado de forma aleatória os valores das larguras e das utilidades dos itens, como também faz o cálculo da demanda máxima para cada classe, fazendo partição deste valor e atribuindo-lo de forma aleatória a cada item. Seguindo o anterior, foram criados 100 exemplares para cada subcategoria q/n, obtendo um total de 500 exemplares por cada categoria q para assim ter um total de 2000 exemplares. Cada um destes exemplares foram sometidos a processo de ajustes dos valores das capacidades dos compartimentos, das facas e os valores das larguras seguindo o feito nas seções 4.1.1, 4.1.2 e 4.1.3, obtendo um total de 2000 exemplares ajustados.

Com os exemplares feitos, foram expostos a simulação sob três modelos que solucionam o Problema da Mochila Compartimentada: O Algoritmo de Decomposição Exaustivo, o Modelo Linear de [20] e o Modelo Linear Forte, que o centro da avaliação.

O Algoritmo de Decomposição Exaustivo, apresentado na seção 2.6.2, é um algoritmo que gera todos os compartimentos factíveis formando o conjunto V (que é o conjunto de todas as soluções factíveis para um PMC) e em seguida resolve o problema usando um problema de mochila irrestrito chamado de Problema de Mochila Mestre (2.36)-(2.40). Os resultados da simulações com o Algoritmo de Decomposição Exaustivo são os apresentados na Tabela 6.2.

O Algoritmo Exaustivo resolve cada exemplo obtendo o valor ótimo, o que será o referente da qualidade das soluções provenientes do modelo Linear Forte. O Modelo Linear do PMCR de [20] formulado em (2.22)-(2.28) que é o modelo que foi restruturado com o intuito de comparar soluções e tempos de execução com o modelo Linear Forte (4.31)-(4.38). Daqui até o final deste trabalho, este tipo de modelo será referido como o Modelo Linear do PMCR e representado por ML. Finalmente com os exemplares que foram fortalecidos foi implementado o Modelo Linear Forte (4.31)-(4.38). Veja-se representado este modelo por MLF. Os resultados da simulações com o Modelo Linear e o Modelo Linear Forte são os apresentados na Tabelas 6.2.

Cada implementação desenvolvida nos modelos Modelo Linear e o Modelo Linear Forte foram feitos as seguentes medições: o gap entre a solução ótima da relaxação linear obtida com os modelos em comparação com a solução ótima inteira obtida com o Algoritmo de Decomposição Exaustivo, o tempo gastado para resolver o problema e o número de nós gerados

no processo do *Branch and Bound* para o cálculo do ótimo inteiro. O cálculo do gap foi definido como $gap = (z_{ub} - z^*_{ip})$, onde z_{ub} faz referencia ao valor da solução ótima da relaxação linear do modelo e z^*_{ip} que é o valor da solução ótima inteira do modelo. A medição dos nós gerados no processo *Branch and Bound* foi feito mediante o uso do atributo "$XPRS\_NODES$" do Xpress Mosel.

Na Tabela 6.2 é apresentado as seguentes informações: a categoria que corresponde os exemplares testados, representado por " $q = k$" (para $k = 5, 10, 30, 40$); a sub-categoria de cada categoria é representado por n (para $n = 5, 10, 20, 30, 40$); a médias das soluções ótimas de cada exemplo testado, representado por $\overline{Obj}$; a média dos tempos de excussão obtidos na resolução de cada exemplo q/n, representado por $\overline{T}$; a média dos desvios padrões dos tempos referidos anteriormente, representado por $\sigma(T)$.

Cada tabela dos resultados das simulações com os Modelos Linear e o Modelo Linear Forte são apresentado as seguentes informações: a média dos gap obtido da sub-categoria q/n, representado por $\overline{gap}$; o menor gap obtido na sub-categoria q/n, representado como gap_{min}; o maior gap obtido na sub-categoria q/n, representado como gap_{max}; a média dos tempos de excussão obtidos na subcategoria q/n, representado por $\overline{T}$ e a média dos nós gerados na categoria q/n representado como $\overline{n_{BB}}$.

A continuação é feito alguns comentários sob os resultados obtidos referidos da Tabela 6.2. Primeiro destaque-se que o Modelo Linear Forte resolveu cada exemplo testado obtendo a sua solução ótima, sendo confirmado por ter os mesmos resultados com os valores obtidos com o Algoritmo de Decomposição Exaustiva, concluindo-se que este modelo é um modelo válido para o PMCR. Observe-se também que o MLF e o Algoritmo de Decomposição Exaustiva consegue melhorar na maioria das sub-categorias os tempos de execução. Pode-se destacar que em cada categoria a maior número de instancias, ou seja, a maior número de itens para cada classe, tem-se um melhor comportamento do MLF, obtendo reduções importantes no tempo.

Veja-se o comportamento computacional do Modelo Linear sob os parâmetros fixados para estes trabalho de dissertação, a diferença aos usados em [20] e em [19]. Para categorias com poucas instancias, o ML não é mais eficiente do que o Algoritmo de Decomposição Exaustiva em termos dos tempos de execução, mas tem um melhor comportamento para instancias grandes como $n = 40$, com um melhoramento aproximado do 90% para $q = 5$ e do 50% para as outras categorias.

Agora observe-se o comportamento computacional para o Modelo Linear Forte. O Modelo para as menores instancias de itens $n = 5, 10$ em todas as categorias não foi mais eficiente do que o Algoritmo de Decomposição Exaustiva, a excepção das sub-categorias $5/10$. A medida que vão-se incrementando o número de instancias para cada categoria, o MLF apresenta um melhor comportamento, destacando-se com melhor tempo de execução para a categoria $q = 40$, um melhoramento de aproximadamente do 80% para as categorias $q = 5, 10$ e de 69% para a categoria $q = 30$.

Faça-se em seguida uma comparação dos modelos ML e MLF. o Modelo Linear Forte

obteve melhores resultados nas medições apresentados tendo melhor média de $\overline{gap}$, gap_{min}, gap_{max}, $\overline{T}$ e $\overline{n_{BB}}$ em comparação ao Modelo Linear em todas as categorias com excepção da sub-categoria $5/40$ para o $\overline{T}$. Fazendo um análise aprofundada nesta sub-categoria $5/40$ observe-se que o rango dos tempos obtidos com Modelo Linear estão entre os valores 0.031 e 235.906, e para o Modelo Linear Forte está entre os valores 0.031 e 193.860, onde, a pesar de MLF ter uma pior média de tempos em comparação com ML, este tem um melhor intervalo de tempos. Com os intervalos, veja-se ainda que o MLF apresenta a melhor agrupação de resultados de tempo, tendo como média da desviação padrão dos tempos de 20.641 em comparação do $26,566$ do ML.

		Alg. Dec. Exaustiva			**Modelo Linear**							**Modelo Linear Forte**						
q	n	$\overline{Obj}$	$\overline{T}$	$\sigma(T)$	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{T}$	$\sigma(T)$	$\overline{n_{BB}}$	$\sigma(n_{BB})$	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{T}$	$\sigma(T)$	$\overline{n_{BB}}$	$\sigma(n_{BB})$
5	5	11.0	0.1	0.1	5.4%	0.4%	17.1%	0.5	0.4	667	2222	4.1%	0.4%	14.8%	0.14	0.12	338	857
	10	11.1	2.4	5.6	1.9%	0.0%	4.0%	12.6	44.1	215237	757793	1.7%	0.0%	3.3%	1.7	3.2	22095	45397
	20	12.0	1.4	1.3	1.8%	0.0%	7.7%	1.8	8.5	19657	152473	1.5%	0.0%	6.7%	0.8	2.3	6829	25210
	30	16.0	4.6	0.7	3.8%	0.3%	7.8%	0.6	0.7	356	1586	2.9%	0.3%	5.5%	0.3	0.3	299	844
	40	14.7	31.3	17.6	1.4%	0.0%	3.2%	4.9	26.6	61548	338284	1.3%	0.0%	2.6%	5.7	20.6	56002	211597
10	5	11.1	0.1	0.1	16.3%	3.3%	40.2%	0.7	0.5	117	386	10.2%	1.5%	28.8%	0.1	0.1	45	217
	10	11.5	1.2	2.6	2.4%	0.2%	4.4%	12.0	32.8	164836	508226	1.7%	0.1%	3.2%	1.5	2.7	13088	32959
	20	12.4	2.4	0.8	2.1%	0.0%	5.4%	1.1	1.5	1974	6933	1.6%	0.0%	4.8%	0.4	0.5	1399	4605
	30	15.7	29.1	58.7	2.4%	0.5%	4.3%	238.5	778.3	1871697	5113562	1.6%	0.1%	3.1%	24.1	60.5	230231	602439
	40	8.6	65.7	25.3	0.5%	0.0%	1.4%	31.2	773.1	283609	1519670	0.5%	0.0%	1.4%	11.2	33.6	112476	345832
30	5	14.4	0.1	0.2	7.0%	2.7%	13.4%	0.7	1.4	2534	12348	1.8%	0.1%	4.5%	0.2	0.3	855	2545
	10	12.0	3.3	13.8	2.5%	0.4%	4.7%	62.2	237.4	369001	1394648	1.4%	0.1%	3.1%	10.0	25.0	50105	119185
	20	14.1	6.4	1.8	13.8%	1.7%	29.6%	137.9	553.2	1043677	4242278	5.2%	1.0%	12.3%	12.9	50.4	68463	279227
	30	17.2	39.9	8.9	6.7%	0.5%	30.1%	62.9	372.4	242982	1508763	2.9%	0.5%	5.1%	5.3	38.1	32623	260563
	40	15.3	326.5	210.9	2.1%	0.2%	5.5%	144.9	773.1	475543	2442412	1.8%	0.2%	2.5%	100.0	536.8	374520	2075972
40	5	14.5	0.2	0.2	6.7%	2.8%	11.2%	1.1	1.4	9210	26711	2.3%	0.1%	4.9%	0.3	0.2	1250	2667
	10	12.1	3.0	7.1	2.6%	0.8%	5.9%	209.3	237.4	928612	3003257	1.3%	0.1%	2.7%	25.5	93.5	122481	457606
	20	13.9	10.0	1.9	10.9%	0.0%	27.4%	99.3	553.2	219957	808255	3.7%	0.0%	11.1%	1.9	5.2	5116	16017
	30	17.4	70.8	20.7	4.6%	0.0%	27.6%	164.8	372.4	646486	3489180	2.2%	0.0%	7.4%	5.2	19.7	39592	165343
	40	15.6	573.9	151.1	1.9%	0.2%	5.8%	288.3	773.1	545855	2111186	1.5%	0.2%	2.8%	3.0	6.5	11226	41060
Média		13.5	58.6	26.5	4.8%	0.7%	12.9%	73.8	277.1	355177.7	1372008.6	2.6%	0.2%	6.5%	10.5	45.0	57451.5	234507.1

Tabela 6.2: Resultados das simulações das categorias de exemplares $q = 5, 10, 30, 40$ feitas com o Algoritmo de Decomposição Exaustiva, o Modelo Linear e o Modelo Linear Forte.

Valide-se a continuação a formulação de um domínio reduzido para a relaxação linear do PMCR (RLPMCR). Dois parâmetros validarão a redução do domínio da RLPMCR: o gap como a qualidade da solução obtida na RLPMCR e em seguida, o número de nós desenvolvidos para o processo do *Branch and Bound*. Veja-se que o MLF apresentou em todas as sub-categorias de exemplares testados redução do número de nós no processo *Branch and Bound*, obtendo uma diminuição em média de $\overline{n_{BB}}$ do 84%, tendo em concordância melhores gap em média de $\overline{gap}$ do 47% em comparação ao ML. Pelo anterior confirma que o fortalecimento feito nas informações dos PMCR produz um domínio reduzido para o PMCR.

Um comentário final. Tem-se uma particularidade detectada no processo de simulações que favoreceu em maior ou menor medida em ter o melhor rendimento global do MLF: todos os exemplares apresentaram diferentes níveis de fortalecimento. Se tiveram exemplares

onde se conseguiu fazer importantes ajustes nas informações do problema, obtendo o melhor rendimento do MLF com estes exemplares, por exemplo na categoria $q = 40$ se apresentou em maior medida este fenômeno. Do anterior também implicou que o fortalecimento das informações para exemplares com categorias grandes tem-se maior sensibilidade nos ajustes , fazendo que o MLF tenha um melhor comportamento computacional em comparação ao ML. Outro tipo de exemplares apresentaram poucas informações ajustadas (por exemplo só se conseguiu ajustar os valores de algumas das facas), como foi nas categorias menores, tendo assim um melhor rendimento do MLF.

6.2 Experimentos Computacionais para o método *Branch and Bound* do PMC

Nesta seção vão-se apresentar a avaliação do Algoritmo *Branch and Bound* feito para o Problema da Mochila Compartimentada estudado neste trabalho de dissertação (veja-se Capítulo 5). Três etapas de avaliação vão-se desenvolver. Na primeira etapa, vão-se avaliar a qualidade da árvore de enumeração apresentadas nos algoritmos (6)-(21), na segunda etapa, vão-se avaliar qualidade dos limitantes superiores obtidos via relaxação Lagrangeana (em referencia ao estudado no Apêndice A) e finalmente, na última etapa, vão-se estudar o comportamento do método do *Branch and Bound* usando a árvore de enumeração dos algoritmos (6)-(21).

Árvore de enumeração proposto para o PMCR

Na Subseção 5.1 foi apresentados os algoritmos (6)-(21) que fornecem um método de enumeração sistemática das soluções sensíveis para o PMCR, na qual, será avaliada nesta seção. O principal critério que foi fixado neste trabalho para avaliar a árvore de enumeração é a qualidade da enumeração completa que faz das soluções factíveis para o PMCR. Um primeiro passo foi estudar para quais instancias de exemplares q/n a árvore de enumeração tem um comportamento computacional aceptável no sentido que não exausta a memória da máquina que é usado para este análise (veja-se as indicações técnicas referidas no inicio do capitulo).

Para o anterior, foram criados exemplares para o PMCR com parâmetros de simulação que foram expostos na seção 6.1 com variações de q e de n obtendo como resultado: a máxima categoria com resultados aceptáveis foi para $q = 5$ e com instancia máxima de $n = 10$. Pelas limitações achadas, foram organizados 3 categorias q/n (com q o número de classes e n o número de itens por classe) com $2/5, 3/5, 4/5$. Como foi explicado na seção 6.1 foram criados 100 para cada categoria definida, obtendo um total de 300 exemplares. Cada exemplo foi sometido ao processo de fortalecimento, obtendo 300 exemplares ajustados.

A árvore de enumeração foi sometida a simulações com os exemplares originais (vão-se representar por AE) e pelos exemplares fortalecidos (representando-se como AEF). A Tabela 6.3 são apresentados os resultados dos testes. Nesta tabela são expostos quatro medições: a média dos gap obtidos na busca exaustiva da árvore de enumeração, representado por $\overline{gap}$, o menor

gap e a maior gap obtidos em cata categoria, sendo representado por gap_{min} e gap_{max}; e a média dos tempos gastados na enumeração feita pela árvore em cata categoria, sendo representada por $\overline{T}$.

	AE				**AEF**			
	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{T}$	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{T}$
2/5	0.17%	0.00%	5.64%	16.190	0.17%	0.00%	5.64%	13.711
3/5	0.02%	0.00%	1.51%	83.464	0.02%	0.00%	1.51%	69.062
4/5	0.00%	0.00%	0.00%	314.801	0.00%	0.00%	0.00%	257.713
Média	0.06%	0.00%	2.38%	138.15	0.06%	0.00%	2.38%	113.49

Tabela 6.3: Resultados das simulações das categorias de exemplares $q/n = 2/5, 3/5, 4/5$ feitas com exploração exaustiva da árvore de enumeração com (AEF) e sem (AE) dados fortes para o PMCR.

Com os resultados mostrados na Tabela 6.3, tem-se dois elementos a destacar. O primeiro, em relação á qualidade da enumeração exaustiva que faz a árvore. A árvore de enumeração com os exemplares com e sim fortalecer obteve em média de $\overline{gap}$ um 0.06% o qual indica que a árvore para a maioria de exemplares testados fez uma enumeração completa. O anterior é favorável, verificando que os algoritmos formulado verdadeiramente faz o processo de *branching* para o PMRC. Mas, os $gap \neq 0$, indicam que falta desenvolver algum algoritmo de controle adicional (veja-se os algoritmos de controle estabelecidos para a árvore de numeração no Capítulo 5) que ainda não foi considerado e que certamente está pulando alguns ramos para estes exemplares. Segundo, as vantagens do uso dos exemplares fortalecidos na árvore de enumeração. A obtenção dos mesmos $\overline{gap}$, gap_{min} e gap_{max} dá para supor que a árvore com e sem dados fortalecidos faz a mesma quantidade de construções de ramos, mas com os dados "fortes" o processo de preenchimento e de verificação de compartimentos é mais eficiente do que com os exemplares originais.

Qualidade do limitante superior tipo Bactracking para o algoritmo Branch and Bound do PMCR

Com a árvore de enumeração anterior validada e com o limitante superior tipo *Bracktracking* formulado em (5.7), defina-se o algoritmo *Branch and Bound* para o PMCR pelos algoritmos (6)-(20) e (22). Com este algoritmo *Branch and Bound* foi sometido a simulação com os mesmos exemplares usados na seção anterior, com categorias $q/n = 2/5, 3/5, 4/5$. Na Tabela 6.4 é apresentado os resultados desde estudo. Aqui são expostos quatro medições: a média dos gap, sendo representado por $\overline{gap}$; o menor gap e a maior gap obtidos em cada categoria, sendo representado por gap_{min} e gap_{max}; e a média dos tempos gastados pelo *Branch and Bound*, sendo representado por $\overline{T}$.

Nos resultados da Tabela 6.4 são representados com F as categorias que não concluíram

favoravelmente os testes feitos por ultrapassar tempos fixados para estas simulações. O algoritmo *Branch and Bound* com dados fortes (BBF) conseguiu ter um melhor comportamento do que o algoritmo *Branch and Bound* sem dados fortes (BB), mas não conseguiu resolver o 100% de exemplares nas categorias $3/5$ e $4/5$, só uma mostra destes foram resolvidos.

	BB				**BBF**			
	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{T}$	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{T}$
2/5	0.17%	0.00%	5.64%	7.517	0.17%	0.00%	5.64%	6.421
3/5	F	F	F	F	F(0.26%)	F(0.00%)	F(5.08%)	F(31.192)
4/5	F	F	F	F	F(0.00%)	F(0.00%)	F(0.00%)	F(139.883)
Média	0.17%	0.00%	5.64%	7.517	F(0.13%)	F(0.00%)	F(2.96%)	F(59.7)

Tabela 6.4: Resultados das simulações das categorias de exemplares $q/n = 2/5, 3/5, 4/5$ feitas no algoritmo *Branch and Bound* do PMCR sem (BB) e com dados fortes (BBF).

Qualidade dos limitante superior gerados via relaxação Lagrangeana para o algoritmo Branch and Bound do PMCR

Nesta última sub-seção é apresentado a qualidade dos limitantes superiores obtidos via relaxação Lagrangeana. Para este propósito o Modelo Linear e o Modelo Linear Forte foram relaxados Lagrangeanamente as restrições R1, R1-2B, R2, R2A, R2B e R3 seguindo um estudo preliminar feita sob estas restrições no Apêndice A. É mantida a dinâmica dos testes feitos até o momento: são usados es exemplares das categorias $q/n = 2/5, 3/5, 4/5$.

Com o limitantes superiores gerados, dois componentes foram estudados. Primeiro a qualidade da heurística usada no método do sub-gradiente, sendo medido o gap do limitante superior e o valor da heurística para cada exemplo testado. Segundo a qualidade geral do limitante superior, medindo o gap entre a solução ótima e a solução do limitante superior. Na Tabela é apresentado os resultados desde estudo. Nesta tabela são expostos sete medições: a média dos gap obtidos na comparação da Heurística das W Capacidades e a solução ótima do exemplo, sendo representado por $\overline{gap}$; o menor gap e a maior gap obtidos em cada categoria e comparação, sendo representado por gap_{min} e gap_{max}; e a média dos tempos gastados na obtenção do limitante superior, sendo representada por $\overline{T}$. Na Tabela 6.5 são apresentados os resultados destes testes.

Para desenvolver analises dos resultados da Tabela 6.5 é preciso revisar com detalhe o estudo computacional feito no Apêndice A, já que os dados estão condicionados pelos comportamentos específicos de cada tipo de relaxação feita. No processo interno na obtenção dos limitantes superiores para o ML e MLF nas restrições R2, R2A, R2B sempre apresentaram complicações nas simulações, uma no sentido de que ultrapassaram os tempos limites para as simulações (foram fixados em uma hora para cada exemplo) e outra de não ter processo de convergência fazendo que o processo iterativo tenha um alto costo computacional em relação á

qualidade (observe-se que foram descartados os resultados para o ML a relaxação da restrição RL, sendo representado estes com F). As restrições que certamente obtiveram um melhor comportamento ao fazer realmente o processo iterativo foram as restrições R1, R3 e R1-R2B, sendo esta última a melhor em relação aos tempos de execução.

Não foi sometida a simulação os limitantes superiores obtidos via relaxação Lagrangeana no algoritmo *Branch and Bound* construído para o PMCR, por não ter ainda tempos competitivos com os apresentados com o limitante superior tipo Backtracking (veja-se a sub-seção anterior). Em relação á qualidade dos limitantes, o limitante por relaxação Lagrangeana é melhor no sentido por não ter dependência do tipo de ordenação feito das informações do problema, além de ter "certo controle" da qualidade do limitante superior obtido fazendo manipulação nos parâmetros do algoritmo do sub-gradiente, o que faz que seja atraente ainda como critério de *bounding* para o problema. Mas, ainda é preciso desenvolver outros caminhos na pesquisa de geração destes limitantes para fazer mais eficiente e diminuir o custo computacional para obter-los. No capítulo 7 são visionados propostas de novas direções para desenvolver melhoras do algoritmo *Branch and Bound* para o PMCR proposto neste trabalho de dissertação.

		Modelo Linear							**Modelo Linear Forte**						
		gap da Sol. Ótima			gap da H. das W cap.			$\overline{T}$	**gap da Sol. Ótima**			**gap da H. das W cap.**			$\overline{T}$
	RL	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{gap}$	gap_{min}	gap_{max}		$\overline{gap}$	gap_{min}	gap_{max}	$\overline{gap}$	gap_{min}	gap_{max}	
2/5	R1	6.96%	0.00%	29.36%	19.21%	0.15%	49.42%	15.141	7.03%	0.00%	29.36%	19.30%	0.15%	0.00%	11.443
	R1_2B	2.24%	0.00%	6.53%	13.91%	0.15%	48.55%	0.968	2.10%	0.00%	6.07%	13.74%	0.16%	0.00%	3.523
	R2	0.00%	0.00%	0.02%	11.41%	0.00%	42.65%	0.733	0.00%	0.00%	0.02%	11.41%	0.00%	0.00%	18.442
	R2A	F	F	F	F	F	F	245.518	0.00%	0.00%	0.00%	11.61%	0.00%	0.00%	114.679
	R2B	0.00%	0.00%	0.02%	11.88%	0.00%	42.65%	24.013	0.00%	0.00%	0.00%	11.88%	0.00%	0.00%	48.576
	R3	2.21%	0.00%	6.45%	13.80%	0.26%	47.17%	1058.517	2.13%	0.00%	6.59%	13.72%	0.00%	0.00%	40.906
3/5	R1	4.96%	0.18%	81.33%	8.07%	0.18%	101.27%	13.075	6.02%	0.16%	91.74%	9.21%	0.18%	112.82%	7.878
	R1_2B	3.36%	0.13%	10.80%	6.31%	0.19%	21.44%	1.605	2.65%	0.10%	8.01%	5.59%	0.19%	21.38%	4.211
	R2	1.05%	0.00%	12.96%	4.01%	0.00%	20.23%	1.067	0.84%	0.00%	12.77%	3.79%	0.00%	20.23%	22.765
	R2A	F	F	F	F	F	F	0.000	0.76%	0.00%	12.30%	3.62%	0.00%	20.23%	90.397
	R2B	0.73%	0.00%	10.05%	3.61%	0.00%	20.23%	134.468	0.35%	0.00%	26.80%	3.04%	0.00%	20.23%	45.727
	R3	3.15%	0.00%	10.60%	6.10%	0.00%	21.14%	457.797	2.66%	0.00%	7.42%	5.59%	0.00%	21.11%	32.840
4/5	R1	4.46%	0.15%	20.10%	7.02%	0.15%	22.87%	15.293	5.18%	0.09%	35.49%	8.07%	0.15%	61.89%	8.573
	R1_2B	2.95%	0.81%	9.39%	5.52%	1.04%	24.14%	2.118	1.90%	0.03%	5.62%	4.42%	0.06%	21.18%	4.370
	R2	1.15%	0.00%	12.75%	3.78%	0.00%	20.23%	1.379	0.91%	0.00%	9.54%	3.54%	0.00%	19.48%	5.388
	R2A	F	F	F	F	F	F	0.000	0.86%	0.00%	9.18%	3.36%	0.00%	19.48%	38.372
	R2B	0.98%	0.00%	9.25%	3.98%	0.00%	19.48%	188.059	0.00%	0.00%	0.00%	2.98%	0.00%	19.48%	18.819
	R3	3.15%	0.69%	12.67%	5.72%	0.81%	23.85%	297.147	2.37%	0.28%	7.55%	4.91%	0.34%	22.14%	28.128
	Média	2.17%	0.14%	14.49%	7.51%	0.20%	33.99%	156.874	1.94%	0.05%	15.54%	7.28%	0.08%	24.28%	20.725

Tabela 6.5: Resultados das simulações das categorias de exemplares $q/n = 2/5, 3/5, 4/5$ na relaxação Lagrangeana para as restrições R1, R1-R2B, R2, R2A, R2B e R3 para o Modelo Linear e o Modelo Linear Forte.

7 CONCLUSÕES E TRABALHOS FUTUROS

A pesquisa sob um algoritmo *Branch and Bound* para o PMCR produz dois grandes contribuições, a formulação de um novo modelo (linear e forte) e á criação de algoritmos que compõem um método especializado *Branch and Bound* como método de solução exata, sendo estes um avanço teórico para o problema.

Como principal resultado deste trabalho de dissertação foi o estudo desenvolvido no fortalecimento das informações para o PMCR, sendo este apresentado no Capítulo 4. Daqui destaque-se dois grandes contribuições. O primeiro, a definição de um domínio reduzido para a RLPMC junto á formulação de uma restrição que faz diminuição de soluções factíveis simétricas; e segundo, pela reformulação das restrições do Modelo Linear [20], onde usando os valores ajustados do PMCR e estabelecendo restrições fortes, é formulado um novo modelo linear forte para o PMRC, sendo este mais eficiente do que o Modelo Linear [20] (veja-se os resultados dos experimentos computacionais feitos no Capítulo 6).

Em relação ao algoritmo *Branch and Bound* para o PMCR desenvolvido neste trabalho pode-se destacar três grandes contribuições. Primeiro, a criação de um algoritmo de enumeração sistemática de soluções factíveis para o PMCR, mediante o uso dos algoritmos (6)-(21). Segundo, a formulação de um limitante superior com qualidade aceptável (no sentido da capacidade de poda que tem na árvore de enumeração) para o PMCR formulado em (5.7) e chamado de Limitante Superior tipo *Backtracking* para o PMCR.

Terceiro, o estudo relacionado na geração de limitantes superiores via Relaxação Lagrangeana apresentado no Capítulo 5, destacando-se a continuação os elementos conseguidos aqui: foi feito um algoritmo para resolver o Problema Dual Lagrangeano para o PMCR que provê os limitantes superiores ao problema, apresentado no Algoritmo 31, junto ao estudo e definição de diversas relaxações Lagrangeanas feitas nas restrições do Modelo Linear para o PMCR de [20] e do Modelo Linear Forte para o PMCR, na qual são apresentadas no Apêndice A. Com a diversidade de relaxações Lagrangeanas ao Problema foi obtido limitantes superiores com qualidades aceptáveis (veja-se o Capítulo 6). A pesar de ter uma qualidade favorável dos limitantes superiores obtidos via relaxação Lagrangeana, seu processo de obtenção precisa de um alto esforço computacional, fazendo que este processo não seja (por enquanto) competitivo com os resultados obtidos com o Limitante Superior tipo *Backtracking*. Para mitigar o custo computacional foi desenvolvido uma heurística no Apêndice A nos algoritmos 32, 33 e 34, chamada de Heurística das t melhores e m melhores eficiências para a RLAGPMCR. Com dita heurística foi melhorado os tempos de execução do Método do Subgradiente para o PMCR mas ainda não foi possível obter tempos competitivos.

7.1 TRABALHOS FUTUROS

Na Sub-seção 4.1.3 foi estudado como incrementar o valor de algumas larguras associados aos itens sob condições que não modifiquem a natureza dos dados do PMCR. Faz-se a proposta de estudar á aplicação de *lifting* a todas as larguras associadas aos itens, seguindo as metodologias expostas em [32] e [30], procurando assim obter um domínio mais reduzido do que foi conseguido neste trabalho para a RLPMCR, esperando assim obter uma maior eficiência computacional para as implementações do Modelo Linear Forte.

Com o algoritmo *Branch and Bound* desenvolvido para o PMCR tem-se como proposta continuar com os seguentes estudos: melhorar o processo algorítmico da árvore de enumeração até atingir para qualquer tipo de exemplo do PMCR o $gap = 0$. Uma nova árvore de enumeração pode ser desenvolvida, esta voltada a não ter dependência na ordenação dos itens, para assim obter uma melhor eficiência no processo de poda de limitantes tipo *Bactracking*.

Os limitantes superiores gerados via Relaxação Lagrangeana tem uma ampla gama de estudo para incrementar a qualidade dos mesmos e reduzir o custo computacional em sua reprodução: estudar as implicações de gerar limitantes via Relaxação Lagrangeana com outras árvores de exploração para o PMCR. Desenvolver novas relaxações com a escolha de outras restrições do Modelo Linear de [20] ou do Modelo Linear Forte que foram feitos neste trabalho. Gerar limitantes superiores via relaxação Lagrangeana usando como base o modelo clássico de [18] do PMCR e abrindo a possibilidade de novos estudos de solução exata ao problema. Aplicar outras heurísticas para o cálculo de limitantes inferiores para alimentar o Método Subgradiente na resoulção dos Problemas Duais Lagrangeanos, como por exemplo a heurística *The Decreasing Classes Heuristic* apresentado em [19] e a heurística *Heuristics of* p_k *Strong Capacities* em [34].

Faça-se a proposta também de estudar outros métodos para resolver o problema dual Lagrangeano, como o Método Subgradiente do Radar [4] e uma versão melhorada do método do subgradiente usando como base a relaxação Lagrangeana/Surrogate em [39].

Além da relaxação Lagrangeana usada para o PMCR, outros tipos de relaxação são usados em problemas de programação inteira, sendo assim sugeridos para seu estudo e aplicação no PMCR, a relaxação *surrogate* e a relaxação LAGSUG. A relaxação *surrogate* consiste em reduzir algumas restrições do um PPI em uma só restrição, e a relaxação LAGSUG consiste em uma combinação das relaxações Lagrangeanas e Surrogate. Recomende-se os trabalhos sob estes temas em [11] e [7].

Finalmente, o modelo clássico do PMCR [18] e o Modelo Linear do PMCR [20] podem ser planejados sob alguns tratamentos adicionais como um problema de programação quadrática, tendo assim novos caminhos de pesquisa e propostas de estudo.

REFERÊNCIAS

[1] ARENALES, M., ARMENTANO, V., MORABITO, R., AND YANASSE, H. *Pesquisa operacional: para cursos de engenharia*. Elsevier Brasil, 2015.

[2] BALAS, E., AND TOTH, P. Branch and bound methods for the traveling salesman problem. *Management Science Research* (1983).

[3] BEASLY, J. S. An algorithm for set covering problem. *European Journal of Operational Research* (1987).

[4] BELTRAN, C., AND HEREDIA, F. J. An effective line search for the subgradient method. *Journal of Optimization Theory and Applications 125*, 1 (Apr 2005), 1–18.

[5] CHVÁTAL, V. *Linear programming*. WH Freeman and Company, New York, 1983.

[6] CRUZ, E. P. Uma abordagem heurística linear para mochilas compartimentadas restritas. Dissertação de mestrado em matemática aplicada e computacional, Universidade Estadual de Londrina, 2010.

[7] ESPEJO, L. G. A., AND GALVÃO, R. D. O uso das relaxações lagrangeanas e surrogate em problemas de programação inteira. *Pesquisa Operacional 22* (07 2002), 387 – 402.

[8] FERREIRA, J. S., NEVES, M., AND CASTRO, P. A two–phase roll cutting problem. *European Journal of Operational Research 44* (1990), 185–196.

[9] FISHER, M. L. The lagrangian relaxation method for solving integer programming problems. *Management Science* (1981).

[10] GAU, T., AND WÄSHER, G. Cutgen1: A problem generator for the standar one-dimensional cutting stock problem. *European Jornal of Operational Research 84*, 3 (1995), 572–579.

[11] GONÇALVES NARCISO, M. *A relaxação Lagrangeana/surrogate e algumas aplicações em otimização combinatória*. Tese de Doutorado em computação aplicada, Instituto Nacional de Pesquisas Espaciais, 1998.

[12] GUINARD, M. Lagrange relaxation. *Sociedad de Investigación e Investigación Operativa TOP* (2003).

[13] HAESSLER, R. Solving the two–stage cutting stock problem. *Omega, The International Journal of Management Science 7* (1979), 145–171.

[14] HOTO, R., ARENALES, M., AND MACULAN, N. O problema da mochila compartimentada. Relatório técnico do departamento de matemáticas, Universidade Estadual de Londrina, 1999.

[15] HOTO, R., FENATO, A., YANNASSE, H., MACULAN, N., AND SPOLADOR, F. Uma nova abordagem para o problema da mochila compartimentada. In *Anel do XXXVIII Simpósio brasileiro de Pesquisa Operacional* (2006).

[16] HOTO, R., FENATO, S., AND MARQUES, F. Resolvendo mochilas compartimentadas restritas. In *Anel do XXXVII Simpósio brasileiro de Pesquisa Operacional* (2005).

[17] HOTO, R. S. Otimizaço no corte de peças unidimensionais com restrições de agrupamento. Dissertação de mestrado em matemática aplicada, ICMSC-USP, 1996.

[18] HOTO, R. S. V. *O Problema da Mochila Compartimentada aplicado no corte de Bobinas de aço*. Tese de doutorado em engheneria de sistemas e computação, Universidade Federal do Rio de Janeiro, 2001.

[19] INAREJOS, O., HOTO, R., AND MACULAN, N. An integer linear optimization model to the compartmentalized knapsack problem. *Internacional Transactions in Operational Research* (2017).

[20] INAREJOS, O. F. Sobre a não linearidade do problema da mochila compartimentada. Dissertação de mestrado em matemática aplicada e computacional, Universidade Estadual de Londrina, 2015.

[21] LAND, A. H., AND DOIG, A. G. An automatic method of solving discrete programming problems. *Econometrica* (1960).

[22] LEÃO, A. A., SANTOS, M. O., HOTO, R., AND ARENALES, M. N. The constrained compartmentalized knapsack problem: mathematical models and solution methods. *European Journal of Operational Research 212*, 3 (2011), 455 – 463.

[23] LEÃO A., A. S. *Extensões em problemas de corte: padrões compartimentados e problemas acoplados*. Tese de doutorado em ciências da computação e matemática computacional, Universidade de São Paulo, 2013.

[24] LITTLE, J. D. C., MURTY, K. G., SWEENEY, D. W., AND KAREL, C. An algorithm for the traveling salesman problem. *Operations Research 11*, 6 (1963), 972–989.

[25] MACULAN, N., AND COSTA F., M. H. *Otimização Linear*. John Wiley & Sons, Inc., 1990.

[26] MARQUES, F. *O problema da mochila compartimentada e aplicações*. Tese de doutorado em ciências da computação e matemática computacional, Universidade de São Paulo, 2000.

[27] MARQUES, F., AND ARENALES, M. N. O Problema da mochila compartimentada e aplicacações. *Pesquisa Operacional 22* (07 2002), 285 – 304.

[28] MARQUES, F., AND ARENALES, M. N. The constrained compartmentalised knapsack problem. *Computers & Operations Research 34*, 7 (2007), 2109 – 2129.

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

[30] MARTIN, R. K. *Large Scale Linear and Integer Optimization*. Kluwer Academic Publishers, 1999.

[31] NAVA MANZO, R. A. *Funciones convexas no diferenciables*. Tese de mestrado em matemática aplicadas e industriais, Universidad Autónoma Metropolitana Unidad Iztapalapa, 2015.

[32] NEMHAUSER, G. L., AND WOLSEY, L. A. *Integer and Combinatorial Optimization*. Wiley Interscience in Discrete Mathematics nad Optimization, 1988.

[33] PEREIRA, M. A. Uma abordagem matemática para o problema de corte e laminação de fitas de aço. Dissertação de Mestrado em Matemática Aplicada, Unicamp, 1993.

[34] QUIROGA OROZCO, J. J., VALERIO DE CARVALHO, J. M., AND HOTO V., R. S. A strong integer linear optimization model to the compartmentalized knapsack problem, 2018. No prelo.

[35] R., H., MACULAN, N., MARQUES, F., AND ARENALES, M. Um problema de corte com padrões compartimentados. *Pesquisa Operacional 23* (01 2003), 169 – 187.

[36] RUST, R. Um algoritmo branch and bound para resolução de problemas de localização capacitados. Dissertação de mestrado, Universidade Federal do Rio, 1983.

[37] VALERIO DE CARVALHO, J. M., AND GUIMARAES RODRIGUES, A. A computer based interactive approach to a two-stage cutting stock problem. *INFOR: Information Systems and Operational Research 32*, 4 (1994), 243–252.

[38] VALÉRIO DE CARVALHO, J. M., AND RODRIGUES, A. An LP based approach to a two–phase cutting stock problem. *European Journal of Operational Research 84* (1995), 580–589.

[39] WANG, W., LUH, P. B., YAN, J. H., AND STERN, G. A. An improved lagrangian relaxation method for discrete optimization applications. In *Power and Energy Society General Meeting 2012 IEEE,* (2008).

APÊNDICE

A ESTUDOS COMPUTACIONAIS DOS LIMITANTES SUPERIORES GERADOS VIA RELAXAÇÃO LAGRANGEANA PARA O PMC

A.1 LIMITANTES SUPERIORES VIA RELAXAÇÃO LAGRANGEANA PARA O PMCR COM O MODELO LINEAR [20]

O capítulo 5 foi dedicado ao estudo de dois grandes tipos de limitantes superiores para o PMCR, o primeiro denominado de Limitante Superior Clássico tipo *Bactracking* inspirado no limitante superior para o *Branch and Bound* do PMR e PMI (veja-se o Capítulo 2 e [5]) e, o segundo, pela geração de subproblemas Lagrangeanos relaxando o PMCR. O Limitante Superior Clássico tipo *Bactracking* formulado em (5.7), é um limitante com um custo computacional baixo, no sentido que se aproveita só de informações conhecidas desenvolvidas na árvore de enumeração, mas foi discutido que, pela ordenação estabelecida para fixar os nós das árvore para o PMCR e a dependência desta ordenação pelo limitante, sua capacidade de poda não será tão efetiva como é o limitante no PMI (veja-se o Capítulo 5). O anterior motiva o estudos de novos limitantes superiores para o problema, na qual pela sugestão do relatório técnico de [14], encaminha-se o estudo da relaxação Lagrangeana. Com a relaxação Lagrangeana para o PMCR tem-se o intuito de gerar bons limitantes superiores e obter um melhor comportamento de podas na árvore em relação ao Limitante Superior Clássico tipo *Bactracking*.

O estudo da relaxação Lagrangeana para o PMCR foi iniciado na Subseção 5.2.2 onde, seguindo o critério de escolher a restrição (o um conjunto delas) que faz o PMCR mais difícil de se resolver, foi relaxado Lagrangeanamente a restrição R2 do problema, formulando a Relaxação Lagrangeana para o PMCR na restrição R2 em (5.27)-(5.31), veja-se as notações do PMCR na subseção 5.2.2. Para avaliar a relaxação Lagrangeana feita em R2 do PMCR, foram criados dois exemplares de controle com a intenção de observar em tempos práticos (menos de um minuto por iteração) o comportamento computacional de dita relaxação. O primeiro exemplo, que será chamado de Exemplar de Controle 1, é um exemplo com 2 classes e com 5 itens para cada classe e o segundo exemplo, chamado de Exemplar de Controle 2, onde tem 5 classes com 10 itens em cada uma delas. Na alvorada deste trabalho de dissertação, foi usado como parâmetros das informações dos exemplares, os que são apresentados em [20], sendo como capacidade da mochila $L = 2200$, com facas $F_1 = 12$ e $F_2 = 8$, valores aleatórios para as larguras, utilidades e demandas via [10] com limites das capacidades dos compartimentos da mochila para todas as classes k com $L^k_{min} = 375$ e $L^k_{max} = 750$ (pode-se ver no Capítulo 6 que foram usados parâmetros realísticos extraídos do trabalho de [18]). Os exemplares de controles foram solucionados usando o modelo linear de [20] obtendo o seguinte: a solução ótima do Exemplar de Controle 1 , foi de 6.284 com um tempo de execução de 0.265 e para o Exemplar de Controle 2 foi de 20.976 com um tempo de execução de 0.424.

Então, usando-se o Exemplar de Controle 1 para avaliar a Relaxação Lagrangeana do PMCR na restrição R2 formulado em (5.27)-(5.31), com Problema Dual Lagrangeano definido em (5.32), foi resolvido usando o Método do Subgradiente desenvolvido no Algoritmo 27 com parâmetros de implementação $\epsilon_0 = 2$, com máximo 100 iterações ($n_{iter} = 100$) e controle de mudança de ϵ_0 a 10 iterações ($cc_\epsilon = 10$); A Tabela A.1 apresenta os resultados da implementação.

Pode-se ver na Tabela A.1 que efetivamente tem-se um limitantes superior em relação á solução ótima do Exemplar de Controle 1, com um valor de 6.866, mas, o processo iterativo não convergiu a um melhor valor em relação à primeira iteração, sendo assim, o limitante superior obtido via relaxação de R2 para o PMCR não é um bom limitante para o problema com os parâmetros estipulados pensando-se em aplicar para qualquer teste.

Pelo anterior surge a questão de examinar aos outras restrições do PMCR, agora voltando-se a ter processos de convergência no Método do Subgradiente e ter melhores limitantes em comparação com o obtido com a relaxação de R2. Da Restrição R2 foi separa em duas restrições, R2A e R2B (ver notação em 5.2.2), para aplicar relaxações Lagrangeanas de forma separada. Foi incluído para relaxar as restrições R1 e R3. Não foi escolhidas as restrições R4 e R5 no estudo por não gerar para alguns testes limitantes superiores.

Então, com as restrições selecionadas, R1, R2A, R2B e R3, formule-se as respectivas relaxações Lagrangeanas, define-se o problema Dual Lagrangeanos para cada um junto ao vetor subgradiente e definição do tamanho do passo, para em seguida usar novamente o método do subgradiente, especialmente o Algoritmo 31) e obter os novos Limitantes, que são obtidos nas seguentes formulações.

Parâmetros						
$\epsilon_0 = 2$		$n_{iter} = 100$		$cc_\epsilon = 10$		
In. iterac.	continuação	cont.	cont.	cont.	cont.	cont.
6.866	7.9359	7.23575	6.99984	6.92348	6.89857	6.88822
7.08673	7.91486	7.1565	7.01977	6.92592	6.89743	6.88753
7.53747	7.91159	7.19301	7.01001	6.91835	6.897	6.88903
7.69284	7.72584	7.17698	6.96176	6.92046	6.90086	6.88538
8.00993	7.79294	7.14216	6.9675	6.91225	6.89435	6.8857
8.20437	7.71508	7.11331	6.9685	6.91347	6.8943	6.88502
8.42424	7.64121	7.09387	6.93657	6.9248	6.89201	6.88506
8.55181	7.84627	7.04269	6.94477	6.91338	6.89162	6.88406
8.68248	7.44305	7.05853	6.95222	6.90532	6.89016	6.88499
8.74242	7.49481	7.00872	6.94512	6.9004	6.88926	6.88408
8.79231	7.32947	7.00847	6.93598	6.89792	6.88914	6.88508
8.33515	7.21788	7.03024	6.93978	6.90119	6.88946	6.88373
8.39281	7.26489	7.02469	6.93602	6.89705	6.8894	6.88393
Z_{lsup}				6.866		
Tempo de execução				5.423		

Tabela A.1: Teste com $n_{iter} = 100$ para RLAGPMCR$_\theta$_R2 com o Exemplar de Controle 1

(RLAGPMCR$_\lambda$_R1) Maximizar:

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} u_i^k a_{ij}^k + \lambda\left(L - \sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} l_i^k a_{ij}^k\right) =$$

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k}\left(u_i^k - \lambda l_i^k\right) a_{ij}^k + \lambda L \tag{A.1}$$

sujeito a: (A.2)

$$\delta_j^k L_{min}^k \leq \sum_{i\in N_k} l_i^k a_{ij}^k \leq \delta_j^k L_{max}^k \text{ com } j = 1, ..., p_k, k = 1, ..., q$$

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k,\ i \in N_k,\ k = 1, ..q \tag{A.3}$$

$$\sum_{k=1}^{q}\sum_{j=1}^{p_k} \delta_j^k \leq F_1 \tag{A.4}$$

$$\sum_{i\in N_k} a_{ij}^k \leq F_2,\ j = 1, ..., p_k, k = 1, ..., q \tag{A.5}$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ e inteiros, } i \in N_k,\ j = 1, ..., p_k,\ k = 1, ..., q \tag{A.6}$$

Seja $\lambda \in \mathbb{R}^+$. Define-se o Problema Dual Lagrangeano, RLAGPMCR para R1, como:

$$(\text{RLAGPMCR\_R1})\ \underset{\theta\geq 0}{min}\ v(\text{RLAGPMCR}_\theta\text{\_R1}) \tag{A.7}$$

Define-se o vetor subgradiente e o tamanho do paso para o RLAGPMCR_R1 para que serem aplicados no Método do Subgradiente como:

$$G = L - \sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} l_i^k X_{ij}^k \tag{A.8}$$

Os valores que o subgradiente X_{ij}^k é soluções da variavel a_{ij}^k para $k = 1, ..., q$, $i \in N_k$ e $j = 1, .., p_k$, correspondente do Problema Lagrangeano Dual (A.7). O tamanho do passo $t_{\bar{x}}$ vão estar definido como:

$$t_{\bar{k}} = \frac{\epsilon_{\bar{k}} \cdot (Z_{lsup} - Z_{linf})}{\sum_{k=1}^{q}\sum_{j=1}^{p_k}(G^2)} \tag{A.9}$$

Antes de que serem aplicado o Método do Subgradiente, pode-se observa da função

objetivo de RLAGPMCR_R1 que os multiplicador Lagrangeano desejados λ devem ter forma que $u_i^k - \lambda l_i^k \geq 0$, procurando que o coeficiente da variável de decisão seja positiva e não faça que este valor poda decrescer o valor da função objetivo com a solução de X_{ij}^k. Então, define-se:

$$C_i^k = u_i^k - \lambda_i^k \tag{A.10}$$

Onde, para os $k = 1, .., q$ e $i \in N_k$ que tem que $C_i^k = 0$, então faça na solução de RLAGPMCR_R1 que $X_{ij}^k = 0$. A condição (A.10) será considerada quando tem-se que alimentar o Método do Subgradiente com a solução do RLAGPMCR$_\lambda$ (veja-se o Algoritmo 31).

(RLAGPMCR$_\lambda$_R2A) Maximizar:

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} u_i^k a_{ij}^k + \sum_{k=1}^{q}\sum_{j=1}^{p_k}\lambda_j^k\left(\sum_{i\in N_k} l_i^k a_{ij}^k - \delta_j^k L_{min}^k\right) =$$

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k}\left(u_i^k + l_i^k\lambda_j^k\right)a_{ij}^k - \sum_{k=1}^{q}\sum_{j=1}^{p_k}\delta_j^k\lambda_j^k L_{min}^k \tag{A.11}$$

sujeito a:

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} l_i^k a_{ij}^k \leq L \tag{A.12}$$

$$\sum_{i\in N_k} l_i^k a_{ij}^k \leq \delta_j^k L_{max}^k \text{ com } j = 1, ..., p_k, k = 1, ..., q \tag{A.13}$$

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k,\ i \in N_k,\ k = 1, ..q \tag{A.14}$$

$$\sum_{k=1}^{q}\sum_{j=1}^{p_k}\delta_j^k \leq F_1 \tag{A.15}$$

$$\sum_{i\in N_k} a_{ij}^k \leq F_2,\ j = 1, ..., p_k, k = 1, ..., q \tag{A.16}$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ e inteiros, } i \in N_k,\ j = 1, ..., p_k,\ k = 1, ..., q \tag{A.17}$$

Seja $\lambda \in \mathbb{R}^m$ e $m = \sum_{k=1}^{q} p_k$. Define-se o Problema Dual Lagrangeano, RLAGPMCR para R2A, como:

$$(\text{RLAGPMCR\_R2A})\ \min_{\lambda\geq 0} v(\text{RLAGPMCR}_\lambda\text{\_R2A}) \tag{A.18}$$

Define-se o vetor subgradiente e o tamanho do paso para o RLAGPMCR_R2A para que serem aplicados no Método do Subgradiente para Funções Convexas, como segue a continuação:

$$G_j^k = \sum_{i \in N_k} l_i^k X_{ij}^k - DEL_j^k L_{min}^k \tag{A.19}$$

Os valores que compõem o vetor subgradiente X_{ij}^k e DEL_j^k são as soluções das variaveis respectivamente a_{ij}^k e δ_j^k para $k = 1, ..., q$, $i \in N_k$ e $j = 1, .., p_k$, correspondente do Problema Lagrangeano Dual (A.18). O tamanho do passo $t_{\bar{x}}$ vão estar definido como:

$$t_{\bar{k}} = \frac{\epsilon_{\bar{k}} \cdot (Z_{lsup} - Z_{linf})}{\sum_{k=1}^{q} \sum_{j=1}^{p_k} \left(\left(G_j^k \right)^2 \right)} \tag{A.20}$$

(RLAGPMCR$_\lambda$_R2B) Maximizar:

$$\sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} u_i^k a_{ij}^k + \sum_{k=1}^{q} \sum_{j=1}^{p_k} \lambda_j^k \left(\delta_j^k L_{max}^k - \sum_{i \in N_k} l_i^k a_{ij}^k \right) =$$

$$\sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} \left(u_i^k - l_i^k \lambda_j^k \right) a_{ij}^k + \sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta_j^k \lambda_j^k L_{max}^k \tag{A.21}$$

sujeito a:

$$\sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} l_i^k a_{ij}^k \leq L \tag{A.22}$$

$$\delta_j^k L_{min}^k \leq \sum_{i \in N_k} l_i^k a_{ij}^k \text{ com } j = 1, ..., p_k, k = 1, ..., q \tag{A.23}$$

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k, \, i \in N_k, \, k = 1, ..q \tag{A.24}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta_j^k \leq F_1 \tag{A.25}$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2, \, j = 1, ..., p_k, k = 1, ..., q \tag{A.26}$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ e inteiros, } i \in N_k, \, j = 1, ..., p_k, \, k = 1, ..., q \tag{A.27}$$

Seja $\lambda \in \mathbb{R}^m$ e $m = \sum_{k=1}^{q} p_k$. Define-se o Problema Dual Lagrangeano, RLAGPMCR para R2B, como:

$$(\text{RLAGPMCR\_R2B}) \underset{\lambda \geq 0}{min}\, v(\text{RLAGPMCR}_{\lambda}\text{\_R2B}) \tag{A.28}$$

Define-se o vetor subgradiente e o tamanho do paso para o RLAGPMCR_R2B para que serem aplicados no Método do Subgradiente para Funções Convexas, como segue a continuação:

$$\overset{\eta}{G}{}_j^k = DEL_j^k L_{max}^k - \sum_{i \in N_k} l_i^k X_{ij}^k \tag{A.29}$$

Os valores que compõem o vetor subgradiente X_{ij}^k e DEL_j^k são as soluções das variáveis respectivamente a_{ij}^k e δ_j^k para $k = 1, ..., q$, $i \in N_k$ e $j = 1, .., p_k$, correspondente do Problema Lagrangeano Dual (A.28). O tamanho do passo $t_{\bar{x}}$ vão estar definido como:

$$t_{\bar{k}} = \frac{\epsilon_{\bar{k}} \cdot (Z_{lsup} - Z_{linf})}{\sum_{k=1}^{q} \sum_{j=1}^{p_k} \left(\left(\overset{\lambda}{G}{}_j^k \right)^2 + \left(\overset{\eta}{G}{}_j^k \right)^2 \right)} \tag{A.30}$$

(RLAGPMCR$_{\lambda}$_R3):

$$\text{Maximizar: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} u_i^k a_{ij}^k + \sum_{k=1}^{q} \sum_{i \in N_k} \lambda_i^k \left(d_i^k - \sum_{j=1}^{p_k} a_{ij}^k \right) =$$
$$\sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} (u_i^k - \lambda_i^k) a_{ij}^k + \sum_{k=1}^{q} \sum_{i \in N_k} \lambda_i^k u_i^k \tag{A.31}$$

sujeito a:

$$\sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} l_i^k a_{ij}^k \leq L \tag{A.32}$$

$$\delta_j^k L_{min}^k \leq \sum_{i \in N_k} l_i a_{ij}^k \leq \delta_j^k L_{max}^k \text{ com } j = 1, ..., p_k, k = 1, ..., q \tag{A.33}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta_j^k \leq F_1 \tag{A.34}$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2, \, j = 1, ..., p_k, k = 1, ..., q \tag{A.35}$$

$$\delta_j^k \in \{0, 1\}\,, \, a_{ij}^k \geq 0 \text{ e inteiros, para } i \in N_k, j = 1, ..., p_k, k = 1, ..., q \tag{A.36}$$

$$G_i = d_i - \sum_{j=1}^{p_k} X_{ij}^k, \text{ com } k = 1, ..., q \text{ e, } i \in N_k \tag{A.37}$$

Onde X_{ij}^k, para $i \in N_k$, $j = 1, ..., p_k$, $k = 1, ..., q$ é a solução ótima do RLAGPMCR$_\lambda$_R3 para certo λ. O tamanho do passo (t_k) esta definido por:

$$t_{\bar{k}} = \frac{\epsilon_{\bar{k}} \cdot (Z_{lsup} - Z_{linf})}{\sum\limits_{i \in N_k} G_i^2} \tag{A.38}$$

$$C_i^k = u_i^k - \lambda_i^k \tag{A.39}$$

Com os Problemas Duais Lagrangeanos definidos, foi feita um teste de controle com o Exemplar de Controle 1 para avaliar o comportamento dos limitantes superiores que está gerando com o uso do Algoritmo do Método do Subgradiente 31. Os parâmetros usados para o teste foi: 100 iterações ($n_{iter} = 100$), $\epsilon_0 = 2$ e controle de de cambio de ϵ a 10 . Pode-se observar os resultados na Tabela A.2:

	Solução Ótima	R1	R2	R2A	R2B	R3
Tempo	0.265	14.564	11.798	24.47	40.002	16.329
Resultado	6.284	6.52754	6.84835	6.82163	6.40002	6.77113
n_{iter}	-	100	100	100	100	100
ϵ_0	-	2	2	2	2	2
$cc_\epsilon = 10$	-	10	10	10	10	10

Tabela A.2: Teste controle para RLAGPMCR$_\lambda$ com Exemplar de Controle 1

Na Tabela A.2 pode-se observar que, o Dual Lagrangeano para o R2A tem um comportamento similar com a restrição R2. As restrições relaxadas R1 e R2B são com os Problemas Duais Lagrangeanos quem gerou os melhores limitantes para o Exemplar de Controle 1, mas, tem tempos de execução, que não são por enquanto, muito favoráveis (especialmente a restrição R2B). Em seguida, tem-se como ideia relaxar as restrições R1 e R2B, aproveitando-se com o objetivo de estudar o comportamento do limitante superior que poderia gerar e as possibilidades de melhorar os tempos de execução. Então, escolhe-se as restrições R1 e R2B para que serem aplicadas a Relaxação Lagrangeana, tendo como modelo da relaxação, o seguinte:

(RLAGPMCR$_\lambda$_R1-R2B) Maximizar:

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} u_i^k a_{ij}^k + \lambda\left(L - \sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} l_i^k a_{ij}^k\right) + \sum_{k=1}^{q}\sum_{j=1}^{p_k}\eta_j^k\left(\delta_j^k L_{max}^k - \sum_{i\in N_k} l_i^k a_{ij}^k\right) =$$

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k}\left(u_i^k - (\lambda+\eta_j^k)l_i^k\right)a_{ij}^k + \sum_{k=1}^{q}\sum_{j=1}^{p_k}\delta_j^k\eta_j^k L_{max}^k + \lambda L \tag{A.40}$$

sujeito a: (A.41)

$$\delta_j^k L_{min}^k \leq \sum_{i\in N_k} l_i^k a_{ij}^k \text{ com } j = 1,...,p_k, k = 1,...,q \tag{A.42}$$

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k,\ i \in N_k,\ k = 1,..q \tag{A.43}$$

$$\sum_{k=1}^{q}\sum_{j=1}^{p_k}\delta_j^k \leq F_1 \tag{A.44}$$

$$\sum_{i\in N_k} a_{ij}^k \leq F_2,\ j = 1,...,p_k, k = 1,...,q \tag{A.45}$$

$$\delta_j^k \in \{0,1\} \text{ e } a_{ij}^k \geq 0 \text{ e inteiros, } i \in N_k,\ j = 1,...,p_k,\ k = 1,...,q \tag{A.46}$$

Seja $\theta \in \mathbb{R}^{m+1}$ e $m = \sum_{k=1}^{q} p_k$ é o vetor de multiplicadores Lagrangeanos, com $\theta = (\lambda\ \eta) \geq 0$, $\lambda \in \mathbb{R}$ e $\eta \in \mathbb{R}^m$. Define-se o Problema Dual Lagrangeano, RLAGPMCR para R1-R2B, como:

$$(\text{RLAGPMCR\_R1-R2B})\ \min_{\theta\geq 0}\ v(\text{RLAGPMCR}_\theta\text{\_R1-R2B}) \tag{A.47}$$

Define-se o vetor subgradiente e o tamanho do paso para o RLAGPMCR_R1-R2B para que serem aplicados no Método do Subgradiente para Funções Convexas, como segue a continuação:

Define-se o sub-gradiente G como $G = \begin{pmatrix} \overset{\lambda}{G} & \overset{\eta}{G} \end{pmatrix}$ onde:

$$\overset{\lambda}{G} = L - \sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_k} l_i^k a_{ij}^k \tag{A.48}$$

$$\overset{\eta}{G}{}_j^k = DEL_j^k L_{max}^k - \sum_{i\in N_k} l_i^k X_{ij}^k \tag{A.49}$$

Os valores que compõem o vetor subgradiente X_{ij}^k e DEL_j^k são as soluções das variáveis respectivamente a_{ij}^k e δ_j^k para $k = 1, ..., q$, $i \in N_k$ e $j = 1, .., p_k$, correspondente do Problema Lagrangeano Dual (A.47). O tamanho do passo $t_{\bar{x}}$ vão estar definido como:

$$t_{\bar{k}} = \frac{\epsilon_{\bar{k}} \cdot (Z_{lsup} - Z_{linf})}{\sum_{k=1}^{q} \sum_{j=1}^{p_k} \left(\left(\overset{\lambda}{G} \right)^2 + \left(\overset{\eta\, k}{G_j} \right)^2 \right)} \tag{A.50}$$

Novamente faz-se testes com o Exemplar de Controle 1. Os testes foram feitos da seguinte forma: cuatro testes para com diferentes número de iterações (n_{iter}), para $n_{iter} = 50$, $n_{iter} = 100$, $n_{iter} = 150$ e $n_{iter} = 200$ com os problemas Lagrangeanos definidos até o momento, e com parâmetros: $n_{iter} = 50, 100, 150, 200$), $\epsilon_0 = 2$ e controle de de cambio de ϵ a 10, obtendo resultados expostos nas tabelas A.3-A.6.

Um primeiro resultado que pode-se observar nas tabelas A.3-A.6, é que tem-se bons candidatos a que serem Limitantes Superiores (falando do tipo de problema Lagrangeano a serem usados). O comportamento não convergente com as restrições R2 e R2A pode-se considerar como restrições que geram limitantes não na qualidade procurada. Em relação ao limitante feito pelas restrições R1 e R2B, gera limitantes superior, não o melhor em comparação aos outros, mas com um tempo aceitável a nível dos testes (no caso do Exemplar de Controle 1, tem bom comportamento para $n_{iter} \leq 200$ comparando as outras restrições). Por ventura, relaxando R1 e R2B é um candidato por enquanto pensando-se incluir na árvore de enumeração do *Branching and Bound*.

	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
Tempo	0.265	4.514	3.632	4.55	10.805	16.174	5.985
Resultado	6.284	6.53281	6.75288	6.866	6.84179	6.40719	6.77113
n_{iter}	-	50	50	50	50	50	50
ϵ_0	-	2	2	2	2	2	2
$cc_\epsilon = 10$	-	10	10	10	10	10	10

Tabela A.3: Teste para $n_{iter} = 50$ para RLAGPMCR$_\lambda$ com Exemplar de Controle 1

	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
Tempo	0.265	14.564	8.739	11.798	24.47	40.002	16.329
Resultado	6.284	6.52754	6.61924	6.84835	6.82163	6.40002	6.77113
n_{iter}	-	100	100	100	100	100	100
ϵ_0	-	2	2	2	2	2	2
$cc_\epsilon = 10$	-	10	10	10	10	10	10

Tabela A.4: Teste para $n_{iter} = 100$ para RLAGPMCR$_\lambda$ com Exemplar de Controle 1

-	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
Tempo	0.265	20.458	15.24	12.999	40.249	70.56	31.787
Resultado	6.284	6.52754	6.56632	6.84835	6.81894	6.39951	6.77113
n_{iter}	-	150	150	150	150	150	150
ϵ_0	-	2	2	2	2	2	2
$cc_\epsilon = 10$	-	10	10	10	10	10	10

Tabela A.5: Teste para $n_{iter} = 150$ para RLAGPMCR$_\lambda$ com Exemplar de Controle 1

	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
Tempo	0.265	18.278	23.835	13.292	39.414	83.282	51.641
Resultado	6.284	6.52754	6.56472	6.84835	6.81894	6.39951	6.77113
n_{iter}	-	200	200	200	200	200	200
ϵ_0	-	2	2	2	2	2	2
$cc_\epsilon = 10$	-	10	10	10	10	10	10

Tabela A.6: Teste para $n_{iter} = 200$ para RLAGPMCR$_\lambda$ com Exemplar de Controle 1

Seguido as anteriores observações, faz-se um segundo grupo de testes com um exemplo maior, com Exemplar de Controle 2, veja-se as condições mencionadas no inicio do capítulo, com $q = 5$ e $|N_k| = 10$ para cada classe e parâmetros: 25, 50 e 100 iterações ($n_{iter} = 25, 50, 100$), $\epsilon_0 = 2$ e controle de de cambio de ϵ a 10 iterações. Lembe-se que a solução ótima do Exemplar de Controle 2 foi de 20.973 com um tempo de execução de $0, 422$. Os testes com o Exemplar de Controle 2 foram feitos com n_{iter} para $n_{iter} = 25$, $n_{iter} = 50$, $n_{iter} = 100$ e $n_{iter} = 100$ com os problemas Lagrangeanos definidos até o momento, obtendo os seguintes resultados:

	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
Tempo	0.422	F	7.283	3.477	F	18.907	F
Resultado	20.973	F	22.3115	21.731	F	21.731	F
n_{iter}	-	25	25	25	25	25	25
ϵ_0	-	2	2	2	2	2	2
$cc_\epsilon = 10$	-	10	10	10	10	10	10

Tabela A.7: Teste para $n_{iter} = 25$ para RLAGPMCR$_\lambda$ com Exemplar de Controle 2

	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
Tempo	0.422	F	9.251	6.655	F	F	F
Resultado	20.973	F	21.6829	21.731	F	F	F
n_{iter}	-	50	50	50	50	50	50
ϵ_0	-	2	2	2	2	2	2
$cc_\epsilon = 10$	-	10	10	10	10	10	10

Tabela A.8: Teste para $n_{iter} = 50$ para RLAGPMCR$_\lambda$ com Exemplar de Controle 2

-	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
Tempo	0.422	F	19.063	F	F	F	F
Resultado	20.973	F	21.5183	F	F	F	F
n_{iter}	-	100	100	100	100	100	100
ϵ_0	-	2	2	2	2	2	2
$cc_\epsilon = 10$	-	10	10	10	10	10	10

Tabela A.9: Teste para $n_{iter} = 100$ para RLAGPMCR$_\lambda$ com Exemplar de Controle 2

Nas tabelas anteriores foi assinalado com F os resultados que tiveram tempo muito ruim nos testes (foram muito devagar na solução do problema Lagrangeano, com tempos de mais de 120 segundos em só uma iteração). Pode-se observar que o comportamento da relaxação das restrições R1, R2 e R3 não apresentam um processo de execução desejado, mas, a relaxação de R1 e R2B nos quatro momentos do teste, sempre apresentaram resultados computacionais aceptáveis para os testes. Desta forma, surge o candidato a serem o Limitante Superior para o PMCR.

Agora, a questão é como melhorar os tempos de execução e se aproximar com tempos de execução de referencia do modelo linear de [20] na implementação com FICO®Xpress Optimizer. Pensa-se alguma forma de resolver mais rápido o problema dual RLAGPMCR.

Para o anterior, foi definido uma heurísticas em relação as informações do problema (classes e eficiências), denominada *Heurística das t melhores classes e m melhores eficiências*.

A idea da Heurística das t melhores classes e m melhores eficiências é simples, em lugar de resolver o problema Lagrangeano RLAGPMCR$_\lambda$ com todas as informações do problema, ou seja, com as q classes e os $|N_k|$ itens para cada classe $k = 1, .., q$, escolhe-se convenientemente (e pelo fato da ordenação feita no problema das classes e seus itens) as melhores t classes e, para cada classe, as melhores m eficiências. Apresenta-se no Algoritmo 34 a Heurística das t melhores classes e m melhores eficiências para o PMCR (veja-se as notações usadas no capitulo 5 na criação da árvore de enumeração).

Em seguida da definição de uma heurística e com as relaxações Lagrangeanas feitas, faça novos testes com o Exemplar de Controle 1. Tem-se como parâmetros para o teste como $t = q = 2$ e $m = 4, 3, 2$. Os resultados são apresentados na Tabela A.10.

Com o anterior teste, pode-se observar que tem-se uma melhoria nos tempos de execução, também pode-se ver que com a heurística, tem-se uma relativa melhoria nos limitantes achado. Procura-se avaliar para exemplares mais grandes o comportamento dos problemas Lagrangeanos Duais com a Heurística definida para sua resolução, mas a agora será só estudado as restrições que apresentam um comportamento aceptável desde os primeiros testes, no sentido em solução e tempo de execução, as restrições R1 e R1_2B.

Algoritmo 31: Controle m melhores eficiências

Entrada: N, u_i^k, l_i^k, d_i^k, L, L_{max}^k, L_{min}^k, r !r é a posição do nó.
Saída: Z_{lsup_λ}
! Melhores m eficiências
se $fecha\_comp(comp - r < m)$ **então**
  se $fecha\_comp(comp) - r > 0$ **então**
    $N_{k_m}(ordemclasses(classe))) := fecha\_comp(comp) - r$
  senão
    $N_{k_m}(ordemclasses(classe)) := 1$
  fim
fim
se $classe$<>q **então**
  para todo $k = classe + 1, ..., q$ **faça**
    $N_{k_m}(ordemclasses(k)) := m$
  fim
senão
  para todo $k = classe, ..., q$ **faça**
    $N_{k_m}(ordemclasses(k)) := m$
  fim
fim

Fonte: Própio Autor

Algoritmo 32: Controle t melhores classes

Entrada: N, u_i^k, l_i^k, d_i^k, L, L_{max}^k, L_{min}^k, r !r é a posição do nó.

Saída: Z_{lsup_λ}

Inicialização: crie-se a matriz ITENS.

!Melhores t classes.

se $q = classe$ **então**
| $ajustclass := q$

senão
| **se** $q - classes + 1 < t$ **então**
| | $ajustclass := (q - classe) + 1$
| **senão**
| | $ajustclass := t$
| **fim**

fim

Faça: $itens := \{\}$;

para todo $k = classe, ..., ajustclass$ **faça**
| $ITENS+ = \{ordemclasses(k)\}$

fim

Fonte: Própio Autor

Algoritmo 33: Heurística das t melhores classes e m melhores eficiências

Entrada: N, u_i^k, l_i^k, d_i^k, L, L_{max}^k, L_{min}^k, r !r é a posição do nó.

Saída: Z_{lsup_λ}

Inicialização: restringe-se os subconjntos de itens de N como N_{k_m} e define-se novos um_i^k, lm_i^k, dm_i^k; !Com as informações requeridas das t classes e as m eficiências.

Faça Algoritmo 32.

Faça Algoritmo 33.

Resolva o RLAGPMCR$_\lambda$ usando um_i^k, lm_i^k, dm_i^k.

Fonte: Própio Autor

Usa-se novamente o Exemplar de Controle 2 ($q = 5$ e $|N_k| = 10$ para cada classe) para desenvolver os ensaios numéricos, inicia-se o primeiro teste estudando as melhores classes e em seguida as melhores eficiências, como se apresenta nas tabelas A.11 e A.12.

Veja-se na Tabela A.11, que com o ajuste heurístico das classes, tem-se a possibilidade de ter resultados para a relaxação de R1, mas o resultado não e um limitante superior para o Exemplar de Controle 2 (este tipo de soluções infrutuosas vão-se representar com FF). Para a relaxação do R1_2B, veja-se que em todo momento continua tendo resultados, mas só provê um limitante superior quando é trabalhado com a heurística a quatro classes.

Solução Ótima	PAR.	R1	R1_2B	R2	R2A	R2B	R3
	n_{iter}	50	50	50	50	50	50
Tempo	ϵ_0	2	2	2	2	2	2
	$cc_\epsilon = 10$	10	10	10	10	10	10
0.265	Tempo	3.622	2.77	3.622	29.738	12.426	16.131
	4-m-ef	6.87711	6.83071	6.866	6.83322	6.50024	6.74004
Solução	Tempo	3.893	3.455	3.463	26.368	FF	15.798
	3-m-ef	6.80039	6.52235	6.866	6.84231	FF	6.74004
6.284	Tempo	3.077	3.022	3.126	12.158	FF	9.106
	2-m-ef	6.31997	6.41251	6.866	6.82966	FF	6.74004

Tabela A.10: Teste RLAGPMCR$_\lambda$ com $n_{iter} = 50$ e m-ef sob Exemplar de Controle 1.

Com os resultados na Tabela A.12, voltados na heurística nas melhores eficiências, tem-se certamente limitantes superiores, com melhores tempos de execução para os modelos Lagrangeanos já apresentados. Aqui, em caso de se trabalhar com a heurista, seja com as melhores classes ou melhores eficiências, surge o problema de escolher de forma adequada a quantidade de classes e de eficiências para cada exemplo, certamente para gerar um bom limitante superior (procurando não obter soluções infrutuosas) e com o melhor tempo de excussão possível.

Solução Ótima	Tempo	0.424
	Solução	20.976

PAR.	R1	R1_2B
n_{iter}	50	50
ϵ_0	2	2
$cc_\epsilon = 10$	10	10
Tempo	F	9.061
4-m-c	F	21.3562
Tempo	F	FF
3-m-c	F	FF
Tempo	9.927	FF
2-m-c	FF (19.3735)	FF

Tabela A.11: Teste RLAGPMCR$_\lambda$ com $n_{iter} = 50$ e m-c sob Exemplar de Controle 2.

Solução Ótima	Tempo	0.424
	Solução	20.976

continuação

PAR.	R1	R1_2B		Tempo	F	9.714
n_{iter}	50	50		6-m-ef	F	22.2871
ϵ_0	2	2		Tempo	F	7.215
$cc_\epsilon = 10$	10	10		5-m-ef	F	21.7683
Tempo	F	9.575		Tempo	5.727	6.902
9-m-ef	F	21.6557		4-m-ef	21.263	21.9921
Tempo	F	18.691		Tempo	5.983	6.37
8-m-ef	F	22.3298		3-m-ef	21.6184	22.0637
Tempo	F	8.94		Tempo	5.534	5.467
7-m-ef	F	21.7427		2-m-ef	25.106	21.3903

Tabela A.12: Teste RLAGPMCR$_\lambda$ com $n_{iter} = 50$ e m-ef sob Exemplar de Controle 2.

Do comentário anterior surge outro planteamento, misturar as m melhores classes com t melhores eficiências poderia melhorar possivelmente os tempos e manter controle na qualidade dos limitantes. Para avaliar o anterior, usa-se novamente o Exemplar de Controle 2 ($q = 5$ e $|N_k| = 10$) e faça o estudo cruzado entre melhores eficiências e classes assim: escolha-se as 4 melhores classes do exemplo e em seguida de forma decrescente vão-se trabalhando com as $9, 8, ..., 2, 1$ melhores eficiências. De forma análoga faz-se para as 3 e 2 melhores eficiências. Os resultados são apresentados na Tabela A.13.

Na Tabela A.13 pode-se observar que em todos os momentos do teste, a restrição relaxada R2B apresentaram resultados, ou seja, em tempos ressoáveis se conseguiu resolver o problema Lagrangeano com os dados fornecidos, mas, por exemplo com a relaxação Lagrangeana de R1, apresentou constantes dificuldades de execução (lembrando que F representa o teste que falhou por causa de sobrepassar o tempo mínimo aceitável para terminar o processo iterativo). Com as restrições R1_R2B, veja-se que sempre apresentou resultados em todos os testes, a pesar de não fornecer para certos parâmetros limitantes superiores par ao problema. Mas, observe-se que, para a heurística com as 4 melhores classes, tem-se desde as 9 melhores eficiências até as 3 melhores eficiências resultados que são limitantes superiores com os melhores tempos de execução para sua geração.

Até aqui, escolhe-se pelo comportamento no estudo computacional, feitos neste apêndice, as restrições em conjunto R1 e R2B para serem relaxadas Lagrangeanamente do PMCR e com seu Problema Dual Lagrangeano associado, quem seja o fornecedor dos limitantes superiores para o critério de poda na árvore de enumeração apresentado no Capítulo 5. Note que esta conclusão está baseada nos parâmetros das informações dos exemplares de controle, dos parâmetros de simulações e no uso do modelo linear do PMCR de [20], pela qual, pode-se definir outros parâmetros e ter possivelmente um direcionamento diferente na seleção da melhor

restrição a serem relaxada.

Outro aspeto a destacar é que todos os testes desenvolvidos neste apêndice pode-se ver como um sondagem para a raiz da árvore do Exemplar de Controle 1 e 2, do fato que involucra a todos os itens do exemplo. O anterior apresenta uma dificuldade, onde os tempos de execução, ainda involucrando a heurística desenvolvida, não são o suficientemente rápidos como para serem competitivos com os tempos de execução de referencia obtidos no modelo linear do PMCR de [20]. Neste sentido, para o método *Branch and Bound* desenvolvido para o PMCR no Capítulo 5, usando a relaxação Lagrangeana para o problema, vão-se obter bons limitantes superiores mas com um alto custo computacional, tendo efeito no tempo de execução. Como proposta para futuros estudos relacionados com limitantes superiores voltados à Relaxação Lagrangeana, é como diminuir o custo computacional para a obtenção dos limitantes superiores, procurando ter tempos competitivos (veja-se o Capítulo 7).

A.1.1 Limitantes Superiores Via Relaxação Lagrangeana para o PMCR com o Modelo Linear Forte

Em um processo similar da seção anterior foi estudado o comportamento computacional das relaxações Lagrangeanas para o Modelo Linear Forte exposto no Capítulo 4, fazendo relaxação ás restrições análogas R1, R1-2B, R2, R2A, R2B e R3. Daqui só vão-se apresentar a relaxação das restrições R1-2B, já que as outras relaxações são similares ás feitas na seção anterior.

Então, defina-se a relaxação Lagrangeana nas restrições R1 e R2B no Modelo Linear Forte como:

(RLAGFPMCR$_\lambda$_R1-R2B) Maximizar:

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_{k^*}} u_i^k a_{ij}^k + \lambda\left(L - \sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_{k^*}} l_i^k a_{ij}^k\right) + \sum_{k=1}^{q}\sum_{j=1}^{p_{k^*}} \eta_j^k\left(\delta_j^k L_{max}^{k^*} - \sum_{i\in N_k} l_i^k a_{ij}^k\right) =$$

$$\sum_{k=1}^{q}\sum_{i\in N_k}\sum_{j=1}^{p_{k^*}} \left(u_i^k - (\lambda+\eta_j^k) l_i^k\right) a_{ij}^k + \sum_{k=1}^{q}\sum_{j=1}^{p_{k^*}} \delta_j^k \eta_j^k L_{max}^{k^*} + \lambda L \tag{A.51}$$

sujeito a: (A.52)

$$\delta_j^k L_{min}^{k^*} \leq \sum_{i\in N_k} l_i^{k^*} a_{ij}^k \text{ com } j = 1,...,p_k, k = 1,...,q \tag{A.53}$$

$$\delta_j^k \geq \delta_{j+1}^k, \text{ para todo } j = 1,...,p_{k^*}, k = 1,...,q \tag{A.54}$$

$$\sum_{j=1}^{p_{k^*}} a_{ij}^k \leq d_i^k \delta_{i1}^k,\ i \in N_k,\ k = 1,..q \tag{A.55}$$

$$\sum_{k=1}^{q}\sum_{j=1}^{p_{k^*}} \delta_j^k \leq F_1^* \tag{A.56}$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2^k \delta_j^k,\ j = 1, ..., p_k, k = 1, ..., q \tag{A.57}$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ e inteiros, } i \in N_k,\ j = 1, ..., p_{k^*},\ k = 1, ..., q \tag{A.58}$$

Seja $\theta \in \mathbb{R}^{m+1}$ e $m = \sum_{k=1}^{q} p_k$ é o vetor de multiplicadores Lagrangeanos, com $\theta = (\lambda\ \eta) \geq 0$, $\lambda \in \mathbb{R}$ e $\eta \in \mathbb{R}^m$. Define-se o Problema Dual Lagrangeano, RLAGFPMCR para R1-R2B, como:

$$(\text{RLAGFPMCR\_R1-R2B}) :\ \min_{\theta \geq 0} v(\text{RLAGFPMCR}_\theta\text{\_R1-R2B}) \tag{A.59}$$

Define-se o vetor subgradiente e o tamanho do passo para o RLAGPMCR_R1-R2B para que serem aplicados no Método do Subgradiente para Funções Convexas, como segue a continuação:

Define-se o sub-gradiente G como $G = \left(\overset{\lambda}{G}\ \overset{\eta}{G}\right)$ onde:

$$\overset{\lambda}{G} = L - \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_{k^*}} l_i^k a_{ij}^k \tag{A.60}$$

$$\overset{\eta}{G}{}_j^k = DEL_j^k L_{max}^{k^*} - \sum_{i \in N_k} l_i^k X_{ij}^k \tag{A.61}$$

Os valores que compõem o vetor subgradiente X_{ij}^k e DEL_j^k são as soluções das variáveis respectivamente a_{ij}^k e δ_j^k para $k = 1, ..., q$, $i \in N_k$ e $j = 1, .., p_k$, correspondente do Problema Lagrangeano Dual (A.59). O tamanho do passo $t_{\bar{x}}$ vão estar definido como:

$$t_{\bar{k}} = \frac{\epsilon_{\bar{k}} \cdot (Z_{lsup} - Z_{linf})}{\sum_{k=1}^{q} \sum_{j=1}^{p_k} \left(\left(\overset{\lambda}{G} \right)^2 + \left(\overset{\eta}{G}{}_j^k \right)^2 \right)} \tag{A.62}$$

Com as relaxações Lagrangeanas estabelecidas para o Modelo Linear Forte, estas foram sometidas a simulações com os Exemplares de Controle 1 e 2 e usando os mesmos critérios de teste que foram apresentados na seção anterior em referencia ao Modelo Linear, com a intenção de obter instancias de comparação.

Solução ótima do exemplo de controle:					Tempo	0.424	Solução	20.976

PAR.	R1	R1_2B	-	R1	R1_2B	-	R1	R1_2B
n_{iter}	50	50	n_{iter}	50	50	n_{iter}	50	50
ϵ_0	2	2	ϵ_0	2	2	ϵ_0	2	2
$cc_\epsilon = 10$	10	10	$cc_\epsilon = 10$	10	10	$cc_\epsilon = 10$	10	10
4 melhores classes			3 melhores classes			2 melhores classes		
Tempo	F	10.323	Tempo	F	8.477	Tempo	F	FF
9-m-ef	F	21.3933	9-m-ef	F	FF (20.3679)	9-m-ef	F	FF
Tempo	F	10.694	Tempo	F	FF	Tempo	FF	FF
8-m-ef	F	21.6295	8-m-ef	F	FF	8-m-ef	FF	FF
Tempo	F	11.881	Tempo	F	FF	Tempo	FF	FF
7-m-ef	F	21.3099	7-m-ef	F	FF	7-m-ef	FF	FF
Tempo	F	9.204	Tempo	5.534	FF	Tempo	FF	FF
6-m-ef	F	21.3089	6-m-ef	FF (20.0227)	FF	6-m-ef	FF	FF
Tempo	5.182	7.996	Tempo	FF	FF	Tempo	FF	FF
5-m-ef	21.0442	21.352	5-m-ef	FF	FF	5-m-ef	FF	FF
Tempo	5.87	8.281	Tempo	FF	FF	Tempo	FF	FF
4-m-ef	21.4768	21.3144	4-m-ef	FF	FF	4-m-ef	FF	FF
Tempo	5.128	6.618	Tempo	FF	FF	Tempo	FF	FF
3-m-ef	20.9158	21.1854	3-m-ef	FF	FF	3-m-ef	FF	FF
Tempo	FF	4.437	Tempo	FF	FF	Tempo	FF	FF
2-m-ef	FF	FF (19.1827)	2-m-ef	FF	FF	2-m-ef	FF	FF

Tabela A.13: Teste RLAGPMCR$_\lambda$ com $n_{iter} = 50$ e m-c(4,3,2)-ef sob Exemplar de Controle 2.

Inicie-se o estudo computacional usando-se o Exemplar de Controle 1 para avaliar a Relaxação Lagrangeana do Modelo Linear Forte PMCR na restrição R2 com Problema Dual Lagrangeano, foi resolvido usando o Método do Subgradiente desenvolvido no Algoritmo 27 com parâmetros de implementação $\epsilon_0 = 1,2$, com máximo 100 iterações ($n_{iter} = 100$) e controle de mudança de ϵ_0 a 10 iterações ($cc_\epsilon = 10$); As tabelas A.15 e A.14apresenta os resultados da implementação.

Parâmetros									
$\epsilon_0 = 2$				$n_{iter} = 100$			$cc_\epsilon = 10$		
Inicio iterac.	cont.	cont.	cont.	cont.	cont.	cont.	cont.	cont.	cont.
6.866	8.41942	7.27004	7.01331	6.91744	6.89011	6.85898	6.85287	6.85358	6.84996
7.29891	7.76646	6.93629	6.89966	6.86859	6.86111	6.86755	6.85842	6.8522	6.8523
7.55952	7.53057	7.14726	6.9543	6.90862	6.87289	6.85905	6.85291	6.85027	6.84997
7.70542	7.59563	7.01257	6.92601	6.8705	6.86519	6.86723	6.8589	6.85249	6.85226
7.85307	7.57559	7.13284	6.94547	6.89778	6.87172	6.85911	6.85458	6.84995	6.84997
8.07653	7.27722	7.00336	6.91308	6.87022	6.86559	6.86693	6.85605	6.85245	6.85043
8.10553	7.40923	7.08353	6.93152	6.89238	6.86146	6.85918	6.85356	6.84996	6.84835
8.26612	7.32777	6.99549	6.91198	6.87039	6.8682	6.86663	6.85596	6.8524	6.85027
8.23198	7.38785	7.05856	6.93359	6.89122	6.8589	6.85924	6.85357	6.84996	6.84854
8.38236	7.25136	6.99235	6.91103	6.87073	6.86787	6.85902	6.85586	6.85235	6.85037
Z_{lsup}							6.84835		
Tempo de execução							11.798		

Tabela A.14: Teste com $n_{iter} = 100$ para RLAGFPMCR$_\theta$_R2 com o Exemplar de Controle 1.

Parâmetros									
$\epsilon_0 = 1$				$n_{iter} = 100$			$cc_\epsilon = 10$		
Inic. iterac.	cont.	cont.	cont.	cont.	cont.	cont.	cont.	cont.	cont.
6.866	7.20012	7.02432	6.9151	6.89922	6.87584	6.85979	6.86044	6.85958	6.86024
7.08245	6.96261	6.90332	6.87222	6.87006	6.86847	6.86011	6.86006	6.8599	6.85986
7.14079	7.10894	6.95338	6.90496	6.87624	6.86785	6.86052	6.85966	6.86032	6.85946
7.15021	6.97431	6.90972	6.87308	6.87907	6.86831	6.86014	6.85998	6.85994	6.85978
7.16942	7.12834	6.93741	6.90341	6.86394	6.86873	6.85974	6.8604	6.85954	6.8602
7.21078	6.98743	6.92055	6.87359	6.88119	6.86819	6.86006	6.86002	6.85986	6.85982
7.18924	7.08838	6.92459	6.90194	6.86187	6.86722	6.86048	6.85962	6.86028	6.85942
7.21386	6.99888	6.9208	6.87406	6.87678	6.86058	6.8601	6.85994	6.8599	6.85974
7.196	7.05148	6.9258	6.90054	6.86988	6.86117	6.8597	6.86036	6.8595	6.86016
7.21247	7.00049	6.91569	6.8745	6.86587	6.86025	6.86002	6.85998	6.85982	6.85978
Z_{lsup}							6.85942		
Tempo de execução							8.092		

Tabela A.15: Teste com $n_{iter} = 100$ com $\epsilon_0 = 1$ para RLAGFPMCR$_\theta$_R2 com o Exemplar de Controle 1.

Com os Problemas Duais Lagrangeanos definidos para o Modelo Linear Forte, foi feita um teste de controle com o Exemplar de Controle 1 para avaliar o comportamento dos limitantes superiores que está gerando com o uso do Algoritmo do Método do Subgradiente 31. Os parâmetros usados para o teste foi:$n_{iter} = 50, 100, 150, 200$, $\epsilon_0 = 2$ e controle de de cambio de ϵ a 10 . Pode-se observar os resultados na Tabela A.16.

	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
ϵ_0	-	2	2	2	2	2	2
cc_ϵ	-	10	10	10	10	10	10
n_{iter}	50						
Tempo	0.265	4.514	3.632	4.55	10.805	16.174	5.985
Resultado	6.284	6.53281	6.75288	6.866	6.84179	6.40719	6.77113
n_{iter}	100						
Tempo	0.265	14.564	8.739	11.798	24.47	40.002	16.329
Resultado	6.284	6.52754	6.61924	6.84835	6.82163	6.40002	6.77113
n_{iter}	150						
Tempo	0.265	20.458	15.24	12.999	40.249	70.56	31.787
Resultado	6.284	6.52754	6.56632	6.84835	6.81894	6.39951	6.77113
n_{iter}	200						
Tempo	0.265	18.278	23.835	13.292	39.414	83.282	51.641
Resultado	6.284	6.52754	6.56472	6.84835	6.81894	6.39951	6.77113

Tabela A.16: Teste para $n_{iter} = 50, 100, 150, 200$ para RLAGFPMCR$_\lambda$ (Modelo Linear Forte) com o Exemplar de Controle 1.

Em seguida foi feita testes com o Exemplar de Controle 2, com parâmetros: $n_{iter} = 25, 50, 100$, $\epsilon_0 = 2$ e controle de de cambio de ϵ a 10 . Pode-se observar os resultados na Tabela A.16.

Também foi testada a *Heurística das m melhores eficiências e das t melhores classes* com as relaxações Lagrangeanas para o Modelo Linear Forte organizada da seguente forma: com o Exemplar de Controle 1 foi testado usando a Heurística das m melhores eficiências ("m-ef") com parâmetros $n_{iter} = 50$, $\epsilon_0 = 2$, $cc_e = 10$, $ef = 4, 3, 2$ e resultados apresentados na Tabela A.18. Com o Exemplar de Controle 2 foi testado a Heurística das m melhores eficiências ("m-ef") om parâmetros $n_{iter} = 50$, $\epsilon_0 = 2$, $cc_e = 10$, $ef = 4, 3, 2$ e resultados na Tabela A.20; e para Heurística das t melhores classes ("m-c") usando como parâmetros $n_{iter} = 50$, $\epsilon_0 = 2$, $cc_e = 10$ e $c = 10, 9, ..., 2$. Os resultados são apresentados na Tabela A.19.

.

	Solução Ótima	R1	R1_2B	R2	R2A	R2B	R3
ϵ_0	-	2	2	2	2	2	2
cc_ϵ	-	10	10	10	10	10	10
n_{iter}	25						
Tempo	0.422	F	4.636	18.243	12.587	11.541	7.345
Resultado	20.973	F	21.2242	21.731	21.6867	20.973	21.6507
n_{iter}	50						
Tempo	0.422	F	8.8	73.544	29.921	O. A.	19.067
Resultado	20.973	F	21.2168	21.4956	21.5032	O. A.	21.504
n_{iter}	100						
Tempo	0.422	F	20.343	F	F	O. A.	58.628
Resultado	20.973	F	21.2156	F	F	O. A.	21.387

Tabela A.17: Teste para $n_{iter} = 25, 50, 100$ para RLAGFPMCR$_\lambda$ (Modelo Linear Forte) com o Exemplar de Controle 2.

Solução Ótima	PAR.	R1	R1_2B	R2	R2A	R2B	R3
	n_{iter}	50	50	50	50	50	50
Tempo	ϵ_0	2	2	2	2	2	2
0.265	cc_ϵ	10	10	10	10	10	10
	Tempo	5.423	3.784	7.912	8.43	23.444	6.208
	4-m-ef	6.87711	6.66301	6.866	6.84614	6.39108	6.77113
Solução	Tempo	5.171	3.509	3.747	7.222	6.57	6.176
6.284	3-m-ef	6.80039	6.63003	6.866	6.83747	FF	6.77113
	Tempo	4.303	2.583	3.689	6.004	FF	6.334
	2-m-ef	6.31997	6.70023	6.866	6.82755	FF	6.77113

Tabela A.18: Teste RLAGFPMCR$_\lambda$ (Modelo Linear Forte) com $n_{iter} = 50$ e m-ef sob Exemplar de Controle 1.

Solução Ótima	Tempo	0.424
	Solução	20.976

PAR.	R1	R1_2B
n_{iter}	50	50
ϵ_0	2	2
$cc_\epsilon = 10$	10	10
Tempo	F	8.163
4-m-c	F	21.0288
Tempo	F	FF
3-m-c	F	FF
Tempo	F	FF
2-m-c	F	FF

Tabela A.19: Teste RLAGFPMCR$_\lambda$ (Modelo Linear Forte) com $n_{iter} = 50$ e m-c sob Exemplar de Controle 2.

Solução Ótima	Tempo	0.424
	Solução	20.976

continuação

PAR.	R1	R1_2B
n_{iter}	50	50
ϵ_0	2	2
$cc_\epsilon = 10$	10	10
Tempo	F	8.113
9-m-ef	F	21.2206
Tempo	F	8.345
8-m-ef	F	21.2188
Tempo	F	7.136
7-m-ef	F	21.2161
Tempo	F	8.662
6-m-ef	F	21.2182
Tempo	F	6.366
5-m-ef	F	21.2379
Tempo	9.805	7.634
4-m-ef	21.263	21.2174
Tempo	9.714	6.815
3-m-ef	21.6184	21.4047
Tempo	9.231	5.153
2-m-ef	25.106	21.219

Tabela A.20: Teste RLAGFPMCR$_\lambda$ com $n_{iter} = 50$ e m-ef sob Exemplar de Controle 2.

Finalmente foi usada Heurística das m melhores eficiências e das t melhores classes para as relaxações Lagrangeanas definida para o Modelo Linear Forte para $ef = 4, 3, 2$ e $c = 9, ..., 2$ com resultados na Tabela A.21.

Solução ótima do exemplo de controle:				Tempo	0.424	Solução	20.976

PAR.	R1	R1_2B	-	R1	R1_2B	-	R1	R1_2B
n_{iter}	50	50	n_{iter}	50	50	n_{iter}	50	50
ϵ_0	2	2	ϵ_0	2	2	ϵ_0	2	2
$cc_\epsilon = 10$	10	10	$cc_\epsilon = 10$	10	10	$cc_\epsilon = 10$	10	10
4 melhores classes			3 melhores classes			2 melhores classes		
Tempo	F	7.774	Tempo	F	7.726	Tempo	F	FF
9-m-ef	F	21.0281	9-m-ef	F	FF(20.0274)	9-m-ef	F	FF
Tempo	F	7.953	Tempo	F	FF	Tempo	FF	FF
8-m-ef	F	21.0278	8-m-ef	F	FF	8-m-ef	FF	FF
Tempo	F	8.42	Tempo	F	FF	Tempo	FF	FF
7-m-ef	F	21.0279	7-m-ef	F	FF	7-m-ef	FF	FF
Tempo	F	7.951	Tempo	7.98	FF	Tempo	FF	FF
6-m-ef	F	21.0291	6-m-ef	FF(20.0227)	FF	6-m-ef	FF	FF
Tempo	9.726	7.38	Tempo	FF	FF	Tempo	FF	FF
5-m-ef	21.0442	21.0281	5-m-ef	FF	FF	5-m-ef	FF	FF
Tempo	8.436	7.329	Tempo	FF	FF	Tempo	FF	FF
4-m-ef	21.4768	21.0286	4-m-ef	FF	FF	4-m-ef	FF	FF
Tempo	8.351	6.643	Tempo	FF	FF	Tempo	FF	FF
3-m-ef	FF(20.9158)	FF(20.6275)	3-m-ef	FF	FF	3-m-ef	FF	FF
Tempo	FF	FF	Tempo	FF	FF	Tempo	FF	FF
2-m-ef	FF	FF	2-m-ef	FF	FF	2-m-ef	FF	FF

Tabela A.21: Teste RLAGFPMCR$_\lambda$ (Modelo Linear Forte) com $n_{iter} = 50$ e m-c(4,3,2)-ef sob Exemplar de Controle 2.

ANEXO

Neste anexo será exibida o artigo *A Strong Integer Linear Optimization Model to the Compartmentalized Knapsack Problem*, submetido à revista *International Transactions in Operational Research* (ITOR).

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. XX (2018) 1–22
DOI: xx.xxxx/itor.xxxxx

A Strong Integer Linear Optimization Model to the Compartmentalized Knapsack Problem

John J. Quiroga-Orozco [a], J.M. Valério de Carvalho[b] and Robinson S. V. Hoto[a]

[a]*Departamento de Matemática, Universidade Estadual de Londrina, Rodovia Celso Garcia Cid, Km 380, Londrina, Brazil*
[b]*Departamento de Produção e Sistemas, Universidade do Minho, Campus Gualtar, Braga 4710-057, Portugal*
E-mail: jjqojj@outlook.com [Quiroga];vc@dps.uminho.pt [Valerio];hoto@uel.br [Hoto]

Received DD MMMM YYYY; received in revised form DD MMMM YYYY; accepted DD MMMM YYYY

Abstract

The Compartmentalized Knapsack Problem (CKP) is a relatively new type of problem with a wide application in industrial processes, arising, for instance, in the case of cutting steel coils in two phases in the metallurgical industry.
In the literature, there are two mathematical formulations for the CKP: a classical formulation, which is a nonlinear integer programming model, and a recent (linear) integer programming formulation, obtained by discretizing the compartments that can be built for each class of items. It is an important contribution, because it makes the problem amenable to solution by mixed-integer linear programming tools. Combinatorial enumeration algorithms and several pseudo-polynomial decomposition heuristics were also developed for the CKP.
This paper presents a new model for the exact solution of the CKP, denoted as the Strong Integer Linear model, derived from the (linear) integer programming formulation by strengthening data, reducing symmetry and lifting, and a new pseudo-polynomial heuristic, the Heuristic of the p_k Strong Capacities. Computational experiments are presented with a large set of instances that show the advantage of the new approaches. The strong model solves the CKP exactly more than seven times faster, and the new heuristic is more efficient, presenting a good balance in the terms of effectiveness.

Keywords: Compartmentalized Knapsack Problem; linear strong model optimization; discrete optimization; Linear programming

1. Introduction

Consider a knapsack with capacity L and set of items of n types, indexed by i in a set N ($i = 1, \ldots, n$, and $i \in N$). Each item of type $i \in N$ has a weight (p_i) and a utility (u_i). The classic Knapsack Problem consists of determining which items and how many (depending on the demand of each item) should

*Author to whom all correspondence should be addressed (e-mail: jjqojj@outlook.com).

fill the knapsack, in order to obtain the maximum utility, respecting the capacity of the knapsack. Now define a variation of this classic problem: classify the items under some criterion and divide them into q disjoint classes, *i.e.*, make a partition of the set N in $N = \bigcup_{k=1}^{q} N_k$ with $\bigcap_{k=1}^{q} N_k = \varnothing$, and consider that filling the knapsack with items $i, j \in N$ is subject to a condition: items of different classes can not be mixed in the knapsack, *i.e.*, items $i \in N_s$ and $j \in N_g$ can not be inserted "together" into the knapsack unless $s = g$, with $s, g \in C = \{1, 2, \ldots, q\}$, where C is the set of classes, or they are inserted into different compartments. In fact, in order to be able to insert items from different classes into the knapsack, we need *to construct*, inside the knapsack, different **compartments,** each grouping items of the same class.

These compartments are not predetermined, they are defined by the items to insert into the knapsack. Now, we can present an informal definition of the Problem of the Compartmentalized Knapsack in its unrestricted version. *The Compartmentalized Knapsack Problem* (CKP) consists of finding the capacities of the compartments created for each class $k = 1, \ldots, q$ (which are determined by the items inside them) aiming at obtaining the *maximum* possible utility.

This problem arises in the industrial metallurgical practice when planning cutting patterns for the two-stage process of cutting steel coils. Here, the items correspond to the ribbon coils that are grouped by their thickness, thus defining the classes of the items. These ribbons are obtained (after two stages) by cutting the steel coils available in stock, which correspond to the knapsack to be filled. The steel coils are slit in a first cutting process, producing sub-coils with the same thickness as the original coil. Then, in a process called "cold rolling", each sub-coil has its thickness reduced, in order to match the desired thickness for the type of items that will be produced later. These sub-coils can be seen as the compartments of the knapsack. Finally, the sub-coils are submitted to a second cutting process to obtain the steel ribbons with the final size. A representation of the process is shown in Fig. 1.

The CKP study is relatively new. In the following, some theoretical advances and the current state of the art are presented. CKP was treated implicitly in its restricted case in works such as Haessler (1979), Ferreira et al. (1990), Pereira (1993), Valério de Carvalho and Rodrigues (1994) and Valério de Carvalho and Rodrigues (1995). The first mathematical elements in the approach and the definition of the CKP are found in Hoto (1996), who later formulated an integer nonlinear optimization model for the unrestricted version of the problem, in his PhD thesis, Hoto (2001), which, in addition to presenting the CKP model, makes a detailed study of the application of the CKP to the Steel Coil Cutting Problem (SCCP) with solution approaches such as the exact compartmentalization algorithm called COMPEX and a compartmentalization heuristic named COMPMT, in which bounds from Martello and Toth (1990) are used. It also defines the $0 - 1$ Compartmentalized Knapsack Problem and describes a branch-and-bound solution method.

Marques and Arenales (2002) studies in-depth the Compartmentalized Knapsack Problem in the Restricted case (CKPR), presenting a model and heuristics, such as the Decomposition Heuristic, the Best Compartment Heuristics and the "z" Best Compartments Heuristic. In Marques and Arenales (2007) the Heuristic of "W" capacities shows better computational results when compared to the heuristics presented in Marques and Arenales (2002). In addition, in the same work, a linear relaxation of the CKP was derived, providing upper bounds for the problem.

A first approach to the CKP as a linear problem was exposed in Hoto et al. (2006), with a new formulation of the objective function and its constraints, along with the presentation of retro feed heuristics "$retro$", the best compartment for W capacities with feedback "$WRetro$" and best-case heuristics for

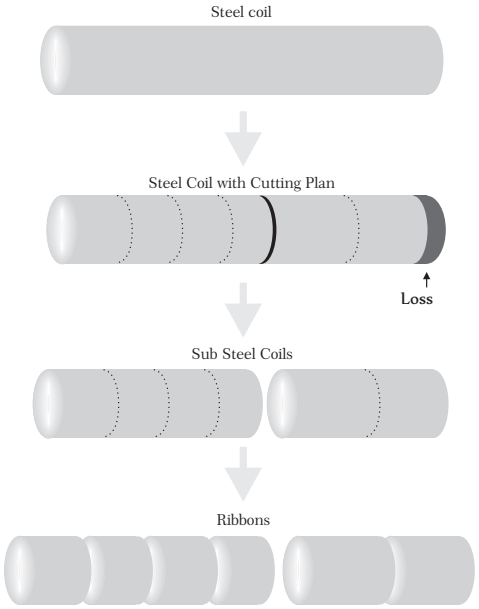

Fig. 1. Cutting in two phases: steel coils (knapsack), sub-coils (compartments) and ribbons (items).

W capacities W times "WW". In Cruz (2010) the linear approaches to the CKPR are further explored with a new modification of the objective function and the constraints presented in Hoto et al. (2006), using the heuristics discussed above. In Inarejos (2015) new linearity studies of the CKP continued the work of Cruz (2010), together with a new algorithm for the exact solution of the problem, called the Exhaustive Decomposition Algorithm for the CKPR.

In Leão et al. (2011), a linear treatment for the CKPR is presented next to a heuristic to solve it, called *Hybrid Heuristic*, which merges the "z" Best Compartments and the "W" Capacities. The paper also presents a method for solving the constrained CKPR master problem using column generation, called the *Dynamic Column Generation Method*. Leão A. (2013) treats the CKP as a two-dimensional problem, formulating several mathematical models for the two-dimensional problem, and considering three-stage guillotine cutting processes.

Inarejos et al. (2017) presents a Linear Model for the CKPR (LMCKPR) and proves its linearity, in addition to exposing a new heuristic for the CKPR, using the classic model of the problem, called *The Decreasing Heuristic Classes* and claiming to be more competitive in relation to the heuristic of the W capacities of Marques and Arenales (2007).

The linearity of LMCKPR is an important contribution of Inarejos et al. (2017), because it makes the model amenable to solution by mixed-integer linear programming tools.

Our work stems from the work of Inarejos et al. (2017), and, in the following, we provide further technical information about the CKPR and describe the LMCKPR model.

The steel coil cutting process in two stages described above introduces technical constraints: the number of sub-coils in the main coil is limited by an integer number F_1, related to the maximum number of knives the first stage cutting machine can support. The feeder of the cold rolling machine also introduces

constraints. Here, limitations are defined in the capacities (in this case widths) of the sub-coils that can be created in each class $k = 1, \ldots, q$ as capacities must be between a minimum value L^k_{min} and a maximum value L^k_{max}. Furthermore, for the second stage cutting process, the number of ribbons obtainable from the sub-rolls is limited by the number of knives available in the machine, being represented as F_2. Another additional issue to be taken into account is the demand for steel ribbons.

For modeling the CKPR, in order to obtain a linear model, Inarejos et al. (2017) discretize the compartments and introduce a limit to the number of compartments that can be created for each class $k = 1, \ldots, q$, calculated as $p_k = \min\left\{F_1, \left\lfloor L/L^k_{min} \right\rfloor\right\}$. To facilitate the handling of all problem information, local indexing of the items of each class from 1 to $|N_k|$ and the compartments for each class from 1 to p_k is used. With the above, define the decision variable a^k_{ij}, which will match the number of items i in the j compartment of the k class. The linear model for the CKPR (see Inarejos et al. (2017)) is the following:

$$\text{maximize: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} u^k_i a^k_{ij} \tag{1}$$

$$\text{subject to: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} l^k_i a^k_{ij} \leq L \tag{2}$$

$$\delta^k_j L^k_{min} \leq \sum_{i \in N_k} l^k_i a^k_{ij} \leq \delta^k_j L^k_{max}, \forall j = 1, \ldots, p_k, k = 1, \ldots, q \tag{3}$$

$$\sum_{j=1}^{p_k} a^k_{ij} \leq d^k_i, i \in N_k, k = 1, \ldots, q \tag{4}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta^k_j \leq F_1 \tag{5}$$

$$\sum_{i \in N_k} a^k_{ij} \leq F_2, j = 1, \ldots, p_k, k = 1, \ldots, q \tag{6}$$

$$\delta^k_j \in \{0, 1\}, a^k_{ij} \geq 0 \text{ and integer}, i \in N_k, j = 1, \ldots, p_k, k = 1, \ldots, q \tag{7}$$

The objective function (1) represents the sum of the utilities of all items chosen in each compartment. Note that the variable a^k_{ij} will be defined only if the item i and the compartment j refer to the same class N_k. The constraint (2) ensures that the sum of the widths of the chosen items is limited by the capacity of the knapsack. The constraints (3) limit the capacity of the nonzero compartments and overrides the compartment coefficients that will not be used with the binary control variable δ^k_j ($\delta^k_j = 1$, if the j compartment of the k class is used, and 0 otherwise). The constraints (4) limit the quantity of each item $i \in N$ by its respective demand. The constraint (5) limits the number of compartments inserted in the original steel coil and constraints (6) the number of items in each compartment. The constraints (7) establish the domain of the variables.

In this paper, new approaches to the CKPR that explore the linearity of the model of Inarejos et al. (2017) are presented. First, the model is strengthened, and its symmetry is reduced, leading to a new

linear model called the *Strong Linear Model to the Compartmentalized Knapsack Problem*. Second, the paper presents a new pseudo-polynomial decomposition heuristic called *The Heuristics of p_k Strong Capacities* based on the Heuristic of W Capacities of Marques and Arenales (2007), where the number of compartments W for each class is set by the number of compartments that can be constructed, in addition to taking advantage of the strengthening of the CKPR model. For further details, see Quiroga-Orozco (2018).

This paper is organized as follows. In Section 2, a study is made concerning building the limiting capacity, the number of knives for the first and second cutting process, the increased width for some items, the reduction of symmetry in feasible solutions in the Linear Model of Inarejos et al. (2017) and the presentation of the new linear model. In Section 3, a general and exposed frame the new heuristics for CKPR is made. In Section 4, computational results are presented, and the final remarks are in Section 5.

2. A Strong Linear Model of the Compartmentalized Knapsack Problem

The book by Nemhauser and Wolsey (1988) is a milestone in Integer Programming (IP), because it places the strength of the model as a central concern in IP modelling. This is new with respect to previous books in integer programming. Research has shown, and it became widely accepted, that a) the quality of the bound provided by the Linear Programming (LP) relaxation of the IP model is a crucial factor in the size of the search trees in branch-and-bound; b) strong models have a better behavior when the size of the instances grows, meaning that, if two models have a similar behavior, in terms of computational time, for a given size of instances, one can anticipate that the stronger model will perform better than the weaker model for larger instances.

Define an integer programming problem IP along its linear relaxation LP as follows:

$$\begin{aligned} \text{(IP) Maximize: } & z_{IP} = cx \\ \text{subj. to: } & Ax \leq b \\ & x \geq 0 \text{ and } x \in \mathbb{Z} \end{aligned} \tag{8}$$

$$\begin{aligned} \text{(LP) Maximize: } & z_{LP} = cx \\ \text{subj. to: } & Ax \leq b \\ & x \geq 0 \text{ and } x \in \mathbb{R} \end{aligned} \tag{9}$$

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and $x \in \mathbb{Z}_+^n$ is a vector of decision variables.

Branch-and-bound is an implicit enumeration process that is based on the LP relaxation of the IP model and uses branching constraints to obtain the optimal solution to the IP model. Generally speaking, the set of solutions of an IP model is a discrete set $X_{IP} = \{x \in \mathbb{Z}_+^n : Ax \leq b\}$. The set of solutions of the LP relaxation of the IP model is a convex set $X_{LP} = \{x \in \mathbb{R}_+^n : Ax \leq b\}$. The set X_{LP} contains all the integer solutions, as $X_{IP} = X_{LP} \cap \mathbb{Z}_+^n$. Typically there are different ways of modelling an IP problem, and any model is valid if its set of feasible integer solutions is X_{IP}. However, there may be differences in their LP relaxations. Some models may provide a closer description of the set of the feasible integer solutions.

Definition 1 (Stronger model). Consider two equivalent IP models, denoted as IP and IP', whose LP relaxations are LP and LP', respectively. Let $X \subseteq \mathbb{R}^n$ and $X' \subseteq \mathbb{R}^n$ be the convex sets of solutions of LP and LP', respectively, meaning that $X_{IP} = X_{IP'} = X \cap \mathbb{Z}_+^n = X' \cap \mathbb{Z}_+^n$. A model IP' with an

LP relaxation with a set X' is said to be stronger than a model IP with an LP relaxation with a set X, if $X' \subset X$. Alternatively one may state that the second model is weaker.

In the following sections, we use the mixed-integer linear programming model of Inarejos et al. (2017), and adjust the value of data coefficients, lift coefficients and other IP tools to obtain a stronger model, with a reduced domain, defined as follows.

Definition 2 (Reduced domain). Consider models IP and IP' with LP relaxations with sets X and X', respectively. We say that the integer programming model IP' has an LP relaxation with a reduced domain X', if $X' \subset X$. Furthermore, as we are dealing with a maximization problem, the following relations hold: $z_{IP} = z_{IP'} \leq z_{LP'} \leq z_{LP}$. Therefore the bound provided by the model LP' is of better quality.

For further readings about strong valid constraints IP models, Nemhauser and Wolsey (1988) and Martin (1999) are recommended.

The LMCKPR model can be strengthened using different strategies. First, in the following subsections, we address the adjustment of the values of data coefficients. If the function with integer variables in a knapsack constraint can not be equal to the right-hand side, then the value of the right-hand side can be reduced. The constraints to be strengthened this way are: the constraints (3) of the compartmentalization of the knapsacks, where it is intended to "tighten" the values of L^k_{min} and L^k_{max} for all $k = 1, \ldots, q$; the constraint (5) which limits the number of compartments that can be inserted into the knapsack, where it is intended to reduce F_1; the constraint (6) which limits the number of items that can be inserted into the knapsack, where it is intended to reduce F_2. In later subsections, we address lifting the coefficients of the weights (widths) associated with items in some constraints and reducing the symmetry of the model.

2.1. Strong information for the CKPR

Recall the input for the CKPR: a knapsack capacity L, a set of items N partitioned into q classes that have a utility u^k_i and a weight l^k_i with $i \in N_k$ and $k = 1, \ldots, q$. Also, for each item $i \in N_k$, with $k = 1, \ldots, q$ there is a demand d^k_i. For each class $k = 1, \ldots, q$, there is a maximum number of compartments that can be created, denoted by p_k, and maximum and minimum capacities that the compartments of each class, L^k_{max} and L^k_{min}, $k = 1, \ldots, q$, respectively. Finally, the quantities of knives available for the first and second cutting processes of the steel coils, which are respectively F_1 and F_2, have been defined.

2.1.1. Adjusting compartment capacities

For each class $k = 1, \ldots, q$, we have technical lower and upper bounds to the capacity of the compartments, which are L^k_{min} and L^k_{max}, respectively. It is desirable to tighten the values of each L^k_{min} and L^k_{max} of each class $k = 1, \ldots, q$, that is, deriving new values of $L^{k^*}_{min}$ and $L^{k^*}_{max}$ (with $L^{k^*}_{min} \leq L^k_{min}$ and $L^{k^*}_{max} \geq L^k_{max}$), such that the inequalities $L^{k^*}_{min} \leq \sum_{i \in N_k} l^k_i a^k_{ij} \leq L^{k^*}_{max}$ are still valid inequalities for the integer CKPR model. Then, the constraints with the technical lower and upper bounds to the capacity of

the compartments are replaced by the stronger inequalities:

$$L^{k^*}_{min} \leq \sum_{i \in N_k} l^k_i a^k_{ij} \leq L^{k^*}_{max}, \forall j = 1, \ldots, p_k, k = 1, \ldots, q \tag{10}$$

To compute the new $L^{k^*}_{min}$ and $L^{k^*}_{max}$, for each class $k = 1, \ldots, q$, it is necessary to evaluate the largest size of the compartment (less than or equal to the technical upper bound L^k_{max}) that can be achieved combining items in the class, keeping in mind the demand of each item, that is, the demand for each item may not be exceeded, and also the upper bound in the number of knives, that is, the quantity of items to be filled in the class may not exceed the number of items F_2. Similarly, for each class $k = 1, \ldots, q$, it is necessary to evaluate the smallest size of the compartment (greater than or equal to the technical lower bound L^k_{min}) that can be achieved, under similar conditions.

The value of $L^{k^*}_{max}$ is calculated for each class $k = 1, \ldots, q$, solving a knapsack (subset sum) problem with additional cardinality constraints, which will be denoted as a Restricted Subset Sum Problem:

$$\text{maximize: } \sum_{i \in N_k} l^k_i x^k_i \tag{11}$$

subject to:

$$\sum_{i \in N_k} l^k_i x^k_i \leq L^k_{max} \tag{12}$$

$$x^k_i \leq d^k_i, \forall i \in N_k \tag{13}$$

$$\sum_{i \in N_k} x^k_i \leq F_2, i \in N_k \tag{14}$$

$$x^k_i \geq 0 \text{ and integer}, \forall i \in N_k, j = 1, \ldots, p_k \tag{15}$$

By a similar procedure, we can obtain the $L^{k^*}_{min}$, for each class $k = 1, \ldots, q$, solving the following problem:

$$\text{minimize: } \sum_{i \in N_k} l^k_i x^k_i \tag{16}$$

subject to:

$$\sum_{i \in N_k} l^k_i x^k_i \geq L^k_{min} \tag{17}$$

$$x^k_i \leq d^k_i, \forall i \in N_k \tag{18}$$

$$\sum_{i \in N_k} x^k_i \leq F_2, i \in N_k \tag{19}$$

$$x^k_i \geq 0 \text{ and integer}, \forall i \in N_k, j = 1, \ldots, p_k \tag{20}$$

Table 1
Data for the CKPR example

Classe (k)	Itens ($i \in N_k$)	u_i^k	l_i^k	d_i^k	L_{min}^k	L_{max}^k	p_k	L	F_1	F_2
1	1	0.82	7	3	9	12	2	23	14	8
	2	0.512	10	2						
	3	0.59	9	1						
2	1	0.283	11	3	5	13	4			
	2	0.975	10	6						
	3	0.594	7	4						
	4	0.286	9	2						
3	1	0.979	8	1	10	14	2			
	2	0.672	6	4						

The problem (16)-(20) can be converted into a knapsack problem with additional constraints using the variable transformation $z_i^k = 1 - x_i^k$, obtaining a reformulation. Then, the values of $L_{min}^{k^*}$ for each class $k = 1, \ldots, q$, are calculated as follows:

$$\text{maximize: } \sum_{i \in N_k} l_i^k z_i^k - \sum_{i \in N_k} l_i^k \tag{21}$$

subject to:

$$\sum_{i \in N_k} l_i^k z_i^k \leq \sum_{i \in N_k} l_i^k - L_{min}^k \tag{22}$$

$$z_i^k \geq 1 - d_i^k, \forall i \in N_k \tag{23}$$

$$\sum_{i \in N_k} z_i^k \geq |N_k| - F_2, \forall i \in N_k \tag{24}$$

$$z_i^k \geq 0 \text{ and integer}, \forall i \in N_k, j = 1, \ldots, p_k \tag{25}$$

One important advantage with the tightening of L_{min}^k and L_{max}^k for each class $k = 1, \ldots, q$ is that the values of p_k can be updated, obtaining tighter values, thus limiting the number of compartments that can be created. Define the new tight class compartment limiter p_k as p_{k^*}, where $p_{k^*} \leq p_k$.

Example 1. Consider the data presented in Table 1 of a simple example of a compartmentalized knapsack problem used to illustrate the strengthening the information of the CKPR.

The lower bounds of the capacity of each class are $L_{min}^1 = 9$, $L_{min}^2 = 5$ and $L_{min}^3 = 10$, respectively. The upper bounds for each class are $L_{max}^1 = 12$, $L_{max}^2 = 13$ e $L_{max}^3 = 14$, respectively. The solutions of the Restricted Subset Sum Problems for each class $k = 1, 2, 3$ are used to obtain the new lower bounds $L_{min}^{1^*} = 9$, $L_{min}^{2^*} = 7$ and $L_{min}^{3^*} = 12$, respectively. For instance, the data in Table 1 shows that the smallest item in class 2 has size 7, and so the lower bound $L_{min}^2 = 5$ can be replaced by $L_{min}^{2^*} = 7$. On the other hand, the new values of the upper bounds are $L_{max}^{1^*} = 10$, $L_{max}^{2^*} = 11$ and $L_{max}^{3^*} = 14$, respectively. The new value $L_{max}^{2^*} = 11$, because there is no combination of items that can achieve the sizes 12 or 13, which is the technical upper bound for this class.

Therefore, one can tighten the number of compartments that can be created in class 2, as $p_{2^*} = \min\{14, \lfloor 23/7 \rfloor\} = 3$, and in class 3, as $p_{3^*} = \min\{14, \lfloor 23/12 \rfloor\} = 1$. The values for class 1 of p_1 remain the same. Table 2 summarizes the new stronger values of the coefficients $L^{k^*}_{min}$, $L^{k^*}_{max}$ and new strong p_k for each class $k = 1, 2, 3$ for the capacities and maximum number of compartments, respectively, for the example.

Table 2
New stronger values $L^{k^*}_{min}$, $L^{k^*}_{max}$ and p_{k^*}

Class (k)	L^k_{min}	$L^{k^*}_{min}$	L^k_{max}	$L^{k^*}_{max}$	p_k	p_{k^*}
1	9	9	12	10	2	2
2	5	7	13	11	4	3
3	10	12	14	14	2	1

2.1.2. Adjusting the values of the F_1 and F_2

Analogously the values of the technical data F_1 and F_2, related to the maximum number of rolls in the intermediate and final stages, respectively, may also be adjusted in some cases. If it is possible to reduce their values, the less-than-or-equal-to constraints in which those values appear in the right-hand-side will be tightened.

We start with the knife adjustment F_1, which limits the number of compartments that can be created in the knapsack. In order to adjust the value of the knife F_1, one has to consider the smallest capacity that the compartments of all the classes can have, that is, with $\min_{k=1,\ldots,q}(L^{k^*}_{min})$, and determine how many times it could be inserted into the knapsack with capacity L. Then, following the above idea, the new value of the knife, F^*_1 (note that $F^*_1 \leq F_1$) is calculated as follows:

$$F^*_1 = \min\left\{ \left\lfloor L / \min_{k=1,\ldots,q}(L^{k^*}_{min}) \right\rfloor, F_1 \right\} \tag{26}$$

As for the adjustment of the value of the knife F_2, we have to consider for each class $k = 1, \ldots, q$ the item with smallest width l_i with $i \in N_k$, and then evaluate how many times that item may be inserted into a compartment, *i.e.*, comparing with respect to the upper bound of the compartment $L^{k^*}_{max}$, but respecting the initial value of F_2. Following the previous reasoning, for each class $k = 1, \ldots, q$, the new value of F^k_2 is calculated as follows:

$$F^k_2 = \min\left\{ \left\lfloor L^{k^*}_{max} / \min_{i \in N_k}(l^k_{min}) \right\rfloor, F_2 \right\} \tag{27}$$

Example 2. Using the information from Table 1, it is assumed that the technical data is $F_1 = 14$ and $F_2 = 8$. Then, the new values are F^*_1 and $F^*_1 = \min\{\lfloor 23/5 \rfloor, 14\} = 4$. Afterwards, we can calculate the new value for class 1, $F^1_2 = \min\{\lfloor 10/7 \rfloor, 8\} = 1$; for the class 2, $F^2_2 = \min\{\lfloor 11/7 \rfloor, 8\} = 1$ and, for the class 3, $F^3_2 = \min\{\lfloor 14/6 \rfloor, 8\} = 2$. In Table 3, there is a summary of the values obtained.

Table 3
New strong values for the values of the knives F_1 and F_2 of the example in Table 1.

Class (k)	F_1	F_1^*	F_2	F_2^k
1				1
2	14	4	8	1
3				2

2.1.3. Lifting item widths

Lifting can be used to adjust the values of the items widths within each class with respect to the capacity of the compartment. Basically lifting increases the value of the weight (width) of an item in a knapsack constraint in a way such that no feasible integer solution of the knapsack constraint is eliminated. However, increasing the values of the coefficients of the items in the knapsack constraint reduces the domain of the LP relaxation of the model (see Nemhauser and Wolsey (1988)).

In this work *lifting* is only applied to items that can be inserted at most once, so dealing only with binary items, associated to binary variables. Nevertheless, lifting can also be applied to integer variables, looking for the possibility of lifting the widths of items $i \in N_k$ of class $k \in \{1, 2, \ldots, q\}$ combined with other items of the same class that can also be inserted more than once.

Sequential lifting is used (see Nemhauser and Wolsey (1988) for details) with items ordered by decreasing values of width. Lifting can be done in constraints (3), because it is possible to decompose these constraints into a set of independent constraints, each related to a different k value. Note that the lifting adjustment described in this section can not be applied to the values of the coefficients l_i^k in constraint (2), because these values represent the space occupied by the item in a compartment of the knapsack, which may also include compartments of other classes.

Example 3. We start with the item widths of the first class, $k = 1$. After the previous tightening operations in this section, the data for the compartmentalization constraint is the following: $L_{min}^{1^*} = 9 \leq \sum_{i \in N_k} l_i^1 a_{ij}^1 \leq L_{max}^{1^*} = 10$.

Let us observe the knapsack inequality on the right and explicitly rewrite the information contained therein (as we will analyze only one compartmentalization, a_{ij}^1 is replaced by x_i^1): $7x_1^1+10x_2^1+9x_3^1 \leq 10$.

Now, we will analyze if the compartments for class 1 are constructed, what items can be inserted. For the 1 item in the 1 class, see that it could only be inserted once, in fact, that the combination of the 1 item with any of the other items in the 1 class, will not comply with the compartmentalization constraint. From the above, the width of the 1 compliance item 7 could be adjusted by the maximum capacity value of the compartment of that class 1 per $L_{max}^{1^*} = 10$, one can adjust l_1^1 per $l_1^{1^*} = 10$. This 1 item width adjustment will fulfill the same function as the old value that one can enter once. Furthermore, with the adjustment, we also have an adjustment of the inequality to the left of the maximum capacity of the compartments of the class 1 as: $10x_1^1 + 10x_2^1 + 9x_3^1 \leq 10$.

The 2 item of the 1 class, as the value of its width is the same as $L_{max}^{1^*}$, has no adjustments to be made. Finally, we will analyze if we can adjust the value of the width associated with the item 3 of class 1. Again, we have that the 3 item can be entered once only, since a combination with any of the other items of the same class will not meet the maximum capacity that the compartment can have. Then, set the value

of $l_3^{1^*} = L_{max}^{1^*} = 10$. Finally, we obtain as new inequality the following: $10x_1^1 + 10x_2^1 + 10x_3^1 \leq 10$.

Note that no feasible binary solution of the knapsack constraint is eliminated, but the domain defined by the LP relaxation of the constraint is tightened.

Thereafter, in an analogous way, the process is repeated for the items of the other classes, under the same reasoning made, looking for the possibility to adjust the associated widths. The following table summarizes the adjustments made:

Table 4
Adjusting the widths of the items in the sample of the values to be strengthened from Table 1.

Class (k)	Itens ($i \in N_k$)	l_i^k	$l_i^{k^*}$
1	1	7	10
	2	10	10
	3	9	10
2	1	11	11
	2	10	11
	3	7	11
	4	9	11
3	1	8	8
	2	6	6

In each step of the sequential lifting, the following restricted knapsack problem for the width l_i^k with $i \in N_k$ and $k \in \{1, 2, \ldots, q\}$, is solved:

$$\text{Maximize: } \sum_{j \in N_k - \{i\}} l_j^k x_j^k \tag{28}$$

sujeito a:

$$\sum_{j \in N_k - \{i\}} l_j^k x_j^k \leq L_{max}^{k^*} - l_i^k \tag{29}$$

$$\sum_{j \in N_k - \{i\}} x_j^k \leq d_j^k \tag{30}$$

$$x_i^k \geq 0 \text{ and integer, } i \in N_k,\ j = 1, \ldots, p_k,\ k = 1, \ldots, q \tag{31}$$

To finalize the section, in the Table 5 one can see in brief the strengthening of the copy of a CKPR with the data referred to in Table 1.

Table 5
Data for strengthened model of a CKP

k	$i \in N_k$	u_i^k	l_i^k	$l_i^{k^*}$	d_i^k	L_{min}^k	L_{min}^{k*}	L_{max}^k	$L_{max}^{k^*}$	p_k	p_{k^*}	L	F_1	F_1^*	F_2	F_2^k
1	1	0.82	7	10	3	9	9	12	10	2	2	23	14	4	8	1
	2	0.512	10	10	2											
	3	0.59	9	10	1											
2	1	0.283	11	11	3	5	7	13	11	4	3					1
	2	0.975	10	11	6											
	3	0.594	7	11	4											
	4	0.286	9	11	2											
3	1	0.979	8	8	1	10	12	14	14	2	1					2
	2	0.672	6	6	4											

2.1.4. Decreasing Symmetry of the Linear Model CKPR

Symmetry arises in an IP model when different solutions of the model, in terms of decision variables, correspond to the same or equivalent physical solutions. This phenomenon delays the search in the branch-and-bound tree, because the same physical solution is evaluated in different nodes of the tree. To realize how this phenomenon happens in the Linear model of Inarejos et al. (2017), consider the following example.

Example 4. Consider the data shown in Table 5 and focus on the items of class 2. One can construct at most 3 compartments for the class 2, that is, we have that $p_{2^*} = 3$ and therefore the model has 3 constraints that limit the minimum and maximum capacities of the compartments (see (3) above) involving the decision variables δ_1^2, δ_2^2 and δ_3^2.

Consider a physical solution with one compartment with one item 1 of class 2. There are 3 solution to the IP model corresponding to this same physical solution with different values of decisions variables, as follows:

1. $\delta_1^2 = 1, \delta_2^2 = 0, \delta_3^2 = 0$.
2. $\delta_1^2 = 0, \delta_2^2 = 1, \delta_3^2 = 0$.
3. $\delta_1^2 = 0, \delta_2^2 = 0, \delta_3^2 = 1$.

2.2. A Strong Linear Model of the Compartmentalized Knapsack Problem

With the adjustment of the capacity constraints for each class $k = 1, \ldots, q$ with $L_{min}^k \leq L_{min}^{k^*}$ and $L_{max}^{k^*} \leq L_{max}^k$, updating the maximum number of compartments that can be constructed by p_{k^*} where $p_{k^*} \leq p_k$, the adjustment of knife values by F_1^* and F_2^k and by making a simple lifting for some widths of the items, produces a *reduced domain* for the linear relaxation of CKPR. From the previous one, manipulating the linear model of Inarejos et al. (2017), one has a redefinition of the constraints of the model, as follows.

For the constraints (3) of the compartmentalization of the CKPR, the following constraint was reformulated by the stronger constraint presented in (10). The constraint (5) of the number of compartments that can be inserted into the knapsack is rewritten with the new knife value F_1^* as follows:

$$\sum_{k=1}^{q}\sum_{j=1}^{p_k}\delta_j^k \leq F_1^* \tag{32}$$

For the constraints on the number of items that can be inserted into the CKPR knapsack, constraints (6), for every class $k = 1, \ldots, q$, a new knife value F_2 has been defined, thus obtaining the new constraint:

$$\sum_{i\in N_k} a_{ij}^k \leq F_2^k, \text{ for all } j = 1, \ldots, p_k \text{ with } k = 1, \ldots, q \tag{33}$$

The new constraint (33) is only interesting if there is a compartment j of class k constructed, *i.e.*, $\delta_j^k = 1$. Therefore, the right-hand size can be multiplied by δ_j^k, strengthening the constraints:

$$\sum_{i\in N_k} a_{ij}^k \leq F_2^k\delta_j^k, j = 1, \ldots, p_k, k = 1, \ldots, q \tag{34}$$

To reduce the symmetry of the CKPR model, following the comments from subsection 2.1.4, the following constraints are added:

$$\delta_j^k \geq \delta_{j+1}^k, \forall j = 1, \ldots, p_k - 1 \text{ e } k = 1, \ldots, q \tag{35}$$

Now, with symmetry constraint, the demand constraint of the CKPR (4) of the items can be reinforced by making only the demand that was defined in the first compartment evaluated, that the compartments with symmetries were if j is a symmetry of the problem, we have that $\delta_j^k = 0$. Thus, the constraint of demand for CKPR items is redefined as follows:

$$\sum_{j=1}^{p_k} a_{ij}^k \leq d_i^k\delta_{i1}^k, i \in N_k, k = 1, \ldots, q \tag{36}$$

Finally, with the new strong constraints (10), (32), (34), (35) and (36) will define a new linear model for the CKPR, being this a model equivalent to the Linear Model of Inarejos et al. (2017), in which it will be called such as **The Strong Linear Model of the Compartmentalized knapsack Problem**, and being formulated as follows:

$$\text{Maximize: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_{k^*}} u_i^k a_{ij}^k \tag{37}$$

$$\text{subjet to: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_{k^*}} l_i^k a_{ij}^k \leq L \tag{38}$$

$$\delta_j^k L_{min}^{k^*} \leq \sum_{i \in N_k} l_i^{k^*} a_{ij}^k \leq \delta_j^k L_{max}^{k^*} \text{ com } j = 1, \ldots, p_{k^*}, k = 1, \ldots, q \tag{39}$$

$$\delta_j^k \geq \delta_{j+1}^k, \forall j = 1, \ldots, p_k - 1 \text{ e } k = 1, \ldots, q \tag{40}$$

$$\sum_{j=1}^{p_{k^*}} a_{ij}^k \leq d_i^k \delta_{i1}^k, \, i \in N_k, \, k = 1, \ldots, q \tag{41}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{p_{k^*}} \delta_j^k \leq F_1^* \tag{42}$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2^k \delta_j^k, \, j = 1, \ldots, p_{k^*}, k = 1, \ldots, q \tag{43}$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ and integer, } i \in N_k, \, j = 1, \ldots, p_{k^*}, k = 1, \ldots, q \tag{44}$$

The objective function (37) represents the sum of the utilities of all items chosen in all compartments. The constraint (38) ensures that the sum of the widths of the items is limited by the capacity of the knapsack. The constraints (39) limit the capacity of the non-zero compartments and override the compartment coefficients that will not be used with the control variable $\delta_j k$ ($\delta_j^k = 1$ if the j compartment of the k class is nonzero and $\delta_j^k = 0$ otherwise). The constraints (40) are the symmetry reduction constraints. The constraints (41) limit the quantity of each item $i \in N$ by its respective demand. The constraint (42) limits the number of non-zero compartments in the knapsack with a tight knife value and in (43) the number of items inserted in each compartment, where for each class there is an adjusted value F_2. And finally, the constraints (44) establish the domain of the variables.

3. Heuristics of p_k Strong Capacities

Taking advantage of strengthening the information it produces a reduced area for linear relaxation of a CKPR was developed one heuristic decomposition called Heuristics of p_k Strong Capacities. The heuristic is based on the Heuristic W Capacities of Marques and Arenales (2007), where sorting criteria of problem information is added along with limiting the number of compartments to be created.

First sort the classes by calculating the average of the efficiencies of your items. In the W Capacities Heuristic, the solution strategy is to create a portion of all feasible compartments for the CKPR so, using a restricted knapsack problem, make its solution. This portion is made by the generation of W compartments in each class. Second, for the Heuristics of p_k Strong Capacities, the value of W for each

class $k = 1, \ldots, q$ is bounded (and set) by the value of p_k (see the impli-cations of p_k in the section 1). In case, the information of a non-reinforcing CKPR is used, with an equal L_{min^k} for all classes, we have that $W = p_k$. Third, with the strengthening of the information, in particular from L_{min^k} to $L_{min^{k^*}}$ for each class, we have that $W(k) = p_{k^*}$, being in the best case fixed several values W.

The creation of each compartment for each class is done by solving the following constrained knapsack problem which generates the best compartment of the k class with capacity between L^{k*}_{min} and L_{cap}, where $L_{cap} \leq L^{k*}_{max}$:

$$\text{Maximize: } \sum_{i \in N_k} u_i^k a_{ij}^k \tag{45}$$

$$\text{subjet to: } L_{min}^{k^*} \leq \sum_{i \in N_k} u_i^k a_{ij}^k \leq L_{cap} \tag{46}$$

$$a_{ij}^k \leq d_i^k,\ i \in N \tag{47}$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2^k \tag{48}$$

$$a_{ij} \geq 0 \text{ and integer},\ i \in N \tag{49}$$

One has as an objective function (45), the constraint (46) limits the capacity of the compartment, the constraint (47) limits the amount of intensi i that can to be filled in the compartment by the available demand and the constraint (49) limits the number of items by the amount of knives available in the second cutting process.

With the generation of all the p_{k*} compartments for each class k is to define the utility of the compartment by U_j^k and the width of the compartment by L_{ij}^k, being calculated by (50) and (51).

$$U_j = \sum_{i \in N_k} u_i^k a_{ij}^k \tag{50}$$

$$L_j = \sum_{i \in N_k} l_i^k a_{ij}^k \tag{51}$$

Finally, by treating the compartments as items with utility (50) and weight (51), the following knapsack problem is used to determine how many compartments of each class have to be built in the knapsack:

$$\text{Maximize: } \sum_{j=1}^{W(1)} U_j y_j^k + \cdots + \sum_{j=1}^{W(q)} U_j y_j^k \tag{52}$$

$$\text{subject to: } \sum_{j=1}^{W(1)} L_j y_j^k + \cdots + \sum_{j=1}^{W(q)} L_j y_j^k \leq L \tag{53}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{W(k)} a_{ij}^k y_j^k \leq d_i^k, \forall i \in N_k \tag{54}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{W(k)} y_j^k \leq F_1^* \tag{55}$$

$$y_j^k \geq 0 \text{ and integer for } j = 1, \ldots, W(k) \text{ with } k = 1, \ldots, q \tag{56}$$

In this model (52) represents the objective function, the constraint (53) corresponds to the constraint of the capacity of the knapsack, the constraints (54) limit the quantity of items that can be used by its demand, the constraint (56) limits the number of compartments that can compose the knapsack by the number of knives available in the first cutting process and the constraint (56) that is domain of the problem. See the Algorithm 1 for the Heuristics of p_k Strong Capacities:

Algorithm 1 Heuristics of p_k Strong Capacities

sort the classes by calculating the average of the efficiency of the items
for $(k = 1, \ldots, q)$ **do**
  set $W(k) = p_{k^*}$;
  set $L_{cap} = L_{max}^{k^*}$;
  for (j=1,...,W(k)) **do**
    solve problem (45)-(49)
    save optimal values a_{ij}^k
    set $L_{cap} = \sum_{i \in N_k} l_j^k a_{ij}^k - 1$
  end for
end for
solve problem (52)-(56)

4. Computational Experiments

In this section is presented tests developed to validate the Linear Model Fort. Computational experiments were done using the FICO® Xpress suite for 64-bit architecture, with the following components: graphical interface FICO Xpress IVE version 1.24.20, language FICO® Xpress Mosel 4.8.1 version and optimization packages with FICO ® Xpress *Optimizer* 32.01.0 version. As computer equipment was used

a computer technical specifications:Intel® Inside™ Xeon® CPU W3520, four-core processor-based frequency of 2.67 GHz, cache of 8 MB and RAM of 8 GB under Microsoft Windows® 10.

In order to develop the evaluation, 4 categories of examples defined by class sizes were organized with $q = 5, 10, 30, 40$, and for each category were organized 5 sub-categories defined by the number of items each class, being represented by n, with $n = 5, 10, 20, 30, 40$. represent the previous one as the subcategories q/n.

To create the examples the following simulation parameters were used following realistic values of two-step steel coil cut-off problems of Hoto (2001): knapsack capacity $L = 1100\ mm$; values of knives $F_1 = 8$ and $F_2 = 12$, compartment capacities for each class $k = 1, \ldots, q$ for between $L^k_{min} = 154$ and $L^k_{max} = 456$, the widths of the items will be defined between the values $53\ mm$ and $230\ mm$ and the utilities of the items will be comprised between the values 0 and 1. To define the demand for each item, we defined for each class $k = 1, \ldots, q$ the *maximum demand of the class* in which it is divided into integers (different) as the demands of two items. This maximum demand will correspond three times the total number of items in the class.

With these parameters a random originator was created with reference in Gau and Wäscher (1995), where for a example of a category q/n is randomly designated the values of the widths and the utilities of the items, as well as the calculating the maximum demand for each class, partitioning this value and assigning it at random to each item. Following the previous one, 100 was created for each subcategory q/n, obtaining a total of 500 copies for each category q to have a total of 2000 examples. Each of these specimen was subjected to a process of adjustment of the values of the capacities of the compartments, of the knives and the values of the widths following the done in the sections 2, obtaining a total of 2000 adjusted examples.

4.1. CKPR's Strong Linear Model Computational Experiments

With the samples made, they were exposed to simulation under three models that solve the Compartmentalized Knapsack Problem: The Exhaustive Decomposition Algorithm Inarejos (2015) (EDA), the Linear Model of Inarejos et al. (2017) formulated in (1) - (7) (ML) and the Strong Linear Model (37)-(44) (SLM), which is under evaluation. The Exhaustive Decomposition Algorithm is an algorithm that generates all feasible compartments and then solves the problem using an unrestricted knapsack problem, obtaining the optimal solution of the example, which will be the referent of the quality of the solutions coming from the Strong Model. The results are presented in Table 6.

In the columns of Table 6, the following information is presented: the category corresponding to the tested examples, represented by "$q = k$", for $k = 5, 10, 30, 40$; the subcategory of each category is represented by n, for $n = 5, 10, 20, 30, 40$; averages of the optimal solutions of each test sample, represented by Obj; the average of the times of execution obtained in the resolution of each example q/n, represented by $\overline{T}$; the average of the standard deviations of the times referred to above, represented by $\sigma(T)$.

Some comments are made below the results obtained from Table 6. Firstly, it should be noted that the Strong Linear Model solved each test sample obtaining an optimum solution value equal to the obtained with the Exhaustive Decomposition Algorithm, a verification done to validate the implementation of the new CKPR strong model. It should also be noted that the SLM and the EDA can improve execution

times in most sub-categories. It can be highlighted that in each category the highest number of instances, that is, the largest number of items for each class, has a better behavior of the MLF, obtaining significant reductions in time.

Table 6
Results of the simulations of all categories of examples $q = 5, 10, 30, 40$ made with the Exhaustive Decomposition Algorithm, the Linear Model and the Strong Linear Model.

		Exhaustive Dec.Alg.			**Linear Model**					**Strong Linear Model**				
		$\overline{Obj}$	$\overline{T}$	$\sigma(T)$	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{T}$	$\overline{n_{BB}}$	$\overline{gap}$	gap_{min}	gap_{max}	$\overline{T}$	$\overline{n_{BB}}$
q=5	5	10,99	0,09	0,10	5,45%	0,44%	17,08%	0,46	667	4,15%	0,36%	14,76%	0,14	338
	10	11,08	2,44	5,58	1,89%	0,00%	4,05%	12,61	215237	1,65%	0,00%	3,31%	1,75	22095
	20	12,00	1,37	1,29	1,78%	0,00%	7,74%	1,76	19657	1,54%	0,00%	6,67%	0,79	6829
	30	16,01	4,60	0,72	3,77%	0,26%	7,78%	0,64	356	2,92%	0,26%	5,49%	0,35	299
	40	14,65	31,35	17,60	1,41%	0,00%	3,16%	4,92	61548	1,34%	0,00%	2,64%	5,68	56002
q=10	5	11,12	0,05	0,05	16,31%	3,27%	40,24%	0,73	117	10,16%	1,47%	28,82%	0,10	45
	10	11,51	1,15	2,58	2,44%	0,17%	4,43%	12,01	164836	1,69%	0,09%	3,19%	1,46	13088
	20	12,43	2,35	0,82	2,08%	0,00%	5,44%	1,14	1974	1,56%	0,00%	4,82%	0,44	1399
	30	15,72	29,14	58,69	2,40%	0,47%	4,33%	238,47	1871697	1,59%	0,09%	3,05%	24,10	230231
	40	8,62	65,74	25,33	0,51%	0,00%	1,42%	31,23	283609	0,51%	0,00%	1,42%	11,16	112476
q=30	5	14,43	0,15	0,18	6,95%	2,72%	13,42%	0,69	2534	1,82%	0,11%	4,51%	0,24	855
	10	11,99	3,32	13,77	2,48%	0,35%	4,75%	62,17	369001	1,36%	0,07%	3,12%	10,04	50105
	20	14,10	6,36	1,79	13,84%	1,73%	29,64%	137,86	1043677	5,24%	0,96%	12,34%	12,92	68463
	30	17,20	39,92	8,88	6,74%	0,50%	30,12%	62,90	242982	2,91%	0,49%	5,14%	5,32	32623
	40	15,30	326,53	210,91	2,08%	0,18%	5,50%	144,86	475543	1,82%	0,18%	2,50%	100,02	374520
q=40	5	14,46	0,25	0,22	6,72%	2,75%	11,17%	1,06	9210	2,27%	0,15%	4,86%	0,26	1250
	10	12,11	2,98	7,09	2,65%	0,80%	5,90%	209,29	928612	1,33%	0,07%	2,69%	25,49	122481
	20	13,88	10,00	1,90	10,91%	0,00%	27,45%	99,27	219957	3,66%	0,00%	11,06%	1,90	5116
	30	17,38	70,76	20,69	4,62%	0,00%	27,58%	164,84	646486	2,21%	0,00%	7,36%	5,23	39592
	40	15,62	573,94	151,14	1,93%	0,22%	5,84%	288,29	545855	1,50%	0,22%	2,84%	3,02	11226
average		13,53	58,62	26,47	4,85%	0,69%	12,85%	73,76	355177,8	2,56%	0,23%	6,53%	10,52	57451,7

Note that the computational behavior of the Linear Model under the parameters set for this paper is different from the obtained in Inarejos et al. (2017). In this work, we used different values for the minimum and maximum capacities of the compartments, similar to the used by Hoto (2001). In our experiments, for categories with few instances, the LM is no more efficient than the EDA in terms of execution times, but it has better behavior for large instances such as $n = 40$, with an approximate improvement of 90% for $q = 5$ and 50% for the other categories.

Now observe the computational behavior for the SLM. The Model for the smallest instances of items $n = 5, 10$ in all categories was not more efficient than the EDA, with the exception of sub-categories $5/10$. As the number of instances for each category is increased, the SLM shows a better behavior, standing out with the best execution time for the category $q = 40$, an improvement of approximately 80% for the categories $q = 5.10$ and 69% for the category $q = 30$.

Then make a comparison of LM and SLM models. The SLM obtained better results in the measure-

ments presented with a better average of $\overline{gap}$, gap_{min}, gap_{max}, $\overline{T}$ and $\overline{n_{BB}}$ compared to the Linear Model in all categories except $5/40$ for $\overline{T}$. We did an in-depth analysis of this $5/40$ sub-category, and observed that the range of times obtained with Linear Model are between 0.031 and 235.906, and for the SLM are between 0.031 and 193.860, where although SLM has a worse average time compared to LM, this has a better time interval. With the intervals, it is still seen that the SLM presents the best grouping of time results, averaging the standard deviation of times $20,641$ compared to the ML $26,566$.

The formulation of a reduced domain for the linear relaxation of CKPR (LRCKPR) is now validated. Two parameters will validate the reduction of the LRCKPR domain: the gap as the quality of the solution obtained in the LRCKPR and then the number of nodes developed for the process of *Branch and Bound*. It is seen that the SLM presented in all sub-categories of tested samples a reduction in the number of nodes in the Branch and Bound process, obtaining an average decrease of $\overline{n_{BB}}$ over 84%, having in agreement best gap in average $\overline{gap}$ over 47% compared to LM. From the previous one it confirms that the strengthening done in the information of the CKPR produces a reduced domain for the CKPR.

One final comment. There is a particularity detected in the simulations process that favored to a greater or lesser extent to have the best overall performance of the SLM: all the specimen presented different levels of strengthening. If they had copies where they were able to make important adjustments in the information of the problem, obtaining the best income of the SLM with these examples, for example in the category $q = 40$ this phenomenon was presented to a greater extent. From the previous one also implied that the strengthening of the information for individuals with large categories has a greater sensitivity in the adjustments, making that the SLM has a better computational behavior in comparison to LM. Another type of specimen presented little adjusted information (for example, it was only possible to adjust the values of some of the knives), as it was in the smaller categories, thus having a better performance of the SLM.

4.2. Computational Experiments of Heuristics of p_k Strong Capacities

The Heuristic of p_k Strong Capacities was subjected to simulation using the examples defined for the Strong Linear Model, as discussed at the beginning of this section. To develop a validation process, the the Heuristic of W Capacities Marques and Arenales (2007) and the Decreasing Classes Heuristic Inarejos et al. (2017) were also simulated. The results are presented in Table 7.

In the Table 7 the following information is presented: the category corresponding to the tested examples, represented by "$q = k$", for $k = 5, 10, 30, 40$; the subcategory of each category is represented by n, for $n = 5, 10, 20, 30, 40$; the average of the gap, obtained in the comparison of the solution of the heuristic with the optimal solution, being represented by $\overline{gap}$, the average of the standard deviations of gap represented by σgap and the average of execution times obtained in the resolution of each example q/n, represented by $\overline{T}$; the average of the standard deviations of the times referred to above, represented by $\sigma(T)$.

Following are some comments made under the results. Note that the Heuristic of p_k Strong Capacities does not present resolution of the examples faster than the Heuristic of W Capacities, but it provides better gap. Comparing now to the Decreasing Classes Heuristic, it behaves similarly in the gap (a difference of 2%) but with a better performance at resolution times, being 7.5% faster. For the lowest category $q = 5$, the Heuristic of p_k Strong Capacities was better in the gap results, and for the highest category

$q = 40$, it was was most efficient at the time of execution for the Decreasing Classes Heuristic.

Table 7
Results of the simulations of all categories of examples $q = 5, 10, 30, 40$ made with the Heuristic of W Capacities, The Decreasing Classes Heuristic and The Heuristic of p_k Strong Capacities.

		Heuristic of W Capacities				**The Decreasing Classes Heuristic**				**Heuristics of p_k Strong Capacities**			
		$\overline{gap}$	$\sigma(gap)$	$\overline{T}$	$\sigma(T)$	$\overline{gap}$	$\sigma(gap)$	$\overline{T}$	$\sigma(T)$	$\overline{gap}$	$\sigma(gap)$	$\overline{T}$	$\sigma(T)$
$q = 5$	5	1.87%	2.31%	0.131	0.039	2.51%	3.01%	0.138	0.035	1.24%	1.92%	0.163	0.042
	10	18.10%	5.08%	0.248	0.062	17.89%	5.24%	0.305	0.077	15.00%	7.17%	0.353	0.078
	20	16.56%	7.26%	0.602	0.104	14.11%	7.78%	0.715	0.114	10.89%	7.52%	0.832	0.109
	30	11.65%	9.70%	1.014	0.117	6.25%	5.02%	1.130	0.140	3.99%	2.63%	1.418	0.163
	40	20.00%	1.69%	0.282	0.068	20.00%	1.69%	0.333	0.078	20.00%	1.69%	1.032	0.369
$q = 10$	5	2.81%	3.24%	0.269	0.053	0.96%	2.08%	0.345	0.062	1.70%	2.65%	0.314	0.065
	10	17.01%	6.93%	0.512	0.112	9.20%	6.43%	0.910	0.261	12.34%	7.63%	0.760	0.195
	20	18.54%	7.42%	1.313	0.157	6.67%	4.09%	2.058	0.299	8.12%	4.68%	1.866	0.229
	30	20.93%	1.83%	1.872	0.195	18.85%	5.11%	2.935	0.306	19.77%	4.17%	2.603	0.264
	40	19.00%	0.94%	1.231	0.278	17.37%	4.41%	2.011	0.481	18.38%	2.90%	3.342	0.567
$q = 30$	5	1.08%	1.56%	0.680	0.056	0.65%	1.23%	0.750	0.070	0.54%	1.08%	0.712	0.054
	10	9.85%	4.63%	1.557	0.221	5.51%	1.99%	2.146	0.268	7.13%	3.26%	1.800	0.290
	20	19.26%	9.18%	3.186	0.281	10.59%	5.47%	4.217	0.357	11.63%	5.06%	3.696	0.315
	30	6.70%	5.21%	5.149	0.513	3.57%	3.56%	6.499	0.773	4.05%	3.68%	5.807	0.650
	40	21.13%	1.77%	2.682	0.353	17.53%	7.30%	4.488	0.497	17.86%	6.81%	4.067	0.835
$q = 40$	5	1.75%	1.81%	0.931	0.055	0.59%	1.13%	1.022	0.084	0.47%	1.03%	0.902	0.051
	10	9.46%	4.69%	2.058	0.251	5.50%	2.34%	2.904	0.283	6.71%	2.65%	2.375	0.291
	20	20.14%	7.08%	4.203	0.312	12.43%	8.36%	5.834	0.417	13.94%	8.28%	4.826	0.370
	30	2.77%	3.37%	6.323	0.583	1.31%	2.42%	8.225	0.780	1.57%	2.54%	7.144	0.679
	40	22.46%	1.81%	3.993	0.490	21.90%	3.63%	6.936	0.598	21.92%	3.53%	6.144	1.007
average		13.05%	4.38%	1.912	0.215	9.67%	4.11%	2.695	0.299	9.86%	4.04%	2.508	0.331

5. Conclusions

In this paper a new strong integer linear programming model for the exact solution of the Restricted Compartmentalized Knapsack Problem and a new pseudo-polynomial p_k Strong Capacities heuristic were presented. The new model improves a recent contribution from the literature that introduced an integer (linear) model for the problem, by strengthening data, reducing symmetry and lifting. Computational experiments show the advantage of deriving stronger models, reaffirming the importance of Integer Programming modelling. The size of the enumeration trees was significantly reduced, and the CKP was solved exactly more than seven times faster, leading to large computational savings.

Furthermore, establishing a reduced domain for Linear Model for the Restricted Compartmentalized Knapsack Problem, by defining stronger constraints, with the strengthening of information and the limitation of the compartments to be created for each class in the knapsack, helped in deriving a new pseudo-polynomial decomposition heuristic for the Restricted Compartmentalized Knapsack Problem, obtaining

attractive solutions with acceptable execution times.

These computational results enable aiming at solving the cutting stock problem with compartmentalized cutting patterns, in which the objective is to cut the steel coils in stock using cutting patterns that are compartmentalized knapsacks, in order to optimize an efficiency function, as for instance the number of steel coils used. The gains in efficiency with the Compartmentalized Knapsack Problem show that we may deem at solving this problem in reasonable computational times. The best strategy is possibly to develop a column generation model with a two-level generation scheme using the heuristic to get a good approximation of the optimal solution, and relying in the exact solution method to guarantee that the optimal solution to the linear programming relaxation of the column generation model is obtained.

Acknowledgments

The first author has been supported by CAPES through the Partnerships Program for Education and Training (PAEC) between the Organization of American States (OAS) and the Coimbra Group of Brazilian Universities (GCUB). The second author has been supported by COMPETE: POCI-01-0145- FEDER-007043 and FCT - Fundação para a Ciência e a Tecnologia within the Project Scope: UID/CEC/00319/2013. We thank to FICO® for giving us the Xpress suite.

References

Leão A., A.S., 2013. Extensões em problemas de corte: padrões compartimentados e problemas acoplados. Tese de doutorado em ciências da computação e matemática computacional, Universidade de São Paulo.

Arenales, M., Armentano, V., Morabito, R., Yanasse, H., 2015. *Pesquisa operacional: para cursos de engenharia*. Elsevier Brasil.

Valério de Carvalho, J.M., Rodrigues, A., 1994. A computer based interactive approach to a two-stage cutting stock problem. *INFOR: Information Systems and Operational Research* 32, 4, 243–252.

Valério de Carvalho, J.M., Rodrigues, A., 1995. An LP based approach to a two–phase cutting stock problem. *European Journal of Operational Research* 84, 580–589.

Chvátal, V., 1983. *Linear programming*. WH Freeman and Company, New York.

Cruz, E.P., 2010. Uma abordagem heurística linear para mochilas compartimentadas restritas. Dissertação de Mestrado em Matemática Aplicada e Computacional, Universidade Estadual de Londrina.

Ferreira, J.S., Neves, M., Castro, P., 1990. A two–phase roll cutting problem. *European Journal of Operational Research* 44, 185–196.

Gau, T., Wäscher, G., 1995. CUTGEN1: a problem generator for the standard one-dimensional cutting stock problem. *European Jornal of Operational Research* 84, 3, 572–579.

Haessler, R., 1979. Solving the two–stage cutting stock problem. *Omega, The International Journal of Management Science* 7, 145–171.

Hoto, R., Arenales, M., Maculan, N., 1999. O Problema da Mochila Compartimentada. Relatório técnico do departamento de matemáticas, Universidade Estadual de Londrina.

Hoto, R., Fenato, A., Yannasse, H., Maculan, N., Spolador, F., 2006. Uma nova abordagem para o problema da mochila compartimentada.

Hoto, R., Fenato, S., Marques, F., 2005. Resolvendo mochilas compartimentadas restritas.

Hoto, R., Maculan, N., Marques, F., Arenales, M., 2003. Um problema de corte com padrões compartimentados. *Pesquisa Operacional* 23, 169 – 187.

Hoto, R.S., 1996. Otimização no corte de peças unidimensionais com restrições de agrupamento. Dissertação de mestrado em

matemática aplicada, ICMSC-USP.

Hoto, R.S., 2001. O Problema da mochila compartimentada aplicado no corte de bobinas de aço. Tese de doutorado em Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro.

Inarejos, O., Hoto, R., Maculan, N., 2017. An integer linear optimization model to the compartmentalized knapsack problem. *International Transactions in Operational Research (in press)*. doi: 10.1111/itor.12490.

Inarejos, O.F., 2015. Sobre a não linearidade do problema da mochila compartimentada. Dissertação de mestrado em matemática aplicada e computacional, Universidade Estadual de Londrina.

Leão, A.A., Santos, M.O., Hoto, R., Arenales, M.N., 2011. The constrained compartmentalized knapsack problem: mathematical models and solution methods. *European Journal of Operational Research* 212, 3, 455 – 463.

Marques, F., 2000. O problema da mochila compartimentada e aplicações. Tese de doutorado em ciências da computação e matemática computacional, Universidade de São Paulo.

Marques, F., Arenales, M.N., 2002. O Problema da mochila compartimentada e aplicacações. *Pesquisa Operacional* 22, 285 – 304.

Marques, F., Arenales, M.N., 2007. The constrained compartmentalised knapsack problem. *Computers & Operations Research* 34, 7, 2109 – 2129.

Martello, S., Toth, P., 1990. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc.

Martin, R.K., 1999. *Large Scale Linear and Integer Optimization*. Kluwer Academic Publishers.

Nemhauser, G.L., Wolsey, L.A., 1988. *Integer and Combinatorial Optimization*. Wiley Interscience in Discrete Mathematics nad Optimization.

Pereira, M.A., 1993. Uma abordagem matemática para o problema de corte e laminação de fitas de aço. Dissertação de Mestrado em Matemática Aplicada, Unicamp.

Quiroga-Orozco, J.J., 2018. Um método branch and bound para o problema da compartimentação das mochilas. Dissertação de Mestrado em Matemática Aplicada e Computacional, Universidade Estadual de Londrina.