Contribuições para um problema prático de empacotamento de caixas em um veículo multicompartimentado

Rodolfo Ranck Jr[1], Horacio Hideki Yanasse[2], Reinaldo Morabito[3]

[1]Programa de Doutorado em Computação Aplicada – CAP, INPE

[2]Laboratório Associado de Computação e Matemática Aplicada – LAC, INPE

[3]Departamento de Engenharia de Produção – DEP, UFSCAR

rodolforanck@gmail.com, horacio@lac.inpe.br, morabito@ufscar.br

Abstract. *We investigate a container loading problem where boxes have to be packed in a multi-compartmented vehicle to be delivered to customers in a defined order. The requisites of stability, load bearing, orientation and balancing have to be met while minimizing the time spent with the cargo handling in each delivery point. We present a heuristic to that problem that consists in packing the items into horizontal layers. Experiments show that the proposed method is able to solve realistic instances of the problem.*

Resumo. *Investiga-se um problema de carregamento de contêineres em que caixas devem ser empacotadas em um veículo multicompartimentado para serem entregues a clientes em uma ordem definida. Deve-se atender a quesitos de estabilidade, empilhamento, orientação e balanceamento enquanto minimiza-se o tempo gasto com o remanejamento da carga em cada ponto de entrega. Apresenta-se uma heurística para este problema que consiste em empacotar caixas em camadas horizontais. Experimentos mostram que a heurística é capaz de resolver instâncias realistas do problema.*

Palavras-chave: *problema de empacotamento, veículos multicompartimentados.*

1. Introdução

Empacotar caixas (itens) em contêineres (objetos) pode ser uma atividade complexa quando se busca eficiência; por exemplo, quando itens requeridos por um cliente precisam ser empacotados em um mínimo de objetos. Cada maneira de arranjar itens em um objeto define um padrão de empacotamento, em que os itens não podem ocupar o mesmo lugar no espaço e devem estar inteiramente contidos no objeto.

Um problema de empacotamento prático é o de carregar itens em contêineres. Dois tipos similares desse problema são o Problema de Carregamento de Contêineres (CLP) e o Problema de Carregamento de Paletes (PLP). Uma possível diferença entre eles refere-se à estabilidade da carga; diferentemente do PLP, no CLP os itens podem ser apoiados diretamente nas paredes do contêiner (Bischoff e Ratcliff, 1995). Segundo Hodgson (1982), o PLP pode dividido em PLP do Produtor (PLP-M), os itens são todos idênticos, e PLP do Distribuidor (PLP-D), os itens podem ser distintos. Os CLPs e PLPs são NP-árduos, pois englobam os problemas de corte e empacotamento unidimensionais. O PLP-M, por considerar itens idênticos, aparenta ser mais fácil do que o PLP-D, mas sua complexidade não é conhecida (Lins et al., 2003).

O transporte da carga em paletes e contêineres pode demandar diversos aspectos práticos não contemplados em problemas de empacotamento mais simples. Em Bischoff e Ratcliff (1995) discutem-se doze desses aspectos para os CLPs ou PLPs.

Os algoritmos exatos propostos na literatura têm apresentado eficácia limitada na resolução de instâncias de tamanhos realistas do CLP e do PLP-D o que tem motivado o desenvolvimento de heurísticas para esses problemas. Algumas delas são baseadas na estratégia de dividir o objeto em partes menores e resolver, para cada uma delas, um problema menor. Segundo Pisinger (2002) essas heurísticas podem ser classificadas de acordo com a maneira de dispor os itens no objeto: Paredes Virtuais, arranjo em camadas horizontais (Bischoff et al., 1995; Morabito e Arenales, 1997) ou verticais (George e Robinson, 1980); Pilhas de Itens, arranjo em colunas verticais (Morabito e Arenales, 1997); Cortes Guilhotinados, arranjos que poderiam ser separados com cortes ortogonais (Morabito e Arenales, 1994); Cubóides, arranjo em blocos (Bortfeldt et al., 2003). Heurísticas para estes problemas também podem ser classificadas em: construtivas (Bischoff et al., 1995; Zhu et al., 2012); meta-heurísticas (Corcoran e Wainwright, 1992) e de busca em árvores (Lim et al., 2003).

Formulações para problemas de empacotamento podem ser encontradas, para o caso bidimensional, em Beasley (1985), Scheithauer e Terno (1993) e Martins (2002); e, para o caso tridimensional, em Tsai et al. (1993), Chen et al. (1995) e Junqueira (2009). A grande maioria dessas formulações contemplam apenas as restrições de conteúdo, sobreposição e ortogonalidade, enquanto a formulação de Junqueira (2009) trata de diversos quesitos práticos de carregamento.

Para uma revisão recente do CLP na literatura, veja Bortfeldt e Washer (2012).

1.1 Definição do Problema

Um veículo dispõe de *n* compartimentos ou contêineres que podem ser acessados independentemente por portas em suas laterais. Itens retangulares devem ser colocados nos compartimentos sobre paletes. Um compartimento comporta no máximo um palete.

O veículo parte carregado do armazém e sua rota bem como as demandas de todos os clientes são conhecidas. A rota de um veículo define uma sequência de paradas que ele deve realizar para atender todos os clientes. Ao empacotar os itens, procura-se atender os quesitos (R1-R5) enquanto o objetivo O1 é buscado: **Orientação (R1),** os itens têm restrições do tipo "este lado para cima"; **Empilhamento (R2),** em qualquer pilha, o item abaixo deve ter resistência suficiente para sustentar os itens colocados acima dele; **Estabilidade da Carga (R3),** um item deve ser sustentado por outros itens ou pelas paredes presentes no contêiner; **Atendimento da demanda (R4),** devem-se empacotar todos os itens demandados pelos clientes; **Balanceamento do veículo (R5),** deve ser mantido durante toda a rota; **Minimizar o tempo de descarregamento dos itens em uma rota (O1).** Para alcançar este objetivo, procura-se diminuir o tempo gasto com o manejo da carga e com o acesso aos itens. Para tal, consideramos que no descarregamento, quando um item não está no topo outros acima dele precisam ser remanejados, tornando o acesso a um item mais rápido quanto mais próximo ele estiver do topo.

1.2 Justificativa do Estudo

O problema estudado neste trabalho é motivado por um problema prático de entrega de bebidas e foi estudado em Ranck et al. (2011). Os autores apresentaram um método de solução que consiste em empacotar os itens demandados em **camadas horizontais**. Essas camadas podem ser: **completas**, se contêm apenas itens de mesma família (de mesmas dimensões, resistência e peso) e na maior quantidade possível; ou **incompletas**, caso contrário. Uma camada incompleta é um agrupamento de itens residuais e tem de ficar no topo de um palete por motivos de estabilidade. Observamos as seguintes defici-

ências na abordagem de Ranck et al. (2011): **(a)** camadas completas e incompletas são empacotadas apenas em etapas separadas, e os itens residuais atribuídos na segunda etapa podem violar as restrições de empilhamento das camadas completas já empacotadas, ou não caberem no espaço fixo reservado a eles *a priori*; **(b)** ao empacotar os itens residuais não se observa o arranjo geométrico, logo, as soluções entendidas como viáveis pelos autores podem ser inviáveis na prática. Este artigo apresenta uma versão do método de Ranck et al. (2011) como alternativa para tratar esses pontos desfavoráveis.

2. Desenvolvimentos

O método de solução aqui proposto (Tabela 1 e Tabela 2) busca empacotar as camadas completas e incompletas (geradas como uma estimativa) de uma única vez. Se uma solução viável é obtida, tenta-se melhorá-la empacotando novamente os itens residuais, de maneira definitiva, em um procedimento recorrente de atribuição e empacotamento.

Seja: $\boldsymbol{L1}^*$: lista contendo somente os itens residuais ordenados hierarquicamente de maneira: inversa às paradas do veículo; decrescente por altura; e decrescente por peso, nessa ordem. $\boldsymbol{L2}^*$: lista inicialmente vazia; $\boldsymbol{NLC}$: conjunto de camadas completas; $\boldsymbol{qL_f}$: número máximo de itens da família f em uma camada; $\boldsymbol{M_f}$: conjunto dos itens demandados da família *f*; $\boldsymbol{M'}$: conjunto dos itens residuais.

Tabela 1. Procedimento principal para empacotar itens nos compartimentos.

1	**Início do Procedimento** *Main()*
2	Gere o maior número possível de camadas completas para cada uma das famílias *f* de itens, $maxL_f^{comp} = \lfloor |M_f|/qL_f \rfloor$, atribuindo itens às camadas em ordem inversa à sequência das paradas;
3	**Repita**
4	Gere as camadas incompletas como uma estimativa com *incPack1*(L1*);
5	**Se** algum item residual não pôde ser empacotado por *incPack1*($L1^*$)
6	**Fim do Procedimento** *Main()*
7	Tente empacotar todas as camadas geradas resolvendo a formulação (1)-(13);
8	**Se** a uma solução viável para (1)-(13) não puder ser obtida
9	**Se** o conjunto NLC for vazio
10	**Fim do Procedimento** *Main()*
11	Atribua ao conjunto M' os itens da camada completa com o menor valor de resistência ao empilhamento e exclua esta camada do conjunto NLC;
12	Exclua as camadas incompletas geradas;
13	**Enquanto** uma solução viável para (1)-(13) não puder ser obtida;
14	Armazene a solução obtida e Exclua as camadas incompletas da solução;
15	Gere novamente as camadas incompletas e tente empacota-las nos topos dos compartimentos com ResPack($L1^*$);
16	**Fim do Procedimento** *Main()*

Tabela 2. Procedimento para tentar empacotar novamente os itens residuais.

1	**Inicio Procedimento** *ResPack(L2*)*
2	**Enquanto** houver itens em $L2^*$;
3	Atribua os itens da lista $L2^*$ aos compartimentos resolvendo (14)-(22);
4	**Se** uma solução viável para o problema (14)-(22) não puder ser obtida
5	**Fim Procedimento** *ResPack()*
6	**Para** $j \in nC$
7	Tente empacotar os itens residuais atribuídos a *j* com *incPack2(j, L2*)*;
8	Remova de $L2^*$ os itens residuais que puderam ser empacotados em *j*;

9	Fixe $x_{ij} = 0, i \in M_f$, se um item da família *f* não pôde ser empacotado em *j;*
10	Fixe $x_{ij} = 1, i \in M'$, se o item i foi empacotado em *j;*
11	**Fim_Enquanto**
12	**Fim_Procedimento** *ResPack()*

Veja detalhes dos procedimentos *incPack1* e *incPack2* em §3.

Sejam: **variáveis**: $\boldsymbol{z_{is'}}$**:** é igual a 1 se a camada *i* contém itens que devem ser descarregados na parada s' e é colocada sobre outra que contém itens a serem descarregados antes da parada s' e igual a 0, caso contrário; $\boldsymbol{u}$**:** auxilia na redução do desbalanceamento do veiculo; $\boldsymbol{y_{ijk}}$**:** é igual a 1 se a camada *i* é colocada na posição *k* do compartimento *j* e igual a 0, caso contrário; $\boldsymbol{x_{ij}}$**:** é igual a 1 se o item residual *i* é colocado no compartimento *j* e igual a 0, caso contrário; **parâmetros:** $\boldsymbol{\omega^u, \omega^z_{is}, \omega^x_{ij}}$**:** custos utilizados para penalizar respectivamente as variáveis u e $z_{is}, e\ x_{ij}$; $\boldsymbol{a_{is}}$**:** é igual a 1 se a camada *i* possui algum item que deve ser descarregado na parada *s* e igual a 0, caso contrário; $\boldsymbol{\varepsilon}$**:** parâmetro de satisfatibilidade para o balanceamento do veículo; $\boldsymbol{fV}$**:** fração do volume disponível no compartimento *j* para colocar itens residuais; $\boldsymbol{l_i, L_j}$: comprimento, respectivamente, do item *i* e do compartimento *j;* $\boldsymbol{w_i, W_j}$**:** largura, respectivamente, do item *i* e do compartimento *j;* $\boldsymbol{h_i, H_j, H_j^{res}}$**:** altura, respectivamente, do item *i*, do compartimento *j* e do espaço disponível no compartimento *j* para os itens residuais; $\boldsymbol{Lt, Rt}$**:** conjunto dos compartimentos localizados, respectivamente, do lado esquerdo e direito do veículo; $\boldsymbol{nC, nS}$**:** respectivamente, a quantidade de compartimentos e paradas disponíveis; $\boldsymbol{nL}$: é a quantidade de camadas disponíveis e admite-se que as camadas *1,... nLC* são completas e o restante delas, incompletas; $\boldsymbol{maxP_j}$ é a capacidade máxima de peso que a pilha de camadas completas do compartimento *j* pode suportar; $\boldsymbol{nS', nS''}$**:** respectivamente, a quantidade de paradas em que as restrições (9) e (21) devem ser aplicadas; $\boldsymbol{p_i}$: peso de um item/camada *i*; $\boldsymbol{p_{is}}$: peso de um item/camada *i* descarregado na parada *s*; $\boldsymbol{P_{is}}$: peso dos itens da camada *i* descarregados na parada *s;* $\boldsymbol{Q^{<}_{ij}}$**:** quantidade de paradas anteriores a *s* em que itens do compartimento *j* devem ser descarregados, em que *s* é a parada onde o item *i* deve ser descarregado; $\boldsymbol{R_i}$: quantidade máxima de peso suportada por uma camada *i;* $\boldsymbol{u_s^{l-r}}$**:** peso das camadas completas empacotadas do lado esquerdo do veículo menos o peso das camadas completas empacotadas do lado direito do veículo antes da parada *s;* $\boldsymbol{ubL_j}$: limitante superior para o número de camadas completas em um compartimento *j* ; η é um número grande. Para $(1) - (13)$ é suficiente que $\eta \geq \sum_{i=1}^{nLC} p_i + \sum_{i=nLC}^{nL} p_i$.

$$Min\ z_1 = \omega^u(u) + \sum_{s=1}^{nS} \sum_{i=1}^{nL} (\omega_{is}^z z_{is}) \quad (1)$$

$$\sum_{i=1}^{nL} \sum_{k=1}^{ubL_j} y_{ijk}\, h_i + \sum_{i=nLC}^{nL} y_{ijubL_j+1}\, h_i \leq H_j, \qquad j = 1, \dots, nC \quad (2)$$

$$\sum_{j=1}^{nC} \sum_{k=1}^{ubL_j} y_{ijk} = 1, \qquad i = 1, \dots, nLC \quad (3)$$

$$\sum_{j=1}^{nC} y_{ij,ubL_j+1} = 1, \qquad i = nLC, \dots, nL \quad (4)$$

$$\sum_{i=1}^{nLC} y_{ijk} \leq 1, \qquad j = 1, \dots, nC;\ k = 1, \dots, ubL_j \quad (5)$$

$$\sum_{i=nLC}^{nL} y_{ij,ubL_j+1} \leq 1, \qquad j = 1, \dots, nC \quad (6)$$

$$\Delta P^l = \sum_{j \in Lt} \left(\sum_{i=1}^{nLC} \sum_{k=1}^{ubL_j} y_{ijk}\, p_i + \sum_{i=nLC}^{nL} y_{ij,ubL_j+1} p_i \right)$$
$$\Delta P^r = \sum_{j \in Rt} \left(\sum_{i=1}^{nLC} \sum_{k=1}^{ubL_j} y_{ijk}\, p_i + \sum_{i=nLC}^{nL} y_{ij,ubL_j+1} p_i \right) \quad (7)$$

$$\Delta P^l - \Delta P^r \begin{matrix} \leq \\ \geq \end{matrix} \begin{matrix} u + \varepsilon \\ -u - \varepsilon \end{matrix} \tag{8}$$

$$\Delta P^l - \Delta P^r + \sum_{s=1}^{s'} \left[\sum_{i=1}^{nLC} \left(\sum_{j \epsilon Lt} \sum_{k=1}^{ubL_j} y_{ijk} P_{is} - \sum_{j \epsilon Rt} \sum_{k=1}^{ubL_j} y_{ijk} P_{is} \right) + \sum_{i=nLC}^{nL} \left(\sum_{j \epsilon Lt} y_{ij,ubL_j+1} P_{is} - \sum_{j \epsilon Rt} y_{ij,ubL_j+1} P_{is} \right) \right] \begin{matrix} \leq \\ \geq \end{matrix} \begin{matrix} u + \varepsilon \\ -u - \varepsilon \end{matrix}, \quad s' = 1, \dots, nS' \tag{9}$$

$$\sum_{i=1}^{nLC} y_{ijk} (R_i - \eta) + \eta \geq \sum_{i=1}^{nLC} \sum_{k'=k+1}^{ubL_j} p_i y_{ijk'} + \sum_{i=nLC}^{nL} p_i y_{ij,ubL_j+1}, \quad \begin{matrix} j = 1, \dots, nC; \\ k = 1, \dots, ubL_j \end{matrix} \tag{10}$$

$$\sum_{i=1}^{nLC} (y_{ijk'} a_{is'} - z_{is'}) \leq 1 - \sum_{i=1}^{nLC} y_{ijk} a_{is}, \quad \begin{matrix} j = 1, \dots, nC \\ k' = 2, \dots, ubL_j; s' = 2, \dots, nS' \\ 1 \leq k < k'; 1 \leq s < s' \end{matrix} \tag{11}$$

$$\sum_{i=nLC}^{nL} (y_{ij,ubL_j+1} a_{is'} - z_{is'}) \leq 1 - \sum_{i=1}^{nLC} y_{ijk} a_{is}, \quad \begin{matrix} j = 1, \dots, nC; s' = 2, \dots, nS' \\ 1 \leq k < ubL_j + 1; 1 \leq s < s' \end{matrix} \tag{12}$$

$$\begin{matrix} y_{ijk} \in \{0,1\}, \quad i = 1, \dots, nL; \; j = 1, \dots, nC; \; k = 1, \dots, ubL_j + 1 \\ z_{is'}, \; \in \{0,1\}, \quad i = 1, \dots, nL; s = 2, \dots, nS \\ u \geq 0, \end{matrix} \tag{13}$$

A função objetivo em (1) reduz custos com desbalanceamento e custos em descarregar itens em uma parada quando há outros sobre eles que devem ser descarregados nas próximas paradas. As restrições: (2) impõem que a altura da pilha de camadas colocada em um compartimento não seja maior que a altura disponível nele; (3)-(4) impõem que todas as camadas sejam empacotadas; (5)-(6) impõem que camadas não se sobreponham; (7)-(9) impõem que o módulo da diferença entre os pesos totais dos itens empacotados dos lados esquerdo e direito do veículo, antes de uma parada s, seja minimizado se maior do que ε; (10) impõem que o peso total das camadas sobre outra não seja maior do que o peso máximo que esta outra camada pode suportar; (11)-(12) impõem custos quando uma camada é colocada sobre outra e itens da camada acima devem ser entregues depois dos da camada abaixo; (13) definem o domínio das variáveis de decisão.

$$min\, z_2 = \sum_{i \epsilon M'} \sum_{j=1}^{nC} \omega_{ij}^{x} Q_{ij}^{\leq} x_{ij} + \widehat{\omega}^{u} u \tag{14}$$

$$\sum_{i \epsilon M'} x_{ij} (l_i w_i h_i) \leq fV (L_j W_j H_j^{res}), \quad j = 1, \dots, nC \tag{15}$$

$$x_{ij} h_i \leq H_j^{res}, \quad i \, \epsilon \, M'; j = 1, \dots, nC \tag{16}$$

$$\sum_{i \epsilon M'} x_{ij} p_i \leq max P_j, \quad j = 1, \dots, nC \tag{17}$$

$$\sum_{j=1}^{nC} x_{ij} = 1, \quad i \, \epsilon \, M' \tag{18}$$

$$\begin{matrix} \Delta P^l = \sum_{i \epsilon M'} \sum_{j \epsilon Lt} x_{ij} p_i \\ \Delta P^r = -\sum_{i \epsilon M'} \sum_{j \epsilon Rt} x_{ij} p_i \end{matrix} \tag{19}$$

$$\Delta P^l - \Delta P^r + u_1^{l-r} \begin{matrix} \leq \\ \geq \end{matrix} \begin{matrix} u + \varepsilon \\ -u - \varepsilon \end{matrix} \tag{20}$$

$$\Delta P^l - \Delta P^r + u_1^{l-r} + u_{s+1}^{l-r} + \sum_{s=1}^{s'} \sum_{i \epsilon M'} \left(\sum_{j \epsilon Lt} x_{ij} p_{is} - \sum_{j \epsilon Rt} x_{ij} p_{is} \right) \begin{matrix} \leq \\ \geq \end{matrix} \begin{matrix} u + \varepsilon \\ -u - \varepsilon \end{matrix}, \quad s' = 1, \dots, nS'' \tag{21}$$

$$x_{ij} \epsilon \{0,1\}, \quad i \, \epsilon \, M'; j = 1, \dots, nC \tag{22}$$

A função objetivo (14) reduz custos com o desbalanceamento e minimiza os custos em atribuir um item residual *i* a um compartimento *j*, dados por $c_{ij} Q_{ij}^{\leq}$. As restrições: (15) impõem que o volume dos itens atribuídos ao compartimento *j* não seja maior do que o volume disponível nele multiplicado por um fator de ajuste; (16) impõem que um item não seja atribuído a um compartimento que não tenha altura disponível suficiente; (17) Impõem que o peso total dos itens atribuídos a um compartimento não seja maior do que a capacidade máxima de peso que uma pilha de camadas neste compartimento pode

suportar; (18) impõem que todos os itens sejam atribuídos; (19)-(21) possuem a mesma função das restrições (7)-(9); (22) definem o domínio das variáveis de decisão.

3. Testes Computacionais

Para a geração de resultados iniciais, os procedimentos *incPack1* e *incPack2* são dados por uma heurística construtiva simples que trata os quesitos R1, R2, R3 e objetivo O1 (veja §1.1). Considere o espaço de itens e objetos discretos; e que itens são empacotados ortogonalmente em um ponto (*p,q,r*) do objeto pelo seu canto frontal inferior esquerdo (que origina os eixos *X*, *Y* e *Z*).

Nessa heurística, tenta-se empacotar os itens residuais sequencialmente em um objeto (camada incompleta) a partir da lista ordenada $L1^*$. Para cada item, avaliam-se os pontos candidatos (adjacentes às faces dos itens empacotados e do objeto) em ordem lexicográfica nos eixo *X*, *Y* e *Z*, nessa ordem. O primeiro ponto em que um item puder ser colocado é escolhido. Se o item não puder ser colocado, ele é rotacionado em 90º no plano horizontal e repete-se a busca. Para determinar se um item *i* pode ser colocado em um ponto (*p, q, r*), além de caber no objeto, as restrições seguintes devem ser atendidas: **Empilhamento**: a pressão máxima suportada por qualquer ponto (*s, t, u*) da face superior do item *i* não pode ser excedida pela pressão exercida pelos itens colocados acima de (*s, t, u*). **Estabilidade**: o total de pontos da face inferior do item i em contato com a face superior de itens *j=1,...,m*, ou com o piso do objeto, deve ser maior do que $\boldsymbol{\alpha}(l_i w_i)$***;*** o total de pontos de sua face lateral esquerda em contato com a face lateral direita de itens *j=1,...,m*, ou com a parede esquerda do objeto, deve ser maior do que $\boldsymbol{\beta}(h_i w_i)$; o total de pontos de sua face lateral da frente em contato com a face lateral do fundo de itens *j=1,...,m*, ou com a parece frontal do objeto, deve ser maior do que $\boldsymbol{\gamma}(l_i h_i)$.

O procedimento *incPack2* é finalizado imediatamente após tentar empacotar todos os itens da lista no objeto *j* selecionado. Em *incPack1*, caso ainda hajam itens não empacotados, se possível, um novo objeto é utilizado e aplica-se novamente a heurística construtiva. No máximo podem ser utilizados *nC* objetos.

As instâncias deste problema são as mesmas utilizadas em Ranck et al. (2011). Sejam: ***nSMax, nFamíliasMax*** e ***nClientesMax***, respectivamente, o número máximo de paradas, de famílias distintas e de clientes gerados; ***dMin*** e ***dMax***, respectivamente, a menor e a maior demanda de um item gerada de uma vez; ***fCorte***, fator compreendido entre 0 e 1 que limita o volume total dos itens. Os dados seguintes foram utilizados para gerar as instâncias aqui avaliadas: *nClientesMax*= $\lceil 1{,}5 * nSMax \rceil$*; nFamíliasMax*=2**nC; dMin*= 10*; dMax*=$Max\left(\left\lceil 0{,}05 * \sum_{j=1}^{nC} H_j W_j L_j / (h_i w_i l_i) \right\rceil, dMin\right)$. A Tabela 3 apresenta outros dados utilizados:

Tabela 3. Dados das instâncias utilizadas.

	Classes de Instâncias							
Parâmetros	**I1**	**I2**	**I3**	**I4**	**I5**	**I6**	**I7**	**I8**
nSMax	10	10	10	10	20	20	20	20
nC	6	6	8	8	6	6	8	8
fCorte	0,6	0,7	0,6	0,7	0,6	0,7	0,6	0,7

Foram geradas 10 instâncias para cada uma das 8 classes avaliadas. Dados dos veículos e dos itens foram fornecidos pela empresa e utilizou-se $\boldsymbol{H_j = 1436}$*;* $\boldsymbol{L_j = 1200}$ *e* $\boldsymbol{W_j = 1000, j = 1, \dots, nC}$ para todos os casos. Ao todo foram consideradas 36 famílias de itens distintas; a Tabela 4 apresenta dados das dimensões dessas famílias:

Tabela 4. Dados estatísticos das famílias de itens utilizadas.

	Altura (mm)	Lado maior (mm)	Lado menor (mm)
Mínimo	60	130	42
Máximo	400	480	330
Média	229,8889	264,931	178,7722
Desvio Padrão	91,29756	92,4753	69,09287

As instâncias de um problema de Programação Linear Inteira são resolvidas com a ajuda do programa IBM ILOG CPLEX 12.4. O tempo máximo para a resolução computacional de uma instância foi de 15 minutos. Definiu-se: $\alpha = 1; \beta = \gamma = 0; fv = 0{,}75$; $\widehat{\omega}^u = \omega^u, = 1$; $\omega_{ij}^x = 1, i \epsilon M', j = 1, \ldots, nC$; $\omega_{is}^z = 1$, *i=1,...,nL, s=1,...,nS.* O valor de ε é definido como 10% do peso total dos itens demandados.

Sejam: ***cB*** *e* ***cT***, respectivamente, os custos com o balanceamento e com o tempo de descarregamento; ***nLD*** e ***nA***, respectivamente, o número de vezes em que os trechos compreendidos pelas linhas (3)-(13) da Tabela 1 e (2)-(11) da Tabela 2 precisaram ser executados; ***nLC*** e ***nLI***, número de camadas completas e incompletas; ***tC***, ***tI***, ***tE***, ***tD***, respectivamente, os tempos gastos para resolver uma instância dos problemas (1)-(13) e (14)-(22), e gastos pelos procedimentos *incPack1* e *incPack2*. A Tabela 5 apresenta os valores médios para as instâncias de uma classe. O cálculo de cB, cT e nLI considera somente a melhor solução encontrada para cada instância.

Tabela 5. Resultados dos testes computacionais.

	I1	I2	I3	I4	I5	I6	I7	I8
cB	0	0	0	0	0	0	0	0
cT	0,5	0,4	0,5	0,8	0,4	1,9	0,6	0,5
nLD	0	0,1	0	0,1	0,4	0,6	1,7	0,9
nA	2,6	2,7	2,8	2,9	2,6	2,9	2,7	2,8
nF	8,7	8,8	10,2	10,6	8,5	8,9	9,7	10,7
|M|	1092,4	1213	1318,3	1675,1	1228,2	915,4	1251,2	1207,1
|M'|	156,8	148,2	141,1	156	154	140	159,6	170,4
nLC	24,3	26,6	32,5	41	23,3	27,7	36,1	36,8
nLI	2,30	2,1	2,5	2,7	2,4	2,6	3,3	2,9
nS	10	10	10	10	12,8	14,1	13,7	15,8
tC	30,7	132,3	99,4	239	76,8	463,3	820,5	332,5
tI	1,6	2,5	2	2,5	2,7	1,3	2	2,3
tE	1,9	3,4	2,6	3,3	3,7	1,7	2	3,8
tD	1,6	2,7	2,1	2,7	3,0	1,4	1,6	3,1

4. Discussão de Resultados e Conclusões

Na Tabela 5, observa-se que o desbalanceamento do veículo permaneceu sempre dentro do limite ($\boldsymbol{\varepsilon}$) estipulado. Os custos com o tempo de descarregamento para o problema (1)-(13) foram baixos e em apenas 37 de 80 casos esses custos são diferentes de 0.

Em todos os casos, foi possível obter soluções factíveis para o procedimento da Tabela 1 e 88,75% delas são ótimas para a última instância de (1)-(13) resolvida nesse procedimento. A estratégia de "desmontar" uma camada completa e tornar seus itens residuais mostrou-se eficaz, e em 16,25% dos casos essa ação foi tomada pelo menos 1 vez para que uma solução factível pudesse ser obtida.

Em apenas 25% dos casos pode-se obter uma solução factível com o procedimento da Tabela 2. Nos casos restantes, alguns itens residuais ficam para fora do veículo (não são empacotados). Nenhuma das soluções obtidas com esse procedimento

foram melhores do que as obtidas com o procedimento da Tabela 1 e um motivo para isso é que os itens residuais precisaram de várias tentativas de atribuição para serem empacotados, diminuindo a qualidade da solução.

Os tempos computacionais gastos com o procedimento da Tabela 1 foram altos, mas para a maioria das instâncias esse tempo foi de até 20 segundos e somente em 23,7% dos casos eles extrapolam 100 segundos. Os casos que gastaram mais tempo são aqueles em que houve múltiplas tentativas de resolução por esse procedimento.

Apresentamos um método de solução para um problema prático de empacotamento. Pela primeira vez as instâncias puderam ser resolvidas considerando o empacotamento geométrico dos itens residuais e também restrições de empilhamento para toda a carga. O método proposto mostrou-se parcialmente eficaz para o conjunto de instâncias avaliado: enquanto soluções factíveis foram obtidas para todos os casos, não foi possível melhorá-las com o procedimento da Tabela 2. Para isso uma proposta de estudo iniciada é tornar o método *incPack2* mais eficiente. Dentre as possibilidades estão: melhorar a heurística construtiva empregada com métodos de busca local; e tentar resolver formulações exatas (ver §1) para esses casos. Pretende-se também estudar variantes do método que empacotam itens em camadas verticais.

Referências

Beasley, J. E. An exact two-dimensional non-guillotine cutting tree search procedure. *Operations Research*, v. 33, n. 1, p. 49-64, 1985.

Bischoff , E. E, Ratcliff, M. S. W. Issues in the development of approaches to container loading. *Omega*, v.23, n. 4, p. 377–390, 1995.

Bischoff, E. E., Janetz, F. E Ratcliff, M. S. W. Loading Pallets with Non-Identical Items. *European Journal of Operational Research*, v. 84, n. 3, p. 681-692, 1995.

Bortfeldt, A., Gehring, H., Mack, D. A parallel tabu search algorithm for solving the container loading problem. *Parallel Computing*, v. 29, n. 5, p. 641-662, 2003.

Bortfeldt, A., Wäscher, G., Container Loading Problems - A State-of-the-Art Review, No 120007, FEMM Working Papers, Otto-von-Guericke University Magdeburg, Faculty of Economics and Management, 2012.

Chen, C. S., Lee, S. M., Shen, Q. S. An analytical model for the container loading problem. *European Journal of Operational Research*, v. 80, n. 1, p. 68-76, 1995.

Corcoran, A. L., Wainwright, R. L. A genetic algorithm for packing in three dimensions. In: *1992 Acm/Sigapp*, 1992, Kansas City. Proceedings... NY: ACM Press, 1992. p. 1021-1030.

George, J. A., Robinson, D. F. A heuristic for packing boxes into a container. *Computers and Operations Research*, v. 7, n. 3, p. 147-156, 1980.

Hodgson TJ. A combined approach to the pallet loading problem. IIE Transac., v.14, n.3, p.175–82, 1982.

Junqueira, L. *Modelos de Programação Matemática para Problemas de Carregamento de Caixas dentro de Contêiners*. 2009. 134p. Tese (Mestrado em Engenharia de Produção) – UFSCAR, São Carlos - SP, 2009.

Lim, A., Rodrigues, B; Wang, Y. A multi-faced buildup algorithm for three-dimensional packing problems, *Omega*, v. 31, n. 6, p. 471-481, 2003.

Lins, L., Lins, S. & Morabito, R. An L-approach for packing (l,w)-rectangles into rectangular and L-shaped pieces. Journal of the Op. Research Society 54, 777-789, 2003.

Martins, G. H. A. Packing in two and three Dimensions. Tese de Doutorado, Naval Postgraduate School, Monterey, California, 2002.

Morabito, R., Arenales, M. An And/Or-graph approach to the container loading problem. *International Transactions in Operational Research*, v. 1, n. 1, p. 59-73, 1994.

Morabito, R., Arenales, M. Abordagens para o problema do carregamento de contêineres. *Pesquisa Operacional*, v. 17, n. 1, p. 29-56, 1997.

Pisinger, D. Heuristics for the container loading problem. *EJOR*, v. 141, n. 2, p. 382–92, 2002.

Ranck, R., Yanasse, H., Morabito, M. Um Problema de Empacotamento em um Veículo Multicompartimentado, XI Workshop de Computação Aplicada - INPE, 2011.

Scheithauer, G., Terno, J. Modelling of packing problems. *Optim.*, v. 28, n. 1, p. 63-84, 1993.

Tsai, R. D., Malstrom, E. M., Kuo, W. Three dimensional palletization of mixed box sizes. IIE Transactions, v. 25, n. 4, p. 64-75, 1993.

Zhu, W., Qin, H., Lim, A., Wang L. A Two-stage Tabu Search Algorithm with Enhanced Packing Heuristics for the 3L-CVRP and M3L-CVRP, Computers & Operations Research ,v*39*,n.*9*, 2178-2195, 2012.