O problema de corte de estoque com
reaproveitamento das sobras de material

Adriana Cristina Cherri

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito: 13/02/2006

Assinatura:_____

O problema de corte de estoque com reaproveitamento das sobras de material

Adriana Cristina Cherri

Orientador: *Prof Dr Marcos Nereu Arenales*

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

USP - São Carlos

Fevereiro/2006

Aos meus pais.

Agradecimentos

À Deus, por ter sempre me guiado e iluminado.

Ao meu orientador Marcos Nereu Arenales pela orientação, dedicação, amizade, e principalmente, pela confiança depositada no desenvolvimento deste trabalho e de outros que ainda deverão ser desenvolvidos.

À meus pais Luiz e Maria, pelo amor, carinho, educação, apoio incondicional, incentivo e presença em todos os anos de minha vida.

Aos meus irmãos César e Henrique, e à toda minha família pelo incentivo e carinho durante esses anos.

Ao meu amado Celso pelo apoio, compreensão, paciência, amizade, amor e por ter estado sempre ao meu lado.

Aos professores e funcionários do ICMC-USP que, direta ou indiretamente, contribuíram com este trabalho.

A todos os meus amigos do Laboratório de Otimização pelo apoio, incentivo, paciência e atenção.

A todos os meus amigos de São Carlos por estarem sempre presentes, seja nas horas de trabalho, seja nos momentos de descontração.

À FAPESP pela credibilidade e apoio financeiro.

E finalmente a todos que colaboraram indiretamente na realização deste trabalho.

Resumo

Os problemas de corte de estoque unidimensional consistem em cortar um conjunto de peças disponíveis em estoque para produzir um conjunto de itens em quantidades especificadas, em que apenas uma dimensão é relevante. Tais problemas têm inúmeras aplicações industriais e são bastante estudados na literatura. Tipicamente, esses problemas de corte apresentam uma característica comum - a minimização das perdas - entretanto, neste trabalho, consideramos que se uma perda é suficientemente grande para ser reaproveitada no futuro, não deve ser contabilizada como perda. Isto introduz uma postura diferente frente ao problema de corte: até que ponto a solução de perda mínima é a mais interessante, já que sobras podem ser reaproveitadas? Algumas características para considerar se uma solução é desejável são definidas e alterações em métodos heurísticos clássicos são propostas, de modo que os padrões de corte com perdas indesejáveis (nem tão grande, nem tão pequena) sejam alterados. As análises das soluções heurísticas são realizadas com base na resolução de um conjunto de classes de exemplos geradas aleatoriamente.

Abstract

One Dimensional Cutting stock problems consist of cutting a set of available pieces in order to produce ordered items in specified amounts, in which just one dimension is important, and in such a way as to optimize an objective function. Such problems have a number of industrial applications and are widely studied in the literature. Typically, those cutting problems goal the minimization of the waste. However, in this work, we considered that if a non used piece is "sufficiently large" to be reused in the future, it should not be considered as waste. This introduces a different posture in facing cutting stock problems: when does a solution with minimum waste is the most interesting, considering that some pieces can be reused? Some desirable characteristics are then defined and modifications in classic heuristic methods are proposed, so that the cutting patterns with undesirable waste (not so large, not so small) are modified. The analysis of the heuristic solutions are achieved based on solving a set of randomly generated instances.

Conteúdo

1	Introdução	1
1.1	Classificação dos problemas de Corte	2
1.2	Motivação e Objetivos	4
1.3	Organização do Texto	5
2	Revisão Bibliográfica	7
3	O Problema de Corte de Estoque Unidimensional	11
3.1	Definição e Formulação Matemática	11
3.1.1	Modelo Básico	12
3.1.2	Vários tipos de barras em estoque em quantidade limitada	14
4	O Problema de Corte com Reaproveitamento das Sobras	17
4.1	Definição do Problema	17
5	Heurísticas Construtivas	21
5.1	Breve Revisão de Heurísticas Clássicas	21
5.1.1	Heurística FFD	21
5.1.2	Heurística Gulosa	23
5.2	Heurísticas Construtivas para o Problema de Corte com reaproveitamento das Sobras de Material	24
5.2.1	Heurística FFD_R	25
5.2.2	Heurística $Gulosa_R$	27
6	Heurísticas Residuais	31
6.1	Problema Residual	31

6.2	Heurísticas Residuais - Estrutura Geral	32
6.3	Aproximação Inteira por Truncamento	33
6.3.1	Heurística Residual FFD	34
6.3.2	Heurística Residual Gulosa	34
6.4	Aproximação por Arredondamento Guloso	34
6.4.1	Heurística RAG - versão 1	36
6.4.2	Heurística RAG - versão 2	36
6.4.3	Heurística RAG - versão 3	37
6.5	Aproximação por Truncamento para o Problema de Reaproveitamento das Sobras de Material	37
6.5.1	Heurística Residual FFD_R	37
6.5.2	Heurística Residual Gulosa $_R$	38
6.6	RAG para o Problema de Reaproveitamento das Sobras de Material	38
7	Resultados Computacionais	41
7.1	Alguns exemplos da literatura	41
7.2	Gerador Aleatório	55
7.3	Resultados e Análises	56
8	Conclusões e Propostas Futuras	75
8.1	Conclusões	75
8.2	Propostas Futuras	76
	Referências Bibliográficas	76
	Apêndice A. Métodos Básicos de Corte	81
	Apêndice A.1. Método Simplex com Geração de Colunas Aplicado ao Problema de Corte	81
	Apêndice A.2. Problema da Mochila	86
	Apêndice A3 - Métodos de Resolução para o Problema da Mochila	87
	Apêndice B. Alguns Exemplos Resolvidos	91
	Apêndice B.1. Exemplo 1	91

Apêndice B.2. Exemplo 2	101
Apêndice B.3. Exemplo 3	111

Capítulo 1

Introdução

Os problemas de corte de estoque consistem em cortar peças maiores (*objetos*) disponíveis em estoque, produzindo um conjunto de peças menores (*itens*), com a finalidade de atender uma certa demanda, otimizando uma determinada função objetivo que pode ser, por exemplo, minimizar o número total de peças em estoque a serem cortadas, ou as perdas, ou custos das peças cortadas, etc. Estes problemas 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, o que estimula pesquisas acadêmicas de modelos de otimização para o controle e planejamento de sistemas produtivos.

Nas últimas quatro décadas, os problemas de corte têm sido estudados por um número crescente de pesquisadores, gerando avanços significativos em diversas áreas. O interesse por esses problemas é em parte explicado por sua aparente simplicidade e grande aplicabilidade prática, entretanto, são problemas de natureza complexa, NP-difíceis.

Devido à diversidade de situações práticas em que surgem os problemas de corte de estoque, é comum restrições ou objetivos novos para os quais as soluções dos modelos tradicionais são de pequena valia. Neste caso, heurísticas simples são utilizadas sem qualquer avaliação de desempenho. Um problema pouco estudado e freqüentemente encontrado na prática consiste em reaproveitar pedaços cortados (não demandados) desde que não sejam tão pequenos. Isto introduz uma mudança no critério de qualificar uma

solução ruim, já que perdas grandes são inaceitáveis quando se objetiva a minimização de perdas. Neste trabalho, com a finalidade de atacar este problema, algumas características são definidas para uma solução desejável (evitamos o termo “solução ótima” pois uma função objetivo que qualitativa as soluções não é bem definida) e algumas modificações em métodos heurísticos clássicos são realizadas.

Na literatura, poucos trabalhos tratam do problema de reaproveitamento das sobras, apenas Gradisar *et al.* [5], [6], [7] e [8] objetivaram em seus trabalhos minimizar as perdas ou então concentrá-las em um único padrão. Nestes trabalhos, os problemas por eles abordados apresentam todos os objetos em estoque com comprimentos diferentes.

1.1 Classificação dos problemas de Corte

Na literatura, existe uma grande diversidade de problemas de corte e empacotamento. Assim, em 1990, Dyckhoff [3] apresentou uma tipologia para classificá-los conforme algumas características, tais como: dimensionalidade, tipo de seleção dos objetos/itens, variedade de objetos/itens, entre outros. Com relação a dimensionalidade, os problemas de corte podem ser classificados como:

Unidimensional: apenas uma das dimensões é relevante no processo de corte. Problemas de corte unidimensional ocorrem no processo de corte de barras de aço com a mesma seção transversal, bobinas de papel, placas de alumínio, tubos para produção de treliças, etc. A Figura 1.1 ilustra este tipo de problema:

Figura 1.1: (a) Objeto (barra) a ser cortado; (b) objeto cortado produzindo 4 itens e uma perda.

Bidimensional: duas dimensões (comprimento e largura) são relevantes no processo de corte, uma vez que todas as peças cortadas têm a mesma espessura. Resolver este tipo

de problema, consiste em combinar geometricamente os itens ao longo do comprimento e da largura dos objetos em estoque. Problemas de corte bidimensional podem ocorrer em indústrias de placas de vidro, de madeiras, etc. A Figura 1.2 ilustra este tipo de problema.

Figura 1.2: (a) Placa a ser cortada; (b) objeto cortado produzindo 8 itens e uma perda.

1.5 - dimensional: duas dimensões são relevantes para a solução do problema, porém, uma delas não tem tamanho fixado. Um exemplo desse tipo de problema é o de um rolo de tecido que tem largura fixa e comprimento suficientemente grande para a produção de roupas. A Figura 1.3 ilustra o problema de corte 1.5 - dimensional, em que o comprimento total cortado L deve ser minimizado.

Figura 1.3: (a) Objeto a ser cortado; (b) objeto cortado produzindo itens e minimizando o comprimento total cortado.

Tridimensional: três dimensões são relevantes no processo de corte. Problemas de corte tridimensional ocorrem em indústrias de espuma para a produção de colchões, travesseiros, etc. Entretanto, suas aplicações mais interessantes aparecem no problema inverso do corte, são os chamados Problemas de Empacotamento, que consistem basicamente no carregamento de caixas dentro de contêineres ou caminhões (Morabito [18]). Na Figura 1.4 ilustra-se um problema de empacotamento.

Figura 1.4: (a) *Contêiner*; (b) *4 caixas empacotadas no contêiner*.

2.5 - dimensional: três dimensões são relevantes na resolução do problema, porém, uma delas é variável. Um exemplo seria um contêiner com largura e altura fixa e suficientemente comprido para acomodar um volume de carga. A Figura 1.3 pode ser interpretada como uma visão lateral deste caso.

Multidimensional: mais de três dimensões são relevantes para a solução. Um exemplo para esse tipo de problema é alocação de tarefas num dia de trabalho as quais utilizam diferentes recursos renováveis, porém limitados (Morabito [18]).

1.2 Motivação e Objetivos

O problema - chave deste trabalho, foi identificado na prática, no corte de tubos metálicos para a produção de pequenos aviões agrícolas. Outras tentativas frustradas de reaproveitamento de retalhos também foram observadas em indústrias metalúrgicas, nas quais os retalhos se tornaram sucatas, devido a dificuldades operacionais com estoques diversos e na dificuldade de otimizar cortes em objetos de dimensões quaisquer.

Embora muitas pesquisas tenham sido e estão sendo realizadas na área de corte e empacotamento, este projeto de mestrado, aborda um problema pouco explorado na literatura, que é o reaproveitamento das sobras para demandas futuras e seus principais objetivos podem ser resumidos como:

- Revisar o problema de corte de estoque;
- Reunir os trabalhos da literatura (aparentemente poucos) que tratam do problema de reaproveitamento da sobra de material decorrida do processo de corte;
- Propor mudanças em métodos clássicos de problemas de corte, tornando-os específicos para resolver o problema de corte com reaproveitamento das sobras para demandas futuras.

1.3 Organização do Texto

O texto está dividido em 8 capítulos e 2 apêndices de acordo com a seguinte estrutura:

No Capítulo 2, apresentamos uma breve revisão bibliográfica dos problemas de corte, orientada para o problema - chave de reaproveitamento.

No Capítulo 3, revisamos os problemas de corte de estoque unidimensional com um único tipo de objeto em estoque e com vários tipos de objeto em estoque, ambos são apresentados com suas formulações matemáticas.

No Capítulo 4, definimos o problema de corte de estoque com reaproveitamento das sobras de material, o objetivo principal desta dissertação.

No Capítulo 5, revisamos duas heurísticas de construção bem conhecidas na literatura para o problema de corte de estoque e, em seguida, apresentamos as modificações realizadas para torná-las específicas para o problema de corte de estoque com reaproveitamento das sobras de material.

No Capítulo 6, revisamos algumas heurísticas residuais para resolver o problema de corte de estoque unidimensional e, em seguida, também apresentamos as modificações realizadas para torná-las específicas para o problema de corte de estoque com reaproveitamento das sobras de material.

A literatura específica para o problema de corte unidimensional com reaproveitamento das sobras de material é escassa e encontramos apenas os trabalhos em [5], [6], [7] e [8]. Desta forma, no Capítulo 7 apresentamos alguns exemplos que guiaram as conclusões nesses trabalhos ([5], [6], [7] e [8]), os quais foram resolvidos pelos métodos heurísticos propostos nesta dissertação. As soluções obtidas foram comparadas e um alguns comentários foram realizados. Ainda neste capítulo, apresentamos o gerador de exemplos desenvolvido para analisar as soluções das heurísticas que tornamos específicas para o problema de reaproveitamento e os testes computacionais realizados.

O Capítulo 8 dedica-se às conclusões e perspectivas dos próximos passos para continuidade deste trabalho.

No Apêndice A, apresentamos o Método Simplex com a técnica de Geração de Colunas aplicado ao problema de corte juntamente com seu algoritmo. Também revisamos o problema da mochila (restrito e irrestrito) com dois métodos de solução: programação dinâmica e enumeração implícita.

Finalmente, no Apêndice B, apresentamos alguns exemplos resolvidos que foram escolhidos aleatoriamente nas classes de exemplos apresentadas no Capítulo 7.

Capítulo 2

Revisão Bibliográfica

Nas últimas décadas, problemas de corte de estoque têm sido estudados por inúmeros pesquisadores de várias áreas do conhecimento. Um fato observado é que as pesquisas nesta área têm caminhado no sentido de desenvolver técnicas heurísticas adequadas para resolução de tais problemas, visto que são da classe NP-completos e técnicas exatas, tais como métodos de enumeração implícita e variantes (por exemplo branch & cut, branch & price), demandam alto tempo computacional, sendo inviáveis para resolver problemas práticos que envolvem várias dezenas de itens a serem produzidos.

Os problemas de corte de estoque podem ser modelados por otimização inteira com um número muito grande de variáveis. A relaxação de tais modelos e a técnica de geração de colunas proposta por Gilmore e Gomory [9], [10] e [11] é a principal abordagem de solução dos problemas de corte de estoque. No Capítulo 3 e no Apêndice, revisamos esta abordagem.

Em Gilmore e Gomory [11], um estudo para problemas de corte de duas dimensões ou mais foi desenvolvido. Neste trabalho, foram impostas algumas restrições como corte guilhotinado, estagiado e irrestrito. Tais restrições entre outras são encontradas na prática com frequência.

Em 1975, Haessler [12] apresenta um procedimento heurístico computacionalmente eficiente, para resolver problemas de corte de estoque unidimensional com custo fixo associado à troca de padrões de corte. Esta heurística utiliza *níveis de aspiração* para

determinar se um padrão de corte encontrado deve ou não ser usado. Mais tarde, o autor propôs mudanças nos procedimentos de Gilmore e Gomory [9] e [10] que geram a solução inicial e os padrões subsequentes a entrar na base. Segundo ele, controlar a geração dos padrões a entrar na base e usar uma formulação mais restrita, porém menos eficiente do problema da mochila, ajuda a reduzir problemas de arredondamento e mudanças de padrões de corte. Minimizar o número de padrões de corte é um outro objetivo típico desejável na prática. Recentemente, Limeira e Yanasse [15] estudaram o problema de redução do número de padrões de corte, mantendo a perda dentro de um limite aceitável. Poldi e Arenales [23] analisam alguns métodos heurísticos para este problema.

Hinxman [14] faz uma revisão dos problemas e métodos de resolução dos problemas de corte, quando formaliza a heurística de repetição exaustiva, bastante utilizada na prática, principalmente quando a demanda dos itens é baixa (Poldi [22]).

Stadtler [20], em 1990, realizou um estudo de caso em uma indústria de alumínio, com o objetivo de minimizar o número de peças em estoque necessárias para atender a demanda dos clientes. Observou que os resultados obtidos pela heurística FFD (*First Fit Decreasing*) não eram satisfatórios, então apresentou um novo método, baseado no processo de Geração de Colunas, proposto por Gilmore e Gomory [9] e [10], acrescido de um procedimento de arredondamento para obtenção de uma solução inteira. Mais tarde, em 1996, Wäscher e Gau [25] reuniram vários métodos heurísticos para a resolução do problema de corte de estoque inteiro unidimensional, e realizaram um estudo computacional. Recentemente, Poldi e Arenales [23] também estudaram o problema de obtenção de uma solução inteira para o problema de corte de estoque unidimensional com ênfase para baixas demandas e vários tamanhos de objetos em estoque e mostraram que a técnica de geração de colunas com procedimentos de arredondamento adequados são superiores as heurísticas construtivas (FFD ou Gulosa).

Para classificar os vários tipos de problemas de corte e empacotamento existentes na literatura, em 1990, Dyckhoff [3] desenvolveu uma tipologia abrangente, integrando estes problemas. A tipologia foi fundada com base na estrutura lógica dos vários tipos problemas de corte e empacotamento com o objetivo de unificar o uso de diferentes

notações na literatura e concentrar mais adiante pesquisas em tipos especiais de problemas. Entretanto, uma mesma classe poderia abrigar problemas de grande diversidade, sendo que modelos e métodos adequados a um problema não se estendem a outro da mesma classe. Em 2006, Wäscher *et al.* [26] apresentam modificações na tipologia de Dyckhoff, além disso, introduzem uma nova categoria que define problemas diferentes dos apresentados anteriormente. A nova tipologia é baseada nas idéias originais de Dyckhoff [3], porém permite que cada problema tenha uma única representação.

Gau e Wäscher [4], em 1995, desenvolveram um gerador de exemplos de problemas de corte de estoque unidimensional padrão, o qual tem sido usado ou estendido para avaliar computacionalmente heurísticas propostas por vários autores.

Vahrenkamp [24], em 1996, faz um estudo comparativo entre o algoritmo de Gilmore e Gomory para a solução do problema de corte de estoque unidimensional com uma heurística baseada em algoritmos genéticos. A estrutura randômica dos algoritmos genéticos leva a uma escolha randômica dos padrões de corte, com o objetivo de minimizar a perda de material.

Gradisar *et al.* [5], em 1997, com o objetivo de criar um plano de corte unidimensional para diminuir a perda ou então concentrá-las em um único objeto, apresentaram um procedimento heurístico para otimizar o corte de rolos em indústrias de tecidos cujos objetos (*rolos*) em estoque são todos de comprimentos diferentes. O algoritmo aproximado (denominado COLA) por eles desenvolvido, considera o problema da mochila com bicritério (duas funções objetivos a serem minimizadas), um limitante para a perda (perdas cujos comprimentos são superiores ao limitante estipulado retornam ao estoque) e permite que no máximo quatro tipos diferentes de itens sejam cortados de um mesmo rolo. Outra característica apresentada pelo algoritmo está relacionada com a disponibilidade do estoque: se não há carência de objetos em estoque, então procura-se padrões de corte que apresentam a maior perda (superior a um determinado limitante que pode ser, por exemplo, o tamanho do menor item demandado e, neste caso será considerada sobra), caso haja escassez de objetos em estoque procura-se padrões que apresentam a menor perda. Em 1999, Gradisar *et al.* [6] propõem uma generalização e aperfeiçoamento

na solução do trabalho anterior. A modelagem matemática do problema é alterada e dividida de acordo com a disponibilidade de objetos em estoque, distribuição e relevância dos itens demandados. O algoritmo COLA é ampliado e desenvolvido em outra linguagem de programação gerando o algoritmo denominado CUT que pôde ser aplicado em outras indústrias.

Ainda em 1999, Gradisar *et al.* [7] com o objetivo de minimizar a perda de material, desenvolveram uma aproximação híbrida para o problema de corte de estoque unidimensional. A aproximação proposta, combina dois métodos: um método de programação linear orientada ao padrão (baseada em Gilmore e Gomory [10]) e um procedimento heurístico orientado ao item. A solução é apresentada como uma combinação desses dois métodos. A finalidade desta combinação é a habilidade para cortar os itens exatamente na quantidade demandada e acumular a perda em um único objeto, assim esta sobra poderá ser utilizada futuramente. O problema proposto para o desenvolvimento desta aproximação, considera a maior parte dos objetos com o mesmo comprimento (considerados inteiros) ou poucos objetos com tamanhos diferentes que são as sobras dos pedidos anteriores.

Em 2005, Gradisar *et al.* [8] ainda com o objetivo de minimizar a perda ou então concentrá-las em um único padrão desenvolveram um algoritmo para encontrar a solução de problemas gerais de corte de estoque unidimensional, em que a principal característica dos problemas é todos os objetos em estoque com comprimentos diferentes. O algoritmo proposto (C-CUT) combina dois métodos: branch & bound e um procedimento heurístico. A idéia do C-CUT (Combined Cutting) é encontrar uma solução temporária com o algoritmo CUT e, se ela não for ótima (a solução é considerada ótima se for composta por padrões que não apresentam perdas nem sobras), tentar melhorá-la resolvendo o subproblema resultante (o subproblema é formado por padrões gerados pelo algoritmo CUT que não são aceitáveis, ou por terem grandes perdas - perdas que não são consideradas sobras - ou muitas sobras, entre outros) por um método branch & bound. Para a solução final, os padrões não aceitos do algoritmo CUT são comparados com os padrões gerados pelo subproblema e, se estes forem melhores, são substituídos.

Capítulo 3

O Problema de Corte de Estoque Unidimensional

Neste capítulo, apresentamos dois modelos básicos do problema de corte de estoque unidimensional, sendo um deles formulado para problemas com estoque de um único comprimento e o outro com vários tipos de objetos em estoque.

3.1 Definição e Formulação Matemática

O problema de corte de estoque unidimensional, quando apenas um tipo de objeto é considerado, pode ser definido da seguinte forma:

Temos em estoque, um número suficientemente grande de *objetos* (barras, bobinas, etc), os quais chamaremos de *peças em estoque* de um determinado comprimento L , e um conjunto de pedidos de barras menores de comprimento $l_i, i = 1, \dots, m$, os quais chamaremos de *itens*. Cada item i deve ser produzido de acordo com sua demanda $d_i, i = 1, \dots, m$. O problema consiste em produzir os itens a partir do corte de peças em estoque de modo a atender a demanda, otimizando uma certa função objetivo que pode ser, por exemplo, minimizar o número total de peças em estoque a serem cortadas, ou minimizar a perda, ou maximizar o lucro, etc.

3.1.1 Modelo Básico

A formulação do Modelo Matemático de um problema de corte de estoque procede em duas etapas:

1. Definir todos os possíveis padrões de corte (maneira como um objeto em estoque é cortado para a produção de itens demandados);
2. Definir quantas vezes cada padrão de corte é utilizado para atender a demanda, que deve ser um número inteiro e não-negativo.

Definição 3.1 Chamamos de **padrão de corte** a maneira como um objeto em estoque é cortado para a produção dos itens demandados. A um padrão de corte associamos um vetor m -dimensional que contabiliza os itens produzidos:

$$\mathbf{a} = (\alpha_1, \alpha_2, \dots, \alpha_m)$$

em que α_i é a quantidade de itens do tipo i no padrão de corte.

No caso unidimensional, um vetor $\mathbf{a}$ corresponde a um padrão de corte se e somente se satisfazer as restrições de capacidade física do Problema da Mochila:

$$\begin{aligned} \ell_1\alpha_1 + \ell_2\alpha_2 + \dots + \ell_m\alpha_m &\leq L \\ \alpha_i &\geq 0, i = 1, \dots, m \text{ e inteiro.} \end{aligned} \tag{3.1}$$

Se o sistema 3.1 tem n soluções, então estas serão denotadas por: $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n$.

Ainda para definição do modelo matemático, temos que considerar:

- x_j = número de vezes que o objeto é cortado usando o padrão de corte j .

Desta forma, o problema de corte de estoque unidimensional, considerando como objetivo minimizar o número de objetos em estoque a serem cortados, pode ser formulado como o seguinte problema de otimização linear inteiro:

$$\begin{aligned} &\text{minimizar} && f(\mathbf{x}) = x_1 + x_2 + \dots + x_n \\ &\text{sujeito a:} && \begin{cases} \mathbf{a}_1x_1 + \mathbf{a}_2x_2 + \dots + \mathbf{a}_nx_n = \mathbf{d} \\ x_j \geq 0, j = 1, \dots, n \text{ e inteiro.} \end{cases} \end{aligned} \tag{3.2}$$

Por simplicidade de notação, escrevemos o modelo (3.2) em notação matricial como:

$$\begin{aligned} &\text{minimizar} && f(\mathbf{x}) = \sum_{j=1}^n x_j \\ &\text{sujeito a:} && \begin{cases} \mathbf{A}\mathbf{x} = \mathbf{d} \\ \mathbf{x} \geq 0 \text{ e inteiro.} \end{cases} \end{aligned} \tag{3.3}$$

sendo que cada coluna da matriz $\mathbf{A} \in \mathbb{R}^{m \times n}$ é um vetor associado a um padrão de corte, conforme a definição 3.1.

Para o modelo matemático (3.3), temos que a função objetivo minimiza o total de objetos a serem cortados. A restrição de igualdade garante que a quantidade de itens produzidos seja exatamente igual à demanda, e a restrição de integralidade garante que a repetição de cada padrão de corte j seja um número inteiro não-negativo. Qualquer solução do sistema linear (3.3), cujas componentes sejam inteiras e não negativas, fornece uma *solução factível* para o problema de corte de estoque.

Outros objetivos, como por exemplo, minimizar a perda total, podem ser considerados para a modelagem do problema de corte de estoque. Neste caso, a função objetivo em (3.3) seria substituída pela função *perda total*, que é definida como:

$$f(\mathbf{x}) = \sum_{j=1}^n c_j x_j \tag{3.4}$$

em que

$$c_j = L - (\alpha_{1j}\ell_1 + \alpha_{2j}\ell_2 + \dots + \alpha_{mj}\ell_m)$$

define a perda no padrão de corte j , $j = 1, \dots, n$.

Observe que a minimização da função $f(\mathbf{x})$ em (3.4) resulta no conjunto de padrões de corte com perda mínima e que neste caso simples, em que há apenas um tipo de objeto em estoque, são equivalentes: minimizar o total de objetos cortados e minimizar a perda total. A prova desta afirmação pode ser encontrada em Poldi [22].

3.1.2 Vários tipos de barras em estoque em quantidade limitada

O modelo matemático apresentado nesta seção foi desenvolvido por Gilmore e Gomory [10] e é utilizado para resolvermos o problema de corte de estoque com reaproveitamento das sobras de material.

Definição e formulação matemática

Considerando vários tipos de objetos em estoque, o problema de corte de estoque unidimensional pode ser formulado como:

Suponha que temos vários objetos em estoque em quantidades limitadas. Por outro lado, temos um conjunto de itens requeridos em quantidades variadas. O problema consiste em produzir os itens requeridos a partir do corte dos objetos em estoque otimizando uma determinada função objetivo.

Para a modelagem matemática deste problema, devemos considerar os seguintes dados:

Dados de demanda:

- m : número de tipos de itens;
- ℓ_i : comprimento do item i , $i = 1, \dots, m$;
- d_i : quantidade demandada do item i , $i = 1, \dots, m$.

Dados de estoque:

- K : número de tipos de objetos em estoque;
- L_k : comprimento dos objetos do tipo k em estoque, $k = 1, \dots, K$;
- e_k : quantidade disponível do objeto k , $k = 1, \dots, K$.

O modelo matemático para o problema de corte com vários objetos em estoque é semelhante ao problema (3.2), entretanto os padrões de corte devem ser definidos para cada tipo de objeto em estoque, isto é, devem satisfazer:

$$\begin{aligned}\ell_1\alpha_{1k} + \ell_2\alpha_{2k} + \dots + \ell_m\alpha_{mk} &\leq L_k \\ \alpha_{ik} &\geq 0, i = 1, \dots, m \text{ e inteiro, } k = 1, \dots, K.\end{aligned}\tag{3.5}$$

Se o sistema (3.5) possuir N_k soluções, então $a_{jk} = (\alpha_{1jk}, \alpha_{2jk}, \dots, \alpha_{mN_kk})$ é o vetor correspondente ao j -ésimo padrão de corte para o objeto de comprimento L_k e α_{ijk} é o número de itens do tipo i presentes no padrão de corte j para o objeto de comprimento L_k , $i = 1, \dots, m, j = 1, \dots, N_k, k = 1, \dots, K$.

Ainda para a descrição do modelo, devemos considerar uma informação adicional:

- c_{jk} é a perda produzida pelo padrão de corte j para o objeto em estoque do tipo k , $j = 1, \dots, N_k, k = 1, \dots, K$ e é dada por: $c_{jk} = L_k - \sum_{i=1}^m \ell_i \alpha_{ijk}$

Observe que c_{jk} pode ser a perda ou simplesmente um custo do objeto k , que independe do padrão de corte: $c_{jk} = c_k$.

Variável de decisão (frequência):

- x_{jk} : número de vezes que o objeto do tipo k é cortado usando o padrão $j, j = 1, \dots, N_k, k = 1, \dots, K$.

A formulação do problema de corte com vários objetos em estoque é definida por:

$$\begin{aligned}\text{minimizar } f(\mathbf{x}) &= \sum_{k=1}^K \sum_{j=1}^{N_k} c_{jk} x_{jk} \\ \text{sujeito a: } &\begin{cases} \sum_{k=1}^K \sum_{j=1}^{N_k} \mathbf{a}_{jk} x_{jk} = \mathbf{d} \\ \sum_{j=1}^{N_k} x_{jk} \leq e_k, \quad k = 1, \dots, K \\ x_{jk} \geq 0, \text{ e inteiro } j = 1, \dots, N_k, \quad k = 1, \dots, K.\end{cases}\end{aligned}\tag{3.6}$$

A função objetivo em (3.6) consiste em minimizar a perda de material. As restrições de igualdade garantem que a quantidade de itens produzidos seja igual a quantidade

de itens demandados (qualquer peça cortada que não seja um item demandado é considerada perda). As restrições de desigualdade, garantem que o número de objetos do tipo k cortados não exceda a disponibilidade e_k , $k = 1, \dots, K$ e, por fim, as últimas restrições em (3.6) garantem que as repetições de cada padrão de corte j sejam um número inteiro não-negativo. Qualquer solução apresentada cujas componentes sejam inteiras e não negativas, fornece uma *solução factível* para o problema de corte de estoque.

Por simplicidade de notação, escrevemos o modelo (3.6) em notação matricial como:

$$\begin{array}{ll} \text{minimizar} & f(\mathbf{x}) = \mathbf{c}^T \mathbf{x} \\ \text{sujeito a:} & \begin{cases} \mathbf{A}\mathbf{x} = \mathbf{d} \\ \mathbf{E}\mathbf{x} \leq \mathbf{e} \\ x \geq 0, \text{ e inteiro} \end{cases} \end{array} \quad (3.7)$$

em que $\mathbf{A}$ é a matriz dos padrões de corte e $\mathbf{E}$ é uma matriz de 0's e 1's.

Um fato que podemos observar, é que algumas vezes, minimizar o custo é mais interessante que minimizar a perda, pois os objetos podem ser comprados com descontos por terem dimensões fora dos padrões ou por serem retalhos de cortes anteriores que retornaram ao estoque e deseja-se que estes sejam utilizados antes que os outros objetos.

Para o problema de corte de estoque com reaproveitamento das sobras de material, problema - chave deste trabalho, o modelo 3.7 é utilizado, visto que uma formulação matemática específica para este problema ainda não foi obtida. O modelo 3.7 é de grande importância pois considera vários tipos de objetos em estoque, como o problema de reaproveitamento em que muito objetos que compõem o estoque são sobras de cortes anteriores e ainda tem por objetivo minimizar as perdas, que é um dos objetivos deste trabalho.

Capítulo 4

O Problema de Corte com Reaproveitamento das Sobras

Neste capítulo descrevemos o problema de corte de estoque em que as sobras geradas pelo processo de corte dos itens demandados são reaproveitadas para atender futuras demandas.

4.1 Definição do Problema

Durante o processo de corte de peças, em várias empresas, perdas inevitáveis ocorrem e eventualmente são descartadas, porém, algumas perdas podem ser reutilizadas como matéria prima, desde que tenham tamanhos significativos. Por outro lado, os métodos de solução para os problemas de corte buscam minimizar perdas, (objetivos alternativos podem ser definidos, mas perdas baixas devem ser perseguidas) sendo que, nesses métodos, considera-se como perda todo pedaço cortado que não seja um item demandado.

Embora perdas baixas sejam ainda um objetivo perseguido, a possibilidade de reuso introduz uma mudança no critério de seleção de uma solução. Uma alternativa para resolver este problema, seria planejar padrões de corte concentrando as perdas em poucos padrões, de modo que sejam grandes o suficiente para voltar ao estoque e serem utilizadas novamente.

Desta forma, apresentamos o problema de corte de estoque unidimensional com reaproveitamento das sobras de material como:

Um conjunto de peças (*itens*) deve ser produzido a partir do corte de unidades grandes (*objetos*), os quais podem ser de tamanhos padronizados ou reduzidos, decorrentes de cortes anteriores. São dados a demanda dos itens e as quantidades disponíveis dos objetos. A demanda deve ser atendida, cortando-se os objetos disponíveis, de modo que as perdas sejam ‘pequenas’ ou ‘suficientemente’ grandes para retornar ao estoque.

Definição 4.1 *Um pedaço cortado, que não seja um item, de comprimento suficientemente grande para ser reaproveitado é chamado **sobra**.*

Uma maneira de estabelecermos um comprimento ‘suficientemente grande’ (mínimo aceitável para *sobra*) seria por alguns critérios, como por exemplo: o comprimento do menor item demandado, a média do comprimento dos itens demandados, o comprimento do maior item, entre outros.

Diferentemente dos problemas clássicos de corte, para os quais funções objetivos são bem definidas (por exemplo, minimizar a perda total, número de objetos cortados, custos, etc.), agora objetivamos perdas ‘pequenas’ ou ‘suficientemente grandes’, sem que os objetivos anteriores sejam descartados. Duas soluções com a mesma perda total são, agora, diferenciadas. Para uma melhor compreensão do problema de corte de estoque com reaproveitamento das sobras de material, considere o seguinte exemplo, em que todo pedaço de tamanho superior ou igual a 4 metros é considerado *sobra*.

Figura 4.1: *Dados de um problema de corte e soluções alternativas.*

Do ponto de vista da função objetivo *perda total*, a Solução 1 (Fig. 4.1 - c) e a Solução 2 (Fig. 4.1 - d) apresentadas são equivalentes, pois têm a mesma perda total igual a 5 metros, porém, para o problema de corte com reaproveitamento, a Solução 2 (Fig. 4.1 - d) é preferível à Solução 1 (Fig. 4.1 - c), pois concentra as perdas em um único objeto e, como é superior a 4 metros, torna-se uma sobra que poderá voltar para o estoque e ser utilizada para atender demandas futuras. Na Solução 1, as perdas estão distribuídas entre os padrões, sendo inferiores a 4 metros e, portanto, serão descartadas. Para nosso problema, classificamos a Solução 1 como **indesejável**, enquanto a Solução 2 é **ideal**. Outra Solução **indesejável** é dada na Fig. 4.1 - e, embora esta também não gere perdas.

Como uma função objetivo para diferenciar tais soluções não é facilmente descrita, qualificamos as soluções segundo a definição a seguir.

Definição 4.2 *Para o problema de corte de estoque com reaproveitamento das sobras de material as soluções são definidas como:*

- **Solução ideal:** *quando todos os padrões tiverem perdas nulas, quase nulas ou, no*

máximo um padrão com sobra;

- **Solução aceitável:** *quando alguns padrões apresentarem perdas pequenas e alguns padrões apresentarem sobras;*
- **Solução indesejável:** *quando vários padrões apresentarem perdas que serão descartadas ou muitas sobras.*

Para precisar o que entendemos por *sobra*, *solução ideal*, *aceitável* ou *indesejável*, tanto a Definição 4.1 como a Definição 4.2 necessitam de alguns parâmetros, ou seja, para a definição de uma *perda pequena* (**aceitável**) e uma *sobra*, utilizamos 3 parâmetros a serem fornecidos pelo usuário:

- θ : Porcentagem aceitável para *perda máxima* para objetos padronizados (i.e., objetos que são comprados pela empresa), ou seja, θL_k é o tamanho da perda aceitável para um objeto de comprimento $L_k, k = 1, \dots, \bar{k}$, em que $\bar{k}$ é a quantidade de tipos de objetos padronizados disponíveis no estoque;
- β : Porcentagem aceitável para *perda máxima* para objetos retalhos (i.e., objetos que são sobras de outros processos de corte), ou seja, βL_k é o tamanho da perda aceitável para um objeto (*sobra*) de comprimento $L_k, k = \bar{k} + 1, \dots, K$;
- δ : Tamanho da *sobra mínima* aceitável (por exemplo, δ pode ser a média aritmética dos comprimentos dos itens demandados).

Os padrões de corte em que as perdas geradas são menores que $\theta L_k, k = 1, \dots, \bar{k}$, $\beta L_k, k = \bar{k} + 1, \dots, K$, ou maiores que δ são aceitáveis, caso contrário, serão alterados de maneira que tornem-se aceitáveis.

Com a finalidade de gerar um conjunto de padrões de corte *aceitáveis*, introduzimos alterações em algumas heurísticas clássicas bem conhecidas na literatura para o problema de corte de estoque unidimensional. Estas alterações serão descritas nas Seções 5.2, 6.5 e 6.6.

Capítulo 5

Heurísticas Construtivas

5.1 Breve Revisão de Heurísticas Clássicas

Apresentamos nesta seção uma breve revisão de duas heurísticas clássicas bem conhecidas na literatura para obtermos a solução do problema de corte de estoque inteiro: FFD e Gulosa.

5.1.1 Heurística FFD

A heurística FFD consiste em colocar o maior item num padrão de corte tantas vezes quanto for possível, ou seja, até que não haja mais espaço para colocar esse item, ou até que sua demanda já tenha sido atendida (os itens maiores são colocados em primeiro lugar, pois são mais difíceis de serem combinados). Assim, quando não for mais possível ou necessária a inclusão do maior item, o segundo maior item é considerado e assim por diante. Quando nenhum novo item pode ser incluído, um padrão de corte é construído.

Para cada objeto k em estoque, um padrão de corte é construído e aquele que apresentar menor perda, é escolhido. Tal padrão de corte é repetido tantas vezes quanto possível, sem que a demanda e a disponibilidade do objeto k seja ultrapassada.

Este procedimento de gerar um bom padrão de corte e repeti-lo tantas vezes quanto possível é conhecido na literatura como ‘heurística de repetição exaustiva’ (Hinxman [14])

e bastante empregado na área de corte e empacotamento. A seguir, apresentamos seu algoritmo:

Algoritmo FFD

Passo 1: $\{Início\}$

Passo 1.1:

Ordene os itens demandados de acordo com o seu tamanho, em ordem não-crescente.

Por simplicidade de notação, suponha que:

$$\ell_1 \geq \ell_2 \geq \dots \geq \ell_m.$$

Passo 1.2: $\{Conjunto de Parâmetros\}$

$r_i = d_i, i = 1, \dots, m$ $\{r_i \text{ é a demanda residual para todo } i = 1, \dots, m\}$

$s_k = e_k, k = 1, \dots, K$ $\{s_k \text{ é o estoque residual para o objeto } k, k = 1, \dots, K\}$

$j = 1$ $\{\text{primeiro padrão de corte}\}$

Passo 2: $\{Construção de um bom padrão de corte para cada objeto } k, k = 1, \dots, K\}$

Para $k = 1, \dots, K$ faça:

Se $s_k > 0$ então

$i = 1;$

$RESTO = L_k;$

$\{RESTO \text{ representa o espaço disponível para a alocação dos itens.}$

$\text{inicialmente será igual ao comprimento total do objeto a ser cortado}\}$

Enquanto $\left(i \leq m \text{ e } RESTO \geq \ell_m\right)$ faça:

$\{\text{isto é, enquanto houver espaço livre para alocação de itens faça}\}:$

$$\alpha_{ik} = \min \left\{ \left\lfloor \frac{RESTO}{\ell_i} \right\rfloor, r_i \right\};$$

$\{\mathbf{a}_k = (\alpha_{1k}, \dots, \alpha_{mk})^T \text{ é o vetor associado com o padrão de corte para o objeto do tipo } k\}$

$RESTO = RESTO - (\alpha_{ik}\ell_i);$

$i = i + 1;$

Fim-Enquanto

Fim-Se

$$w_k = L_k - RESTO; \quad \{ \text{perda no padrão de corte} \}$$

Fim-Para

Passo 3: $\{ \text{Determine o melhor padrão de corte e o número de vezes que ele é usado} \}$

Determine p tal que:

$\{ \text{critério para escolher o padrão de corte com perda mínima} \}$

$$w_p = \min \{ w_k, \text{ tal que } s_k > 0, \quad k = 1, \dots, K \}.$$

Determine o número de vezes que o padrão de corte é usado:

$$x_{jp} = \min \left\{ s_p, \left\lfloor \frac{r_i}{\alpha_{ip}} \right\rfloor, \text{ onde } \alpha_{ip} \neq 0, \quad i = 1, \dots, m \right\}.$$

Passo 4: $\{ \text{atualização da demanda e do estoque disponível} \}$

$$r_i = r_i - x_{jp}\alpha_{ip}, \quad i = 1, \dots, m;$$

$$s_p = s_p - x_{jp};$$

Se $r_i > 0$ e $s_p > 0$ então

$$j = j + 1;$$

Volte para o passo 2

Senão **PARE**.

5.1.2 Heurística Gulosa

Outra maneira de construir um bom padrão de corte para a heurística de repetição exaustiva, é gerar um padrão de corte pela resolução do Problema da Mochila:

$$\begin{aligned} z(\alpha) = \text{maximizar} \quad & \ell_1\alpha_{1k} + \ell_2\alpha_{2k} + \dots + \ell_m\alpha_{mk} \\ \text{sujeito a:} \quad & \begin{cases} \ell_1\alpha_{1k} + \ell_2\alpha_{2k} + \dots + \ell_m\alpha_{mk} \leq L_k \\ 0 \leq \alpha_{ik} \leq r_i, \text{ inteiro, } i = 1, \dots, m; \end{cases} \end{aligned} \quad (5.1)$$

o qual tem demanda r_i atualizada após ter sido escolhido o padrão de corte com a menor perda.

Neste trabalho, foi utilizado o método de enumeração implícita (Branch & Bound), desenvolvido por Gimore e Gomory [10] para resolver o problema 5.1.

A estrutura da heurística Gulosa é a mesma do algoritmo descrito anteriormente (seção 5.1.1) para a heurística FFD, porém, não é necessário o passo 1.1 e no passo 2 as mudanças realizadas são:

Algoritmo Guloso:

Passo 2: $\{ \text{Construir um bom padrão de corte para cada objeto } k \text{ em estoque, } k = 1, \dots, K \}$

Para $k = 1, \dots, K$ faça:

Se $s_k > 0$, então

Resolva o Problema da Mochila:

$$\begin{aligned} v_k = \text{maximizar} \quad & \ell_1 \alpha_{1k} + \ell_2 \alpha_{2k} + \dots + \ell_m \alpha_{mk} \\ \text{sujeito a:} \quad & \begin{cases} \ell_1 \alpha_{1k} + \ell_2 \alpha_{2k} + \dots + \ell_m \alpha_{mk} \leq L_k \\ 0 \leq \alpha_{ik} \leq r_i, \text{ inteiro, } i = 1, \dots, m; \end{cases} \end{aligned} \quad (5.2)$$

Faça: $w_k = L_k - v_k$; $\{ \text{perda no padrão de corte} \}$

Fim-Se

Fim-Para

5.2 Heurísticas Construtivas para o Problema de Corte com reaproveitamento das Sobras de Material

Nesta seção, apresentamos modificações que realizamos nas heurísticas FFD e Gulosa. Tais modificações originaram as heurísticas FFD_R e Gulosa_R que são específicas para resolvermos o problema de corte de estoque com reaproveitamento das sobras de material.

5.2.1 Heurística FFD_R

A heurística FFD_R consiste em aplicar a Heurística FFD para obter um padrão de corte e, logo após, a perda/sobra é analisada. Se a perda/sobra estiver dentro de limites aceitáveis (definidos previamente), o padrão para o próximo objeto é gerado. Se perda/sobra estiver fora de limites aceitáveis, um item do padrão (o maior) é retirado. Assim, para a sobra gerada com a retirada do item é resolvido o problema da mochila (5.1), cuja capacidade é a perda no padrão adicionada ao tamanho do item retirado. Depois de resolvida a mochila, a perda/sobra gerada é analisada e, se ainda não estiver dentro de limites aceitáveis, outro item do padrão inicial (segundo maior) é retirado. Novamente para a sobra gerada é resolvido o problema da mochila (5.1). Caso tenhamos retirado um item de cada comprimento dentre todos que compõem o padrão, voltamos a retirar o primeiro maior. Este procedimento é repetido até que a perda/sobra esteja dentro de limites aceitáveis ou o padrão inicial tenha sido anulado.

Com este procedimento, os últimos padrões são compostos por itens maiores, em geral, mais difíceis de serem combinados com os demais. Isto faz com que estes padrões (que devem combinar itens grandes) tenham perdas grandes, isto é, *sobras*, algo não totalmente indesejável. A seguir, apresentamos os principais passos do algoritmo FFD_R .

Passos do Algoritmo FFD_R

$k = 1$;

$\text{demanda} = d_i, i = 1, \dots, m$;

Passo 1: {Início}

Aplicar a heurística FFD para o objeto k (Seção 5.1.1);

Obter o padrão de corte FFD, cujo vetor associado é denotado por α_k^{FFD} .

Passo 2: {Análise da perda}

Se a perda/sobra for aceitável então

Início

padrão α_k^{FFD} é aceito e armazenado.

$demanda = demanda - \alpha_k^{FFD};$

$k = k + 1;$

Se $demanda = 0$ ou $k = K + 1$ então **PARE**

Senão volte ao passo 1.

Fim

Passo 3: $\{Parâmetro\}$

i = índice do item de maior comprimento no padrão FFD.

Passo 4: $\{Altera padrão com perda indesejável\}$

Retire o item i do padrão FFD (padrão FFD é alterado);

ESPAÇO = perda no padrão FFD + ℓ_i ;

Atualizar a demanda, ou seja, retire da demanda os itens que compõem o padrão α_k^{FFD} .

Passo 5:

Resolva o Problema da Mochila (5.1) com capacidade física ESPAÇO gerada no passo 4 e demanda atualizada;

Obter o padrão mochila, cujo vetor associado é denotado por $\alpha^{mochila}$;

Passo 6:

Se $\alpha_k^{FFD} = 0$, então

Início

padrão $\alpha^{mochila}$ é analisado.

Se a perda/sobra é aceitável, então o padrão $\alpha^{mochila}$ é aceito e armazenado.

Senão p = índice do item de maior comprimento no padrão $\alpha^{mochila}$.

Retire o item p do padrão $\alpha^{mochila}$ até obter uma sobra aceitável.

$demanda = demanda - \alpha^{mochila};$

$k = k + 1;$

Se $demanda = 0$ ou $K = k + 1$ então **PARE**

Senão volte ao passo 1.

Fim

Passo 7: $\{ \text{Novo padrão} \}$

Novo padrão: FFD + Mochila cujo vetor associado é: $\alpha_k^{FFD} + \alpha^{mochila}$;

Passo 8: $\{ \text{Análise da perda - padrão FFD + Mochila} \}$

Analisar a perda/sobra do padrão FFD + Mochila;

Se a perda/sobra for aceitável então

Início

padrão FFD + Mochila é aceito e armazenado;

$demanda = demanda - (\alpha_k^{FFD} + \alpha^{mochila})$;

$k = k + 1$;

Se $demanda = 0$ ou $k = K + 1$ então **PARE**

Senão volte ao passo 1.

Fim.

Passo 9: $\{ \text{Atualização do parâmetro} \}$

Se um item de cada tipo dentre todos que compõem o padrão já foi retirado do padrão α_k^{FFD} , então $i =$ índice do item de maior comprimento no padrão α_k^{FFD}

Senão $i =$ próximo maior item do padrão α_k^{FFD} ainda não retirado (o padrão FFD + Mochila é desconsiderado);

Volte ao passo 4.

5.2.2 Heurística Gulosa_R

A heurística Gulosa_R consiste em aplicar a Heurística Gulosa para obter um padrão e, logo após, a perda/sobra é analisada. Se a perda/sobra estiver dentro de limites aceitáveis (definidos previamente), o padrão para o próximo objeto é gerado. Se a perda/sobra estiver fora de limites aceitáveis, um item do padrão (o maior) é retirado e a perda/sobra novamente é analisada, se ela ainda não estiver em limites aceitáveis, outro item (segundo maior) é retirado. Este processo é repetido até que tenhamos uma sobra aceitável. A seguir, apresentamos os principais passos do algoritmo Guloso_R.

Principais Passos do Algoritmo Guloso_R

$k = 1$;

$demanda = d_i, i = 1, \dots, m$;

Passo 1: {Início}

Aplicar a heurística Gulosa para o objeto k (Seção 5.1.2).

Obter o padrão guloso, cujo vetor associado é α_k^{guloso} .

Passo 2: {Análise da perda}

Se a perda/sobra for aceitável então

Início

padrão α_k^{guloso} é aceito e armazenado.

$demanda = demanda - \alpha_k^{guloso}$;

$k = k + 1$;

Se $demanda = 0$ ou $k = K + 1$ então **PARE**

Senão volte ao passo 1.

Fim

Passo 3: {Parâmetro}

i = índice do item de maior comprimento no padrão guloso.

Passo 4: {Altera padrão com perda indesejável}

Retire o item i , do padrão guloso (padrão guloso é alterado);

Obter o padrão $\alpha_k^{guloso} - \ell_i$, cujo vetor associado é denotado por $\alpha_k^{guloso-\ell_i}$;

Passo 5: {Análise da perda}

Analisar a perda/sobra do padrão $\alpha_k^{guloso-\ell_i}$;

Se a perda/sobra for aceitável então

Início

padrão $\alpha_k^{guloso-\ell_i}$ é aceito e armazenado;

$demanda = demanda - \alpha_k^{guloso-\ell_i}$;

$k = k + 1$;

Se $demanda = 0$ ou $k = K + 1$ então **PARE**

Senão volte ao passo 1.

Fim

Passo 6: *{Atualização do parâmetro}*

i = próximo maior item do padrão α_k^{gulosa} ainda não retirado;

Volte ao passo 4.

Capítulo 6

Heurísticas Residuais

As heurísticas apresentadas neste capítulo, geram uma solução inteira para o problema de corte de estoque unidimensional a partir da solução ótima fracionária do problema (3.7). Estas heurísticas são: Residual FFD, Residual Gulosa, Residual por Arredondamento Guloso (RAG) - versões 1, 2 e 3. Ainda, com a finalidade de resolver o problema de corte de estoque com reaproveitamento das sobras de material, realizamos modificações nestas heurísticas e apresentamos as heurísticas Residual FFD_R, Residual Gulosa_R, RAG_R - versões 1, 2 e 3.

6.1 Problema Residual

Consideremos o problema de corte de estoque:

$$\begin{array}{ll} \text{minimizar} & f(\mathbf{x}) = \mathbf{c}^T \mathbf{x} \\ \text{sujeito a:} & \left\{ \begin{array}{l} \mathbf{Ax} = \mathbf{d} \\ \mathbf{Ex} \leq \mathbf{e} \\ x \geq 0 \end{array} \right. \end{array} \quad (6.1)$$

com $\mathbf{x}$ sendo uma solução ótima fracionária e $\mathbf{y}$ uma solução inteira aproximada para $\mathbf{x}$ (em certo sentido, $\mathbf{y} \cong \mathbf{x}$) tal que:

$$\mathbf{A}\mathbf{y} \leq \mathbf{d} \quad (6.2)$$

$$\mathbf{E}\mathbf{y} \leq \mathbf{e}. \quad (6.3)$$

Uma maneira usual de obter $\mathbf{y}$ é arredondar $\mathbf{x}$ por um **truncamento** trivial:

$$\mathbf{y} = (\lfloor x_1 \rfloor, \lfloor x_2 \rfloor, \dots, \lfloor x_n \rfloor)$$

o qual satisfaz (6.2) e (6.3) uma vez que todos os coeficientes de $\mathbf{A}$ são não-negativos e $\mathbf{x}$ satisfaz $\mathbf{A}\mathbf{x} = \mathbf{d}$ e $\mathbf{E}\mathbf{x} \leq \mathbf{e}$. Na Seção (6.4), vemos outra maneira de se obter uma solução inteira aproximada.

Definição 6.1 (O Problema Residual)

Seja $\mathbf{y}$ uma solução inteira aproximada para $\mathbf{x}$, $\mathbf{r} = \mathbf{d} - \mathbf{A}\mathbf{y}$ a demanda residual e $\mathbf{s} = \mathbf{e} - \mathbf{E}\mathbf{y}$ o estoque residual de objetos disponíveis. O problema residual é dado por:

$$\begin{array}{ll} \text{minimizar} & f(\mathbf{x}) = \mathbf{c}^T \mathbf{x} \\ \text{sujeito a :} & \left\{ \begin{array}{l} \mathbf{A}\mathbf{x} = \mathbf{r} \\ \mathbf{E}\mathbf{x} \leq \mathbf{s} \\ x \geq 0, \quad \text{e inteiro} \end{array} \right. \end{array} \quad (6.4)$$

6.2 Heurísticas Residuais - Estrutura Geral

Em oposição às heurísticas construtivas, que geram um bom padrão de corte e o utilizam à exaustão, as heurísticas residuais consistem em resolver o problema 6.1 relaxado, obter uma solução inteira aproximada, resolver o problema residual relaxado resultante (6.4), obter uma solução inteira e assim por diante. A seguir, apresentamos uma estrutura geral dessas heurísticas.

Algoritmo - heurísticas residuais

Passo 1: { *Início* }

Faça $\ell = 0$, $\mathbf{r}^0 = \mathbf{d}$ e $s^0 = e$;

Passo 2: { *Determinação da solução ótima contínua* }

Resolva o problema residual (6.4) com $r = r^\ell$ e $s = s^\ell$;

Seja $\mathbf{x}^\ell$ a solução contínua (a técnica de geração de colunas é usada);

Se $\mathbf{x}^\ell$ é uma solução inteira, **então** PARE.

Passo 3: { *Determinação da solução inteira aproximada* }

Determine uma solução inteira aproximada $\mathbf{y}^\ell$ para a solução contínua $\mathbf{x}^\ell$;

Se $\mathbf{y}^\ell$ é um vetor nulo, **então** vá para o passo final.

Passo 4: { *Atualização* }

Determine a nova demanda residual e estoque residual

$$\mathbf{r}^{\ell+1} = \mathbf{r}^\ell - \mathbf{A}\mathbf{y}^\ell;$$

$$\mathbf{s}^{\ell+1} = \mathbf{s}^\ell - \mathbf{E}\mathbf{y}^\ell;$$

$$\ell = \ell + 1;$$

Volte ao passo 2.

Passo Final:

Resolva o problema residual final com poucos itens por algum método (heurístico ou exato).

Para que o algoritmo geral de heurísticas residuais seja totalmente definido, especificamos como determinar $\mathbf{y}^\ell$, solução inteira aproximada no passo 3, e como resolver o problema residual final no passo final.

6.3 Aproximação Inteira por Truncamento

As heurísticas residuais FFD e Gulosa a seguir, apresentam no algoritmo da Seção 6.2 o mesmo passo 3, elas diferenciam-se apenas no passo final o qual especificaremos nas

Seções (6.3.1 e 6.3.2).

A solução inteira aproximada y^ℓ no passo 3 é determinada por um truncamento da solução relaxada x^ℓ , ou seja,

Passo 3: { *Determinação da solução inteira aproximada* }

Truncar a solução contínua:

$$\mathbf{y}^\ell = \lfloor \mathbf{x}^\ell \rfloor$$

6.3.1 Heurística Residual FFD

Passo Final:

Resolver o problema residual pela Heurística FFD (seção 5.1.1)

6.3.2 Heurística Residual Gulosa

Passo Final:

Resolver o problema residual pela Heurística Gulosa (seção 5.1.2).

6.4 Aproximação por Arredondamento Guloso

Nesta seção, apresentamos uma técnica diferente para se obter uma aproximação inteira para uma solução contínua $\mathbf{x}$, que satisfaça (6.2) - (6.3) e três versões desenvolvidas e analisadas por Poldi e Arenales [23], as quais definem novas heurísticas residuais quando aplicadas no passo 3 do algoritmo residual (seção 6.2). O passo 3 é dividido em duas partes: passo 3.1 {*Pré processamento*} e passo 3.2 {*Arredondamento*}.

As formas de ordenação, *Pré processamentos*, da solução contínua $\mathbf{x}^\ell$ obtida no passo 2, são descritas nas seções (6.4.1), (6.4.2) e (6.4.3).

Considerando o passo 3.2 (*Arredondamento*) e iniciando com o primeiro padrão de corte na solução contínua, a frequência é arredondada para o inteiro superior, ou seja, o primeiro padrão de corte para o objeto j é dado por: $y_{1j} = \lceil x_{1j} \rceil$ e sua factibilidade é testada, (excesso de itens não é permitido) caso haja excesso, isto é, $\mathbf{y} = (y_{1j}, 0, \dots, 0)$ é tal que $\mathbf{A}\mathbf{y} \geq \mathbf{d}$, então a frequência é reduzida de um ($y_{1j} = y_{1j} - 1$) até que o excesso seja eliminado. A seguir, com a primeira componente já arredondada, repetimos este processo para o segundo padrão da solução contínua, $\mathbf{y} = (y_{1j}, y_{2k}, 0, \dots, 0)$ e assim sucessivamente até que o último padrão de corte gerado seja examinado. Este procedimento gera um vetor $\mathbf{y}$ que é uma solução inteira aproximada para o problema de corte de estoque.

Observe que o procedimento de geração de colunas é tal que cada padrão de corte gerado pode ser utilizado pelo menos uma vez (ver a.9), de modo que pelo menos a frequência de um padrão de corte seja arredondada para o inteiro superior. Isto assegura que a demanda residual se torne pequena na próxima iteração, e assim, com um número finito de passos, seja nula.

Notação: por simplicidade, denotamos até o momento, x_{jk} a frequência do padrão de corte j para o objeto do tipo k . Entretanto, temos agora uma lista de padrões de corte, cada um associado com um objeto em estoque, então usaremos a notação:

$$x_{jk_j}$$

para denotar a frequência do padrão de corte j , para o objeto do tipo k_j . Assim, para os padrões de corte $(x_{11}, x_{23}, \dots)$ temos $j = 1, k_1 = 1; j = 2, k_j = 3$.

A seguir, especificamos o passo 3 destas heurísticas para o algoritmo residual da seção (6.2).

Algoritmo Residual RAG

Passo 3: *{Determinação da solução da solução inteira aproximada}*

Passo 3.1 *{Pré processamento}*

Ordene o vetor de frequência $\mathbf{x}^\ell$ dos padrões de corte obtido no passo 2 segundo

algum critério (três são dados a seguir).

Passo 3.2 *{Arredondamento}*

Seja T o número total de padrões de corte gerados na solução ótima contínua do passo 2:

Para $i = 1, \dots, T$ faça

$y_{ik_i} = \min\{\lceil x_{ik_i} \rceil, s_{k_i}\}$ *{ y_{ik_i} é a frequência inteira aproximada do padrão de corte i para o objeto k_i }*;

$s_{k_i} = s_{k_i} - y_{ik_i}$ *{atualização do estoque residual disponível}*;

$(y = y_{1k_1} \ y_{2k_2} \ \dots \ y_{ik_i} \ 0 \ 0 \ \dots \ 0)$;

Enquanto $\mathbf{A}\mathbf{y} \not\leq \mathbf{r}$, faça:

{enquanto existir excesso de itens}

$y_{ik_i} = y_{ik_i} - 1$;

$s_{ik_i} = s_{ik_i} + 1$;

6.4.1 Heurística RAG - versão 1

Para organizarmos os padrões no passo 3.1 o vetor de frequência $\mathbf{x}$ é ordenado de forma não-crescente, isto é, o padrão mais utilizado tem prioridade maior.

Passo 3.1 *{Pré processamento: prioridade para a maior frequência}*

Ordene as componentes de $\mathbf{x}$ tal que,

$$x_{1k_1} \geq x_{2k_2} \geq \dots \geq x_{Tk_T}.$$

6.4.2 Heurística RAG - versão 2

Esta versão, utiliza outro critério para a ordenação dos padrões de corte. Seja p_{jk} a perda no padrão de corte j do objeto k .

Passo 3.1: *{Pré processamento: prioridade menor perda}*

Ordene as componente de $\mathbf{p}$ tal que,

$$p_{1k_1} \leq p_{2k_2} \leq \dots \leq p_{Tk_T}.$$

6.4.3 Heurística RAG - versão 3

A Heurística RAG - versão 3 também utiliza outro critério para a ordenação dos padrões de corte. Seja $f_{jk} = x_{jk} - \lfloor x_{jk} \rfloor$ o valor fracionário de x_{jk} .

Passo 3.1: *{Pré processamento: prioridade para o maior valor fracionário}*

Ordene as componentes de $\mathbf{f}$ tal que,

$$f_{1k_1} \geq f_{2k_2} \geq \dots \geq f_{Tk_T}.$$

6.5 Aproximação por Truncamento para o Problema de Reaproveitamento das Sobras de Material

Apresentamos nesta seção modificações nas Heurísticas Residual FFD e Residual Gulosa de modo a orientá-las para o reaproveitamento das sobras de material. Estas Heurísticas Residuais, FFD_R e Gulosa_R , apresentam no algoritmo da Seção 6.2 o mesmo passo 3, ou seja, a solução inteira aproximada $\mathbf{y}^\ell$ é determinada por um truncamento trivial que definimos na seção (6.1), e se diferenciam apenas no passo final que especificaremos a seguir.

6.5.1 Heurística Residual FFD_R

A Heurística Residual FFD_R , desenvolvida com o objetivo de resolver problemas de corte de estoque com reaproveitamento das sobras de material apresenta o seguinte passo final no algoritmo da seção 6.2:

Passo Final:

Resolver o problema residual pela Heurística FFD_R (Seção 5.2.1).

6.5.2 Heurística Residual Gulosa_R

Passo Final:

Resolver o problema residual pela Heurística Gulosa_R (seção 5.2.2).

6.6 RAG para o Problema de Reaproveitamento das Sobras de Material

Para resolver o problema de corte de estoque com reaproveitamento das sobras de material também realizamos modificações nas heurísticas RAG - versões 1, 2 e 3 e desta forma, apresentamos nesta seção as Heurísticas RAG_R - versões 1, 2 e 3.

Estas heurísticas apresentam uma maneira diferente de obter o tamanho aceitável para a perda, ou seja, a perda é calculada conforme os passos a seguir.

Cálculo do nível aceitável de perda

Para calcularmos o comprimento aceitável para a perda, considere os padrões de corte obtidos por uma das versões da heurística RAG (Seções 6.4.1, 6.4.2 ou, 6.4.3).

Passo 1:

Calcular a perda gerada por cada padrão e dividir pelo comprimento do objeto correspondente. Obtenha λ_{ik_i} , ou seja,

$$\lambda_{ik_i} = \frac{p_{ik_i}}{L_{k_i}}$$

Passo 2:

Somar as perdas geradas pelos padrões de corte que forem menores ou iguais que o tamanho da sobra (δ) já estabelecido e obtenha σ , ou seja:

$$\sigma = \sum_{i/p_{ik_i} \leq \delta} p_{ik_i}$$

Passo 3:

Somar o comprimento total dos objetos utilizados para gerar os padrões em que a perda é menor ou igual que o tamanho da sobra (δ) já estabelecido e obtenha γ , ou seja,

$$\gamma = \sum_{i/p_{ik_i} \leq \delta} L_{k_i}$$

Passo 4: $\{ \text{Tamanho aceitável para a perda} \}$

Obter o tamanho aceitável para a perda, que é dado por $\rho = \frac{\sigma}{\gamma}$

A seguir, apresentamos os principais passos da estrutura geral da Heurística RAG_R .

Principais Passos do Algoritmo RAG_R **Passo 1:** $\{ \text{Início} \}$

Determine uma solução inicial pela Heurística RAG - versões 1, 2 ou 3.

Passo 2: $\{ \text{Análise das perdas geradas pelos padrões} \}$

Analisar a perda/sobra de todos os padrões gerados no passo 1;

Se a perda for *aceitável*, ou seja, $\lambda_{ik_i} \leq \rho$ (obtidos no cálculo do nível aceitável de perda) então o padrão é aceito.

Passo 3: $\{ \text{Rejeição de padrões} \}$

Rejeitar os padrões em que houver sobras ou perdas não aceitáveis.

Passo 4:

Aplicar a Heurística FFD_R para resolver o problema residual formado pelos itens dos padrões rejeitados no passo 3.

Capítulo 7

Resultados Computacionais

Neste capítulo, realizamos alguns testes computacionais para analisar o desempenho das heurísticas propostas para resolver o problema de corte de estoque com reaproveitamento das sobras de material. Na seção 7.1 apresentamos alguns exemplos que foram enviados pelo Professor Peter Trkman (o Professor Peter Trkman, juntamente com o Gradisar *et al.* buscam resolver o problema de corte com reaproveitamento de sobras, porém, em seus problemas, consideram o estoque composto por um único objeto de cada tipo), da University of Ljubljana, juntamente com as soluções obtidas pelo algoritmo CUT e pelas heurísticas de reaproveitamento dos Capítulos 5 e 6. Na seção 7.3 apresentamos 16 classes, com 20 exemplos em cada classe, geradas aleatoriamente para avaliar os procedimentos heurísticos.

Todos os algoritmos apresentados neste trabalho foram implementados na linguagem de programação DELPHI 5 e os exemplos foram executados em um computador Pentium IV (3 GHz, com 2 GB de memória RAM).

7.1 Alguns exemplos da literatura

Nesta seção, apresentamos alguns exemplos numéricos (arquivos enviados pelo Professor Peter Trkman - University of Ljubljana) de problemas de corte juntamente com as

soluções obtidas pelo algoritmo CUT (Gradisar *et al.* [6]), desenvolvido com a finalidade de minimizar a perda de material ou se possível, concentrá-las em um único objeto de modo que se tornem uma sobra que pode ser utilizada no futuro. Para estes exemplos, também apresentamos as soluções obtidas pelos procedimentos heurísticos que desenvolvemos com o objetivo de minimizar a perda e reaproveitar as sobras de material geradas pelos processos de corte. Após apresentadas as soluções, alguns comentários são realizados.

Para todos os exemplos a seguir, temos vários tipos de objetos em estoque, porém, a disponibilidade é de apenas um objeto para cada tipo. Desta forma, o número de objetos cortados é equivalente à quantidade de padrões de corte gerados pelo algoritmo CUT e por cada heurística.

Como comparamos as soluções obtidas pelo algoritmo CUT, utilizamos nestes exemplos os mesmos critérios adotados por Gradisar *et al.*[6] para estipular *perda* e *sobra*: todo pedaço restante em um padrão de corte que seja maior ou igual que o menor item demandado é de tamanho suficientemente grande para retornar ao estoque e é considerado *sobra*, caso contrário é considerado *perda* e, conseqüentemente, descartado.

Exemplo 1:

Temos em estoque $K = 20$ tipos de objetos com comprimentos entre 2.000 cm e 5.500 cm e disponibilidade de uma unidade para cada tipo, cujos dados estão na Tabela 7.1, e uma demanda de 5 tipos de itens, ou seja, $m = 5$. O comprimento de cada item $i, i = 1, \dots, 5$ e sua respectiva demanda estão na Tabela 7.2.

Id Objeto	Tamanho (cm)	Estoque
1	5420	1
2	4006	1
3	3922	1
4	2228	1
5	2483	1
6	4499	1
7	3012	1
8	2893	1
9	2507	1
10	3816	1
11	4685	1
12	4680	1
13	2463	1
14	2055	1
15	4461	1
16	4909	1
17	4812	1
18	2687	1
19	5108	1
20	2556	1

Tabela 7.1: *Dados do Exemplo 1 - Estoque*

Item	Comprimento (cm)	Demanda
1	437	13
2	516	33
3	296	37
4	302	12
5	345	55

Tabela 7.2: *Dados do Exemplo 1 - Itens*

Para este exemplo, o limite inferior para sobra é 296 *cm*, ou seja, toda perda superior a 296 *cm* é considerada sobra.

As soluções do exemplo 1 obtidas pelo algoritmo CUT (Gradisar *et al.* [6]) e pelas heurísticas específicas para resolver o problema de reaproveitamento estão na Tabela 7.3.

		Construtiva		Residual				
	CUT	FFD_R	$Gulosa_R$	FFD_R	$Gulosa_R$	$RAG_R\ 1$	$RAG_R\ 2$	$RAG_R\ 3$
Objetos cortados	14	18	17	16	16	15	15	16
Perda total	62	18	3	2	2	1	0	1
Sobra total	2408	6907	5106	3276	3276	2225	2264	2202
Padrões com Sobra	1	7	9	2	2	2	2	4
Padrões com Perda	7	8	1	2	2	1	0	1

Tabela 7.3: *Solução do Exemplo 1.*

Pela Tabela 7.3, podemos observar que o Algoritmo CUT utiliza menos objetos para atender a demanda e concentra a sobra (2408 cm) em apenas 1 padrão de corte. Entretanto, a heurística $RAG_R\ 2$, embora não concentre a sobra em um único padrão, não gera perda, enquanto que o algoritmo CUT apresenta uma perda de 62 cm. As demais heurísticas residuais também não concentram a sobra em um único padrão, mas apresentam uma perda menor que o algoritmo CUT. Se classificarmos as soluções de acordo com a definição 4.2, temos que as soluções das heurísticas residuais, com exceção da heurística $RAG_R\ 3$ (com o dobro de padrões de corte com sobra), são consideradas *aceitáveis* para este exemplo. As soluções das heurísticas construtivas e das residuais FFD_R e $Gulosa_R$ são dominadas pelas outras, isto é, apresentam valores maiores ou iguais em todos os quesitos.

Exemplo 2:

Temos em estoque $K = 20$ tipos de objetos com comprimentos entre 2.200 cm e 6.000 cm e disponibilidade de uma unidade para cada tipo (descritos na Tabela 7.4) e uma demanda de 5 tipos de itens, ou seja, $m = 5$, cujos dados estão na Tabela 7.5.

Id Objeto	Tamanho (cm)	Estoque
1	3531	1
2	4088	1
3	5896	1
4	2924	1
5	2440	1
6	4866	1
7	3595	1
8	4549	1
9	3269	1
10	2365	1
11	3791	1
12	4875	1
13	4658	1
14	2529	1
15	4663	1
16	3448	1
17	4973	1
18	4010	1
19	5907	1
20	3050	1

Tabela 7.4: *Dados do Exemplo 2 - Estoque*

Item	Tamanho (cm)	Demanda
1	235	4
2	200	51
3	347	42
4	471	16
5	274	37

Tabela 7.5: *Dados do exemplo 2 - Itens*

Para este exemplo, toda perda superior a 200 *cm* é considerada sobra.

		Construtiva		Residual				
	CUT	FFD_R	$Gulosa_R$	FFD_R	$Gulosa_R$	RAG_R 1	RAG_R 2	RAG_R 3
Objetos cortados	11	13	12	11	11	10	10	10
Perda total	5	10	1	0	3	0	0	0
Sobra total	743	4765	2715	2125	2852	1857	1857	1560
Padrões com Sobra	1	7	4	3	2	1	1	2
Padrões com Perda	3	5	1	0	1	0	0	0

Tabela 7.6: *Solução do Exemplo 2.*

Como podemos observar na Tabela 7.6, as heurísticas RAG_R - versões 1, 2 e 3 utilizam menos objetos que o algoritmo CUT para atender a demanda e assim como ele, as heurísticas RAG_R - versões 1 e 2 concentram a sobra em um único padrão. Com relação às perdas, temos que estas duas heurísticas não geram perdas e as demais, com exceção da heurística construtiva FFD_R , embora não concentrem a sobra em um único padrão, apresentam uma perda menor que o algoritmo CUT. Para este exemplo, temos que a solução das heurísticas RAG_R - versões 1 e 2 são melhores que a apresentada pelo algoritmo CUT e podem ser classificadas como *ideais*.

Exemplo 3:

Temos em estoque $K = 20$ tipos de objetos com comprimentos entre 2.100 *cm* e 5.000 *cm* e disponibilidade de uma unidade para cada tipo, cujos dados estão na Tabela 7.7, e uma demanda de 5 tipos de itens, ou seja, $m = 5$. Os dados relacionados aos itens estão na Tabela 7.8.

Id Objeto	Tamanho (cm)	Estoque
1	2115	1
2	5050	1
3	2963	1
4	3204	1
5	3620	1
6	3469	1
7	2270	1
8	3914	1
9	5905	1
10	3869	1
11	3887	1
12	4350	1
13	5046	1
14	3373	1
15	5611	1
16	2981	1
17	4869	1
18	2117	1
19	2340	1
20	3566	1

Tabela 7.7: *Dados do Exemplo 3 - Estoque*

Item	Tamanho (cm)	Demanda
1	549	39
2	433	27
3	207	43
4	308	39
5	583	2

Tabela 7.8: *Dados do exemplo 3 - Itens*

Para este exemplo, temos que toda perda superior a 207 *cm* é considerada sobra.

		Construtiva		Residual				
	CUT	FFD_R	$Gulosa_R$	FFD_R	$Gulosa_R$	$RAG_R\ 1$	$RAG_R\ 2$	$RAG_R\ 3$
Objetos cortados	15	16	16	15	15	15	16	15
Perda total	104	11	17	3	9	6	3	4
Sobra total	1017	4618	1902	3132	1066	2367	3071	4901
Padrões com Sobra	1	7	2	3	1	2	4	3
Padrões com Perda	10	5	8	3	5	4	3	4

Tabela 7.9: *Solução do Exemplo 3.*

Embora utilize a mesma quantidade de objetos do estoque para atender a demanda, temos que heurística residual $Gulosa_R$ apresenta uma solução melhor que ao algoritmo CUT , pois gera uma perda muito menor e além disso, a sobra também está concentrada em um único padrão. As demais soluções obtidas pelas heurísticas residuais, embora não concentrem as sobras em um único padrão, apresentam uma perda inferior à gerada pelo algoritmo CUT . Para este exemplo, estas soluções heurísticas podem ser classificadas como *aceitáveis*, enquanto que a solução da heurística residual $Gulosa_R$ é considerada *ideal*.

Para os próximos exemplos, temos em estoque $K = 90$ tipos de objetos com comprimentos entre $3.000\ cm$ e $9.000\ cm$, sendo que a disponibilidade é de apenas uma unidade para cada tipo.

Exemplo 4:

Temos uma demanda de 15 tipos de itens, ou seja, $m = 15$. O comprimento de cada $i, i = 1, \dots, 15$ e sua respectiva demanda estão na Tabela 7.10.

Item	Tamanho (cm)	Demanda
1	569	34
2	718	26
3	520	25
4	540	12
5	492	30
6	547	2
7	632	6
8	430	36
9	750	7
10	387	20
11	804	3
12	389	32
13	835	18
14	684	39
15	687	10

Tabela 7.10: *Dados do exemplo 4 - Itens*

O comprimento aceitável para a sobra deve ser superior a 387 *cm*.

		Construtiva		Residual				
	<i>CUT</i>	<i>FFD_R</i>	<i>Gulosa_R</i>	<i>FFD_R</i>	<i>Gulosa_R</i>	<i>RAG_R 1</i>	<i>RAG_R 2</i>	<i>RAG_R 3</i>
Objetos cortados	27	34	30	27	25	22	22	22
Perda total	2	18	0	7	0	0	0	0
Sobra total	1456	1883	4768	877	4850	3098	3098	1943
Padrões com Sobra	1	3	6	1	3	1	1	1
Padrões com Perda	2	12	0	5	0	0	0	0

Tabela 7.11: *Solução do Exemplo 4.*

Para este exemplo em que temos 90 tipos de objetos disponíveis em estoque, podemos observar pela Tabela 7.11 que as soluções apresentadas pelas heurísticas RAG_R - versões 1, 2 e 3 podem ser consideradas *ideais*, pois concentram as sobras em um único padrão e não geram perdas, entretanto, o algoritmo *CUT*, embora concentre as sobras em um único padrão, gera uma pequena perda e utiliza mais objetos do estoque para atender a demanda.

Exemplo 5:

A demanda para este exemplo é de 15 tipos de itens, cujos dados estão apresentados na Tabela 7.12.

Item	Tamanho (cm)	Demanda
1	548	36
2	492	45
3	641	33
4	414	24
5	303	46
6	861	16
7	498	23
8	748	23
9	733	39
10	658	16
11	570	39
12	898	21
13	327	47
14	801	40
15	471	32

Tabela 7.12: *Dados do exemplo 5 - Itens*

Para este exemplo, o comprimento aceitável para a sobra deve ser superior a 303 *cm*.

		Construtiva		Residual				
	<i>CUT</i>	<i>FFD_R</i>	<i>Gulosa_R</i>	<i>FFD_R</i>	<i>Gulosa_R</i>	<i>RAG_R 1</i>	<i>RAG_R 2</i>	<i>RAG_R 3</i>
Objetos cortados	37	49	45	39	38	34	34	34
Perda total	3	20	0	5	0	0	0	1
Sobra total	2366	9104	1568	454	6753	635	635	2489
Padrões c/ Sobra	1	4	3	1	2	1	1	1
Padrões c/ Perda	2	17	0	4	0	0	0	1

Tabela 7.13: *Solução do Exemplo 5.*

Pela Tabela 7.13, podemos observar que as soluções apresentadas pelas heurísticas RAG - versões 1, 2 e 3 são melhores que a solução apresentada pelo algoritmo CUT,

pois utilizam menos objetos para atender a demanda, as sobras estão concentradas em um único padrão e com excessão da heurística RAG_R - versão 3 que apresenta um perda de 1 *cm*, as outras duas não apresentam perdas. Observe que estas soluções dominam a solução CUT.

Exemplo 6:

Neste exemplo também temos uma demanda de 15 tipos de itens, cujos dados estão apresentados na Tabela 7.14.

Item	Tamanho (cm)	Demanda
1	861	3
2	359	37
3	312	52
4	826	31
5	598	35
6	738	42
7	538	29
8	817	15
9	661	33
10	678	14
11	642	35
12	670	44
13	899	12
14	525	34
15	510	34

Tabela 7.14: *Dados do exemplo 6 - Itens*

Para este exemplo, o limite inferior para sobra é 312 *cm*, ou seja, toda perda superior a 312 *cm* é considerada sobra.

		Construtiva		Residual				
	CUT	FFD_R	$Gulosa_R$	FFD_R	$Gulosa_R$	RAG_R 1	RAG_R 2	RAG_R 3
Objetos cortados	43	49	46	39	40	35	35	34
Perda total	1	16	0	3	0	0	0	0
Sobra total	2080	3962	5094	3081	4786	2987	2987	1990
Padrões com Sobra	1	2	4	2	2	1	1	1
Padrões com Perda	1	13	0	3	0	0	0	0

Tabela 7.15: *Solução do Exemplo 6.*

Como podemos observar, a heurística RAG_R - versão 3 apresenta a melhor solução, sendo superior ao algoritmo CUT , pois utiliza menos objetos do estoque para atender a demanda, não gera perda e a sobra está concentrada em um único padrão. As heurísticas RAG_R - versões 1 e 2 também apresentam soluções *ideais*, utilizando apenas um objeto a mais que a heurística RAG_R - versão 3 para atender a demanda. As demais heurísticas, apresentam soluções *aceitáveis*.

Exemplo 7:

Para este exemplo, temos uma demanda de 15 tipos de itens. O comprimento de cada item $i, i = 1 \dots, 15$ e sua respectiva demanda estão apresentados na Tabela 7.16.

Item	Tamanho (cm)	Demanda
1	858	33
2	521	22
3	885	28
4	650	43
5	741	45
6	578	10
7	476	21
8	467	17
9	655	47
10	887	36
11	716	16
12	467	27
13	655	48
14	887	47
15	716	10

Tabela 7.16: *Dados do exemplo 7 - Itens*

Com esta demanda, o comprimento aceitável para a sobra deve ser superior a 427 cm.

		Construtiva		Residual				
	<i>CUT</i>	<i>FFD_R</i>	<i>Gulosa_R</i>	<i>FFD_R</i>	<i>Gulosa_R</i>	<i>RAG_R 1</i>	<i>RAG_R 2</i>	<i>RAG_R 3</i>
Objetos cortados	50	56	57	49	46	43	43	44
Perda total	8	29	4	6	1	0	0	1
Sobra total	923	8092	15273	3029	1267	607	607	4519
Padrões com Sobra	1	9	14	2	1	1	1	2
Padrões com Perda	8	20	3	6	1	0	0	1

Tabela 7.17: *Solução do Exemplo 7.*

Pela Tabela 7.17, observamos que as soluções das heurísticas construtivas FFD_R e $Gulosa_R$, apresentam pequenas perdas, porém possuem uma solução *indesejável*, visto que apresentam sobras em muitos padrões, entretanto, as heurísticas RAG_R - versões 1 e 2 apresentam soluções *ideais*, sendo superior a obtida pelo algoritmo CUT. As demais

heurísticas apresentam soluções *aceitáveis* para este exemplo.

Exemplo 8:

Temos uma demanda de 15 tipos de itens, cujos dados referentes a comprimento e demanda estão apresentados na Tabela 7.18.

Item	Tamanho (cm)	Demanda
1	757	44
2	307	29
3	665	53
4	805	23
5	433	11
6	457	62
7	387	42
8	541	14
9	308	2
10	506	47
11	828	32
12	822	23
13	442	35
14	779	15
15	719	18

Tabela 7.18: *Dados do exemplo 8 - Itens*

O comprimento aceitável para a sobra deve ser superior a 307 *cm*.

		Construtiva		Residual				
	CUT	FFD_R	$Gulosa_R$	FFD_R	$Gulosa_R$	RAG_R 1	RAG_R 2	RAG_R 3
Objetos cortados	43	51	45	41	40	36	36	36
Perda total	5	19	1	4	0	0	0	0
Sobra total	767	4976	3605	1418	4195	3608	3608	1334
Padrões com Sobra	1	7	6	2	2	2	2	1
Padrões com Perda	2	16	1	4	0	0	0	0

Tabela 7.19: *Solução do Exemplo 8.*

Para este exemplo, podemos observar que a heurística RAG - versão 3 apresentou uma solução *ideal*, superior a solução do algoritmo CUT, enquanto que as demais heurísticas residuais apresentaram soluções *aceitáveis*.

De modo geral, pudemos observar que em todos os exemplos as heurísticas residuais apresentaram as melhores soluções, e em todos eles, ao menos uma heurística apresentou solução superior ao algoritmo CUT.

7.2 Gerador Aleatório

O gerador aleatório desenvolvido é baseado em Poldi [22], sendo que algumas modificações foram realizadas e outros parâmetros de entrada foram necessários. Os dados para o gerador são mostrados a seguir:

- *Número de tipos de objetos padronizados:* $\bar{k} = 2$;
- *Número de tipos de objetos retalhos:* $k = 3, 5$ e 7 ;
- *Número total de objetos disponíveis em estoque:* $K = 5, 7$ e 9 ;
- *Número de tipos de itens:* $m = 10, 20$ e 40 ;
- *Disponibilidades em estoque dos objetos padronizados:* $e_1 = e_2 = 100$ objetos;
- *Disponibilidades em estoque dos objetos retalhos:* Os valores de $e_k, k = 3, \dots, K$, são gerados aleatoriamente no intervalo $[1, 10]$;

- *Comprimentos dos objetos padronizados:* $L_1 = 1000$ e $L_2 = 1100$;
- *Comprimentos dos itens:* Os itens de comprimento ℓ_i são gerados aleatoriamente no intervalo $[v_1 L, v_2 L]$, em que L é o valor médio de $L_k, k = 1, 2, v_1 = 0,01, v_2 = 0,2$ e $0,8$. Combinando estes valores, classes de exemplos são geradas, em que para itens pequenos consideramos $v_2 = 0,2$ e para itens médios $v_2 = 0,8$;
- *Tamanho aceitável para a sobra:* $\left[\delta = \frac{1}{m} \sum_{i=1}^m \ell_i \right]$;
- *Comprimentos dos objetos retalhos:* $L_k, k = 3, \dots, K$ são gerados aleatoriamente no intervalo de $[\delta, \frac{L_1}{2}]$;
- *Demanda:* a demanda d_i é gerada aleatoriamente no intervalo $\left[0, 02 \frac{\sum_{k=1}^K e_k L_k}{\sum_{i=1}^m \ell_i}, \frac{\sum_{k=1}^K e_k L_k}{\sum_{i=1}^m \ell_i} \right]$,
 $i = 1, \dots, m$. Combinado estes valores, garantimos que o comprimento total dos itens demandados não exceda o comprimento total dos objetos em estoque;
- *Porcentagem de perda aceitável para os objetos padronizados:* $0.005L_k, k = 1, 2$;
- *Porcentagem de perda aceitável para os objetos retalhos:* $0.05L_k, k = 3, \dots, K$;

Na construção do gerador aleatório, definimos uma porcentagem de perda maior para os objetos retalhos, visto que estes devem ter maior prioridade para serem utilizados.

Neste trabalho, 16 classes de exemplos foram criadas combinando os parâmetros $K = (5, 7 \text{ ou } 9)$, $m = (10 \text{ e } 20 \text{ ou } 40)$ e ℓ_i (pequeno e médio).

7.3 Resultados e Análises

Na Tabela 7.20 a seguir, estão descritas as 16 classes de exemplos criadas, sendo que, 20 exemplos foram gerados aleatoriamente para cada uma destas classes. Na Tabela 7.21, apresentamos o número médio de objetos retalhos utilizados pelas heurísticas em cada classe e, na Tabela 7.22 número médio de objetos padronizados utilizados. Na Tabela 7.23, mostramos a perda média de cada classe, as quais devem ser minimizadas, na

Tabela 7.24 mostramos as sobras médias geradas por cada classe, as quais têm comprimentos suficientemente grandes para voltar ao estoque. As Tabelas 7.25, 7.26 e 7.27 são referentes às médias das quantidades de padrões de corte com sobra, com perda e sem perda respectivamente. O tempo médio computacional é dado em segundos e apresentado na Tabela 7.28. Em todas Tabelas apresentadas, os piores resultados estão escritos em itálico e os melhores resultados encontram-se em negrito.

	Parâmetros		
Class	K	m	Itens
1	5	10	P
2	5	10	M
3	5	20	P
4	5	20	M
5	5	40	P
6	5	40	M
7	7	10	P
8	7	10	M
9	7	20	P
10	7	20	M
11	7	40	P
12	7	40	M
13	9	10	P
14	9	10	M
15	9	20	P
16	9	20	M

Tabela 7.20: *Descrição das classes*

	Número de Padrões de Corte com Objetos Retalhos													
	Construtiva				Residual									
	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>RAG 1</i>	<i>RAG_R 1</i>	<i>RAG 2</i>	<i>RAG_R 2</i>	<i>RAG 3</i>	<i>RAG_R 3</i>
1	9, 90	8, 15	7, 20	8, 25	9, 35	6, 85	4, 45	4, 80	<i>3, 55</i>	5, 15	4, 00	5, 05	4, 70	6, 45
2	7, 70	5, 35	3, 90	4, 25	5, 15	4, 05	5, 00	5, 20	3, 75	4, 75	4, 85	4, 95	<i>3, 25</i>	3, 65
3	7, 75	7, 75	5, 85	7, 60	7, 55	6, 55	4, 50	4, 35	4, 20	4, 80	4, 45	4, 80	<i>3, 80</i>	4, 60
4	9, 40	6, 30	6, 00	5, 50	6, 35	4, 10	4, 00	3, 35	<i>2, 25</i>	3, 50	3, 15	3, 60	2, 65	3, 10
5	7, 15	6, 65	6, 85	6, 55	7, 10	5, 30	6, 05	5, 00	3, 60	4, 20	3, 70	4, 20	<i>3, 45</i>	3, 70
6	9, 00	5, 60	8, 85	8, 25	7, 10	3, 40	2, 65	3, 45	<i>2, 35</i>	3, 20	2, 55	3, 15	2, 65	3, 40
7	12, 90	8, 00	9, 80	12, 20	12, 05	7, 40	4, 90	5, 05	<i>3, 65</i>	4, 65	3, 95	4, 70	4, 00	4, 80
8	11, 00	8, 05	9, 40	9, 80	8, 25	5, 95	5, 05	5, 85	<i>2, 55</i>	5, 15	3, 30	4, 90	3, 70	6, 65
9	11, 10	8, 15	8, 75	11, 80	8, 90	7, 40	5, 45	5, 25	<i>2, 70</i>	3, 75	3, 75	3, 05	3, 05	4, 80
10	10, 70	10, 70	8, 70	10, 00	10, 10	6, 70	2, 85	3, 60	3, 05	3, 95	3, 70	3, 90	<i>2, 80</i>	3, 95
11	7, 50	5, 45	7, 40	7, 80	7, 35	6, 90	5, 20	4, 65	<i>4, 30</i>	4, 90	4, 55	4, 90	5, 10	5, 30
12	13, 15	6, 65	8, 10	12, 40	9, 25	4, 40	4, 20	4, 30	2, 70	3, 15	2, 80	3, 15	<i>2, 30</i>	3, 10
13	11, 90	7, 30	11, 20	14, 30	8, 55	5, 80	4, 80	5, 00	<i>3, 75</i>	4, 45	3, 90	4, 45	<i>3, 75</i>	4, 50
14	14, 00	8, 65	14, 65	20, 80	10, 10	5, 80	4, 32	5, 00	<i>3, 45</i>	5, 00	5, 20	6, 65	6, 35	9, 80
15	9, 55	6, 80	11, 70	13, 95	7, 45	7, 40	5, 40	6, 00	<i>2, 75</i>	3, 10	3, 10	3, 10	3, 10	3, 70
16	15, 25	10, 80	13, 45	13, 95	14, 40	8, 85	8, 85	9, 90	<i>3, 50</i>	4, 10	3, 65	4, 30	4, 30	5, 35

Tabela 7.21: Média dos 20 exemplos em cada uma das 16 classes.

Na Tabela 7.21, podemos observar que as heurísticas FFD, FFD_R e $Gulosa_R$ são as que mais utilizam objetos retalhos na construção dos padrões de corte, em especial, a heurística FFD. Por outro lado, as heurísticas RAG - versões 1 e 3 são as que menos utilizam objetos retalhos, ou seja, são heurísticas que permitem que o estoque de retalhos aumente muito. Considerando as heurísticas para o problema de reaproveitamento, observamos que as Heurísticas FFD_R e $Gulosa_R$ são as que apresentam maior aproveitamento de objetos retalhos, enquanto que as Heurísticas RAG_R - versões 1,2 e 3 são as que apresentam menor utilização destes objetos. Observe que existe grandes discrepância entre as soluções.

	Número de Padrões de Corte com Objetos Padronizados													
	Construtiva				Residual									
	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>RAG 1</i>	<i>RAG_R 1</i>	<i>RAG 2</i>	<i>RAG_R 2</i>	<i>RAG 3</i>	<i>RAG_R 3</i>
1	106,00	107,80	<i>109,25</i>	109,05	102,00	102,50	102,65	102,70	102,10	102,05	102,10	102,05	101,80	101,75
2	100,20	111,05	105,10	<i>117,75</i>	99,60	100,05	99,70	100,25	99,75	103,90	99,65	103,60	99,95	104,40
3	103,55	105,70	<i>106,65</i>	106,45	98,60	99,70	99,75	99,80	98,65	98,65	98,95	98,75	98,75	99,85
4	100,10	104,10	105,40	<i>116,30</i>	97,80	98,80	98,70	99,35	98,80	100,70	98,65	100,15	98,60	99,80
5	103,45	105,05	<i>105,45</i>	105,35	97,20	98,25	98,35	98,15	97,65	97,40	97,65	97,40	97,60	97,25
6	100,25	103,50	102,95	<i>106,35</i>	96,20	97,70	97,70	97,65	97,10	97,05	97,10	97,05	97,05	97,10
7	108,75	111,40	<i>111,65</i>	111,25	104,10	104,90	105,00	105,25	104,20	104,25	104,20	104,25	104,15	104,15
8	100,35	103,45	103,60	<i>105,15</i>	98,20	98,70	98,90	98,95	99,05	98,75	98,85	98,75	98,80	98,70
9	98,80	101,25	<i>101,30</i>	101,05	94,15	94,65	94,85	94,75	94,10	94,15	94,10	94,15	94,05	94,00
10	104,80	106,00	<i>110,95</i>	110,55	99,80	101,05	104,85	104,80	104,45	104,35	104,30	104,35	104,55	104,80
11	112,35	113,45	<i>113,55</i>	113,00	106,00	106,25	106,35	106,30	105,00	105,05	105,00	105,05	104,95	104,90
12	101,40	105,95	<i>106,20</i>	106,05	98,35	100,30	100,45	100,45	99,85	99,90	99,85	99,90	100,10	100,00
13	111,90	115,40	<i>116,05</i>	115,85	109,75	110,35	110,45	110,55	109,75	109,85	109,75	109,85	109,75	109,70
14	106,30	107,90	106,70	<i>108,10</i>	102,60	103,25	104,00	104,00	103,65	103,50	103,55	103,50	103,65	103,55
15	109,05	110,90	<i>110,95</i>	110,45	104,00	104,30	104,45	104,55	104,10	104,10	104,15	104,10	104,10	104,15
16	109,40	111,30	112,75	<i>115,90</i>	104,75	106,60	106,85	106,85	106,55	106,55	106,55	106,55	106,35	106,20

Tabela 7.22: Média dos 20 exemplos em cada uma das 16 classes.

Como podemos observar, a heurística residual FFD e a RAG_R - versão 3 apresentam melhores soluções em termos de objetos padronizados utilizados, entretanto, as heurísticas Gulosa e $Gulosa_R$, pela facilidade de alocar (combinar) itens em objetos com comprimentos maiores, apresentam soluções piores. Entre as heurísticas para o problema de reaproveitamento, a heurística que apresenta melhor resultado (menor uso de objetos padronizados) é a heurística RAG_R - versão 3. Observe que as discrepâncias não são muito grandes.

	Perda total													
	Construtiva				Residual									
	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>RAG 1</i>	<i>RAG_R 1</i>	<i>RAG 2</i>	<i>RAG_R 2</i>	<i>RAG 3</i>	<i>RAG_R 3</i>
1	230, 50	21, 55	157, 15	10, 50	31, 05	7, 25	38, 30	4, 25	17, 55	4, 35	18, 60	4, 30	30, 75	12, 85
2	2056, 15	85, 80	2148, 10	52, 95	578, 30	380, 05	553, 50	378, 35	492, 20	106, 90	552, 55	111, 20	537, 35	104, 15
3	101, 90	8, 55	124, 50	12, 10	17, 95	5, 50	33, 55	2, 65	25, 70	4, 05	29, 75	4, 10	24, 80	5, 55
4	832, 45	83, 30	1451, 45	46, 80	167, 00	58, 95	232, 45	54, 55	132, 85	29, 80	194, 45	32, 75	174, 10	41, 95
5	40, 30	5, 80	95, 20	17, 25	12, 35	2, 45	72, 45	4, 45	21, 90	3, 30	21, 45	3, 30	22, 75	3, 55
6	324, 80	29, 30	765, 00	69, 95	54, 05	6, 25	85, 55	7, 35	46, 65	6, 35	44, 95	5, 80	79, 85	7, 85
7	160, 35	30, 00	62, 70	18, 35	23, 15	4, 15	19, 30	1, 90	12, 05	3, 45	11, 70	3, 45	15, 00	6, 10
8	1524, 35	76, 75	858, 05	38, 65	83, 95	12, 35	91, 75	9, 70	47, 05	13, 35	69, 45	11, 20	64, 20	15, 25
9	45, 40	12, 05	57, 10	10, 70	9, 00	3, 75	8, 70	1, 55	10, 75	1, 65	19, 55	1, 65	11, 10	3, 70
10	989, 35	57, 85	721, 30	42, 55	102, 15	9, 90	51, 25	6, 25	90, 25	5, 00	111, 75	4, 25	90, 55	4, 90
11	12, 00	8, 70	59, 65	11, 25	6, 50	1, 35	20, 25	1, 70	14, 10	1, 45	14, 75	1, 20	18, 45	1, 20
12	160, 90	15, 95	264, 35	35, 75	33, 90	2, 90	128, 40	7, 30	51, 05	1, 95	42, 65	1, 95	45, 70	4, 10
13	94, 45	19, 55	76, 35	21, 50	6, 85	5, 32	5, 35	1, 50	9, 00	1, 55	8, 80	1, 55	9, 35	4, 25
14	1376, 70	73, 80	659, 80	59, 05	69, 60	5, 05	39, 30	5, 75	34, 20	6, 70	38, 80	6, 95	63, 25	15, 05
15	34, 60	13, 20	99, 65	29, 40	3, 45	2, 45	6, 80	3, 25	9, 70	2, 15	12, 90	2, 10	7, 50	1, 10
16	748, 55	32, 85	900, 65	40, 80	74, 50	6, 05	28, 30	7, 65	50, 05	4, 60	61, 35	4, 60	39, 95	8, 85

Tabela 7.23: Média dos 20 exemplos em cada uma das 16 classes.

Pela Tabela 7.23, observamos que as heurísticas residuais apresentam melhores soluções quando a perda é analisada, com grandes discrepâncias, como já observado por Poldi e Arenales [23]. Um fato que devemos destacar, é que mesmo permitindo uma perda maior para os objetos retalhos (uma maneira destes terem preferência), as soluções apresentadas pelas heurísticas específicas para o problema com reaproveitamento de sobras são, entre as heurísticas residuais, as que apresentam melhores soluções. Entre as heurísticas de reaproveitamento, podemos observar que as piores soluções são das heurísticas FFD_R e $Gulosa_R$, enquanto que para as heurísticas clássicas para o problema de corte, as piores soluções são das heurísticas FFD e $Gulosa$. Uma consequência das heurísticas de reaproveitamento quando priorizamos a redução das perdas, é o aumento na quantidade de padrões que geram sobras, como veremos na Tabela 7.24, pois sempre que uma perda *indesejável* não pode ser melhorada, uma sobra no padrão é gerada.

	Sobra Total													
	Construtiva				Residual									
	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>RAG 1</i>	<i>RAG_R 1</i>	<i>RAG 2</i>	<i>RAG_R 2</i>	<i>RAG 3</i>	<i>RAG_R 3</i>
1	515, 25	600, 45	610, 75	1132, 30	279, 10	421, 75	165, 50	422, 00	49, 35	251, 20	82, 75	242, 55	29, 55	236, 00
2	350, 95	10492, 55	840, 40	18940, 05	122, 65	553, 50	180, 30	1079, 50	12, 40	4618, 45	34, 80	4335, 55	36, 55	4709, 60
3	483, 75	531, 45	328, 60	840, 90	270, 85	544, 50	241, 40	384, 55	23, 95	216, 05	44, 75	215, 35	65, 05	269, 90
4	484, 00	2831, 40	1086, 15	15985, 50	190, 70	612, 25	147, 00	1430, 65	103, 95	2582, 05	65, 25	2096, 00	9, 70	1585, 70
5	495, 65	592, 95	299, 30	613, 30	420, 15	307, 15	279, 60	269, 90	40, 95	309, 20	45, 75	309, 20	35, 30	243, 20
6	466, 85	789, 85	361, 40	6088, 05	343, 80	339, 90	38, 80	574, 65	83, 10	338, 65	112, 05	330, 80	43, 55	460, 20
7	390, 95	575, 55	590, 40	921, 05	350, 70	389, 85	173, 45	488, 45	33, 80	144, 15	33, 70	145, 80	48, 00	142, 90
8	484, 30	3681, 00	1207, 40	5649, 70	321, 45	526, 60	397, 75	829, 15	62, 45	528, 10	85, 05	502, 65	85, 25	863, 70
9	509, 15	391, 00	198, 95	810, 90	393, 15	264, 40	273, 75	231, 35	31, 80	143, 40	55, 45	143, 40	35, 85	122, 50
10	393, 80	858, 40	2023, 90	5179, 20	88, 70	367, 70	95, 10	376, 45	14, 25	303, 25	6, 75	311, 25	30, 20	490, 15
11	549, 60	383, 85	244, 75	523, 85	486, 30	479, 25	272, 80	330, 00	30, 55	175, 50	34, 15	175, 75	30, 10	249, 20
12	627, 55	537, 50	427, 30	2426, 05	276, 80	355, 55	194, 55	518, 25	72, 20	291, 30	85, 75	291, 30	111, 05	352, 25
13	414, 50	453, 20	312, 30	925, 30	205, 15	280, 70	164, 90	278, 65	31, 80	176, 60	42, 00	176, 60	17, 90	155, 90
14	378, 25	73, 80	447, 55	5574, 95	207, 80	307, 95	240, 95	433, 05	32, 35	266, 25	47, 15	308, 25	25, 85	653, 10
15	516, 70	482, 95	410, 10	811, 45	352, 60	318, 90	157, 80	189, 85	18, 40	110, 70	50, 60	110, 70	49, 60	60, 25
16	439, 25	745, 70	1132, 15	5829, 70	226, 00	302, 90	774, 55	978, 95	10, 90	219, 35	10, 90	219, 35	34, 95	189, 20

Tabela 7.24: Média dos 20 exemplos em cada uma das 16 classes.

Quando perdas *aceitáveis* não são obtidas, sobras tornam-se também um objetivo desejado, visto que estas podem ser utilizadas para atender demandas futuras. Assim, como observamos na Tabela 7.24, as soluções apresentadas pela heurística construtiva $Gulosa_R$ poderiam ser as mais apropriadas, pois apresentam maiores sobras que podem ser utilizadas com mais facilidade no futuro. Porém, a seleção de uma solução baseada tão somente nas sobras, pode nos levar a um equívoco. Por exemplo, na classe C2 a maior sobra é gerada pela heurística $Gulosa_R$ (18.940,05), entretanto, considerando o número de padrões de corte com sobra (Tabela 7.25), temos que esta solução é ruim, pois apresenta maior distribuição das sobras (5,65 objetos retalhos).

Entretanto, pela definição de solução *ideal*, desejamos apenas um padrão de corte com sobra ou então, para que seja uma solução *aceitável*, poucos padrões com sobras. Considerando estes critérios (solução *ideal* e *aceitável*) e relacionando a sobra (Tabela 7.24) com a quantidade de padrões de corte com sobra (Tabela 7.25), observamos que a heurística $Gulosa_R$ não teria a solução *ideal* nem *aceitável*, pois apresenta maior quantidade de padrões com sobra. Desta forma, para a escolha da melhor solução na Tabela 7.24, algumas características devem ser consideradas. Se objetivarmos perda mínima, a análise da sobra deve ser relacionada com as menores perdas (Tabela 7.23), ou seja, nossa escolha estaria entre as heurísticas que apresentam menores perdas, $Gulosa_R$, RAG_R - versões 1, 2 e 3. No entanto, se o objetivo fosse a redução do estoque de retalhos, para a escolha da melhor solução, teríamos que relacionar as sobras (Tabela 7.24) com o número de padrões de corte com objetos retalhos (Tabelas 7.21) e a quantidade de padrões com sobra (Tabela 7.25).

Como podemos observar, a escolha do melhor método heurístico para resolvermos problemas de corte com reaproveitamento não é trivial, pois envolve a análise de várias soluções simultaneamente.

	Número de Padrões de Corte com Sobra													
	Construtiva				Residual									
	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>RAG 1</i>	<i>RAG_R 1</i>	<i>RAG 2</i>	<i>RAG_R 2</i>	<i>RAG 3</i>	<i>RAG_R 3</i>
1	1,00	1,40	1,20	<i>2, 10</i>	0,60	1,35	0,30	1,60	0,55	1,50	0,70	1,45	0,35	1,40
2	0,65	4,40	0,50	<i>5, 65</i>	0,30	1,60	0,30	2,40	0,05	3,35	0,05	3,15	0,10	3,15
3	0,95	1,10	0,70	<i>2, 30</i>	0,70	1,40	0,70	1,40	0,30	1,50	0,35	1,50	0,55	1,30
4	0,85	3,55	1,15	<i>6, 65</i>	0,45	1,80	0,15	2,60	0,20	3,15	0,10	2,90	0,05	2,50
5	0,95	1,00	0,70	<i>1, 70</i>	0,75	1,25	0,65	1,65	0,60	1,65	0,60	1,65	0,55	1,60
6	0,85	1,90	0,75	<i>6, 80</i>	0,60	1,25	0,15	1,75	0,35	1,40	0,50	1,35	0,20	1,55
7	0,90	1,10	1,00	<i>2, 25</i>	0,60	1,25	0,35	1,25	0,50	1,25	0,50	1,25	0,50	1,10
8	0,90	4,45	1,60	<i>5, 30</i>	0,85	2,35	0,85	2,50	0,20	1,90	0,40	1,85	0,45	2,50
9	0,90	1,05	0,60	<i>2, 45</i>	0,85	1,10	0,75	1,10	0,45	1,15	0,50	1,15	0,45	1,40
10	0,85	1,95	2,65	<i>6, 35</i>	0,80	1,75	0,50	1,65	0,10	1,40	0,05	1,40	0,20	2,15
11	1,00	0,95	0,60	<i>2, 00</i>	0,80	1,05	0,60	1,25	0,40	1,60	0,45	1,60	0,35	1,40
12	0,95	1,25	1,00	<i>4, 05</i>	0,60	1,55	0,65	1,85	0,40	1,40	0,45	1,40	0,55	1,55
13	0,95	1,05	0,85	<i>2, 20</i>	0,75	1,00	0,65	1,00	0,35	1,30	0,40	1,30	0,35	1,05
14	0,90	2,05	0,75	<i>5, 75</i>	0,70	1,45	0,65	1,50	0,25	1,35	0,30	1,45	0,20	1,85
15	1,00	1,10	1,30	<i>2, 85</i>	0,70	1,00	0,65	1,25	0,55	0,90	0,60	0,90	0,60	1,20
16	0,80	1,55	2,30	<i>5, 45</i>	0,55	1,30	1,30	1,40	0,10	1,30	0,10	1,30	0,30	1,15
média	0,90	1,87	1,08	3,97	0,66	1,40	0,58	1,68	0,33	1,63	0,37	1,60	0,37	1,68

Tabela 7.25: Média dos 20 exemplos em cada uma das 16 classes.

Como observamos na Tabela 7.25, as heurísticas de reaproveitamento apresentam maior quantidade de padrões com sobra. Em geral, isto ocorre pelo fato destas heurísticas eliminarem perdas *indesejáveis* tornando-as sobras *aceitáveis*. Por outro lado, como as heurísticas clássicas que têm por objetivo apenas a minimização das perdas, apresentam quantidades de padrões com sobra menores, em especial as Heurísticas RAG - versões 1,2 e 3, que apresentam os melhores resultados, confirmando conclusões em Poldi e Arenales [23].

	Número de Padrões de Corte com Perda													
	Construtiva				Residual									
	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>RAG 1</i>	<i>RAG_R 1</i>	<i>RAG 2</i>	<i>RAG_R 2</i>	<i>RAG 3</i>	<i>RAG_R 3</i>
1	12, 80	3, 20	2, 40	1, 15	4, 25	1, 60	2, 05	1, 00	1, 80	1, 20	2, 05	1, 25	2, 45	1, 55
2	15, 80	4, 75	7, 65	2, 40	7, 50	4, 70	5, 90	3, 95	6, 10	4, 10	6, 60	3, 90	6, 20	3, 65
3	18, 20	2, 30	2, 50	1, 20	4, 80	1, 25	1, 40	0, 60	2, 05	1, 25	2, 10	1, 20	1, 75	1, 00
4	27, 00	8, 15	9, 60	3, 30	10, 00	4, 75	6, 00	3, 70	4, 75	4, 00	5, 65	4, 05	5, 80	4, 15
5	16, 55	1, 95	3, 00	1, 70	3, 85	1, 00	2, 20	0, 95	1, 55	0, 80	1, 65	0, 80	1, 75	0, 85
6	37, 55	6, 45	12, 40	6, 30	7, 75	1, 15	2, 60	1, 40	1, 45	1, 00	1, 40	1, 00	2, 20	1, 35
7	10, 90	4, 05	2, 30	1, 40	3, 45	1, 25	1, 35	0, 40	1, 50	0, 80	1, 60	0, 80	1, 60	0, 90
8	15, 20	5, 55	6, 05	2, 75	5, 10	1, 80	2, 60	1, 40	2, 90	2, 05	3, 05	1, 90	3, 75	2, 75
9	11, 45	3, 30	2, 70	1, 50	2, 80	1, 15	0, 95	0, 60	1, 10	0, 50	1, 90	0, 50	1, 45	0, 95
10	27, 95	8, 20	5, 35	2, 75	8, 25	1, 90	1, 75	0, 95	2, 50	1, 25	2, 90	1, 15	2, 50	0, 95
11	6, 85	3, 45	2, 55	1, 20	1, 80	1, 50	1, 50	0, 60	1, 65	0, 60	1, 85	0, 55	1, 80	0, 55
12	33, 10	4, 55	6, 40	4, 20	5, 25	0, 80	2, 40	1, 25	1, 50	0, 60	1, 45	0, 60	1, 40	0, 70
13	9, 45	4, 15	3, 45	2, 40	1, 75	1, 10	0, 90	0, 60	1, 25	0, 50	1, 30	0, 50	1, 75	1, 05
14	15, 15	5, 85	6, 00	2, 25	5, 10	1, 65	1, 85	1, 00	2, 55	1, 50	2, 65	1, 40	3, 75	2, 30
15	9, 30	3, 30	2, 75	1, 65	1, 25	0, 80	0, 95	0, 55	0, 95	0, 85	1, 20	0, 85	1, 10	0, 45
16	27, 65	7, 95	6, 30	2, 75	7, 20	1, 55	2, 10	1, 50	2, 00	1, 00	2, 20	1, 00	2, 50	1, 65

Tabela 7.26: Média dos 20 exemplos em cada uma das 16 classes.

Pela Tabela 7.26, observamos que na maioria das heurísticas para o problema de reaproveitamento a redução no número de padrões de corte com perda é muito grande quando comparada com esta mesma heurística clássica. Isto ocorre pelo fato de os padrões com perdas *indesejáveis* nas heurísticas clássicas serem alterados, tornando-se padrões com sobras que serão reaproveitadas. Observe que as heurísticas residuais são as que apresentam melhores soluções enquanto que a heurística FFD apresenta maior quantidade de padrões com perdas. Uma consequência destes resultados é o aumento no número de padrões de corte com sobra, como podemos observar na Tabela 7.25, em que as heurísticas clássicas são as que apresentam menores quantidades de padrões com sobra, porém, como estamos interessados no reaproveitamento, podemos notar que as heurísticas residuais $Gulosa_R$, RAG_R - versões 1, 2 e 3 são as heurísticas que apresentam melhores soluções.

	Número de Padrões de Corte sem Perda/Sobra													
	Construtiva				Residual									
	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>RAG 1</i>	<i>RAG_R 1</i>	<i>RAG 2</i>	<i>RAG_R 2</i>	<i>RAG 3</i>	<i>RAG_R 3</i>
1	6, 00	16, 30	17, 20	17, 20	13, 00	14, 70	14, 85	14, 85	14, 05	14, 40	14, 15	14, 40	15, 35	15, 65
2	1, 30	6, 45	7, 80	7, 80	8, 25	9, 60	10, 00	10, 00	7, 55	8, 35	7, 45	7, 90	7, 55	8, 30
3	16, 35	33, 75	32, 10	32, 10	24, 85	27, 60	27, 25	27, 25	26, 35	26, 45	26, 45	26, 50	29, 40	29, 50
4	3, 15	15, 50	17, 95	17, 95	17, 50	20, 85	20, 75	20, 75	17, 45	17, 85	17, 40	17, 50	19, 35	19, 65
5	40, 40	56, 60	55, 10	55, 10	46, 10	48, 05	47, 35	47, 35	44, 60	44, 75	44, 65	44, 75	47, 65	47, 70
6	12, 40	39, 85	36, 60	36, 60	38, 75	43, 10	41, 95	41, 95	35, 45	35, 65	35, 55	35, 65	40, 65	40, 95
7	9, 15	16, 20	18, 15	18, 15	14, 70	15, 10	15, 50	15, 50	14, 75	15, 00	14, 85	15, 10	16, 75	17, 10
8	2, 65	8, 45	11, 55	11, 55	11, 70	13, 10	13, 25	13, 25	11, 00	11, 55	11, 10	11, 60	11, 20	11, 80
9	22, 15	30, 35	31, 75	31, 75	26, 80	27, 50	27, 35	27, 35	26, 30	26, 45	26, 30	26, 45	27, 80	28, 10
10	4, 70	21, 30	23, 80	23, 80	21, 75	25, 60	25, 40	25, 40	21, 65	22, 55	21, 80	22, 55	24, 60	24, 65
11	52, 85	56, 60	58, 05	58, 05	49, 05	49, 35	49, 10	49, 10	47, 25	47, 50	47, 30	47, 55	50, 75	50, 95
12	19, 35	48, 15	45, 35	45, 35	43, 15	45, 10	43, 75	43, 75	37, 90	38, 30	38, 00	38, 30	43, 00	43, 35
13	10, 65	16, 25	18, 00	18, 00	15, 70	15, 25	16, 20	16, 20	15, 35	15, 45	15, 35	15, 45	15, 70	16, 00
14	4, 45	11, 10	12, 80	12, 80	11, 85	13, 90	13, 80	13, 80	12, 30	12, 85	12, 35	12, 80	12, 40	13, 05
15	26, 60	32, 45	33, 75	33, 75	28, 35	28, 70	28, 45	28, 45	27, 15	27, 15	27, 15	27, 15	29, 60	29, 75
16	7, 25	25, 10	26, 55	26, 55	23, 45	25, 95	26, 00	26, 00	23, 95	24, 05	23, 95	24, 05	27, 10	27, 55

Tabela 7.27: Média dos 20 exemplos em cada uma das 16 classes.

Como podemos observar na Tabela 7.27, as heurísticas construtivas e residuais $Gulosa$ e $Gulosa_R$ são as que apresentam maior quantidade de padrões de corte sem perda, ou seja, perda igual a 0. Note que estas heurísticas apresentam a mesma solução, pois como os padrões de corte desta heurística são gerados pela mochila 5.1, sempre que uma perda é *indesejável*, não conseguimos alterar o padrão tornando-a menor ou nula, então neste caso, geramos uma sobra, ou seja, a quantidade de padrões com perda igual a zero é sempre a mesma.

	Tempo Computacional													
	Construtiva				Residual									
	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>FFD</i>	<i>FFD_R</i>	<i>Gulosa</i>	<i>Gulosa_R</i>	<i>RAG 1</i>	<i>RAG_R 1</i>	<i>RAG 2</i>	<i>RAG_R 2</i>	<i>RAG 3</i>	<i>RAG_R 3</i>
cada	0,002	0,078	0,728	0,912	11,950	12,015	12,593	12,616	19,910	19,920	19,857	20,701	98,065	98,359
total	0,6	24,8	232,8	291,8	3824,0	3844,8	4029,8	4037,2	6371,2	6374,4	6354,2	6624,2	31380,8	31474,8

Tabela 7.28: *Tempo médio computacional (em segundos) e total para resolver os 320 exemplos.*

Na linha “**média**” da Tabela 7.28 apresentamos o tempo médio computacional para resolver cada exemplo, num total de 320 exemplos para cada método. Na linha “**total**” apresentamos o tempo computacional para resolver todos os 320 exemplos por cada método.

Capítulo 8

Conclusões e Propostas Futuras

8.1 Conclusões

Neste trabalho definimos um problema de corte de estoque em que as sobras geradas pelo processo de corte são reaproveitadas para atender futuras demandas. Para resolver este problema, alteramos métodos heurísticos clássicos bem conhecidos na literatura para o problema de corte de estoque, os quais têm a minimização das perdas como objetivo e incluímos a possibilidade de sobras para estoque. Um gerador de exemplos para analisar as soluções obtidas também foi construído. Com relação à identificação das soluções melhores, pudemos notar que existe uma grande dificuldade na avaliação do melhor resultado, pois características podem se relacionar umas às outras e tal relação depende da decisão a ser tomada: minimizar perda, minimizar o estoque de sobras, entre outras. Em nossa análise, minimizar as perdas, observamos que as heurísticas modificadas para o reaproveitamento, em especial as heurísticas residual Gulosa_R, RAG_R - versões 1, 2 e 3, apresentam soluções superiores às heurísticas clássicas para o problema de corte de estoque. Como consequência da minimização das perdas tivemos, para estas heurísticas, o aumento na quantidade de padrões de corte com sobras, porém, como elas retornaram ao estoque em quantidades menor que a utilizada para atender a demanda, não seria um fator que desqualificaria nossas soluções. Se compararmos a Tabela 7.21 com a Tabela 7.25 podemos notar que mesmo gerando mais padrões com sobras, em geral, o estoque de sobras diminui consideravelmente, por exemplo, na Classe 10, a Heurística construtiva

FFD_R utilizou 10,70 objetos retalhos durante o processo de corte e retornou 1,95. Neste caso, mesmo produzindo mais padrões de corte com sobra que a Heurística FFD, o estoque de retalhos diminuiu 9,75 objetos. Como podemos observar, as heurísticas de reaproveitamento embora gerem mais sobras que as heurísticas clássicas, apresentam soluções muito satisfatórias quando analisamos as soluções simultaneamente.

8.2 Propostas Futuras

Continuações para este trabalho pode ser:

- ajustar os parâmetros utilizados na resolução do problema de corte com reaproveitamento das sobras de material, como por exemplo, atribuir custos diferenciados aos objetos a serem cortados;
- escrever um modelo matemático para o problema de corte com reaproveitamento das sobras de material;
- investigar trabalhos da literatura que tratam do reaproveitamento de sobras de material para problemas de corte bidimensionais. Até o momento encontra-se na literatura somente o reaproveitamento de sobras para problemas unidimensionais;
- estender as heurísticas desenvolvidas para o problema de corte de estoque unidimensional com reaproveitamento das sobras de material para o problema bidimensional;
- estender as heurísticas desenvolvidas para o problema de corte de estoque unidimensional com reaproveitamento das sobras de material para o problema de corte de estoque multiperíodo.

Bibliografia

- [1] ARENALES, M. N., MORABITO, R., YANASSE, H. H., (editores), O problema de corte e empacotamento, Livro-texto de Mini curso. In: Congresso Nacional de Matemática Aplicada e Computacional, 26, 2004, São João del Rei - MG.
- [2] CHERRI, A. C., ARENALES, M. N., O problema de corte de estoque com reaproveitamento das sobras de material-Heurística FFD Modificada. In: Simpósio Brasileiro de Pesquisa Operacional, 38, 2005 Gramado - RS (2005).
- [3] DYCKHOFF, H., *A typology of cutting and packing problems*. European Journal Operational Research, 44: 145-159, (1990).
- [4] GAU, T., WÄSCHER, *CUTGEN: A problem generator for the standard one-dimensional cutting stock problem*. European Journal of Operational Research, 84: 572-579, (1995).
- [5] GRADISAR, M., JESENKO, J., RESINOVIC, C., *Optimization of roll cutting in clothing industry*. Computers & Operational Research, 10: 945-953, (1997).
- [6] GRADISAR, M., KLJAJIC, M., RESINOVIC, C., JESENKO, J., *A sequential heuristic procedure for one-dimentional cutting*. European Journal of Operational Research, 114: 557-568, (1999).
- [7] GRADISAR, M., RESINOVIC, C., KLJAJIC, M., *A hybrid approach for optimization of one-dimentional cutting*. European Journal of Operational Research, 119: 719-728, (1999).

- [8] GRADISAR, M., TRKMAN, P., *A combined approach to the solution to the general one-dimensional cutting stock problem*. Computers & Operations Research, 32: 1793-1807, (2005).
- [9] GILMORE, P. C., GOMORY, R. E., *A linear programming approach to the cutting stock problem*. Operations Research, 9: 848-859, (1961).
- [10] GILMORY, P. C., GOMORY, R. E., *A linear programming approach to the cutting stock problem - Part II*. Operations Research, 11: 863-888, (1963).
- [11] GILMORY, P. C., GOMORY, R. E., *Multi-stage cutting stock problems of two and more dimensions*. Operations Research, 13: 94-120, (1965).
- [12] HAESSLER, R. W., *Controlling cutting pattern changes in one-dimensional trim loss problems*. Operations Research, 23: 483-493, (1975).
- [13] HAESSLER, R. W., *A note on computational modifications to the Gilmore-Gomory cutting stock algorithm*. Operations Research, 28: 1001-1005, (1980).
- [14] HINXMAN, A., *The trim-loss and assortment problems: a survey*. European Journal of Operational Research, 5: 8-18, (1980).
- [15] LIMEIRA, M. S., YANASSE, H.H., *Uma heurística para o problema de redução de padrões de corte*. Anais da V Oficina Nacional de Problemas de Corte e Empacotamento , São José dos Campos, SP, pp 137-145, (2001).
- [16] LUENBERGER, D.G., *Linear and Nonlinear Programming*. 2.ed. Reading Mass.: Addison-Wesley , 1989, c1984.491p.
- [17] MARTELLO, S., TOTH, P., *Knapsack Problems*. Chichester: Wiley, 1990.296p.
- [18] MORABITO, R., *Uma abordagem em grafo-e-ou para o problema de empacotamento: Aplicação ao carregamento de paletes e contêineres*. Tese de Doutorado, EESC - USP, (1992).
- [19] MORABITO, R., *Problemas de corte e empacotamento*. Livro-texto de Mini curso, Elavio, Montevideu - Uruguai, (2004).

- [20] STADTLER, H., *A one-dimensional cutting stock problem in the Aluminium Industry and its solution*. European Journal of Operational Research, 44: 209-223, (1990).
- [21] PINTO, M. J., *O problema de corte de estoque inteiro*. Dissertação de Mestrado, ICMC - USP, (1999).
- [22] POLDI, K. C., *Algumas extensões do problema de corte de estoque*. Dissertação de Mestrado, ICMC - USP, (2003).
- [23] POLDI, K. C., ARENALES, M. N., *Dealing with small demand in integer cutting stock problems with limited different stock lengths*. Notas do ICMC - Série Computação, número 85, ICMC - USP, (2005).
- [24] VAHRENKAMP, R., *Random search in the one-dimensional cutting stock problem*. European Journal of Operational Research, 95: 191-200, (1996).
- [25] WÄSCHER, G., GAU, T., *Heuristics for the integer one-dimensional cutting stock problem: a computational study*. OR Spektrum, 18: 131-144, (1996).
- [26] WÄSCHER, G., HAUßNER, H., SCHUMANN, H., *An improved typology cutting and packing problems*". European Journal of Operational Research (Edição especial em problemas de corte e empacotamento, a aparecer).

Apêndice A. Métodos Básicos de Corte

Neste apêndice, como requisitos básicos para nosso estudo, revisamos o Método Simplex com a técnica de geração de colunas e também um problema muito conhecido na literatura, o Problema da Mochila, juntamente com dois métodos computacionais para sua resolução: programação dinâmica e enumeração implícita.

A.1. Método Simplex com Geração de Colunas Aplicado ao Problema de Corte

A condição de integralidade sobre as variáveis x_j e x_{jk} torna os problemas de corte (3.3) e (3.7) difíceis de serem resolvido computacionalmente quando m é da ordem de algumas dezenas. Uma estratégia para contornar esta dificuldade é relaxar a condição de integralidade sobre as variáveis x_j e x_{jk} e resolver o problema relaxado resultante utilizando o procedimento de geração de colunas, proposto por Gilmore e Gomory [9]. Esta estratégia é uma variante do método simplex que tem sido adotada para a relaxação do problema de corte de estoque e consiste basicamente em, a cada iteração do método simplex, substituir um dos padrões básicos (coluna) por um novo padrão de corte (coluna) que melhora a solução básica corrente. Tal padrão de corte (coluna), é determinado resolvendo-se um Problema da Mochila com as restrições de capacidade dadas em (3.1 ou 3.5). A partir da solução ótima do problema relaxado, que geralmente não é inteira, determina-se uma solução inteira para o problema de corte de estoque original. Esta

solução inteira é determinada por procedimentos heurísticos que vêm sendo desenvolvidos por vários pesquisadores na área: Wäscher e Gau [25], Poldi [22], Stadtler [20] entre outros. No Capítulo (6), alguns desses procedimentos heurísticos são apresentados.

Método Simplex e a Técnica de Geração de Colunas

Geralmente, em problemas práticos, o número de variáveis n (*colunas*), que determina o número de padrões de corte, é muito grande. Para tanto, o método simplex que determina, a cada iteração, uma nova coluna não básica para entrar na base dentre todas não-básicas, torna-se um problema impraticável devido ao grande número de colunas que devem ser investigadas. Uma técnica bastante eficiente para contornarmos este tipo de problema, é a *Geração de Colunas*, que foi proposta por Gilmore e Gomory [9].

Por simplicidade, consideremos o problema de otimização linear (3.3). O Método Simplex com a Técnica de Geração de Colunas consiste em:

Seja $\mathbf{B}$ uma matriz básica. A partir de uma base inicial, numa iteração qualquer do Método Simplex, calculamos a solução básica por:

$$x_B = B^{-1}b \quad (a.1)$$

e o vetor multiplicador simplex:

$$\pi^T = c_B^T B^{-1} \quad (a.2)$$

em que, c_B é o vetor dos custos básicos.

O próximo passo do Método Simplex, consiste em determinarmos uma coluna a entrar na base. Essa escolha é realizada selecionando-se o custo relativo ($\hat{c}_j = c_j - \pi^T a_j$) que apresente um valor negativo. Uma escolha usual, chamada regra de Dantzig, que procura a variável x_k com o menor custo relativo é dada por:

$$c_k - \pi^T a_k = \min\{c_j - \pi^T a_j, j = 1, 2, \dots, n\} \quad (a.3)$$

Se o custo relativo, $c_k - \pi a_k \geq 0$ (as variáveis básicas, incluídas no cálculo do mínimo acima, têm custos relativos nulos), então a solução básica (a.1) é ótima, caso contrário, a coluna a_k entra na base.

Se considerarmos como objetivo minimizar o total de objetos cortados (modelo 3.3), teremos o vetor custo $\mathbf{c} = (1, 1, \dots, 1)$ assim:

$$\min\{c_j - \pi^T a_j\} = 1 - \max\{\pi^T a_j\} \quad (a.4)$$

Desta forma, encontrar uma coluna não-básica, digamos $(\alpha_1, \alpha_2, \dots, \alpha_m)^T$ que substituirá uma coluna básica, corresponde a resolver o problema:

$$\begin{aligned} &\text{maximizar} && \pi_1 \alpha_1 + \pi_2 \alpha_2 + \dots + \pi_m \alpha_m \\ &\text{sujeito a que} && (\alpha_1, \alpha_2, \dots, \alpha_m)^T \text{ corresponda a um padrão de corte.} \end{aligned} \quad (a.5)$$

Para que a coluna $\mathbf{a}_j = (\alpha_1, \alpha_2, \dots, \alpha_m)^T$ corresponda a um padrão de corte, basta que satisfaça as restrições:

$$\begin{aligned} &\ell_1 \alpha_1 + \ell_2 \alpha_2 + \dots + \ell_m \alpha_m \leq L \\ &0 \leq \alpha_i \leq d_i, i = 1, \dots, m \text{ e inteiro.} \end{aligned} \quad (a.6)$$

Assim, o problema (a.5) poderá ser reescrito como:

$$\begin{aligned} &\text{maximizar} && g(a) = \sum_{i=1}^m \pi_i \alpha_i \\ &\text{sujeito a :} && \left\{ \begin{aligned} &\sum_{i=1}^m \ell_i \alpha_i \leq L \\ &0 \leq \alpha_i \leq d_i \text{ e inteiro } i = 1, \dots, m \end{aligned} \right. \end{aligned} \quad (a.7)$$

que é um problema da mochila.

No caso em que $c_j = L - \sum_{i=1}^m \ell_i a_i$, ou seja, o objetivo for minimizar a perda total, o problema (a.7) é ligeiramente modificado.

Como a solução obtida por (a.7) nos fornece a coluna $(\alpha_1, \alpha_2, \dots, \alpha_m)^T$ com o menor custo relativo, podemos garantir que :

- Se $\hat{c} = 0$, a solução é ótima;
- Se $\hat{c} < 0$, a coluna $(\alpha_1, \alpha_2, \dots, \alpha_m)^T$ entra na base (a coluna a sair da base segue os passos usuais do Método Simplex).

A seguir, apresentamos o algoritmo do Método Simplex com Geração Colunas aplicado ao problema de corte de estoque unidimensional.

Algoritmo

FASE I:

- Determine uma base inicial $\mathbf{B}$ que será composta por padrões de cortes homogêneos (padrões que produzem um único tipo de item), da seguinte forma:

$$\mathbf{B} = \begin{pmatrix} \min\left\{\left\lfloor \frac{L}{\ell_1} \right\rfloor, d_1\right\} & 0 & \cdots & 0 \\ 0 & \min\left\{\left\lfloor \frac{L}{\ell_2} \right\rfloor, d_2\right\} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \min\left\{\left\lfloor \frac{L}{\ell_m} \right\rfloor, d_m\right\} \end{pmatrix}$$

- Faça: $\text{PARE} = \text{Falso}$ e $\text{IT} = 0$.

{A variável lógica PARE terá valor Falso até que a condição de otimalidade seja verificada e IT indica o número da iteração corrente.}

FASE II:

Enquanto $\text{PARE} = \text{Falso}$, faça: $\{\text{Início da iteração Simplex}\}$

Passo 1 $\{\text{Determine a solução básica corrente}\}$

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}.$$

Passo 2 {*Verificação da Otimalidade*}

Passo 2.1 {*Determine o vetor multiplicador simplex*}

$$\pi^T = \mathbf{c}_B^T \mathbf{B}^{-1}$$

Passo 2.2 {*Cálculo do custo relativo*}

Resolva o Problema da Mochila: {*gera uma coluna para entrar na base*}

$$\text{maximizar } g(\mathbf{a}) \quad \pi_1 \alpha_1 + \pi_2 \alpha_2 + \dots + \pi_m \alpha_m$$

$$s.a. \quad \begin{cases} \ell_1 \alpha_1 + \ell_2 \alpha_2 + \dots + \ell_m \alpha_m \leq L \\ 0 \leq \alpha_i \leq d_i \text{ e inteiro} \end{cases}$$

Passo 2.3 {*Teste de Otimalidade*}

Se $1 - \sum_{i=1}^m \pi_i a_i \geq 0$ então
PARE = Verdadeiro

{*solução atual é ótima e foi obtida em IT iterações.*}

Senão

Ir para o passo 3

Passo 3 {*Determine as coordenadas básicas da direção simplex $\mathbf{y}$* }

$$\mathbf{B}\mathbf{y} = -\mathbf{a}.$$

($\mathbf{a}$ é a solução do problema da mochila no passo 2.2)

Passo 3.1 {*Determinação do Tamanho do passo*}

Encontre k tal que:

$$-\frac{a_k}{y_k} = \min \left\{ -\frac{a_i}{y_i} \mid y_i < 0. \right\}$$

Passo 3.2 {*Atualização*}

Atualize a base, colocando a coluna $\mathbf{a}$ na posição da k -ésima coluna da matriz $\mathbf{B}$.

Faça: $IT = IT + 1$.

A.2. Problema da Mochila

Suponha que um *objeto* deva ser cortado ao longo de seu comprimento L em *itens* de comprimentos $\ell_i, i = 1, \dots, m$. Cada item tem um valor associado que denominamos de *valor de utilidade*. Itens cujos comprimentos não foram especificados são considerados *perdas* e têm valor de utilidade nulos. Sendo assim, surge o problema de otimização combinatória que é *Maximizar o Valor de Utilidade Total*.

Se considerarmos que a quantidade máxima de itens demandados não seja especificada, teremos o *Problema da Mochila Irrestrito*.

Variável de decisão:

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

Modelagem matemática:

$$\begin{aligned} & \text{maximizar } g(a) = v_1\alpha_1 + v_2\alpha_2 + \dots + v_m\alpha_m \\ & \text{sujeito a: } \begin{cases} \ell_1\alpha_1 + \ell_2\alpha_2 + \dots + \ell_m\alpha_m \leq L \\ \alpha_i \geq 0, i = 1, \dots, m \text{ e inteiros} \end{cases} \end{aligned} \quad (a.8)$$

em que:

L = comprimento da peça em estoque;

m = número de tipos de itens;

ℓ_i = comprimento do item i , $i = 1, \dots, m$;

v_i = valor de utilidade do item i , $i = 1, \dots, m$.

Quando ocorre limitação na quantidade de itens de um determinado tipo, o problema passa a ser chamado de *Problema da Mochila Restrito ou Limitado*. Neste caso, além dos dados necessários para o modelo descrito anteriormente, devemos considerar ainda:

d_i = quantidade máxima de itens do tipo i que pode ser selecionada, $i = 1, \dots, m$.

Variável de decisão:

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

Modelagem matemática:

$$\begin{aligned} & \text{maximizar } g(a) = v_1\alpha_1 + v_2\alpha_2 + \dots + v_m\alpha_m \\ & \text{sujeito a: } \begin{cases} \ell_1\alpha_1 + \ell_2\alpha_2 + \dots + \ell_m\alpha_m \leq L \\ 0 \leq \alpha_i \leq d_i, i = 1, \dots, m \text{ e inteiros} \end{cases} \end{aligned} \quad (a.9)$$

A.3. Métodos de Resolução para o Problema da Mochila

Programação Dinâmica

Em Gilmore e Gomory [9], um método de programação dinâmica para o problema de corte foi proposto. Este método apresenta uma fórmula recursiva para a resolução do Problema da Mochila utilizando uma estratégia de percorrer todas as possíveis decisões, ou seja, a solução de um nó representando uma mochila de tamanho X , $X = 1, \dots, L$, cujo valor da função objetivo é denotado por $G_i(X)$. Assim que os nós predecessores de X : $X - \ell_1, X - \ell_2, \dots, X - \ell_m$ estiverem resolvidos, isto é, se são conhecidos os valores de $G(X - \ell_1), G(X - \ell_2), \dots, G(X - \ell_m)$, é possível determinar $G(X)$.

Método da Enumeração Implícita (*Branch-and-Bound*)

O método da Enumeração Implícita foi proposto em 1963 por Gilmore e Gomory [10]. Este método é implementado utilizando primeiramente uma busca em profundidade numa árvore de decisões e basicamente consiste em enumerar implicitamente todas as soluções do Problema da Mochila. Para isto, utiliza-se *limitantes* para o problema que permitem que soluções “piores” que a atual sejam descartadas, sem perder a otimalidade. Este método é importante, pois gerar explicitamente todas as possíveis soluções

do problema da mochila, (para exemplos de tamanhos moderados) é um trabalho quase impossível de ser resolvido computacionalmente.

O algoritmo a seguir, descreve o método de enumeração implícita para a resolução do Problema da Mochila (a.9).

Algoritmo

Passo 1: *{Defina o problema segundo as variáveis mais valiosas}*

Faça: $\theta_i = \frac{v_i}{\ell_i}$, $i = 1, \dots, m$, e reordene as variáveis, tal que:

$$\theta_1 \geq \theta_2 \geq \dots \geq \theta_m$$

Passo 2: *{Determine a solução inicial, utilizando busca em profundidade primeiro}*

Determine a solução $\mathbf{a} = (\alpha_1, \alpha_2, \dots, \alpha_m)$, tal que:

$$\begin{aligned} \alpha_1 &= \left\{ \left\lfloor \frac{L}{\ell_1} \right\rfloor, d_1 \right\} & \Rightarrow & \text{sobra : } L - \alpha_1 \ell_1 \\ \alpha_2 &= \left\{ \left\lfloor \frac{L - \alpha_1 \ell_1}{\ell_2} \right\rfloor, d_2 \right\} & \Rightarrow & \text{sobra : } L - \alpha_1 \ell_1 - \alpha_2 \ell_2 \\ & \vdots & & \\ \alpha_k &= \left\{ \left\lfloor \frac{L - \sum_{j=1}^{k-1} \ell_j \alpha_j}{\ell_k} \right\rfloor, d_k \right\} & k &= 2, \dots, m \end{aligned}$$

Passo 3: *{Avalie a solução atual e armazene a mais valiosa}*

Determine:

$$g(\mathbf{a}) = \sum_{i=1}^m v_i \alpha_i$$

(Considere inicialmente $\underline{G} = 0$). Se $\underline{G} < g(\mathbf{a})$ então faça:

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

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

Passo 4: *{ Teste a otimalidade e calcule o limitante superior}*

Determine k o maior índice tal que: $\alpha_k \neq 0$.

Se $\mathbf{a} = \mathbf{0}$ então PARE, a melhor solução guardada em $\mathbf{a}^*$ é uma solução ótima

Senão, calcule: (se $k = m$, considere $v_{m+1} = 0$ e $\ell_{m+1} = 1$)

$$\bar{G} = v_1x_1 + \dots + v_k(x_k - 1) + v_{k+1}x_{k+1} + \dots + v_jx_j.$$

Passo 5: *{ Backtracking }*

Passo 5.1: *{ Retorno longo}*

Se $\bar{G}(\mathbf{a}) \leq \underline{G}$, então faça: $\alpha_k = 0$ e volte ao passo 4.

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

Se $\bar{G}(\mathbf{x}) > \underline{G}$, então faça: $\alpha_k \leftarrow \alpha_k - 1$ e defina a nova solução $\mathbf{a}$:

$$\alpha_j = \left\{ \left\lfloor \frac{L - \sum_{i=1}^{j-1} \ell_i \alpha_i}{\ell_j} \right\rfloor, d_j \right\} \quad j = k+1, \dots, m$$

volte ao passo 3.

Para resolver o Problema da Mochila (a.8), torna-se necessário considerar algumas simples modificações no algoritmo acima.

Apêndice B. Alguns Exemplos Resolvidos

Neste apêndice, apresentamos alguns exemplos de problema de corte de estoque. Esses exemplos foram escolhidos aleatoriamente nas classes apresentadas no Seção 7.3 e resolvidos utilizando cada um dos métodos apresentados anteriormente.

Para todos exemplos a seguir, temos que a porcentagem de perda aceitável para os objetos padronizados é $0.005L_k, k = 1, 2$ e a porcentagem de perda aceitável para os objetos retalhos é $0.05L_k, k = 3, \dots, K$, além disso, o comprimento aceitável para a sobra é obtido pela média dos comprimentos dos itens demandados.

B.1. Exemplo 1

Exemplo da Classe 1, com número de tipos de barras: $K = 5$ e número de itens: $m = 10$.

Id Objeto	Comprimento (cm)	Estoque
1	1000	100
2	1100	100
3	50	6
4	265	5
5	495	9

Tabela b.1: *Dados do Exemplo 1 - Objetos.*

Item	Tamanho (cm)	Demanda
1	82	191
2	72	14
3	61	148
4	50	374
5	51	462
6	40	276
7	28	53
8	17	48
9	19	116
10	7	46

Tabela b.2: *Dados do Exemplo 1 - Itens.*

Para este exemplo, o limite inferior para a sobra é 42 cm.

Solução da heurística FFD

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
3	6	0	0	0	1	0	0	0	0	0	0	0	0
1	15	12	0	0	0	0	0	0	0	0	2	2	0
1	1	11	1	0	0	0	0	0	0	1	1	0	0
2	1	0	13	2	0	0	1	0	0	0	0	2	0
2	8	0	0	18	0	0	0	0	0	0	0	2	0
1	1	0	0	2	0	17	0	0	0	0	1	4	0
1	23	0	0	0	0	19	0	1	0	0	0	3	0
2	1	0	0	0	13	8	1	0	0	0	0	2	0
4	5	0	0	0	5	0	0	0	0	0	2	1	0
5	9	0	0	0	9	0	1	0	0	0	0	5	0
2	11	0	0	0	22	0	0	0	0	0	0	0	0
2	1	0	0	0	7	0	18	1	0	0	0	2	0
1	9	0	0	0	0	0	25	0	0	0	0	0	0
1	1	0	0	0	0	0	22	4	0	0	1	1	0
1	1	0	0	0	0	0	0	25	0	15	2	1	0
1	1	0	0	0	0	0	0	0	0	52	1	5	0
2	1	0	0	0	0	0	0	0	11	48	0	1	0
1	1	0	0	0	0	0	0	0	37	0	0	0	371

Tabela b.3: *Solução da heurística FFD.*

Solução da heurística FFD_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	14	13	0	0	0	0	0	0	2	0	0	0	0
3	6	0	0	0	1	0	0	0	0	0	0	0	0
2	1	9	5	0	0	0	0	0	0	0	0	2	0
1	1	0	9	5	0	0	1	0	0	0	1	0	0
1	8	0	0	16	0	0	0	0	1	0	1	0	0
1	1	0	0	15	0	1	0	0	2	0	0	0	0
1	10	0	0	0	0	19	0	0	1	0	2	0	0
1	8	0	0	0	0	18	1	1	0	0	2	0	0
1	1	0	0	0	0	18	0	2	0	1	1	0	0
1	6	0	0	0	0	18	2	0	0	0	0	2	0
5	1	0	0	0	8	1	1	0	0	0	0	4	0
5	8	0	0	0	9	0	1	0	0	0	0	5	0
2	13	0	0	0	22	0	0	0	0	0	0	0	0
2	1	0	0	0	2	0	25	0	0	0	0	0	0
1	8	0	0	0	0	0	25	0	0	0	0	0	0
1	1	0	0	0	0	0	21	5	0	1	0	1	0
1	1	0	0	0	0	0	0	35	0	1	0	1	0
1	1	0	0	0	0	0	0	3	0	48	0	4	0
1	1	0	0	0	0	0	0	0	0	50	0	0	50
1	1	0	0	0	0	0	0	0	0	15	0	0	715

Tabela b.4: Solução da heurística FFD_R .

Solução da heurística Gulosa

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
1	14	11	1	0	0	0	0	0	0	1	1	0	0
1	3	11	0	0	1	0	0	0	2	0	2	0	0
1	1	4	0	10	0	0	0	1	2	0	0	0	0
1	8	0	0	16	0	0	0	0	1	0	1	0	0
1	1	0	0	10	7	0	1	0	0	0	0	0	0
1	18	0	0	0	20	0	0	0	0	0	0	0	0
1	1	0	0	0	4	15	0	1	0	0	1	0	0
1	8	0	0	0	0	19	0	0	1	0	2	0	0
1	1	0	0	0	0	18	0	2	0	1	1	0	0
1	12	0	0	0	0	17	2	0	2	1	0	0	0
1	4	0	0	0	0	17	0	0	0	7	0	0	0
1	1	0	0	0	0	5	13	6	0	3	0	0	0
1	9	0	0	0	0	0	25	0	0	0	0	0	0
1	1	0	0	0	0	0	13	9	0	12	0	0	0
1	1	0	0	0	0	0	0	33	0	4	0	0	0
5	1	0	0	0	0	0	0	0	0	26	0	1	0
3	1	0	0	0	0	0	0	1	0	1	0	3	0
3	5	0	0	0	0	0	0	0	0	2	0	12	0
1	1	0	0	0	0	0	0	0	0	5	0	0	905

Tabela b.5: *Solução da heurística Gulosa*

Solução da heurística Gulosa_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
1	14	11	1	0	0	0	0	0	0	1	1	0	0
1	3	11	0	0	1	0	0	0	2	0	2	0	0
1	1	4	0	10	0	0	0	1	2	0	0	0	0
1	8	0	0	16	0	0	0	0	1	0	1	0	0
1	1	0	0	10	7	0	1	0	0	0	0	0	0
1	18	0	0	0	20	0	0	0	0	0	0	0	0
1	1	0	0	0	4	15	0	1	0	0	1	0	0
1	8	0	0	0	0	19	0	0	1	0	2	0	0
1	1	0	0	0	0	18	0	2	0	1	1	0	0
1	12	0	0	0	0	17	2	0	2	1	0	0	0
1	4	0	0	0	0	17	0	0	0	7	0	0	0
1	1	0	0	0	0	5	13	6	0	3	0	0	0
1	9	0	0	0	0	0	25	0	0	0	0	0	0
1	1	0	0	0	0	0	13	9	0	12	0	0	0
1	1	0	0	0	0	0	0	33	0	4	0	0	0
5	1	0	0	0	0	0	0	0	0	26	0	1	0
4	1	0	0	0	0	0	0	1	0	12	0	9	0
4	1	0	0	0	0	0	0	0	0	4	0	0	189

Tabela b.6: *Solução da heurística Gulosa_R*

Solução da heurística Residual FFD

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	13	12	0	0	0	2	0	0	0	0	2	0	0
2	8	0	0	17	0	0	0	2	0	0	1	0	0
2	2	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	21	0	0	0	0	20	2	0	0	0	0	0	0
2	9	0	0	0	0	0	24	0	0	7	1	0	0
2	1	0	0	0	0	16	0	2	0	12	0	0	0
2	1	4	0	12	0	0	1	0	0	0	0	0	0
2	1	0	14	0	0	0	0	2	1	1	0	0	0
2	1	0	0	0	0	0	0	0	39	23	0	0	0
1	1	5	0	0	0	0	14	1	0	0	0	2	0
3	3	0	0	0	0	0	1	0	0	0	1	3	0
5	1	0	0	0	0	0	0	17	0	1	0	0	0
3	3	0	0	0	0	0	0	1	0	1	0	3	0
1	1	0	0	0	0	0	0	12	4	13	0	0	349

Tabela b.7: *Solução da heurística Residual FFD*

Solução da heurística Residual FFD_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	13	12	0	0	0	2	0	0	0	0	2	0	0
2	8	0	0	17	0	0	0	2	0	0	1	0	0
2	2	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	21	0	0	0	0	20	2	0	0	0	0	0	0
2	9	0	0	0	0	0	24	0	0	7	1	0	0
2	1	0	0	0	0	16	0	2	0	12	0	0	0
2	1	4	0	12	0	0	1	0	0	0	0	0	0
2	1	0	14	0	0	0	0	2	1	1	0	0	0
2	1	0	0	0	0	0	0	0	39	23	0	0	0
1	1	4	0	0	0	0	15	0	2	2	0	0	0
2	1	1	0	0	0	0	2	33	0	0	2	0	0
4	1	0	0	0	0	0	0	0	1	13	0	1	0
3	1	0	0	0	0	0	0	0	0	0	1	0	43
4	1	0	0	0	0	0	0	0	1	2	0	0	210

Tabela b.8: *Solução da heurística Residual FFD_R*

Solução da heurística Residual Gulosa

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	13	12	0	0	0	2	0	0	0	0	2	0	0
2	8	0	0	17	0	0	0	2	0	0	1	0	0
2	2	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	21	0	0	0	0	20	2	0	0	0	0	0	0
2	9	0	0	0	0	0	24	0	0	7	1	0	0
2	1	0	0	0	0	16	0	2	0	12	0	0	0
2	1	4	0	12	0	0	1	0	0	0	0	0	0
2	1	0	14	0	0	0	0	2	1	1	0	0	0
2	1	0	0	0	0	0	0	0	39	23	0	0	0
1	1	5	0	0	0	0	13	2	0	0	2	0	0
1	1	0	0	0	0	0	4	30	0	0	0	0	0
4	1	0	0	0	0	0	0	0	4	10	1	0	0
3	1	0	0	0	0	0	0	1	0	1	0	3	0
3	3	0	0	0	0	0	0	0	0	2	0	12	0

Tabela b.9: *Solução da heurística Residual Gulosa*

Solução da heurística Residual Gulosa_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	13	12	0	0	0	2	0	0	0	0	2	0	0
2	8	0	0	17	0	0	0	2	0	0	1	0	0
2	2	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	21	0	0	0	0	20	2	0	0	0	0	0	0
2	9	0	0	0	0	0	24	0	0	7	1	0	0
2	1	0	0	0	0	16	0	2	0	12	0	0	0
2	1	4	0	12	0	0	1	0	0	0	0	0	0
2	1	0	14	0	0	0	0	2	1	1	0	0	0
2	1	0	0	0	0	0	0	0	39	23	0	0	0
1	1	5	0	0	0	0	13	2	0	0	2	0	0
1	1	0	0	0	0	0	4	30	0	0	0	0	0
4	1	0	0	0	0	0	0	0	4	10	1	0	0
4	1	0	0	0	0	0	0	1	0	7	0	0	104

Tabela b.10: *Solução da heurística Residual Gulosa_R*

Solução da heurística RAG - versão 1

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	11	12	0	0	0	2	0	0	0	0	2	0	0
2	8	0	0	17	0	0	0	2	0	0	1	0	0
2	3	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	22	0	0	0	0	20	2	0	0	0	0	0	0
2	1	0	14	0	0	0	0	1	0	3	1	0	0
2	9	0	0	0	0	0	24	0	0	7	1	0	0
2	1	13	0	0	0	0	0	0	2	0	0	0	0
2	1	0	0	0	0	0	16	11	0	8	0	0	0
2	1	0	0	12	0	0	0	0	20	0	4	0	0
2	1	3	0	0	0	0	0	22	14	0	0	0	0
2	1	3	0	0	0	0	0	2	0	42	0	0	0
4	1	1	0	0	0	0	0	1	6	0	2	39	0

Tabela b.11: *Solução da heurística RAG - versão 1*

Solução da heurística RAG_R - versão 1

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	11	12	0	0	0	2	0	0	0	0	2	0	0
2	8	0	0	17	0	0	0	2	0	0	1	0	0
2	3	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	22	0	0	0	0	20	2	0	0	0	0	0	0
2	1	0	14	0	0	0	0	1	0	3	1	0	0
2	9	0	0	0	0	0	24	0	0	7	1	0	0
2	1	13	0	0	0	0	0	0	2	0	0	0	0
2	1	0	0	0	0	0	16	11	0	8	0	0	0
2	1	0	0	12	0	0	0	0	20	0	4	0	0
2	1	3	0	0	0	0	0	22	14	0	0	0	0
2	1	3	0	0	0	0	0	2	0	42	0	0	0
3	1	0	0	0	0	0	0	0	2	0	2	2	0
4	1	1	0	0	0	0	0	1	4	0	0	0	87

Tabela b.12: Solução da heurística RAG_R - versão 1

Solução da heurística RAG - versão 2

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	11	12	0	0	0	2	0	0	0	0	2	0	0
2	8	0	0	17	0	0	0	2	0	0	1	0	0
2	3	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	22	0	0	0	0	20	2	0	0	0	0	0	0
2	1	0	14	0	0	0	0	1	0	3	1	0	0
2	9	0	0	0	0	0	24	0	0	7	1	0	0
2	1	13	0	0	0	0	0	0	2	0	0	0	0
2	1	0	0	0	0	0	16	11	0	8	0	0	0
2	1	0	0	12	0	0	0	0	20	0	4	0	0
2	1	3	0	0	0	0	0	22	14	0	0	0	0
2	1	3	0	0	0	0	0	2	0	42	0	0	0
3	1	0	0	0	0	0	0	0	2	0	2	2	0
3	1	0	0	0	0	0	0	0	2	0	0	16	0
3	1	0	0	0	0	0	0	1	1	0	0	5	0
3	1	0	0	0	0	0	0	0	1	0	0	33	0
4	1	1	0	0	0	0	0	0	0	0	0	0	183

Tabela b.13: Solução da heurística RAG - versão 2

Solução da heurística RAG_R - versão 2

A solução encontrada pela heurística RAG_R - versão 2 foi a mesma encontrada pela heurística RAG_R - versão 1 apresentada anteriormente.

Solução da heurística RAG - versão 3

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	11	12	0	0	0	2	0	0	0	0	2	0	0
2	7	0	0	17	0	0	0	2	0	0	1	0	0
2	3	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	22	0	0	0	0	20	2	0	0	0	0	0	0
2	1	0	14	0	0	0	0	1	0	3	1	0	0
2	1	0	0	0	0	0	0	38	1	1	0	0	0
2	1	0	0	0	0	0	0	0	1	57	0	0	0
2	7	0	0	0	0	0	24	0	0	7	1	0	0
2	1	10	0	0	0	0	0	0	12	4	0	0	0
2	1	0	0	17	0	0	0	0	0	0	9	0	0
1	1	0	0	0	0	0	25	0	0	0	0	0	0
2	1	0	0	0	0	0	24	0	6	2	0	0	0
2	1	10	0	3	0	0	2	0	1	0	0	0	0
5	1	0	0	3	0	0	1	0	16	0	0	0	0
5	1	0	0	1	0	0	10	0	2	0	0	0	0
4	1	0	0	2	0	0	2	0	3	0	0	12	0
4	1	0	0	3	0	0	0	0	0	0	0	0	82

Tabela b.14: *Solução da heurística RAG - versão 3*

Solução da heurística RAG_R - versão 3

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	11	12	0	0	0	2	0	0	0	0	2	0	0
2	7	0	0	17	0	0	0	2	0	0	1	0	0
2	3	13	0	0	0	0	0	0	2	0	0	0	0
2	17	0	0	0	22	0	0	0	0	0	0	0	0
2	22	0	0	0	0	20	2	0	0	0	0	0	0
2	1	0	14	0	0	0	0	1	0	3	1	0	0
2	1	0	0	0	0	0	0	38	1	1	0	0	0
2	1	0	0	0	0	0	0	0	1	57	0	0	0
2	7	0	0	0	0	0	24	0	0	7	1	0	0
2	1	10	0	0	0	0	0	0	12	4	0	0	0
2	1	0	0	17	0	0	0	0	0	0	9	0	0
1	1	0	0	0	0	0	25	0	0	0	0	0	0
2	1	0	0	0	0	0	24	0	6	2	0	0	0
2	1	10	0	3	0	0	2	0	1	0	0	0	0
5	1	0	0	3	0	0	1	0	16	0	0	0	0
5	1	0	0	1	0	0	10	0	2	0	0	0	0
4	1	0	0	3	0	0	2	0	0	0	0	2	0
4	1	0	0	2	0	0	0	0	3	0	0	0	92

Tabela b.15: *Solução da heurística RAG_R - versão 3*

Solução do Exemplo 1

	Objetos Retalhos	Objetos Padrões	Perda Total	Sobra Total	Padrões com Perda	Padrões com Sobra	Padrões sem Perda	Tempo (s)
FFD	20	76	183	371	13	1	4	0,001
FFD_R	15	77	64	765	7	2	11	0,031
Gulosa	7	84	64	905	3	1	15	0,031
Gulosa_R	3	83	10	189	2	1	15	0,047
Residual FFD	7	76	20	349	3	1	11	0,842
Residual FFD_R	3	76	1	253	1	2	12	0,813
Residual Gulosa	5	76	39	0	2	0	13	0,812
Residual Gulosa_R	2	76	0	104	0	1	13	0,813
RAG - versão 1	1	76	39	0	1	0	12	0,422
RAG_R -versão 1	2	76	2	87	1	1	12	0,422
RAG - versão 2	5	76	56	183	4	1	12	0,547
RAG_R -versão 2	2	76	2	87	1	1	12	0,422
RAG - versão 3	4	75	12	82	1	1	16	0,547
RAG_R -versão 3	4	75	2	92	1	1	16	0,563

Tabela b.16: *Solução do Exemplo 1*

B.2. Exemplo 2

Exemplo da Classe 13, com número de tipos de barras: $K = 9$ e número de itens: $m = 10$.

Id Objeto	Comprimento (cm)	Estoque
1	1000	100
2	1100	100
3	115	6
4	353	5
5	102	4
6	340	9
7	104	4
8	327	9
9	91	4

Tabela b.17: *Dados do Exemplo 2 - Objetos*

Item	Tamanho (cm)	Demanda
1	41	583
2	35	288
3	36	385
4	30	725
5	23	585
6	17	537
7	18	789
8	11	29
9	5	359
10	44	248

Tabela b.18: *Dados do Exemplo 2 - Itens*

Para este exemplo, o limite inferior para a sobra é 26 cm.

Solução da heurística FFD

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
7	4	0	0	0	0	0	0	0	1	1	2	0	0
5	4	0	0	0	0	0	0	0	1	0	2	3	0
3	6	0	0	0	0	1	0	0	0	0	2	4	0
1	10	0	0	0	1	0	0	0	0	0	22	2	0
9	4	2	0	0	0	0	0	0	0	1	0	4	0
1	21	24	0	0	0	0	0	0	1	1	0	0	0
6	8	8	0	0	0	0	0	0	0	2	0	2	0
1	1	7	0	19	0	1	0	0	0	1	0	1	0
6	1	0	0	9	0	0	0	0	0	3	0	1	0
4	5	0	0	9	0	1	0	0	0	1	0	1	0
1	11	0	0	27	0	1	0	0	0	1	0	0	0
1	1	0	13	15	0	0	0	0	0	1	0	0	0
8	9	0	9	0	0	0	0	0	0	2	0	2	0
2	6	0	31	0	0	0	0	0	0	3	0	0	0
2	1	0	8	0	27	0	0	0	0	2	0	0	0
1	20	0	0	0	33	0	0	0	0	2	0	0	0
2	1	0	0	0	28	11	0	0	0	1	0	2	0
1	12	0	0	0	0	43	0	0	0	2	0	1	0
1	1	0	0	0	0	35	0	10	0	3	0	0	0
1	14	0	0	0	0	0	0	55	0	2	0	0	0
1	1	0	0	0	0	0	49	9	0	1	0	0	0
1	8	0	0	0	0	0	58	0	0	2	0	4	0
2	1	0	0	0	0	0	24	0	0	138	0	2	0
2	1	0	0	0	0	0	0	0	0	4	0	0	1080

Tabela b.19: *Solução da heurística FFD*

Solução da heurística FFD_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
7	4	0	0	0	0	0	0	0	1	1	2	0	0
8	9	1	0	0	0	0	1	0	0	1	6	0	0
5	4	1	0	0	0	0	1	0	0	0	1	0	0
3	6	0	0	0	0	1	0	0	0	0	2	4	0
1	7	0	0	0	1	0	0	0	0	0	22	2	0
1	1	7	0	1	0	0	1	0	0	0	15	0	0
9	4	1	1	0	0	0	0	0	0	3	0	0	0
1	23	24	0	0	0	0	0	0	1	1	0	0	0
2	1	7	0	22	0	0	0	0	1	2	0	0	0
4	1	0	0	8	0	0	0	0	1	2	1	0	0
2	11	0	0	30	0	0	0	0	0	4	0	0	0
1	1	0	3	24	1	0	0	0	0	0	0	1	0
6	9	0	9	0	0	0	0	0	0	5	0	0	0
1	7	0	28	0	0	0	0	0	0	4	0	0	0
1	1	0	4	0	28	0	0	0	0	4	0	0	0
2	19	0	0	0	36	0	0	0	0	4	0	0	0
4	1	0	0	0	4	8	2	0	0	3	0	0	0
2	12	0	0	0	0	46	1	0	0	5	0	0	0
1	1	0	0	0	0	19	0	31	0	1	0	0	0
2	11	0	0	0	0	0	0	60	0	4	0	0	0
1	1	0	0	0	0	0	0	55	0	2	0	0	0
1	1	0	0	0	0	0	13	43	0	0	0	5	0
1	1	0	0	0	1	0	57	0	0	0	0	1	0
1	7	0	0	0	0	0	57	0	0	0	0	0	31
1	1	0	0	0	0	0	40	0	0	0	0	0	320

Tabela b.20: Solução da heurística FFD_R .

Solução da heurística Gulosa

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
1	24	24	0	0	0	0	0	0	1	1	0	0	0
1	1	7	19	0	1	0	0	1	0	0	0	0	0
1	9	0	28	0	0	0	0	0	0	4	0	0	0
1	1	0	17	10	1	0	0	0	0	3	0	0	0
1	13	0	0	27	0	1	0	0	0	1	0	0	0
1	1	0	0	24	4	0	0	0	1	1	0	0	0
1	21	0	0	0	33	0	0	0	0	2	0	0	0
1	1	0	0	0	26	8	0	2	0	0	0	0	0
1	4	0	0	0	0	43	0	0	1	0	0	0	0
1	9	0	0	0	0	42	2	0	0	0	0	0	0
1	1	0	0	0	0	14	39	0	0	3	0	0	0
1	8	0	0	0	0	0	56	1	0	6	0	0	0
1	1	0	0	0	0	0	32	22	0	12	0	0	0
1	6	0	0	0	0	0	0	55	0	2	0	0	0
2	7	0	0	0	0	0	0	60	0	4	0	0	0
2	1	0	0	0	0	0	0	6	0	128	8	0	0
2	9	0	0	0	0	0	0	0	0	0	25	0	0
4	1	0	0	0	0	0	0	0	0	9	7	0	0
4	1	0	0	0	0	0	0	0	0	0	8	1	0

Tabela b.21: *Solução da heurística Gulosa.*

Solução da heurística Gulosa_R

A solução encontrada pela heurística Gulosa_R foi a mesma encontrada pela heurística Gulosa apresentada anteriormente.

Solução da heurística Residual FFD

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	8	0	31	0	0	0	0	0	0	3	0	0	0
2	11	0	0	28	0	4	0	0	0	0	0	0	0
2	24	0	0	3	30	4	0	0	0	0	0	0	0
2	9	0	2	0	0	44	0	1	0	0	0	0	0
2	7	0	0	0	0	0	63	1	1	0	0	0	0
2	12	0	0	0	0	0	0	60	0	4	0	0	0
2	1	0	0	0	0	0	0	0	0	220	0	0	0
2	9	0	0	0	0	0	0	0	0	0	25	0	0
1	24	23	0	0	0	1	2	0	0	0	0	0	0
2	1	26	0	0	0	0	0	1	1	1	0	0	0
7	4	0	0	0	0	0	0	0	1	1	2	0	0
5	4	0	0	0	0	0	0	0	1	0	2	3	0
3	3	0	0	0	0	1	0	0	0	0	2	4	0
9	1	1	0	0	0	0	0	0	0	1	1	1	0
9	2	2	0	0	0	0	0	0	0	1	0	4	0
8	1	0	4	5	0	0	0	0	0	1	0	2	0
3	3	0	3	0	0	0	0	0	0	2	0	0	0
2	1	0	9	0	5	22	0	7	0	0	0	3	0
6	2	0	0	0	0	0	0	18	1	1	0	0	0
1	1	0	0	0	0	0	48	9	2	0	0	0	0
8	1	0	0	0	0	0	0	0	9	45	0	3	0
9	1	0	0	0	0	0	0	0	0	5	0	0	66

Tabela b.22: *Solução da heurística Residual FFD.*

Solução da heurística Residual FFD_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	8	0	31	0	0	0	0	0	0	3	0	0	0
2	11	0	0	28	0	4	0	0	0	0	0	0	0
2	24	0	0	3	30	4	0	0	0	0	0	0	0
2	9	0	2	0	0	44	0	1	0	0	0	0	0
2	7	0	0	0	0	0	63	1	1	0	0	0	0
2	12	0	0	0	0	0	0	60	0	4	0	0	0
2	1	0	0	0	0	0	0	0	0	220	0	0	0
2	9	0	0	0	0	0	0	0	0	0	25	0	0
1	24	23	0	0	0	1	2	0	0	0	0	0	0
2	1	26	0	0	0	0	0	1	1	1	0	0	0
2	1	2	1	0	0	0	0	0	0	3	22	0	0
9	3	1	1	0	0	0	0	0	0	3	0	0	0
7	2	0	0	2	0	0	1	0	0	3	0	0	0
7	1	0	1	1	0	1	0	0	0	2	0	0	0
7	1	0	1	0	0	0	0	0	0	5	1	0	0
9	1	0	2	0	0	0	0	0	1	2	0	0	0
6	1	0	9	0	0	0	0	0	0	5	0	0	0
3	1	0	3	0	0	0	0	0	0	2	0	0	0
3	1	0	2	0	1	0	0	0	0	3	0	0	0
5	2	0	0	0	2	0	1	0	0	5	0	0	0
5	2	0	0	0	0	4	0	0	0	2	0	0	0
3	3	0	0	0	0	5	0	0	0	0	0	0	0
1	1	0	0	0	0	0	2	52	0	6	0	0	0
3	1	0	0	0	0	0	5	0	0	6	0	0	0
1	1	0	0	0	0	1	37	0	20	3	0	0	113

Tabela b.23: Solução da heurística Residual FFD_R .

Solução da heurística Residual Gulosa

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	8	0	31	0	0	0	0	0	0	3	0	0	0
2	11	0	0	28	0	4	0	0	0	0	0	0	0
2	24	0	0	3	30	4	0	0	0	0	0	0	0
2	9	0	2	0	0	44	0	1	0	0	0	0	0
2	7	0	0	0	0	0	63	1	1	0	0	0	0
2	12	0	0	0	0	0	0	60	0	4	0	0	0
2	1	0	0	0	0	0	0	0	0	220	0	0	0
2	9	0	0	0	0	0	0	0	0	0	25	0	0
1	24	23	0	0	0	1	2	0	0	0	0	0	0
2	1	26	0	0	0	0	0	1	1	1	0	0	0
1	1	5	22	0	0	0	0	0	0	5	0	0	0
1	1	0	0	5	5	25	5	0	0	2	0	0	0
1	1	0	0	0	0	0	43	13	0	7	0	0	0
1	1	0	0	0	0	0	0	39	19	9	1	0	0
1	1	0	0	0	0	0	0	0	2	2	22	0	0
3	1	0	0	0	0	0	0	0	0	23	0	0	0
9	1	0	0	0	0	0	0	0	0	18	0	1	0

Tabela b.24: *Solução da heurística Residual Gulosa.*

Solução da heurística Residual Gulosa_R

A solução encontrada pela heurística Residual Gulosa_R foi a mesma encontrada pela heurística Residual Gulosa apresentada anteriormente.

Solução da heurística RAG - versão 1

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	8	0	31	0	0	0	0	0	0	3	0	0	0
2	11	0	0	28	0	4	0	0	0	0	0	0	0
2	24	0	0	3	30	4	0	0	0	0	0	0	0
2	9	0	2	0	0	44	0	1	0	0	0	0	0
2	13	0	0	0	0	0	0	60	0	4	0	0	0
2	1	0	0	0	0	0	0	0	0	220	0	0	0
2	9	0	0	0	0	0	0	0	0	0	25	0	0
1	25	23	0	0	0	1	2	0	0	0	0	0	0
2	1	0	22	0	5	0	0	0	15	3	0	0	0
2	1	0	0	0	0	0	50	0	0	50	0	0	0
2	7	0	0	0	0	2	62	0	0	0	0	0	0
2	1	1	0	0	0	0	1	0	0	6	23	0	0
4	1	4	0	4	0	0	2	0	1	0	0	0	0
4	1	3	0	0	0	10	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0	9	1	0	0	0
5	1	0	0	1	0	0	0	0	4	3	0	7	0

Tabela b.25: Solução da heurística RAG - versão 1.

Solução da heurística RAG_R - versão 1

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	8	0	31	0	0	0	0	0	0	3	0	0	0
2	11	0	0	28	0	4	0	0	0	0	0	0	0
2	24	0	0	3	30	4	0	0	0	0	0	0	0
2	9	0	2	0	0	44	0	1	0	0	0	0	0
2	13	0	0	0	0	0	0	60	0	4	0	0	0
2	1	0	0	0	0	0	0	0	0	220	0	0	0
2	9	0	0	0	0	0	0	0	0	0	25	0	0
1	25	23	0	0	0	1	2	0	0	0	0	0	0
2	1	0	22	0	5	0	0	0	15	3	0	0	0
2	1	0	0	0	0	0	50	0	0	50	0	0	0
2	7	0	0	0	0	2	62	0	0	0	0	0	0
2	1	1	0	0	0	0	1	0	0	6	23	0	0
4	1	4	0	4	0	0	2	0	1	0	0	0	0
4	1	3	0	0	0	10	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0	9	1	0	0	0
9	1	0	0	1	0	0	0	0	4	2	0	1	0
9	1	0	0	0	0	0	0	0	0	1	0	0	86

Tabela b.26: Solução da heurística RAG_R - versão 1.

Solução da heurística RAG - versão 2

A solução encontrada pela heurística RAG - versão 2 foi a mesma encontrada pela heurística RAG - versão 1 apresentada anteriormente.

Solução da heurística RAG_R - versão 2

A solução encontrada pela heurística RAG_R - versão 2 foi a mesma encontrada pela heurística RAG_R - versão 1 apresentada anteriormente.

Solução da heurística RAG - versão 3

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	9	0	31	0	0	0	0	0	0	3	0	0	0
2	11	0	0	28	0	4	0	0	0	0	0	0	0
2	24	0	0	3	30	4	0	0	0	0	0	0	0
2	2	0	2	0	0	44	0	1	0	0	0	0	0
2	7	0	0	0	0	0	63	1	1	0	0	0	0
2	13	0	0	0	0	0	0	60	0	4	0	0	0
2	1	0	0	0	0	0	0	0	0	220	0	0	0
2	9	0	0	0	0	0	0	0	0	0	25	0	0
1	25	23	0	0	0	1	2	0	0	0	0	0	0
2	1	0	0	0	0	7	41	0	22	0	0	0	0
2	4	0	0	0	0	45	0	0	0	13	0	0	0
2	1	0	1	0	1	1	0	0	0	0	23	0	0
2	1	0	0	0	2	45	0	0	0	1	0	0	0
2	1	8	3	0	0	29	0	0	0	0	0	0	0
2	1	0	0	2	0	41	5	0	0	0	0	0	0
8	1	0	0	0	0	14	0	0	0	1	0	0	0
4	1	0	1	0	2	11	0	0	0	1	0	0	0
3	1	0	0	2	0	1	0	0	0	4	0	0	0
3	1	0	0	1	0	3	0	0	0	1	0	5	0

Tabela b.27: *Solução da heurística RAG - versão 3.*

Solução da heurística RAG_R - versão 3

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	9	0	31	0	0	0	0	0	0	3	0	0	0
2	11	0	0	28	0	4	0	0	0	0	0	0	0
2	24	0	0	3	30	4	0	0	0	0	0	0	0
2	2	0	2	0	0	44	0	1	0	0	0	0	0
2	7	0	0	0	0	0	63	1	1	0	0	0	0
2	13	0	0	0	0	0	0	60	0	4	0	0	0
2	1	0	0	0	0	0	0	0	0	220	0	0	0
2	9	0	0	0	0	0	0	0	0	0	25	0	0
1	25	23	0	0	0	1	2	0	0	0	0	0	0
2	1	0	0	0	0	7	41	0	22	0	0	0	0
2	4	0	0	0	0	45	0	0	0	13	0	0	0
2	1	0	1	0	1	1	0	0	0	0	23	0	0
2	1	0	0	0	2	45	0	0	0	1	0	0	0
2	1	8	3	0	0	29	0	0	0	0	0	0	0
2	1	0	0	2	0	41	5	0	0	0	0	0	0
8	1	0	0	0	0	14	0	0	0	1	0	0	0
4	1	0	1	0	2	11	0	0	0	1	0	0	0
3	1	0	0	2	0	1	0	0	0	4	0	0	0
9	1	0	0	1	0	2	0	0	0	1	0	4	0
9	1	0	0	0	0	1	0	0	0	0	0	0	68

Tabela b.28: Solução da heurística RAG_R - versão 3.

Solução do Exemplo 2

	Objetos Retalhos	Objetos Padrões	Perda Total	Sobra Total	Padrões com Perda	Padrões com Sobra	Padrões sem Perda	Tempo (s)
FFD	41	110	161	1008	13	1	10	0,001
FFD_R	38	106	45	537	5	2	18	0,016
Gulosa	2	117	1	0	1	0	18	0,031
Gulosa_R	2	117	1	0	1	0	18	0,047
Residual FFD	22	108	41	66	7	1	14	0,672
Residual FFD_R	19	109	0	113	0	1	24	0,687
Residual Gulosa	2	111	1	0	1	0	16	0,703
Residual Gulosa_R	2	111	1	0	1	0	16	0,703
RAG - versão 1	4	110	7	0	1	0	15	10,610
RAG_R -versão 1	5	110	1	86	1	1	15	11,70
RAG - versão 2	4	110	7	0	1	0	15	10,610
RAG_R -versão 2	5	110	1	86	1	1	15	11,70
RAG - versão 3	4	110	5	0	1	0	18	2,234
RAG_R -versão 3	5	110	4	68	1	1	18	2,438

Tabela b.29: Solução do Exemplo 2

B.3. Exemplo 3

Exemplo da Classe 14, com número de tipos de barras: $K = 9$ e número de itens: $m = 10$.

Id Objeto	Comprimento (cm)	Estoque
1	1000	100
2	1100	100
3	216	2
4	420	7
5	204	2
6	409	7
7	193	6
8	398	5
9	195	4

Tabela b.30: *Dados do Exemplo 3 - Objetos*

Item	Tamanho (cm)	Demanda
1	112	202
2	113	161
3	106	147
4	100	94
5	94	121
6	95	83
7	88	203
8	82	165
9	76	17
10	70	203

Tabela b.31: *Dados do Exemplo 3 - Itens*

Para este exemplo, o limite inferior para a sobra é 93 cm.

Solução da heurística FFD

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
9	4	0	1	0	0	0	0	0	1	0	0	0	0
6	7	0	3	0	0	0	0	0	0	0	1	0	0
3	2	0	1	0	1	0	0	0	0	0	0	3	6
1	16	0	8	0	0	0	1	0	0	0	0	1	0
2	1	3	6	0	0	0	0	0	1	0	0	4	0
2	22	9	0	0	0	0	0	1	0	0	0	4	0
2	1	1	0	9	0	0	0	0	0	0	0	34	0
2	13	0	0	10	0	0	0	0	0	0	0	40	0
1	1	0	0	8	1	0	0	0	0	0	0	52	0
1	9	0	0	0	10	0	0	0	0	0	0	0	0
1	1	0	0	0	1	0	9	0	0	0	0	45	0
1	5	0	0	0	0	0	10	0	0	0	0	50	0
1	1	0	0	0	0	2	8	0	0	0	0	52	0
1	11	0	0	0	0	10	0	0	0	0	0	60	0
2	1	0	0	0	0	9	0	2	0	1	0	2	0
1	16	0	0	0	0	0	0	11	0	0	0	32	0
1	1	0	0	0	0	0	0	3	8	1	0	4	0
1	12	0	0	0	0	0	0	0	12	0	0	16	0
1	1	0	0	0	0	0	0	0	8	4	0	40	0
1	1	0	0	0	0	0	0	0	0	11	2	24	0
1	13	0	0	0	0	0	0	0	0	0	14	20	0
1	1	0	0	0	0	0	0	0	0	0	12	0	160

Tabela b.32: *Solução da heurística FFD.*

Solução da heurística FFD_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
4	7	1	2	0	0	0	0	0	1	0	0	0	0
3	2	0	1	0	1	0	0	0	0	0	0	3	0
1	18	0	8	0	0	0	1	0	0	0	0	1	0
2	1	8	1	0	0	0	0	1	0	0	0	3	0
1	23	8	0	0	1	0	0	0	0	0	0	4	0
1	1	2	0	6	0	0	0	0	0	0	2	0	0
1	17	0	0	8	0	0	0	0	1	0	1	0	0
1	1	0	0	5	4	0	0	0	0	0	1	0	0
1	1	1	0	0	8	0	0	1	0	0	0	0	0
1	5	0	0	0	10	0	0	0	0	0	0	0	0
1	1	0	0	0	6	0	2	0	0	0	3	0	0
1	7	0	0	0	0	0	9	0	0	0	2	5	0
2	12	0	0	0	0	10	0	1	0	0	1	2	0
1	23	0	0	0	0	0	0	8	1	0	3	4	0
2	1	0	0	0	0	1	0	5	6	0	1	4	0
2	11	0	0	0	0	0	0	0	10	0	4	0	0
1	1	0	0	0	0	0	0	0	2	11	0	0	0
1	2	0	0	0	0	0	0	0	0	3	11	2	0
2	1	0	0	0	0	0	0	0	0	0	14	0	120
2	1	0	0	0	1	0	0	0	0	0	4	0	720

Tabela b.33: Solução da heurística FFD_R .

Solução da heurística Gulosa

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
1	17	7	0	0	0	0	0	0	0	1	2	0	0
1	13	6	0	1	0	0	0	0	1	0	2	0	0
1	1	5	0	2	0	0	0	1	0	0	2	0	0
1	26	0	6	0	1	0	0	0	1	0	2	0	0
1	1	0	5	1	0	1	1	0	0	0	2	0	0
1	16	0	0	8	0	0	0	0	1	0	1	0	0
1	1	0	0	3	6	0	0	0	1	0	0	0	0
1	6	0	0	0	10	0	0	0	0	0	0	0	0
1	1	0	0	0	2	5	2	0	0	0	2	0	0
1	16	0	0	0	0	7	2	0	1	0	1	0	0
1	1	0	0	0	0	3	2	6	0	0	0	0	0
1	1	0	0	0	0	0	8	1	1	0	1	0	0
2	4	0	0	0	0	0	8	2	2	0	0	0	0
2	1	0	0	0	0	0	4	5	0	0	4	0	0
2	1	0	0	0	0	0	2	0	0	0	13	0	0
2	5	0	0	0	0	0	0	6	1	0	7	0	0
2	9	0	0	0	0	0	0	5	8	0	0	4	0
4	1	0	0	0	0	0	0	1	4	0	0	4	0
7	6	0	0	0	0	0	0	2	0	0	0	17	0
5	2	0	0	0	0	0	0	2	0	0	0	28	0
3	2	0	0	0	0	0	0	2	0	0	0	40	0
2	7	0	0	0	0	0	0	12	0	0	0	44	0
2	1	0	0	0	0	0	0	2	3	0	0	0	678

Tabela b.34: *Solução da heurística Gulosa.*

Solução da heurística Gulosa_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
1	17	7	0	0	0	0	0	0	0	1	2	0	0
1	13	6	0	1	0	0	0	0	1	0	2	0	0
1	1	5	0	2	0	0	0	1	0	0	2	0	0
1	26	0	6	0	1	0	0	0	1	0	2	0	0
1	1	0	5	1	0	1	1	0	0	0	2	0	0
1	16	0	0	8	0	0	0	0	1	0	1	0	0
1	1	0	0	3	6	0	0	0	1	0	0	0	0
1	6	0	0	0	10	0	0	0	0	0	0	0	0
1	1	0	0	0	2	5	2	0	0	0	2	0	0
1	16	0	0	0	0	7	2	0	1	0	1	0	0
1	1	0	0	0	0	3	2	6	0	0	0	0	0
1	1	0	0	0	0	0	8	1	1	0	1	0	0
2	4	0	0	0	0	0	8	2	2	0	0	0	0
2	1	0	0	0	0	0	4	5	0	0	4	0	0
2	1	0	0	0	0	0	2	0	0	0	13	0	0
2	5	0	0	0	0	0	0	6	1	0	7	0	0
2	9	0	0	0	0	0	0	5	8	0	0	4	0
4	1	0	0	0	0	0	0	1	4	0	0	4	0
9	4	0	0	0	0	0	0	1	0	0	0	0	107
8	5	0	0	0	0	0	0	3	0	0	0	0	134
7	6	0	0	0	0	0	0	1	0	0	0	0	105
6	7	0	0	0	0	0	0	3	0	0	0	0	145
5	2	0	0	0	0	0	0	1	0	0	0	0	116
4	6	0	0	0	0	0	0	3	0	0	0	0	156
3	2	0	0	0	0	0	0	1	0	0	0	0	128
2	3	0	0	0	0	0	0	11	0	0	0	0	132
2	1	0	0	0	0	0	0	5	3	0	0	0	414

Tabela b.35: *Solução da heurística Gulosa_R.*

Solução da heurística Residual FFD

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	14	0	6	2	0	0	0	0	0	0	3	0	0
2	13	0	0	9	0	0	0	0	0	1	1	0	0
2	8	0	0	0	11	0	0	0	0	0	0	0	0
2	6	0	1	0	0	8	1	0	0	0	2	0	0
2	9	0	0	0	0	0	8	2	2	0	0	0	0
2	20	0	2	0	0	0	0	9	1	0	0	0	0
2	11	0	0	0	0	0	0	0	10	0	4	0	0
2	3	1	6	0	0	0	0	0	2	1	1	0	0
2	1	0	0	0	0	0	2	0	0	0	13	0	0
1	33	6	0	0	0	2	0	0	0	0	2	0	0
2	1	0	6	0	0	0	0	4	0	0	1	0	0
9	4	0	1	0	0	0	0	0	1	0	0	0	0
6	1	0	3	0	0	0	0	0	0	0	1	0	0
1	1	1	0	2	6	0	0	0	0	1	0	0	0
7	1	0	0	0	0	0	2	0	0	0	0	3	0
1	1	0	0	0	0	7	1	1	1	0	1	7	0
8	1	0	0	0	0	0	0	0	4	0	1	0	0
8	1	0	0	0	0	0	0	0	2	0	3	24	0
3	1	0	0	0	0	0	0	0	0	0	3	6	0

Tabela b.36: *Solução da heurística Residual FFD.*

Solução da heurística Residual FFD_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	14	0	6	2	0	0	0	0	0	0	3	0	0
2	13	0	0	9	0	0	0	0	0	1	1	0	0
2	8	0	0	0	11	0	0	0	0	0	0	0	0
2	6	0	1	0	0	8	1	0	0	0	2	0	0
2	9	0	0	0	0	0	8	2	2	0	0	0	0
2	20	0	2	0	0	0	0	9	1	0	0	0	0
2	11	0	0	0	0	0	0	0	10	0	4	0	0
2	3	1	6	0	0	0	0	0	2	1	1	0	0
2	1	0	0	0	0	0	2	0	0	0	13	0	0
1	33	6	0	0	0	2	0	0	0	0	2	0	0
2	1	0	6	0	0	0	0	4	0	0	1	0	0
4	1	1	2	0	0	0	0	0	1	0	0	0	0
9	4	0	1	0	0	0	0	0	1	0	0	0	0
1	1	0	0	2	6	2	0	0	0	0	0	0	0
1	1	0	0	0	0	5	3	1	1	0	1	5	0
3	1	0	0	0	0	0	0	0	0	1	2	0	0
8	1	0	0	0	0	0	0	0	4	0	1	0	0
6	1	0	1	0	0	0	0	0	1	0	3	4	0
7	2	0	0	0	0	0	0	0	0	0	1	0	123

Tabela b.37: *Solução da heurística Residual FFD_R .*

Solução da heurística Residual Gulosa

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	14	0	6	2	0	0	0	0	0	0	3	0	0
2	13	0	0	9	0	0	0	0	0	1	1	0	0
2	8	0	0	0	11	0	0	0	0	0	0	0	0
2	6	0	1	0	0	8	1	0	0	0	2	0	0
2	9	0	0	0	0	0	8	2	2	0	0	0	0
2	20	0	2	0	0	0	0	9	1	0	0	0	0
2	11	0	0	0	0	0	0	0	10	0	4	0	0
2	3	1	6	0	0	0	0	0	2	1	1	0	0
2	1	0	0	0	0	0	2	0	0	0	13	0	0
1	33	6	0	0	0	2	0	0	0	0	2	0	0
2	1	0	6	0	0	0	0	4	0	0	1	0	0
1	1	1	6	0	0	0	0	0	0	0	3	0	0
1	1	0	1	2	3	0	1	0	0	0	4	0	0
1	1	0	0	0	3	3	2	1	0	0	2	0	0
1	1	0	0	0	0	1	0	0	11	0	0	4	0
7	1	0	0	0	0	2	0	0	0	0	0	5	0
7	1	0	0	0	0	1	0	0	0	1	0	23	0

Tabela b.38: *Solução da heurística Residual Gulosa.*

Solução da heurística Residual Gulosa_R

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	14	0	6	2	0	0	0	0	0	0	3	0	0
2	13	0	0	9	0	0	0	0	0	1	1	0	0
2	8	0	0	0	11	0	0	0	0	0	0	0	0
2	6	0	1	0	0	8	1	0	0	0	2	0	0
2	9	0	0	0	0	0	8	2	2	0	0	0	0
2	20	0	2	0	0	0	0	9	1	0	0	0	0
2	11	0	0	0	0	0	0	0	10	0	4	0	0
2	3	1	6	0	0	0	0	0	2	1	1	0	0
2	1	0	0	0	0	0	2	0	0	0	13	0	0
1	33	6	0	0	0	2	0	0	0	0	2	0	0
2	1	0	6	0	0	0	0	4	0	0	1	0	0
1	1	1	6	0	0	0	0	0	0	0	3	0	0
1	1	0	1	2	3	0	1	0	0	0	4	0	0
1	1	0	0	0	3	3	2	1	0	0	2	0	0
1	1	0	0	0	0	1	0	0	11	0	0	4	0
7	1	0	0	0	0	2	0	0	0	0	0	5	0
9	1	0	0	0	0	0	0	0	0	1	0	0	119
7	1	0	0	0	0	1	0	0	0	0	0	0	99

Tabela b.39: Solução da heurística Residual Gulosa_R.

Solução da heurística RAG - versão 1

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	15	0	6	2	0	0	0	0	0	0	3	0	0
2	13	0	0	9	0	0	0	0	0	1	1	0	0
2	8	0	0	0	11	0	0	0	0	0	0	0	0
2	6	0	1	0	0	8	1	0	0	0	2	0	0
2	7	0	0	0	0	0	8	2	2	0	0	0	0
2	21	0	2	0	0	0	0	9	1	0	0	0	0
2	12	0	0	0	0	0	0	0	10	0	4	0	0
2	3	1	6	0	0	0	0	0	2	1	1	0	0
2	1	0	0	0	0	0	2	0	0	0	13	0	0
1	33	6	0	0	0	2	0	0	0	0	2	0	0
2	1	1	1	0	4	0	5	0	0	0	0	0	0
2	1	0	2	0	0	1	6	0	0	0	3	0	0
2	1	0	0	0	0	1	8	0	3	0	0	0	0
7	1	0	0	0	0	2	0	0	0	0	0	5	0
7	1	0	1	0	0	0	0	0	0	1	0	4	0
9	1	0	1	0	0	0	0	0	1	0	0	0	0
9	2	0	0	0	1	1	0	0	0	0	0	1	0
7	1	0	0	0	0	1	0	0	0	0	0	0	99

Tabela b.40: Solução da heurística RAG - versão 1.

Solução da heurística RAG_R - versão 1

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	15	0	6	2	0	0	0	0	0	0	3	0	0
2	13	0	0	9	0	0	0	0	0	1	1	0	0
2	8	0	0	0	11	0	0	0	0	0	0	0	0
2	6	0	1	0	0	8	1	0	0	0	2	0	0
2	7	0	0	0	0	0	8	2	2	0	0	0	0
2	21	0	2	0	0	0	0	9	1	0	0	0	0
2	12	0	0	0	0	0	0	0	10	0	4	0	0
2	3	1	6	0	0	0	0	0	2	1	1	0	0
2	1	0	0	0	0	0	2	0	0	0	13	0	0
1	33	6	0	0	0	2	0	0	0	0	2	0	0
2	1	1	1	0	4	0	5	0	0	0	0	0	0
2	1	0	2	0	0	1	6	0	0	0	3	0	0
2	1	0	0	0	0	1	8	0	3	0	0	0	0
9	1	0	1	0	0	0	0	0	1	0	0	0	0
9	2	0	0	0	1	1	0	0	0	0	0	1	0
8	1	0	1	0	0	3	0	0	0	0	0	3	0
7	1	0	0	0	0	0	0	0	0	1	0	0	117

Tabela b.41: *Solução da heurística RAG_R - versão 1.*

Solução da heurística RAG - versão 2

A solução encontrada pela heurística RAG - versão 2 foi a mesma encontrada pela heurística RAG - versão 1 apresentada anteriormente.

Solução da heurística RAG_R - versão 2

A solução encontrada pela heurística RAG_R - versão 2 foi a mesma encontrada pela heurística RAG_R - versão 1 apresentada anteriormente.

Solução da heurística RAG - versão 3

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	15	0	6	2	0	0	0	0	0	0	3	0	0
2	13	0	0	9	0	0	0	0	0	1	1	0	0
2	8	0	0	0	11	0	0	0	0	0	0	0	0
2	7	0	1	0	0	8	1	0	0	0	2	0	0
2	8	0	0	0	0	0	8	2	2	0	0	0	0
2	20	0	2	0	0	0	0	9	1	0	0	0	0
2	12	0	0	0	0	0	0	0	10	0	4	0	0
2	4	1	6	0	0	0	0	0	2	1	1	0	0
2	2	0	0	0	0	0	2	0	0	0	13	0	0
1	26	6	0	0	0	2	0	0	0	0	2	0	0
2	3	8	0	0	2	0	0	0	0	0	0	4	0
2	1	5	0	0	0	2	0	4	0	0	0	0	0
1	1	4	0	0	0	5	0	0	1	0	0	0	0
1	1	3	0	0	0	1	6	0	0	0	0	0	0
2	1	5	0	0	0	5	0	0	0	0	1	0	0
7	1	0	0	0	0	0	2	0	0	0	0	3	0
7	1	0	0	0	0	0	0	2	0	0	0	17	0
5	1	1	0	0	0	0	0	1	0	0	0	4	0

Tabela b.42: Solução da heurística RAG - versão 3.

Solução da heurística RAG_R - versão 3

Id Objeto	x	Padrão de corte										Perdas (cm)	Sobras (cm)
2	15	0	6	2	0	0	0	0	0	0	3	0	0
2	13	0	0	9	0	0	0	0	0	1	1	0	0
2	8	0	0	0	11	0	0	0	0	0	0	0	0
2	7	0	1	0	0	8	1	0	0	0	2	0	0
2	8	0	0	0	0	0	8	2	2	0	0	0	0
2	20	0	2	0	0	0	0	9	1	0	0	0	0
2	12	0	0	0	0	0	0	0	10	0	4	0	0
2	4	1	6	0	0	0	0	0	2	1	1	0	0
2	2	0	0	0	0	0	2	0	0	0	13	0	0
1	26	6	0	0	0	2	0	0	0	0	2	0	0
2	1	5	0	0	0	2	0	4	0	0	0	0	0
1	1	4	0	0	0	5	0	0	1	0	0	0	0
1	1	3	0	0	0	1	6	0	0	0	0	0	0
2	1	5	0	0	0	5	0	0	0	0	1	0	0
2	2	9	0	0	0	0	0	1	0	0	0	4	0
4	2	2	0	0	1	0	1	0	0	0	0	1	0
3	2	1	0	0	1	0	0	0	0	0	0	4	0
1	1	1	0	0	2	0	0	1	0	0	0	0	600

Tabela b.43: Solução da heurística RAG_R - versão 3.

Solução do Exemplo 3

	Objetos Retalhos	Objetos Padrões	Perda Total	Sobra Total	Padrões com Perda	Padrões com Sobra	Padrões sem Perda	Tempo (s)
FFD	13	127	2761	160	18	1	3	0,016
FFD_R	9	127	278	840	9	2	9	0,062
Gulosa	11	128	586	678	6	1	16	0,172
Gulosa_R	33	124	40	4977	2	9	16	0,172
Residual FFD	9	121	40	0	4	0	15	1,265
Residual FFD_R	10	121	9	246	2	1	16	1,266
Residual Gulosa	2	123	32	0	3	0	14	1,282
Residual Gulosa_R	3	123	9	218	2	2	14	1,283
RAG - versão 1	6	122	11	99	3	1	14	0,797
RAG_R -versão 1	5	122	5	117	2	1	14	0,828
RAG - versão 2	6	122	11	99	3	1	14	0,797
RAG_R -versão 2	5	122	5	117	2	1	14	0,828
RAG - versão 3	3	122	36	0	2	0	14	0,625
RAG_R -versão 3	4	122	18	600	14	1	14	0,719

Tabela b.44: *Solução do Exemplo 3*