Aplicação do Algoritmo $\varepsilon-$Restrito com uma Heurística de Arredondamento no Problema de Corte Unidimensional Inteiro Multiobjetivo

Angelo Aliano Filho
IMECC - UNICAMP
Rua Sérgio Buarque de Holanda, Campinas, SP, Brasil
angeloaliano@hotmail.com

Antônio Carlos Moretti
IMECC - UNICAMP
Rua Sérgio Buarque de Holanda, Campinas, SP, Brasil
moretti@ime.unicamp.br

Margarida Vaz Pato
ISEG e CMAFCIO, ULisboa
Rua do Quelhas nº 6, Lisboa, Portugal
mpato@iseg.utl.pt

RESUMO

O presente trabalho trata o Problema de Corte Unidimensional Inteiro Multiobjetivo. Este problema possui uma enorme importância prática, no entanto, é de um alto nível de complexidade. O modelo biobjetivo considerado visa minimizar a frequência de padrões de corte para atender à demanda mínima para cada item requisitado e ao número de diferentes padrões a usar, sendo estas metas conflitantes entre si. Neste estudo, aplicou-se o método $\varepsilon-$Restrito para obter a fronteira de Pareto aproximada para este problema. Além disso, um procedimento de arredondamento de soluções não inteiras é apresentado, assim, acelerando a obtenção de soluções potencialmente não-dominadas.

PALAVRAS CHAVE. Otimização multiobjetivo, Métodos clássicos, Problema de corte.

Área Principal: Corte e empacotamento, Otimização combinatória, Otimização multiobjetivo

ABSTRACT

This paper deals with the Multiobjective One-Dimensional Cutting Stock Problem. This problem has an enormous practical importance, however, it is of a high level of complexity. The bi-objective model aims to minimize the frequency of cutting patterns to attend to the minimum required demand for each item and simultaneously minimize the number of different patterns to be used. These objectives are conflicting. In this study, we applied the $\varepsilon-$Constrained method to obtain an approximation to the Pareto front of the problem. Also, a rounding procedure for the non-integer solutions is presented, thus accelerating the obtaining of potentially non-dominated solutions.

KEYWORDS. Multiobjective optimization. Classical methods. Cutting problem

1. Introdução

O Problema de Corte Unidimensional Inteiro (PCUI) é um dos problemas mais estudados dentro do campo da otimização combinatória devido, principalmente, a sua aplicabilidade no mundo da engenharia de produção, fazendo parte do planejamento de uma diversidade de indústrias cujos procedimentos envolvem corte de papel, móveis, vidro, plásticos, tecido, entre outras matérias. Apesar de fácil entendimento e modelagem, este problema tem um elevado nível de complexidade, tendo sido classificado na literatura como $\mathcal{NP}$-difícil [2].

Este trabalho visa resolver o Problema de Corte Unidimensional Inteiro Multiobjetivo (PCUIM), que busca minimizar concomitantemente a quantidade de padrões de corte a cortar para atender à mínima demanda solicitada e ao número de padrões de corte distintos, denominado de *setup*. O segundo objetivo tem sua importância, pois a mudança de um padrão de corte para outro envolve vários agravantes, como custo e tempo para mudanças/ajustes das facas nas máquinas que fazem a cortagem do material. Assim, faz-se sendo necessário estabelecer um compromisso entre estas duas metas para ajudar o decisor/gestor numa tomada de decisão.

O modelo matemático biobjetivo resultante tem vários agravantes, que dificultam consideravelmente sua resolução. O simples fato de se considerar o *setup* faz o modelo duplicar o número de variáveis inteiras, aumentando o grau de complexidade para resolvê-lo. Poucos trabalhos na literatura abordam tal consideração. O único trabalho encontrado que utiliza métodos clássicos e que aparenta com o presente é o de [10], há 15 anos. No entanto, aplica somente à instâncias muito pequenas e não gera a fronteira de Pareto como aqui. Outro trabalho mais recente que considera *setup* num PCUI é dado em [11], onde uma suavização da função objetivo é adotada. No entanto, esse trabalho não explora o aspecto bi-objetivo deste problema, tratando as funções-objetivo de forma ponderada. Outro trabalho que considera múltiplos objetivos no PCUI é o artigo [9]. Nesse trabalho, uma fronteira de Pareto é obtida para o PCUIM, mas utilizando-se algoritmos evolutivos.

Um outro grande fator que torna o PCUIM mais difícil se comparado com o PCUI, é a existência de várias soluções ótimas que são igualmente importantes ao problema. Esse fato é gerado pelo conflito existente entre os objetivos, isto é, não existe uma única solução que os otimize ao mesmo tempo. Nesse caso, esse conjunto não unitário de soluções, é chamado de *eficiente*, fornecendo um compromisso entre os objetivos levados em consideração.

As soluções eficientes podem ser geradas por estratégias de escalarizações, isto é, o problema multiobjetivo é transformado num problema escalar que, ao ser otimizado, gera uma solução eficiente. No entanto, ao escalarizar o PCUIM, os sub-problemas gerados são PCUIs, de complexidade $\mathcal{NP}$-difícil. Dito de outra forma, a geração de t soluções eficientes para o PCUIM demanda a resolução de pelo menos t problemas $\mathcal{NP}$-difíceis. Em conclusão, obter todas as soluções eficientes deste problema é uma tarefa extremamente cara do ponto de vista computacional.

Muitos trabalhos já foram desenvolvidos na área de otimização combinatória bi e multiobjetivo. Pode-se citar as referências [5], [14], [15] e [16] que desenvolveram métodos específicos e que geram completamente a fronteira de Pareto para problemas inteiros. Existem muitos outros métodos mais específicos e elaborados, que fazem uso de restrições, normas-p, norma-∞, conceitos de não-dominâncias para encontrar novas soluções. Todos estes métodos são revistos e pontuados no artigo [1].

Embora exista muitos algoritmos e métodos de otimização multiobjetivo, a proposta deste estudo foi aplicar a técnica de escalarização $\varepsilon-$Restrito no PCUIM, a fim de enumerar totalmente o conjunto de soluções eficientes. Sua escolha é motivada pela sua simplicidade na implementação e por desejáveis características teóricas, explanadas ao longo do texto. Um outro aspecto explorado neste trabalho consiste em relaxar as condições de integralidade do número de padrões utilizados e aplicar um algoritmo especialmente desenvolvido de pós-otimização para obter, a partir destas soluções não inteiras, soluções factíveis para o PCUIM. Essa ideia tem a finalidade de reduzir o esforço computacional para resolver estes sub-problemas, gerando-se mais rapidamente uma fronteira de Pareto aproximada.

Estes métodos foram explorados e testados, e mostram ser ferramentas aplicáveis a instâncias do PCUIM de moderada dimensão.

Este trabalho está organizado em seis capítulos. O Capítulo 2 apresenta os modelos para o PCUIM e a notação empregada bem como uma heurística de arredondamento de soluções especialmente desenvolvida para o PCUI. Os Capítulos 3 e 4 ilustram como o $\varepsilon-$Restrito foi utilizado. O Capítulo 5 trás alguns resultados preliminares e, finalmente, no Capítulo 6 pontuam-se algumas considerações e futuras direções de pesquisa.

2. Modelagem Matemática

Para modelar o PCUIM, considera-se b rolos-mestre em estoque, de largura L_k com $k \in K = \{1, ..., b\}$ e m o número de itens demandados. Cada item i tem largura $l_i < L_k$ e ao menos d_i unidades precisam ser produzidas a fim de atender a demanda necessária, $i \in I = \{1, ..., m\}$. Os objetivos consiste em minimizar o número de peças a cortar a fim de atender à esta demanda mínima exigida e minimizar o número de padrões de corte. Do ponto de vista operacional, apenas faz sentido cortar um número inteiro de padrões destes rolos-mestres.

Nesta formulação, um padrão de corte é um vetor $m-$dimensional $a_{jk} = (a_{1jk}, a_{2jk}, ..., a_{mjk})^T$ onde a_{ijk} denota o número de vezes que o item i é cortado no padrão de corte j, na bobina mestre k.

Um padrão de corte é factível se

$$\begin{aligned} \sum_{i=1}^{m} l_i \cdot a_{ijk} &\leq L_k, \\ \sum_{i=1}^{m} l_i \cdot a_{ijk} &\geq L_k - \Delta, \\ \sum_{i=1}^{m} a_{ijk} &\leq q, \end{aligned}$$

onde $\Delta = \min_{1 \leq i \leq m} \{l_i\}$ e q é o número de facas máximo permitido para produzi-lo.

A Figura (1) ilustra $m = 4$ itens sendo cortados de dois padrões de corte de um mesmo rolo-mestre, e dá uma ideia de como o processo de cortagem de materiais em uma dimensão é realizado.

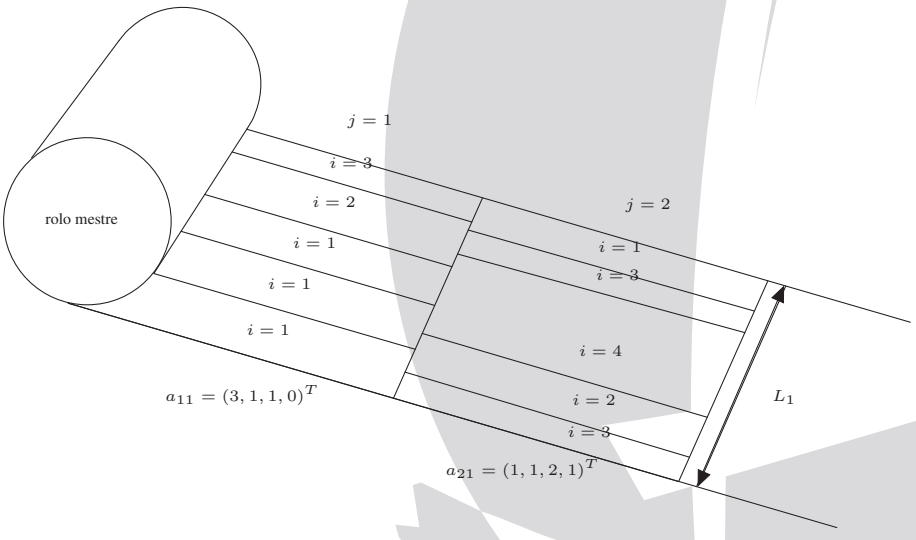

Figura 1: Exemplo de padrões de corte com um rolo-mestre

A variável x_{jk} indica a frequência do padrão de corte j na bobina mestre k, com $j \in P$, onde P é o conjunto dos padrões de corte, $|P| = p$ (que pode ser fornecido pelo usuário ou não). A variável y_{jk} contabiliza o *setup*, isto é,

$$y_{jk} = \begin{cases} 1, & \text{se } x_{jk} > 0 \\ 0, & \text{caso contrário,} \end{cases}$$

para todo $j \in P$ e $k \in K$.

Apresenta-se um modelo não-linear bi-objetivo, onde se admite que os padrões de corte *não* são fornecidos pelo usuário, a priori. Este modelo tem prós e contras. Um fator pró consiste no fato de se ter garantia de as soluções ótimas para linearização adotada (fazendo-se com que a_{ijk} seja parâmetro em vez de variável) serem ótimas globais para o problema. A desvantagem consiste no aumento considerável do modelo em termos de variáveis e restrições, comprometendo a eficiência de qualquer pacote computacional de programação linear inteira para resolver o modelo linearizado. Sendo assim, o uso prático deste modelo se restringe apenas a exemplares com um número muito reduzido de itens.

O modelo não-linear para o PCUIM é apresentado a seguir, onde N é um limitante superior para x_{jk}.

$$\begin{aligned}
\text{Minimize} \quad & z_1 = \sum_{k=1}^{b}\sum_{j=1}^{p} x_{jk} \\
\text{Minimize} \quad & z_2 = \sum_{k=1}^{b}\sum_{j=1}^{p} y_{jk} \\
\text{sujeito a} \quad & \sum_{k=1}^{b}\sum_{j=1}^{p} a_{ijk} \cdot x_{jk} \geq d_i, \quad i \in I, \\
& \sum_{i=1}^{m} l_i \cdot a_{ijk} \leq L_k, && j \in P, \quad k \in K, \qquad (1) \\
& \sum_{i=1}^{m} l_i \cdot a_{ijk} \geq L_k - \Delta, && j \in P, \quad k \in K, \\
& \sum_{i=1}^{m} a_{ijk} \leq q, && j \in P, \quad k \in K, \\
& x_{jk} \leq N \cdot y_{jk}, && j \in P, \quad k \in K, \\
& x_{jk} \geq y_{jk}, && j \in P, \quad k \in K, \\
& y_{jk} \in \mathbb{B},\ x_{jk},\ a_{ijk} \in \mathbb{N}, && i \in I, \quad j \in P, \quad k \in K.
\end{aligned}$$

Uma alternativa para contornar o citado inconveniente consiste em fornecer apenas alguns padrões de corte para o modelo. A seguir, propõe-se uma técnica para tal obtenção.

2.1. Linearização e Aproximação do Modelo Não-Linear para o Problema de Corte

Este trabalho enfoca o modelo para o PCUIM onde os padrões de corte a_{jk} são fornecidos pelo algoritmo de Geração de Colunas (GC), pioneiramente desenvolvido por [7]. Para $b = 1$, tem-se $p = m$ padrões, mas caso $b > 1$, pode-se ter $p > m$ e uma adaptação do algoritmo de GC para tratar as restrições de estoque é dada em [13]. Nesse caso, é possível redefinir o PCUIM eliminando-se as restrições que impõem a definição de um padrão de corte factível. O modelo simplificado e linear, com os padrões de corte a^*_{jk} fornecidos pelo método de GC, é o seguinte:

$$
\begin{aligned}
\text{Minimize} \quad & z_1 = \sum_{k=1}^{b}\sum_{j=1}^{p} x_{jk} \\
\text{Minimize} \quad & z_2 = \sum_{k=1}^{b}\sum_{j=1}^{p} y_{jk} \\
\text{sujeito a} \quad & \sum_{k=1}^{b}\sum_{j=1}^{p} a^*_{ijk} \cdot x_{jk} \geq d_i, \quad i \in I, \\
& x_{jk} \leq N \cdot y_{jk}, \qquad\qquad j \in P, \quad k \in K, \\
& x_{jk} \geq y_{jk}, \qquad\qquad j \in P, \quad k \in K, \\
& y_{jk} \in \mathbb{B},\ x_{jk} \in \mathbb{N}, \qquad j \in P, \quad k \in K.
\end{aligned}
\tag{2}
$$

A partir desta formulação, é obtida uma fronteira de Pareto aproximada, se comparada com a fronteira resultante da formulação original em (1). Aqui nesta abordagem, tem-se apenas $2 \cdot p$ variáveis, sendo metade delas binárias e a outra metade inteiras.

Mesmo com esta simplificação, para instâncias com um número de itens da ordem de dezenas ou centenas, a aplicação de técnicas exatas de otimização multiobjetivo na formulação (2) pode levar a dificuldades computacionais, pois o modelo escalar deve ser re-otimizado várias vezes para a geração de soluções potencialmente eficientes. Uma maneira de acelerar este processo, consiste, relaxar as condições de integralidade de x_{jk}, obtendo-se uma formulação parcialmente relaxada, como a seguir:

$$
\begin{aligned}
\text{Minimize} \quad & z_1 = \sum_{k=1}^{b}\sum_{j=1}^{p} x_{jk} \\
\text{Minimize} \quad & z_2 = \sum_{k=1}^{b}\sum_{j=1}^{p} y_{jk} \\
\text{sujeito a} \quad & \sum_{k=1}^{b}\sum_{j=1}^{p} a^*_{ijk} \cdot x_{jk} \geq d_i, \quad i \in I, \\
& x_{jk} \leq N \cdot y_{jk}, \qquad\qquad j \in P, \quad k \in K, \\
& x_{jk} \geq y_{jk}, \qquad\qquad j \in P, \quad k \in K, \\
& y_{jk} \in \mathbb{B},\ x_{jk} \in \mathbb{R}_+, \qquad j \in P, \quad k \in K.
\end{aligned}
\tag{3}
$$

Esta relaxação se baseia no fato de ela ser forte para a formulação original, daí pode ser considerada uma *boa* aproximação para o modelo inteiro, como constata [17]. A vantagem é que nela se tem apenas p variáveis binárias, a metade do número de variáveis inteiras da formulação em (2). Isto pode representar um enorme ganho em termos computacionais para a geração de uma fronteira de Pareto aproximada para o PCUIM. A próxima seção apresenta um procedimento heurístico, capaz de arredondar eficientemente as soluções provenientes do modelo (3).

2.2. Heurística de Arredondamento

Alguns problemas de Programação Inteira, como o PCUI, possuem uma relaxação linear forte, logo pode-se aproveitar esta característica e conseguir soluções inteiras de boa qualidade por meio de métodos heurísticos de arredondamento. Um desses métodos pode ser encontrado no trabalho de [8].

Assim, para cada solução eficiente $\tilde{x}$ proveniente do modelo (3), propõe-se um algoritmo de arredondamento pós-otimização, organizado no pseudocódigo a seguir. Como entrada, fornece-se o vetor a ser arredondado $\tilde{x}$, a matriz A^* que contém os padrões de corte gerados pelo algoritmo de GC e o vetor das demandas d. Como saída deste procedimento, obtém-se a solução $\bar{x}$ integralizada pela heurística. Suponha por simplicidade que $b = 1$ neste algoritmo.

Algorithm 1 Arredondamento com melhoramento gradual

1: Input: $\tilde{x}$, $A^* = [a^*_{ij}]_{i\in I, j=1,...,p}$, d e $v_j = \{i : a^*_{ij} > 0\}$ para todo $j = 1, ..., p$
2: $\bar{x} \leftarrow \lceil \tilde{x} \rceil$
3: $r = A^*\bar{x} - d$
4: **while** $r \geq 0$ e na iteração anterior alguma variável diminuiu **do**
5: **for** $j = 1, ..., p$ **do**
6: **if** $\bar{x}_j > 0$ **then**
7: $\bar{x}_j \leftarrow \bar{x}_j - 1$
8: $r_i = r_i - a^*_{ij}$ para todo $i \in v_j$
9: **if** $r_i < 0$ para algum $i \in v_j$ **then**
10: $\bar{x}_j \leftarrow \bar{x}_j + 1$
11: $r_i = r_i + a^*_{ij}$ para todo $i \in v_j$
12: **end if**
13: **end if**
14: **end for**
15: **end while**
16: Output $\bar{x}$

A ideia básica do arredondamento com melhoramento consiste em tomar como primeira solução inteira a solução relaxada arredondada ao menor inteiro superior (linha 1). A seguir, analisa-se cada componente j do vetor $\tilde{x}$, diminuindo-se em uma unidade. Se ao final de uma iteração completa em que se conseguiu diminuir o valor de alguma das p variáveis o resíduo r é não negativo, a redução é autorizada para aquela componente, caso contrário não. Este procedimento é realizado para as demais posições deste vetor. Se ao final de uma iteração completa o resíduo r é positivo, a análise de redução inicia novamente, caso contrário, o procedimento é finalizado.

3. Método $\varepsilon-$Restrito no Problema de Corte Multiobjetivo Inteiro

Nesta abordagem, escolhe-se como função objetivo uma das funções que definem o problema multiobjetivo e as demais são tratadas como restrições. Resultados teóricos, que podem ser encontrados em [3], [4] e [12] atestam que este procedimento de escalarização é capaz de determinar todas as soluções eficientes para o problema multiobjetivo combinatório, desde que o lado direito ε das restrições adicionais seja convenientemente tomado. Neste problema é apresentado um processo de fazer esta variação de modo a conseguir todas estas soluções.

No PCUIM, foi mais conveniente considerar como função objetivo z_1 para o problema escalar P_ε e z_2 como restrição, pois a amplitude de variação desta função é muito menor se comparada com z_1. Geralmente $1 \leq z_2 \leq m$, e m é da ordem das dezenas. Por outro lado, dependendo dos valores da demanda, z_1 pode ter uma amplitude da ordem de milhares. Essa escolha foi determinante para que este procedimento de escalarização fosse aplicado aqui.

Então P_ε com a imposição de um *setup* de até ε é definido a seguir, onde $\rho > 0$ é uma pequena constante que multiplica z_2 e levada à função objetivo, a fim de evitar soluções não-eficientes:

$$
P_\varepsilon \left| \begin{array}{lll}
\text{Minimize} & \bar{z}_\varepsilon = \displaystyle\sum_{k=1}^{b}\sum_{j=1}^{p} x_{jk} + \rho \cdot \sum_{k=1}^{b}\sum_{j=1}^{p} y_{jk} & \\
\text{sujeito a} & \displaystyle\sum_{k=1}^{b}\sum_{j=1}^{p} a^*_{ijk} \cdot x_{jk} \geq d_i, & i \in I, \\
& x_{jk} \leq N \cdot y_{jk}, & j \in P, \quad k \in K, \\
& x_{jk} \geq y_{jk}, & j \in P, \quad k \in K, \\
& \displaystyle\sum_{j=1}^{p}\sum_{j=1}^{b} y_{jk} \leq \varepsilon, & \\
& y_{jk} \in \mathbb{B},\ x_{jk} \in \mathbb{Z}_+, & j \in P, \quad k \in K.
\end{array} \right.
$$

Os subproblemas escalares P_ε são resolvidos sequencialmente, variando-se ε. A cada sub-problema distinto, uma nova solução eficiente x^*_ε com *setup* até ε é obtida. A grande vantagem desta técnica é a geração dos pontos suportados e não-suportados na fronteira de Pareto, isto é, o algoritmo consegue determinar *todas* as soluções eficientes.

Como a segunda função objetivo para o PCUIM toma apenas valores inteiros num intervalo fechado, é razoável considerar os valores inteiros de ε variando no intervalo $\mathcal{I} = [z_2^-, z_2^+]$, onde z_2^- e z_2^+ é o menor e o maior *setup* que o problema pode ter, respectivamente. A cada valor de ε inteiro no intervalo $\mathcal{I}$, uma nova solução eficiente é obtida ao se resolver o problema P_ε. Conforme a restrição $z_2 \leq \varepsilon$ vai sendo deslocada verticalmente, outras soluções de Pareto vão sendo geradas. A imposição desta restrição e a minimização na direção de $z_1 + \rho \cdot z_2$, "forçam" a obtenção de soluções eficientes.

O valor de z_2^- consegue-se por resolução do subproblema P_ε^- a seguir:

$$
P_\varepsilon^- \left| \begin{array}{lll}
\text{Minimize} & \bar{z}_\varepsilon = \displaystyle\sum_{k=1}^{b}\sum_{j=1}^{p} y_{jk} + \rho \cdot \sum_{k=1}^{b}\sum_{j=1}^{p} x_{jk} & \\
\text{sujeito a} & \displaystyle\sum_{k=1}^{b}\sum_{j=1}^{p} a^*_{ijk} \cdot x_{jk} \geq d_i, & i \in I, \\
& x_{jk} \leq N \cdot y_{jk}, & j \in P, \quad k \in K, \\
& x_{jk} \geq y_{jk}, & j \in P, \quad k \in K, \\
& y_{jk} \in \mathbb{B},\ x_{jk} \in \mathbb{Z}_+, & j \in P, \quad k \in K,
\end{array} \right.
$$

e calculando-se o *setup* da solução resultante. Analogamente, o valor para z_2^+ pode ser calculado resolvendo-se o problema P_ε^+

$$
P_\varepsilon^+ \left| \begin{array}{lll}
\text{Minimize} & \bar{z}_\varepsilon = \displaystyle\sum_{k=1}^{b}\sum_{j=1}^{p} x_{jk} + \rho \cdot \sum_{k=1}^{b}\sum_{j=1}^{p} y_{jk} & \\
\text{sujeito a} & \displaystyle\sum_{k=1}^{b}\sum_{j=1}^{p} a^*_{ijk} \cdot x_{jk} \geq d_i, & i \in I, \\
& x_{jk} \leq N \cdot y_{jk}, & j \in P, \quad k \in K, \\
& x_{jk} \geq y_{jk}, & j \in P, \quad k \in K, \\
& y_{jk} \in \mathbb{B},\ x_{jk} \in \mathbb{Z}_+, & j \in P, \quad k \in K,
\end{array} \right.
$$

e, novamente, calculando-se o *setup* desta solução. Denota-se por $\mathcal{Z}^*$, o conjunto de soluções não-dominadas obtidas por este procedimento.

Como visto, se o PCUIM possui $t \geq 2$ soluções de Pareto, então serão necessários $t-2$ subproblemas P_ε a serem resolvidos e mais dois subproblemas auxiliares para determinar os pontos

lexicográficos, P_{ε}^{-} e P_{ε}^{+}. Cada um desses envolve $2 \cdot p$ variáveis inteiras e $m + 2 \cdot p \cdot b$ restrições. O custo computacional de toda esta operação pode ficar extremamente caro à medida que m e t aumentarem, e este crescimento pode ser exponencial. Na próxima seção, é proposto um método heurístico que aproxima o conjunto das soluções eficientes com um custo computacional inferior.

4. Método $\varepsilon-$Restrito no Problema de Corte Multiobjetivo Inteiro Parcialmente Relaxado

Como destaca o trabalho de [11], o PCUI possui uma descontinuidade e não-linearidade na função objetivo e que, desta forma, não é possível relaxar concomitantemente as variáveis x_{jk} e y_{jk} e usar o método de GC. Para contornar este inconveniente e acelerar a obtenção da fronteira de Pareto, em vez de resolver os subproblemas P_{ε}, propõe-se relaxar as condições de integralidade apenas das variáveis x_{jk}. Isto é, aplica-se o algoritmo GC no PCUI sem levar em conta o *setup*, obtendo-se as colunas a_{jk} e as frequências potencialmente não inteiras. Após isto, usam-se os subproblema PM_{ε} de programação linear inteiro misto no algoritmo $\varepsilon-$R como anteriormente. A grande vantagem é que estes subproblemas têm apenas p variáveis inteiras (binárias), precisamente as que controlarão o *setup*.

Aplicando-se o mesmo algoritmo apresentado na seção anterior com estes subproblemas parcialmente relaxados, obtém-se uma fronteira de Pareto do problema parcialmente relaxado $\tilde{\mathcal{Z}}$ como resultado. Após esta fase, o procedimento de arredondamento de soluções proposto na seção 2.1 é empregado. Assim se obtém um conjunto $\bar{\mathcal{Z}}$ de soluções inteiras e factíveis para o PCUIM. Após arredondar todas as soluções eficientes da relaxação parcial, recalcula-se o *setup* de cada solução e extrai-se apenas os pontos não-dominados. A seção de experimentos computacionais atesta que as fronteiras $\mathcal{Z}^*$ e $\bar{\mathcal{Z}}$ ficam muito próximas, porém, $\bar{\mathcal{Z}}$ é muito mais rapidamente obtida.

5. Resultados Computacionais

Os testes computacionais e os algoritmos desenvolvidos para este problema foram codificados usando o software Matlab, versão 7.10.0 R2010a. Os subproblemas foram resolvidos através da API do CPLEX 12.5 com este software. Os algoritmos foram simulados em um computador Core i7 com 8GB de memória, do Instituto de Matemática, Estatística e Computação Científica - IMECC/UNICAMP, Campinas, SP, Brasil. Comparações com outros algoritmos não foram realizadas neste estudo, tendo em vista as diferença na maneira de obter as soluções eficientes pelos métodos concorrentes, como [9] e [10].

A fim de testar e comparar os métodos desenvolvidos, gerou-se randomicamente instâncias para o PCUIM através de uma adaptação do gerador CUTGEN desenvolvido por [6], amplamente utilizado na literatura. Esta adaptação foi baseada no trabalho de [13]. As instâncias foram divididas em 27 classes, cada uma com 20 problemas teste.

As 27 classes contemplam as combinações possíveis entre itens pequenos, médios e grandes, quantidade de itens e número de rolos-mestre, da seguinte forma:

- O número de itens m demandados foi igual a 10 e 20 e 40.
- As larguras dos itens l_i foram geradas entre $[v_1 \cdot \bar{L}, v_2 \cdot \bar{L}]$, onde L_k foi escolhido no intervalo $[300, 1.000]$ e $\bar{L}$ é a média dos valores L_k. Para itens pequenos (P) utilizou-se $v_1 = 0,01$ e $v_2 = 0,2$, para itens de diversos tamanhos (M) $v_1 = 0,01$ e $v_2 = 0,8$ e para itens grandes (G) utilizou-se $v_1 = 0,2$ e $v_2 = 0,8$.
- O número de facas foi fixo, igual a $q = \left\lceil \sum_{i=1}^{m} \frac{L}{m \cdot l_i} \right\rceil$.
- A demanda d_i para cada item foi gerada aleatoriamente no intervalo $[10, 200]$.
- O número de rolos-mestre b variou entre 1, 3 e 5.

As características de cada classe são mostradas na Tabela (1).

Tabela 1: Classes para o PCUIM

Classe	m	Tipo do item	b	Classe	m	Tipo do item	b
1	10	P	1	14	20	M	3
2	10	P	3	15	20	M	5
3	10	P	5	16	20	G	1
4	10	M	1	17	20	G	3
5	10	M	3	18	20	G	5
6	10	M	5	19	40	P	1
7	10	G	1	20	40	P	3
8	10	G	3	21	40	P	5
9	10	G	5	22	40	M	1
10	20	P	1	23	40	M	3
11	20	P	3	24	40	M	5
12	20	P	5	25	40	G	1
13	20	M	1	26	40	G	3
-	-	-	-	27	40	G	5

Na Tabela (2) ilustra-se algumas métricas que comparam as fronteiras[1] $\mathcal{Z}^*$ e $\bar{\mathcal{Z}}$ descritas a seguir:

- σ^1: denota a porção de soluções de $\bar{\mathcal{Z}}$ que *não* coincidiram com as de $\mathcal{Z}^*$.
- σ^2: denota o distanciamento médio entre soluções não-dominadas das fronteiras.
- σ^3: denota a razão entre a área delimitada pela fronteira $\bar{\mathcal{Z}}$ com relação à área delimitada por $\mathcal{Z}^*$. Para isto, utilizou-se como ponto de referência para cálculo dessas áreas o vetor nadir.
- σ^{4^*} e $\bar{\sigma^4}$: denotam as cardinalidades dos conjuntos $\mathcal{Z}^*$ e $\bar{\mathcal{Z}}$, respectivamente.

Além disso, t^* é o tempo transcorrido pelo $\varepsilon-$Restrito para obter $\mathcal{Z}^*$ e $\bar{t}$ é o tempo do mesmo algoritmo para obter $\bar{\mathcal{Z}}$ (em segundos).

Cada resultado na Tabela (2) representa o valor médio destas métricas dentre as 20 simulações para cada classe. Em negrito destaca-se as médias para cada dimensão m.

Pelos resultados iniciais apresentados nesta tabela, observa-se uma grande porcentagem de soluções da fronteira $\bar{\mathcal{Z}}$ coincidindo com as soluções da fronteira $\mathcal{Z}^*$, isto é, para $m = 10$, 20 e 40, tem-se, respectivamente, 96%, 91% e 84% dos pontos coincidentes, mostrando que a heurística de arredondamento determina soluções inteiras de ótima qualidade.

A segunda coluna da mesma tabela indica o distanciamento médio (na norma Euclidiana) entre essas fronteiras no espaço objetivo. Em média, esta distância foi gradativamente aumentando à medida que a dimensão das instâncias resolvidas cresceu. Para $m = 10$ tem-se um distanciamento de apenas $0,02$; para $m = 20$ passa-se para $0,07$ e, finalmente, para $m = 40$, uma distância igual a $0,16$, valores que podem ser considerado muito baixos.

A métrica σ^3 também comprova que a heurística de arredondamento de soluções forneceu ótimos resultados nesse problema. Comparando-se a razão das áreas delimitadas por estas fronteiras, nota-se que a fronteira heurística ocupa quase 99% da área da fronteira exata. Com respeito ao número de soluções encontradas, o arredondamento e refinamento pode perder algumas soluções eficientes. No entanto, em média, apenas $2,1\%$ das soluções são suprimidas por ele.

Com relação ao $\varepsilon-$Restrito, pode-se afirmar que é um método de escalarização simples e eficaz, capaz de gerar todas as soluções eficientes para o PCUIM. Para as instâncias consideradas, o tempo médio de execução ficou abaixo dos 2 segundos para obtenção de $9,48$ soluções para $\mathcal{Z}^*$; no entanto para exemplares com maior número de itens esse tempo de execução pode ficar muito

[1] No espaço objetivo.

Tabela 2: Resultados computacionais médios do $\varepsilon-$Restrito no PCUIM

Classe	σ^1	σ^2	σ^3	σ^{4*}	$\bar{\sigma^4}$	t^*	$\bar{t}$
1	0,06	0,07	1,000	6,10	6,10	0,22	0,18
2	0,02	0,02	1,000	6,20	6,10	0,31	0,21
3	0,01	0,01	0,810	6,35	6,00	0,33	0,24
4	0,00	0,00	0,667	3,10	3,10	0,12	0,10
5	0,04	0,06	1,000	4,30	4,25	0,14	0,12
6	0,04	0,04	1,000	4,05	4,00	0,16	0,13
7	0,00	0,00	1,000	2,65	2,65	0,09	0,10
8	0,00	0,00	1,000	3,75	3,65	0,12	0,11
9	0,00	0,00	1,000	4,00	3,80	0,12	0,11
Média	**0,04**	**0,02**	**0,942**	**4,50**	**4,41**	**0,18**	**0,14**
10	0,05	0,05	0,992	10,70	10,60	0,92	0,91
11	0,09	0,09	1,000	10,30	10,30	1,04	0,65
12	0,07	0,08	0,986	10,70	10,30	1,13	0,85
13	0,06	0,06	1,000	6,70	6,60	0,32	0,24
14	0,03	0,03	0,979	8,05	7,75	0,55	0,33
15	0,06	0,08	0,976	7,70	7,65	0,58	0,40
16	0,00	0,00	1,000	5,10	5,05	0,15	0,13
17	0,07	0,07	1,000	7,75	7,70	0,34	0,26
18	0,13	0,13	0,958	6,80	6,55	0,33	0,24
Média	**0,09**	**0,07**	**0,984**	**8,21**	**8,06**	**0,59**	**0,45**
19	0,21	0,25	0,983	18,30	17,95	6,46	5,59
20	0,15	0,17	0,984	18,80	18,25	7,25	5,83
21	0,20	0,22	0,981	17,95	17,15	9,24	6,43
22	0,15	0,15	0,994	14,80	14,50	4,50	3,00
23	0,17	0,20	0,993	16,85	16,50	6,28	3,82
24	0,18	0,21	0,981	15,70	15,20	7,73	4,43
25	0,04	0,04	1,000	11,75	11,75	0,91	1,16
26	0,03	0,03	0,994	11,95	11,90	0,76	0,90
27	0,17	0,19	0,993	15,40	15,15	3,18	2,49
Média	**0,16**	**0,16**	**0,986**	**15,72**	**15,37**	**5,14**	**3,74**
Média Geral	**0,08**	**0,08**	**0,988**	**9,48**	**9,28**	**1,97**	**1,44**

mais prolongado. O uso da relaxação e posteriormente o arredondamento foi vantajoso em termos do tempo de execução. Essa diferença ficou pouco perceptível para $m = 10$ e $m = 20$, no entanto, para $m = 40$ o uso desta abordagem utilizou apenas 61% do tempo gasto da técnica concorrente.

A Figura (2) dá uma ilustração de três fronteiras de Pareto obtidas para um único problema teste com $m = 100$. Marcada por losangos está a fronteira relaxada $\tilde{\mathcal{Z}}$, em quadrados a fronteira $\mathcal{Z}^*$ e em triângulos a fronteira relaxada arredondada $\bar{\mathcal{Z}}$.

Frisa-se no entanto, que neste único exemplo, para encontrar a fronteira $\mathcal{Z}^*$ foram necessários 93 segundos, ao passo que para obter a fronteira $\tilde{\mathcal{Z}}$ e fazer seu arredondamento foram necessários apenas 37 segundos, isto é, quase $1/3$ do tempo computacional. Essa evidência fica cada vez mais forte à medida que a instância do problema cresce.

6. Conclusões e Perspectivas Futuras

Este trabalho ilustrou a utilização de um modelo linear para o Problema de Corte Unidimensional Inteiro Multiobjetivo. Apresentou-se uma técnica de escalarização, capaz de garantir matematicamente que todas as soluções eficientes são captadas. O procedimento pode se tornar excessivamente caro, sob a ótica computacional, utilizando tal técnica. No entanto, através dos experimentos computacionais iniciais realizados, fazendo-se a relaxação das condições de integralidade das variáveis que contam o número de padrões de corte e, posteriormente, aplicando-se um procedimento de arredondamento nas soluções encontradas pelo método $\varepsilon-$Restrito, comprovou-se experimentalmente que há relativos ganhos em termos de tempo computacional sem que haja uma extensa depreciação das soluções encontradas.

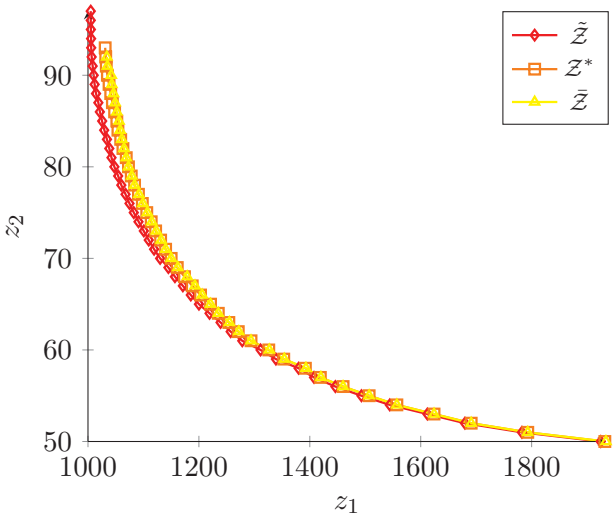

Figura 2: Ilustração das possíveis fronteiras de Pareto para um problema-teste com $m = 100$

Como direções de pesquisa, apontam-se outras técnicas de escalarização e respectivas comparações.

Agradecimentos

Os autores agradecem ao Instituto de Matemática, Estatística e Computação Científica na UNICAMP, ao Centro de Matemática e Aplicações Fundamentais e Investigação Operacional (CMAF-CIO) e ao ISEG da Universidade de Lisboa pelo suporte técnico e à FAPESP, processo 2013/06035-0 e 2014/22665-7 e à FCT (Portugal), projeto UID/MAT/04561/2013 pelo financiamento desta pesquisa.

Referências

[1] **Alves, M. J. and Clímaco, J. C. N.** An Interactive Reference Point Approach for Multiobjective Mixed-Integer Programming Using Branch-and-Bound. *European Journal of Operational Research*, 124(3):478–494, 2000.

[2] **Coffman,E. and Garey, M. and Johnson, D.** *Approximation Algorithms for Bin Packing: A Survey.* Boston. PWS, 1996. Approximation algorithms for NP-hard problems.

[3] **Cohon, J. L.** *Multiobjective Programming and Planning*. Academic Press, California, 1978.

[4] **Ehrgott, M.** *Multicriteria optimization*. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, 2005.

[5] **Ehrgott, M. and Ruzika, S.** Improved $\varepsilon-$Constraint Method for Multiobjective Programming. *Journal of Optimization Theory and Applications*, 138:375–396, 2008.

[6] **Gau, T. and Wascher, G.** CUTGEN: A Problem Generator for the Standard One-Dimensional Cutting Stock Problem. *European Journal of Operational Research*, 84(3):572 – 579, 1995. Cutting and Packing.

[7] **Gilmore, P. C. and Gomory, R. E.** A Linear Programming Approach to the Cutting-Stock Problem. *Operations Research*, 9:848–859, 1961.

[8] **Glover, F.** Heuristics for Integer Programming Using Surrogate Constraints. *Decision Sciences*, 8 (1):156–166, 1977.

[9] **Golfeto, R. R. and Moretti, A. C. and Sales, L. L. N.** A Genetic Symbiotic Algorithm Applied to the One-dimensional Cutting Stock Problem. *Pesquisa Operacional (impresso)*, 9:365–382, 2009.

[10] **Kolen, A. W. J. and Spieksma, F.C.R.** Solving a Bi-criterion Cuttins Stock Problem with Open-ended Demand: A Case Study. *European Journal of Operational Research*, 51:1238–1247, 2000.

[11] **Leduíno, L. S. N.** *Modelo Não Linear para Minimizar o Número de Objetos Processados e o Setup num Problema de Corte Unidimensional*. Tese de doutorado, Universidade Estadual de Campinas, IMECC - Campinas, 2005.

[12] **Miettinen, K.** *Nonlinear Multiobjective Optimization*. International Series in Operations Research & Management Science. Springer US, 1999. ISBN 9780792382782. LCCN 98037888.

[13] **Poldi, K. C. and Arenales, M. N.** Heuristics for the One-dimensional Cutting Stock Problem with Limited Multiple Stock Lengths. *Computers and Operations Research*, 36:2074 – 2081, 2009.

[14] **Solanki, R.** Generating the Noninferior Set in Mixed Integer Biobjective Linear Programs: An Application to a Location Problem. *Computers & OR*, 18(1):1–15, 1991.

[15] **Sylva, J. and Crema, A.** A Method for Finding the Set of Non-dominated Vectors for Multiple Objective Integer Linear Programs. *European Journal of Operational Research*, 158:46–55, 2004.

[16] **Sylva, J. and Crema, A.** A Method for Finding Well-dispersed Subsets of Non-dominated Vectors for Multiple Mixed Integer Linear Programs. *European Journal of Operational Research*, 180:1011–1027, 2007.

[17] **Vanderbeck, F.** An Exact Algorithm for IP Column Generation. *Operations Research Letters*, 19: 151–159, 1996.