versão impressa ISSN 0101-7438 / versão online ISSN 1678-5142

HEURÍSTICAS PARA O PROBLEMA DE CORTE DE ESTOQUE UNIDIMENSIONAL INTEIRO

Kelly Cristina Poldi *
Marcos Nereu Arenales
Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo (USP)
São Carlos – SP
kelly@icmc.usp.br
arenales@icmc.usp.br

* *Corresponding author* / autor para quem as correspondências devem ser encaminhadas

Recebido em 12/2004; aceito em 04/2006 após 1 revisão
Received December 2004; accepted April 2006 after one revision

Resumo

O problema de corte de estoque unidimensional consiste em cortar objetos disponíveis para a produção de *itens* de modo a atender uma demanda especificada, em que apenas uma dimensão é relevante para o corte (barras, bobinas, etc.). O problema pode ser formulado como um problema de programação linear inteira de grande porte, cuja solução ótima, via de regra, não pode ser obtida na prática, quando tipicamente dezenas de itens devem ser produzidas. Neste artigo tratamos o problema de determinar soluções inteiras para o problema de corte de estoque unidimensional, dando atenção especial a problemas com baixa demanda. Revisamos métodos heurísticos bem conhecidos e algumas variações. Esses métodos são comparados em relação à qualidade de suas soluções, número de padrões de corte e tempo computacional. Nossa análise está baseada na resolução de exemplares gerados aleatoriamente.

Palavras-chave: problema de corte de estoque; programação inteira; geração de colunas.

Abstract

One-dimensional cutting stock problems consist of cutting standard objects available in stock into smaller pieces called *items* in order to meet a known demand. Only one dimension of the stock objects is considered in the cutting process (rolls, bars, etc.) This problem might be formulated as a large scale integer linear programming problem, which an optimal solution cannot be obtained in practice, especially when dozens of items have to be produced. This paper addresses the problem of finding integer solutions to the one-dimensional cutting stock problem, with special attention to problems with low demand. We study some heuristic approaches proposed in literature and some straightforward variants. These methods are compared with respect to solution quality, number of cutting patterns and computational time. Our evaluation is based on solving randomly generated instances.

Keywords: cutting stock problem; integer programming; column generation.

1. Introdução

Problemas de corte de estoque consistem na otimização do processo de corte de peças maiores, disponíveis em estoque, para a produção de peças menores, em quantidades encomendadas, os quais podem ser modelados por um problema de otimização linear inteira de grande porte, que consiste em definir os possíveis padrões de corte (i.e., como as peças maiores devem ser cortadas para a produção de peças menores) e as variáveis de decisão representam o número de vezes que as peças maiores são cortadas, segundo um padrão. Estes problemas são aparentemente simples, com grande aplicabilidade prática, porém são problemas NP-difíceis (Garey & Johnson, 1979; Dyckhoff, 1990). Dentre suas várias aplicações práticas podemos citar o corte de bobinas de aço, bobinas de papel, chapas de madeira, de metal, de vidro etc. Problemas de carregamento de contêineres de navios, caminhões e vagões de trens são alguns exemplos práticos de problemas de empacotamento, 'similares' aos problemas de corte em sua formulação e resolução.

Tais problemas vêm sendo investigados nas últimas quatro décadas, desde os trabalhos pioneiros de Gilmore & Gomory (1961, 1963), que propuseram uma técnica de geração de colunas para obtenção de uma solução ótima contínua (i.e., modelo com a condição de integralidade relaxada). Na maioria das situações práticas, não faz sentido utilizar um padrão de corte por um número fracionário de vezes, isto é, ou a peça é cortada segundo um padrão de corte ou não. O principal objetivo deste artigo consiste em determinar soluções inteiras para o problema de corte de estoque unidimensional. Problemas de corte de estoque com baixa demanda ocorrem, freqüentemente, na prática em indústrias de pequeno porte, com carteira de pedidos bem variada. Tipicamente, tais indústrias têm baixa padronização de seus produtos e atendem pedidos personalizados. Estas empresas precisam de uma solução razoável que seja rapidamente obtida para se manterem no mercado competidor. Há um folclore na área dos problemas de corte e empacotamento, no qual problemas de corte com demanda alta são bem resolvidos pela técnica de geração de colunas de Gilmore & Gomory, porém se a demanda for baixa, então heurísticas construtivas devem ser utilizadas (Hinxam, 1980; Riehme *et al.*, 1996). Entendemos que a razão deste folclore está no fato de que um problema com demandas baixas têm as soluções fracionárias, muitas vezes menores que um e, a maioria das heurísticas trabalha com aproximação para o inteiro inferior, muitas vezes zero e, portanto, os padrões obtidos pelo relaxamento da condição de integralidade são inúteis. Embora algumas poucas aproximações alternativas para a solução inteira tenham sido propostas (Stadtler, 1990), a questão da demanda baixa não foi ainda devidamente levada em conta. Por exemplo, Wäscher & Gau (1996) fazem um estudo computacional amplo de diversas heurísticas para obtenção de soluções inteiras para o problema de corte, mas omitem o caso de demanda baixa.

Poucos artigos na literatura são orientados para demanda baixa, embora indústrias pequenas, que têm boa parte de seus produtos personalizados, necessitam de soluções boas e rápidas. Soluções gulosas são úteis, mas um operador treinado é capaz de produzir soluções tão boas ou melhores para vários problemas, o que leva a um descrédito no sistema computacional e seu abandono. Riehme *et al.* (1996) estudaram um problema bidimensional, com cortes guilhotinados em 2-estágios, com grande variabilidade na demanda (alguns itens podem ter demanda alta, enquanto outros têm demanda baixa). Entretanto, a severa limitação devido à demanda baixa é apenas parcialmente considerada na construção de padrões de corte. Neste artigo propomos heurísticas de aproximação para obtenção de soluções inteiras que superam a dificuldade das heurísticas anteriores em face de demandas baixas.

Alguns poucos algoritmos exatos para encontrar a solução ótima inteira de problemas de corte de estoque são conhecidos na literatura (Vanderbeck & Wolsey, 1996; Carvalho, 1999), mas devido a limitações computacionais de tempo e/ou espaço, podem ser aplicados apenas a problemas de pequeno porte e, por isto, não serão analisados neste artigo, já que estamos interessados em soluções boas e rápidas, mesmo que a otimalidade não seja garantida. A eficácia das heurísticas será analisada pela comparação dos resultados de uma com a outra e, a capacidade de gerar soluções ótimas, quando um bom limitante inferior é conhecido, além do tempo computacional requerido.

A organização deste texto é a seguinte: na seção 2 apresentamos o problema a ser resolvido: problema de corte de estoque unidimensional inteiro, com um objeto de comprimento padrão disponível em estoque em quantidade suficiente para atender toda a demanda de itens; sua modelagem matemática e o método de geração de colunas. Na seção 3 está descrito um limitante inferior utilizado para determinar se uma solução inteira encontrada é ótima. Na seção 4 estão descritos os métodos heurísticos utilizados: construtivos e residuais. Na seção 5 descrevemos os testes computacionais realizados bem como uma análise de seus resultados. Na seção 6, apresentamos conclusões e propostas de continuidade.

2. Definição e Modelagem do Problema

O *problema de corte de estoque unidimensional inteiro* pode ser definido como:

Considere que temos disponível em estoque um número suficientemente grande de objetos (barras, bobinas, etc.) de comprimento L e um conjunto de pedidos de peças menores, com demanda conhecida d_i, $i = 1, ..., m$, que chamamos de *itens*, de comprimentos ℓ_i, $i = 1, ..., m$ (os comprimentos dos itens são tais que: $\ell_i \leq L$). O problema consiste em produzir os itens demandados a partir do corte dos objetos em estoque, atendendo a demanda e de modo que o número de objetos em estoque necessários para satisfazer a demanda seja minimizado. Outros critérios, tais como, perda, custo, etc., podem ser definidos, sem perda de generalidade deste estudo. Apenas a seção 4 é específica ao critério de número mínimo de objetos.

Não é permitido o excesso de produção e os objetos em estoque devem ser cortados completamente, ou seja, não é permitido cortar uma parte de um objeto e devolver o restante para o estoque. Pedaços do corte que não sejam os itens demandados são considerados perda.

Definição 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:

$$\mathbf{a} = (\alpha_1, \alpha_2, \ldots, \alpha_m)^T$$

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

Observe que um vetor $\mathbf{a} = (\alpha_1, \alpha_2, \ldots, \alpha_m)^T$ corresponde a um padrão de corte se e somente se satisfizer:

$$\ell_1 \alpha_1 + \ell_2 \alpha_2 + \ldots + \ell_m \alpha_m \leq L \tag{1.1}$$

$$0 \leq \alpha_i \leq d_i,\ i = 1, \ldots, m \text{ e inteiros,} \tag{1.2}$$

em que d_i é a demanda do item i, que não deve ser excedida.

Outras restrições de processo podem ocorrer em problemas práticos, como por exemplo, (1) o número de facas das máquinas que executam o corte é limitado; (2) os cortes devem ser executados em duas etapas, como no corte de bobinas de aço, em que a espessura da bobina deve ser reduzida por um processo de laminação (Marques & Arenales, 2006; Hoto *et al*., 2003), etc.

Definição 2: Um padrão de corte que produza apenas um tipo de item é chamado *padrão de corte homogêneo*.

Em outras palavras, um padrão de corte é homogêneo se o vetor associado tem apenas uma coordenada não-nula: $(0, \ldots, \alpha_i, \ldots, 0)^T$, $\alpha_i \neq 0$. Note que sempre teremos m padrões homogêneos, cujos vetores associados definem uma matriz diagonal.

Após definirmos os padrões de corte, o próximo passo será determinar o número de vezes que cada padrão será utilizado para resolver o problema, ou seja, a modelagem matemática de um problema de corte de estoque é feita em duas etapas:

1. Definir todos os possíveis padrões de corte (supondo n o número total de padrões obtidos);
2. Definir quantas vezes cada padrão de corte será utilizado para atender a demanda, o que deverá ser um número inteiro e não-negativo.

Sejam

$$\mathbf{a}_1 = \begin{pmatrix} \alpha_{11} \\ \alpha_{21} \\ \vdots \\ \alpha_{m1} \end{pmatrix}, \quad \mathbf{a}_2 = \begin{pmatrix} \alpha_{12} \\ \alpha_{22} \\ \vdots \\ \alpha_{m2} \end{pmatrix}, \quad \ldots, \quad \mathbf{a}_n = \begin{pmatrix} \alpha_{1n} \\ \alpha_{2n} \\ \vdots \\ \alpha_{mn} \end{pmatrix},$$

os vetores associados aos possíveis padrões de corte, em que α_{ij} é o número de vezes que o item de comprimento ℓ_i, $i = 1, \ldots, m$, aparece no padrão de corte j. Os vetores acima são todas as possíveis soluções do sistema (1.1)-(1.2).

Seja x_j = *número de objetos cortados conforme o padrão de corte j*. O problema de corte de estoque, em que o total de objetos em estoque cortado deva ser minimizado, é dado por (Gilmore & Gomory, 1961):

$$\text{minimizar} \quad f(\boldsymbol{x}) = x_1 + x_2 + \ldots + x_n \tag{2.1}$$

$$\text{sujeito a:} \quad \mathbf{a}_1 x_1 + \mathbf{a}_2 x_2 + \ldots + \mathbf{a}_n x_n = \boldsymbol{d} \tag{2.2}$$

$$x_1, \ldots, x_n \geq 0 \text{ e inteiros,} \tag{2.3}$$

em que $\mathbf{a}_i \in \mathcal{Z}^m$ é definido acima e $\boldsymbol{d} \in \mathcal{Z}^m$ é o vetor com as demandas dos itens.

Em notação matricial o problema (3) é escrito como:

$$\text{minimizar} \quad f(\mathbf{x}) = \sum_{j=1}^{n} x_j \tag{3.1}$$

$$\text{sujeito a:} \quad \boldsymbol{A}\,\boldsymbol{x} = \boldsymbol{d} \tag{3.2}$$

$$\boldsymbol{x} \geq \boldsymbol{0} \text{ e inteiro,} \tag{3.3}$$

em que cada coluna da matriz $\boldsymbol{A} \in \mathcal{Z}^{m \times n}$ é um vetor associado a um padrão de corte, n é o número de padrões de corte.

A condição de integralidade sobre as variáveis x_j torna o problema (3) difícil de ser resolvido computacionalmente em tempo aceitável na prática. Em problemas práticos, m (que é a quantidade de tipos de itens) é da ordem de dezenas, enquanto que n (o qual depende de m, L e ℓ_i, $i = 1, ..., m$) pode ser da ordem de centenas de milhares.

Uma abordagem prática para resolver o problema (3) consiste em relaxar a condição de integralidade (3.3) e resolver o problema relaxado (problema de otimização linear) pelo *método simplex revisado*, desenvolvido por Dantzig em 1947, utilizando um processo de *Geração de Colunas*, proposto por Gilmore & Gomory (1961, 1963), de modo que não é preciso gerar todas as colunas antecipadamente. O procedimento começa com um conjunto de m padrões de corte homogêneos (colunas) que constituem uma base da matriz $\boldsymbol{A}$. A cada iteração simplex, um destes padrões básicos é substituído por um padrão de corte que melhora a solução básica atual, a menos de degeneração. Tal padrão de corte (coluna) é determinado pelo menor custo relativo, isto é: mínimo$_j$ $\{1 - \boldsymbol{\pi}^T \mathbf{a}_j\}$, em que $\boldsymbol{\pi}$ é o vetor multiplicador simplex e $\mathbf{a}_j$ é uma coluna de $\boldsymbol{A}$ e, portanto, satisfaz as restrições (1.1)-(1.2). O sub-problema de geração de colunas, que consiste na determinação do menor custo relativo é então formulado como um problema da mochila (pequenas alterações ocorreriam se diferentes funções objetivos fossem usadas ao invés de (3.1) (Gilmore & Gomory, 1963; Arenales, 2003). A solução ótima obtida para o problema (3) relaxado é, em geral, fracionária. O simples arredondamento para o inteiro superior do valor das variáveis acarreta na produção de itens em quantidades muito superiores às quantidades demandas. Assim, heurísticas têm sido desenvolvidas para determinar um bom arredondamento da solução, com tempo computacional aceitável. Serão descritos quatro métodos heurísticos para este arredondamento, sendo três conhecidos na literatura e um novo, com variantes.

3. Critério para Solução Inteira Ótima

Para analisar a qualidade das soluções encontradas pelas heurísticas, bons limitantes para o valor da função objetivo são essenciais.

Um *limitante inferior* para a função objetivo do problema de corte de estoque (3) é dado por:

$$LI = \lceil f(\boldsymbol{x}) \rceil,$$

em que $\boldsymbol{x}$ é a solução linear do problema (3), isto é, um arredondamento do valor da função objetivo para o inteiro superior (Usamos a notação $\lceil \tau \rceil$ para indicar o menor inteiro maior ou igual a τ. De forma análoga, usamos a notação $\lfloor \tau \rfloor$ para indicar o maior inteiro menor ou igual a τ.).

Seja $\boldsymbol{y}$ uma solução qualquer factível (inteira) para o problema (3), se $f(\boldsymbol{y}) = \lceil f(\boldsymbol{x}) \rceil$ então podemos dizer que a solução $\boldsymbol{y}$ é ótima. Uma classe de problemas que satisfaça esta propriedade é dita ter IRUP (*Integer Round Up Property*). Equivalentemente, a igualdade $f(\boldsymbol{y}) = \lceil f(\boldsymbol{x}) \rceil$ implica que:

$$f(\boldsymbol{y}) - f(\boldsymbol{x}) < 1 ,$$

$f(\boldsymbol{y}) - f(\boldsymbol{x})$ é chamado de *gap de integralidade*. Marcotte (1986), porém, mostrou que a IRUP não vale para problemas de corte de estoque unidimensional. Scheithauer & Terno (1995) conjecturam que o problema de corte de estoque unidimensional (3) possui MIRUP (*Modified Integer Round Up Property*), dada por: se $\boldsymbol{y}$ é ótimo, então $f(\boldsymbol{y}) \leq \lceil f(\boldsymbol{x}) \rceil + 1$. Em 1996, Wäscher & Gau (1996) encontraram um *gap* de 1,06667. Pouco tempo depois, Schwerin & Wäscher (1997) apresentaram um exemplo com 200 itens cujo *gap* é 1,14435. O maior encontrado até o momento foi publicado por Rietz *et al*. (2000) e é de 7/6.

4. Métodos de Solução

Há duas maneiras bem distintas para determinarmos soluções inteiras e aproximadas para o problema de corte de estoque. A primeira delas é aplicar heurísticas de repetição exaustiva (Hinxman, 1980), ou seja, construímos um bom padrão de corte e o utilizamos o máximo possível de vezes, isto é o que chamamos de *heurísticas construtivas*. A outra maneira é utilizar uma abordagem residual, ou seja, resolvemos o problema relaxado por geração de colunas, aproximamos sua solução fracionária por uma solução inteira, restando um problema de menor demanda, chamado residual, o qual deve ser resolvido. Gilmore & Gomory (1961) foram os primeiros a proporem esta abordagem, porém não apresentaram estudos a este respeito, apenas sugeriram que a aproximação deveria ser para o inteiro inferior e o problema residual resolvido por algum método *ad hoc*. Esta tem sido a abordagem mais utilizada na prática. Neste trabalho, apresentamos alternativas para o arredondamento.

4.1 Heurísticas construtivas

Estas heurísticas, como já mencionamos, são de repetição exaustiva e são bem conhecidas na literatura: Hinxman (1980), Stadtler (1990), Wäscher & Gau (1996), Pinto (1999), Hoto *et al.* (2003), entre outros, e seguem o seguinte algoritmo:

Passo 1: Construa um bom padrão de corte;

Passo 2: Utilize-o o número máximo possível de vezes, sem que a demanda seja excedida;

Passo 3: Atualize a demanda.

Repita os passos 1-2-3 até que toda a demanda seja atendida.

Estudamos dois métodos clássicos da literatura para "construir um bom padrão de corte" (i.e., passo 1 da heurística de repetição exaustiva): o método FFD – *First Fit Decreasing* e um método guloso para a construção do padrão de corte. Estes métodos estão descritos a seguir.

4.1.1 Heurística construtiva FFD

A heurística *FFD* (*First-Fit-Decreasing*) consiste, de forma geral, em colocar o item maior num padrão de corte tantas vezes quanto for possível, ou seja, até que este item não caiba mais ou, até que sua demanda já tenha sido atendida. A seguir, colocar o segundo maior item e assim por diante. Quando o último item (menor comprimento) for examinado, um padrão de corte foi construído. Com este padrão construído, execute os passos 2 e 3 do algoritmo em 4.1. É comum na literatura uma implementação diferente para a heurística FFD, esgotando-se o maior item em tantos objetos quantos necessários, isto é, alocando o maior item no primeiro objeto tantas vezes quanto possível. Se ainda restarem unidades do maior item para serem alocadas, aloque-as num segundo objeto e assim por diante, até que o maior item seja esgotado. Proceda de forma análoga com o segundo maior item, alocando-o, se possível, no espaço restante do primeiro objeto, segundo objeto e, assim por diante, até que o segundo maior item seja esgotado. Analogamente para os demais itens. Utilizamos a primeira implementação, pois nos parece mais apropriada para tratar o problema de corte de estoque.

4.1.2 Heurística construtiva gulosa

A heurística *gulosa* consiste em resolver o problema da mochila (4.1)-(4.3) com uma função objetivo apropriada, o *valor de utilidade* $v_i = \ell_i$, o que equivale minimizar a perda no padrão.

$$\text{maximizar} \quad z(\mathbf{a}) = \ell_1 \alpha_1 + \ell_2 \alpha_2 + \ldots + \ell_m \alpha_m \tag{4.1}$$

$$\text{sujeito a:} \quad \ell_1 \alpha_1 + \ell_2 \alpha_2 + \ldots + \ell_m \alpha_m \leq L \tag{4.2}$$

$$0 \leq \alpha_i \leq d_i \,,\ i = 1, \ldots, m \ \text{ e inteiros.} \tag{4.3}$$

Então, a heurística gulosa consiste em, a cada iteração, resolver o problema (4) para gerar um padrão de corte. O problema da mochila (4) foi resolvido por um algoritmo de enumeração implícita. Com este padrão assim gerado, execute os passos 2 e 3 do algoritmo descrito na seção 4.1.

4.2 Heurísticas residuais

Os procedimentos apresentados nesta seção são baseados na idéia de aproximar a solução ótima do problema (3) relaxado por uma solução inteira. Supomos que pelo menos uma componente da solução ótima do problema relaxado seja não-inteira, pois, caso contrário, o problema de corte de estoque unidimensional inteiro já estaria resolvido.

Definição 3: Seja $\boldsymbol{x} \in \Re^n$ a solução do problema (3) com a condição de integralidade relaxada e seja $\boldsymbol{y}$ um vetor de inteiros "próximo", de alguma forma, à $\boldsymbol{x}$, isto é, uma aproximação inteira para $\boldsymbol{x}$ tal que:

$$\boldsymbol{A}\,\mathbf{y} \leq \boldsymbol{d}$$

$$\boldsymbol{y} \geq \boldsymbol{0}$$

O vetor $\boldsymbol{y}$ é chamado de solução inteira aproximada para $\boldsymbol{x}$.

Por exemplo, uma forma de obter-se uma solução inteira aproximada é fazer o arredondamento para o maior inteiro menor que o valor a ser arredondado: $\boldsymbol{y} = (\lfloor x_1 \rfloor, \lfloor x_2 \rfloor, \ldots \lfloor x_n \rfloor)$.

Este é o modo usual para se obter uma solução inteira aproximada para $\boldsymbol{x}$ que, no nosso entender, é a raiz das dificuldades das heurísticas residuais para demanda baixa e origem do folclore na área, de que em face de baixa demanda, heurísticas construtivas devem ser utilizadas. Salientamos que as heurísticas residuais já foram trabalhadas exaustivamente em Wäscher & Gau (1996) e se mostraram superiores para problemas com alta demanda. Neste artigo, mostramos que as heurísticas residuais são também bem superiores do que as construtivas, mesmo para problemas com baixa demanda, particularmente quando a solução inteira aproximada para $\boldsymbol{x}$ busca arredondamento para o inteiro superior. Na seção 4.2.4 apresentamos um novo procedimento para o cálculo de uma solução inteira aproximada.

Definição 4: Sejam $\boldsymbol{y} \in \mathcal{Z}^n$ uma solução aproximada para $\boldsymbol{x}$ e $\boldsymbol{r} = \boldsymbol{d} - \boldsymbol{A}\boldsymbol{y}$, a demanda residual. O problema residual é dado por:

$$\text{minimizar} \quad f(\boldsymbol{x}) = \sum_{j=1}^{n} x_j \tag{5.1}$$

$$\text{sujeito a:} \quad \boldsymbol{A}\,\boldsymbol{x} = \boldsymbol{r} \tag{5.2}$$

$$\boldsymbol{x} \geq \boldsymbol{0} \text{ e inteiro.} \tag{5.3}$$

4.2.1 Heurísticas residuais – procedimento básico

Nesta seção apresentamos um breve algoritmo para as heurísticas do tipo residuais e nas próximas seções esclarecemos alguns passos, que diferem de uma heurística para a outra. A estrutura geral é a seguinte:

Passo 1: { *Inicialização* }
Sejam $k = 1$, $\boldsymbol{r}^1 = \boldsymbol{d}$, os dados para o problema residual inicial.

Passo 2: { *Determinar uma solução ótima contínua* }
Resolver o problema (5) relaxado. (método simplex com geração de colunas)
Seja $\boldsymbol{x}^k$ a solução contínua obtida.
Se $\boldsymbol{x}^k \in \mathcal{Z}^n$, ou seja, é uma solução inteira, então PARE.

Passo 3: { *Determinar uma solução inteira aproximada* }
Determine uma solução inteira aproximada para $\boldsymbol{x}^k$ e denote-a por $\boldsymbol{y}^k$.
Se $\boldsymbol{y}^k$ for um vetor nulo, então vá para o passo final.

Passo 4: { *Atualizações* }
Determine a nova demanda residual: $\boldsymbol{r}^{k+1} = \boldsymbol{r}^k - \boldsymbol{A}\boldsymbol{y}^k$.
$k = k + 1$.
Repita o **Passo 2**.

Passo Final:
Resolva o *problema residual final*.

Este algoritmo será completamente definido quando especificarmos como determinar a solução inteira aproximada $\boldsymbol{y}^k$, no **Passo 3** e como resolver o problema residual final no **Passo Final**.

As três próximas heurísticas a serem definidas: *residual FFD*, *residual gulosa* e *residual bin-packing*, têm o mesmo **Passo 3**, no algoritmo geral descrito na seção anterior. A solução inteira aproximada $\boldsymbol{y}^k$ é determinada arredondando-se para o inteiro inferior a solução contínua $\boldsymbol{x}^k$. Esta solução aproximada pode levar ao problema residual final prematuramente, em caso de baixa demanda.

4.2.2 Heurística residual FFD

Passo 3: { *Determinar uma solução inteira aproximada:* $\boldsymbol{y}^k$ }

Arredondar as componentes do vetor $\boldsymbol{x}^k \in \Re^n$ para o maior inteiro menor ou igual a $\boldsymbol{x}^k$, isto é: $\boldsymbol{y}^k = \lfloor \boldsymbol{x}^k \rfloor$.

Passo Final: Resolver o problema residual final pela heurística construtiva FFD, descrita na seção 4.1.1.

4.2.3 Heurística residual gulosa

Passo 3: { *Determinar uma solução inteira aproximada:* $\boldsymbol{y}^k$ }

Arredondar as componentes do vetor $\boldsymbol{x}^k \in \Re^n$ para o maior inteiro menor ou igual a $\boldsymbol{x}^k$, isto é: $\boldsymbol{y}^k = \lfloor \boldsymbol{x}^k \rfloor$.

Passo Final: Resolver o problema residual final pela heurística construtiva gulosa, descrita na seção 4.1.2.

4.2.4 Heurística residual *bin-packing*

Passo 3: { *Determinar uma solução inteira aproximada:* $\boldsymbol{y}^k$ }

Arredondar as componentes do vetor $\boldsymbol{x}^k \in \Re^n$ para o maior inteiro menor ou igual a $\boldsymbol{x}^k$, isto é: $\boldsymbol{y}^k = \lfloor \boldsymbol{x}^k \rfloor$.

Passo Final: O problema de corte de estoque residual final (poucos itens restantes) é modelado como um problema *bin-packing* e resolvido de forma exata pelo pacote CPLEX 7.5.

O problema *bin-packing* é definido a seguir. Sejam L o comprimento dos q objetos disponíveis em estoque e ℓ_i, $i = 1, ..., p$ o comprimento dos p itens a serem cortados. Na formulação do problema *bin-packing*, a repetição de itens não é levada em conta, cada item tem seu próprio comprimento, podendo ocorrer de mais de um item ter o mesmo comprimento. Sejam as variáveis de decisão:

$$y_j = \begin{cases} 1, & \text{se o objeto j é usado,} \\ 0, & \text{caso contrário} \end{cases}$$

$$x_{ij} = \begin{cases} 1, & \text{se o item i é cortado do objeto j,} \\ 0, & \text{caso contrário} \end{cases}$$

O problema consiste em (Martello & Toth, 1990):

$$\text{minimizar} \quad g(y)=\sum_{j=1}^{q} y_j \tag{6.1}$$

$$\text{sujeito a:} \quad \sum_{i=1}^{p} \ell_i x_{ij} \;\le\; L\, y_j \qquad j=1,\ldots,q \tag{6.2}$$

$$\sum_{j=1}^{q} x_{ij} = 1 \qquad i=1,\ldots,p \tag{6.3}$$

$$y_j=\{0,1\},\;\; x_{ij}=\{0,1\} \qquad i=1,\ldots,p,\; j=1,\ldots,q. \tag{6.4}$$

Observe que o modelo (6.1)-(6.4) pode ser usado para modelar o problema original, porém com um alto número de itens (aqui considerados um a um, ignorando repetições; p. ex., na Tabela 1 o número total de itens a ser produzido é $d_1+d_2+d_3$=18, ao invés dos 3 tipos de itens do modelo (2.1)-(2.3), que considera a repetição). Para os problemas residuais, os quais envolvem poucos itens, o modelo (6.1)-(6.4) pode ser resolvido por um método de enumeração implícita. Entretanto, os experimentos computacionais na próxima seção atestam a dificuldade desta abordagem.

4.2.5 Heurísticas residuais novas

Apresentamos uma nova heurística, em três versões diferentes. Esta heurística difere das três heurísticas residuais anteriores na maneira de determinar a solução inteira aproximada $\boldsymbol{y}$, nas quais um arredondamento simples é sempre feito. O arredondamento para o inteiro superior é infactível para todas as freqüências (ou seja, todas as componentes do vetor $\boldsymbol{x}$), mas algumas delas poderiam ser arredondadas para o inteiro superior, enquanto outras seriam arredondadas para o inteiro inferior. As heurísticas novas são baseadas nesta idéia. Estas heurísticas consistem basicamente em, a cada iteração, resolver um problema de corte de estoque relaxado e ordenar o vetor solução de forma específica (analisamos três formas de ordenação). Para cada posição deste vetor, na ordem especificada, arredonda-se a freqüência para o número inteiro acima do fracionário obtido e testa-se a factibilidade desta solução (no sentido de que excessos de itens não sejam gerados). Caso não seja factível (isto é, houve excesso de itens), a freqüência é reduzida de uma unidade até que excessos sejam eliminados. Quando o último padrão de corte gerado for examinado, atualiza-se a demanda, resultando num problema residual, que será tratado da mesma forma. É importante notar que no processo de geração de colunas, ao menos um dos padrões de corte gerados pode ser utilizado pelo menos uma vez. Isto garante que a demanda residual fica cada vez menor a cada iteração e, ao final, a demanda residual é nula. Ou seja, o passo final é desnecessário.

Para definirmos esta heurística, devemos especificar o **Passo 3** do algoritmo residual geral, descrito na seção 4.1.1. Aqui, o **Passo 3** está dividido em duas partes: o pré-processamento (ordenação dos padrões) e o arredondamento.

Passo 3.1: { *pré-processamento* }

Ordene o vetor solução $\boldsymbol{x}^k \in \Re^n$ obtido no **Passo 2** segundo um critério a ser definido a seguir para cada uma das três versões da heurística nova.

Passo 3.2: { *arredondamento* }

Suponha que N padrões de corte gerados na solução ótima $\boldsymbol{x}^k$ no **Passo 2** tenham freqüência positiva ($x_1^k > 0, \cdots, x_N^k > 0$)

Para $i = 1, \ldots, N$, faça:

$y_i^k = \lceil x_i^k \rceil$ e $y_j^k = 0$, $j = i+1,...,m$. {i.e., $\boldsymbol{y}^k = (y_1^k, ..., y_i^k, 0,...,0)$}

Enquanto $\boldsymbol{y}^k$ for infactível (ou seja, há itens em excesso), faça: $y_i^k = y_i^k - 1$.

Após $i = N$, todas as freqüências foram 'arredondadas' (algumas freqüências foram arredondadas possivelmente para o inteiro superior, outras para o inteiro inferior ou valores menores, podendo-se até mesmo anular uma freqüência maior que um), gerando-se uma solução aproximada inteira $\boldsymbol{y}^k$.

4.2.6 Heurística nova 1

Para definirmos a heurística residual nova 1, basta definirmos o Passo 3.1 do algoritmo anterior, que trata da ordenação dos padrões de corte. Na versão 1, os padrões são ordenados de forma não-decrescente da freqüência de utilização, ou seja, damos prioridade aos padrões mais utilizados.

Passo 3.1: { *pré-processamento* }

Ordene o vetor solução $\boldsymbol{x}^k \in \Re^n$ obtido no **Passo 2** tal que: $x_1^k \geq x_2^k \geq \ldots \geq x_N^k$.

4.2.7 Heurística nova 2

Para definirmos a heurística residual nova 2, basta definirmos o **Passo 3.1** do algoritmo anterior, que trata da ordenação dos padrões de corte. A heurística nova 2 ordena os padrões de corte conforme a perda que eles apresentam, de forma não-crescente, assim, tentamos utilizar primeiro os padrões de corte com a menor perda.

Passo 3.1: { *pré-processamento* }

Seja w_j a perda no padrão de corte j, $j = 1, \ldots, N$, ou seja, $w_j = L - \sum_{i=1}^{m} \ell_i \alpha_{ij}$, $\forall j$;

Ordene o vetor solução $\boldsymbol{x}^k$ obtido no **Passo 2** tal que: $w_1 \leq w_2 \leq \ldots \leq w_N$.

4.2.8 Heurística nova 3

Para definirmos a heurística residual nova 3, basta definirmos o **Passo 3.1** do algoritmo anterior, que trata da ordenação dos padrões de corte. A heurística nova 3 ordena os padrões de corte conforme a parte fracionária das freqüências dos padrões de corte.

Passo 3.1: { *pré-processamento* }

Seja $f_j = x_j - \lfloor x_j \rfloor$, $j = 1, \ldots, N$. (f é a parte fracionária).

Ordene o vetor solução $\boldsymbol{x}^k$ obtido no **Passo 2** tal que: $f_1 \geq f_2 \geq \ldots \geq f_N$.

4.2.9 Exemplo numérico

Para ilustrarmos a heurística nova, apresentamos um exemplo numérico. Suponha que temos disponível em estoque objetos de comprimento $L = 100$, em quantidade suficiente para atender toda a demanda de itens que está especificada na Tabela 1, a seguir.

Tabela 1 – Dados dos itens demandados.

m = 3	***comprimento***	***demanda***
1	34	6
2	28	9
3	16	3

Inicialmente, resolvemos o problema pelo método simplex com geração de colunas e obtemos a seguinte solução, dada na Tabela 2:

Tabela 2 – Solução pelo método simplex com geração de colunas – primeira iteração da heurística nova 1.

x	***Padrão de corte*** $(\alpha_{1j}, \alpha_{2j}, \alpha_{3j})^T$
2,5714	$(2, 1, 0)^T$
2,1428	$(0, 3, 1)^T$
0,4285	$(2, 0, 2)^T$

As componentes do vetor $\boldsymbol{x}$ apresentadas na Tabela 2 já estão ordenadas conforme critério da heurística nova 1. Assim, a primeira freqüência a ser arredondada é $x_1 = 2{,}5714$, o que nos fornece a solução parcial $\boldsymbol{y} = (\lceil x_1 \rceil, 0, 0)^T = (\mathbf{3}, 0, 0)^T$. Esta solução é aceita pois produz $(6, 3, 0)^T$ sem exceder a demanda de $(6, 9, 3)^T$. A segunda freqüência a ser arredondada é $x_2 = 2{,}1428$, o que nos fornece a solução $\boldsymbol{y} = (3, \mathbf{3}, 0)^T$. Esta solução não é aceita, pois produz $(6, 12, 3)^T$ o que excede a demanda de itens. Assim, diminuímos em uma unidade a última componente arredondada ($y_2 = y_2 - 1$, isto é, $y_2 = 3 - 1 = 2$) e temos agora a solução $\boldsymbol{y} = (3, \mathbf{2}, 0)^T$, que produz $(6, 9, 2)^T$ e não excede a demanda. Analisando a última componente do vetor solução $x_3 = 0{,}4285$, temos a solução $\boldsymbol{y} = (3, 2, \mathbf{1})^T$. Esta solução produz $(8, 9, 3)^T$, o que excede a demanda. Assim, diminuímos em uma unidade a última componente arredondada ($y_3 = y_3 - 1$, isto é, $y_3 = 1 - 1 = 0$) e obtemos a solução $\boldsymbol{y} = (3, 2, \mathbf{0})^T$. Todas as componentes do vetor solução $\boldsymbol{x}$ foram analisadas, então calculamos a demanda residual, que é dada por $\boldsymbol{r} = \boldsymbol{d} - \boldsymbol{A}\boldsymbol{y} = (0, 0, 2)^T$. Resolvemos o problema residual pelo método simplex com geração de colunas (neste exemplo, pela simplicidade, não seria necessário) e obtemos a solução dada na Tabela 3, a seguir:

Tabela 3 – Solução pelo método simplex com geração de colunas – segunda iteração da heurística nova 1.

x	***Padrão de corte***
1	$(0, 0, 2)^T$

Assim, chegamos ao final da execução da heurística nova 1, com a seguinte solução inteira, dada na Tabela 4:

Tabela 4 – Solução inteira obtida pela heurística nova 1.

y	***Padrão de corte***
3	$(2, 1, 0)^t$
2	$(0, 3, 1)^t$
1	$(0, 0, 2)^t$

Como $f(\mathbf{y}) = \lceil f(\boldsymbol{x}) \rceil = 6$, segue que é uma solução ótima.

5. Testes Computacionais

Foram realizados alguns testes computacionais para comparar os resultados obtidos pelas heurísticas propostas na literatura que foram revisados nesse artigo e as três heurísticas de arredondamento propostas. Os algoritmos do método simplex com geração de colunas (método simplex revisado (Arenales, 2003; Chvátal, 1983)), de resolução do problema da mochila restrito (enumeração implícita (Gilmore & Gomory, 1963; Marques & Arenales, 2006)) e os procedimentos heurísticos foram implementados em Delphi 5. O problema *bin-packing* foi resolvido pelo *software* CPLEX 7.5. Os testes foram executados em um micro-computador Pentium IV com 512 Mb de RAM.

Realizamos a análise empírica das heurísticas descritas neste artigo baseada em dois geradores aleatórios de exemplares. O gerador CUTGEN1 proposto por Gau & Wäscher (1995) tem sido bastante utilizado neste tipo de análise e está disponível em http://ctpc22.inescn.pt/esicup/. Entretanto, as demandas são geradas em torno de uma média pré-estabelecida, não apresentam grandes variações. Como pretendemos analisar as heurísticas propostas para situações com demandas muito baixas para alguns itens, descrevemos um novo gerador, baseado no CUTGEN1, porém de forma que as demandas sejam mais variadas dentro de um intervalo fixo. As conclusões podem ser contraditórias, típico em análise empírica, que depende do gerador dos exemplares, com um algoritmo apresentando desempenho melhor para os exemplares de um gerador do que o outro gerador. Apresentamos os resultados obtidos pelo novo gerador, na seção 5.1 e a seguir, na seção 5.2, os resultados obtidos pelo gerador CUTGEN1.

5.1 Um gerador aleatório

Nossos experimentos estão divididos em 10 classes de problemas e para cada classe foram gerados 20 problemas-teste. A seguir, mostramos em detalhes os critérios usados para gerar os dados e os resultados obtidos.

- *Dimensão do objeto em estoque:* O comprimento do objeto em estoque: $L = 10000$.
- *Número de tipos de itens a serem produzidos:* Foram utilizados diferentes valores de $m = 10, 20, 30, 40$ e 50.

- *Dimensão dos itens a serem produzidos:* Os itens foram considerados em dois grupos: itens pequenos (P) e itens médios (M). Os comprimentos dos itens ℓ_i foram gerados aleatoriamente no intervalo $[v_1, v_2]$, em que $v_1 = 0{,}01L$ e $v_2 = 0{,}2L$, para os itens (P) e, $v_1 = 0{,}1L$ e $v_2 = 0{,}4L$, para os itens (M).
- *Demanda:* A demanda considerada é: $d \in [1, 10]$.

A Tabela 5 mostra para as 10 classes de problemas: os parâmetros que definem cada classe e o número de soluções ótimas obtidas (dentre os 20 problemas-teste). O critério utilizado para identificar uma solução ótima $f(\mathbf{y}) = \lceil f(\mathbf{x}) \rceil$. A última linha da Tabela 5 apresenta o total de exemplares para os quais uma solução ótima foi encontrada, por cada heurística, nos 200 exemplares. Pode-se observar, pela coluna da heurística nova 1 que nenhum dos exemplares gerados violou a propriedade IRUP. Como esperado, as classes de exemplares com itens pequenos (classes ímpares) são mais facilmente resolvidas e quase todas as heurísticas apresentam bons resultados.

Tabela 5 – Número exemplos cujas soluções ótimas foram obtidas, segundo critério dado na seção 3.

	Dados			***Número de soluções ótimas***							
				Construtivas		***Residuais***					
	m	***v_1***	***v_2***	***FFD***	***Gulosa***	***FFD***	***Gulosa***	***BPR***	***Nova 1***	***Nova 2***	***Nova 3***
C1	10	0,01	0,2	19	20	20	20	20	20	20	20
C2	10	0,1	0,4	3	6	19	20	20	20	20	19
C3	20	0,01	0,2	18	20	20	20	20	20	20	20
C4	20	0,1	0,4	2	10	18	20	20	20	20	20
C5	30	0,01	0,2	18	20	20	20	20	20	20	20
C6	30	0,1	0,4	0	9	13	18	18	20	19	18
C7	40	0,01	0,2	17	20	19	20	19	20	20	20
C8	40	0,1	0,4	0	8	18	20	18	20	20	20
C9	50	0,01	0,2	12	20	18	20	19	20	20	20
C10	50	0,1	0,4	0	8	9	19	9	20	20	20
Total				89	141	174	197	183	200	199	197

A Tabela 6 mostra o número médio de padrões de corte para cada uma das classes. O problema de determinar o menor número de padrões de cortes é um problema importante e vem sendo analisado na literatura (Haessler, 1975; Foerster & Wäscher, 2000; Vanderbeck, 2000; Limeira, 2003; Poldi & Arenales, 2003; Poldi, 2003; Umetami *et al.*, 2003; Salles Neto, 2005; entre outros). Aqui mostramos o número de padrões que cada uma das heurísticas gera, apenas como um subproduto das heurísticas, já que este objetivo não foi perseguido. As heurísticas novas tendem a resolver o problema de corte de estoque com um número menor de padrões de corte que as outras abordagens. Esta característica pode ser explicada devido ao esforço das heurísticas novas de utilizarem mais vezes os padrões gerados, quando buscam o arredondamento para o inteiro superior. Este pode ser considerado um sub-produto importante das heurísticas novas, já que o custo da preparação das máquinas para trocar padrões de corte pode ser alto e relevante para o processo produtivo como um todo, além de perdas pequenas.

Tabela 6 – Média dos padrões de corte obtidos dos 20 exemplares para cada uma das 10 classes.

	Número de padrões de corte							
	Construtivas		***Residuais***					
	FFD	***Gulosa***	***FFD***	***Gulosa***	***BPR***	***Nova 1***	***Nova 2***	***Nova 3***
C1	9,35	8,70	8,90	8,85	8,90	7,10	7,30	8,15
C2	13,20	11,95	11,35	11,30	11,30	9,55	9,75	10,85
C3	19,85	17,40	17,65	17,65	17,65	15,00	15,00	16,25
C4	25,05	21,60	23,45	23,30	23,25	18,00	18,45	20,70
C5	27,60	26,25	25,00	25,00	25,00	21,35	21,35	23,25
C6	39,05	29,95	34,85	34,60	34,85	26,10	26,25	29,90
C7	36,70	35,20	31,70	31,65	31,70	27,95	27,95	30,65
C8	51,35	39,85	45,95	45,75	46,00	34,05	34,20	38,65
C9	46,45	45,30	41,00	40,90	40,95	36,65	36,65	40,05
C10	63,55	47,20	56,00	55,40	56,00	41,35	41,80	46,95

As heurísticas arredondaram pelo menos uma das freqüências fracionárias (solução do problema linear) para o inteiro superior em 100% dos exemplos resolvidos. A Tabela 7 a seguir, mostra o número médio (dos 200 exemplares) de variáveis fracionárias que foram arredondadas para o inteiro superior. Por exemplo, para os exemplares da classe C1, a heurística nova 1 arredondou para o inteiro superior, em média, 4,35 variáveis, enquanto a versão 3 desta mesma heurística arredondou, em média, 5,30. A heurística nova 3 age mais gulosamente no arredondamento para inteiro superior, o que faz inibir o uso de outros padrões de corte gerados pela solução fracionária, deixando de encontrar soluções ótimas quando as outras versões da heurística nova encontraram, como atesta a Tabela 5.

Tabela 7 – Número médio de componentes do vetor solução que são arredondadas para o inteiro superior.

	Nova 1	***Nova 2***	***Nova 3***
C1	4,35	4,25	5,30
C2	4,00	4,00	4,65
C3	8,70	8,70	10,10
C4	10,60	10,30	12,40
C5	13,20	13,20	15,40
C6	15,60	15,65	18,05
C7	17,10	17,05	20,00
C8	21,70	21,70	24,30
C9	21,30	21,30	25,85
C10	25,90	25,60	29,65

O tempo computacional (média dos 200 exemplares) está na Tabela 8 a seguir:

Tabela 8 – Tempo computacional médio e número total de soluções ótimas dos 200 exemplares.

	Construtivas		***Residuais***					
	FFD	***Gulosa***	***FFD***	***Gulosa***	***BPR****	***Nova 1***	***Nova 2***	***Nova 3***
Tempo médio (seg)	0,00	0,24	6,18	6,21	6,11+87,1	2,96	2,97	2,92

* O primeiro valor de tempo é para determinar o problema residual e o segundo valor de tempo é para resolver o problema *bin-packing*.

Observação: Nos 17 exemplos em que a heurística *Bin-Packing Residual* não obteve uma solução ótima, o problema residual final foi executado pelo CPLEX até que não houvesse mais memória suficiente. Assim, o tempo computacional médio ficou comprometido, pois alguns exemplos chegaram a 10 horas, o que é inaceitável na prática em um ambiente industrial. O tempo computacional para estes 17 exemplos não está considerado na média exibida na Tabela 8.

5.2 O gerador CUTGEN1

Gau & Wäscher (1995) propõem um gerador de exemplares para o problema de corte de estoque unidimensional: O CUTGEN1 e Wäscher & Gau (1996) definem um conjunto de parâmetros para que 4000 exemplares fossem gerados aleatoriamente e usados para análise dos algoritmos estudados por eles; esses exemplares foram reproduzidos e resolvidos com as heurísticas propostas Considera-se o comprimento dos objetos disponíveis em estoque $L = 10000$, o número de tipos de itens demandados é $m = 10$, 20, 30, 40 e 50. O comprimento dos itens é gerado em relação ao objeto a ser cortado, considerou-se quatro intervalos para o comprimento dos itens: $\ell_i \in [v_1, v_2]$, em que $v_1 = 0{,}0001L$ e $v_2 = 0{,}25L$; $0{,}50L$; $0{,}75L$ e $1{,}00L$.

A demanda média utilizada foi: $\overline{d} = 10$ e 50, isto é, demanda média de 10 unidades de cada item e demanda média de 50 unidades de cada item. Para cada classe, foram gerados e resolvidos 100 exemplares. Os resultados são mostrados na Tabela 9. Para cada classe m, na tabela, temos 800 exemplares, ou seja, 400 com demanda média de 10 unidades e 400 com demanda média de 50 unidades. Esses 400 exemplares estão, por sua vez, divididos em quatro classes de exemplos, com 100 exemplos em cada. O que caracteriza cada classe é o comprimento dos itens em relação ao comprimento do objeto a ser cortado, conforme foi descrito no parágrafo anterior, os valores utilizados foram: $v_2 = 0{,}25L$; $0{,}50L$; $0{,}75L$ e $1{,}00L$. para cada classe de exemplos gerada, uma semente foi utilizada. A semente utilizada no gerador é formada por sete dígitos, os dois primeiros são referentes ao número de tipos de itens (m), os próximos três dígitos são referentes ao valor de v_2 (025, 050, 075 e 100). Os dois últimos dígitos que completam a semente são referentes a demanda $\overline{d}$ (10 e 50). Por exemplo, a primeira classe de exemplos gerada, com $m = 10$ tipos de itens, $v_2 = 0{,}25$ e demanda $\overline{d} = 10$, a semente é dada por: 1002510.

As colunas da Tabela 9 mostram o número de soluções ótimas obtidas pelos cinco métodos de solução residuais. A última linha da Tabela 9 mostra o total de exemplos (dente os 4000) que foram resolvidos à otimalidade, segundo critério dado na seção 3.

Tabela 9 – Número exemplos cujas soluções ótimas foram obtidas, segundo critério dado na seção 3.

	Residuais				
m	***FFD***	***Gulosa***	***Nova 1***	***Nova 2***	***Nova 3***
10	787	799	775	767	774
20	780	789	774	759	773
30	771	764	782	758	784
40	773	746	786	763	790
50	760	710	784	765	777
Total	3871	3808	3901	3812	3898

As heurísticas destacadas por Wäscher & Gau (1996) resolvem o problema *bin-packing* para o último problema residual, o que demanda grande esforço computacional (veja Tabela 8). Entretanto, a heurística nova 1 obteve resultados similares sem a necessidade resolver o problema *bin-packing*, com um tempo computacional pequeno.

Em uma tentativa de gerar exemplos com baixa demanda, utilizamos o CUTGEN1 com os mesmo parâmetros descritos anteriormente, a menos da demanda, que passamos a considerar $\overline{d} = 2$. Os resultados estão na Tabela 10.

Tabela 10 – Número exemplos com soluções ótimas (de 400 exemplos), $\overline{d} = 2$.

	Construtivas		***Residuais***				
m	***FFD***	***Gulosa***	***FFD***	***Gulosa***	***Nova 1***	***Nova 2***	***Nova 3***
10	381	342	394	400	400	400	400
20	367	207	389	394	400	399	400
30	365	158	389	381	398	395	399
40	356	129	381	375	399	393	400
50	333	105	374	334	395	390	396
Total	1802	941	1927	1884	1992	1977	1995

Os resultados na Tabela 10 reforçam as conclusões de Wäscher & Gau (1996) e da seção 5.2, que as heurísticas residuais apresentam melhores desempenhos quando comparadas com as construtivas, mesmo para problemas com baixa demanda. Em particular, as heurísticas novas foram superiores.

5.3. Análise dos resultados

Podemos observar na Tabela 5 que a heurística residual nova 1 obteve solução ótima para todos os exemplares gerados. Como já apontado por Wäscher & Gau (1996), a abordagem de resolver o problema original relaxado da condição de integralidade (usando o método simplex com geração de colunas) e procurar uma estratégia de arredondamento é bem eficiente (heurísticas residuais). Nossos resultados corroboram com esta conclusão, como também mostram que é válida mesmo para problemas com baixa demanda de itens. Além disso, as heurísticas novas, também do tipo residual, melhoram a superioridade já salientada.

Quanto ao tempo computacional, a heurística residual *bin-packing* é muito cara, devido à complexidade do problema *bin-packing* e o tempo utilizado pelo *software* CPLEX 7.5 é alto, o que inviabiliza este tipo de método de solução na prática, em ambiente industrial. As demais heurísticas exigem esforço computacional muito menor quando comparadas à heurística residual *bin-packing*. As heurísticas construtivas se mostram bastantes sensíveis ao gerador, enquanto que as residuais foram mais robustas.

Quanto ao número de padrões de corte, podemos notar na Tabela 6 que a heurística nova 1 obteve em média o menor número de padrões de corte. Embora este não tenha sido o objetivo perseguido pelos métodos, foi destacado por ser um sub-produto importante e como motivação para investigações futuras.

6. Conclusões e Perspectivas

Neste trabalho revisamos alguns procedimentos heurísticos para obtenção de soluções inteiras para o problema de corte de estoque unidimensional e apresentamos novas heurísticas. Claramente, os procedimentos residuais foram superiores quando comparados com as heurísticas de repetição exaustiva FFD e gulosa. Quando a demanda, ou parte dela, é baixa, há um folclore na área de corte e empacotamento, de que a geração de colunas não seria adequada e heurísticas construtivas deveriam ser utilizadas. Em nosso entender, este "saber folclórico" se deve ao fato de que a solução inteira aproximada ser basicamente obtida pelo truncamento (arredondamento para o inteiro inferior). Isto faz com que os padrões obtidos pela técnica de geração de colunas sejam inúteis, quando a demanda é muito baixa. Contudo, as heurísticas construtivas produzem os primeiros padrões muito bons, porém os padrões finais apresentam perdas altas. Os resultados deste trabalho indicam que, mesmo diante de demandas pequenas, é mais adequado o uso da técnica de geração de colunas para resolver o problema de corte inteiro (i.e., abordagens residuais), em particular as heurísticas novas, que mostraram bom desempenho quanto à qualidade da solução e o tempo computacional. Do conhecimento dos autores, as heurísticas novas propostas neste artigo são as únicas da literatura que evitam o problema residual final. A heurística de Stadtler, que também usa uma estratégia de arredondar para o inteiro superior uma das freqüências por iteração, deve resolver um problema residual final. Além disso, as heurísticas aqui apresentadas podem ser estendidas para problemas com vários tamanhos de objetos em estoque (Poldi, 2003), bem como para problemas de dimensões maiores.

Agradecimentos

Os autores agradecem aos revisores anônimos pelos comentários que enriqueceram o trabalho. Este trabalho teve apoio da FAPESP e CNPq.

Referências Bibliográficas

(1) Arenales, M.N. (2003). Problemas de corte e empacotamento. **In:** *XI Escola de Informática da SBC Paraná* [edited by M.A. Camargo-Brunetto], 09/2003, capítulo **5**, ISBN 858844262-0.

(2) Carvalho, J.M.V. (1999). Exact solution of bin-packing problems using column generation and branch-and-bound. *Annals of Operations Research*, **86**, 629-659.

(3) Chvátal, V. (1983). *Linear programming*. W.H. Freeman.

(4) Dyckhoff, H. (1990). A typology of cutting and packing problems. *European Journal of Operational Research*, **44**, 145-159.

(5) Foerster, H. & Wäscher, G. (2000). Pattern reduction in one-dimensional cutting stock problem. *International Journal of Production Research*, **38**, 1657-76.

(6) Garey, M.R. & Johnson, D.S. (1979). *Computers and intractability: a guide to the NP-completeness*. W.H. Freeman.

(7) Gau, T. & Wäscher, G. (1995). CUTGEN1: A problem generator for the standard one-dimensional cutting stock problem. *European Journal of Operational Research*, **84**, 572-579.

(8) Gilmore, P.C. & Gomory, R.E. (1961). A linear programming approach to the cutting stock problem. *Operations Research*, **9,** 848-859.

(9) Gilmore, P.C. & Gomory, R.E. (1963). A linear programming approach to the cutting stock problem – Part II. *Operations Research*, **11**, 863-888.

(10) Haessler, R.W. (1975). Controlling cutting pattern changes in one-dimensional trim problems. *Operations Research*, **23**, 483-93.

(11) Hinxman, A. (1980). The trim-loss and assortment problems: a survey. *European Journal of Operational Research*, **5**, 8-18.

(12) Hoto, R.; Maculan, N.; Prado, F.M. & Arenales, M.N. (2003). Um problema de corte com padrões compartimentados. *Pesquisa Operacional*, **23**, 169-187.

(13) Limeira, M.S. (2003). Redução do número de padrões em problemas de corte de estoque. Tese de Doutorado, INPE – Instituto Nacional de Pesquisas Espaciais, São José dos Campos/SP.

(14) Marcotte, O. (1986). An instance of the cutting stock problem for which the rounding property does not hold. *Operations Research Letters*, **4**, 239-243.

(15) Marques, F.P. & Arenales, M.N. (2006). The constrained compartmentalised knapsack problem. *Computer & Operations Research* (a aparecer).

(16) Martello, S. & Toth, P. (1990). *Knapsack problems: algorithms and computer implementations*. John Wiley.

(17) Pinto, M.J. (1999). O problema de corte de estoque inteiro. Dissertação de Mestrado, USP – Universidade de São Paulo, ICMC – Instituto de Ciências Matemáticas e de Computação, São Carlos/SP.

(18) Poldi, K.C. & Arenales, M.N. (2003). Sobre uma heurística de redução de padrões de corte para o problema de corte de estoque. *Revista TEMA – Tendências em Matemática Aplicada e Computacional*, **4**, 227-236.

(19) Poldi, K.C. (2003). Algumas extensões do problema de corte de estoque unidimensional. Dissertação de Mestrado, USP – Universidade de São Paulo, ICMC – Instituto de Ciências Matemáticas e de Computação, São Carlos/SP.

(20) Riehme, J.; Scheithauer, G. & Terno, J. (1996). The solution of two-stage guillotine cutting stock problems having extremely varying order demands. *European Journal of Operational Research*, **91**, 543-552.

(21) Rietz, J.; Scheithauer, G. & Terno, J. (2000). Tighter bounds for the gap and non-IRUP constructions in the one-dimensional cutting stock problem. Manuskript, Technische Universität Dresden. Dresden, Alemanha.

(22) Salles Neto, L.L. (2005). Modelo não-linear para minimizar o número de objetos e o setup num problema de corte unidimensional. Tese de doutorado, UNICAMP – Universidade Estadual de Campinas, Campinas/SP.

(23) Scheithauer, G. & Terno, J. (1995). The modified integer round-up property of the one-dimensional cutting stock problem. *European Journal of Operational Research*, **84**, 562-571.

(24) Schwerin, P. & Wäscher, G. (1997). The bin-packing problem: a problem generator and some numerical experiments with FFD packing and MTP. *International Transactions in Operations Research*, **4**, 377-389.

(25) Stadtler, H. (1990). A one-dimensional cutting stock problem in the aluminium industry and its solution. *European Journal of Operational Research*, **44**, 209-223.

(26) Umetami, S.; Yagiura, M. & Ibaraki, T. (2003), One-dimensional cutting stock problem to minimize the number of different patterns. *European Journal of Operational Research*, **146**, 388-402.

(27) Vanderbeck, F. & Wolsey, L.A. (1996). An exact algorithm for IP column generation. *Operations Research Letters*, **19**, 151-159.

(28) Vanderbeck, F. (2000). Exact algorithm for minimizing the number of setups in the one-dimensional cutting stock problem. *Operations Research*, **48**, 915-26.

(29) Wäscher, G. & Gau, T. (1996). Heuristics for the integer one-dimensional cutting stock problem: a computational study. *OR Spektrum*, **18**, 131-144.