Estudo de Malhas utilizando Atualização de Itens para Problemas de Empacotamento Bidimensional

Jéssica Gabriela de Almeida Cunha e Thiago Alves de Queiroz
Unidade de Matemática e Tecnologia - UFG/Regional Catalão,
Av. Dr. Lamartine P. Avelar, 1120, Setor Universitário, 75704-020, Catalão-GO.
jessicagabriela.1201@gmail.com
taq@ufg.br

RESUMO

Neste trabalho é investigado o procedimento de atualização do tamanho dos itens juntamente com o uso de diferentes estratégias de discretização de malhas para empacotar itens. Considera-se os seguintes tipos de discretização: *canonical dissections*, *reduced raster points*, *regular normal patterns* e *meet-in-the-middle patterns*. Então, busca-se verificar se existe equivalência entre as malhas antes e depois de aplicar um procedimento de atualização do tamanho do itens. Para tanto, experimentos computacionais foram realizados em instâncias da literatura de médio e grande porte, evidenciando que uma grande redução ocorreu ao usar a malha *canonical dissections*, ao passo que a malha de *reduced raster points* possui a menor quantidade de pontos no geral. Além disso, a aplicação do procedimento de atualização dos itens produziu bons resultados, pois a quantidade de pontos em todas as malhas foi reduzida satisfatoriamente.

PALAVRAS CHAVE. Atualização do tamanho dos itens. Malha de pontos. Problemas de empacotamento.

OC - Otimização Combinatória; PM - Programação Matemática

ABSTRACT

In this work, we investigate a procedure to update the item sizes along with the use of different strategies of grid discretization in order to pack items. We consider the following types of grid discretization: *canonical dissections*, *reduced raster points*, *regular normal patterns*, and *meet-in-the-middle patterns*. Therefore, we aim to verify whether there exists any equivalence among the grids before and after to apply the procedure to update items size. For that, computational experiments have been done over medium and large-sized instances from the literature, showing that a large reduction happened for the grid of canonical dissections, although the grid of reduced raster points has in general the smallest number of points. Besides that, the use of the procedure to update item sizes has given good results because the number of points in all of the grids has been reduced satisfactorily.

KEYWORDS. Update item size. Grid of points. Packing problems.

OC - Combinatorial Optimization; PM - Mathematical Programming

1. Introdução

As empresas de logística em geral buscam organizar da melhor maneira possível as mercadorias que devem ser armazenadas e carregadas nos caminhões ou contêineres. De um modo geral, problemas como esse objetivam empacotar um conjunto de itens dentro de um ou mais recipientes a fim de evitar desperdícios na ocupação sempre que possível. O empacotamento exige que os itens não se sobreponham e que respeitem as dimensões dos recipientes. Problemas de empacotamento, em geral, são classificados como NP-Difíceis [Garey e Johnson, 1979]. Essa classificação significa que, para esses problemas, não se espera que exista algum método exato que os resolva na otimalidade em tempo polinomial no tamanho da entrada, em particular, polinomial no tamanho do conjunto de itens disponíveis para o empacotamento.

Existem soluções de problemas de empacotamento que contêm uma pequena área do recipiente vazia, visto que não cabe o empacotamento de qualquer item disponível. Segundo Boschetti et al. [2002] e Boschetti e Mingozzi [2003], um dos itens pertencentes a solução pode, então, ter o seu tamanho atualizado a fim de preencher esse espaço vago. Assim, obtém-se um padrão de empacotamento equivalente ao anterior, porém com um espaço ocupado maior. A atualização do tamanho dos itens pode ser feita, sem perda de generalidade, conforme proposto por Boschetti e Mingozzi [2003].

Uma forma de realizar o empacotamento é associar os itens a pontos de uma malha discretizada sobre o recipiente. Essa malha pode ser discretizada por diversas estratégias formuladas na literatura, a saber: *canonical dissections* [Herz, 1972], *useful numbers* [Carnieri et al., 1994], *reduced raster points* [Scheithauer e Terno, 1996], *corner points* [Martello et al., 2000], *regular normal patterns* [Boschetti et al., 2002], *extreme points* [Crainic et al., 2008] e *meet-in-the-middle patterns* [Côté e Iori, 2016]. De acordo com o tipo de malha utilizada, tem-se um maior número de posições possíveis para realizar o empacotamento, o que pode tornar a busca por uma solução ótima do problema mais lenta. Neste trabalho não serão investigadas as malhas *useful numbers*, pela sua equivalência com a *canonical dissections*, bem como as malhas *corner points* e *extreme points*, uma vez que os seus pontos são gerados por um processo enumerativo que já empacota os itens.

O uso de malhas na resolução de problemas de empacotamento pode ser encontrado em diversos trabalhos da literatura. Para o problema de carregamento de paletes, o qual busca realizar o empacotamento do maior número possível de itens dentro de um recipiente, utilizaram-se as malhas *canonical dissections* e *reduced raster points* por Alvarez-Valdés et al. [2005] e Birgin et al. [2010]. A malha *corner points* foi utilizada por de Almeida e Figueiredo [2010], na resolução do problema de carregamento de contêineres, que visa minimizar a quantidade de contêineres utilizados no empacotamento de todas as caixas disponíveis. Para uma variante desse problema, Junqueira et al. [2010] e Junqueira et al. [2012] utilizaram a malha *canonical dissections*.

O problema da mochila ilimitada, no qual os itens podem ter ilimitadas cópias no empacotamento, em sua versão bidimensional, foi resolvido utilizando a malha *reduced raster points* por Del Valle et al. [2012] e Gonçalves e Queiroz [2014]. Para o problema em sua versão tridimensional, utilizando essa mesma malha, tem-se o estudo de Queiroz et al. [2012]. Por meio dos experimentos computacionais, os autores observaram que a malha utilizada não trouxe perda de solução ótima sobre as instâncias testadas.

A malha *reduced raster points* também foi aplicada por Queiroz et al. [2015] na resolução do problema da mochila ilimitada com cortes não guilhotinados. Para esse problema, utilizando a abordagem L desenvolvida por Lins et al. [2003], os autores provaram que a malha não perde a solução ótima. Nascimento et al. [2016] utilizaram as malhas *canonical dissections* e *reduced raster points* na resolução do problema de empacotamento ortogonal em suas versões bi- e tridimensional. Por meio de experimentos computacionais utilizando instâncias da literatura, os autores observaram que o problema é resolvido com tempo computacional bem distinto conforme o tipo de malha usada. A malha que obteve a solução em menor tempo foi a *reduced raster points* e ainda sem perder a solução ótima.

Todo o desenvolvimento das malhas e a atualização do tamanho dos itens realizados neste trabalho foram feitos baseados em problemas de empacotamento bidimensional. Todavia, esse estudo pode ser estendido para problemas tridimensionais ao se considerar a altura (terceira dimensão) dos itens e do recipiente. O objetivo deste trabalho é observar a quantidade de pontos que cada malha gera antes e depois de aplicar um procedimento de atualização do tamanho dos itens. Assim, busca-se analisar se há equivalência entre algum tipo de malha antes e depois de realizar tal procedimento.

A descrição geral de problemas de empacotamento é apresentada na Seção 2 contando ainda com um algoritmo de atualização de tamanho dos itens seguindo a proposta de Boschetti e Mingozzi [2003]. As malhas utilizadas nesse trabalho são definidas na Seção 3, de forma que o uso dessas malhas com o procedimento de atualização de tamanho dos itens é também exemplificado. Na Seção 4 os experimentos computacionais são realizados e os resultados discutidos. E, por fim, as conclusões e as perspectivas futuras são apresentadas na Seção 5.

2. Problemas de Empacotamento Bidimensionais

Seja um ou mais recipientes retangulares, cada um com largura L e comprimento C, e um conjunto $I = \{1, 2, \cdots, n\}$ de n itens retangulares, cada item i com largura l_i, comprimento c_i e valor associado v_i. Considera-se que as dimensões dos itens e do recipiente são valores inteiros positivos, sem perda de generalidade, pois basta fazer uma mudança de escala quando apropriado. Em certos casos, o valor v_i de cada item $i \in I$ pode ser dado pela sua área, isto é, $v_i = l_i c_i$.

Conforme Wäscher et al. [2007], certos problemas de empacotamento possuem como objetivo maximizar o valor total empacotado ou minimizar o desperdício, quando se empacota apenas um subconjunto de itens e usa um único recipiente. No caso de vários recipientes é comum empacotar todos os itens usando a menor quantidade possível de recipientes. Em outros casos, o recipiente tem uma dimensão aberta, tal que se quer empacotar todos os itens buscando minimizar o tamanho da dimensão aberta. O empacotamento é realizado de forma que os itens não se sobreponham e respeitem as dimensões do recipiente.

Uma forma de resolver problemas de empacotamento é fazendo uma discretização do recipiente em uma malha de pontos. O empacotamento dos itens é realizado associando um de seus cantos a algum dos pontos pertencentes a malha. Ao usar um modelo de programação linear inteira, a variável de decisão para o problema é binária, de modo que, se o item i for empacotado com seu canto inferior esquerdo no ponto $k = (p, q)$ da malha, tem-se que ela assume valor 1, caso contrário, assume valor 0.

O ponto p pertence ao conjunto X de coordenadas referente a largura, e q ao conjunto Y de coordenadas referente ao comprimento. A quantidade de elementos nos conjuntos X e Y varia de acordo com o tipo de discretização utilizada para a resolução do problema. Com isso, pode-se observar que quanto mais coordenadas houverem nesses conjuntos, mais pontos há de serem testados para o empacotamento dos itens, o que pode causar um maior gasto de tempo na busca por uma solução ótima.

Como o empacotamento não pode ter itens se sobrepondo, então um item não pode ser empacotado em qualquer ponto que já esteja ocupado/sobreposto por outro item. Com o intuito de evitar que isso aconteça, restringe-se o empacotamento, de modo que apenas um item i possa ser empacotado no ponto $k = (p, q)$ de formar a cobrir qualquer ponto (s, t) da malha, em que $p \in [0, L]$, tal que $s - l_i + 1 \leqslant p \leqslant s$, e $q \in [0, C]$, tal que $t - c_i + 1 \leqslant q \leqslant t$.

Seja P um padrão de empacotamento que contenha o conjunto $I' \subseteq I$ de itens empacotados no recipiente. Segundo Boschetti et al. [2002] e Boschetti e Mingozzi [2003], tem-se que cada item $i' \in I'$ pode ter sua largura atualizada por $l_{i'} = l_{i'} + (L - L^*_{i'})$ e, de modo análogo, seu comprimento por $c_{i'} = c_{i'} + (C - C^*_{i'})$. Essa atualização do tamanho dos itens gera um novo padrão de empacotamento P' com o mesmo conjunto I', porém com área ocupada possivelmente maior. Os valores $L^*_{i'}$ e $C^*_{i'}$ são as soluções ótimas encontradas pela resolução do problema da mochila 0-1 conforme dados nas eqs. (1) e (2), respectivamente.

$$L^*_{i'} = max \left\{ l = \sum_{k \in I' \backslash \{i'\}} \varepsilon_k \, l_k + l_{i'} \;\middle|\; l \leqslant L,\ \varepsilon_k \in \{0,1\},\ k \in I' \backslash \{i'\} \right\}, \tag{1}$$

$$C^*_{i'} = max \left\{ c = \sum_{k \in I' \backslash \{i'\}} \varepsilon_k \, c_k + c_{i'} \;\middle|\; c \leqslant C,\ \varepsilon_k \in \{0,1\},\ k \in I' \backslash \{i'\} \right\}. \tag{2}$$

Para atualizar o tamanho dos itens, utiliza-se o Algoritmo 1, que considera um conjunto no qual os itens são ordenados de forma decrescente de acordo com as dimensões (primeiro com relação a largura quando atualizando as larguras dos itens e, depois, com relação ao comprimento quando atualizando o comprimento dos itens). Assim, o primeiro item na ordem dada é selecionado e, a partir dele, resolve-se o problema da mochila 0-1 visando obter a máxima largura (comprimento) que se poder conseguir caso esse primeiro item já estivesse empacotado. O espaço vazio que sobra entre a solução e o tamanho do recipiente é adicionado ao tamanho desse primeiro item sem perda de generalidade. Após isso, o item que teve seu tamanho aumentado e os outros itens que estavam na solução retornada não podem ter os seus tamanhos atualizados. Esse procedimento é repetido enquanto houver itens ainda não usados em alguma solução, sendo feito individualmente para cada dimensão, primeiro ao longo da largura dos itens, depois ao longo do comprimento.

Algoritmo 1: Atualização do tamanho dos itens.

Passo 1. (Ordenação pela largura dos itens)
Seja $J = I$ o conjunto no qual os itens são ordenados de forma decrescente com relação a largura ($l_1 \geqslant l_2 \geqslant \cdots \geqslant l_n$).

Passo 2. (Atualização ao longo da largura)
Considere o primeiro item $j \in J$ na ordem dada.
Seja ε^* a solução ótima do problema da mochila em (1) para $i' = j$ e $I' = J$.
Atualize $l_j = l_j + (L - L^*_j)$.
Faça $J = J \backslash (\{j\} \cup \{k \in J \backslash \{j\} | \varepsilon^*_k = 1\})$.
Se $J \neq \emptyset$, então volte ao Passo 2.

Passo 4. (Ordenação pelo comprimento dos itens)
Seja $J = I$ o conjunto no qual os itens são ordenados de forma decrescente com relação ao comprimento ($c_1 \geqslant c_2 \geqslant \cdots \geqslant c_n$).

Passo 5. (Atualização ao longo do comprimento)
Considere o primeiro item $j \in J$.
Seja ε^* a solução ótima do problema da mochila em (2) para $i' = j$ e $I' = J$.
Atualize $c_j = c_j + (C - C^*_j)$.
Faça $J = J \backslash (\{j\} \cup \{k \in J \backslash \{j\} | \varepsilon^*_k = 1\})$.
Se $J \neq \emptyset$, então volte ao Passo 5.

Para resolver o problema da mochila 0-1, utiliza-se a programação dinâmica apresentada no Algoritmo 2. Os itens que são alocados dentro do recipiente são representados na matriz binária D. O valor da solução ótima obtida para o problema da mochila é dado por $F(n, L)$, neste caso ao longo da largura, tal que para um item $j \in J$ escolhido para atualizar a largura, tem-se $L^*_j = F(n, L)$. A resolução do problema da mochila ao longo do comprimento é feito de modo análogo trocando L por C e l_j por c_j. Além disso, é importante destacar que o algoritmo recebe o conjunto $J = J \backslash \{j\}$ e, assim, n reflete a nova quantidade de itens em J, e L e C a área restante após o empacotamento do item j.

Algoritmo 2: Programação Dinâmica para a Mochila 0-1.

para $j = 0$ **até** L **faça**
  $F(0, j) = 0$;
para $i = 1$ **até** n **faça**
  para $j = 0$ **até** L **faça**
    se $((l_i \leqslant j)$ ***E*** $(l_i + F(i-1, j-l_i) > F(i-1, j)))$ **então**
      $F(i, j) = l_i + F(i-1, j-l_i)$;
      $D(i, j) = 1$;
    senão
      $F(i, j) = F(i-1, j)$;
      $D(i, j) = 0$;
para $i = n$ **até** 1 **faça**
  se $D(i, j) == 1$ **então**
    Mostre i;
    $j = j - l_i$;
Retorne $F(n, L)$;

3. Discretizações da Malha de Pontos

O recipiente pode ser discretizado em uma malha de pontos usando os diferentes métodos descritos pela literatura, sendo que alguns geram malhas semelhantes, enquanto outros produzem malhas mais refinadas, com menos pontos. No geral, as malhas são calculadas de acordo com as dimensões dos itens e do recipiente, e não ultrapassam as coordenadas $L - \min_{1 \leq i \leq n}\{l_i\}$ e $C - \min_{1 \leq i \leq n}\{c_i\}$ referentes, respectivamente, a largura e ao comprimento do recipiente. É perceptível que as coordenadas posteriores a essas são inviáveis, não sendo possível empacotar qualquer item que caiba inteiramente dentro do recipiente.

As malhas são obtidas fazendo o produto cartesiano entre os conjuntos de coordenadas obtidos ao longo da largura e do comprimento do recipiente. A malha mais comum possui discretização unitária, uma vez que os itens e o recipiente possuem dimensões inteiras, e pode ser adquirida por meio das eqs. (3) e (4) referentes, respectivamente, a largura e ao comprimento.

$$X_c = \{p \in \mathbb{Z} \mid 0 \leqslant p \leqslant L - \min_{1 \leq i \leq n}\{l_i\}\}; \tag{3}$$

$$Y_c = \{q \in \mathbb{Z} \mid 0 \leqslant q \leqslant C - \min_{1 \leq i \leq n}\{c_i\}\}. \tag{4}$$

Observa-se que à medida que a quantidade de pontos na malha aumenta, estima-se que mais tempo e esforço computacional é preciso na busca por uma solução do problema. É nesse sentido que foram desenvolvidas outras formas para calcular malhas mais refinadas que a malha unitária.

3.1. *Canonical Dissections*

Um dos primeiros métodos de discretização é conhecido por *canonical dissections* de Herz [1972], também sendo referenciado como *normal patterns* por Christofides e Whitlock [1977]. Esses autores mostraram que é válido alocar os itens sempre mais abaixo e a esquerda até encostar em outro item ou nos lados do recipiente. Deste modo, a malha *canonical dissections* é basicamente constituída pelas combinações cônicas das dimensões dos itens disponíveis para o empacotamento. A eq. (5) calcula as coordenadas ao longo da largura, ao passo que a eq. (6) está relacionada com as coordenadas ao longo do comprimento.

$$X_d = \{p \in \mathbb{Z} \mid p = \sum_{i=1}^{n} \varepsilon_i \, l_i, \; 0 \leqslant p \leqslant L - \min_{1 \leq i \leq n} \{l_i\}, \; \varepsilon_i \in \{0,1\}, \; i = 1, \ldots, n\}; \tag{5}$$

$$Y_d = \{q \in \mathbb{Z} \mid q = \sum_{i=1}^{n} \varepsilon_i \, c_i, \; 0 \leqslant q \leqslant C - \min_{1 \leq i \leq n} \{c_i\}, \; \varepsilon_i \in \{0,1\}, \; i = 1, \ldots, n\}. \tag{6}$$

Herz [1972] e Christofides e Whitlock [1977] afirmaram que esse tipo de discretização da malha exclui as posições redundantes presentes em uma malha unitária. Com isso, há um refinamento na malha unitária, diminuindo a quantidade de pontos, sem, para tanto, perder a solução ótima do problema.

3.2. *Reduced Raster Points*

Com o propósito de refinar a quantidade de pontos na malha *canonical dissections*, Scheithauer e Terno [1996] definiram a discretização conhecida por *reduced raster points*. A malha *reduced raster points* é um subconjunto da *canonical dissections*, no entanto, ainda não foi provado pela literatura se o seu uso garante que a solução ótima seja preservada. As eqs. (7) e (8) referem-se, respectivamente, a obtenção dos conjunto de coordenadas ao longo da largura e do comprimento do recipiente.

$$X_r = \{(L - p)_x \mid \forall p \in X_d\} \cup \{0\}, \text{ sendo } (L-p)_x = \max\{s \in X_d \mid s \leqslant L - p\}; \tag{7}$$

$$Y_r = \{(C - q)_y \mid \forall q \in Y_d\} \cup \{0\}, \text{ sendo } (C-q)_y = \max\{t \in Y_d \mid t \leqslant C - q\}. \tag{8}$$

3.3. *Regular Normal Patterns*

Boschetti et al. [2002] desenvolveram a malha *regular normal patterns* fundamentada no princípio da *canonical dissections*. A malha *regular normal patterns* é computada individualmente para cada item $i \in I$ ao longo da largura e do comprimento, respectivamente, por meio das eqs. (9) e (10).

$$X_{b:i} = \{p \in \mathbb{Z} \mid p = \sum_{j \in I \setminus \{i\}} \varepsilon_j \, l_j, \; 0 \leqslant p \leqslant L - l_i, \; \varepsilon_j \in \{0,1\}, \; j \in I \setminus \{i\}\}; \tag{9}$$

$$Y_{b:i} = \{q \in \mathbb{Z} \mid q = \sum_{j \in I \setminus \{i\}} \varepsilon_j \, c_j, \; 0 \leqslant q \leqslant C - c_i, \; \varepsilon_j \in \{0,1\}, \; j \in I \setminus \{i\}\}. \tag{10}$$

A malha geral envolvendo todos itens é obtida, ao longo de cada dimensão, fazendo $X_b = \cup_{i \in I} X_{b:i}$ e $Y_b = \cup_{i \in I} Y_{b:i}$, eliminando as possíveis coordenadas redundantes que surgem em cada conjunto.

3.4. *Meet-in-the-Middle Patterns*

Um método de discretização recente foi apresentado por Côté e Iori [2016] e conceituado como *meet-in-the-middle patterns* (MIM). Essa malha de pontos é gerada para cada item $i \in I$ e um dado fator $t_x \in \{1, 2, \ldots, L\}$, referente a largura, e $t_y \in \{1, 2, \ldots, C\}$, referente ao comprimento.

O princípio dessa malha é basicamente unir os conjuntos de coordenadas obtidas partindo da esquerda do recipiente e depois da direita. Desta forma, $X_{m:i,t_x} = Ex_{i,t_x} \cup Dx_{i,t_x}$ é o conjunto de coordenadas, ao longo da largura, para empacotar o item i dado um fator t_x.

$$Ex_{i,t_x} = \{p \in \mathbb{Z} \mid p = \sum_{j \in I \setminus \{i\}} \varepsilon_j \, l_j, \; 0 \leqslant p \leqslant \min\{t_x - 1, L - l_i\}, \; \varepsilon_j \in \{0,1\}, \; j \in I \setminus \{i\}\}; \tag{11}$$

$$Dx_{i,t_x} = \{p' \in \mathbb{Z} \mid p' = L - l_i - p,\ p = \sum_{j \in I \setminus \{i\}} \varepsilon_j\, l_j,\ 0 \leqslant p \leqslant L - l_i - t_x, \\ \varepsilon_j \in \{0,1\},\ j \in I \setminus \{i\}\}. \tag{12}$$

O conjunto das coordenadas, para todo item $i \in I$, ao longo da largura, para um dado t_x é definido por $X_{m:t_x} = \cup_{i \in I} X_{m:i,t_x}$. A obtenção do conjunto final é feita assumindo todo $t_x \in \{1, 2, \ldots, L\}$, de modo que a malha MIM mínima, ao longo da largura, é determinada na eq. (13).

$$X_m = \left\{ X_{m:t_x} \;\middle|\; t_x = \text{argmin}_{s \in \{1,2,\ldots,L\}} \left\{ \sum_{i \in I} |X_{m:i,s}| \right\} \right\}. \tag{13}$$

O desenvolvimento é realizado, de modo análogo, para o cálculo da malha MIM mínima referente ao comprimento, substituindo L por C e l_i por c_i. Côté e Iori [2016] mostraram que o uso da malha MIM não gera a perda da solução ótima, assim como as malhas *canonical dissections* e *regular normal patterns*. O algoritmo implementado para calcular a malha MIM consiste no Algoritmo 3 de Côté e Iori [2016].

3.5. Exemplo das Malhas envolvendo a Atualização dos Itens

Para ilustrar as malhas de pontos e o procedimento de atualização do tamanho dos itens, considera-se um exemplo de um problema de empacotamento bidimensional. Esse exemplo considera quatro itens distintos ($n = 4$), e um recipiente de largura $L = 12$ e comprimento $C = 10$. Cada item i possui dimensões (l_i, c_i) dadas por: $(2, 3)$, $(4, 5)$, $(6, 3)$ e $(9, 7)$; e, valor v_i sendo igual a área. A partir disso, tem-se os conjuntos de coordenadas ao longo da largura, do comprimento e a quantidade total de pontos na malha.

- Malha unitária: $X_c = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ e $Y_c = \{0, 1, 2, 3, 4, 5, 6, 7\}$ - 88 pontos;
- *Canonical dissections*: $X_d = \{0, 2, 4, 6, 8, 9, 10\}$ e $Y_d = \{0, 3, 5, 6, 7\}$ - 35 pontos;
- *Reduced raster points*: $X_r = \{0, 2, 4, 6, 8, 10\}$ e $Y_r = \{0, 3, 5, 7\}$ - 24 pontos;
- *Regular normal patterns*: $X_b = \{0, 2, 4, 6, 8, 9, 10\}$ e $Y_b = \{0, 3, 5, 7\}$ - 28 pontos;
- *Meet-in-the-middle patterns*: $X_m = \{0, 2, 3, 4, 6, 8, 10\}$ e $Y_m = \{0, 3, 4, 5, 7\}$ - 35 pontos.

Nota-se que a malha *reduced raster points* é a que possui a menor quantidade de pontos, reduzindo a malha unitária em aproximadamente 73% e a malha *canonical dissections* em cerca de 31%. Para a aplicação do procedimento de atualização do tamanho dos itens, os vetores de largura e comprimento dos itens devem ser ordenados, cada um independentemente, de forma decrescente. Assim, tem-se $L = 12$, $C = 10$, $l = (9, 6, 4, 2)$ e $c = (7, 5, 3, 3)$.

Primeiro, realiza-se o procedimento de atualização ao longo da largura. Seleciona-se o primeiro item ($l = 9$), empacotando-o dentro do recipiente de forma que sobram 3 unidades de largura. Ao aplicar o algoritmo de programação dinâmica, tem-se o item de largura 2 como solução. Deste modo, restará 1 unidade de largura que não suporta o empacotamento de qualquer item. Segue que o primeiro item tem sua largura atualizada para 10 e o item de largura 2 não pode ter seu tamanho atualizado por ter feito parte de uma solução.

O próximo item a ser analisado é o de largura 6 que, quando empacotado, gera 6 unidades de largura restante. Ao aplicar a programação dinâmica, tem-se o item de largura 4 e de largura 2 como parte da solução e, assim, toda largura do recipiente está ocupada, não havendo como

aumentar o item de largura 6. Segue que todos os itens foram avaliados ou usados e, então, o procedimento finaliza para a largura.

O procedimento de atualização é feito agora ao longo do comprimento para o conjunto c. Seleciona-se o primeiro item com $c = 7$, empacotando-o dentro do recipiente e sobrando 3 unidades de comprimento. Ao aplicar a programação dinâmica, tem-se o item de comprimento 3 como solução e que preenche todo o recipiente. Segue que os itens de comprimento 7 e 3 não podem ter seus tamanhos aumentados.

O próximo item a ser testado é o de comprimento 5 que, ao ser empacotado, gera 5 unidades de comprimento livre. A aplicação da programação dinâmica fornece como solução o item de comprimento 3, o que resulta em 2 unidades de comprimento ainda restante. Com isso, o item de comprimento 5 passa a ter comprimento 7 e o outro item de comprimento 3 não pode ter seu tamanho atualizado. O procedimento finaliza, pois todos os itens foram avaliados ou usados.

Após aplicar o procedimento de atualização do tamanho dos itens, tem-se as novas dimensões para o conjunto de itens, a saber: $(2, 3)$, $(4, 7)$, $(6, 3)$ e $(10, 7)$. Obtendo as malhas sobre essas novas dimensões, chega-se em:

- Malha unitária: $X_c = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ e $Y_c = \{0, 1, 2, 3, 4, 5, 6, 7\}$ - 88 pontos;
- *Canonical dissections*: $X_d = \{0, 2, 4, 6, 8, 10\}$ e $Y_d = \{0, 3, 6, 7\}$ - 24 pontos;
- *Reduced raster points*: $X_r = \{0, 2, 4, 6, 8, 10\}$ e $Y_r = \{0, 3, 7\}$ - 18 pontos;
- *Regular normal patterns*: $X_b = \{0, 2, 4, 6, 8, 10\}$ e $Y_b = \{0, 3, 7\}$ - 18 pontos;
- *Meet-in-the-middle patterns*: $X_m = \{0, 2, 4, 6, 8, 10\}$ e $Y_m = \{0, 3, 4, 7\}$ - 24 pontos.

Percebe-se que a simples atualização do tamanho dos itens permitiu reduzir em cerca de 31% a malha *canonical dissections* e *meet-in-the-midde patterns*, 25% a *reduced raster points* e 36% a *regular normal patterns*. Deste modo, o procedimento de atualização do tamanho dos itens, além de aproveitar melhor o espaço do empacotamento, também ajuda a reduzir a quantidade de pontos nas malhas.

4. Experimentos Computacionais

A análise das malhas é feita por meio de experimentos computacionais realizados em um computador com processador Intel Core i3 2.40 GHz e 3 GB de memória RAM. Os métodos para a discretização do recipiente e o procedimento de atualização do tamanho dos itens foram implementados em linguagem de programação C++.

A realização dos experimentos utilizou 55 instâncias, de médio e grande porte, tratadas em Birgin et al. [2012] e com as seguintes características: apt1-apt20 com $n \in [31,\ 59]$, $L \in [1674,\ 2899]$ e $C \in [1612,\ 2994]$; b1-b7 com $n \in [30,\ 180]$ e $(L, C) = (4000,\ 2000)$; gcut1-gcut17 com $n \in [10,\ 82]$, $L \in [250,\ 3500]$ e $C \in [250,\ 3500]$; e, por fim, uu1-uu11 com $n \in [25,\ 60]$, $L \in [500,\ 3500]$ e $C \in [500,\ 3765]$. Em tais instâncias, cada tipo de item i possui apenas uma cópia a ser empacotada. O nome da instância e a quantidade de coordenadas nos conjuntos das diferentes malhas, antes e após a atualização do tamanho dos itens, são apresentados na Tabela 1, ao longo da largura, e na Tabela 2, ao longo do comprimento.

Por meio das Tabelas 1 e 2, nota-se que 21 instâncias tiveram a largura e 23 o comprimento aumentado de alguns de seus itens. A partir disso, observa-se que as malhas para as instâncias que tiveram os itens atualizados de tamanhos possuem menos pontos do que as geradas antes dessa atualização. As malhas tiveram as seguintes reduções percentuais médias após a atualização do tamanho dos itens: *canonical dissections* de $7,51\%$, com desvio padrão de $14,75\%$; *reduced raster points* de $0,41\%$, com desvio padrão de $6,68\%$; *regular normal patterns* de $7,50\%$, com desvio padrão de $14,71\%$; e, *meet-in-the-middle patterns* de $7,35\%$, com desvio padrão de $14,68\%$. Deste

modo, além do aumento dos itens permitir um melhor aproveitamento do espaço do recipiente, ele também refina a quantidade de pontos presentes nos diferentes tipos de malha.

Tabela 1: Resultado para as coordenadas ao longo da largura.

Instância	Antes da atualização do tamanho dos itens					Depois da atualização do tamanho dos itens					Tempo (s)
	Unitária	Canonical	Raster	Regular	MIM	Unitária	Canonical	Raster	Regular	MIM	
apt1	1993	1620	1351	1620	1351	1993	1620	1351	1620	1351	1,766880
apt2	2465	1981	1632	1981	1632	2465	1981	1632	1981	1632	3,183133
apt3	2523	1765	1146	1765	1159	2523	1765	1146	1765	1159	0,777952
apt4	1574	1302	1130	1302	1130	1574	1302	1130	1302	1130	1,380428
apt5	1981	1598	1324	1598	1324	1981	1598	1324	1598	1324	1,206282
apt6	2082	1567	1192	1567	1194	2082	1567	1192	1567	1194	1,327382
apt7	2710	2058	1595	2058	1596	2710	2058	1595	2058	1596	2,735629
apt8	2185	1827	1597	1827	1597	2185	1827	1597	1827	1597	3,094017
apt9	2620	1906	1347	1906	1352	2620	1906	1347	1906	1352	0,907709
apt10	2155	1516	1006	1516	1017	2155	1516	1006	1516	1017	0,597028
apt11	2674	2147	1786	2147	1786	2674	2147	1786	2147	1786	2,676376
apt12	2627	1929	1470	1929	1474	2627	1929	1470	1929	1474	2,053900
apt13	2545	2005	1631	2005	1631	2545	2005	1631	2005	1631	2,650598
apt14	1761	1356	1046	1356	1046	1761	1356	1046	1356	1046	1,188351
apt15	1940	1554	1298	1554	1298	1940	1554	1298	1554	1298	1,340998
apt16	2727	2118	1667	2118	1667	2727	2118	1667	2118	1667	1,696977
apt17	2231	1646	1189	1646	1193	2231	1646	1189	1646	1193	1,18365
apt18	1692	1355	1119	1355	1119	1692	1355	1119	1355	1119	1,139697
apt19	1898	1519	1262	1519	1262	1898	1519	1262	1519	1262	1,607867
apt20	1741	1391	1139	1391	1139	1741	1391	1139	1391	1139	1,729781
b1	3801	3225	2848	3225	2848	3801	3225	2848	3225	2848	4,492793
b2	3781	3078	2594	3078	2594	3781	3078	2594	3078	2594	3,874312
b3	3795	3132	2674	3132	2674	3795	3132	2674	3132	2674	3,957516
b4	3799	3149	2700	3149	2700	3799	3149	2700	3149	2700	4,018727
b5	3795	3086	2582	3086	2582	3795	3086	2582	3086	2582	3,833155
b6	3797	3069	2544	3069	2544	3797	3069	2544	3069	2544	3,695798
b7	3801	3454	3306	3454	3306	3801	3454	3306	3454	3306	154,12023
gcut1	185	22	12	22	22	185	16	11	16	16	0,000767
gcut2	185	34	14	33	34	180	16	13	16	16	0,001069
gcut3	188	77	43	77	77	188	77	43	77	77	0,004005
gcut4	189	84	44	84	84	189	83	45	83	83	0,007134
gcut5	369	15	9	15	15	369	7	7	7	7	0,000872
gcut6	369	32	11	30	32	369	18	11	17	18	0,001539
gcut7	372	62	22	62	62	372	45	25	45	45	0,003507
gcut8	370	94	43	94	94	370	88	43	88	88	0,005607
gcut9	709	25	14	25	25	709	18	13	18	18	0,001305
gcut10	732	27	11	21	27	732	14	11	11	14	0,003062
gcut11	735	62	17	62	62	716	25	17	25	25	0,003629
gcut12	747	147	46	146	147	747	137	47	136	137	0,019567
gcut13	2636	1386	542	1386	655	2636	1386	542	1386	655	0,485336
gcut14	3209	2364	1810	2364	1810	3209	2364	1810	2364	1810	3,237154
gcut15	3232	2407	1850	2407	1850	3232	2407	1850	2407	1850	4,759524
gcut16	3235	2551	2132	2551	2132	3235	2551	2132	2551	2132	8,360177
gcut17	3247	2667	2340	2667	2340	3247	2667	2340	2667	2340	16,537238
uu1	382	67	26	67	67	382	34	21	34	34	0,004099
uu2	598	166	54	165	128	598	163	54	163	127	0,008099
uu3	874	157	41	156	118	874	130	41	127	98	0,005482
uu4	785	204	58	204	127	785	199	58	199	119	0,014141
uu5	1152	379	104	379	241	1152	372	100	372	238	0,035121
uu6	1627	242	54	240	102	1627	166	54	164	81	0,011891
uu7	1165	428	133	428	213	1165	421	133	421	209	0,040037
uu8	1595	487	105	486	350	1595	468	105	465	333	0,049991
uu9	1950	467	113	466	371	1950	444	113	443	352	0,057553
uu10	2789	574	105	572	379	2789	532	105	528	345	0,064516
uu11	3464	2346	1282	2346	1394	3464	2346	1282	2346	1394	0,533788

Ademais, tem-se que a malha *canonical dissections* diminuiu a malha unitária, na média, em aproximadamente 46%, com desvio padrão de aproximadamente 30%. A *reduced raster points*

é a mais refinada entre as malhas em estudo, diminuindo a malha unitária, na média, em aproximadamente 61%, com desvio padrão 28%. Vale ressaltar que o tempo computacional gasto para gerar as malhas e realizar a atualização do tamanho dos itens, ao longo de cada dimensão, é, em geral, muito pequeno, sendo no pior caso de aproximadamente 154 segundos.

Tabela 2: Resultado para as coordenadas ao longo do comprimento.

Instância	Antes da atualização do tamanho dos itens					Depois da atualização do tamanho dos itens					Tempo (s)
	Unitária	Canonical	Raster	Regular	MIM	Unitária	Canonical	Raster	Regular	MIM	
apt1	1623	1374	1215	1374	1215	1623	1374	1215	1374	1215	1,483606
apt2	1532	1344	1236	1344	1236	1532	1344	1236	1344	1236	1,912325
apt3	1843	1234	723	1234	755	1843	1234	723	1234	755	0,316164
apt4	1986	1666	1450	1666	1450	1986	1666	1450	1666	1450	2,355100
apt5	1995	1476	1100	1476	1104	1995	1476	1100	1476	1104	0,790257
apt6	2582	2069	1700	2069	1700	2582	2069	1700	2069	1700	2,538203
apt7	2466	1905	1492	1905	1492	2466	1905	1492	1905	1492	2,350937
apt8	1865	1522	1276	1522	1276	1865	1522	1276	1522	1276	2,048296
apt9	1998	1297	703	1297	773	1998	1297	703	1297	773	0,262694
apt10	2840	2057	1428	2057	1435	2840	2057	1428	2057	1435	1,086712
apt11	2692	2045	1564	2045	1565	2692	2045	1564	2045	1565	2,132104
apt12	1685	1299	1012	1299	1013	1685	1299	1012	1299	1013	0,891663
apt13	1999	1597	1306	1597	1306	1999	1597	1306	1597	1306	1,767041
apt14	2490	1887	1430	1887	1431	2490	1887	1430	1887	1431	2,281900
apt15	2594	2125	1791	2125	1791	2594	2125	1791	2125	1791	2,601845
apt16	1623	1154	777	1154	784	1623	1154	777	1154	784	0,338958
apt17	1575	1261	1028	1261	1028	1575	1261	1028	1261	1028	0,906806
apt18	1778	1308	935	1308	936	1778	1308	935	1308	936	0,842916
apt19	2649	2235	1968	2235	1968	2649	2235	1968	2235	1968	4,398158
apt20	2681	2128	1723	2128	1723	2681	2128	1723	2128	1723	3,722216
b1	1794	1107	626	1107	659	1794	1107	626	1107	659	0,253355
b2	1800	1201	802	1201	809	1800	1201	802	1201	809	0,339908
b3	1797	1157	720	1157	749	1797	1157	720	1157	749	0,275118
b4	1801	1190	778	1190	788	1801	1190	778	1190	788	0,330847
b5	1796	1136	680	1136	717	1796	1136	680	1136	717	0,272742
b6	1776	1199	846	1199	847	1776	1199	846	1199	847	0,369881
b7	1801	1464	1326	1464	1326	1801	1464	1326	1464	1326	18,828665
gcut1	165	8	4	8	8	165	3	3	3	3	0,000662
gcut2	183	47	23	47	47	183	42	23	42	42	0,001806
gcut3	180	40	25	40	40	180	35	25	35	35	0,001284
gcut4	188	80	48	80	80	188	76	47	76	76	0,007102
gcut5	356	22	10	22	22	356	21	15	21	21	0,000570
gcut6	367	37	17	37	37	367	22	15	22	22	0,001352
gcut7	354	32	18	27	32	354	21	17	17	21	0,003681
gcut8	374	132	58	132	132	374	129	57	129	129	0,012508
gcut9	660	10	6	10	10	660	7	7	7	7	0,001161
gcut10	741	48	17	46	48	741	35	17	35	35	0,002485
gcut11	727	103	35	103	103	727	91	35	91	91	0,010337
gcut12	732	114	41	114	114	732	106	41	106	106	0,010431
gcut13	2885	2227	1684	2227	1691	2885	2227	1684	2227	1691	1,165023
gcut14	3385	2815	2360	2815	2360	3385	2815	2360	2815	2360	4,110175
gcut15	3385	2902	2534	2902	2534	3385	2902	2534	2902	2534	7,149933
gcut16	3385	2925	2580	2925	2580	3385	2925	2580	2925	2580	10,507114
gcut17	3385	2937	2604	2937	2604	3385	2937	2604	2937	2604	18,155462
uu1	400	95	36	95	46	400	76	33	76	40	0,003649
uu2	627	142	35	138	115	627	107	35	104	87	0,009819
uu3	779	110	34	109	110	779	88	34	88	88	0,003795
uu4	956	306	76	304	195	956	278	76	275	173	0,018656
uu5	1039	367	111	367	195	1039	366	111	366	191	0,030544
uu6	1116	180	49	179	160	1116	172	49	172	145	0,014175
uu7	1614	505	128	505	358	1614	498	128	498	351	0,051816
uu8	1535	347	84	347	314	1535	320	81	320	287	0,033870
uu9	1959	510	120	510	209	1959	473	117	473	194	0,053632
uu10	2750	798	170	798	273	2750	758	166	758	304	0,140902
uu11	3658	1589	335	1583	564	3658	1564	339	1558	547	0,101550

5. Conclusões e Perspectivas Futuras

Neste trabalho realizou-se um estudo a cerca de diferentes estratégias de discretização da malha de pontos e sobre a atualização do tamanho dos itens em problemas de empacotamento. As malhas investigadas foram *canonical dissections*, *reduced raster points*, *regular normal patterns* e *meet-in-the-middle patterns*.

Por meio de experimentos computacionais em instâncias da literatura foi possível analisar se existe equivalência entre as malha antes e depois da atualização do tamanho dos itens. Assim, pode-se concluir que a malha *reduced raster points* é a que contém a menor quantidade de pontos quando comparada com as outras malhas em estudo. No entanto, não há garantia que ela preserva a solução ótima do problema, sendo ainda uma questão em aberto na literatura. Em contrapartida, a malha *canonical dissections*, *regular normal patterns* e *meet-in-the-middle patterns* não são as mais refinadas, porém reduzem consideravelmente a quantidade de pontos presentes na malha unitária sem que ocorra perda da solução ótima.

Ademais, as malhas após a atualização do tamanho dos itens geraram conjuntos com menos pontos, quando comparada com as malhas geradas antes, o que mostra que não existe equivalência, mas sim uma dominância da malha com os itens atualizados. A atualização dos itens permite refinar um pouco mais as malhas de pontos, reduzindo o número de posições para empacotar os itens, o que pode gerar um menor esforço e tempo computacional na busca por uma solução ótima.

Para trabalho futuro, busca-se desenvolver um método que resolva o problema de empacotamento considerando as diferentes estratégias de discretização do recipiente. Assim, analisar, em termos de solução, as diferenças entre as malhas de pontos, dado um tempo limite de resolução.

Agradecimentos

Os autores agradecem o apoio financeiro recebido da Fundação de Amparo à Pesquisa do Estado de Goiás (FAPEG) e do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq - processo 308312/2016-3).

Referências

Alvarez-Valdés, R., Parreño, F., e Tamarit, J. M. (2005). A branch-and-cut algorithm for the pallet loading problem. *Computers & Operations Research*, 32(11):3007–3029.

Birgin, E. G., Lobato, R. D., e Morabito, R. (2010). An effective recursive partitioning approach for the packing of identical rectangles in a rectangle. *Journal of the Operational Research Society*, 61(2):306–320.

Birgin, E. G., Lobato, R. D., e Morabito, R. (2012). Generating unconstrained two-dimensional non-guillotine cutting patterns by a recursive partitioning algorithm. *Journal of the Operational Research Society*, 63(2):183–200.

Boschetti, M. A. e Mingozzi, A. (2003). The two-dimensional finite bin packing problem. part i: New lower bounds for the oriented case. *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 1(1):27–42.

Boschetti, M. A., Mingozzi, A., e Hadjiconstantinou, E. (2002). New upper bounds for the two-dimensional orthogonal non-guillotine cutting stock problem. *IMA Journal of Management Mathematics*, 13(2):95–119.

Carnieri, C., Mendoza, G. A., e Gavinho, L. G. (1994). Solution procedures for cutting lumber into furniture parts. *European Journal of Operational Research*, 73(3):495–501.

Christofides, N. e Whitlock, C. (1977). An algorithm for two-dimensional cutting problems. *Operations Research*, 25(1):30–44.

Côté, J.-F. e Iori, M. (2016). The meet-in-the-middle principle for cutting and packing problems. Technical Report CIRRELT-2016-28, Centre interuniversitaire de recherche sur les reseaux d'entreprise, la logistique et le transport.

Crainic, T. G., Perboli, G., e Tadei, R. (2008). Extreme point-based heuristics for three-dimensional bin packing. *Informs Journal on computing*, 20(3):368–384.

de Almeida, A. e Figueiredo, M. B. (2010). A particular approach for the three-dimensional packing problem with additional constraints. *Computers & Operations Research*, 37(11):1968–1976.

Del Valle, A. M., De Queiroz, T. A., Miyazawa, F. K., e Xavier, E. C. (2012). Heuristics for two-dimensional knapsack and cutting stock problems with items of irregular shape. *Expert Systems with Applications*, 39(16):12589–12598.

Garey, M. R. e Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-Completeness*. San Francisco: Freeman.

Gonçalves, R. F. e Queiroz, T. A. (2014). The knapsack problem with three practical constraints. *Procedia Computer Science*, 29:2192–2200.

Herz, J. C. (1972). Recursive computational procedure for two-dimensional stock cutting. *IBM Journal of Research and Development*, 16(5):462–469.

Junqueira, L., Morabito, R., e Yamashita, D. S. (2010). Modelos de otimização para problemas de carregamento de contêineres com considerações de estabilidade e de empilhamento. *Pesquisa Operacional*, 30(1):73–98.

Junqueira, L., Morabito, R., e Yamashita, D. S. (2012). Three-dimensional container loading models with cargo stability and load bearing constraints. *Computers & Operations Research*, 39(1): 74–85.

Lins, L., Lins, S., e Morabito, R. (2003). An l-approach for packing (l, w)-rectangles into rectangular and l-shaped pieces. *Journal of the Operational Research Society*, 54(7):777–789.

Martello, S., Pisinger, D., e Vigo, D. (2000). The three-dimensional bin packing problem. *Operations Research*, 48(2):256–267.

Nascimento, O. X., Cunha, J. G. d. A., e de Queiroz, T. A. (2016). Resolução do problema de empacotamento ortogonal com diferentes malhas e restrições reais. *Pesquisa Operacional para o Desenvolvimento*, 8(3):236–264.

Queiroz, T. A., Miyazawa, F. K., Wakabayashi, Y., e Xavier, E. C. (2012). Algorithms for 3d guillotine cutting problems: Unbounded knapsack, cutting stock and strip packing. *Computers & Operations Research*, 39(2):200–212.

Queiroz, T. A., Miyazawa, F. K., e Wakabayashi, Y. (2015). On the l-approach for generating unconstrained two-dimensional non-guillotine cutting patterns. *4OR*, 13(2):199–219.

Scheithauer, G. e Terno, J. (1996). The g4-heuristic for the pallet loading problem. *Journal of the Operational Research Society*, 47(4):511–522.

Wäscher, G., Haußner, H., e Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183(3):1109–1130.