Data de Depósito: 25.10.2004

Assinatura:

O problema da mochila compartimentada e aplicações*

Fabiano do Prado Marques

Orientador: Prof. Dr. Marcos Nereu Arenales

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional.

**USP – São Carlos
Outubro de 2004**

* Projeto financiado pela FAPESP.

A Comissão Julgadora:

Prof. Dr. Marcos Nereu Arenales

Prof. Dr. Reinaldo Morabito Neto

Prof. Dr. Cid Carvalho de Souza

Prof. Dr. Horácio Hideki Yanasse

Prof. Dr. Carlos Eduardo Ferreira

Agradecimentos

A Deus, pela força que tem me dado durante toda a minha vida.

Ao Marcos Arenales pela orientação, amizade e por ter acreditado, a todo instante, na busca por um desafio maior.

Aos meus pais, Carlos e Regina, e meus irmãos, Silvana, Renata e Tiago, sobrinhos e cunhado pelo amor, incentivo e presença em todos os anos de minha vida.

A Ana pelo carinho, paciência, ajuda e compreensão.

Ao amigo Robinson pelas dicas e grande ajuda durante o desenvolvimento desta pesquisa.

Aos amigos do laboratório, funcionários e professores do ICMC – USP.

Aos amigos e amigas dos times de futsal do CAASO.

Aos amigos de Araxá que mesmo à distância sempre estiveram comigo.

A FAPESP pelo apoio financeiro.

E finalmente, aos conhecidos fatores externos...

“O homem que venceu na vida é aquele que viveu bem, riu muitas vezes e amou muito; que conquistou o respeito de homens inteligentes e o amor das crianças, que preencheu um lugar e cumpriu uma missão; que deixa o mundo melhor do que encontrou, seja com uma flor, um poema perfeito ou o salvamento de uma alma; que procurou o melhor nos outros e deu o melhor de si”.

(R. L. Stevenson)

Resumo

Este trabalho aborda o *Problema da Mochila Compartimentada* que é uma variação do clássico problema da mochila e pode ser enunciado considerando-se a seguinte situação hipotética: um alpinista deve carregar sua mochila com possíveis itens de seu interesse. A cada item atribui-se o seu *peso* e um *valor de utilidade* (até aqui, o problema coincide com o clássico Problema da Mochila). Entretanto, os itens são de *agrupamentos* distintos (alimentos, medicamentos, utensílios, etc.) e devem estar em compartimentos separados na mochila. Os compartimentos da mochila são flexíveis e têm capacidades limitadas. A inclusão de um compartimento tem um *custo* fixo que depende do agrupamento com que foi preenchido, além de introduzir uma perda da capacidade da mochila. O problema consiste em determinar as capacidades adequadas de cada compartimento e como esses devem ser carregados, maximizando o valor de utilidade total, descontado o custo de incluir compartimentos. Neste trabalho propomos um modelo de otimização não linear inteiro para o problema e algumas heurísticas para sua resolução, para as quais apresentamos os resultados computacionais obtidos. Uma aplicação prática que surge no corte de bobinas de aço, sujeito à laminação é detalhada.

Palavras-chave: Problemas da Mochila, Problemas de Corte e Empacotamento, Otimização Inteira e Combinatória.

Abstract

This work approaches the Compartmentalized Knapsack Problem which is a variation of the classical knapsack problem and it can be stated as the following hypothetical situation: a climber must load his/her knapsack with a number of items. Two numbers, an weight and a utility value are given for each item (so far, the problem coincides with the classical integer Knapsack Problem). However, the items are of different classes (food, medicine, utensils, etc.) and they have to be packed in separate compartments inside the knapsack. The compartments are flexible and have limited capacities. Each compartment has a fixed cost that depends on the class of items chosen to load it and, in addition, each new compartment introduces a loss of capacity of the knapsack. The Compartmentalized Knapsack Problem consists of determining suitable capacities of each compartment and how these compartments should be packed. The objective is to maximize a total utility value paid off the cost of the compartments. The problem can be modeled as an integer non-linear optimization problem and we have designed some heuristic methods for its solution, for which we present a computational study. A practical application arises in cutting steel rolls subject to a lamination process.

Keywords: Knapsack Problem, Cutting and Packing Problems, Integer Programming.

Lista de Tabelas

Capítulo 1. Introdução	14
Capítulo 2. Problemas da Mochila.....	21
Capítulo 3. O Problema da Mochila Compartimentada.....	40
Tabela 3.1. Dados do agrupamento 1	43
Tabela 3.2. Dados do agrupamento 2	43
Tabela 3.3. Dados do agrupamento 3 (<i>itens livres</i>).....	43
Tabela 3.4. Dados dos itens dos agrupamentos 1 e 2.....	61
Capítulo 4. Heurísticas para o Problema da Mochila Compartimentada	67
Tabela 4.1. Dados do agrupamento 1	74
Tabela 4.2. Dados do agrupamento 2	74
Tabela 4.3. Dados do agrupamento 3 (<i>itens livres</i>).....	74
Tabela 4.4. Resultados obtidos pela Heurística de Decomposição	75
Tabela 4.5. Resultados obtidos pela Heurística do Melhor Compartimento.....	77
Tabela 4.6. Resultados obtidos pela Heurística dos “z” Melhores Compartimentos.....	78
Tabela 4.7. Resultados obtidos pela Heurística do Melhor Compartimento para “w” Capacidades.....	78
Capítulo 5. Experimentos Computacionais.....	80
Tabela 5.1. Valores médios da função objetivo, obtidos executando-se as Heurísticas de Decomposição e do Melhor Compartimento	83

Tabela 5.2. Valores médios da função objetivo obtidos executando a Heurística dos “z” Melhores Compartimentos, com “z” = 2, 3 e 5	85
Tabela 5.3. Valores médios da função objetivo obtidos executando a Heurística do Melhor Compartimento para “w” Capacidades, com “w” = 2, 3 e 5	86
Tabela 5.4. Valores médios da função objetivo alcançados por todas as heurísticas para os conjuntos de dados.....	87
Tabela 5.5. Evolução das soluções das heurísticas em proximidade do valor ótimo.....	89
Tabela 5.6. Tempos médios, em milissegundos, de cada heurística para os exemplos resolvidos	92
 Capítulo 6. O Problema de Corte de Estoque.....	94
 Capítulo 7. Conclusões e Trabalhos Futuros	102
 Apêndice A. Algoritmos de Enumeração Implícita.....	110
 Apêndice B. Características de Implementação para o Problema da Mochila Compartimentada.....	124

Lista de Figuras

Capítulo 1. Introdução	14
Figura 1.1. Padrão de Corte.....	15
Figura 1.2. Problema de corte bidimensional.....	17
Figura 1.3. Problema de corte tridimensional	17
 Capítulo 2. Problemas da Mochila.....	21
Figura 2.1. Corte de bobina com restrição no número de facas	24
Figura 2.2. Árvore de decisões.....	33
Figura 2.3. Descida na árvore de decisões para a obtenção de uma solução factível melhorada	36
Figura 2.4. Operação de retorno curto.....	37
Figura 2.5. <i>Backtracking</i> saltando vários nós intermediários (retorno longo)	39
 Capítulo 3. O Problema da Mochila Compartimentada.....	40
Figura 3.1. Representação da capacidade do objeto.....	42
Figura 3.2. Representação dos itens	43
Figura 3.3. Possíveis compartimentos para os agrupamentos do exemplo numérico	44
Figura 3.4. Padrão de corte compartimentado.....	44
Figura 3.5. Os comprimentos dos itens 1 e 2 devem ser iguais para serem cortados na mesma recortadeira	45
Figura 3.6. Esquema do processo de corte de uma bobina mestre em indústrias de corte de aço.....	46
Figura 3.7. O processo de corte em bobinas de aço	47
Figura 3.8. Corte em bobinas de aço com refugos embutidos	47
Figura 3.9. Perfil do processo de corte em bobinas de aço	48

Figura 3.10. Perfil do processo de laminação	49
Figura 3.11. Laminação das bobinas intermediárias em uma mesma bobina mestre	50
Figura 3.12. Fluxo da produção de tubos de aço.....	51
Figura 3.13. Solução infactível para o problema e admitida por (16.1 – 16.5).....	57
Figura 3.14. Padrão de corte segundo Carvalho (1991).....	62
Figura 3.15. Ilustração da heurística de Hoto (1996).....	65
Capítulo 4. Heurísticas para o Problema da Mochila Compartimentada	67
Figura 4.1. Representação dos itens	74
Figura 4.2. Representação do padrão de corte gerado pela Heurística de Decomposição	75
Figura 4.3. Representação do padrão de corte gerado pela Heurística do Melhor Compartimento.....	77
Figura 4.4. Representação do padrão de corte gerado pela Heurística dos “z” Melhores Compartimentos ($z = 2$).....	77
Figura 4.5. Representação do padrão de corte gerado pela Heurística do Melhor Compartimento para “w” Capacidades ($w = 2$)	78
Capítulo 5. Experimentos Computacionais.....	80
Figura 5.1. Gráfico com as curvas que representam os valores médios da função objetivo, obtidos executando as Heurísticas de Decomposição e do Melhor Compartimento.....	84
Figura 5.2. Gráfico com as curvas comparativas que representam os valores médios da função objetivo obtidos executando a Heurística dos “z” Melhores Compartimentos	85
Figura 5.3. Gráfico com as curvas comparativas que representam os valores médios da função objetivo obtidos executando a Heurística do Melhor Compartimento para “w” Capacidades	86
Figura 5.4. Gráfico com as curvas comparativas que representam os valores médios da função objetivo obtidos executando as quatro heurísticas	88
Figura 5.5. Gráfico com as curvas que mostram a evolução das soluções das heurísticas em proximidade do valor ótimo	89
Figura 5.6. Gráfico de tempos médios de cada heurística para os exemplos resolvidos.	93

Capítulo 6. O Problema de Corte de Estoque	94
Figura 6.1. Problema de corte de estoque com vários tipos de bobinas em estoque.....	95
Figura 6.2. Exemplos de padrões de corte factíveis para o problema de corte de estoque com vários tipos de bobinas em estoque	96
Figura 6.3. Laminação das bobinas intermediárias de uma mesma bobina mestre	98
Figura 6.4. Relação $\eta_{kb} \times R_{kb}$	99
Figura 6.5. Relação custo de laminação $\times \eta_{kb}$	99
 Capítulo 7. Conclusões e Trabalhos Futuros	102
 Apêndice A. Algoritmos de Enumeração Implícita.....	110
 Apêndice B. Características de Implementação para o Problema da Mochila Compartimentada.....	124
Figura B.1. Registro para armazenamento dos dados	125
Figura B.2. Esboço da disposição das listas de armazenamento dos dados.....	126
Figura B.3. Geração de um padrão de corte utilizando as listas encadeadas	127
Figura B.4. Geração de um padrão de corte utilizando as listas encadeadas para as heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w”Capacidades	128
Figura B.5. DFD de nível 0 para o programa desenvolvido	129
Figura B.6. DFD expandido para o programa em desenvolvimento.....	130
Figura B.7. DFD para a Heurística de Decomposição	131
Figura B.8. DFD para a Heurística do Melhor Compartimento.....	131
Figura B.9. DFD para a Heurística dos “z” Melhores Compartimentos.....	131
Figura B.10. DFD para a Heurística dos Melhores Compartimentos para “w” Capacidades.....	132

Sumário

Capítulo 1. Introdução	14
1.1. Classificação dos Problemas de Cortes.....	16
1.1.1. Problema Unidimensional.....	16
1.1.2. Problema Bidimensional.....	16
1.1.3. Problema Tridimensional.....	17
1.1.4. Problemas $1\frac{1}{2}$ -dimensional e $2\frac{1}{2}$ -dimensional.....	18
1.1.5. Problema Multidimensional.....	18
1.2. Organização do Texto	19
 Capítulo 2. Problemas da Mochila.....	21
2.1. Considerações Iniciais.....	21
2.1.1. Problema da Mochila Inteiro	22
2.1.2. Problema da Mochila 0-1.....	22
2.1.3. Problema da Mochila Restrito	23
2.1.4. Problema da Mochila de Escolha Múltipla.....	24
2.1.5. Problema da Soma de Subconjuntos.....	25
2.1.6. Problema do Troco.....	25
2.1.7. Problema da Mochila Quadrático	26
2.1.8. Problema de Múltiplas Mochilas 0-1	27
2.1.9. Problema de Designação Generalizada.....	28
2.1.10. Problema de Múltiplas Mochilas Idênticas (<i>Bin-Packing</i>)	29
2.1.11. Problema da Mochila Subdividida (<i>Nested Knapsack Problem</i>).....	30
2.2. Um Método de Resolução para o Problema da Mochila.....	31

2.2.1. Considerações Iniciais	32
2.2.2. O Método Enumeração Implícita de Gilmore e Gomory (1963).....	34
Capítulo 3. O Problema da Mochila Compartimentada.....	40
3.1. Definição do Problema.....	40
3.2. Um Exemplo Numérico para o Problema da Mochila Compartimentada	41
3.3. Uma Aplicação Prática: O Problema de Corte de Bobinas de Aço com Laminação.....	45
3.3.1. Considerações Iniciais	45
3.3.2. O Processo de Corte.....	46
3.3.3. A Cortadeira.....	48
3.3.4. Laminação e Recorte.....	49
3.3.5. Custos dos Processos Envolvidos no Problema.....	51
3.4. Um Modelo Matemático para o Problema da Mochila Compartimentada	51
3.5. Um Limitante Superior para o Problema da Mochila Compartimentada	54
3.5.1. Solução do Modelo (16.1–16.5) Inviável para o Problema da Mochila Compartimentada.....	56
3.5.2. Relaxação de Integralidade (Novo Limitante Superior)	57
3.6. Estudos Relacionados com Problema da Mochila Compartimentada.....	59
3.6.1. O Problema da Mochila Compartimentada Irrestrito.....	59
3.6.2. Abordagens Heurísticas	61
3.6.3. Uma Abordagem por Programação Linear Inteira (Pereira, 1993).....	65
3.6.4. Abordagens para o Problema de Corte em Bobinas de Papel em Duas Fases.....	66
Capítulo 4. Heurísticas para o Problema da Mochila Compartimentada.....	67
4.1. Heurística de Decomposição.....	67
4.2. Heurística do Melhor Compartimento	69
4.3. Heurística dos “z” Melhores Compartimentos.....	71
4.4. Heurística do Melhor Compartimento para “w” Capacidades.....	72
4.5. Um Exemplo Numérico para o Problema da Mochila Compartimentada	73
4.5.1. Heurística de Decomposição	75
4.5.2. Heurística do Melhor Compartimento.....	75
4.5.3. Heurística dos “z” Melhores Compartimentos ($z = 2$)	77

4.5.4. Heurística do Melhor Compartimento para “w” Capacidades ($w = 2$)	78
Capítulo 5. Experimentos Computacionais.....	80
5.1. Considerações Iniciais.....	80
5.2. Conjuntos de Dados	80
5.3. Resultados Obtidos.....	83
5.4. Análise dos Resultados Obtidos.....	90
5.5. Análise dos Tempos de Execução.....	92
Capítulo 6. O Problema de Corte de Estoque.....	94
6.1. Considerações Iniciais.....	94
6.2. O Corte de Bobinas de Aço com Vários Tipos de Bobinas em Estoque	95
6.3. Definição de uma Função Custo para o Problema de Corte de Estoque Envolvendo Mochilas Compartimentadas.....	97
Capítulo 7. Conclusões e Trabalhos Futuros	102
Referências Bibliográficas	105
Apêndice A. Algoritmos de Enumeração Implícita.....	110
A.1. Considerações Gerais	110
A.2. O Problema Restrito	113
A.3. As “z” Melhores Soluções.....	115
A.4. O Problema (31.1-31.5).....	118
Apêndice B. Características de Implementação para o Problema da Mochila Compartimentada.....	124
B.1. Recursos Técnicos	124
B.2. Estruturas de Armazenamento dos Dados do Problema.....	124
B.3. Fluxo das Informações no Programa	129
B.4. Diagramas de Fluxo de Dados para as Heurísticas Propostas para o Problema da Mochila Compartimentada.....	130

Capítulo 1. Introdução

Cortar objetos grandes (por exemplo, bobinas, placas, paralelepípedos) para a produção de itens menores em quantidades bem definidas, ou empacotar itens pequenos dentro de espaços bem definidos são problemas idênticos, considerando que um item cortado de uma certa posição pode ser visto como ocupando aquela posição (daí a referência na literatura a Problemas de Corte e Empacotamento).

Os problemas de cortes são essenciais para o planejamento da produção em muitas indústrias, tais como indústrias de papel, vidro, móveis, metalúrgica, plástica, têxtil, etc. Nessas indústrias, a redução dos custos de produção e a melhoria da eficiência estão, freqüentemente, associadas à utilização de estratégias adequadas de cortes.

A importância econômica e operacional de tais problemas, bem como a dificuldade de resolução e os trabalhos pioneiros de Gilmore e Gomory, (1961, 1963, 1965), despertaram grande interesse na comunidade de pesquisadores da otimização combinatória por problemas de corte e empacotamento. Tal interesse pode ser notado pelos vários trabalhos de revisão em problemas de corte e empacotamento que têm sido publicados nas últimas décadas, como (Brown, 1971), (Hinxman, 1980), (Dyckhoff et al., 1985), (Dyckhoff, 1990), (Martello e Toth, 1990), (Dowsland e Dowsland, 1992), (Dyckhoff e Finke, 1992), (Sweeney e Parternoster, 1992), (Wäscher e Gau, 1996), (Dyckhoff et al., 1997), (Lodi et al., 2002). Além disso, revistas de prestígio em Pesquisa Operacional têm publicado edições especiais sobre o tema tais como (Dyckhoff e Wäscher, 1990), (Morabito e Arenales, 1992), (Bischoff e Wäscher, 1995), (Arenales et al., 1999) e (Wang e Wäscher, 2002).

O problema de empacotamento consiste em empacotar itens pequenos dentro de um espaço maior, de forma que certo objetivo seja otimizado. Um exemplo desse problema consiste em arranjar o maior volume possível de caixas dentro de um contêiner.

Por outro lado, o problema de corte consiste em cortar um objeto grande, que esteja disponível, para a produção de um conjunto de itens menores requisitados. As formas e medidas do objeto e dos itens são bem especificadas.

Conforme os itens solicitados, o número de combinações possíveis destes dentro de um objeto é, em geral, muito grande. A cada combinação possível denominamos padrão de corte. A tentativa de enumerar todos os padrões de corte é inviável do ponto de vista prático.

Uma função objetivo pode ser definida medindo, por exemplo, perdas no caso do problema de corte, ou vazios no caso de empacotamento, ou custos, etc. É comum que funções objetivos conflitantes sejam definidas (Pileggi, 2002). Um problema de corte e empacotamento consiste em determinar um padrão de corte que otimize uma certa função objetivo.

O número elevado de padrões de corte possíveis exige que técnicas bem elaboradas sejam desenvolvidas para determinar o padrão ótimo ou quase-ótimo. Dentre essas técnicas podemos citar: enumeração implícita, programação dinâmica, relaxação lagrangiana, busca em grafos e heurísticas. Na figura 1.1 podemos visualizar um exemplo de padrão de corte gerado para uma bobina.

Figura 1.1: Padrão de corte.

Vale salientar que, dificilmente obtemos um padrão de corte que utilize todo o objeto. Temos, então, uma perda como ilustra a figura 1.1.

Os problemas de cortes, conforme Garey e Johnson (1979), pertencem a uma classe de problemas denominada NP-completo. Grosso modo, podemos dizer que são problemas improváveis de serem resolvidos num tempo limitado por uma função polinomial. É interessante observar que existem resultados teóricos afirmando que, se um problema desta classe puder ser resolvido em um tempo polinomial, então todos os problemas da classe terão solução em tempo polinomial. Porém parece pouco provável a obtenção de tal resultado. Este tem sido um dos motivos da pesquisa em métodos heurísticos que buscam soluções “boas” em sacrifício da otimalidade.

Quando uma quantidade elevada de itens deve ser produzida, temos um problema em que a solução exige o corte de vários objetos em estoque e a repetição de vários padrões de corte. Este problema é conhecido na literatura como problema de corte de estoque e o objetivo pode ser, entre outros, o menor número de objetos cortados, ou o menor custo total dos objetos cortados, considerando diferentes custos para os objetos em estoque. Como ressaltado anteriormente, tal problema é de grande importância para o planejamento da produção em muitas indústrias e uma aplicação prática será discutida posteriormente com algum detalhe.

1.1 Classificação dos Problemas de Cortes

Há, na literatura, um grande número de problemas de corte e empacotamento e houve uma tentativa no trabalho de Dyckhoff e Finke (1992) de classificá-los conforme algumas características, como: dimensionalidade, tipo de seleção dos objetos/itens, formas dos objetos/itens, entre outros. Entretanto, uma mesma classe pode abrigar problemas de grande diversidade, no sentido de que modelos e métodos adequados a um problema não se estendem a outro da mesma classe. A seguir, distinguimos os problemas pela sua dimensionalidade.

1.1.1 Problema Unidimensional

Um problema é dito unidimensional quando apenas uma dimensão é relevante no processo de corte. Como casos típicos de problemas unidimensionais podemos citar o corte de materiais como papel, tecido, plástico e aço, para serem utilizados nos mais diversos setores. A figura 1.1 exibe um corte unidimensional.

O problema de corte unidimensional mais simples pode ser modelado pelo clássico problema da mochila, para o qual faremos uma revisão no capítulo 2.

1.1.2 Problema Bidimensional

No problema bidimensional, duas dimensões (comprimento e largura) são relevantes na obtenção da solução, enquanto a espessura é constante. As dificuldades aumentam consideravelmente para se gerar arranjos sem que ocorra sobreposição de itens nos padrões de corte. A figura 1.2 exibe uma representação de problemas de corte em duas dimensões, em

que o objeto e os itens a serem cortados são retângulos. Obviamente, tanto o objeto como os itens poderiam ter formas gerais.

Figura 1.2: Problema de corte bidimensional.

Entre os problemas bidimensionais podemos citar alguns já bastante estudados, como o corte de placas de madeira na indústria de móveis, chapas de aço, placas de vidro, empacotamento de paletes, entre outros. Alguns estudos de problemas bidimensionais podem ser encontrados em (Gilmore e Gomory, 1965), (Herz, 1972), (Christofides e Whithlock, 1977), (Wang, 1983), (Beasley, 1985), (Yanasse *et al.*, 1990), (Morabito e Arenales, 1995) e (Gramani, 1997).

1.1.3 Problema Tridimensional

No problema tridimensional, três dimensões (comprimento, largura e altura) são relevantes para a obtenção da solução. Trata-se de arranjar itens espaciais, sem interseccioná-los, dentro de objetos maiores. Podemos citar como exemplos de problemas tridimensionais o Problema de Carregamento de Contêineres e cortes em indústrias de colchões (Morabito, 1992). A figura 1.3 exibe uma representação de problemas de cortes em três dimensões.

Figura 1.3: Problema de corte tridimensional.

1.1.4 Problemas $1\frac{1}{2}$ -dimensional e $2\frac{1}{2}$ -dimensional

Ainda sob o aspecto geométrico, é possível encontrar problemas que são essencialmente bidimensionais, porém uma das duas dimensões consideradas é variável. Tais problemas são chamados $1\frac{1}{2}$ -dimensionais. Este caso tem aplicação no corte de peças de vestuário. Outros são problemas do tipo $2\frac{1}{2}$ -dimensionais, onde uma das três dimensões é variável. Uma aplicação é o problema de se efetuar o carregamento de unidades dentro de caixas abertas, ou seja, as bases estão definidas, mas a altura deverá ser definida (Dyckhoff e Finke, 1992).

1.1.5 Problema Multidimensional

Além dos problemas já expostos, problemas multidimensionais também podem surgir. Uma ocorrência desse tipo de problema pode aparecer associada ao Problema de Alocação de Tarefas (Morabito, 1992). Por exemplo, a figura 1.2 pode ser vista como a alocação de tarefas num intervalo de tempo L quando estão disponíveis W unidades de um recurso. Cada item (tarefa) requer l_i unidades de tempo e w_i unidades de recurso. Tarefas i e j podem ser executadas ao mesmo tempo, desde que o total de recursos necessários $w_i + w_j$ não exceda W . Obviamente, outras restrições devem ser consideradas, tais como precedência entre as tarefas. Este exemplo simples mostra que restrições adicionais podem surgir, dependendo da aplicação, além das restrições geométricas.

Neste projeto enfocamos o problema unidimensional, uma variação do clássico problema da mochila, chamado de *Problema da Mochila Compartimentada*, no qual a mochila deve ser dividida em compartimentos onde são acomodados itens com características semelhantes tais como alimentos, produtos de limpeza, roupas, etc (cada compartimento pode ser visto isoladamente como um problema da mochila clássico).

Este problema modela uma aplicação importante do corte de bobinas de aço com laminação, a qual foi estudada por Ferreira et al. (1990), Carvalho (1991), Carvalho e Rodrigues (1994), que evitaram a formalização do Problema da Mochila Compartimentada e utilizaram heurísticas simples para tratá-lo. Pereira (1993) apresenta, para o problema de corte

em bobinas de aço, um modelo de otimização linear inteira que se mostrou ineficiente na produção de boas soluções. Nesse modelo, o Problema da Mochila Compartimentada também não foi formalizado. Mais tarde, tal problema foi definido, heurísticas e modelos aproximados foram desenvolvidos por Hoto (1996), Hoto e Arenales (1996) e Marques (2000). Hoto (2001) e Hoto et al. (2002) modelaram e desenvolveram um algoritmo exato e outro heurístico para o Problema da Mochila Compartimentada irrestrito, que serão discutidos mais adiante.

1.2 Organização do Texto

O texto está dividido em 7 capítulos e 2 apêndices de acordo com a seguinte estrutura:

No capítulo 1 demos uma visão geral sobre problemas de corte e empacotamento e uma classificação segundo a dimensionalidade dos problemas foi apresentada.

No capítulo 2, um resumo dos vários tipos de problemas da mochila é apresentado. Este resumo consta de uma breve descrição dos principais problemas, bem como de suas modelagens segundo os trabalhos de Martello e Toth (1990), Johnston e Khan (1995) e Lin (1998). Em tal descrição não consta o problema estudado neste projeto. Além disso, apresentamos um método de enumeração implícita, bastante explorado neste estudo, proposto por Gilmore e Gomory (1963) para resolver o problema da mochila irrestrito.

No capítulo 3 definimos o Problema da Mochila Compartimentada de uma maneira genérica. Na sequência, apresentamos uma aplicação do problema detectado na indústria metalúrgica para o caso de cortes em bobinas de aço sujeitas à laminação. Apresentamos, também, uma modelagem matemática para o Problema da Mochila Compartimentada e descrevemos um limitante para o mesmo. Por fim, alguns estudos encontrados na literatura são revisados.

No capítulo 4 apresentamos métodos de resolução. Descrevemos quatro heurísticas propostas para a resolução do Problema da Mochila Compartimentada e descrevemos os passos dos seus respectivos algoritmos.

No capítulo 5, são apresentados resultados computacionais obtidos na execução das heurísticas propostas.

No capítulo 6 fazemos algumas considerações sobre o problema de corte de estoque com mochilas compartimentadas.

O capítulo 7 dedica-se às conclusões e pesquisas futuras.

No apêndice A procuramos detalhar as modificações propostas para o algoritmo de enumeração implícita de Gilmore e Gomory (1963). Tais algoritmos modificados são essenciais na resolução do Problema da Mochila Compartimentada através das heurísticas propostas.

Por fim, no apêndice B, caracterizamos alguns aspectos computacionais explorados na implementação dos métodos propostos.

Capítulo 2. Problemas da Mochila

Neste capítulo fazemos uma revisão dos principais tipos de problemas da mochila estudados na literatura. Apesar de sua importância, o problema enfocado neste projeto, embora possa ser catalogado como um problema da mochila, não foi relacionado na literatura, como nos trabalhos de Martello e Toth (1990), Johnston e Khan (1995) e Lin (1998).

Os problemas da mochila caracterizam uma classe de problemas da programação linear inteira e são classificados na literatura segundo as suas complexidades de resolução, como problemas NP-difíceis (Garey e Johnson, 1979).

2.1 Considerações Iniciais

O problema da mochila pode ser definido do seguinte modo: suponha que um alpinista deva carregar sua mochila de itens, dentre vários disponíveis, considerando a máxima capacidade suportável, que denominamos de capacidade da mochila. A cada item é atribuído um valor de utilidade e o alpinista deve selecioná-los buscando maximizar o valor de utilidade total.

Para o problema do alpinista enunciado anteriormente, considere os seguintes dados:

- m : número de tipos de itens disponíveis;
- v_i : valor de utilidade do item tipo i , $i=1, \dots, m$;
- l_i : peso do item tipo i , $i=1, \dots, m$;
- L : capacidade da mochila.

2.1.1 Problema da Mochila Inteiro

O problema modelado a seguir é chamado na literatura de Problema da Mochila Inteiro ou simplesmente Problema da Mochila. Não há limitação nas quantidades de itens selecionados. Este problema representa o corte de uma barra, que deve ser cortada ao longo de seu comprimento, sem que uma quantidade máxima de itens seja especificada. Temos, então:

Variável de decisão:

- x_i : quantidade de itens do tipo i selecionados, $i=1, \dots, m$.

Modelagem Matemática:

$$\text{Maximizar } \Phi = \sum_{i=1}^m v_i x_i \quad (1.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i \leq L \quad (1.2)$$

$$x_i \geq 0 \text{ e inteiro, } i = 1, \dots, m. \quad (1.3)$$

No Problema da Mochila Inteiro os itens são carregados na mochila sem que sejam armazenados dentro de compartimentos. Devido às restrições impostas na construção dos compartimentos (veja capítulo 3), um carregamento qualquer pode não ser possível de acomodar em compartimentos. Sendo assim, o modelo do Problema da Mochila Inteiro é uma relaxação do Problema da Mochila Compartimentada.

2.1.2 Problema da Mochila 0-1

O Problema da Mochila 0-1 é, talvez, o mais importante problema da mochila e um dos mais estudados problemas de otimização discreta. A razão para tal interesse está, basicamente, ligada a quatro fatores:

- a) Pode ser visto como o mais simples problema de otimização linear inteira;
- b) Aparece como um subproblema em muitos outros problemas mais complexos;
- c) Pode representar uma gama muito grande de situações práticas;

d) Qualquer problema de Programação Linear Inteira pode ser transformado num Problema da Mochila 0-1.

No Problema da Mochila 0-1 temos a situação em que um único exemplar de cada item pode ser selecionado. Temos, então:

Variável de decisão:

$$x_i = \begin{cases} 1, & \text{se o item } i \text{ for selecionado;} \\ 0, & \text{caso contrário.} \end{cases} \quad i = 1, \dots, m,$$

Modelagem Matemática:

$$\text{Maximizar } \Phi = \sum_{i=1}^m v_i x_i \quad (2.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i \leq L \quad (2.2)$$

$$x_i = 0 \text{ ou } 1, i = 1, \dots, m. \quad (2.3)$$

Durante as últimas décadas, o Problema da Mochila 0-1 tem sido estudado por diferentes abordagens, tais como a programação dinâmica e a enumeração implícita.

2.1.3 Problema da Mochila Restrito

Alguns problemas podem apresentar condições adicionais, como por exemplo, a limitação da quantidade de itens de um determinado tipo a serem selecionados. Neste caso, o problema passa a ser chamado de Problema da Mochila Restrito. Para o Problema da Mochila Restrito, além dos dados necessários para os modelos anteriores, considere ainda:

- d_i : quantidade máxima de itens tipo i que pode ser selecionada, $i=1, \dots, m$;

Modelagem Matemática:

$$\text{Maximizar } \Phi = \sum_{i=1}^m v_i x_i \quad (3.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i \leq L \quad (3.2)$$

$$0 \leq x_i \leq d_i \text{ e inteiro, } i = 1, \dots, m. \quad (3.3)$$

Tal problema pode ser visto como uma generalização do Problema da Mochila 0-1, onde $d_i = 1$ para $i = 1, \dots, m$. Uma outra restrição que poderia ser inserida ao problema diz respeito ao limite no número total de itens selecionados a serem carregados na mochila. Sendo assim, considerando F o número máximo de itens na mochila, temos a seguinte restrição adicional ao problema (3.1–3.3):

$$\sum_{i=1}^m x_i \leq F \quad (3.4)$$

As restrições (3.4) estão relacionadas, por exemplo, ao número de facas presentes na máquina que efetua o corte. A figura 2.1 mostra o corte de uma bobina sujeito a uma limitação de 7 facas.

Figura 2.1: Corte de bobina com restrição no número de facas.

2.1.4 Problema da Mochila de Escolha Múltipla

O Problema da Mochila de Escolha Múltipla é um Problema da Mochila 0-1 em que os m tipos de itens estão particionados em K classes A_k , $k = 1, \dots, K$. Neste caso é exigido que exatamente um tipo de item de cada classe seja selecionado para ser carregado na mochila.

Modelagem Matemática:

$$\text{Maximizar } \Phi = \sum_{i=1}^m v_i x_i \quad (4.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i \leq L \quad (4.2)$$

$$\sum_{i \in A_k} x_i = 1, \quad k = 1, \dots, K \quad (4.3)$$

$$x_i = 0 \text{ ou } 1, \quad i \in M = \{1, \dots, m\} = \bigcup_{k=1}^K A_k. \quad (4.4)$$

Para o Problema da Mochila de Escolha Múltipla, assumimos que $A_h \cap A_k = \emptyset$ para todo $h \neq k$. Apesar dos itens pertencerem a classes (por exemplo, alimentos, vestuário, etc.) este problema difere do Problema da Mochila Compartimentada, dentre outras coisas, devido à exigência de que apenas um item de cada classe seja selecionado para preencher a mochila. Além disso, a ausência de restrições de capacidade imposta na construção dos compartimentos (veja capítulo 3) é característica fundamental na diferenciação dos problemas.

2.1.5 Problema da Soma de Subconjuntos

O Problema da Soma de Subconjuntos consiste em selecionar um subconjunto de itens cuja soma total dos pesos dos itens escolhidos se aproxime ao máximo de L , sem excedê-lo.

Modelagem Matemática:

$$\text{Maximizar } \Phi = \sum_{i=1}^m l_i x_i \quad (5.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i \leq L \quad (5.2)$$

$$x_i = 0 \text{ ou } 1, i = 1, \dots, m. \quad (5.3)$$

O Problema da Soma de Subconjuntos é um caso particular do Problema da Mochila 0-1, onde $v_i = l_i$ para $i = 1, \dots, m$. Tal problema é conhecido, também, como Problema da Mochila de Valor Independente.

2.1.6 Problema do Troco

O Problema do Troco consiste em selecionar um número x_i ($i = 1, \dots, m$) de itens do tipo i (por exemplo, moeda de valor l_i) tal que o peso total seja L e o número total de itens selecionados seja mínimo.

Modelagem Matemática:

$$\text{Minimizar } \Phi = \sum_{i=1}^m x_i \quad (6.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i = L \quad (6.2)$$

$$x_i \geq 0 \text{ e inteiro, } i = 1, \dots, m. \quad (6.3)$$

O problema é chamado Problema do Troco, pois pode ser interpretado como o problema de um caixa (supermercado, banco, etc) que deve devolver uma determinada quantia, L , usando para isto, um número mínimo de moedas (ou notas) de valores específicos l_i ($i = 1, \dots, m$). Nesse caso, para cada valor, um número ilimitado de moedas está disponível. A quantidade de moedas (ou notas) de um determinado valor disponíveis pode ser limitada. Neste caso a restrição (6.3) é representada por: $0 \leq x_i \leq d_i$ e inteiro, $i = 1, \dots, m$.

A condição de igualdade imposta em (6.2) pode fazer com que inexista uma solução para o problema.

2.1.7 Problema da Mochila Quadrático

O Problema da Mochila Quadrático é um dos mais estudados entre os problemas da mochila não lineares. Um problema da mochila não linear é aquele com função objetivo não linear e/ou que envolve restrições não lineares. Um Problema da Mochila Quadrático pode ser formulado matematicamente da seguinte forma:

Modelagem Matemática:

$$\text{Maximizar } \Phi = x^T Q x + c^T x \quad (7.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i \leq L \quad (7.2)$$

$$x_i \geq 0 \text{ e inteiros, } i = 1, \dots, m. \quad (7.3)$$

com $x^T = (x_1, x_2, \dots, x_m)$, $c^T = (c_1, c_2, \dots, c_m)$ e Q uma matriz $m \times m$, simétrica e não nula.

Convém observar que o Problema da Mochila Compartimentada pode ser modelado como um problema quadrático, onde a função objetivo e algumas restrições são quadráticas (seção 3.2). Até aqui, os problemas da mochila consistiram em carregar apenas uma mochila.

Os próximos problemas, também classificados por Martello e Toth (1990), Johnston e Khan (1995) e Lin (1998) como problemas da mochila, consistem em carregar várias mochilas.

2.1.8 Problema de Múltiplas Mochilas 0-1

O Problema de Múltiplas Mochilas 0-1 consiste em carregar um conjunto de n mochilas ($n \leq m$) cujas capacidades são dadas por: $L_j, j = 1, \dots, n$. Os demais dados do problema são os mesmos do problema para uma única mochila.

O problema consiste em seleccionar subconjuntos disjuntos de itens e associá-los a diferentes mochilas, cujas capacidades não sejam excedidas, tal que o valor de utilidade dos itens seleccionados seja máximo.

Variável de decisão:

$$x_{ij} = \begin{cases} 1, & \text{se o item } i \text{ está associado à mochila } j; \\ 0, & \text{caso contrário.} \end{cases}$$

Modelagem Matemática:

$$\text{Maximizar } \Phi = \sum_{j=1}^n \sum_{i=1}^m v_i x_{ij} \quad (8.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_{ij} \leq L_j, \quad j = 1, \dots, n \quad (8.2)$$

$$\sum_{j=1}^n x_{ij} \leq 1, \quad i = 1, \dots, m \quad (8.3)$$

$$x_{ij} = 0 \text{ ou } 1, \quad i = 1, \dots, m \text{ e } j = 1, \dots, n. \quad (8.4)$$

Quando $n = 1$, o Problema de Múltiplas Mochilas 0-1 se reduz a um Problema da Mochila 0-1 simples, pois a restrição (8.3) torna-se redundante.

Note que este problema preserva a característica dos problemas anteriores (que apenas uma mochila deveria ser preenchida) de que uma seleção de itens deve ser feita, ou seja, as mochilas são insuficientes para carregar todos os itens e a seleção é tal que o valor de utilidade seja maximizado (exceto o problema do troco, mas que também é de seleção de itens). Nos próximos dois problemas, todos os itens serão alocados, de modo que há uma seleção de mochilas.

(Dyckhoff e Fink, 1992) classificam os problemas de corte e empacotamento (problemas da mochila são casos particulares) destacando esta característica de seleção de itens ou seleção de mochilas, além da dimensão do problema, como descrito na introdução deste trabalho.

Note ainda que, embora cada mochila possa ser vista como um compartimento de capacidade L_j a ser alocado numa mochila maior, este problema não modela o caso da mochila compartimentada, pois além das capacidades ocupadas pelos compartimentos serem incógnitas, somente itens de mesma característica podem ser alocados ao mesmo compartimento, o que não é considerado em (8.1-8.4).

2.1.9 Problema de Designação Generalizada

O Problema de Designação Generalizada pode ser descrito usando a terminologia aplicada para os problemas da mochila e é semelhante ao Problema de Múltiplas Mochilas, porém o valor de utilidade e o peso do item i podem variar se associados a mochilas diferentes. Considere os seguintes dados:

- v_{ij} : ganho fornecido pela associação do item i à mochila j , $i=1,...,m$ e $j=1,...,n$;
- l_{ij} : peso do item i se associado à mochila j , $i=1,...,m$ e $j=1,...,n$.

Modelagem Matemática:

$$\text{Maximizar } \Phi = \sum_{j=1}^n \sum_{i=1}^m v_{ij} x_{ij} \quad (9.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_{ij} x_{ij} \leq L_j, \quad j = 1, \dots, n \quad (9.2)$$

$$\sum_{j=1}^n x_{ij} = 1, \quad i = 1, \dots, m \quad (9.3)$$

$$x_{ij} = 0 \text{ ou } 1, \quad i = 1, \dots, m \text{ e } j = 1, \dots, n. \quad (9.4)$$

O problema é, freqüentemente, descrito na literatura como o problema de designação ótima de m tarefas a n processadores, dados o ganho v_{ij} e a quantidade de recursos l_{ij} correspondente à associação da tarefa i ao processador j e a quantidade total de recursos L_j suportados por cada processador j .

Note que este problema difere dos anteriores em dois aspectos: (i) todos os itens devem ser alocados (restrições (9.3)) e, (ii) os itens sofrem “deformações” se alocados em mochilas distintas.

2.1.10 Problema de Múltiplas Mochilas Idênticas (*Bin-Packing*)

O Problema de Múltiplas Mochilas Idênticas, mais conhecido na literatura por problema *Bin-Packing* considera:

- l_i : peso do item i , $i=1, \dots, m$;
- L : capacidade de cada caixa.

Todo item deve ser associado a uma única caixa, tal que o peso total dos itens em cada caixa não exceda L e o número de caixas usadas seja mínimo.

Variáveis de decisão:

$$y_j = \begin{cases} 1, & \text{se a caixa } j \text{ é usada;} \\ 0, & \text{caso contrário.} \end{cases}$$

$$x_{ij} = \begin{cases} 1, & \text{se o item } i \text{ está associado à caixa } j; \\ 0, & \text{caso contrário.} \end{cases}$$

Modelagem Matemática:

$$\text{Minimizar } \Phi = \sum_{j=1}^n y_j \quad (10.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_{ij} \leq L \cdot y_j, \quad j = 1, \dots, n \quad (10.2)$$

$$\sum_{j=1}^n x_{ij} = 1, \quad i = 1, \dots, m \quad (10.3)$$

$$y_j = 0 \text{ ou } 1, \quad x_{ij} = 0 \text{ ou } 1, \quad i = 1, \dots, m \text{ e } j = 1, \dots, n. \quad (10.4)$$

2.1.11 Problema da Mochila Subdividida (*Nested Knapsack Problem*)

Dentre os problemas estudados na literatura, o que mais se assemelha ao Problema da Mochila Compartimentada, objeto de estudo desta tese, é o Problema da Mochila Subdividida (Johnston e Khan, 1995), descrito a seguir.

O problema consiste em construir subdivisões de tamanhos pré-definidos que deverão abrigar itens de interesse. No interior de cada subdivisão podem ser construídas novas subdivisões, também de tamanhos pré-definidos, e assim sucessivamente. Esta modalidade de mochila, embora muito semelhante à mochila compartimentada, onde são construídos compartimentos, difere substancialmente pelo fato das subdivisões terem seus tamanhos fixos, ao passo que os compartimentos apresentam tamanhos variáveis. Outra diferença está relacionada ao fato de que os compartimentos devem ser preenchidos apenas com itens que possuam características semelhantes (por exemplo, roupas, comidas, remédios, etc.) enquanto as subdivisões podem ser preenchidas com itens diversos.

O exemplo a seguir ilustra uma mochila subdividida: itens devem ser selecionados para serem levados até uma região montanhosa de difícil acesso. No início eles são transportados no vagão de um trem até um ponto em que seja possível apenas o transporte rodoviário. Daí em diante, os itens são transportados por meio de pequenos furgões até um outro ponto em que o transporte só pode ser feito por carregadores. A carga de cada carregador deve ser proveniente de um (e apenas um) dos furgões. Em cada sistema de transporte (via trem, furgão e carregador) existe uma capacidade de carga e custo de transporte associado. O problema consiste em selecionar itens para as respectivas mochilas em cada estágio, de modo que seja maximizada a utilidade dos itens menos os custos de transporte.

Essencialmente, o problema descrito representa uma situação em que os itens selecionados no primeiro estágio (transporte via trem) devem ser novamente selecionados nos estágios subseqüentes (transporte via furgões e carregadores) e, ainda mais, todo item selecionado para a mochila do primeiro estágio deve ser incluído em uma (e somente uma) mochila de cada um dos estágios subseqüentes.

Johnston e Khan (1995) apresentam um modelo para o Problema da Mochila Subdividida, cujas variáveis de decisão são:

$$x_{ij}^s = \begin{cases} 1, & \text{se o item } i \text{ foi designado para a mochila } j \text{ no estágio } s; \\ 0, & \text{caso contrário.} \end{cases}$$

$$y_j^s = \begin{cases} 1, & \text{se a mochila } j \text{ é utilizada no estágio } s; \\ 0, & \text{caso contrário,} \end{cases}$$

em que $i, j = 1, \dots, m$ e $s = 1, \dots, S$, S é o total de estágios. Também são considerados os parâmetros c_s e L_s que representam, respectivamente, o custo e a capacidade das mochilas do estágio $s = 1, \dots, S$.

O Problema da Mochila Subdividida pode ser modelado por:

$$\text{Maximizar } \Phi = \sum_{i=1}^m v_i x_{i1}^1 - \sum_{j=1}^m \sum_{s=2}^S c_s y_j^s \quad (11.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_{i1}^1 \leq L_1 \quad (11.2)$$

$$\sum_{i=1}^m l_i x_{ij}^s \leq L_s y_j^s, \quad j = 1, \dots, m, \quad s = 2, \dots, S \quad (11.3)$$

$$x_{i1}^1 \leq \sum_{j=1}^m x_{ij}^s, \quad i = 1, \dots, m, \quad s = 1, \dots, S \quad (11.4)$$

$$\max_j (x_{pj}^s + x_{qj}^s) \leq \max_j (x_{pj}^{s-1} + x_{qj}^{s-1}), \quad (11.5)$$

$$p = 1, \dots, m, \quad q = 1, \dots, m, \quad s = 3, \dots, S, \quad p \neq q.$$

$$x_{ij}^s = 0 \text{ ou } 1, \quad y_j^s = 0 \text{ ou } 1, \quad i = 1, \dots, m, \quad j = 1, \dots, m \text{ e } s = 1, \dots, S. \quad (11.6)$$

2.2 Um Método de Resolução para o Problema da Mochila

Nesta seção apresentamos um método de enumeração implícita proposto em (Gilmore e Gomory, 1963) para a resolução do seguinte Problema da Mochila Inteiro:

$$\text{Maximizar } \Phi = \sum_{i=1}^m v_i x_i \quad (12.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i \leq L \quad (12.2)$$

$$x_i \geq 0 \text{ e inteiro, } i = 1, \dots, m, \quad (12.3)$$

Onde x_i é a quantidade de itens do tipo i selecionados.

2.2.1 Considerações Iniciais

O problema (12.1-12.3) como já mencionado, é conhecido na literatura como Problema da Mochila e pode ser resolvido por uma seqüência de decisões, que consiste em incluir um item por vez, dentre os m tipos de itens disponíveis. Como a mochila inicialmente está vazia, a capacidade disponível inicial é L . Ao tomarmos a decisão de colocar o item do tipo i com peso l_i , $i = 1, 2, \dots, m$, restará uma capacidade de $L - l_i$ a ser preenchida. Em seguida, esta capacidade resultante poderá ser preenchida, escolhendo-se um item do tipo j , restando, assim, $L - l_i - l_j$. Este processo poderá ser repetido até que a capacidade resultante não seja mais suficiente para a inclusão de qualquer outro item. Quando todas as possibilidades de inclusão forem investigadas, teremos percorrido todas as possíveis soluções do Problema da Mochila. Essas possíveis decisões podem ser representadas numa árvore, denominada árvore de decisões, onde cada nó representa a capacidade restante a ser preenchida. Ao nó inicial dá-se o nome de nó raiz, e a capacidade neste nó é a própria capacidade da mochila. Aos nós finais, para os quais a capacidade restante da mochila não permite a inclusão de nenhum outro item, dá-se o nome de nó folha. Além dos nós, temos ainda os ramos da árvore que representam as decisões tomadas (por exemplo, colocar um item do tipo i na mochila). A uma seqüência de nós partindo do nó raiz até o nó folha de uma árvore dá-se o nome de caminho completo. Uma árvore de decisões é mostrada na figura 2.2.

Para exemplificar o processo de tomada de decisões e geração da árvore de decisões para o problema da mochila, considere os seguintes dados:

Dados do problema:

- $m = 2$;
- $v_1 = 3$ e $v_2 = 4$;

- $l_1 = 3$ e $l_2 = 5$;
- $L = 11$.

Variáveis de decisão:

- x_i : quantidade de itens do tipo i , $i = 1$ e 2 , incluída na mochila.

Para os dados apresentados, temos o seguinte modelo para o Problema da Mochila (12.1-12.3):

$$\text{Maximizar } \phi = 3x_1 + 4x_2$$

$$\text{Sujeito a: } 3x_1 + 5x_2 \leq 11$$

$$x_1 \geq 0, x_2 \geq 0 \text{ e inteiros.}$$

Temos, então, a seguinte árvore de decisões:

Figura 2.2: Árvore de decisões.

Como foi dito anteriormente, uma árvore de decisões representa toda e qualquer solução para o problema da mochila. Esta representação é também conhecida por enumeração completa. Como podemos notar na figura 2.2, na enumeração completa, várias soluções iguais, ou redundantes, podem ser encontradas.

Sabe-se que percorrer todas as possíveis soluções do problema da mochila, isto é, enumerar explicitamente todos os caminhos na árvore de decisões, é, em geral, computacionalmente impraticável para problemas de tamanhos moderados. As soluções podem, entretanto, ser implicitamente enumeradas, isto é, podemos descartar soluções sem

perder a solução ótima do problema e, ao mesmo tempo, evitar as soluções redundantes. Isso pode ser feito com o uso de limitantes para o problema e regras de ordenação na busca. A seguir, apresentamos o método de enumeração implícita proposto por Gilmore e Gomory (1963) e que foi bastante modificado nesta tese para resolver problemas que surgem nas heurísticas propostas para o Problema da Mochila Compartimentada (capítulo 4).

2.2.2 O Método Enumeração Implícita de Gilmore e Gomory (1963)

Nesta seção, apresentamos um método eficiente para o problema da mochila inteiro proposto em (Gilmore e Gomory, 1963), que consiste num método de enumeração implícita (*branch and bound*) explorando busca em profundidade numa árvore de decisões.

Passo 1: Pré-processamento

Para cada item do tipo i (com valor v_i e peso l_i associados) definimos: $\pi_i = v_i / l_i$, $i=1, \dots, m$, que indica o valor do item i por unidade de peso, e ordenamos numa sequência decrescente. Por simplicidade de notação, suponha que:

$$\pi_1 \geq \pi_2 \geq \dots \geq \pi_m,$$

isto é o item 1 é o mais valioso, o item 2 o segundo mais valioso, etc.

Passo 2: Obtenção de uma Solução Factível Inicial

Percorrendo a sequência de itens de 1 a m (isto é, partindo do mais valioso), busca-se, agora, obter uma solução factível. Um vetor $\mathbf{X} = (x_1, x_2, \dots, x_m)$ fornece uma solução em que x_i é o número de itens do tipo i . Este vetor define um caminho na árvore de decisões. Por exemplo, na figura 2.2, $\mathbf{X} = (3 \ 0)$ fornece o caminho obtido pela sequência de incluir 3 vezes o item 1. Com isto, $\mathbf{X}$ representa cada nó da árvore de decisões. Além disso, pode ser fácil retornar ao nó precedente na árvore. Por exemplo, $\mathbf{X} = (2 \ 0)$ é o precedente de $(3 \ 0)$, basta subtrair 1 da última decisão, isto é, elemento não nulo de maior índice. Observe que eventuais duplicações na árvore de decisões são representadas pelo mesmo vetor. Por exemplo, a

seqüência de decisões: item 1 – item 2 – item 1 é representada por $\mathbf{X} = (2 \ 1)$, que consiste na seqüência equivalente: item 1 – item 1 – item 2. Para evitar duplicações na busca (isto é, obter a mesma solução $\mathbf{X}$), basta proibir que após a inclusão de um item t sejam incluídos itens 1, 2, ..., $t-1$. Sendo assim, se a seqüência item 1 – item 2, representada por $\mathbf{X} = (1 \ 1)$, não tem sucessor, significa que não é possível acrescentar outro item 2. A inclusão de outro item 1 só seria obtida por outra seqüência de decisões.

A solução homogênea é uma solução factível que pode ser facilmente obtida colocando-se a quantidade máxima (permitida pela capacidade da mochila) do item mais valioso, dada por:

$$x_1 = \left\lfloor \frac{L}{l_1} \right\rfloor, \quad x_2 = 0, \dots, \quad x_m = 0.$$

Ou seja, apenas o item de tipo 1 será incluído na mochila (neste caso, o mais valioso).

Uma vez obtida a solução homogênea, uma sobra na capacidade pode existir, de modo que outros itens podem ser incluídos, obtendo-se uma solução factível melhorada. Seguindo a ordem dos itens mais valiosos, fazemos então:

$$\begin{aligned} x_1 &= \left\lfloor \frac{L}{l_1} \right\rfloor \Rightarrow \text{sobra: } L - l_1 x_1, \\ x_2 &= \left\lfloor \frac{L - l_1 x_1}{l_2} \right\rfloor \Rightarrow \text{sobra: } L - l_1 x_1 - l_2 x_2, \\ \dots \quad x_{t+1} &= \left\lfloor \frac{L - \sum_{i=1}^t l_i x_i}{l_{t+1}} \right\rfloor, \quad t = 1, \dots, m-1. \end{aligned}$$

Esta solução melhorada pode ser vista como uma “descida” (busca em profundidade) na árvore de decisões e a inclusão de mais um item na mochila faria com que a sua capacidade fosse ultrapassada, tornando a solução infactível. Por exemplo, na figura 2.2, a solução inicial é $\mathbf{X} = (3 \ 0)$. A figura 2.3 ilustra o processo de “descida” na árvore de decisões.

Figura 2.3: Descida na árvore de decisões para a obtenção de uma solução factível melhorada.

Passo 3: Retorno Curto

A solução melhorada é uma solução factível muito boa, em geral, porém, pode não ser uma solução ótima. As demais soluções podem ser recuperadas e comparadas com a melhor solução disponível (que inicialmente será a solução melhorada). Usando um processo de *backtracking* (caminho de volta), que recupera o nó predecessor, podemos recuperar os demais caminhos. Isto pode ser facilmente implementado notando-se que para um nó na árvore de decisões, uma sequência de decisões foi tomada, isto é, temos associado uma solução factível:

$$X = (x_1, x_2, \dots, x_t, 0, \dots, 0),$$

supondo $x_t \neq 0$. Isto significa que a última decisão tomada foi a inclusão de um item de tipo t (veja a figura 2.3). Assim, o nó predecessor está associado à solução:

$$\mathbf{X} = (x_1, x_2, \dots, x_t - 1, 0, \dots, 0)$$

A figura 2.4 ilustra a operação de retorno curto.

Figura 2.4: Operação de retorno curto.

Sendo assim, podemos enumerar os nós da árvore de decisões com auxílio de um único vetor que mostra as decisões tomadas até um nó qualquer.

Porém, como percorrermos soluções alternativas, as quais estarão no vetor $\mathbf{X}$, guardamos num outro vetor a melhor solução encontrada: $\mathbf{X}^* = (x_1, x_2, \dots, x_t, 0, \dots, 0)$ e $\underline{G} = v_1 x_1 + \dots + v_t x_t$ é um limitante inferior, uma vez que $\mathbf{X}^*$ é apenas uma solução factível.

Passo 4: Cálculo do Limitante Superior

O retorno ao nó precedente “libera” uma certa capacidade da mochila que estava sendo ocupada pelo item retirado. Esta capacidade pode ser ocupada por outros itens, gerando novos caminhos na árvore de decisões (novas soluções factíveis para o problema da mochila).

Ao retornarmos ao nó precedente, com o peso de l_t sendo liberado, temos a mochila com capacidade ociosa:

$$Sobra = L - x_1 l_1 - x_2 l_2 - \dots - (x_t - 1) l_t$$

Com solução associada ao nó precedente sendo:

$$\mathbf{X} = (x_1, x_2, \dots, x_t - 1, 0, \dots, 0)$$

Como a inclusão dos itens de ordem $1, 2, \dots, t-1$ não é possível, uma vez que já foram incluídos ao máximo e a inclusão do item de ordem t retornaria à mesma solução já pesquisada, resta-nos, para investigar novas soluções, considerar os itens de ordem $t+1, t+2, \dots, m$, para preencher a capacidade ociosa na mochila.

Agora o item de ordem $t+1$ é o mais valioso. Consideramos, então, o limitante superior para o nó precedente, incluindo as decisões já tomadas $(x_1, x_2, \dots, x_t-1, 0, \dots, 0)$:

$$\overline{G}(\mathbf{X}) = \underbrace{v_1 x_1 + \dots + v_t (x_t - 1)}_{\text{Valor dos itens já incluídos.}} + \underbrace{\frac{v_{t+1}}{l_{t+1}} (L - l_1 x_1 - \dots - l_t (x_t - 1))}_{\text{Limitante superior pela inclusão de itens } t+1, t+2, \dots, m.}$$

Portanto, se $\overline{G} \leq \underline{G}$ então não há soluções nos caminhos decorrentes das inclusões de itens do tipo $t+1, t+2, \dots, m$, que superem aquela solução já conhecida, cujo valor é $\underline{G}$.

Entretanto, se $\overline{G} > \underline{G}$, existe ainda a possibilidade de que a melhor solução disponível venha a ser superada por algum caminho, a partir do nó precedente, decorrente da decisão de incluir itens $t+1, \dots, m$. Preenchemos a capacidade ociosa na mochila, relativa ao nó precedente, com os itens de ordem $t+1, t+2, \dots, m$ analogamente à solução do passo 2.

Passo 5: Testes

Se $\overline{G} \leq \underline{G}$ (isto é, não compensa a liberação de um item de ordem t para a inclusão de outros do tipo $t+1, t+2, \dots, m$), então não compensa também a liberação de dois ou mais itens do tipo t para a inclusão de outros de ordem $t+1, \dots, m$. Esta afirmação pode ser mostrada facilmente.

A conclusão é que, neste caso, podemos retornar ao nó cuja solução é $\mathbf{X} = (x_1, x_2, \dots, x_t-1, 0, \dots, 0)$, pois os caminhos que emanam dos nós cujas soluções:

$$\begin{aligned} &(x_1, x_2, \dots, x_t - 1, 0, \dots, 0), \\ &(x_1, x_2, \dots, x_t - 2, 0, \dots, 0), \dots \\ &(x_1, x_2, \dots, 1, 0, \dots, 0), \end{aligned}$$

não produzem soluções melhores àquela já conhecida. A figura 2.5 ilustra como o processo de *backtracking* pode saltar vários nós intermediários.

Figura 2.5: *Backtracking* saltando vários nós intermediários (retorno longo).

Dessa forma, retorna-se ao nó $X = (x_1, x_2, \dots, x_{t-1}, 0, \dots, 0)$, faz-se $t = t - 1$ e volta-se ao passo 4 até que o *backtracking* leve de volta ao nó raiz. Quando isto ocorrer, a solução armazenada no vetor X^* é a solução ótima e o processo termina.

Com o desenvolvimento da pesquisa envolvendo o Problema da Mochila Compartmentada muitas modificações no método de enumeração implícita de Gilmore e Gomory foram necessárias. Tais modificações são contribuições secundárias desta tese. No apêndice A daremos mais detalhes destas modificações e em que contexto elas foram utilizadas.

Neste capítulo, revisamos os vários problemas da mochila catalogados na literatura segundo Martello e Toth (1990), Johnston e Khan (1995) e Lin (1998) e um conhecido método para a resolução do clássico problema da mochila proposto em (Gilmore e Gomory, 1963). Como veremos nos próximos capítulos, estes problemas diferem do Problema da Mochila Compartmentada.

Capítulo 3. O Problema da Mochila Compartimentada

O problema principal desta tese, o Problema da Mochila Compartimentada, tem sido pouco estudado na literatura (Hoto, 1996), (Marques, 2000), (Hoto, 2001), (Marques e Arenales, 2002) e (Hoto et al., 2003), sendo implicitamente tratado em (Ferreira et al., 1990), (Carvalho, 1991) e (Carvalho e Rodrigues, 1994). Neste capítulo definimos o Problema da Mochila Compartimentada, mostramos uma aplicação prática em problemas de corte de bobinas de aço sujeitas a um processo de laminação e apresentamos um modelo matemático para o mesmo. Além disso, damos um modelo agregado que fornece um limitante superior para o Problema da Mochila Compartimentada e fazemos uma revisão de alguns estudos relacionados ao problema.

3.1 Definição do Problema

Um alpinista deve carregar sua mochila com m possíveis itens de sua utilidade. A cada item i , o alpinista atribui um valor de utilidade v_i e seu peso l_i . O máximo peso que o alpinista suporta em sua jornada é de L . Até este ponto, a descrição do problema coincide com a formulação clássica do problema da mochila. Porém, os itens são de agrupamentos distintos (por exemplo: agrupamento 1: alimentos, agrupamento 2: utensílios, agrupamento 3: roupas, agrupamento 4: calçados, etc.) e devem estar em compartimentos separados dentro da mochila (isto é, cada compartimento contém somente itens de um mesmo agrupamento). Além disso, o número de itens de um determinado tipo i pode ser limitado, digamos, por d_i . As capacidades dos compartimentos da mochila são flexíveis, permitindo que o alpinista carregue maior peso em alimentos do que em roupas, por exemplo. Entretanto, essas capacidades (incógnitas) são limitadas inferior e superiormente (de acordo com os agrupamentos), caso estes sejam

criados, digamos por: $L_{\min}^k$ e $L_{\max}^k$ (isto é, a soma dos pesos l_i dos itens, de um agrupamento k , alocados no compartimento deve ser superior ou igual a $L_{\min}^k$ e inferior ou igual a $L_{\max}^k$). A cada compartimento é associado um custo c_k , caso o compartimento seja preenchido com itens do agrupamento k e, além disso, cada compartimento incluído na mochila provocará uma perda de capacidade, digamos, S_k (isto é, se, por exemplo, três compartimentos com itens de um agrupamento k forem alocados na mochila, então a capacidade total disponível na mochila será de $L-3S_k$, ao invés de L). É importante ressaltar que vários compartimentos, de capacidades diferentes, podem ser criados para carregar itens de um mesmo agrupamento ou, em outras palavras, itens de um mesmo agrupamento podem ser alocados em mais do que um compartimento.

O Problema da Mochila Compartimentada consiste em determinar as capacidades adequadas de cada compartimento e como estes devem ser carregados, de modo que o valor de utilidade total (soma dos valores de utilidade de todos os itens selecionados) seja máximo. A este valor, desconta-se os custos dos compartimentos, os quais dependem dos agrupamentos com que foram preenchidos (c_k). Na seção seguinte, apresentamos um exemplo numérico do Problema da Mochila Compartimentada para ilustrar a definição aqui apresentada.

Como poderemos constatar na sequência deste capítulo, tal problema surge, por exemplo, no corte de bobinas de aço, onde os itens são fitas para a produção de tubos soldados, os compartimentos são as bobinas intermediárias que serão laminadas até as espessuras das fitas e a mochila a ser preenchida é uma bobina mestre a ser cortada.

3.2 Um Exemplo Numérico para o Problema da Mochila Compartimentada

Para o Problema da Mochila Compartimentada, temos que a cada compartimento da mochila tem-se associado um único agrupamento k de itens, ou seja, não se admite itens de agrupamentos distintos no mesmo compartimento. Para ilustrar, usamos um exemplo numérico. Os dados necessários são:

Dados do problema:

- Dados da Mochila:
 - L : capacidade (Kg) da mochila.
- Dados dos Itens:

- $M = \{1, \dots, m\}$: conjunto dos tipos de itens;
- l_i : peso (Kg) do item i ($l_i > 0$), $i=1, \dots, m$;
- v_i : valor de utilidade do item i ($v_i \geq 0$), $i=1, \dots, m$;
- d_i : limite máximo de itens i na mochila, $i=1, \dots, m$.
- Dados dos Agrupamentos:
 - K : número total de agrupamentos distintos;
 - A_k : agrupamento k , $k=1, \dots, K$, ($M = A_1 \cup A_2 \dots \cup A_K$, $A_i \cap A_j = \emptyset$, com $i \neq j$);
 - c_k : custo de inclusão de um compartimento de itens do agrupamento A_k , onde $c_k \geq 0$, $k=1, \dots, K$;
 - L_{min}^k : capacidade mínima (Kg) de cada compartimento associado ao agrupamento A_k ;
 - L_{max}^k : capacidade máxima (Kg) de cada compartimento associado ao agrupamento A_k ($L_{min}^k \leq L_{max}^k \leq L$);
 - S_k : perda (Kg) decorrente da inclusão de um novo compartimento preenchido com itens do agrupamento A_k na mochila;
 - N_k : número total de possíveis compartimentos para o agrupamento A_k (por exemplo: $A_1 = \{1, 2\}$, então $L_{11} = l_1 + l_2 + S_1$, $L_{21} = 2l_1 + l_2 + S_1$, etc., são duas possíveis capacidades de compartimentos para o agrupamento 1. L_{jk} referencia o j -ésimo compartimento associado ao agrupamento A_k . O número N_k depende essencialmente dos itens do agrupamento A_k e dos limitantes L_{min}^k e L_{max}^k).

Suponha um objeto (mochila) com capacidade $L = 30$, como ilustrado no objeto da figura 3.1:

Figura 3.1: Representação da capacidade do objeto.

Além disso, considere 3 agrupamentos ($K = 3$), em que o último representa um agrupamento especial de itens livres (o terceiro agrupamento) que possui características particulares, tais como $L_{min}^K = 0$, $L_{max}^K = L$ e $c_K = 0$. Isto significa que os itens do terceiro

agrupamento (itens livres) não precisam ser compartimentados. As tabelas 3.1 a 3.3 fornecem os agrupamentos, bem como os limites de capacidade e os custos dos compartimentos:

Agrupamento 1: $L_{min}^1 = 7, L_{max}^1 = 9, S_1 = 1$ e $c_1 = 0,5$			
Itens	Peso (l_i)	Utilidade (v_i)	Limites (d_i)
1	3	2	2
2	4	3	3
3	6	5	3

Tabela 3.1: Dados do agrupamento 1.

Agrupamento 2: $L_{min}^2 = 7, L_{max}^2 = 9, S_2 = 1$ e $c_2 = 0,8$			
Itens	Peso (l_i)	Utilidade (v_i)	Limites (d_i)
4	2	1	3
5	4	2	2

Tabela 3.2: Dados do agrupamento 2.

Agrupamento 3: $L_{min}^3 = 0, L_{max}^3 = 30, S_3 = 0$ e $c_3 = 0$			
Itens	Peso (l_i)	Utilidade (v_i)	Limites (d_i)
6	5	2	1
7	3	1	2

Tabela 3.3: Dados do agrupamento 3 (itens livres).

Figura 3.2: Representação dos itens.

Considerando estes dados, temos: $m = 7$, $M = \{1,2,3,4,5,6,7\}$, $A_1 = \{1,2,3\}$, $A_2 = \{4,5\}$ e $A_3 = \{6,7\}$. Além disso, os possíveis compartimentos associados aos agrupamentos $k = 1, 2$ e 3 obedecem aos intervalos impostos por L_{min}^k e L_{max}^k , têm perdas laterais definidas por S_k e possuem custos c_k como mostrados, respectivamente, nas tabelas 3.1, 3.2 e 3.3. Levando em consideração estes dados, teríamos os seguintes compartimentos possíveis para os agrupamentos (a perda S_k é incluída):

Agrupamento 1: $N_1 = 4$	Agrupamento 2: $N_2 = 4$	Agrupamento 3: $N_3 = 5$
$L_{11} = 7$	$L_{12} = 7$	$L_{13} = 5$
$L_{21} = 8$	$L_{22} = 9$	$L_{23} = 8$
$L_{31} = 9$	$L_{32} = 7$	$L_{33} = 11$
$L_{41} = 7$	$L_{42} = 9$	$L_{43} = 3$
		$L_{53} = 6$

Figura 3.3: Possíveis compartimentos para os agrupamentos do exemplo numérico.

Note que para o agrupamento 1, por exemplo, não podemos carregar a mesma mochila com os compartimentos de pesos L_{11} e L_{21} pois teríamos, neste caso, 3 unidades do item tipo 1 na mochila, enquanto o limite máximo (d_i) é de 2 unidades. Tal característica introduz uma dificuldade a mais na resolução do problema, pois não basta determinar todos os possíveis compartimentos, identificá-los como “superitens” e aloca-los na mochila. Deve-se levar em conta os itens dentro de cada compartimento.

Considerando, então, os compartimentos exibidos na figura 3.3 e respeitando as restrições impostas na definição do problema, apresentamos, na figura 3.4, um possível padrão de corte compartimentado.

Figura 3.4: Padrão de corte compartimentado.

A seguir, descrevemos uma aplicação prática para o Problema da Mochila Compartimentada que serviu como principal motivação para esta tese. O problema surge nas linhas de produção de indústrias metalúrgicas que utilizam o corte de bobinas de aço para a produção de tubos soldados. Um importante processo técnico de laminação do aço pode ser necessário para a produção dos tubos. Esta característica formaliza a necessidade de se efetuar o corte da bobina em duas fases, de modo que as bobinas obtidas do primeiro corte possam passar pelo processo de laminação.

Na indústria de papel, há também a necessidade do corte em duas fases. Neste caso, a bobina jumbo (como é chamada a bobina mestre) deve ser cortada em sub-bobinas, as quais são novamente cortadas em máquinas (chamadas recortadeiras) que restringem as larguras das

sub-bobinas entre L_{min} e L_{max} . Alguns itens que não passam pela cortadeira (há clientes que demandam sub-bobinas) são itens livres. Itens (na realidade, retângulos), de larguras diversas, são considerados de mesmo agrupamento se tiverem o mesmo comprimento. Veja a figura 3.5.

Figura 3.5: Os comprimentos dos itens 1 e 2 devem ser iguais para serem cortados na mesma recortadeira.

Esses problemas têm sido tratados na literatura por heurísticas muito simplistas (Correia et al., 2004), evitando-se abordar o Problema da Mochila Compartimentada.

3.3 Uma Aplicação Prática: O Problema de Corte de Bobinas de Aço com Laminação

Nesta seção descrevemos o corte em bobinas de aço sujeitas à laminação e procuramos enfatizar os principais aspectos inerentes ao problema, além de fazermos uma descrição dos custos envolvidos no processo de produção.

Esse problema surge em algumas empresas metalúrgicas que produzem tubos de aço para diversas aplicações. Fitas devem ser produzidas a partir do corte de bobinas de aço em estoque. Essas fitas, por sua vez, serão utilizadas na confecção dos tubos soldados, que terão finalidades específicas.

3.3.1. Considerações Iniciais

Antes de prosseguir, consideremos as seguintes definições encontradas em (Ferreira et al., 1990), (Carvalho, 1991), (Pereira, 1993), (Carvalho e Rodrigues, 1994), (Hoto, 1996) e (Hoto, 2001).

Definições:

- Bobinas mestres são os objetos a serem cortados (mochilas a serem preenchidas). Tais bobinas são identificadas pelos seus pesos (entre 10000 e 13500 Kg), larguras (variando de 1000 e 1200 mm), espessura do aço e teor de carbono do aço segundo os critérios do SAE (*Society of Automotive Engineers*). Os mais utilizados são: SAE 1008, SAE 1010, SAE 1012, SAE 1017 e SAE 1021.
- Bobinas intermediárias são as bobinas obtidas durante a primeira etapa de corte (os compartimentos). As bobinas intermediárias herdam algumas de suas características das bobinas mestres, como a espessura e o teor de carbono do aço. Porém, por sua vez, as bobinas intermediárias terão seus próprios pesos e larguras.
- Fitas são os itens obtidos durante a segunda etapa de corte, a partir das bobinas intermediárias. As fitas possuem características bem definidas como a largura (de acordo com o diâmetro dos tubos a serem produzidos), a espessura e o tipo de aço.

A figura 3.6 representa o processo de obtenção dessas fitas (ou rolos, como pode ser visto) a partir de uma bobina mestre. Vale salientar que, das etapas apresentadas na figura, a etapa de recozimento não apresenta características relevantes ao problema estudado.

Figura 3.6: Esquema do processo de corte de uma bobina mestre em indústrias de corte de aço.

3.3.2. O Processo de Corte

Uma peculiaridade do problema ao se efetuar cortes em bobinas de aço sujeitas a um processo técnico de laminação, reside no fato de que as bobinas mestres não podem ser laminadas devido a sua largura, pois existem restrições físicas, impostas pelo laminador, estabelecendo limites nas larguras das bobinas para que essas possam ser laminadas (veja seção 3.3.4). Portanto, as bobinas mestres devem ser cortadas em bobinas intermediárias, cujas larguras respeitem as restrições do laminador e, posteriormente, recortadas em fitas. O

processo de laminação é executado visando ajustar a espessura das bobinas intermediárias de acordo com as fitas demandadas (na prática, cerca de 60% da produção necessita de laminação). A figura 3.7 dá uma idéia mais clara do processo de corte das bobinas.

Figura 3.7: O processo de corte em bobinas de aço.

Denomina-se “Refugo” todo tipo de perda inevitável de material durante o processo de obtenção das fitas, e “Sobra” todas as outras categorias de perdas de material. Temos dois refugos intrínsecos: um durante a primeira etapa de corte e outro durante a segunda etapa de corte (Ferreira et al., 1990). A figura 3.8 ilustra melhor este fato:

Figura 3.8: Corte em bobinas de aço com refugos embutidos.

Na prática, por uma restrição no número de facas, o número de bobinas intermediárias na primeira etapa é limitado.

Uma dificuldade no processo de corte decorre da existência de demandas baixas, de maneira que a utilização de uma bobina mestre com peso consideravelmente alto poderá proporcionar excesso de produção, mesmo que apenas uma fita de um determinado tipo apareça no padrão de corte. Para evitar este excesso, Hoto (2001) considera uma formulação $1\frac{1}{2}$ -dimensional, introduzindo uma variável no comprimento da bobina mestre. Embora essa abordagem pareça interessante, há uma resistência operacional que a impede de ser implementada na prática, portanto, não vamos considerá-la neste trabalho.

3.3.3. A Cortadeira

Grosso modo, como visto na figura 3.6, uma bobina mestre é desenrolada e o processo de corte para obtenção das bobinas intermediárias é feito longitudinalmente por “discos de corte” (não há cortes transversais e por isso podemos entender que o problema é unidimensional). Nesta etapa, existem perdas intrínsecas (que podem ser visualizadas na figura 3.8 como refugos) nas laterais da bobina, para eliminar as irregularidades das bordas, variando entre 6 e 8 mm. Posteriormente, as bobinas intermediárias são enroladas e passam, ou não, pelo processo de laminação. Se as espessuras das fitas forem idênticas à espessura da bobina mestre, não é necessário o processo de laminação. O processo de corte é ilustrado na figura 3.9.

Figura 3.9: Perfil do processo de corte em bobinas de aço.

3.3.4. Laminação e Recorte

A laminação, incluída no processo de produção ilustrado na figura 3.6, tem por objetivo diminuir (quando necessário) a espessura do material de uma bobina intermediária. Esse processo se dá à temperatura ambiente (laminação a frio) por meio de cilindros de laminação que efetuam pressão sobre a lâmina de aço (figura 3.10).

Figura 3.10: Perfil do processo de laminação.

Cabe notar que, eventualmente, uma mesma bobina intermediária poderá sofrer mais de uma laminação. Isto é necessário para que se possa obter a espessura desejada do material (a redução é de 15 a 20% da espessura em cada passo chegando ao máximo de redução entre 50 e 60% da espessura inicial). Como todas as outras, a máquina que executará a laminação (o laminador) possui restrições do tipo: Não é possível laminar uma bobina intermediária com menos de 154 mm e mais de 456 mm de largura (isto limita os tamanhos dos compartimentos, conforme definição do Problema da Mochila Compartimentada, seção 3.1). A figura 3.11 mostra uma bobina mestre cortada em 3 bobinas intermediárias e cada uma delas submetida a um processo de laminação com reduções de espessura variadas.

Figura 3.11: Laminação das bobinas intermediárias em uma mesma bobina mestre.

Devemos salientar que, quando submetemos uma mesma bobina intermediária ao processo de laminação, o material adquire alguns fatores físicos indesejáveis. Em face desse inconveniente é necessário um tratamento térmico ao material, denominado recozimento. No recozimento, as bobinas intermediárias são acondicionadas em fornos especiais onde permanecem por um certo período e, em seguida, são lentamente resfriadas até atingirem a temperatura ambiente. O processo todo leva cerca de 24 horas e constitui um gargalo na produção. Daí a importância de se laminar o mínimo possível. Após esse processo, as bobinas intermediárias passam por um novo processo de laminação onde um tratamento adequado de acabamento é efetuado. Por fim, as bobinas intermediárias são submetidas à “recortadeira”, uma máquina especial, semelhante à cortadeira, porém preparada para que se obtenha as fitas. Nessa máquina, como na cortadeira, existe uma perda intrínseca nas laterais das bobinas intermediárias variando de 4 a 8 mm. O limite de largura imposto pelas máquinas que farão o recorte é de 600mm.

Basicamente, um resumo do fluxo da produção pode ser verificado pelo esquema representado na figura 3.12. Entretanto, vale salientar a possibilidade de haver pequenas alterações.

Figura 3.12: Fluxo da produção de tubos de aço.

3.3.5. Custos dos Processos Envolvidos no Problema

Como visto, custos inerentes ao processo e relativos às perdas de material são típicos do problema. Dentre estes custos, podemos destacar: o custo de laminação; custos devido a refugos que, de uma maneira geral, estão intimamente ligados aos tipos de equipamentos disponíveis e a fatores tecnológicos e custos devido aos acertos de corte (troca de padrões).

O custo de laminação é uma preocupação fundamental e, neste sentido, tenta-se evitar ao máximo possível este tipo de operação, pois laminar, como já ressaltamos, é um processo caro. Naturalmente, o processo de laminação nem sempre poderá ser evitado, portanto, nesse caso é interessante buscar uma minimização deste processo (Hoto, 1996).

Em síntese, as diversas peculiaridades da linha de produção aqui citadas apontam para um problema complexo. Além disso, a necessidade do processo de laminação induz padrões de corte pouco evidentes, pois nem todas as fitas têm a mesma espessura. Uma empresa na capital paulista que trabalha com este problema indica uma perda de material entre 10% e 15%, o que corresponde a uma média mensal entre 500 e 750 toneladas de aço (Hoto, 2001).

3.4 Um Modelo Matemático para o Problema da Mochila Compartimentada

Nesta seção, apresentamos uma modelagem matemática para o Problema da Mochila Compartimentada. Lembramos que a cada compartimento da mochila tem-se associado um

único agrupamento k de itens, ou seja, não se admite itens de agrupamentos distintos no mesmo compartimento. Os dados (já apresentados na seção 3.2) são:

- Dados da Mochila:
 - L : capacidade (Kg) da mochila.
- Dados dos Itens:
 - $M = \{1, \dots, m\}$: conjunto dos tipos de itens;
 - l_i : peso (Kg) do item i ($l_i > 0$), $i=1, \dots, m$;
 - v_i : valor de utilidade do item i ($v_i \geq 0$), $i=1, \dots, m$;
 - d_i : limite máximo de itens i na mochila, $i=1, \dots, m$.
- Dados dos Agrupamentos:
 - K : número total de agrupamentos distintos;
 - A_k : agrupamento k , $k=1, \dots, K$, ($M = A_1 \cup A_2 \dots \cup A_K$, $A_i \cap A_j = \emptyset$, com $i \neq j$);
 - c_k : custo de inclusão de um compartimento de itens do agrupamento A_k , onde $c_k \geq 0$, $k=1, \dots, K$;
 - L_{min}^k : capacidade mínima (Kg) de cada compartimento associado ao agrupamento A_k ;
 - L_{max}^k : capacidade máxima (Kg) de cada compartimento associado ao agrupamento A_k ($L_{min}^k \leq L_{max}^k \leq L$);
 - S_k : perda (Kg) decorrente da inclusão de um novo compartimento preenchido com itens do agrupamento A_k na mochila;
 - N_k : número total de possíveis compartimentos para o agrupamento A_k (por exemplo: $A_1 = \{1, 2\}$, então $L_{11} = l_1 + l_2 + S_1$, $L_{21} = 2l_1 + l_2 + S_1$, etc., são duas possíveis capacidades de compartimentos para o agrupamento 1. L_{jk} referencia o j -ésimo compartimento associado ao agrupamento A_k . O número N_k depende essencialmente dos itens do agrupamento A_k e dos limitantes L_{min}^k e L_{max}^k).

Variáveis:

- α_{ijk} : número de itens do tipo i , do agrupamento A_k , no compartimento do tipo j , $i=1, \dots, m$, $k=1, \dots, K$ e $j=1, \dots, N_k$;
- β_{jk} : número de vezes que o compartimento do tipo j alocado com o agrupamento A_k é repetido na mochila, $k=1, \dots, K$ e $j=1, \dots, N_k$.

Assim, o j -ésimo compartimento com itens do agrupamento A_k tem:

- A capacidade dada por (itens e refugo):

$$L_{jk} = \sum_{i \in A_k} l_i \alpha_{ijk} + S_k, \quad k = 1, \dots, K \text{ e } j = 1, \dots, N_k. \quad (13)$$

- O valor de utilidade dado por:

$$V_{jk} = \sum_{i \in A_k} v_i \alpha_{ijk}, \quad k = 1, \dots, K \text{ e } j = 1, \dots, N_k. \quad (14)$$

Um modelo matemático para o problema da mochila compartimentada pode ser escrito por:

$$\text{Maximizar } \sum_{k=1}^K \sum_{j=1}^{N_k} (V_{jk} - c_k) \beta_{jk} \quad (15.1)$$

$$\text{Sujeito a: } V_{jk} = \sum_{i \in A_k} v_i \alpha_{ijk}, \quad k = 1, \dots, K \text{ e } j = 1, \dots, N_k \quad (15.2)$$

$$L_{jk} = \sum_{i \in A_k} l_i \alpha_{ijk} + S_k, \quad k = 1, \dots, K \text{ e } j = 1, \dots, N_k \quad (15.3)$$

$$L_{min}^k \leq L_{jk} \leq L_{max}^k, \quad k = 1, \dots, K \text{ e } j = 1, \dots, N_k \quad (15.4)$$

$$\sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk} \leq d_i, \quad i \in A_k, \quad k = 1, \dots, K \quad (15.5)$$

$$\sum_{k=1}^K \sum_{j=1}^{N_k} L_{jk} \beta_{jk} \leq L \quad (15.6)$$

$$\alpha_{ijk} \geq 0, \text{ inteiro}, \quad i = 1, \dots, m, \quad k = 1, \dots, K \text{ e } j = 1, \dots, N_k \quad (15.7)$$

$$\beta_{jk} \geq 0, \text{ inteiro}, \quad k = 1, \dots, K \text{ e } j = 1, \dots, N_k. \quad (15.8)$$

A função objetivo (15.1), a ser maximizada, fornece o valor de utilidade total, descontado o custo de alocar os compartimentos; as restrições (15.2) e (15.3) fornecem, respectivamente, o valor de utilidade e a capacidade de cada compartimento (obviamente, estas restrições podem ser eliminadas, substituindo-as nas demais restrições); as restrições (15.4) impõem limites físicos aos compartimentos; as restrições (15.5) limitam o número de itens na mochila e, por fim, a restrição (15.6) corresponde aos limites físicos da mochila. Observe que o problema (15.1-15.8) é um problema de otimização não linear inteiro.

O modelo (15.1-15.8) permite considerar um agrupamento especial de itens, chamados “itens livres”, os quais podem ser incluídos na mochila em um compartimento com características diferenciadas (veja seção 3.2). Nas heurísticas propostas no próximo capítulo, denominamos o agrupamento K como o agrupamento de itens livres. Assim, para $k = K$, a

restrição (15.4) é desnecessária (ou seja, $L_{min}^K = 0$ e $L_{max}^K = L$, de modo que é redundante) e $c_K=0$. Na aplicação em interesse, isto é, no corte de bobinas de aço sujeito à laminação, os limitantes dos compartimentos são idênticos: $L_{min}^k = L_{min}$ e $L_{max}^k = L_{max}$, exceto para o compartimento dos itens livres.

3.5 Um Limitante Superior para o Problema da Mochila Compartimentada

Marques (2000) apresenta um modelo relaxado para o Problema da Mochila Compartimentada, o qual será explorado no capítulo 5 para avaliar a qualidade das soluções heurísticas (capítulo 4).

Considere agora as seguintes variáveis:

- x_i : número de itens do tipo i na mochila, $i = 1, \dots, m$;
- y_k : número de compartimentos ocupados por itens da classe k , $k = 1, \dots, K$.

O modelo relaxado para (15.1-15.8) é dado por:

$$\text{Maximizar } \sum_{i=1}^m v_i x_i - \sum_{k=1}^K c_k y_k \quad (16.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i \leq L - \left(\sum_{k=1}^K S_k y_k \right) \quad (16.2)$$

$$y_k \cdot L_{min}^k \leq \sum_{i \in A_k} l_i x_i + S_k y_k \leq y_k \cdot L_{max}^k, \quad k = 1, \dots, K \quad (16.3)$$

$$0 \leq x_i \leq d_i, \text{ inteiro}, \quad i = 1, \dots, m \quad (16.4)$$

$$y_k \geq 0, \text{ inteiro}, \quad k = 1, \dots, K. \quad (16.5)$$

Note que o problema (16.1-16.5) é um modelo de otimização linear inteiro e pode ser resolvido por um pacote convencional de otimização inteira, ao contrário do problema (15.1-15.8) que envolve muitas variáveis e é não linear.

A partir do modelo (15.1-15.8) é possível chegar ao modelo (16.1 – 16.5) apenas efetuando algumas operações sobre as equações do modelo (15.1-15.8). As definições das variáveis dos problemas (15.1-15.8) e (16.1-16.5) permitem estabelecer:

$$x_i = \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk}, \quad i \in A_k, k = 1, \dots, K. \quad (17)$$

$$y_k = \sum_{j=1}^{N_k} \beta_{jk}, \quad k = 1, \dots, K. \quad (18)$$

Para (15.1), substituindo-se (15.2):

$$\sum_{k=1}^K \sum_{j=1}^{N_k} \left(\sum_{i \in A_k} v_i \alpha_{ijk} - c_k \right) \beta_{jk} = \sum_{k=1}^K \sum_{i \in A_k} v_i \left(\sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk} \right) - \sum_{k=1}^K c_k \sum_{j=1}^{N_k} \beta_{jk} \quad (19)$$

Considerando $(M = \{1, \dots, m\} = A_1 \cup A_2 \dots \cup A_K, A_k \cap A_l = \emptyset, \text{ com } k \neq l)$ e (17) e (18), podemos reescrever (19) por:

$$\sum_{i=1}^m v_i x_i - \sum_{k=1}^K c_k y_k,$$

o que coincide com (16.1).

Para (15.4), substituindo-se (15.3): $L_{\min}^k \leq \sum_{i \in A_k} l_i \alpha_{ijk} + S_k \leq L_{\max}^k, k = 1, \dots, K.$

Multiplicando os termos das inequações por β_{jk} e somando-se em j , encontramos

$$L_{\min}^k \cdot \sum_{j=1}^{N_k} \beta_{jk} \leq \sum_{j=1}^{N_k} \left(\sum_{i \in A_k} l_i \alpha_{ijk} + S_k \right) \cdot \beta_{jk} \leq L_{\max}^k \cdot \sum_{j=1}^{N_k} \beta_{jk}.$$

$$\text{Portanto, } L_{\min}^k \cdot \sum_{j=1}^{N_k} \beta_{jk} \leq \sum_{i=1}^m l_i \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk} + S_k \sum_{j=1}^{N_k} \beta_{jk} \leq L_{\max}^k \cdot \sum_{j=1}^{N_k} \beta_{jk}.$$

Substituindo (17) e (18) segue $L_{\min}^k \cdot y_k \leq \sum_{i=1}^m l_i x_i + S_k y_k \leq L_{\max}^k \cdot y_k$ como em (16.3). Ou

seja, as restrições (16.3) são obtidas a partir das restrições do problema (15.1-15.8).

Para (15.6) substituindo (15.3), temos:

$$\begin{aligned} \sum_{k=1}^K \sum_{j=1}^{N_k} L_{jk} \beta_{jk} &\leq L = \sum_{k=1}^K \sum_{j=1}^{N_k} \left(\sum_{i \in A_k} l_i \alpha_{ijk} + S_k \right) \beta_{jk} \leq L = \sum_{k=1}^K \sum_{i \in A_k} l_i \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk} + \sum_{k=1}^K S_k \sum_{j=1}^{N_k} \beta_{jk} \leq L = \\ &= \sum_{i=1}^m l_i x_i + \sum_{k=1}^K S_k y_k \leq L, \text{ como em (16.2).} \end{aligned}$$

Para (15.5), considerando (15.7) e (17) segue: $0 \leq \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk} \leq d_i, i \in A_k, k = 1, \dots, K$, e,

portanto, $0 \leq x_i \leq d_i, i = 1, \dots, m$, como em (16.4).

A restrição (16.5) segue diretamente de (15.8).

Desta forma, demonstramos o seguinte teorema:

Teorema: O problema (16.1-16.5) é uma relaxação de (15.1-15.8).

3.5.1 Solução do Modelo (16.1–16.5) Inviável para o Problema da Mochila Compartimentada

Considere o exemplo numérico da seção 3.2 e um possível resultado obtido a partir do modelo (16.1-16.5) com os seguintes valores para as variáveis $x_i, i = 1, \dots, m$ e $y_k, k = 1, \dots, K$:

Vetor solução $\mathbf{X} = (0, 1, 3, 0, 0, 0, 2)$ e

Vetor solução $\mathbf{Y} = (3, 0, 1)$.

Essa solução é factível ao problema (16.1-16.5), porém, como veremos a seguir, não é possível obter uma solução factível para o modelo (15.1-15.8). Vejamos, a solução $\mathbf{X}$ e $\mathbf{Y}$ nos indica que teríamos 3 compartimentos preenchidos com o agrupamento 1, porém um arranjo dos itens nesses compartimentos que satisfaça as restrições impostas no modelo (15.1–15.8) não é possível (observe que um compartimento não comporta uma unidade do item tipo 2 e uma unidade do item tipo 3: $l_2 + l_3 + S_l = 11 > L_{max}$). A figura 3.13 ilustra a situação:

Figura 3.13: Solução infactível para o problema e admitida por (16.1 – 16.5).

A impossibilidade de compartimentação decorre da restrição (16.3), que considera y_k compartimentos do agrupamento k na mochila sem, porém, se preocupar em como os itens alocados nesses y_k compartimentos estarão divididos. Como vimos no exemplo da figura 3.13, em certos casos, não é possível compartimentar os itens seleccionados nos y_k compartimentos da solução obtida. Este exemplo mostra que o problema (16.1–16.5) pode fornecer um limitante superior estrito para o Problema da Mochila Compartimentada.

3.5.2 Relaxação de Integralidade (Novo Limitante Superior)

Como veremos a seguir, o modelo (16.1-16.5) pode ser reduzido ao clássico Problema da Mochila Restrito se relaxarmos a condição de integralidade da variável y_k . A restrição (16.3) é satisfeita com igualdade no limite superior. Seja:

$$y_k = \frac{\sum_{i \in A_k} l_i x_i}{L_{max}^k S_k}, \quad k = 1, \dots, K \quad (20)$$

Substituindo (20) no modelo (16.1 – 16.5), temos:

$$\text{Maximizar } \sum_{i=1}^m v_i x_i - \sum_{k=1}^K \left(c_k \frac{\sum_{i \in A_k} l_i x_i}{L_{\max}^k - S_k} \right) \quad (21.1)$$

$$\text{Sujeito a: } \sum_{i=1}^m l_i x_i + \sum_{k=1}^K \left(S_k \frac{\sum_{i \in A_k} l_i x_i}{L_{\max}^k - S_k} \right) \leq L \quad (21.2)$$

$$0 \leq x_i \leq d_i \text{ e inteiro, } i = 1, \dots, m. \quad (21.3)$$

Reescrevendo, temos:

$$\text{Maximizar } \sum_{k=1}^K \sum_{i \in A_k} \left(v_i - \frac{c_k l_i}{L_{\max}^k - S_k} \right) x_i, \quad i \in A_k \quad (22.1)$$

$$\text{Sujeito a: } \sum_{k=1}^K \sum_{i \in A_k} \left(l_i + \frac{S_k l_i}{L_{\max}^k - S_k} \right) x_i \leq L, \quad i \in A_k \quad (22.2)$$

$$0 \leq x_i \leq d_i \text{ e inteiro, } k = 1, \dots, K, i \in A_k. \quad (22.3)$$

Fazendo:

$$\begin{aligned} \bar{v}_i &= v_i - \frac{c_k l_i}{L_{\max}^k - S_k} \\ \bar{l}_i &= l_i + \frac{S_k l_i}{L_{\max}^k - S_k} \quad \text{para } k=1, \dots, K, i \in A_k \text{ (ou, } i=1, \dots, m). \end{aligned}$$

Temos, então:

$$\text{Maximizar } \bar{v}_1 x_1 + \dots + \bar{v}_m x_m \quad (23.1)$$

$$\text{Sujeito a: } \bar{l}_1 x_1 + \dots + \bar{l}_m x_m \leq L \quad (23.2)$$

$$0 \leq x_i \leq d_i \text{ e inteiro, } i = 1, \dots, m. \quad (23.3)$$

que é o clássico Problema da Mochila Restrito.

Portanto, ao relaxarmos a condição de integralidade da variável y_k e desenvolvendo o modelo (16.1-16.5) chegamos a um modelo que representa o clássico Problema da Mochila Restrito. O problema (23.1–23.3) também fornece um limitante superior (de pior qualidade que o fornecido pelo modelo (16.1–16.5)) para o Problema da Mochila Compartmentada.

3.6 Estudos Relacionados com Problema da Mochila Compartimentada

Nesta seção, fazemos uma revisão de alguns estudos da literatura que tratam o Problema da Mochila Compartimentada. Hoto (2001) estudou o que denominamos de Problema da Mochila Compartimentada Irrestrito, ou seja, sem considerar a restrição (15.5). Pereira (1993), Carvalho (1991), Ferreira et al. (1990) e Hoto (1996) estudaram implicitamente o Problema da Mochila Compartimentada focando seus estudos no problema de corte de bobinas de aço com laminação descrito na seção 3.3. Por sua vez, Zak (2002) estudou um problema de corte em duas fases, que surge na indústria de corte de papéis, porém sem o agrupamento de itens. A seguir, fazemos uma breve revisão desses estudos.

3.6.1 O Problema da Mochila Compartimentada Irrestrito

Hoto (2001) estudou o que chamamos de Problema da Mochila Compartimentada Irrestrito, ou seja, desconsiderando a restrição (15.5). Hoto propõe dois métodos para a geração de padrões de corte compartimentados. O primeiro, denominado COMPEX, encontra a solução ótima para o Problema da Mochila Compartimentada Irrestrito e, resumidamente, tem seu algoritmo apresentado a seguir:

Algoritmo COMPEX:

Passo 1: Pré Processamento

1.1 Entrada de dados.

Passo 2: Determinação dos Compartimentos e suas Utilidades

2.1 Para $k = 1$ até K (todos os agrupamentos), faça:

2.2 Resolva a mochila (24.1–24.3) variando $w_j = L_{\max}^k$ até $L_{\min}^k$:

$$\text{Maximizar } V_j = \sum_{i \in A_k} v_i \alpha_{ij} \quad (24.1)$$

$$\text{Sujeito a: } \sum_{i \in A_k} l_i \alpha_{ij} + S_k = w_j \quad (24.2)$$

$$\alpha_{ij} \geq 0 \text{ e inteiro, } i \in A_k. \quad (24.3)$$

2.3 Armazene os valores de V_j para todos os compartimentos construídos e calcule um custo de utilização, c_j , para cada compartimento construído, que pode depender dos itens alocados.

Passo 3: Preenchimento da Mochila

3.1 Construa o subconjunto W' dos índices de todos os compartimentos construídos no passo 2.

3.2 Resolva a mochila (25.1-25.3) e encontre β_j :

$$\text{Maximizar } \sum_{j \in W'} (V_j - c_j) \beta_j \quad (25.1)$$

$$\text{Sujeito a: } \sum_{j \in W'} w_j \beta_j \leq L \quad (25.2)$$

$$\beta_j \geq 0 \text{ e inteiro, } j \in W'. \quad (25.3)$$

Passo 4: Recuperação dos Compartimentos Escolhidos

4.1 Armazene os compartimentos construídos com $\beta_j \neq 0$.

Note que na solução do problema (25.1-25.3), o qual aloca os compartimentos obtidos pelo problema (24.1-24.3), não leva em consideração a limitação dos itens.

O segundo método proposto (COMPMT) tenta contornar a resolução dos inúmeros problemas da mochila no passo 2 do COMPLEX e, assim, acelerar a resolução do problema. Sendo assim, substitui-se a resolução do passo 2 pelo simples cálculo do limitante de Martello e Toth (1990) o qual fornece uma boa aproximação superior para as utilidades dos compartimentos. Os custos c_j , que dependiam dos itens associados aos compartimentos construídos são definidos por, agora, c_k , constante para todos os compartimentos do agrupamento k . Feito isso, o passo 3 é realizado normalmente deixando para, no passo 4, construir efetivamente os compartimentos cujo valor $\beta_j \neq 0$.

Tais abordagens, embora eficientes, desconsideram a restrição (15.5) do modelo (15.1-15.8) e que aparece naturalmente no problema de corte de bobinas de aço sujeitas a laminação. Considerando que as demandas das fitas de aço (itens demandados) podem ser bastante reduzidas, os métodos de Hoto, ao desconsiderarem a restrição (15.5), permitem que

uma quantidade considerável de padrões inviáveis do ponto de vista prático (constroem padrões com excesso de produção de fitas) sejam gerados. As heurísticas a serem apresentadas no capítulo seguinte evitam que isso aconteça e mostram algumas dificuldades a mais que as restrições (15.5) inserem na resolução do Problema da Mochila Compartimentada, além de destacar as diferenças entre os dois estudos. Por fim, vale ressaltar que as abordagens de Hoto (2001) foram as primeiras a tratar formalmente o Problema da Mochila Compartimentada.

3.6.2 Abordagens Heurísticas

A seguir, apresentamos, brevemente, três abordagens para o problema de corte em bobinas de aço sujeitas ao processo técnico de laminação, encontradas em (Carvalho, 1991), (Ferreira et al., 1990) e (Hoto, 1996).

Um Problema de Corte em Duas Fases (Carvalho, 1991)

Carvalho (1991), estimulado por problemas encontrados na indústria de cortes em bobinas de aço, estudou o problema, procurando resolvê-lo de forma a elaborar um sistema de apoio ao planejamento da produção em uma indústria de aço que, até então, era executado manualmente por uma equipe de programadores experientes.

Nesse caso, Carvalho (1991) resolve o problema de corte de estoque utilizando uma técnica de geração de colunas, onde um padrão de corte compartimentado é construído a cada iteração do método. Para preencher uma bobina do estoque é preciso que as bobinas intermediárias estejam definidas. A proposta de Carvalho foi construir, para cada item considerado, todos os compartimentos homogêneos (isto é, compostos por apenas um tipo de item). Para ilustrarmos essa idéia, imagine o exemplo dado no início deste capítulo, considerando apenas os agrupamentos 1 e 2. Temos, então:

	Agrupamento 1			Agrupamento 2	
Item (i)	1	2	3	4	5
Utilidade (v_i)	2	3	5	1	2
Largura (l_i)	3	4	6	2	4

Tabela 3.4 – Dados dos itens dos agrupamentos 1 e 2.

Considere, ainda, $L_{\min}^k = 7$, $L_{\max}^k = 9$, $S_k = 0$ e $c_k = 0$ (Carvalho não considera perdas laterais nem custos pela alocação de compartimentos) para $k = 1$ e 2. Assim, segundo a proposta de Carvalho (1991), são construídas apenas as seguintes bobinas intermediárias:

- 1) Para o item 1, uma bobina intermediária de largura $L_{11} = 3 * 3 = 9$ e utilidade $V_{11} = 6$.
- 2) Para o item 2, uma bobina intermediária de largura $L_{21} = 2 * 4 = 8$ e utilidade $V_{21} = 6$.
- 3) Para o item 3, é inviável a construção de um padrão homogêneo que satisfaça os limites mínimo e máximo de compartimentação.
- 4) Para o item 4, uma bobina intermediária de largura $L_{41} = 4 * 2 = 8$ e utilidade $V_{41} = 4$.
- 5) Para o item 5, uma bobina intermediária de largura $L_{51} = 2 * 4 = 8$ e utilidade $V_{51} = 4$.

Uma vez construídas as bobinas intermediárias homogêneas, Carvalho preenche a bobina resolvendo um problema da mochila. Considerando um bobina mestre de largura $L=30$, o seguinte padrão de corte poderia ser obtido:

Figura 3.14: Padrão de corte segundo Carvalho (1991).

Neste mesmo exemplo, entretanto, observamos que esta estratégia não considera padrões mais expressivos. Sendo assim, muitas soluções são desconsideradas comprometendo o desempenho desta abordagem.

Fica claro, a partir de um exemplo bastante simplificado, que a abordagem adotada por Carvalho (1991) ignora uma grande quantidade dos possíveis padrões compartimentados, reduzindo consideravelmente o espaço de busca.

Um Problema de Corte de Bobinas em Duas Fases (Ferreira et al., 1990)

Ferreira et al. (1990) foram um dos primeiros a estudar o problema de corte em bobinas de aço envolvendo um processo técnico de laminação. Nesse estudo, Ferreira et al. descrevem um procedimento heurístico para resolver o problema que consiste nos seguintes passos:

- Passo 1.** Ordene os itens, em ordem decrescente, de acordo com um fator dado pela multiplicação da demanda atual do item pela respectiva largura.
- Passo 2.** Escolha um subconjunto de itens (5 a 9 itens são os valores típicos) com maior valor para o fator calculado no passo 1 e constrói-se 3 bobinas intermediárias.
- Passo 3.** Ordene a lista dos itens escolhidos no passo 2 de forma decrescente, segundo suas larguras.
- Passo 4.** Gere padrões de corte segundo a heurística FFD (*First Fit Decreasing*).
- Passo 5.** Verifique se padrões viáveis (aqueles que não produzem excesso de demanda) foram gerados e vá ao passo 6. Caso contrário, incremente o número de bobinas intermediárias e a lista de itens escolhidos no passo 2 e retorne ao passo 3.
- Passo 6.** Defina um outro fator (que combina alto uso com baixa perda) para cada padrão e selecione o “melhor” padrão viável.
- Passo 7.** Verifique se o padrão selecionado no passo 6 é “aceitável” (considerar o agrupamento dos itens e a definição das bobinas intermediárias). Caso contrário, faça essa verificação para o próximo melhor padrão viável. Se nenhum padrão aceitável foi encontrado, incremente o número de bobinas intermediárias e a lista de itens escolhidos no passo 2 e retorne ao passo 3.
- Passo 8.** Use o padrão selecionado e atualize as demandas. Se houver demanda remanescente retorne ao passo 1, caso contrário PARE.

Os autores relatam que bons padrões são gerados por meio deste procedimento, entretanto, muitos padrões podem ser eventualmente descartados após sua construção, como descrevem os passos 5 e 7. Com isso, Ferreira et al. (1990) dedica um esforço considerável sem garantir que soluções viáveis sejam obtidas. Entendemos que é fundamental a construção de padrões que efetivamente sejam aceitáveis para uso.

Um Problema de Corte Unidimensional Sujeito a Restrições de Agrupamento (Hoto, 1996)

Em seus primeiros estudos sobre o problema de corte em bobinas de aço sujeitas à laminação, Hoto enfatizou a busca por uma heurística gulosa que resolvesse o problema. Inicialmente, um processo de ordenação das fitas dentro de cada um dos agrupamentos tornou-se necessário de forma a dar prioridades de alocação àquelas fitas de maior largura combinada com maior demanda, relação que foi denominada “eficiência da fita”. Além disto, uma ordenação dos agrupamentos foi realizada para que, da mesma forma das fitas, fosse dada prioridade àqueles agrupamentos “mais eficazes”.

A estratégia de Hoto baseava-se, primeiramente, em gerar um padrão de referência e, a partir deste, os demais padrões. Este padrão de referência era formado tomando o agrupamento mais eficaz e alocando suas fitas segundo a ordenação pré-estabelecida de suas respectivas eficiências, respeitando a largura útil da bobina mestre, a largura máxima de laminação, o número máximo de fitas que podem aparecer numa bobina intermediária, bem como o fato de não exceder a demanda da fita.

Uma vez obtido este primeiro padrão de corte (padrão de referência) tomava-se a primeira bobina intermediária, dela retirava-se a maior quantidade possível de fitas de mais baixa eficiência, respeitando o comprimento mínimo permissível para laminação. Em seguida calculava-se a largura útil disponível e alocava-se novas fitas segundo a mesma estratégia utilizada para gerar o padrão de referência.

Desta forma, obtinha-se o segundo padrão de corte. Para que fosse obtido o terceiro padrão, tomava-se a primeira e segunda bobinas intermediárias do segundo padrão e procedia-se com o mesmo raciocínio até que não fosse mais possível alocar novas bobinas intermediárias. A figura 3.15 ilustra este processo, onde um padrão gerado coincide parcialmente com o padrão anterior.

Figura 3.15: Ilustração da heurística de Hoto (1996).

Com esta heurística, Hoto buscava resolver o problema de corte em bobinas de aço sujeitas ao processo técnico de laminação. Os resultados obtidos não demonstram que a heurística proposta resolvia o problema de forma satisfatória.

3.6.3 Uma Abordagem por Programação Linear Inteira (Pereira, 1993)

Pereira (1993), em seus estudos sobre o problema de corte em bobinas de aço, procurou resolver o problema propondo uma modelagem baseada em programação linear inteira. O modelo proposto foi resolvido, utilizando-se, para isto, o método *branch-and-bound* (ou de enumeração implícita). Com o objetivo de diminuir o processo de busca, uma estimativa do limitante inferior do modelo foi utilizada, a fim de fornecer uma solução inicial. O emprego desse recurso gerou uma imprevisibilidade na qualidade das soluções, contribuindo para seu posterior abandono. O autor relata que os resultados foram insatisfatórios.

Um outro problema constatado durante a sua resolução está relacionado ao excesso de produção. Esse inconveniente poderia ser contornado com a introdução de um termo na função objetivo do modelo, que penalizasse o excedente da produção.

Com tal estudo, podemos perceber que, dado às especificidades do problema, é justificável a busca de um algoritmo especializado com abordagem heurística para sua resolução.

3.6.4 Abordagens para o Problema de Corte em Bobinas de Papel em Duas Fases

Zak (2002) aborda o problema de corte de estoque unidimensional, em que bobinas são cortadas em duas fases, gerando um conjunto de bobinas intermediárias (compartimentos). Nesse artigo, um método de geração de linhas e colunas é proposto, no qual a geração de linhas decorre da necessidade de novas bobinas intermediárias e a geração de colunas a novos padrões, combinando as bobinas intermediárias. Embora o corte da bobina em estoque tenha a organização de compartimentos, este problema difere do que definimos por mochila compartimentada, devido aos itens poderem pertencer indistintamente a qualquer compartimento, isto é, há um único agrupamento dos itens. Entretanto, há recortadeiras (máquinas de corte das bobinas intermediárias) que exigem que os comprimentos dos itens sejam iguais, o que definiria agrupamentos (itens de mesmo tipo de papel e mesmo comprimento). Esta restrição não é considerada no método de geração de linhas e colunas de Zak.

Por outro lado, Correia et al. (2004) também estudaram o problema de corte de estoque unidimensional em uma indústria de papel e geraram heurísticamente um conjunto de padrões de corte, rejeitando aqueles que não pudessem ser compartimentados. Isto foi feito por não disporem de métodos para a resolução do Problema da Mochila Compartimentada.

No próximo capítulo, apresentamos quatro heurísticas propostas para a resolução do Problema da Mochila Compartimentada, modelado em (15.1–15.8), e que têm apresentado bons resultados.

Capítulo 4. Heurísticas para o Problema da Mochila Compartimentada

Neste capítulo, são apresentadas quatro heurísticas para a resolução do problema de preenchimento de uma única mochila compartimentada, considerando uma quantidade máxima de cada item a ser alocado na mochila (d_i , para $i = 1, \dots, m$). Das quatro heurísticas, três foram propostas inicialmente em (Marques, 2000) e modificadas em (Marques e Arenales, 2002). Uma quarta heurística foi proposta a partir de observações feitas dos resultados das três primeiras. As heurísticas apostam na direção de que o custo de incluir um compartimento é alto, de modo que esses devem ser construídos o mais próximo possível dos limites máximos para cada agrupamento $L_{\max}^k$, $k = 1, \dots, m$ (no caso do corte de bobinas de aço esse custo corresponde à laminação, que é um processo caro e demorado, justificando a aposta das heurísticas). Além disso, mostramos como as heurísticas se comportam a partir do exemplo numérico para o Problema da Mochila Compartimentada dado no capítulo 3. A seguir, fazemos uma exposição das heurísticas propostas.

4.1 Heurística de Decomposição

Esta heurística consiste de duas fases: Na primeira fase são resolvidos $(K-1)$ problemas da mochila de capacidade $L_{\max}^k$, $k = 1, \dots, K-1$, um para cada agrupamento (exceto o agrupamento K , dos itens livres), o que resulta nos melhores compartimentos associados aos agrupamentos. Na segunda fase, um problema da mochila clássico é resolvido, considerando os compartimentos obtidos na primeira fase como superitens (na verdade, uma combinação de itens) juntamente com os itens livres, para o carregamento da mochila.

Uma solução particular para (15.1-15.8) é construída com as seguintes características:

- Com o intuito de maximizar a função objetivo (15.1) é conveniente a construção de V_{jk} grande e, portanto, para cada agrupamento k , construímos apenas o melhor compartimento ($N_k = 1$, de modo que j pode ser suprimido) maximizando (15.2) restrito a (15.3), (15.4) e (15.7) (veja problema (26.1-26.3)). O valor de α_{ik} (j suprimido) é determinado, isto é, para cada agrupamento k , temos um único compartimento;
- Com $N_k = 1$ (apenas um compartimento por agrupamento) e α_{ik} já calculado: determina-se L_k (tamanho do compartimento k , veja (15.3), com o índice j suprimido). O valor de β_k é determinado de modo a otimizar (15.1), sujeito às demais restrições do problema (15.1-15.8) (veja problema (27.1-27.5)).

O algoritmo da Heurística de Decomposição é apresentado a seguir.

Algoritmo:

Passo 1: Selecionar o melhor compartimento para o agrupamento k , $k=1, \dots, K-1$, resolvendo o seguinte problema da mochila, de capacidade $L_{\max}^k$:

$$V_k = \text{Maximizar } \sum_{i \in A_k} v_i \alpha_{ik} \quad (26.1)$$

$$\text{Sujeito a: } \sum_{i \in A_k} l_i \alpha_{ik} + S_k \leq L_{\max}^k \quad (26.2)$$

$$0 \leq \alpha_{ik} \leq d_i \text{ e inteiro, para } i \in A_k, k = 1, \dots, K-1. \quad (26.3)$$

Seja $L_k = \sum_{i \in A_k} l_i \alpha_{ik} + S_k$: tamanho do compartimento k (note que α_{ik} são fixados neste passo).

Passo 2: Resolver o seguinte problema da mochila envolvendo os compartimentos determinados no passo 1 e os itens livres:

- β_k é o número de vezes que o compartimento associado ao agrupamento k é repetido na mochila,
- γ_k é o número de vezes que o item livre k é repetido na mochila.

$$\text{Maximizar } \sum_{k=1}^{K-1} (V_k - c_k) \beta_k + \sum_{k \in A_K} v_k \gamma_k \quad (27.1)$$

$$\text{Sujeito a: } \sum_{k=1}^{K-1} L_k \beta_k + \sum_{k \in A_K} l_k \gamma_k \leq L \quad (27.2)$$

$$\alpha_{ik} \beta_k \leq d_i, \quad i \in A_k, \quad k = 1, \dots, K-1 \quad (27.3)$$

$$\gamma_k \leq d_k, \quad k \in A_K, \quad (27.4)$$

$$\alpha_{ik}, \gamma_k \geq 0 \text{ e inteiros, } i \in A_k, k = 1, \dots, K-1, k \in A_K. \quad (27.5)$$

Note que a somatória original em (15.6) desaparece, pois $N_k = 1$, ou seja, apenas um compartimento é considerado por agrupamento.

4.2 Heurística do Melhor Compartimento

Nesta heurística, como na anterior, o melhor compartimento para cada um dos agrupamentos de itens é determinado. Temos definido, então:

$$\begin{aligned} L'_k &= \begin{cases} (L_k + S_k), & k = 1, \dots, (K-1) \\ l_i, & k = (K-1) + i, i = 1, \dots, |A_K| \end{cases} \\ V'_k &= \begin{cases} V_k - c_k, & k = 1, \dots, (K-1) \\ v_i, & k = (K-1) + i, i = 1, \dots, |A_K| \end{cases} \end{aligned} \quad (28)$$

Em seguida, escolhe-se o compartimento de maior valor por unidade (V'_k / L'_k), o qual é alocado na mochila. Atualizam-se os dados e repete-se o processo até que nenhum novo compartimento possa ser incluído. Com isto é evitado o problema (27.1-27.5) do algoritmo anterior. Porém, apesar de apenas um compartimento ser gerado por agrupamento, a cada passo, é possível ter diferentes compartimentos com o mesmo agrupamento, algo excluído na heurística anterior, onde era permitida apenas a repetição do melhor compartimento. O algoritmo é apresentado passo a passo a seguir.

Algoritmo:

Passo 1: $lutil \leftarrow L$; “ $lutil$ guarda a capacidade disponível na mochila a cada iteração”

Passo 2: Para $k = 1$ a $K-1$, faça:

2.1 Se ($lutil \geq L_{max}^k$)

então $L_compartimento = L_{max}^k$

senão Se ($lutil < L_{min}^k$)

então $L_compartimento = lutil$ e considera-se apenas os itens livres em (29.1-29.3),

senão ($L_{min}^k \leq lutil < L_{max}^k$) $L_compartimento = lutil$

2.2 Selecionar o melhor compartimento para o agrupamento k resolvendo o seguinte problema da mochila de capacidade $L_compartimento$:

$$V_k = \text{Maximizar } \sum_{i \in A_k} v_i \alpha_{ik} \quad (29.1)$$

$$\text{Sujeito a: } \sum_{i \in A_k} l_i \alpha_{ik} + S_k \leq L_compartimento \quad (29.2)$$

$$0 \leq \alpha_{ik} \leq d_i \text{ e inteiro, para } i \in A_k, k = 1, \dots, K-1. \quad (29.3)$$

Passo 3:

3.1 Escolha k^* tal que: $\text{Max} \left\{ \frac{V_k}{L_k}, k = 1, \dots, (K-1) + |A_K| \right\}$

3.2 Atualização:

$lutil \leftarrow lutil - L_{k^*}$

Se k^* em (3.1) corresponder a um compartimento

então $d_i \leftarrow d_i - \alpha_{ik^*}, \forall i \in A_k$

senão $d_i \leftarrow d_i - 1$

3.3 Se $lutil \geq \text{Min} \{ L_k^*, k = 1, \dots, (K-1) + |A_K| \}$, volte ao Passo 2, senão PARE.

4.3 Heurística dos “z” Melhores Compartimentos

Esta heurística consiste basicamente na determinação dos “z” melhores compartimentos associados a cada agrupamento, ou seja, nas “z” melhores soluções do problema da mochila para cada agrupamento. A resolução de um problema de otimização linear inteira considerando esses compartimentos como superitens, juntamente com os itens livres, determina um padrão de corte para a mochila compartimentada. O algoritmo é apresentado a seguir.

Algoritmo:

Passo 1: Selecionar os “z” melhores compartimentos para o agrupamento k determinando as “z” melhores soluções para o seguinte problema da mochila de capacidade $L_{\max}^k$, sendo $V_{1k} \geq V_{2k} \geq \dots \geq V_{zk}$ os “z” melhores valores de (30.1) e suas respectivas soluções $\alpha_{ijk}, j = 1, \dots, z$.

Para $k = 1, \dots, K-1$ e $j = 1, \dots, z$, resolva:

$$V_{jk} = \text{Maximizar } \sum_{i \in A_k} v_i \alpha_{ijk} \quad (30.1)$$

$$\text{Sujeito a: } \sum_{i \in A_k} l_i \alpha_{ijk} + S_k \leq L_{\max}^k \quad (30.2)$$

$$0 \leq \alpha_{ijk} \leq d_i \text{ e inteiro, para } i \in A_k. \quad (30.3)$$

Passo 2: Resolver o seguinte problema de programação inteira envolvendo os $((K-1)*z)$ compartimentos selecionados e os $|A_K|$ itens livres:

- β_{jk} é o número de vezes que o compartimento j para o agrupamento k é repetido na mochila,
- γ_k é o número de vezes que o item livre k é repetido na mochila.

$$\text{Maximizar } \sum_{k=1}^{K-1} \sum_{j=1}^z (V_{jk} - c_k) \beta_{jk} + \sum_{k \in A_K} v_k \gamma_k \quad (31.1)$$

$$\text{Sujeito a: } \sum_{k=1}^{K-1} \sum_{j=1}^z L_{jk} \beta_{jk} + \sum_{k \in A_K} l_k \gamma_k \leq L \quad (31.2)$$

$$\sum_{j=1}^z \alpha_{ijk} \beta_{jk} \leq d_i, \quad i \in A_k \text{ e } k = 1, \dots, K-1. \quad (31.3)$$

$$0 \leq \gamma_k \leq d_k, \quad k \in A_K. \quad (31.4)$$

$$\beta_{jk} \geq 0, \text{ inteiro, } k = 1, \dots, K-1 \text{ e } j = 1, \dots, z. \quad (31.5)$$

$$\text{em que } L_{jk} = \sum_{i \in A_k} l_i \alpha_{ijk} + S_k, \quad k = 1, \dots, K-1.$$

Nota: Embora a Heurística de Decomposição seja um caso particular da Heurística dos “z” Melhores Compartimentos, com $z = 1$, destacamo-la por resolver uma seqüência de problemas da mochila. O problema (31.1-31.5) é um problema de otimização inteira com várias restrições, ao contrário de (27.1-27.5) que é um problema da mochila restrito. Para resolver os subproblemas que surgem nas heurísticas, propusemos algumas modificações no método de enumeração implícita proposto por Gilmore e Gomory (1963). Este algoritmo modificado é detalhado no apêndice A.

4.4 Heurística do Melhor Compartimento para “w” Capacidades

Esta heurística consiste basicamente na determinação, para cada agrupamento, do melhor compartimento para “w” possíveis capacidades diferentes dos compartimentos, ou seja, para cada agrupamento teremos “w” compartimentos com capacidades diferentes. Como na heurística anterior, a resolução de um problema de programação linear inteira (considerando como itens os $(w*(K-1))$ compartimentos obtidos na primeira fase e os $|A_K|$ itens livres) determina um padrão de corte para a mochila compartimentada, o qual é resolvido pelo mesmo algoritmo utilizado para resolver o problema (31.1–31.5). O algoritmo da Heurística do Melhor Compartimento para “w” Capacidades é apresentado a seguir.

Algoritmo:

Passo 1: Selecionar o melhor compartimento para cada uma das “w” capacidades do agrupamento k . Para $k = 1, \dots, K-1$, faça:

$$1.1 \ L_compartimento = L_{max}^k,$$

$$1.2 \ j = 1,$$

Enquanto $(L_compartimento > L_{min}^k)$ e $(j \leq w)$, faça:

1.2.1 Resolva:

$$V_{jk} = \text{Maximizar} \sum_{i \in A_k} v_i \alpha_{ijk} \quad (32.1)$$

$$\text{Sujeito a: } \sum_{i \in A_k} l_i \alpha_{ijk} + S_k \leq L_compartimento \quad (32.2)$$

$$0 \leq \alpha_{ijk} \leq d_i \text{ e inteiro, para } i \in A_k. \quad (32.3)$$

$$1.2.2 \ L_{jk} = \sum_{i \in A_k} l_i \alpha_{ijk} + S_k,$$

$$1.2.3 \ L_compartimento = L_{jk} - l,$$

Passo 2: Resolver o problema de otimização inteira envolvendo os $((K-1)*w)$ compartimentos selecionados e os $|A_K|$ itens livres como em (31.1–31.5).

Na próxima seção apresentamos o exemplo numérico para o Problema da Mochila Compartimentada apresentado no capítulo anterior e ilustramos como cada heurística resolve o problema.

4.5 Um Exemplo Numérico para o Problema da Mochila Compartimentada

Considere novamente o exemplo numérico dado no capítulo 3. Ilustramos, agora, o comportamento das heurísticas apresentadas na seção anterior. Para facilitar a compreensão, reproduziremos os dados do exemplo.

Suponha um objeto (mochila) com capacidade $L = 30$ e considere 3 agrupamentos ($K=3$) em que o último representa o agrupamento de itens livres. Tal agrupamento (K) possui características particulares tais como $L_{\min}^K = 0$, $L_{\max}^K = L$ e $c_K = 0$, sem perda de generalidade para o problema definido. As tabelas 4.1 a 4.3 reproduzem os dados dos agrupamentos, bem como os limites de capacidade e os custos dos compartimentos:

Agrupamento 1: $L_{\min}^1 = 7, L_{\max}^1 = 9, S_1 = 1$ e $c_1 = 0,5$			
Itens	Peso (l_i)	Utilidade (v_i)	Limites (d_i)
1	3	2	2
2	4	3	3
3	6	5	3

Tabela 4.1: Dados do agrupamento 1.

Agrupamento 2: $L_{\min}^2 = 7, L_{\max}^2 = 9, S_2 = 1$ e $c_2 = 0,8$			
Itens	Peso (l_i)	Utilidade (v_i)	Limites (d_i)
4	2	1	3
5	4	2	2

Tabela 4.2: Dados do agrupamento 2.

Agrupamento 3: $L_{\min}^3 = 0, L_{\max}^3 = 30, S_3 = 0$ e $c_3 = 0$			
Itens	Peso (l_i)	Utilidade (v_i)	Limites (d_i)
6	5	2	1
7	3	1	2

Tabela 4.3: Dados do agrupamento 3 (itens livres).

Figura 4.1: Representação dos itens.

A seguir, apresentamos os padrões de corte gerados por cada uma das heurísticas, para o conjunto de dados considerado.

4.5.1 Heurística de Decomposição:

No passo 1 da Heurística de Decomposição, os seguintes compartimentos são construídos para cada agrupamento:

Agrupamento 1: $L_1 = 9$ e $\bar{v}_1 = 5,5$
Representação do compartimento para o agrupamento 1.

Agrupamento 2: $L_2 = 9$ e $V_2 = 3,2$
Representação do compartimento para o agrupamento 2.

Agrupamento 3: $l_6 = 5$ e $v_6 = 2$
Representação do item livre 6.

$l_7 = 3$ e $v_7 = 1$
Representação do item livre 7.

Passo 2 - Padrão de corte gerado:

Figura 4.2: Representação do padrão de corte gerado pela Heurística de Decomposição.

Valor da Função Objetivo	Capacidade Utilizada	Capacidade Ociosa
12,7	29	1

Tabela 4.4: Resultados obtidos pela Heurística de Decomposição.

4.5.2 Heurística do Melhor Compartimento:

No passo 1, temos: $l_{util} = 30$;

No passo 2, temos os compartimentos construídos para cada agrupamento:

Agrupamento 1: $L_1 = 9$ e $V_1 = 5,5$
Representação do compartimento para o agrupamento 1.

Agrupamento 2: $L_2 = 9$ e $V_2 = 3,2$
Representação do compartimento para o agrupamento 2.

Agrupamento 3: $l_6 = 5$ e $v_6 = 2$
Representação do item livre 6.
$l_7 = 3$ e $v_7 = 1$
Representação do item livre 7.

Passo 3 - Compartimento selecionado para preencher a mochila: A_1

Atualizações: $l_{util} = 30 - 9 = 21$

$$d_2 = 3 - 2 = 1$$

Passo 2 - Novo compartimento construído para o agrupamento 1 (agrupamento escolhido no passo anterior):

Agrupamento 1: $L_1 = 7$ e $V_1 = 4,5$
Representação do compartimento para o agrupamento 1.

Passo 3 - Compartimento selecionado para preencher a mochila: A_1

Atualizações: $l_{util} = 23 - 7 = 16$

$$d_3 = 3 - 1 = 2.$$

Passo 2 - Novo compartimento construído para o agrupamento 1 (agrupamento escolhido no passo anterior):

Agrupamento 1: $L_1 = 7$ e $V_1 = 4,5$
Representação do compartimento para o agrupamento 1.

Passo 3 - Compartimento selecionado para preencher a mochila: A_1

Atualizações: $l_{util} = 16 - 7 = 9$

$$d_3 = 2 - 1 = 1.$$

Passo 2 - Novo compartimento construído para o agrupamento 1 (agrupamento escolhido no passo anterior):

Agrupamento 1: $L_1 = 7$ e $V_1 = 4,5$
Representação do compartimento para o agrupamento 1.

Passo 3 - Compartimento selecionado para preencher a mochila: A_1

Atualizações: $l_{util} = 9 - 9 = 0 \Rightarrow$ Padrão concluído!

$$d_2 = 1 - 1 = 0.$$

Padrão de corte gerado:

Figura 4.3: Representação do padrão de corte gerado pela Heurística do Melhor Compartimento.

Valor da Função Objetivo	Capacidade Utilizada	Capacidade Ociosa
19,0	30	0

Tabela 4.5: Resultados obtidos pela Heurística do Melhor Compartimento.

4.5.3 Heurística dos “z” Melhores Compartimentos ($z = 2$):

No passo 1 da Heurística dos “z” Melhores Compartimentos, os seguintes compartimentos são construídos para cada agrupamento:

Agrupamento 1: $L_{11} = 9$ e $V_{11} = 5,5$
 Representação do compartimento A_{11} .
$L_{12} = 7$ e $V_{12} = 4,5$
 Representação do compartimento A_{12} .

Agrupamento 2: $L_{21} = 9$ e $V_{21} = 3,2$
 Representação do compartimento A_{21} .
$L_{22} = 9$ e $V_{22} = 3,2$
 Representação do compartimento A_{22} .

Agrupamento 3: $L_6 = 5$ e $v_6 = 2$
 Representação do item livre 6.
$l_7 = 3$ e $v_7 = 1$
 Representação do item livre 7.

Passo 2 - Padrão de corte gerado:

Figura 4.4: Representação do padrão de corte gerado pela Heurística dos “z” Melhores Compartimentos ($z = 2$).

Valor da Função Objetivo	Capacidade Utilizada	Capacidade Ociosa
19,0	30	0

Tabela 4.6: Resultados obtidos pela Heurística dos “z” Melhores Compartimentos.

4.5.4 Heurística do Melhor Compartimento para “w” Capacidades ($w = 2$):

No passo 1 da Heurística do Melhor Compartimento para “w” Capacidades, os seguintes compartimentos são construídos para cada agrupamento:

Agrupamento 1: $L_{11} = 9$ e $V_{11} = 5,5$
Representação do compartimento A_{11} .
$L_{12} = 7$ e $V_{12} = 4,5$
Representação do compartimento A_{12} .

Agrupamento 2: $L_{21} = 9$ e $V_{21} = 3,2$
Representação do compartimento A_{21} .
$L_{22} = 7$ e $V_{22} = 2,2$
Representação do compartimento A_{22} .

Agrupamento 3: $L_6 = 5$ e $v_6 = 2$
Representação do item livre 6.
$l_7 = 3$ e $v_7 = 1$
Representação do item livre 7.

Passo 2 - Padrão de corte gerado:

Figura 4.5: Representação do padrão de corte gerado pela Heurística do Melhor Compartimento para “w” Capacidades ($w = 2$).

Valor da Função Objetivo	Capacidade Utilizada	Capacidade Ociosa
19,0	30	0

Tabela 4.7: Resultados obtidos pela Heurística do Melhor Compartimento para “w” Capacidades.

Embora as soluções obtidas pelas heurísticas tenham sido bem parecidas, o exemplo mostra claramente como os compartimentos gerados são diferentes. No capítulo seguinte,

conjuntos de exemplos gerados aleatoriamente são resolvidos e os desempenhos das heurísticas são analisados.

Capítulo 5. Experimentos Computacionais

Neste capítulo, apresentamos os resultados obtidos a partir da implementação das heurísticas de Decomposição, do Melhor Compartimento, dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades propostas no capítulo anterior. Detalhes dos algoritmos para resolução dos problemas da mochila, que surgem em cada heurística, estão no apêndice A.

5.1 Considerações Iniciais

As implementações desenvolvidas neste trabalho foram efetuadas utilizando o ambiente integrado de desenvolvimento de programação Delphi. A máquina utilizada possui um processador Intel Celeron 650MHz com 320 MBytes de memória RAM. Além disso, para a obtenção dos valores dos limitantes propostos no capítulo anterior, foi usado o pacote CPLEX 7.5 instalado em uma estação de trabalho *Sun*. Os resultados expostos neste capítulo foram obtidos a partir destas especificações.

5.2 Conjuntos de Dados

As heurísticas propostas foram testadas em uma série de exemplos gerados aleatoriamente, classificados de acordo com as quatro seguintes características:

- i)* número de agrupamentos;
- ii)* número de itens por agrupamento;
- iii)* número de itens livres e,
- iv)* limites sobre o número de itens.

Tais características tentam simular situações observadas com dados reais. A seguir apresentamos as características que foram consideradas para cada conjunto de dados. Alguns parâmetros foram fixados em todos os exemplos, simulando situações encontradas em uma indústria metalúrgica. Adotamos o comprimento (e não o peso, como feito até agora) como característica de ocupação dos itens na mochila.

- **Parâmetros fixados:**

1. Capacidade da mochila: $L = 1200$ mm;
2. Perda embutida no preenchimento da mochila $S = 12$ mm;
3. Capacidades mínima e máxima para os compartimentos: $L_{\min}^k = 154$ mm e $L_{\max}^k = 456$ mm, $k=1, \dots, K-1$. Caso $k = K$, $L_{\min}^k = 0$ mm e $L_{\max}^k = 1200$ mm.
4. Perda devido à inclusão de compartimento (imperfeições nas bordas): $S_k = 8$ mm, $k=1, \dots, K-1$. Caso $k = K$, $S_k = 0$ mm.

- **Parâmetros gerados aleatoriamente:**

5. Comprimento dos itens gerados aleatoriamente: $l_i \in [20, 444]$, $i \in A_k$, $k = 1, \dots, K$.
6. $v_i = l_i + \sigma$, $\sigma \in [0, l_i]$, (Obs: Na geração de colunas, quando o objetivo é minimizar a perda, cada mochila compartimentada terá os valores de utilidade neste formato, em que σ é o multiplicador simplex).
7. $c_k \in [1, 100]$, $k = 1, \dots, K-1$ (Para $k = K$, temos $c_k = 0$).

- **Parâmetros variados: (para cada parâmetro, consideramos duas possibilidades)**

- i) **Número de Agrupamentos:**

1: número pequeno de agrupamentos: $K \in [3, 10]$.

2: número grande de agrupamentos: $K \in [11, 20]$.

ii) Número de Itens por Agrupamento:

3: número pequeno de itens: $|A_k| \in [1,6], k = 1, \dots, K-1$.

4: número grande de itens: $|A_k| \in [7,15] k = 1, \dots, K-1$.

iii) Número de Itens Livres:

5: número pequeno de itens livres: $|A_K| \in [1,7]$.

6: número grande de itens livres: $|A_K| \in [8,20]$.

iv) Limites sobre o Número de Itens em uma Mesma Mochila:

7: limitante baixo: $d_i \in [1, 3], i \in A_k, k = 1, \dots, K$.

8: limitante alto: $d_i \in [4, 15], i \in A_k, k = 1, \dots, K$.

Na prática das indústrias metalúrgicas, numa das aplicações do Problema da Mochila Compartimentada, o percentual de itens livres é bastante variado podendo ser da ordem de 60% (ou seja, 60% dos itens demandados não necessitam de laminação). Embora as heurísticas descritas sejam projetadas para uma única mochila, devem ser executadas em outras ocasiões (por exemplo, na técnica de geração de colunas de Gilmore e Gomory, 1961, 1963, uma mochila compartimentada deve ser resolvida a cada iteração do método simplex ou como geradora de padrões eficientes na heurística de exaustão, Hinxman, 1980).

A seguir, é feito um resumo dos resultados obtidos para vários exemplos gerados. Para cada conjunto relacionado na tabela 5.1, 30 exemplos foram gerados aleatoriamente e resolvidos pelas 4 heurísticas (Decomposição, Melhor Compartimento, “z” Melhores Compartimentos e Melhores Compartimentos para “w” Capacidades). Nos casos das heurísticas dos “z” Melhores Compartimentos e dos Melhores Compartimentos para “w” Capacidades estes exemplos foram resolvidos com alguns valores diferentes de “z” e “w” (utilizamos $z = 2, 3$ e 5 e $w = 2, 3$ e 5).

5.3 Resultados Obtidos

Nesta seção, apresentamos os resultados obtidos na execução de 30 exemplos para cada conjunto de dados. Para isto, mostramos inicialmente os resultados obtidos para as Heurísticas de Decomposição e do Melhor Compartimento. Na sequência, mostramos os resultados obtidos para as Heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades. Em cada uma das heurísticas variamos os valores de “z” e “w” e observamos o comportamento dos valores das funções objetivos alcançados para cada situação. Por fim, mostramos um comparativo entre todas as heurísticas propostas. Os valores na coluna “limitantes” são as médias dos valores obtidos para os limitantes nos respectivos exemplos, resolvendo-se o modelo (16.1-16.5), através da utilização do pacote CPLEX. O gap representa a diferença em percentual entre o valor do limitante e o valor obtido pelas heurísticas.

A seguir, a tabela 5.1 mostra os valores médios obtidos para os exemplos gerados aleatoriamente, seguindo ao parâmetros de cada conjunto de dados, utilizando as Heurísticas de Decomposição e do Melhor Compartimento.

Conjuntos	Parâmetros	Decomposição		Melhor Compartimento		Limitantes
		H. DEC	GAP	H. MEL	GAP	
A	1,3,5,7	2060,90	1,89%	2002,47	4,67%	2100,63
B	1,3,5,8	2158,27	1,26%	2100,67	3,89%	2185,70
C	1,3,6,7	2200,83	0,70%	2115,17	4,57%	2216,43
D	1,3,6,8	2267,73	0,13%	2174,00	4,25%	2270,60
E	1,4,5,7	2061,97	5,31%	2086,93	4,17%	2177,70
F	1,4,5,8	2137,07	3,91%	2150,90	3,29%	2223,97
G	1,4,6,7	2222,30	1,19%	2154,57	4,21%	2249,17
H	1,4,6,8	2249,97	0,76%	2171,87	4,21%	2267,27
I	2,3,5,7	2105,90	2,12%	2053,70	4,55%	2151,50
J	2,3,5,8	2160,83	2,48%	2128,87	3,92%	2215,73
K	2,3,6,7	2202,67	0,67%	2116,17	4,57%	2217,47
L	2,3,6,8	2269,10	0,53%	2184,63	4,24%	2281,30
M	2,4,5,7	2066,43	6,74%	2131,57	3,80%	2215,80
N	2,4,5,8	2140,73	4,99%	2190,03	2,81%	2253,27
O	2,4,6,7	2197,37	1,91%	2077,60	7,25%	2240,07
P	2,4,6,8	2258,57	0,87%	2166,10	4,93%	2278,37
Média		2172,54	2,21%	2125,33	4,33%	2221,56

Tabela 5.1: Valores médios da função objetivo obtidos executando-se as Heurísticas de Decomposição e do Melhor Compartimento.

A figura 5.1 mostra um gráfico que permite a visualização dos dados na tabela 5.1.

Figura 5.1: Gráfico com as curvas que representam os valores médios da função objetivo obtidos executando as Heurísticas de Decomposição e do Melhor Compartimento.

Observe que a Heurística de Decomposição apresenta, em geral, um desempenho melhor que a Heurística do Melhor Compartimento (gap de 2,21% contra 4,33%), embora não seja dominante.

A seguir, apresentamos os valores médios obtidos pela resolução dos exemplos para cada conjunto de dados, com as Heurísticas dos “z” Melhores Compartimentos (tabela 5.2 e figura 5.2) e do Melhor Compartimento para “w” Capacidades (tabela 5.3 e figura 5.3). Em cada uma das heurísticas variamos os valores de “z” e “w” entre 2, 3 e 5 e observamos o comportamento dos valores das funções objetivos alcançados para cada situação. A tabela 5.2, a seguir, mostra os resultados obtidos utilizando a Heurística dos “z” Melhores Compartimentos.

		"z" Melhores (2)		"z" Melhores (3)		"z" Melhores (5)		
Conj.	Parâmetros	Z = 2	GAP	Z = 3	GAP	Z = 5	GAP	Limitantes
A	1,3,5,7	2086,17	0,69%	2088,10	0,60%	2092,33	0,40%	2100,63
B	1,3,5,8	2167,40	0,84%	2169,53	0,74%	2174,17	0,53%	2185,70
C	1,3,6,7	2212,43	0,18%	2213,30	0,14%	2213,30	0,14%	2216,43
D	1,3,6,8	2267,93	0,12%	2267,93	0,12%	2270,13	0,02%	2270,60
E	1,4,5,7	2072,50	4,83%	2101,77	3,49%	2125,67	2,39%	2177,70
F	1,4,5,8	2147,17	3,45%	2158,27	2,95%	2171,27	2,37%	2223,97
G	1,4,6,7	2226,93	0,99%	2229,20	0,89%	2232,50	0,74%	2249,17
H	1,4,6,8	2250,70	0,73%	2250,70	0,73%	2252,53	0,65%	2267,27
I	2,3,5,7	2124,30	1,26%	2133,33	0,84%	2141,90	0,45%	2151,50
J	2,3,5,8	2175,50	1,82%	2196,53	0,87%	2205,63	0,46%	2215,73
K	2,3,6,7	2209,47	0,36%	2210,77	0,30%	2212,77	0,21%	2217,47
L	2,3,6,8	2275,27	0,26%	2277,90	0,15%	2277,90	0,15%	2281,30
M	2,4,5,7	2108,10	4,86%	2144,77	3,21%	2160,50	2,50%	2215,80
N	2,4,5,8	2167,70	3,80%	2183,43	3,10%	2209,17	1,96%	2253,27
O	2,4,6,7	2203,97	1,61%	2208,47	1,41%	2219,17	0,93%	2240,07
P	2,4,6,8	2260,27	0,79%	2263,53	0,65%	2264,17	0,62%	2278,37
Média		2184,74	1,66%	2193,60	1,26%	2201,44	0,91%	2221,56

Tabela 5.2: Valores médios da função objetivo obtidos executando a Heurística dos "z" Melhores Compartimentos, com "z" = 2, 3 e 5.

A tabela 5.2 pode ser visualizada na figura 5.2 facilitando a comparação dos resultados quando variamos o valor de "z".

Figura 5.2: Gráfico com as curvas comparativas que representam os valores médios da função objetivo obtidos executando a Heurística dos "z" Melhores Compartimentos.

Para a Heurística do Melhor Compartimento para “w” Capacidades obtivemos os seguintes resultados:

Conj.	Parâmetros	"w" Capacidades (2)		"w" Capacidades (3)		"w" Capacidades (5)		Limitantes
		W = 2	GAP	W = 3	GAP	W = 5	GAP	
A	1,3,5,7	2084,03	0.79%	2088,23	0.59%	2090,23	0.50%	2100,63
B	1,3,5,8	2165,80	0.91%	2174,00	0.54%	2176,57	0.42%	2185,70
C	1,3,6,7	2214,63	0.08%	2215,57	0.04%	2215,57	0.04%	2216,43
D	1,3,6,8	2270,13	0.02%	2270,13	0.02%	2270,13	0.02%	2270,60
E	1,4,5,7	2093,70	3.86%	2125,10	2.42%	2150,53	1.25%	2177,70
F	1,4,5,8	2155,37	3.08%	2174,37	2.23%	2202,83	0.95%	2223,97
G	1,4,6,7	2224,27	1.11%	2232,03	0.76%	2239,80	0.42%	2249,17
H	1,4,6,8	2252,30	0.66%	2252,40	0.66%	2253,93	0.59%	2267,27
I	2,3,5,7	2131,53	0.93%	2136,83	0.68%	2145,07	0.30%	2151,50
J	2,3,5,8	2186,73	1.31%	2198,83	0.76%	2208,73	0.32%	2215,73
K	2,3,6,7	2208,00	0.43%	2211,07	0.29%	2214,80	0.12%	2217,47
L	2,3,6,8	2275,33	0.26%	2277,97	0.15%	2278,23	0.13%	2281,30
M	2,4,5,7	2131,80	3.79%	2161,33	2.46%	2199,90	0.72%	2215,80
N	2,4,5,8	2181,13	3.20%	2214,20	1.73%	2237,90	0.68%	2253,27
O	2,4,6,7	2207,50	1.45%	2213,57	1.18%	2224,43	0.70%	2240,07
P	2,4,6,8	2261,60	0.74%	2267,57	0.47%	2271,60	0.30%	2278,37
Média		2190,24	1.41%	2200,83	0.93%	2211,27	0.46%	2221,56

Tabela 5.3: Valores médios da função objetivo obtidos executando a Heurística do Melhor Compartimento para “w” Capacidades, com “w” = 2, 3 e 5.

A figura 5.3 facilita a comparação dos resultados quando variamos o valor de “w”.

Figura 5.3: Gráfico com as curvas comparativas que representam os valores médios da função objetivo obtidos executando a Heurística do Melhor Compartimento para “w” Capacidades.

A seguir, apresentamos a tabela 5.4 (e, posteriormente a figura 5.4) que faz um resumo dos resultados das quatro heurísticas propostas. Para efeito de comparação, utilizaremos o valor $z = 5$ para a Heurística dos “z” Melhores Compartimentos e $w = 5$ para a Heurística do Melhor Compartimento para “w” Capacidades.

Conj.	Param.	Decomposição		Melhor Compartimento		“z” Compartimentos (5)		“w” Capacidades (5)		Limitantes
		H. DEC	GAP	H. MEL	GAP	Z = 5	GAP	W = 5	GAP	
A	1,3,5,7	2060,9	1,89%	2002,5	4,67%	2092,3	0,40%	2090,2	0,50%	2100,6
B	1,3,5,8	2158,3	1,26%	2100,7	3,89%	2174,2	0,53%	2176,6	0,42%	2185,7
C	1,3,6,7	2200,9	0,70%	2115,2	4,57%	2213,3	0,14%	2215,6	0,04%	2216,4
D	1,3,6,8	2267,7	0,13%	2174,0	4,25%	2270,1	0,02%	2270,1	0,02%	2270,6
E	1,4,5,7	2062,0	5,31%	2086,9	4,17%	2125,7	2,39%	2150,5	1,25%	2177,7
F	1,4,5,8	2137,0	3,91%	2150,9	3,29%	2171,3	2,37%	2202,8	0,95%	2224,0
G	1,4,6,7	2222,3	1,19%	2154,6	4,21%	2232,5	0,74%	2239,8	0,42%	2249,2
H	1,4,6,8	2250,0	0,76%	2171,9	4,21%	2252,5	0,65%	2253,9	0,59%	2267,3
I	2,3,5,7	2106,0	2,12%	2053,7	4,55%	2141,9	0,45%	2145,1	0,30%	2151,5
J	2,3,5,8	2160,8	2,48%	2128,9	3,92%	2205,6	0,46%	2208,7	0,32%	2215,7
K	2,3,6,7	2202,7	0,67%	2116,2	4,57%	2212,8	0,21%	2214,8	0,12%	2217,5
L	2,3,6,8	2269,1	0,53%	2184,6	4,24%	2277,9	0,15%	2278,2	0,13%	2281,3
M	2,4,5,7	2066,4	6,74%	2131,6	3,80%	2160,5	2,50%	2199,9	0,72%	2215,8
N	2,4,5,8	2140,7	4,99%	2190,0	2,81%	2209,2	1,96%	2237,9	0,68%	2253,3
O	2,4,6,7	2197,4	1,91%	2077,6	7,25%	2219,2	0,93%	2224,4	0,70%	2240,1
P	2,4,6,8	2258,6	0,87%	2166,1	4,93%	2264,2	0,62%	2271,6	0,30%	2278,4
Média		2172,5	2,21%	2125,3	4,33%	2201,4	0,91%	2211,3	0,46%	2221,6

Tabela 5.4: Valores médios da função objetivo alcançados por todas as heurísticas para os conjuntos de dados.

Os dados na tabela 5.4 são ilustrados na figura 5.4, como segue.

Figura 5.4: Gráfico com as curvas comparativas que representam os valores médios da função objetivo obtidos executando as quatro heurísticas.

Por fim, analisamos os resultados das heurísticas para todos os 480 exemplos resolvidos, de acordo com a proximidade que os valores obtidos ficaram dos limitantes. Para isso, definiremos uma solução p -ótima, em que o parâmetro p é dado por:

$$p = \frac{(f(x_i^*) - f(x_i^j))}{f(x_i^*)} \quad (33),$$

com

$f(x_i^*)$: limitante para o exemplo i , $i = 1, \dots, 480$ e

$f(x_i^j)$: o valor alcançado pela heurística j , $j = 1, \dots, 4$ (1 – Decomposição, 2 – Melhor Compartimento, 3 – “z” Melhores Compartimentos e 4 – Melhor Compartimento para “w” Capacidades), para o exemplo i , $i = 1, \dots, 480$.

Sendo assim, dizemos que uma solução é, por exemplo, 0,025-ótima quando o resultado de (33) é menor que 0,025, ou seja, a solução está a 2,5% do ótimo. A tabela 5.5, a seguir, mostra a evolução das soluções de todos os exemplos resolvidos para as quatro heurísticas e o percentual de exemplares cujas soluções são p -ótimas, com p variando entre 0 e 0,275. Por exemplo, 34,38% dos exemplares gerados foram resolvidos otimamente pela

Heurística da Decomposição e 100% dos problemas foram resolvidos a 5% do ótimo com a Heurística do Melhor Compartimento para “w” Capacidades ($w = 5$).

	Decomposição	Melhor Compartimento	“z” Compartimentos			“w” Capacidades		
	H. DEC	H. MEL	Z = 2	Z = 3	Z = 5	W = 2	W = 3	W = 5
0-ótimo	34,4%	11,7%	41,5%	46,3%	49,8%	44,4%	50,6%	60,4%
0,025-ótimo	67,9%	40,4%	75,8%	81,7%	87,1%	79,4%	86,5%	95,4%
0,05-ótimo	86,7%	61,9%	91,5%	94,2%	97,7%	93,1%	97,1%	100,0%
0,075-ótimo	93,5%	76,7%	95,2%	97,1%	99,0%	96,9%	99,6%	100,0%
0,10-ótimo	97,1%	91,0%	97,9%	99,4%	99,6%	98,3%	99,8%	100,0%
0,125-ótimo	97,9%	99,6%	98,5%	99,6%	100,0%	99,0%	100,0%	100,0%
0,15-ótimo	98,5%	100,0%	99,0%	99,8%	100,0%	99,2%	100,0%	100,0%
0,175-ótimo	99,2%	100,0%	99,8%	100,0%	100,0%	100,0%	100,0%	100,0%
0,20-ótimo	99,4%	100,0%	99,8%	100,0%	100,0%	100,0%	100,0%	100,0%
0,225-ótimo	99,4%	100,0%	99,8%	100,0%	100,0%	100,0%	100,0%	100,0%
0,25-ótimo	99,8%	100,0%	99,8%	100,0%	100,0%	100,0%	100,0%	100,0%
0,275-ótimo	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%

Tabela 5.5: Evolução das soluções das heurísticas em proximidade do valor ótimo.

A figura 5.5 ilustra os dados representados na tabela 5.5. Para o gráfico desta figura consideramos, apenas, $z = 5$ para a Heurística dos “z” Melhores Compartimentos e $w = 5$ para a Heurística do Melhor Compartimento para “w” Capacidades.

Figura 5.5: Gráfico com as curvas que mostram a evolução das soluções das heurísticas em proximidade do valor ótimo.

Pela figura 5.5 verificamos que a Heurística do Melhor Compartimento para “w” Capacidades aproxima-se do valor ótimo mais rapidamente em relação às demais, porém, mais adiante, veremos que esta situação merece uma discussão mais ampla. Na próxima seção, fazemos uma análise mais detalhada dos resultados obtidos.

5.4 Análise dos Resultados Obtidos

A execução dos conjuntos de exemplos nos forneceu resultados que permitiram analisar com detalhes o comportamento das quatro heurísticas propostas. A seguir, discutimos comparativamente, os resultados obtidos para cada heurística e analisamos o desempenho de cada uma delas.

A Heurística de Decomposição mostrou-se, dentre todas, a mais dependente dos itens livres. O fato de se ter apenas um compartimento por agrupamento, construído visando atingir a capacidade máxima ($L_{\max}^k$, $k = 1, \dots, K-1$), dificulta a combinação dos compartimentos para o preenchimento da mochila. Tal situação agrava se não tivermos muitos itens livres para preencher a mochila. Apesar disso, 86,67% dos resultados da Heurística de Decomposição ficaram a 5% dos valores obtidos para os limitantes. Apesar de apresentar resultados ruins em alguns exemplos (a tabela 5.1 mostra isso), a Heurística de Decomposição mostrou-se melhor que a Heurística do Melhor Compartimento.

A Heurística do Melhor Compartimento possibilita a construção de compartimentos com capacidades diferentes a medida em que vamos preenchendo a mochila. Tal característica diminui a dependência desta heurística em relação ao número de itens livres ($|A_k|$) pois permite que compartimentos menores sejam construídos e alocados. Por outro lado, se tivermos um número pequeno de agrupamentos e, principalmente, de itens nesses agrupamentos, o desempenho desta heurística fica prejudicado. Outra situação desfavorável a esta heurística ocorre quando temos um baixo valor para os limites dos itens de um determinado tipo na mochila (d_i). Com isto, de maneira geral, a Heurística do Melhor Compartimento foi a que apresentou o pior desempenho, fornecendo 61,88% dos resultados a 5% do ótimo.

Por sua vez, a Heurística dos “z” Melhores Compartimentos apresentou uma melhora considerável no desempenho da Heurística de Decomposição (lembre que $z = 1$ corresponde a Heurística de Decomposição) quando inserimos um parâmetro “z” que indique quantos

compartimentos devem ser construídos para cada agrupamento. Como era de se esperar, a medida que aumentamos o valor do parâmetro “z”, a heurística tem uma melhora. Pela tabela 5.4, podemos ver que:

- para $z = 2$: 91,46% das soluções ficaram a 5% do ótimo,
- para $z = 3$: 94,17% das soluções ficaram a 5% do ótimo e
- para $z = 5$: 97,71% das soluções ficaram a 5% do ótimo.

Embora a melhora seja sensível, pudemos notar que, para valores baixos de “z” tal heurística constrói compartimentos, em geral, de mesma capacidade (pois estamos buscando os “z” melhores valores da função objetivo), não fornecendo uma melhora considerável no número de combinações, em relação a tamanho dos compartimentos, para o preenchimento da mochila. Por este motivo, podemos perceber que o desempenho da Heurística dos “z” Melhores Compartimentos continua dependendo do número de itens livres, para valores baixos de “z”. Ao aumentarmos o valor do parâmetro “z”, as soluções desta heurística caminham para o ótimo. Fazendo isso, estaríamos, ao mesmo tempo, buscando várias opções de compartimentos (com combinações de itens diferentes) de um mesmo tamanho e construindo compartimentos com tamanhos diferentes. Isto nos levaria a construção de todos os compartimentos possíveis para um agrupamento e conseqüentemente poderíamos alcançar a solução ótima do problema. Por outro lado, o desempenho em termos de tempo ficaria comprometido, pois envolveria uma enumeração completa para cada agrupamento, exigindo um processamento computacional considerável nas situações com um número nem tão elevado de itens por agrupamento.

Por fim, a Heurística do Melhor Compartimento para “w” Capacidades apresentou resultados muito bons para os exemplos considerados, sem que tivéssemos que elevar muito o valor do parâmetro “w”. O fato desta heurística buscar aumentar o poder combinatório das estratégias estabelecidas nas heurísticas anteriores, gerando “w” compartimentos de tamanhos diferentes para cada agrupamento, fez com que a sua curva de desempenho crescesse mais rápido que as demais. Pela tabela 5.4, vemos que:

- para $w = 2$: 93,13% das soluções ficaram a 5% do ótimo,
- para $w = 3$: 97,08% das soluções ficaram a 5% do ótimo e
- para $w = 5$: 100% das soluções ficaram a 5% do ótimo.

Vale frisar que tal heurística descarta as chamadas “soluções alternativas” consideradas pela Heurística dos “z” Melhores Compartimentos, em que dois compartimentos de mesmo tamanho e mesmo valor da função objetivo podem ser construídos com combinações de itens diferentes. Em alguns casos, tal fato inviabiliza a obtenção da solução ótima para o Problema da Mochila Compartimentada pois podemos descartar compartimentos importantes no preenchimento da mochila se considerarmos o fato que existe um limite máximo no número de itens de um determinado tipo dentro da mochila. Apesar disso, a Heurística do Melhor Compartimento para “w” Capacidades se mostrou uma alternativa muito boa pois oferece soluções de alta qualidade exigindo, relativamente, menos recursos computacionais e de tempo (seção 5.5). Tal heurística pode ser muito útil na resolução de um problema de corte de estoque envolvendo mochilas compartimentadas encontrado em indústrias de corte de bobinas de aço com laminação.

5.5 Análise dos Tempos de Execução

Nesta seção, analisamos os tempos médios gastos por cada uma das heurísticas na execução dos 480 exemplos. Estes tempos foram medidos em milisegundos e comprovam a rapidez com que os exemplos foram resolvidos. A tabela 5.6 mostra os tempos médios obtidos para cada heurística, lembrando que, para as heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades, foram utilizados 2, 3 e 5 como valores para os parâmetros “z” e “w”.

	Decomposição	Melhor Compartimento	“z” Compartimentos			“w” Capacidades		
	H. DEC	H. MEL	Z = 2	Z = 3	Z = 5	W = 2	W = 3	W = 5
Tempo Médio	1,04ms	4,35ms	15,98ms	28,83ms	35,71ms	3,88ms	7,23ms	9,71ms

Tabela 5.6: Tempos médios, em milisegundos, de cada heurística para os exemplos resolvidos.

Note que a Heurística de Decomposição foi a mais rápida entre todas. Tal fato era esperado devido à construção de um único compartimento para cada agrupamento envolvido no problema. Por sua vez, a Heurística do Melhor Compartimento teve um desempenho um pouco pior devido a necessidade de se construir mais compartimentos, elevando um pouco o tempo médio obtido pela heurística anterior. A Heurística dos “z” Melhores Compartimentos mostrou-se a de pior desempenho em relação ao tempo, apresentando, inclusive, um esperado crescimento a medida em que o valor do parâmetro “z” aumentava. É importante frisar que tal

desempenho é justificável devido às modificações efetuadas no algoritmo de enumeração implícita proposto por Gilmore e Gomory (1963) (método de enumeração implícita com a estratégia de busca em profundidade primeiro), de modo que a poda na árvore deva ser mais relaxada para que não se descarte as z melhores soluções, o que faz perder a eficiência do método. Por sua vez, a Heurística do Melhor Compartimento para “ w ” Capacidades requer a solução de “ w ” compartimentos (problema da mochila) para cada agrupamento. Como o método de busca em profundidade, com uso de limitantes, consegue, em muitos casos, a solução ótima do problema explorando poucas ramificações na árvore de decisões, o fato de aumentar o número de compartimentos calculados não tem tanta influência no desempenho da heurística, ao contrário do que ocorreu com a Heurística dos “ z ” Melhores Compartimentos.

A seguir, os resultados são mostrados no gráfico da figura 5.6 facilitando a comparação entre os valores obtidos.

Figura 5.6: Gráfico de tempos médios de cada heurística para os exemplos resolvidos.

Por fim, vale considerar que os baixos tempos médios obtidos pelas heurísticas, justificam a utilização das mesmas na resolução do problema de corte de estoque envolvendo mochilas compartimentadas através de métodos como, por exemplo, o método de geração de colunas, em que a cada iteração uma nova mochila deve ser resolvida.

No próximo capítulo definiremos o problema de corte de estoque envolvendo mochilas compartimentadas e faremos algumas considerações sobre o mesmo.

Capítulo 6. O Problema de Corte de Estoque

Neste capítulo, descrevemos um problema de corte de estoque envolvendo mochilas compartimentadas encontrado em algumas indústrias que lidam com o corte de bobinas de aço sujeitas a um processo técnico de laminação (veja capítulo 3). Analisamos o caso em que vários tipos de bobinas estão disponíveis e fazemos algumas considerações relevantes para um estudo futuro.

6.1 Considerações Iniciais

Em visitas a indústrias que trabalham com corte de bobinas de aço sujeito a um processo técnico de laminação, onde surge o Problema da Mochila Compartimentada, deparamo-nos com um problema no qual os itens (ou fitas, como são tratados na indústria) possuem demandas a serem cumpridas, ou seja, todos os itens devem ser produzidos em quantidades determinadas (d_i). Estas demandas são provenientes de pedidos de clientes e devem obedecer a uma série de especificidades como, por exemplo, tipo de aço, largura, espessura (que irá determinar ou não o processo de laminação), entre outras, como visto no capítulo 3. Neste caso, um único padrão de corte não é suficiente, de modo que teremos um problema no qual várias bobinas (note que bobinas agora substituem mochilas) devem ser cortadas e um objetivo global deve ser adotado, como por exemplo, a menor perda de material. Este problema é conhecido na literatura como problema de corte de estoque e tem sido muito estudado desde os trabalhos pioneiros de Gilmore e Gomory (1961, 1963, 1965).

6.2 O Corte de Bobinas de Aço com Vários Tipos de Bobinas em Estoque

Considerando o problema de corte de estoque apresentado na seção anterior, podemos modelá-lo levando em conta os seguintes dados:

- B : número de tipos de bobinas em estoque;
- L_b : largura da bobina b , $b = 1, \dots, B$;
- e_b : disponibilidade em estoque da bobina b , $b = 1, \dots, B$;
- P_b : número de padrões de corte associados a bobina b , $b = 1, \dots, B$;
- c_{pb} : custo da bobina b , segundo o padrão de corte p , $p = 1, \dots, P_b$, $b = 1, \dots, B$.

Os demais dados utilizados já foram introduzidos nos capítulos anteriores.

A figura 6.1 ilustra o problema de corte de estoque com vários tipos de bobinas em estoque.

Estoque:	Demanda:
Bobina 1 $L_1 = 1000$ $e_1 = 10$	Agrupamento 1: $L'_{min} = 154$ $L'_{max} = 456$ $S_1 = 8$ <div> Item 1 $l_1 = 40$ $d_1 = 15$ </div> <div> Item 2 $l_2 = 70$ $d_2 = 12$ </div> <div> Item 3 $l_3 = 90$ $d_3 = 23$ </div>
Agrupamento 2: $L'_{min} = 154$ $L'_{max} = 456$ $S_2 = 8$ <div> Item 4 $l_4 = 50$ $d_4 = 51$ </div> <div> Item 5 $l_5 = 95$ $d_5 = 27$ </div>
Agrupamento 3: $L^3_{min} = 0$ $L^3_{max} = L_b$ $S_3 = 0$ <div> Item 6 $l_6 = 80$ $d_6 = 11$ </div> <div> Item 7 $l_7 = 120$ $d_7 = 30$ </div>

Figura 6.1: Problema de corte de estoque com vários tipos de bobinas em estoque.

Considerando os dados na figura 6.1, podemos definir uma grande quantidade de padrões de corte factíveis obedecendo as restrições impostas pelo modelo matemático para o Problema da Mochila Compartimentada (15.1–15.8) apresentado no capítulo 3. A figura 6.2 exhibe alguns padrões que poderiam ser obtidos.

Figura 6.2: Exemplos de padrões de corte factíveis para o problema de corte de estoque com vários tipos de bobinas em estoque.

Observe que c_{pb} pode representar a perda de um padrão p associado à uma bobina b :

$$c_{pb} = L_b - \sum_{k=1}^K \sum_{j=1}^{N_k} L_{jk}^{pb} \beta_{jk}^{pb} \quad (\text{em que } \sum_{k=1}^K \sum_{j=1}^{N_k} L_{jk}^{pb} \beta_{jk}^{pb} \text{ representa a capacidade utilizada no padrão } p$$

associado à bobina b). Outra opção seria considerar o custo da bobina b (que independe do padrão de corte). Neste caso, teríamos $c_{pb} = c_b$. Note que adicionamos os índices p e b aos dados apresentados no capítulo 3 relacionando-os ao padrão p associado à bobina b . A seguir, mostramos um modelo conhecido, e bastante explorado na literatura, para o problema de corte de estoque em que múltiplos tipos de bobinas em quantidades limitadas são considerados. A função objetivo apresentada neste modelo busca minimizar o número de bobinas a serem cortadas.

Variáveis de Decisão:

- x_{pb} : número de vezes que a bobina do tipo b é cortada usando o padrão p , $p = 1, \dots, P_b$, $b = 1, \dots, B$.

Temos, então, o seguinte modelo:

$$\begin{aligned}
 &\text{Minimizar } \sum_{p=1}^{P_1} c_{p1} x_{p1} + \dots + \sum_{p=1}^{P_B} c_{pB} x_{pB} \\
 &\text{Sujeto a: } \sum_{p=1}^{P_1} a_{p1} x_{p1} + \dots + \sum_{p=1}^{P_B} a_{pB} x_{pB} = \mathbf{d} \\
 &\quad \sum_{p=1}^{P_1} x_{p1} \leq e_1 \\
 &\quad \quad \quad \sum_{p=1}^{P_2} x_{p2} \leq e_2 \\
 &\quad \quad \quad \vdots \\
 &\quad \quad \quad \sum_{p=1}^{P_B} x_{pB} \leq e_B \\
 &x_{pb} \geq 0 \text{ e inteiro, } p = 1, \dots, P_b, b = 1, \dots, B,
 \end{aligned} \tag{34}$$

em que $\mathbf{d}$ é o vetor de demandas dos itens. Tal vetor pode estar associado aos valores de d_i , $i=1, \dots, m$ do problema (15.1–15.8).

A condição de integralidade sobre as variáveis x_{pb} torna o problema (34) difícil, senão impossível, de ser resolvido computacionalmente, para problemas de dimensões moderadas. Uma abordagem prática para resolver o problema (34) consiste em relaxar esta condição de integralidade da variável x_{pb} e resolver o problema relaxado (problema de otimização linear) pelo método Simplex, utilizando o processo de geração de colunas, proposto por Gilmore e Gomory (1961), de modo que não seja preciso gerar todas as colunas antecipadamente.

6.3 Definição de uma Função Custo para o Problema de Corte de Estoque Envolvendo Mochilas Compartmentadas

A função objetivo (15.1), apresentada no capítulo 3, foi modelada de forma genérica para o Problema da Mochila Compartmentada. Com o estudo do problema de corte de bobinas de aço sujeitas ao processo de laminação e a possibilidade de termos vários tipos de bobinas em estoque associados ao problema de corte de estoque (34), modelado na seção anterior, surgiu a necessidade de detalharmos uma função objetivo que representasse melhor o problema. A seguir, fazemos algumas considerações sobre essa função objetivo.

Como vimos no capítulo 3, o processo de corte considera que uma bobina mestre pode ser cortada em algumas bobinas intermediárias (desde que tenham as mesmas exigências quanto ao tipo de aço), as quais podem ser laminadas para diferentes espessuras de material.

Lembre-se que as fitas cortadas de uma mesma bobina intermediária devem ter a mesma espessura, o que justifica o processo de laminação ser efetuado nas bobinas intermediárias. Uma ilustração deste processo pode ser vista na figura 6.3, em que uma mesma bobina mestre é cortada em três bobinas intermediárias que serão recortadas dando origem a fitas de três agrupamentos distintos (com espessuras t_1, t_2, t_3).

Figura 6.3: Laminação das bobinas intermediárias de uma mesma bobina mestre.

Considerando os seguintes dados adicionais:

- T_b : espessura da bobina em estoque, $b = 1, \dots, B$.
- t_k : espessura exigida para as fitas do agrupamento k , $k = 1, \dots, K$.
- R_{kb} : fator de redução da espessura T_b da bobina mestre b , pelo processo de laminação, para a espessura t_k . Este fator pode ser dado por: $R_{kb} = \frac{T_b}{t_k}$.
- η_{kb} : número de ciclos de laminação necessários para reduzir a espessura de T_b para t_k .
- φ : custo por unidade de largura do aço (\$/mm).
- σ : custo fixo pela execução de um ciclo de laminação.

Definidos estes dados, se considerarmos que num único ciclo de laminação podemos reduzir a espessura da bobina mestre em até 20%, é possível traçar uma relação entre o fator de redução R_{kb} e o número de ciclos de laminação η_{kb} pelo seguinte gráfico:

Figura 6.4: Relação $\eta_{kb} \times R_{kb}$

Por sua vez, considerando o custo de um ciclo de laminação σ , podemos ter uma nova relação, desta vez entre o custo de laminação e η_{kb} , representada por:

Figura 6.5: Relação custo de laminação $\times \eta_{kb}$

Feito isso, resta-nos, agora definir a nova função objetivo para o problema de corte de bobinas de aço sujeito a laminação. É importante frisar que esta função tem importante significado no método simplex com geração de colunas, pois o custo relativo é definido por:

$$c^* = c_j - \pi^T a_j \quad (35),$$

em que, pelo modelo (15.1–15.8),

$\mathbf{a}_j = (\alpha_1, \alpha_2, \dots, \alpha_m)$, com $\alpha_i = \sum_{k=1}^K \sum_{i \in A_k} \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk}$, $i = 1, \dots, m$ (36) e π^T representa o vetor multiplicador simplex.

$$\text{Temos, então, } c^* = c_j - \sum_{k=1}^K \sum_{i \in A_k} \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk} \pi_i \quad (37).$$

Como c_j é o custo da coluna a entrar na base, ou seja, o custo do padrão gerado segundo o problema de corte de bobinas de aço, este pode ser definido da seguinte forma:

$$c_j = (L_b - \sum_{k=1}^K (\sum_{i \in A_k} \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk}) l_i) \varphi + (\sum_{k=1}^K \sum_{j=1}^{N_k} \eta_{kb} \beta_{jk}) \sigma \quad (38),$$

com a bobina b determinada, onde $(L_b - \sum_{k=1}^K (\sum_{i \in A_k} \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk}) l_i) \varphi$ representa o custo relacionado a perda de material e $(\sum_{k=1}^K \sum_{j=1}^{N_k} \eta_{kb} \beta_{jk}) \sigma$ representa o custo envolvido no processo de laminação.

Substituindo (38) em (37), temos:

$$c^* = (L_b - \sum_{k=1}^K (\sum_{i \in A_k} \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk}) l_i) \varphi + (\sum_{k=1}^K \sum_{j=1}^{N_k} \eta_{kb} \beta_{jk}) \sigma - \sum_{k=1}^K \sum_{i \in A_k} \sum_{j=1}^{N_k} \alpha_{ijk} \beta_{jk} \pi_i \quad (39),$$

ou seja,

$$c^* = L_b - (\sum_{k=1}^K \sum_{j=1}^{N_k} (\sum_{i \in A_k} (l_i \varphi + \pi_i) \alpha_{ijk} - \eta_{kb} \sigma) \beta_{jk}) \quad (40)$$

Como o método simplex busca minimizar c^* (custo relativo), temos, então, um problema a ser resolvido dado por:

$$\text{Maximizar } \sum_{k=1}^K \sum_{j=1}^{N_k} (\sum_{i \in A_k} (l_i \varphi + \pi_i) \alpha_{ijk} - \eta_{kb} \sigma) \beta_{jk} \quad (41),$$

quando fixamos uma bobina b a ser cortada. Sendo assim, podemos comparar (41) com a função objetivo (15.1): Maximizar $\sum_{k=1}^K \sum_{j=1}^{N_k} (V_{jk} - c_k) \beta_{jk}$, proposta para o Problema da Mochila Compartimentada, no capítulo 3. Teríamos, então, as seguintes relações:

$$V_{jk} = \sum_{i \in A_k} (l_i \varphi + \pi_i) \alpha_{ijk} \quad \text{e} \quad c_k = \eta_{kb} \sigma \quad (42)$$

Com isso, podemos ver como os coeficientes da função objetivo para o Problema da Mochila Compartimentada (gerador de colunas) pode ser definida, considerando-se os custos de um padrão de corte compartimentado como uma soma dos custos decorrentes da perda e da laminação.

Capítulo 7. Conclusões e Trabalhos Futuros

Neste trabalho foi estudado o Problema da Mochila Compartimentada, um problema de otimização combinatória, que é uma variação do clássico problema da mochila, porém, sua formulação matemática sofre profundas mudanças. Tal problema foi definido a partir de uma importante aplicação prática na indústria metalúrgica, que surge no processo de corte de bobinas de aço, as quais estão sujeitas a um processo laminação, que faz com que os modelos tradicionais não se apliquem. Poucos trabalhos foram publicados e quase todos evitam formalizar o Problema da Mochila Compartimentada, tratando-o de forma implícita por métodos heurísticos simples (Ferreira et al., 1990), (Carvalho, 1991) e (Carvalho e Rodrigues, 1994). Mais recentemente, Marques (2000), Hoto (2001), Hoto et al. (2002, 2003) e Marques e Arenales (2002) têm a abordagem desta tese, seja considerando problemas mais simples (por exemplo, caso irrestrito, para o qual um método exato é viável) ou métodos heurísticos mais simples. Nesta tese estendemos estes trabalhos.

Um modelo matemático de otimização não linear inteiro foi proposto e apesar de algumas dificuldades impostas pela não linearidade do problema, desenvolvemos vários métodos heurísticos para a sua resolução. Nesses métodos surgiram a necessidade de resolver vários problemas da mochila, com pequenas variações, para os quais propusemos modificações no algoritmo de enumeração implícita de Gilmore e Gomory (1963), cujos detalhes são encontrados no apêndice A. Os métodos foram implementados, testados e comparados entre si, bem como avaliados a partir de limitantes para o problema, de modo que se pode afirmar sobre a otimalidade ou quase-otimalidade de um número de exemplares resolvidos. A partir destes testes, pudemos comprovar a eficiência dos métodos propostos, seja com relação à qualidade da solução obtida, como também em relação ao tempo computacional, uma vez que para problemas de corte de estoque (em que uma demanda deve ser atendida) muitos problemas da mochila compartimentada devem ser resolvidos (por exemplo, geração de colunas no métodos simplex).

A implementação destas heurísticas envolveu aspectos computacionais, como a utilização de estruturas de dados encadeadas e alocadas dinamicamente, além de técnicas de engenharia de software que pudessem melhor explorar as principais características dos algoritmos utilizados. Tais aspectos são descritos no apêndice B.

A continuação deste trabalho envolve naturalmente a implementação do método Simplex com geração de colunas, para resolver o problema de corte de estoque compartimentado (introduzido no capítulo 6), utilizando-se as heurísticas propostas para a geração de colunas. Também uma questão computacional surge na representação da matriz de padrões gerados. Tal matriz necessita de uma estrutura que possa explorar as características de padrões compartimentados, de modo que as informações a respeito dos arranjos dos itens nos compartimentos da mochila sejam facilmente recuperadas, diferentemente dos problemas de corte unidimensionais em que a recuperação do padrão de corte é simples. Tal estrutura necessita ser melhor avaliada, o que será feito a medida que o método Simplex com geração de colunas for implementado. Outro ponto relevante a ser investigado, diz respeito à baixa demanda, algo bastante comum na prática, uma vez que uma bobina mestre é bastante pesada e o corte de uma única tira pode exceder a demanda. Múltiplos padrões de corte associados a uma mesma bobina mestre (isto é, cortes transversais dividiriam a bobina mestre em várias sub-bobinas que seriam finalmente cortadas em padrões compartimentados) envolveria uma mudança de processo na prática e um estudo de viabilização deve ser aprofundado.

Novas aplicações para o Problema da Mochila Compartimentada parecem ser naturais. Dentre essas aplicações, podemos apresentar um problema que surge na indústria de papel, em que muitos itens são retângulos e as bobinas jumbos são cortadas em dois estágios: num primeiro estágio obtém-se sub-bobinas (compartimentos, como no corte de bobinas de aço apresentado no capítulo 3), para então serem cortadas nas larguras dos itens de mesmo comprimento (o processo de corte transversal é obtido por facas num cilindro que corta as bobinas menores em retângulos), gerando padrões de corte compartimentados conforme vimos no capítulo 3, ou seja, itens de mesmo comprimento pertencem a um mesmo agrupamento.

Podemos ter também casos bidimensionais envolvendo situações como, por exemplo, o problema de corte de placas de circuitos impressos (os compartimentos são sub-retângulos menores da placa original, sobre os quais incidem os mesmos processos, como banhos especiais) e até problemas tridimensionais, como no caso de empacotamento de caixas que devem estar agrupadas (compartimentadas) por cliente.

Em resumo, a contribuição desta tese foi orientada para o caso unidimensional, motivado por aplicações práticas de difícil solução, e abre possibilidades de identificar novos problemas que apresentem estruturas semelhantes e possam se beneficiar das idéias aqui desenvolvidas.

Referências Bibliográficas

ARENALES, M.; MORABITO, R.; YANASSE H. (1999) Cutting and packing problems. *Pesquisa Operacional*, v.19, n.2, p.107-299.

BEASLEY, J.E. (1985) Algorithms for unconstrained two-dimensional guillotine cutting. *Journal of the Operational Research Society*, v.36, n.4, p.297-306.

BISCHOFF, E.; WÄSCHER, G. (Eds.) (1995) Cutting and packing. *European Journal of Operational Research*, v.84, n.3, special issue.

BROWN, A. R. (1971) *Optimum Packing and Depletion*. Macdonald - London and American Elsevier Inc. - New York.

CARVALHO, J.M.V. (1991) *Um Problema de Corte em Duas Fases*. Tese de Doutorado, Universidade do Minho, Portugal.

CARVALHO, J.M.V.; RODRIGUES, A.J.G. (1994) A Computer Based Interactive Approach to a Two-stage Cutting Stock Problem. *INFOR*, v.32, n.4, p.243-252.

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

CORREIA, M. H.; OLIVEIRA, J.F.; FERREIRA, J.S. (2004) Reel and Sheet Cutting at a Paper Mill. *Computers and Operations Research*, v.31, p.1223-1243.

DOWSLAND, K.; DOWSLAND, W. (1992) Packing problems. *European Journal of Operational Research*, v.56, p.2-14.

DYCKHOFF, H.; ABEL, D. and GAL, T. (1985) Trim Loss and Related Problems. *Omega*, v.13, p.59-72.

- DYCKHOFF, H. (1990) A typology of cutting and packing problems. *European Journal of Operational Research*, v.44, p.145-159.
- DYCKHOFF, H.; FINKE, U. (1992) *Cutting and packing in production and distribution: Typology and bibliography*. Heidelberg : Springer-Verlag.
- DYCKHOFF, H.; SCHEITHAUER, G.; TERNO, J. (1997) *Cutting and packing*. In: AMICO, M.; MAFFIOLI, F.; MARTELLO, S. (Ed.) *An noted bibliographies in combinatorial optimization*. New York: John Wiley & Sons. p.393-414.
- DYCKHOFF, H.; WÄSCHER, G. (Eds.) (1990) Cutting and packing. *European Journal of Operational Research*, v.44, n.2, special issue.
- FERREIRA, J.S.; NEVES, M.A.; Castro, P.F. (1990) A Two-phase Roll Cutting Problem. *European Journal of Operational Research*, V.44:185-196.
- GAREY, M.R.; JOHSON, D.S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., San Francisco, 1979.
- GILMORE, P.; GOMORY, R. (1961) A linear programming approach to the cutting stock problem. *Operations Research*, v.9, p.849-859.
- GILMORE, P.; GOMORY, R. (1963) A linear programming approach to the cutting-stock problem II. *Operations Research*, v.11, p.863-888.
- GILMORE, P.; GOMORY, R. (1965) Multistage cutting stock problems of two and more dimensions. *Operations Research*, v.14, p.94-120.
- GRAMANI, M.C.N. (1997) *O problema de corte bidimensional guilhotinado em 2-estágios e restrito*. 100f. Dissertação de Mestrado, ICMC, Universidade de São Paulo, São Carlos.
- HAESSLER, R. W. (1974) Controlling Cutting Pattern Changes in One-dimensional Trim Problems. *Operations Research*, v.23, n.3, p.483-493.
- HAESSLER, R. W. (1979) A note on computational modifications to the Gilmore-Gomory cutting stock algorithm. *Operations Research*, v.28, n.4, p.1001-1005.
- HERZ, J.C. (1972) A recursive computing procedure for two-dimensional stock cutting. *IBM Journal of Research and Development*, v.16, p.462-469.

- HINXMAN, A. (1980) The trim-loss and assortment problems: a survey. *European Journal of Operational Research*, v.5, p.8-18.
- HOTO, R.S.V. (1996) *Otimização no Corte de Peças Unidimensionais com Restrições de Agrupamento*, Dissertação de Mestrado, ICMSC-USP, São Carlos, São Paulo.
- HOTO, R.S.V.; ARENALES, M.N. (1996) Um Problema de Corte Unidimensional com Restrições de Agrupamento e Aplicações Industriais, *I Oficina Nacional em Problemas de Corte e Empacotamento*, São Paulo, p.31-36.
- HOTO, R.S.V. (2001) *O problema da mochila compartimentada aplicado no corte de bobinas de aço*. Tese de Doutorado, COPPE, Universidade Federal do Rio de Janeiro.
- HOTO, R.S.V; MACULAN, N.; ARENALES, M.N.; MARQUES, F.P (2002) Um novo procedimento para o cálculo de mochilas compartimentadas. *Investigação Operacional*, v.22, p.213-234.
- HOTO, R.S.V; MACULAN, N.; MARQUES, F.P; ARENALES, M.N. (2003) Um problema de corte com padrões compartimentados. *Pesquisa Operacional*, v.23, n.1, p.169-187.
- JOHNSTON, R.E.; KHAN, L.R. (1995) Bounds for nested knapsack problem. *European Journal of Operational Research*, v.81, p.154-165.
- LIN, E.Y.H. (1998) A Bibliographical Survey on Some Well-Known Non-Standard Knapsack Problems. *INFOR*, v.36, n.4, p.274-317.
- LODI, A.; MARTELLO, S.; MONACI, M. (2002) Two-dimensional packing problems: A survey. *European Journal of Operational Research*, v.141, p.241-252.
- MARQUES, F.P. (2000) *O Problema da Mochila Compartimentada*. 106f. Dissertação de Mestrado, ICMC, Universidade de São Paulo, São Carlos.
- MARQUES, F.P.; ARENALES, M. (2002). O problema da mochila compartimentada e aplicações. *Pesquisa Operacional*, v.22, n.3, p. 285-304.
- MARTELLO, S.; TOTH, P. (1990) *Knapsack problems: algorithms and computer implementations*. Chichester : John Wiley & Sons.

- MORABITO, R. (1992) *Uma abordagem em grafo e-ou para o problema de empacotamento: aplicação ao carregamento de paletes e contêineres*. 212f. Tese de Doutorado, FEESC, Universidade de São Paulo, São Carlos.
- MORABITO, R.; ARENALES, M. (1992) Um exame dos problemas de corte e empacotamento. *Pesquisa Operacional*, v.12, n.1, p.1-20.
- MORABITO, R.; ARENALES, M.N. (1995) Performance of two heuristics for solving large scale two-dimensional guillotine cutting problems. *INFOR*, v.33, n.2, p.145-155.
- MORABITO, R.; GARCIA, V. (1998) The cutting stock problem in a hardboard industry: a case study. *Computers and Operations Research*, v.25, n.6, p.469-485.
- PEREIRA, M.A. (1993) *Uma Abordagem Matemática para o Problema de Corte e Laminação de Fitas de Aço*, Dissertação de Mestrado, UNICAMP, Campinas, São Paulo.
- PILEGGI, G. (2002) *Abordagens para otimização integrada dos problemas de geração e seqüenciamento de padrões de corte*. Tese de Doutorado, ICMC, Universidade de São Paulo, São Carlos.
- PRESSMAN, R. S. (1995) Engenharia de Software. MAKRON Books do Brasil Editora Ltda, São Paulo, Brasil.
- SWEENEY, P.; PATERNOSTER, E. (1992) Cutting and packing problems: a categorized, application-oriented research bibliography. *Journal of the Operational Research Society*, v.43, p.691-706.
- YANASSE, H.H.; ZINOBER, A.S.I.; HARRIS, R.G. (1990) A Heuristic Procedure for a Two-dimensional Guillotine Cutting Stock Problem, *Paper presented at INFOR*, Athens, Grece.
- YANASSE, H.H.; SOMA, N.Y.; MACULAN, N. (2000) An algorithm for determining the k-best solutions of one-dimensional knapsack problem. *Pesquisa Operacional*, v.20, n.1, p.117-134.
- WANG, P.Y. (1983) Two algorithms for constrained two-dimensional cutting stock problems. *Operations Research*, v.31, n.3, p.573-586.
- WANG, P.Y.; WÄSCHER, G. (2002) Cutting and packing. *European Journal of Operational Research*, v.141, n.2, p.239-469.

WÄSCHER, G.; GAU, T. (1996) Heuristics for the integer one-dimensional cutting stock problem: a computational study. *Operations Research Spectrum*, v.18, p.131-144.

ZAK, E. (2002) Row and column generation technique for a multistage cutting stock problem. *Operations Research*, v.29, p.1143-1156.

Apêndice A. Algoritmos de Enumeração Implícita

Neste apêndice descrevemos os algoritmos utilizados neste trabalho para auxiliar na resolução do Problema da Mochila Compartimentada. Lembramos que os algoritmos aqui apresentados são modificações que propomos para o método de enumeração implícita de Gilmore e Gomory (1963), com o objetivo de resolver problemas específicos que surgem no Problema da Mochila Compartimentada. A seguir, apresentamos as três modificações propostas: resolução do problema da mochila restrito, resolução do problema da mochila obtendo as “z” melhores soluções e resolução de um problema de otimização restrito com limitações no número de compartimentos, sem violação no limite da quantidade de itens na mochila. Este último algoritmo é aplicado na resolução do problema (31.1–31.5) que surge nas heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades.

A.1. Considerações Gerais

O método de enumeração implícita proposto por Gilmore e Gomory (1963) mostrado no capítulo 2 resolve eficientemente o problema da mochila irrestrito, nos tamanhos que ocorrem no contexto dos problemas de corte, isto é, com algumas dezenas de itens. A implementação usa a estratégia *backtracking* de modo que o espaço de soluções é percorrido de maneira bem simples, usando basicamente um vetor solução $\mathbf{X} = (x_1 \ x_2 \dots x_k \ 0 \dots 0)$, o qual tem o número de coordenadas igual ao número de itens da mochila, sendo x_i o número de itens do tipo i , e k o último item incluído na mochila ($x_k > 0$). O vetor solução é suficiente para caracterizar um nó na árvore de busca, de modo que o nó anterior é caracterizado por $\mathbf{X}' = (x_1 \ x_2 \dots x_{k-1} \ 0 \dots 0)$. Regras simples para incluir novos itens na mochila (portanto, novas soluções ou novos nós na árvore de busca) são facilmente derivadas, dada a simplicidade das

restrições da mochila, bem como limitantes superiores. Sempre que uma solução mais valiosa é identificada, seu valor é armazenado, descartando-se a anterior. Entretanto, os problemas da mochila que necessitamos resolver dentro de cada heurística apresentam características adicionais que podem ser incluídas no processo de busca, são elas: *i)* a limitação do número de itens na mochila, *ii)* a determinação das “z” melhores soluções e, *iii)* restrições adicionais dadas por (31.3). A inclusão de restrições adicionais no processo de busca no método de enumeração implícita de Gilmore e Gomory é bastante utilizada, porém nem sempre publicada. Por exemplo, os autores do método, em seu artigo de 1963, sugeriram que restrições simples, do tipo limitação do número de facas, podem ser incluídas no processo de busca. Morabito e Garcia (1998) também incluíram restrições adicionais no processo de busca quando estudaram um problema prático de corte (originário de uma indústria de madeira reconstituída) de onde surgiram restrições difíceis de serem descritas, mas razoavelmente simples de serem incluídas no processo de busca e anunciaram excelentes resultados.

Antes de seguir, apresentamos um algoritmo que representa o método de Gilmore e Gomory mostrado no capítulo 2. Para facilitar a visualização das modificações propostas, enumeraremos os passos do algoritmo.

Algoritmo de Enumeração Implícita

Passo 1. {Definição do problema segundo as variáveis mais valiosas}

1.1 Defina π_i da seguinte forma:

Para $i = 1, \dots, m$ definimos a razão:

$$\pi_i = \frac{v_i}{l_i}$$

1.2 Definidos os valores $\pi_i, i = 1, \dots, m$, definimos a seguinte ordenação:

$$\pi_1 \geq \pi_2 \geq \dots \geq \pi_m$$

1.3 Defina $\underline{G} = 0$: o melhor valor para o objetivo da mochila encontrado.

Passo 2. {Determinação da solução inicial - busca em profundidade primeiro}

2.1 Determine a solução $\mathbf{X} = (x_1, x_2, \dots, x_m)$:

$$x_1 = \left\lfloor \frac{L}{l_1} \right\rfloor \Rightarrow \text{sobra: } L - l_1 x_1,$$

$$x_2 = \left\lfloor \frac{L - l_1 x_1}{l_2} \right\rfloor \Rightarrow \text{sobra: } L - l_1 x_1 - l_2 x_2,$$

$$\dots x_{i+1} = \left\lfloor \frac{L - \sum_{j=1}^i l_j x_j}{l_{i+1}} \right\rfloor \Rightarrow \text{sobra: } L - l_1 x_1 - l_2 x_2 - \dots - l_{i+1} x_{i+1}.$$

Passo 3. {Avaliação da solução corrente e armazenamento da mais valiosa}

3.1 Determine: $g(\mathbf{X}) = \sum_{i=1}^m v_i x_i$

3.2 Se $\underline{G} < g(\mathbf{X})$ então faça:

$$\underline{G} = g(\mathbf{X})$$

e guarde a solução correspondente: $\mathbf{X}^* = \mathbf{X}$.

Passo 4. {Teste de parada e cálculo de limite superior}

4.1 Determine r o maior índice tal que: $x_r \neq 0$.

4.2 Se $\mathbf{X} = (0, 0, \dots, 0)$ então PARE, a melhor solução guardada em $\mathbf{X}^*$, é uma solução adotada.

$$\bar{G}(\mathbf{X}) = v_1 x_1 + v_2 x_2 + \dots + v_r (x_r - 1) + \frac{v_{r+1}}{l_{r+1}} (L - l_1 x_1 - l_2 x_2 - \dots - l_r (x_r - 1)).$$

Caso contrário, calcule: (se $r = m$, considere $v_{m+1} = 0$ e $l_{m+1} = 1$)

Passo 5. {Backtracking}

5.1 {Retorno longo}

Se $\bar{G}(\mathbf{X}) \leq \underline{G}$, faça:

$x_r = 0$ e volte ao passo 4.

5.2 {Retorno ao nó precedente e nova busca em profundidade}

Se $\bar{G}(\mathbf{X}) > \underline{G}$ então faça:

$x_r = x_r - 1$ e defina a nova solução $\mathbf{X}$:

$$x_{j+1} = \left\lfloor \frac{L - \sum_{i=1}^j l_i x_i}{l_{j+1}} \right\rfloor, j = r, \dots, m-1.$$

volte ao passo 3.

{ Fim do Algoritmo }

Na sequência, damos uma breve descrição de como incluímos as características dos problemas da mochila que surgem nas heurísticas (seções 4.1–4.4).

A.2. O Problema Restrito

Os problemas de mochilas restritas que surgem em (26.1-26.3), (27.1-27.5) e (29.1-29.3) foram resolvidos por uma modificação simples no algoritmo descrito na seção anterior. As variáveis devem obedecer à limitação na quantidade de cada item dentro da mochila. Para isto, o valor atribuído à variável associada ao item i (que consiste numa busca em profundidade na árvore de decisões) é calculado por:

$$x_i = \text{Mínimo} \left\{ d_i, \left\lfloor \frac{L - \sum_{j=1}^i l_j x_j}{l_i} \right\rfloor \right\}, i = 1, \dots, m. \quad (43)$$

onde $L - \sum_{j=1}^i l_j x_j$ é a capacidade ociosa na mochila, atualizada a cada nó percorrido na árvore de busca.

O cálculo do limitante superior (passo 4), importante para a eficiência do método de enumeração implícita sofre também uma ligeira modificação, devido à canalização das variáveis. A modificação é efetuada da seguinte forma:

Passo 4:

4.1.1 Determine r o maior índice tal que: $x_r \neq 0$.

Seja: $\bar{L} = L - l_1 x_1 - \dots - l_r (x_r - 1)$.

(se $r = m$, considere $v_{m+1} = 0$, $l_{m+1} = \bar{L} + 1$ e $d_{m+1} = 1$)

4.1.2 Determine h , tal que:
$$\begin{cases} \sum_{i=r+1}^{h-1} l_i d_i \leq \bar{L} \\ \sum_{i=r+1}^h l_i d_i > \bar{L} \end{cases} \quad (44)$$

Agora, para a solução do problema da mochila restrita, temos:

$$\begin{aligned} x_i &= d_i, i = r+1, \dots, h-1, \text{ e} \\ x_h &= \frac{\bar{L} - \sum_{i=r+1}^{h-1} l_i x_i}{l_h} \end{aligned} \quad (45)$$

Isto faz com que o limitante superior, no passo 4.2, seja definido como:

$$\bar{G}(\mathbf{X}) = v_1 x_1 + \dots + v_r (x_r - 1) + v_{r+1} x_{r+1} + \dots + v_h x_h \quad (46)$$

Por fim, no passo 5, o cálculo da nova solução é feito considerando:

Passo 5:

5.1 {Retorno longo}

Se $\bar{G}(\mathbf{X}) \leq \underline{G}$, faça:

$x_r = 0$ e volte ao passo 4.

5.2 {Retorno ao nó precedente e nova busca em profundidade}

Se $\bar{G}(\mathbf{X}) > \underline{G}$ então faça:

$x_r = x_r - 1$ e defina a nova solução $\mathbf{X}$:

$$x_{j+1} = \text{Mínimo} \left\{ d_{j+1}, \left\lfloor \frac{L - \sum_{i=1}^j l_i x_i}{l_{j+1}} \right\rfloor \right\}, \text{ para } j = r, \dots, m-1.$$

volte ao passo 3.

A.3. As “z” Melhores Soluções

Os problemas (30.1-30.3) consistem nas “z” melhores soluções de um problema da mochila. Para determinar tais soluções, basta armazenar no processo de busca as “z” melhores soluções, ao invés de armazenar apenas a melhor, como é feito no algoritmo da seção 1 deste apêndice. Assim, sempre que uma nova solução é determinada, o correspondente valor da função objetivo, digamos V , é comparado com os “z” valores previamente armazenados (por facilidade estão ordenados, por exemplo: $V_1 \geq V_2 \geq \dots \geq V_z$), sendo rejeitado caso $V \leq V_z$, ou V_z é descartado e o valor de V armazenado ordenadamente. Observe que a nova solução gerada difere das anteriores, pois o processo de busca do algoritmo de Gilmore e Gomory evita repetições de soluções. Um método alternativo pseudo-polinomial para o cálculo das melhores soluções do problema da mochila pode ser encontrado em Yanasse et al. (2000).

Para implementar tais modificações, propomos a substituição do vetor de soluções $\mathbf{X}$ por uma matriz $\mathbf{X}$, onde a j -ésima linha da matriz corresponde à j -ésima melhor solução. Vale observar que o vetor $\mathbf{V}$, descrito no parágrafo anterior deverá ser criado, pois não existe na versão original. O algoritmo modificado é descrito a seguir.

Algoritmo de Enumeração Implícita para as “z” Melhores Soluções

Passo 1. {Definição do problema segundo os itens mais valiosos do agrupamento k }

1.1 Defina π_i da seguinte forma:

Para $i = 1, \dots, |A_k|$ definimos a razão:

$$\pi_i = \frac{v_i}{l_i}$$

1.2 Definidos os valores π_i , $i = 1, \dots, |A_k|$, definimos a seguinte ordenação:

$$\pi_1 \geq \pi_2 \geq \dots \geq \pi_{|A_k|}$$

Passo 2. {Determinação das “z” soluções iniciais - busca em profundidade primeiro}

2.1 Determine as “z” soluções dadas pelos vetores $(\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_z)$ da matriz $\mathbf{X}$ da seguinte forma:

Para $j = 1, \dots, z$ definimos as soluções:

2.1.1 Determine o vetor solução $\mathbf{X}_j = (x_1, x_2, \dots, x_{|A_k|})$ e seu respectivo valor V_j :

Para $i = j, \dots, |A_k|$, definimos:

Início

$$x_{ji} = \min \left\{ d_i, \left\lfloor \frac{\bar{L}}{l_i} \right\rfloor \right\}.$$

$$\bar{L} = L - \sum_{t=j}^{i-1} l_t x_t,$$

Fim.

$$V_j = \sum_{t=j}^m v_t x_t.$$

Neste caso, ao final do passo 2, teremos uma matriz de soluções da seguinte forma:

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1|A_k|} \\ 0 & x_{22} & \dots & x_{2|A_k|} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & x_{z|A_k|} \end{bmatrix},$$

com vetor $\mathbf{V}$ contendo os respectivos valores para as funções objetivos:

$$\mathbf{V}^T = [V_1 \quad V_2 \quad \dots \quad V_z].$$

2.1.2 Ordenamos as linhas de $\mathbf{X}$ segundo os valores de V_j , $j = 1, \dots, z$, de seguinte forma:

$$V_1 \geq V_2 \geq \dots \geq V_z.$$

Passo 3. {Definição e avaliação da solução corrente e armazenamento da mais valiosa}

3.1 Defina $\mathbf{X}_{Atual} = \mathbf{X}_1$

3.2 Determine: $g(\mathbf{X}_{Atual}) = \sum_{i=1}^{|A_k|} v_i x_i$

3.3 Se $\underline{G}(\mathbf{X}) < g(\mathbf{X}_{Atual})$ (considere inicialmente $\underline{G}(\mathbf{X}) = V_z$) então faça:

3.3.1 Guarde a solução correspondente $\mathbf{X}_{Atual}$ na posição da j -ésima linha da matriz $\mathbf{X}$ e V_j na j -ésima posição do vetor $\mathbf{V}$ de modo que este continue ordenado.

3.3.2 $\underline{G}(\mathbf{X}) = V_z$

Passo 4: {Teste de parada e cálculo de limite superior}

4.1 Em $\mathbf{X}_{Atual}$, determine r o maior índice tal que: $x_r \neq 0$.

Seja: $\bar{L} = L - l_1 x_1 - \dots - l_r (x_r - 1)$.

(se $r = |A_k|$, considere $v_{|A_k|+1} = 0$, $l_{|A_k|+1} = \bar{L} + 1$ e $d_{|A_k|+1} = 1$)

4.2 Em $\mathbf{X}_{Atual}$, determine h , tal que:

$$\begin{cases} \sum_{i=r+1}^{h-1} l_i d_i \leq \bar{L} \\ \sum_{i=r+1}^h l_i d_i > \bar{L} \end{cases}$$

4.3 Se $\mathbf{X}_{Atual} = (0, 0, \dots, 0)$ então PARE, a melhor solução guardada em $\mathbf{X}$, é a solução adotada.

Caso contrário, calcule o limitante superior:

$$\bar{G}(\mathbf{X}) = v_1 x_1 + \dots + v_r (x_r - 1) + v_{r+1} x_{r+1} + \dots + v_h x_h.$$

Passo 5. {Backtracking}

5.1 {Retorno longo}

Se $\bar{G}(\mathbf{X}) \leq \underline{G}(\mathbf{X})$, faça:

$x_r = 0$ e volte ao passo 4.

5.2 {Retorno ao nó precedente e nova busca em profundidade}

Se $\overline{G}(\mathbf{X}) > \underline{G}(\mathbf{X})$ então faça:

$$x_{j+1} = \text{Mínimo} \left\{ d_{j+1}, \left\lfloor \frac{L - \sum_{i=1}^j l_i x_i}{l_{j+1}} \right\rfloor \right\}, \text{ para } j = r, \dots, |A_k| - 1.$$

$x_r = x_r - 1$ e defina a nova solução $\mathbf{X}_{Atual}$:

volte ao passo 3.

{ Fim do Algoritmo }

A.4. O Problema (31.1-31.5)

Dentre os subproblemas que surgem nas heurísticas, o problema (31.1-31.5) é o mais complexo, pois várias restrições (31.3) são adicionadas à restrição da mochila (31.2). Procedemos analogamente ao caso restrito, introduzindo uma modificação no processo de busca do método de enumeração implícita de Gilmore e Gomory (1963), de modo a garantir a satisfação das restrições (31.3) e (31.4). Como tratamos com superitens (isto é, compartimentos que armazenam itens de um mesmo agrupamento) e considerando que temos “z” superitens associados a um mesmo agrupamento, surge a necessidade de um maior cuidado durante a busca para que as quantidades máximas d_i , $i=1, \dots, m$, não sejam violadas (veja restrições (31.3)). Para evitar tal violação é necessário primeiramente limitar a inclusão do superitem (j, k) , $j = 1, \dots, z$ (ou w) e $k = 1, \dots, K-1$ (j -ésimo compartimento associado ao agrupamento k que pode ser obtido pelo problema (30.1-30.3) no caso da Heurística dos “z” Melhores Compartimentos ou pelo problema (32.1- 32.3) no caso da Heurística do Melhor Compartimento para “w” Capacidades) por:

$$\beta_{jk} \leq \text{Mínimo} \left\{ \left\lfloor \frac{d_i}{\alpha_{ijk}} \right\rfloor, \text{ para todo } i \in A_k \right\}. \quad (47)$$

Isto reduz ao caso restrito já discutido, mas não é suficiente. Por exemplo, se $\beta_{1k} > 0$ e $\beta_{2k} > 0$ (isto é, o primeiro e segundo compartimentos associados ao agrupamento k são

carregados na mochila) e, além disso, $\alpha_{i1k} > 0$ e $\alpha_{i2k} > 0$ (isto é, o item i aparece nos compartimentos 1 e 2) então o total de itens do tipo i já incluídos na mochila é dado por: $\alpha_{i1k} \beta_{1k} + \alpha_{i2k} \beta_{2k}$ (veja restrição (31.3)). Portanto, a quantidade de superitem do tipo 3, por exemplo, é limitada por:

$$\beta_{3k} \leq \left\lfloor \frac{\overline{d_i}}{\alpha_{i3k}} \right\rfloor, \quad (48)$$

onde $\overline{d_i} = d_i - \alpha_{i1k} \beta_{1k} - \alpha_{i2k} \beta_{2k}$ é a folga, para o item $i \in A_k$, da restrição (31.3). A busca é similar ao caso anterior, porém considerando tais limitações para o número de compartimentos.

Sabe-se que após o passo 1 das heurísticas dos “z” Melhores Compartimentos e dos Melhores Compartimentos para “w” Capacidades (veja capítulo 4), temos definidos os valores de V_{jk} , L_{jk} , α_{ijk} , para $i \in A_k$, $k = 1, \dots, K-1$ e $j = 1, \dots, N_k$, além dos dados originais: v_i e l_i , $i \in A_k$, $k = K$, d_i , $i = 1, \dots, m$.

Para apresentar o algoritmo, faremos as seguintes redefinições:

- $q = \begin{cases} z, \text{Heurística dos “z” Melhores Compartimentos} \\ w, \text{Heurística dos Melhores Compartimentos para “w” Capacidades} \end{cases}$
- $T = ((K-1)*q) + |A_K|$, ou seja, número total de compartimentos, somado ao número de itens livres.

Algoritmo de Enumeração Implícita para o Problema (31.1-31.5)

Os procedimentos de atualização (A), (B) e (C), que aparecem no algoritmo, são detalhados no final. As atualizações surgem após incluir um novo item (descer na árvore) ou retirar uma ou mais unidades de um item (retorno).

Passo 1. {Definição do problema segundo os itens mais valiosos}

1.1 Defina valores auxiliares $\hat{\pi}_i$, $\hat{l}_i$, $\hat{v}_i$ e $\hat{d}_i$ da seguinte forma:

$t = 1$

Para $k = 1, \dots, (K-1)$ faça:

Para $j = 1, \dots, N_k$ faça:

Início

$$\begin{aligned}\hat{\pi}_t &= \frac{V_{jk} - c_k}{L_{jk}}, & \hat{l}_t &= L_{jk}, & \hat{v}_t &= V_{jk} - c_k, \\ \hat{d}_t &= \text{Mínimo} \left\{ \left\lfloor \frac{d_t}{\alpha_{ijk}} \right\rfloor, \text{para todo } i \in A_k \right\}, \\ t &= t + 1 \\ \text{Fim}\end{aligned}$$

Para todo *item* $\in A_k$ com $k = K$ faça:

Início

$$\begin{aligned}\hat{\pi}_t &= \frac{v_{item}}{l_{item}}, & \hat{l}_t &= l_{item}, & \hat{v}_t &= v_{item}, \\ \hat{d}_t &= d_{item}, \\ t &= t + 1 \\ \text{Fim}\end{aligned}$$

1.2 Definidos os dados $\hat{\pi}_t, t = 1, \dots, T$, definimos a seguinte ordenação:

$$\hat{\pi}_1 \geq \hat{\pi}_2 \geq \dots \geq \hat{\pi}_T$$

Passo 2. {Determinação da solução inicial - busca em profundidade primeiro}

2.1 Determine a solução $\mathbf{X} = (x_1, x_2, \dots, x_T)$:

$$\begin{aligned}x_1 &= \text{Mínimo} \left\{ \hat{d}_1, \left\lfloor \frac{L}{\hat{l}_1} \right\rfloor \right\} \Rightarrow \text{sobra} : L - \hat{l}_1 x_1, \\ \text{Execute } (A), \\ x_2 &= \text{Mínimo} \left\{ \hat{d}_2, \left\lfloor \frac{L - \hat{l}_1 x_1}{\hat{l}_2} \right\rfloor \right\} \Rightarrow \text{sobra} : L - \hat{l}_1 x_1 - \hat{l}_2 x_2, \\ \text{Execute } (A), \\ \dots \ x_{t+1} &= \text{Mínimo} \left\{ \hat{d}_{t+1}, \left\lfloor \frac{L - \sum_{j=1}^t \hat{l}_j x_j}{\hat{l}_{t+1}} \right\rfloor \right\} \Rightarrow \text{sobra} : L - \hat{l}_1 x_1 - \hat{l}_2 x_2 - \dots - \hat{l}_{t+1} x_{t+1} \text{ e} \\ \text{Execute } (A), & \text{ para } t = 1, \dots, T-1.\end{aligned}$$

Passo 3. {Avaliação da solução corrente e armazenamento da mais valiosa}

$$3.1 \quad \text{Determine: } g(\mathbf{X}) = \sum_{t=1}^T \hat{v}_t x_t$$

3.2 Se $\underline{G} < g(\mathbf{X})$ então faça (considere inicialmente $\underline{G} = 0$) :

$$\underline{G} = g(\mathbf{X})$$

e guarde a solução correspondente: $\mathbf{X}^* = \mathbf{X}$.

Passo 4. {Teste de parada e cálculo de limite superior}

4.1.1 Determine r o maior índice tal que: $x_r \neq 0$.

Seja: $\bar{L} = L - \hat{l}_1 x_1 - \dots - \hat{l}_r (x_r - 1)$.

(se $r = m$, considere $\hat{v}_{r+1} = 0$, $\hat{l}_{r+1} = \bar{L} + 1$ e $\hat{d}_{r+1} = 1$)

4.1.2 *Execute* (C).

4.1.3 Determine h , tal que:
$$\begin{cases} \sum_{i=r+1}^{h-1} \hat{l}_i \hat{d}_i \leq \bar{L} \\ \sum_{i=r+1}^h \hat{l}_i \hat{d}_i > \bar{L} \end{cases}$$

4.2 Se $\mathbf{X} = (0, 0, \dots, 0)$ então PARE, a melhor solução guardada em $\mathbf{X}^*$, é uma solução adotada.

Caso contrário, calcule o limitante superior:

$$\bar{G}(\mathbf{X}) = \hat{v}_1 x_1 + \dots + \hat{v}_r (x_r - 1) + \hat{v}_{r+1} x_{r+1} + \dots + \hat{v}_h x_h.$$

Passo 5. {Backtracking}

5.1 {Retorno longo}

Se $\bar{G}(\mathbf{X}) \leq \underline{G}$, faça:

Execute (B),

$x_r = 0$ e volte ao passo 4.

5.2 {Retorno ao nó precedente e nova busca em profundidade}

Se $\bar{G}(\mathbf{X}) > \underline{G}$ então faça:

$x_r = x_r - 1$ e defina a nova solução $\mathbf{X}$:

$$x_{t+1} = \text{Mínimo} \left\{ \hat{d}_t, \left\lfloor \frac{L - \sum_{j=1}^t \hat{l}_j x_j}{\hat{l}_{t+1}} \right\rfloor \right\} \text{ e } \textit{Execute} (A), \text{ para } t = r, \dots, T-1.$$

volte ao passo 3.

{ Fim do Algoritmo }

O algoritmo apresentado é uma modificação do algoritmo proposto em (Gilmore e Gomory, 1963). As modificações que o adaptam para a resolução do problema (31.1-31.5)

consistem na execução dos passos apresentados a seguir e marcados no algoritmo com as marcas (A), (B) e (C). Os passos são os seguintes:

(A) Atualizar os limites máximos para os itens envolvidos no cálculo de x_t , da seguinte forma:

A.1 Se t representa um item livre

Então temos: $\hat{d}_t = \hat{d}_t - x_t$ para o item associado a t ,

Senão, atualizamos os limites máximos dos itens envolvidos no compartimento t da seguinte forma:

Para todo $i \in A_k$, onde k é o agrupamento associado ao compartimento t , temos:

$d_i = d_i - (x_t * \alpha_{ijk})$ em que $\alpha_{ijk}, j = 1, \dots, N_k$, foi determinado no passo 1 das heurísticas dos “z” Melhores Compartimentos ou do Melhor Compartimento para “w” Capacidades.

A.2 Recalcular os limites máximos permitidos para todos os $j=1, \dots, N_k$ compartimentos do agrupamento k associado a t , fazendo:

$$\hat{d}_t = \text{Mínimo} \left\{ \left\lfloor \frac{d_i}{\alpha_{ijk}} \right\rfloor, \text{para todo } i \in A_k \right\}.$$

(B) Atualizar os limites máximos para os itens envolvidos na mudança do valor de x_r , da seguinte forma:

B.1 Se r representa um item livre

Então temos: $\hat{d}_r = \hat{d}_r + (x_r - 1)$ para o item associado a r ,

Senão, atualizamos os limites máximos dos itens envolvidos no compartimento r da seguinte forma:

Para todo $i \in A_k$, onde k é o agrupamento associado ao compartimento r , temos:

$d_i = d_i + ((x_r - 1) * \alpha_{ijk})$ onde $\alpha_{ijk}, j = 1, \dots, N_k$, foi determinado no passo 1 das heurísticas dos “z” Melhores Compartimentos ou do Melhor Compartimento para “w” Capacidades.

B.2 Recalcular os limites máximos permitidos para todos os $j=1, \dots, N_k$ compartimentos do agrupamento k associado a r , fazendo:

$$\hat{d}_r = \text{Mínimo} \left\{ \left\lfloor \frac{d_i}{\alpha_{ijk}} \right\rfloor, \text{para todo } i \in A_k \right\}.$$

(C) Atualizar os limites máximos para os itens envolvidos na mudança do valor de x_r , da seguinte forma:

C.1 Se r representa um item livre

Então temos: $\hat{d}_r = \hat{d}_r + I$ para o item associado a r ,

Senão, atualizamos os limites máximos dos itens envolvidos no compartimento r da seguinte forma:

Para todo $i \in A_k$, onde k é o agrupamento associado ao compartimento r , temos:

$d_i = d_i + \alpha_{ijk}$ onde $\alpha_{ijk}, j = 1, \dots, N_k$, foi determinado no passo 1 das heurísticas dos “z”

Melhores Compartimentos ou do Melhor Compartimento para “w” Capacidades.

C.2 Feito isso, recalculamos os limites máximos permitidos para todos os $j=1, \dots, N_k$ compartimentos do agrupamento k associado a r , fazendo:

$$\hat{d}_r = \text{Minimo} \left\{ \left\lfloor \frac{d_i}{\alpha_{ijk}} \right\rfloor, \text{para todo } i \in A_k \right\}.$$

Com este algoritmo, resolvemos heurísticamente o problema (31.1-31.5) das heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades. A forma gulosa como o algoritmo de Gilmore e Gomory percorre a árvore do problema da mochila não perde a solução ótima, mas agora não há garantias de otimalidade. Para os propósitos deste trabalho, este algoritmo foi muito eficiente, pois mostrou-se capaz de encontrar uma solução rápida para o problema (31.1-31.5) (um problema de otimização linear inteira). Além disso, a qualidade das soluções foram muito boas, a ponto destacar as heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades como as melhores (algo que não seria possível se este algoritmo produzisse soluções pobres para o problema (31.1-31.5)).

Como foi dito inicialmente, tal algoritmo necessita basicamente de um vetor para percorrer todo o espaço de soluções. Porém, como veremos no próximo apêndice, utilizamos listas encadeadas alocadas dinamicamente buscando explorar técnicas mais robustas de implementação.

Apêndice B. Características de Implementação para o Problema da Mochila Compartimentada

Neste apêndice, descrevemos algumas das características de implementação que foram desenvolvidas, visando à utilização dos algoritmos implementados para a resolução do Problema da Mochila Compartimentada, segundo as heurísticas propostas (capítulo 4).

B.1. Recursos Técnicos

A implementação foi desenvolvida em ambiente de programação Delphi. Para isto, utilizamos um microcomputador com processador Intel Celeron 650 MHz e 320 MBytes de memória RAM.

Baseado nas heurísticas propostas, uma estrutura de armazenamento de dados adequada ao problema foi desenvolvida. A seguir, apresentamos esta estrutura de dados.

B.2. Estruturas de Armazenamento dos Dados do Problema

O programa desenvolvido tem acesso a arquivos de dados de entrada relativos às características dos itens, dos agrupamentos e da mochila a ser preenchida. Com base neste conjunto de dados, o programa tem capacidade para determinar todos os dados necessários à execução do programa, como, por exemplo, os possíveis compartimentos gerados para cada um dos agrupamentos.

Num primeiro momento da implementação, estávamos trabalhando com três tipos diferentes de registros para armazenar todas as informações relativas aos itens, agrupamentos, compartimentos e planos de corte. Estes dados eram armazenados em listas encadeadas

estáticas (vetores) que, por sua vez, forçavam-nos a pré-estabelecer a quantidade de memória a ser utilizada.

Com o avanço das implementações, procuramos armazenar os dados em um único tipo de registro (estrutura) que busca explorar as características impostas pelos algoritmos utilizados. Tal registro foi definido de forma a comportar tanto os dados dos itens e dos agrupamentos, quanto os dados dos compartimentos e dos padrões de corte. A figura B.1 mostra a estrutura definida para este registro com seus respectivos campos de dados.

Identificador			Peso		Valor de Utilidade	
Limite de Ocorrências			Ocorrências		Ponteiro “ <i>Próximo</i> ”	
Agrupamento = Falso				Agrupamento = Verdadeiro		
Data de Entrega				Custo do Agrupamento		Número de Itens do Agrupamento
Dia	Mês	Ano	Capacidade Mínima		Capacidade Máxima	
			Ponteiro “ <i>Itens</i> ”			

Figura B.1: Registro para armazenamento dos dados.

Para entender a figura anterior considere:

- 1) Identificador, Peso, Valor de Utilidade, Limite de Ocorrências, Ocorrências e Ponteiro “Próximo” são dados comuns tanto na representação de itens, quanto na de agrupamentos;
- 2) O campo agrupamento é uma variável lógica (pode receber valores falso ou verdadeiro). Se o campo tiver valor falso, isto indica que o registro armazena dados de um item e os campos com o dia, mês e ano da entrega devem ser preenchidos. Por outro lado, se o campo agrupamento tiver valor verdadeiro é sinal de que os dados do registro são de um agrupamento e o seu custo, o número de itens do agrupamento e as capacidades mínima e máxima dos seus compartimentos devem ser preenchidos, além da existência de um campo (Ponteiro “Itens”) que apontará para uma lista encadeada contendo seus itens.

Uma vez definido o registro que armazena os dados de um item ou agrupamento, é preciso entender como armazenar todo o conjunto de itens devidamente agrupados. Para isto, implementamos listas encadeadas dinâmicas (note que no registro temos dois campos que são

ponteiros). A alocação dos dados nestas listas é feita dinamicamente, ou seja, à medida em que os itens e agrupamentos vão sendo lidos de um arquivo, novos “espaços” de memória vão sendo alocados para estes dados, o que otimiza a utilização de memória. Além disso, as listas são encadeadas de tal forma a valorizar a sua exploração pelas heurísticas propostas. Para entender a disposição das listas considere a seguinte legenda:

- **AG** → Registro que armazena os dados de um agrupamento com um ponteiro que aponta para os itens que o compõem.
- **IT** → Registro que armazena os dados de um item pertencente a um determinado agrupamento (não considera itens livres).
- **IL 5** → Registro que armazena os dados de um item livre.
- **PRIM** → Ponteiro para o primeiro agrupamento da lista de agrupamentos.
- → → Ponteiro para o registro seguinte da lista.

Um esboço da disposição das listas encadeadas é mostrado na figura B.2.

Figura B.2: Esboço da disposição das listas de armazenamento dos dados.

Esta disposição das listas nos permite gerar um padrão de corte percorrendo cada linha indicada pelos agrupamentos (AG) (geração dos compartimentos associados ao agrupamento) e, depois, percorrendo uma única lista encadeada representada por uma coluna (geração do padrão de corte). Vale lembrar que os dados dos compartimentos gerados para os agrupamentos são armazenados no próprio registro AG. A figura B.3 ilustra este processo.

Figura B.3: Geração de um padrão de corte utilizando as listas encadeadas.

Com o desenvolvimento desta estrutura de dados conseguimos facilitar a resolução do Problema da Mochila Compartimentada. Como vimos no apêndice A, os algoritmos de enumeração implícita implementados necessitam apenas de um vetor para percorrer o espaço de busca das soluções. Considerando que utilizamos listas encadeadas em substituição a estes vetores, ao aplicarmos um dos algoritmos em uma das linhas que representam os agrupamentos, estaremos gerando um compartimento e armazenando os seus dados no registro indicado por AG. Uma vez repetido o processo para todos os agrupamentos, teremos todos os registros indicados por AG preenchidos com os dados dos respectivos agrupamentos. A partir disto, basta aplicarmos o algoritmo de enumeração implícita na coluna de agrupamentos e itens livres para obtermos um padrão de corte. Este processo serve tanto para a Heurística de Decomposição quanto para a Heurística do Melhor Compartimento.

Para as heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades, a medida que vamos aplicando o algoritmo nas linhas dos agrupamentos, uma nova linha vai sendo criada (lembre-se que estamos alocando registros dinamicamente) contendo os dados dos compartimentos gerados. Feito isto, o método de enumeração implícita para o problema (31.1-31.5) é aplicado sobre a coluna de agrupamentos e itens livres. A Figura B.4 ilustra o processo de geração de padrões para as heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades, considerando o parâmetro (z ou w) igual a 2.

Figura 4: Geração de um padrão de corte utilizando as listas encadeadas para as heurísticas dos “z” Melhores Compartimentos e do Melhor Compartimento para “w” Capacidades.

Para evitar que muitos dados sejam armazenados repetidamente sem que estejam sendo usados, uma vez gerados os novos compartimentos, realiza-se uma operação de exclusão dos registros cujo campo ocorrência estejam zerados (não estão no compartimento). Desta forma, mantemos apenas os registros relacionados aos itens que estão efetivamente

compondo os compartimentos gerados. Para cada agrupamento, uma lista encadeada (a original – veja figura B.3) é mantida intacta para não perdermos a composição dos agrupamentos.

B.3. Fluxo das Informações no Programa

As implementações efetuadas seguiram o paradigma de programação estruturada. Para facilitar o seu desenvolvimento, ferramentas de análise estruturada que enfocassem o fluxo de dados no programa foram utilizadas. Tal abordagem é útil quando as informações são processadas sequencialmente e, à medida que são processadas pelo programa, novos dados vão sendo obtidos até que se chegue aos resultados esperados (Pressman, 1995). Para melhor entendermos o fluxo das informações dentro do programa, Diagramas de Fluxo de Dados (DFD) foram elaborados. Inicialmente iremos expor um DFD de nível 0, visto na figura B.5.

Figura B.5: DFD de nível 0 para o programa desenvolvido.

O diagrama nível 0 (figura B.5) pode ser detalhado procurando especificar melhor as ações a serem executadas. Um detalhamento do diagrama anterior (“explosão das bolhas”) para o programa principal originaria o seguinte DFD expandido.

Figura B.6: DFD expandido para o programa em desenvolvimento.

Note que pelo DFD anterior, num certo momento, a opção de resolver o Problema da Mochila Compartimentada através de uma das heurísticas deve ser fornecida para que o programa siga o determinado fluxo. No restante deste apêndice mostraremos os diagramas de fluxo de dados para as heurísticas propostas no capítulo 4.

B.4. Diagramas de Fluxo de Dados para as Heurísticas Propostas para o Problema da Mochila Compartimentada

Como visto no capítulo 4, cada uma das heurísticas propostas difere das outras no que diz respeito ao fluxo das informações. Estas diferenças são importantes para o desenvolvimento do programa até que os resultados sejam obtidos. A seguir, apresentaremos os diagramas de fluxo de dados para as quatro heurísticas propostas no capítulo 4.

Figura B.7: DFD para a Heurística de Decomposição.

Figura B.8: DFD para a Heurística do Melhor Compartimento.

Figura B.9: DFD para a Heurística dos "z" Melhores Compartimentos.

Figura B.10: DFD para a Heurística dos Melhores Compartimentos para “w” Capacidades.