Algoritmo GRASP/VND para o problema clássico de empacotamento tridimensional e bidimensional

Anderson Zudio[1], **Igor M. Coelho**[2], **Paulo Eustáquio Duarte Pinto**[3]
Instituto de Matemática e Estatística - Universidade do Estado do Rio de Janeiro (UERJ)
Rio de Janeiro - RJ - Brasil
azudio@ime.uerj.br[1], igor.machado@ime.uerj.br[2],
pauloedp@ime.uerj.br[3]

Celso C. Ribeiro
Instituto de Computação - Universidade Federal Fluminense (UFF)
Niterói - RJ - Brasil
celso@ic.uff.br

RESUMO

O problema de empacotamento consiste em armazenar ortogonalmente um conjunto de itens no menor número possível de caixas homogêneas. A versão clássica assume que os itens têm orientação fixa e não podem ser alocados com sobreposição. O problema de empacotamento tridimensional generaliza o problema unidimensional e é NP-Difícil. A versão tridimensional do problema, particularmente, tem várias aplicações industriais. Além disso, a versão clássica se relaciona a outros problemas complexos como os de corte, repartição e agendamento. Este trabalho propõe um algoritmo GRASP/VND híbrido simples de ser implementado para a versão clássica do problema de empacotamento tridimensional e bidimensional, onde o usuário precisa calibrar um único parâmetro. O empacotamento é baseado em uma estratégia de área maximal que modela o espaço remanescente nas caixas. Um teste computacional com 820 instâncias mostra que o algoritmo produz soluções com a mesma qualidade daquelas obtidas pelos melhores métodos da literatura.

PALAVRAS CHAVE. Empacotamento, Metaheurística, GRASP.

Tópico – Metaheurísticas

ABSTRACT

The bin packing problem consists in packing a set of items into the minimum number of bins. The classic version assumes itens with fixed orientation packed without overlapping. The three-dimensional case generalizes the unidimensional binpacking problem and it is a NP-Hard problem. Many industrial aplications has particular interest in the three-dimensional case. It relates to many complex problems like cutting, partitioning and scheduling. In this paper, a hybrid algorithm GRASP/VND is proposed for the classic three-dimensional and bi-dimensional binpacking problem that is simple to implement. The user needs to calibrate a single parameter. A packing strategy based on maximal areas is used for modeling the empty space left in the bins. Computional results on 820 test instances show that the algorithm obtains solutions with the same quality of those obtained by state-of-the-art methods in the literature.

KEYWORDS. Bin packing, Metaheuristic, GRASP.

Paper Topic – Metaheuristics

1. Introdução

O problema clássico tridimensional de empacotamento (3BP) consiste em empacotar ortogonalmente um conjunto de n itens retangulares caracterizados por sua largura w_i, altura h_i e profundidade d_i, $i \in S = \{1, 2, ..., n\}$, no menor número possível de caixas homogêneas de largura W, altura H e profundidade D. A quantidade de caixas é ilimitada, os itens tem orientação fixa e não podem ser alocados com sobreposição. Nenhuma outra restrição é imposta nesta versão do problema. Um modelo matemático do 3BP pode ser visto em [Hifi et al., 2014]. Considerando que todo item n tem sua profundidade $d_i = D$, o 3BP generaliza o problema clássico de empacotamento bidimensional (2BP). Análogo, o 3BP generaliza o problema bem conhecido de uma única dimensão (1BP). Assim, o problema 3BP é NP-difícil [Karp, 1972].

Várias aplicações industriais tem interesse no 3BP, por exemplo: aplicações de carregamento de veículos e manuseio de carga em aviões. Apesar da versão clássica ser uma simplificação do problema no mundo real, o problema de empacotamento aparece como parte de outros problemas complexos como os de corte, repartição ou até mesmo de agendamento.

Inicialmente, [Martello et al., 2000] propôs um método exato *Branch & Bound* que resolve o 3BP. O algoritmo só resolvia uma versão especial do problema. Portanto, foi modificado em [Martello et al., 2007] para resolver a versão geral do 3BP. Anteriormente, [Martello e Vigo, 1998] propôs um algoritmo *Branch & Bound* para a resolução do 2BP. [Fekete e Schepers, 2004a] lida com o problema em sua versão multi-dimensional modelando soluções através de uma representação com grafo. Os autores definem um grafo que descreve os itens que se sobrepõe através de sua posição relativa dentro da caixa e de sua projeção nos eixos ortogonais. Deste modo, [Fekete e Schepers, 2004b] propõe um *framework* utilizando as características desta modelagem para resolver instâncias grandes do problema de empacotamento multi-dimensional.

Os trabalhos já citados, [Martello e Vigo, 1998; Martello et al., 2000], estabelecem um limite inferior para o 3BP e 2BP estendendo um modelo para o limite inferior do 1BP. Um novo limite inferior foi introduzido por [Fekete e Schepers, 2004b] para o problema em sua versão bidimensional e tridimensional. [Boschetti e Mingozzi, 2003a,b] combina os resultados dos trabalhos anteriores apresentando um limite inferior para o 3BP e 2BP. Atualmente, os autores de [Liao e Hsu, 2013] estabelecem o atual limite inferior do 3BP dominante na literatura. Eles também analisam o caso no qual os itens podem ser rotacionados.

Vários algoritmos construtivos e heurísticas foram propostos para resolver problemas de empacotamento. O método exato proposto por [Martello et al., 2000, 2007] pode ser executado com um limite de tempo ou de iterações para fornecer soluções aproximadas. [Lodi et al., 1999a, 2002] desenvolveu algoritmos Busca Tabu baseado em procedimentos construtivos para o 2BP e 3BP. Assim, [Lodi et al., 2004] propõe uma heurística unificada Busca Tabu que resolve o problema multi-dimensional. [Faroe et al., 2003] propõe uma heurística Busca Local Guiada para o 3BP que também resolve o 2BP. O algoritmo se baseia em soluções iterativas do problema de satisfação de restrições. [Crainic et al., 2009] desenvolveu uma heurística Busca Tabu de dois níveis utilizando a representação de [Fekete e Schepers, 2004a], onde o primeiro nível foca em reduzir o número de caixas utilizadas e o segundo otimiza o empacotamento dos itens. Uma heurística híbrida GRASP/VND é proposta em [Parreño et al., 2010]. A fase de construção utiliza um valor máximo para as caixas, onde nem todos os itens precisam ser empacotados. Então, o algoritmo utiliza uma estrutura VND com diversas estratégias de reempacotamento para obter uma solução onde todos os itens são empacotados. Desta maneira, na fase de construção da próxima iteração GRASP, o número máximo de caixas é subtraído por um. O processo construtivo foi originalmente designado para resolver o problema de carregamento de contêiner proposto em [Parreño et al., 2008]. [Crainic et al., 2012] combina um algoritmo guloso com mecanismos de aprendizado para sintetizar uma heurística que resolve o 3BP. [Gonçalves e Resende, 2013] apresenta um Algoritmo Genético de chave Aleatória Viciada para o 3BP e 2BP que utiliza a estratégia de áreas maximais para efetuar o empacotamento. Recentemente, [Hifi et al., 2014] utiliza uma estratégia de programação linear

inteira sem o auxílio de meta-heurísticas para resolver o 3BP. O resultado é uma heurística separada em dois procedimentos. O primeiro método consiste de duas fases combinadas de aproximação e seleção, o segundo é uma extensão do primeiro com uma fase de otimização. Ambos são combinados para formar a heurística que determina a solução final.

Para a resolução do 2BP, [Boschetti e Mingozzi, 2003a,b] propõe uma heurística construtiva que determina uma pontuação para cada item. [Monaci e Toth, 2006] apresenta uma heurística baseada em cobertura de conjunto, onde a primeira fase gera uma coluna de itens para o empacotamento. O método utiliza várias heurísticas, incluindo [Boschetti e Mingozzi, 2003b] e [Martello e Vigo, 1998], limitadas por tempo de execução. Na segunda fase, estratégias que resolvem o problema de cobertura de conjunto são utilizadas para fornecer uma solução final para o 2BP.

Este trabalho apresenta um novo algoritmo GRASP (*Greedy Randomized Adaptive Search Procedure*) que resolve o 3BP e o 2BP. O método de empacotamento utiliza uma estratégia de áreas maximais para modelar o espaço vazio de uma caixa. O objetivo é preenchê-la começando pelos cantos, depois os lados e por fim o centro. Combinando quatro estratégias de reempacotamento dos itens em uma estrutura VND, o resultado é um algoritmo GRASP/VND híbrido simples de ser implementado e calibrado que obtém soluções de boa qualidade para instâncias grandes. A heurística proposta foi submetida em um teste composto de 820 instâncias. Tais instâncias são utilizadas como a base de teste de vários trabalhos na literatura. Os testes computacionais mostram que o algoritmo obtém soluções equivalente aos resultados obtidos pelos melhores métodos na literatura, que utilizam procedimentos complexos.

O restante deste trabalho é organizado da seguinte forma: a Seção 2 apresenta a estratégia utilizada para o empacotamento seguido pelo algoritmo GRASP/VND proposto; a Seção 3 reporta os resultados computacionais com as instâncias do 3BP e 2BP e a Seção 4 conclui o trabalho.

2. Algoritmo GRASP/VND

O método proposto neste trabalho utiliza a metaheurística GRASP. O GRASP é um procedimento iterativo, composto de uma fase de construção e uma de melhora, para resolver problemas de otimização combinatória [Resende e Ribeiro, 2016]. A fase de construção é composta por uma heurística construtiva gulosa com uma estratégia de randomização. Já a de melhora é originalmente composta por uma busca local, porém substituímos a busca local por uma estrutura VND (Variable Neighborhood Descent) que combina quatro vizinhanças de reempacotamento.

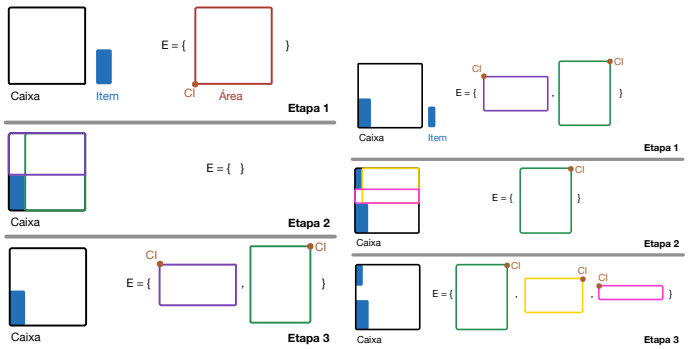
O procedimento que realiza o empacotamento dos itens utiliza a estratégia de Área Maximal (AM) para representar o espaço vazio de uma caixa. Ela é definida como o maior paralelepípedo vazio válido para a inserção de um item dentro de uma caixa. Portanto, cada AM é candidata para o empacotamento. No caso do 2BP, a definição segue o mesmo principio: uma AM é qualquer retângulo válido para a inserção de um item com a maior área possível. Esta estratégia é utilizada em [Parreño et al., 2010; Gonçalves e Resende, 2013].

Cada AM possui um ponto especial, Canto de Inserção (CI), para determinar a posição de empacotamento do item quando ela é selecionada. A escolha do CI é baseada no canto da AM que está mais próximo de um dos oito cantos de sua respectiva caixa. Caso mais de um canto da área seja apropriado, um dos cantos é escolhido aleatoriamente. A ideia desta estratégia é preencher os cantos da caixa primeiro, depois seus lados e finalizar com seu centro.

A representação da solução consiste de um vetor S que contém a configuração das m caixas B_i utilizadas, com $1 \leq i \leq m$. Cada caixa B_i é um vetor com n_i itens empacotados. Uma estrutura auxiliar E é utilizada para armazenar todas as AM disponíveis para o empacotamento de um novo item. Cada AM $\in E$ é associada com sua respectiva caixa B_i.

2.1. Fase de Construção

A heurística construtiva consiste em percorrer a lista de itens sequencialmente para empacotá-los. Assim, para cada item procura-se uma AM $\in E$ adequada para armazena-lo. Se nenhuma AM for apropriada, então uma nova caixa é aberta. São consideradas duas estratégias:



(a) Empcotamento do primeiro item.

(b) Empacotamento do segundo item.

Figura 1: Exemplo bidimensional do procedimento de empacotamento em três etapas.

- Best Fit – Considera a AM que resulta no maior aumento de ocupação de uma caixa considerando todos os itens já empacotados nela.

- Best AM – Considera a AM que resulta na maior ocupação do próprio espaço desconsiderando a ocupação da caixa.

Se empatar, a AM com o CI mais próximo do canto de sua respectiva caixa é selecionada. A ideia intuitiva do Best Fit é empacotar o item na caixa que gera a maior ocupação possível, ou seja, escolher a caixa onde sobre o menor espaço possível nela. Assim, o Best Fit preenche as caixas que já se encontram próximas do seu limite de espaço. O Best AM seleciona uma AM desconsiderando a ocupação de sua respectiva caixa, ou seja, nenhuma informação referente aos itens que já estão na caixa é utilizada na escolha. A ideia é preencher, o máximo possível, um determinado espaço vazio em uma caixa já aberta.

A heurística proposta utiliza duas funções importantes para o empacotamento dos itens. A primeira consiste em dividir a AM selecionada em três AM menores (duas no caso bidimensional). A segunda função analisa cada AM $\in E$ referente à respectiva caixa B_i em que o item foi empacotado, a fim de atualiza-la caso o item colida com a mesma. Note que a estrutura E comporta naturalmente elementos com intersecção, o que é intencional para o funcionamento do algoritmo.

A Figura 1 exemplifica um corte na AM bidimensional durante o processo de empacotamento. O procedimento é realizado em três etapas ilustradas na figura. A Etapa 1 consiste na escolha da AM apropriada que pode ser o através do Best Fit ou Best AM. A Etapa 2 consiste da remoção da AM da estrutura E. Logo após, identificam-se as novas AM que serão montadas através de um corte baseado na dimensão do item. A Etapa 3 consiste na alocação das duas novas AM em E com os seus respectivos CI determinados. Como citado anteriormente, há intersecção entre as AM de uma caixa já aberta, ou seja, o item empacotado pode estar dentro de várias AM. Portanto, um teste de colisão deve ser efetuado entre o item empacotado e cada AM $\in E$ referentes à caixa que acabou de ser utilizada. O elemento que colidiu é atualizado de forma análoga através de cortes nas dimensões do item, ou seja, o elemento é removido e novas AM são formadas com seus respectivos CI calculados. Outra observação é que cada AM sintetizada através das inserções e atualizações pode ser inválida. Estes casos são identificados quando o espaço da AM é menor que o espaço do menor item do problema. O Pseudocódigo 1 descreve a heurística construtiva, cuja função de empacotamento f varia de acordo Best Fit ou Best AM. A função *empacota()* efetua o processo de três etapas ilustrado na Figura 1, e a função *colisão()* atualiza cada AM $\in E$.

Para a heurística construtiva aleatória utilizada no GRASP/VND, foi posto um parâmetro de randomização nas estratégias de Best Fit e Best AM. A ideia é substituir a verificação da linha 8 do Pseudocódigo 1 por um vetor de AM candidatas, de modo que se deve manter as $100\alpha\%$ melhores AM para a escolha aleatória, onde α é o parâmetro que deve ser calibrado, $0 < \alpha < 1$.

Pseudocódigo 1 Heurística Construtiva

```
1:  função CONSTRUTIVA(Lista de Itens L)
2:      S ← Caixa 1                                   ▷ Solução: Lista de caixas
3:      E ← Nova AM relacionada a Caixa 1             ▷ Lista de AM
4:      para cada item I ∈ L faça
5:          AM* ← Nenhum                              ▷ Melhor AM
6:          para cada AM ∈ E faça
7:              se I cabe na AM então
8:                  se f(AM) > f(AM*) então           ▷ Best Fit ou Best AM
9:                      AM* ← AM
10:         se AM* ≠ Nenhum então
11:             empacota(I, AM*, E)        ▷ Empacota o item na melhor AM e atualiza E
12:             ID ← Identificação da caixa de AM*
13:             colisão(I, E, ID)          ▷ Atualiza E em busca de colisões com I
14:         senão                                     ▷ Não há AM para inserir I
15:             S ← Nova Caixa
16:             A ← Nova AM relacionada com a Nova Caixa
17:             empacota(I, A, E)
18:     Retornar S
```

2.2. Fase de Melhora

Esta fase busca soluções que melhorem a solução atual através de estratégias de reempacotamento dos itens com a heurística construtiva aleatória. Quatro vizinhanças V_k foram desenvolvidas para a estrutura VND (Variable Neighborhood Descent) [Hansen e Mladenović, 2014].

A primeira vizinhança V_1 consiste numa busca local que esvazia completamente duas caixas da solução S ignorando qualquer caixa com 100% de espaço utilizado através de um procedimento de reempacotamento. Tal procedimento é efetuado com a heurística construtiva aleatória da seção anterior. Assim, a vizinhança se baseia em esvaziar caixas duas a duas e reempacotar os itens. A diferença do procedimento de reempacotamento para a heurística construtiva proposta no Pseudocódigo 1 é que a solução atual S é passada como parâmetro de entrada em conjunto com a estrutura auxiliar E e a lista L dos itens que não estão empacotados. Além disso, quando mais de duas caixa são abertas, o reempacotamento desconsidera a solução atual e termina sua execução, a fim de evitar investigar soluções onde o número de caixas abertas seja maior que a solução da iteração atual. Foram consideradas as estratégias de Best Improvement e First Improvement.

A vizinhança V_2 esvazia metade das caixas da solução atual priorizando as com menor ocupação. Logo após, utiliza o mesmo procedimento de V_1 para o reempacotamento dos itens, portanto, V_2 só gera uma solução. A ideia intuitiva é reempacotar itens que são difíceis de empacotar, pois as caixas com menor têm tendência a manter os mesmos itens durante o processo iterativo do algoritmo. Seguindo esta mesma estratégia, a vizinhança V_3 também esvazia metade das caixas da solução atual, mas prioriza as caixas com maior ocupação. A ideia é reempacotar os itens que aparecem com frequência em caixas que ficam cheias para encontrar soluções que forcem estes itens a serem empacotados com itens mais difíceis de serem alocados em conjunto. Deste modo, V_3 gera uma única solução também. Finalmente, V_4 consiste em esvaziar metade das caixas escolhidas aleatoriamente. Em seguida, o procedimento de reempacotamento é aplicado. A ideia de V_4 é atingir soluções diversas, assim, ela é aplicada mais de uma vez escalando com a quantidade de iterações GRASP. Como dito antes, os métodos de reempacotamento aleatório desistem de empacotar itens caso tenham de abrir mais caixas do que as já presentes na solução de entrada. Então, as iterações do VND devem acelerar com o tempo. Logo, V_4 é repetida 3 vezes na iteração inicial e acumula mais uma tentativa a cada 10 iterações GRASP até atingir um acúmulo máximo de 20 tentativas.

3. Experimentos Computacionais

O objetivo desta seção é mostrar que o algoritmo produz soluções de boa qualidade para instâncias grandes do 2BP e do 3BP em pouco tempo de computação quando executado em um notebook padrão. O sistema foi implementado na linguagem de programação *C++*. O ambiente computacional utilizado possui a seguinte configuração: Processador Intel Core i5-240CM @2.5 GHz (1 núcleo utilizado); 8 GB RAM; Sistema Operacional Ubuntu 16.10 (x64). As soluções são comparadas com resultados obtidos na literatura em 320 instâncias do 3BP e 500 instâncias do 2BP.

3.1. Instâncias do 3BP

Tabela 1: Configuração dos tipos de itens gerados nas instâncias propostas para o 3BP.

Tipo	Dimensão dos itens		
	w	h	d
1	$[1, \frac{1}{2}W]$	$[\frac{2}{3}H, H]$	$[\frac{2}{3}D, D]$
2	$[\frac{2}{3}W, W]$	$[1, \frac{1}{2}H]$	$[\frac{2}{3}D, D]$
3	$[\frac{2}{3}W, W]$	$[\frac{2}{3}H, H]$	$[1, \frac{1}{2}D]$
4	$[\frac{1}{2}W, W]$	$[\frac{1}{2}H, H]$	$[\frac{1}{2}D, D]$
5	$[1, \frac{1}{2}W]$	$[1, \frac{1}{2}H]$	$[1, \frac{1}{2}D]$

O conjunto de testes para o 3BP é composto de 320 instâncias propostas por [Martello et al., 2000]. O gerador está disponível em `http://www.diku.dk/~pisinger/codes.html`. Este conjunto de instâncias foi utilizado em vários trabalhos da literatura [Lodi et al., 2002; Faroe et al., 2003; Lodi et al., 2004; Crainic et al., 2009; Parreño et al., 2010; Crainic et al., 2012; Gonçalves e Resende, 2013; Hifi et al., 2014]. Ele é dividido em 8 classes com 40 instâncias para os valores $n \in \{50, 100, 150, 200\}$, ou seja, 10 instâncias para cada valor n. As instâncias de classe $k = 1, ..., 5$ utilizam caixas de dimensão $W = H = D = 100$ com os itens gerados de acordo com os tipos apresentados na Tabela 1. Cada tipo conta com diferentes intervalos para as dimensões dos itens, a configuração da classe k gera 60% dos itens conforme o tipo k e o restante com 10% para cada um dos outros 4 tipos. As classes $k = 6, 7, 8$ seguem a seguinte configuração:

- Classe 6: $W = H = D = 10$; $w, h, d \in [1, 10]$
- Classe 7: $W = H = D = 40$; $w, h, d \in [1, 35]$
- Classe 8: $W = H = D = 100$; $w, h, d \in [1, 100]$.

3.2. Instâncias do 2BP

As instâncias para o caso bidimensional foram geradas por [Martello e Vigo, 1998; Berkey e Wang, 1987]. O conjunto com 500 instâncias é estruturado em 10 classes, onde cada classe possui 50 problemas para os valores $n \in \{20, 40, 60, 80, 100\}$. Todas as instâncias e sua correspondente melhor solução conhecida estão disponíveis em `http://or.dei.unibo.it/library/two-dimensional-bin-packing-problem`. Vários resultados na literatura são reportados utilizando este conjunto de testes [Lodi et al., 1999b, 2002; Faroe et al., 2003; Boschetti e Mingozzi, 2003b; Monaci e Toth, 2006; Parreño et al., 2010; Gonçalves e Resende, 2013].

3.3. Estratégias Selecionadas

Esta seção objetiva apresentar um teste preliminar para determinar as melhores estratégias para o algoritmo proposto. Inicialmente, uma calibração do parâmetro α foi realizada com um subconjunto de instâncias do 3BP e do 2BP. Os melhores valores obtidos foram 0,7 e 0,2 para o Best Fit e o Best AM respectivamente, assim, eles são utilizados durante todos os testes computacionais. O conjunto de testes utilizado nos resultados exibidos nesta seção é composto pelas instâncias de classe $k = 1, \ldots, 4$ do 3BP com $n \in \{50, 100, 150, 200\}$, ou seja, um total de 160 instâncias

do 3BP. Cada instância foi executada 30 vezes com sementes no intervalo sequencial $[1, 30]$ para controlar o âmbito de execução. A solução tomada é o valor médio obtido nas execuções.

O primeiro teste utiliza a heurística construtiva aleatória para avaliar o melhor método de empacotamento. Foi considerada a ordenação dos itens antes de utilizar o procedimento, de modo aos maiores itens serem empacotados primeiro. A Tabela 2 sumariza o resultado obtido através da relação entre as estratégias e o total de caixas acumuladas no teste. Os dados mostram que a ordenação dos itens melhora consideravelmente o resultado final. O resultado mostra que a estratégia de empacotamento Best AM fornece as melhores soluções.

Tabela 2: Comparação entre os métodos de empacotamento da heurística construtiva aleatória.

Empacotamento	Total Caixas
Best Fit sem ordenação	7210
Best AM sem ordenação	7170
Best Fit com ordenação	6927
Best AM com ordenação	**6922**

O segundo teste utiliza a busca local proposta na vizinhança V_1 da Seção 2 para avaliar as estratégias de First Improvement e Best Improvement. O reempacotamento aleatório também foi avaliado. A Tabela 3 exibe os resultados obtidos na execução do algoritmo GRASP original, onde o critério de parada foi de 10 segundos de tempo de execução em CPU. Os resultados mostram que o Best Improvement melhora a qualidade das soluções. Também mostra que o Best AM gera soluções de melhor qualidade quando comparado a estratégia de Best Fit.

Tabela 3: Comparação entre as estratégias de Busca Local.

Reempacotamento	Estratégia	Total Caixas
Best Fit	First Improvement	6850
	Best Improvement	6834
Best AM	First Improvement	6841
	Best Improvement	**6826**

O último teste avalia o impacto das estratégias propostas para a fase de melhora. Os resultados prévios mostraram que o Best AM gera as melhores soluções. Então, todos os métodos de reempacotamento das vizinhanças usam essa estratégia. A Tabela 4 mostra o algoritmo GRASP comparado com o GRASP/VND usando as vizinhanças estabelecidas. O VND-k consiste do uso das vizinhanças de 1 até k. O resultado mostra o impacto no uso da estrutura, de forma que as quatro vizinhanças combinadas geram os melhores resultados com o mesmo critério de parada.

Tabela 4: Comparação entre as estratégias da fase de melhora.

Fase de Melhora	Total Caixas
Busca Local	6826
VND-2	6787
VND-3	6782
VND-4	**6758**

3.4. Comparação Computacional

Seguindo o resultado da seção anterior, a configuração utilizada do GRASP/VND consiste em: ordenar os itens de entrada, aplicar a estratégia de Best AM com $\alpha = 0{,}2$ para o empacotamento em ambas fases, executar a busca local com Best Improvement junto com as outras 3 vizinhanças propostas. Logo, o GRASP/VND executou 30 vezes com sementes sequenciais no intervalo de

$[1, 30]$ para cada instância do conjunto estabelecido de 320 entradas do 3BP e 500 do 2BP. O critério de parada é de 10 segundos de execução em CPU, e a solução é o resultado médio dessas execuções.

Para avaliar a qualidade das soluções, comparamos o algoritmo proposto com outros métodos da literatura de acordo com a Tabela 5. Ela relaciona os algoritmos da literatura com sua estratégia de resolução. O algoritmo MVP foi executado no mesmo ambiente computacional deste trabalho, o seu código implementado em *C* está disponível em: `http://www.diku.dk/~pisinger/codes.html`.

Tabela 5: Métodos comparados da literatura.

Algoritmo Literatura	Referência	Estratégia
MVP	[Martello et al., 2000]	Branch-and-Bound Truncado (3BP)
TS3	[Lodi et al., 2002]	Busca Tabu (3BP/2BP)
GLS	[Faroe et al., 2003]	Busca Local Guiada (3BP/2BP)
GASP	[Crainic et al., 2012]	Heurística gulosa (3BP)
EHGH2	[Hifi et al., 2014]	Heurística baseada em programação linear inteira (3BP)
GVND	[Parreño et al., 2010]	GRASP/VND híbrido (3BP/2BP)
BRKGA	[Gonçalves e Resende, 2013]	Algoritmo Genético de Chave Aleatória Viciada (3BP/2BP)
HPB	[Boschetti e Mingozzi, 2003b]	Heurística (2BP)
SCH	[Monaci e Toth, 2006]	Heurística baseada em cobertura de conjunto (3BP)

A Tabela 6 mostra os resultados para as instâncias do 3BP. Os valores descritos nas linhas são as médias das 10 instâncias da classe para a quantidade de itens n relacionadas nas colunas 1 e 2. A coluna 3 mostra o limite inferior LB calculado por [Martello et al., 2000]. As colunas 4 e 5 exibem os resultados do GRASP/VND proposto com o tempo de execução em CPU para a melhor solução TT e a melhor solução Z que é média das caixas utilizadas. As colunas restantes mostram os resultados dos algoritmos da literatura reportados nos trabalhos referenciados. A exceção é o algoritmo MPV, pois os valores reportados são os obtidos no recurso computacional deste trabalho com o tempo limite de CPU de 60 segundos. As duas últimas linhas reportam o número total acumulado de caixas utilizadas em todas as instâncias para a comparação entre os algoritmos.

Os dados mostram que algumas soluções são equivalentes com as melhores reportadas na literatura, particularmente todas as 40 instâncias da classe 4 atingem o provável valor ótimo em poucas iterações. Observamos também que as outras soluções se aproximam da melhor reportada, ultrapassando a qualidade de outros procedimentos complexos. Além disso, existem vários casos em que a solução do conjunto de 10 instâncias utiliza somente 1 ou 2 caixas a mais do que a melhor reportada, por exemplo, as classes 1 e 5 com $n = [100, 150, 200]$. Finalmente, os dados mostram que o algoritmo proposto fornece soluções melhores que o MVP em todas as instâncias (executado no mesmo ambiente computacional com um intervalo de tempo 6 vezes maior); soluções melhores ou equivalentes ao GASP e o EHGH2; grande número de soluções melhores ou equivalentes ao TS3 e o GLS com um acúmulo final de caixas melhor; e algumas soluções equivalentes ao GVND e BRKGA que obtém soluções melhores que o algoritmo proposto. Notamos que os métodos da literatura consistem de procedimentos complexos que em alguns casos, como o BRKGA, exigem a calibração de várias estratégias. Para decidir se as diferenças que são observadas na tabela podem ser estatisticamente significantes, aplicamos o teste de Wilcoxon pareado. Ele mostra que o GRASP/VND é significativamente melhor que o MVP, GASP e o EHGH2; equivalente ao TS3 e o GLS; e pior que o GVND e o BRKGA.

Tabela 6: Resultado para as instâncias do 3BP

Classe	n	LB	TT	Z	MVP	TS3	GLS	GASP	EHGH2	GVND	BRKGA
1	50	12,5	4,1	**13,4**	13,6	13,4	13,4	13,4	13,8	13,4	13,4
	100	25,1	3,2	26,7	27,4	26,6	26,7	26,9	27,6	26,6	26,6
	150	34,7	2,8	36,6	37,7	36,7	37,0	37,0	39,8	36,4	36,4
	200	48,4	3,1	51,0	52,0	51,2	51,2	51,6	50,6	50,9	50,8
2	50	12,7	0,8	**13,8**	13,9	13,8	–	–	–	13,8	13,8
	100	24,1	2,8	26,0	26,4	25,7	–	–	–	25,7	25,6
	150	35,1	1,0	37,0	38,1	37,2	–	–	–	36,9	36,6
	200	47,5	1,8	49,6	50,2	50,1	–	–	–	49,4	49,4
3	50	12,3	0,9	**13,3**	13,6	13,3	–	–	–	13,3	13,3
	100	24,7	3,8	26,2	27,1	26,0	–	–	–	26,0	25,9
	150	36,0	0,4	37,7	39,2	37,7	–	–	–	37,6	37,5
	200	47,8	2,7	50,5	51,0	50,5	–	–	–	50,0	49,8
4	50	28,7	0,0	**29,4**	29,4	29,4	29,4	29,4	29,4	29,4	29,4
	100	57,6	0,1	**59,0**	59,2	59,0	59,0	59,0	59,5	59,0	59,0
	150	85,2	0,1	**86,8**	87,5	86,8	86,8	86,8	90,4	86,8	86,8
	200	116,3	0,3	**118,8**	119,0	118,8	119,0	118,8	119,0	118,8	118,8
5	50	7,3	0,0	8,4	9,2	8,4	8,3	8,4	7,9	8,3	8,3
	100	12,9	0,0	15,1	17,1	15,0	15,1	15,1	14,6	15,0	15,0
	150	17,4	0,3	20,2	24,0	20,4	20,2	20,6	21,5	20,1	20,0
	200	24,4	0,3	27,2	28,2	27,6	27,2	27,7	29,6	27,1	27,1
6	50	8,7	1,4	9,8	9,8	9,9	9,8	9,9	11,8	9,8	9,7
	100	17,5	0,9	19,1	19,3	19,1	19,1	19,1	19,2	19,0	18,9
	150	26,9	6,0	29,3	30,3	29,4	29,4	29,5	29,8	29,2	29,0
	200	35,0	8,1	37,7	37,8	37,7	37,7	38,0	38,7	37,4	37,3
7	50	6,3	0,6	**7,4**	7,9	7,5	7,4	7,5	7,4	7,4	7,4
	100	10,9	0,3	12,5	15,5	12,5	12,3	12,7	13,5	12,5	12,2
	150	13,7	0,2	16,0	19,9	16,1	15,8	16,6	18,2	16,0	15,3
	200	21,0	1,4	23,5	24,2	23,9	23,5	24,2	24,1	23,5	23,4
8	50	8,0	0,1	**9,2**	9,5	9,3	9,2	9,3	9,4	9,2	9,2
	100	17,5	0,4	**18,9**	21,4	18,9	18,9	19,0	18,9	18,9	18,9
	150	21,3	1,0	24,2	27,5	24,1	23,9	24,8	26,0	24,1	23,6
	200	26,7	4,5	29,9	31,1	30,3	29,9	31,1	35,8	29,8	29,3
Total (Classes 1, 4 – 8)				7301	7585	7320	7302	7364	7565	7286	7250
Total (Classes 1 – 8)				9842	10180	9863				9813	9769

*Valores em negrito correspondem a soluções equivalentes às melhores reportadas

A Tabela 7 exibe os resultados para o 2BP. Os valores reportados são as médias obtidas para o conjunto de instâncias da classe com seu respectivo valor n relacionados nas colunas 1 e 2. O limite inferior LB, reportado na coluna 3, foi obtido por [Monaci e Toth, 2006] através de um procedimento que aplica um algoritmo exato com um longo tempo de computação. As colunas 4 e 5 reportam os resultados referentes ao GRASP/VND para o 2BP com os mesmos dados da Tabela 6. As outras colunas reportam os valores obtidos pelos métodos da literatura.

Tabela 7: Resultado para as instâncias do 2BP.

Classe	n	LB	TT	Z	TS3	GLS	HBP	SCH	GVND	BRKGA
1	20	7,1	0,0	**7,1**	7,1	7,1	7,1	7.1	7,1	7,1
	40	13,4	0,0	**13,4**	13,5	13,4	13,4	13,4	13,4	13,4
	60	19,7	1,7	**20,0**	20,1	20,1	20,1	20,0	20,0	20,0
	80	27,4	0,0	**27,5**	28,2	27,5	27,5	27,5	27,5	27,5
	100	31,7	0,0	31,8	32,6	32,1	31,8	31,7	31,7	31,7
2	20	1,0	0,0	**1,0**	1,0	1,0	1,0	1,0	1,0	1,0
	40	1,9	0,1	**1,9**	2,0	1,9	1,9	1,9	1,9	1,9
	60	2,5	0,0	**2,5**	2,7	2,5	2,5	2,5	2,5	2,5
	80	3,1	0,0	**3,1**	3,3	3,1	3,1	3,1	3,1	3,1
	100	3,9	0,0	**3,9**	4,0	3,9	3,9	3,9	3,9	3,9
3	20	5,1	0,9	**5,1**	5,5	5,1	5,1	5,1	5,1	5,1
	40	9,2	0,4	9,5	9,7	9,4	9,5	9,4	9,4	9,4
	60	13,6	0,0	14,0	14,0	14,0	14,0	13,9	13,9	13,9
	80	18,7	0,1	19,0	19,8	19,1	19,1	18,9	18,9	18,9
	100	22,1	2,9	**22,3**	23,6	22,6	22,6	22,3	22,3	22,3
4	20	1,0	0,0	**1,0**	1,0	1,0	1,0	1,0	1,0	1,0
	40	1,9	0,0	**1,9**	1,9	1,9	1,9	1,9	1,9	1,9
	60	2,3	0,0	**2,5**	2,6	2,5	2,5	2,5	2,5	2,5
	80	3,0	0,0	3,2	3,3	3,3	3,3	3,2	3,1	3,1
	100	3,7	0,0	3,8	4,0	3,8	3,8	3,8	3,8	3,7
5	20	6,5	0,0	**6,5**	6,6	6,5	6,5	6,5	6,5	6,5
	40	11,9	0,2	**11,9**	11,9	11,9	11,9	11,9	11,9	11,9
	60	17,9	0,0	18,1	18,2	18,1	18,0	18,0	18,0	18,0
	80	24,1	0,0	**24,7**	25,1	24,9	24,8	24,7	24,7	24,7
	100	27,9	1,6	28,2	29,5	28,8	28,7	28,2	28,2	28,1
6	20	1,0	0,0	**1,0**	1,0	1,0	1,0	1,0	1,0	1,0
	40	1,5	1,2	1,7	1,9	1,8	1,8	1,7	1,7	1,6
	60	2,1	0,0	**2,1**	2,2	2,2	2,1	2,1	2,1	2,1
	80	3,0	0,0	**3,0**	3,0	3,0	3,0	3,0	3,0	3,0
	100	3,2	0,0	3,4	3,4	3,4	3,4	3,4	3,4	3,3
7	20	5,5	0,0	**5,5**	5,5	5,5	5,5	5,5	5,5	5,5
	40	10,9	0,0	11,4	11,4	11,3	11,1	11,1	11,1	11,1
	60	15,6	0,1	15,9	16,2	15,9	16,0	15,8	15,9	15,8
	80	22,4	0,0	**23,2**	23,2	23,2	23,2	23,2	23,2	23,2
	100	26,9	0,0	27,2	27,7	27,5	27,4	27,1	27,1	27,1
8	20	5,8	0,0	**5,8**	5,8	5,8	5,8	5,8	5,8	5,8
	40	11,2	0,0	11,4	11,4	11,4	11,3	11,3	11,3	11,3
	60	15,9	0,0	16,2	16,2	16,3	16,2	16,2	16,1	16,1
	80	22,3	0,0	22,5	22,6	22,5	22,6	22,4	22,4	22,4
	100	27,4	0,0	**27,8**	28,4	28,1	28,0	27,9	27,8	27,8
9	20	14,3	0,0	**14,3**	14,3	14,3	14,3	14,3	14,3	14,3
	40	27,5	0,0	**27,8**	27,8	27,8	27,8	27,8	27,8	27,8
	60	43,5	0,0	**43,7**	43,8	43,7	43,7	43,7	43,7	43,7
	80	57,3	0,0	**57,7**	57,7	57,7	57,7	57,7	57,7	57,7
	100	69,3	0,0	**69,5**	69,5	69,5	69,5	69,5	69,5	69,5
10	20	4,1	0,0	**4,2**	4,3	4,2	4,3	4,2	4,2	4,2
	40	7,2	0,0	**7,4**	7,5	7,4	7,4	7,4	7,4	7,4
	60	9,9	0,7	10,2	10,4	10,2	10,2	10,1	10,0	10,0
	80	12,5	0,0	13,0	13,0	13,0	13,0	12,8	12,9	12,8
	100	15,4	0,2	15,9	16,6	16,2	16,2	15,9	15,9	15,8
Total (Classes 1 – 10)				7257	7360	7284	7275	7243	7241	7234

*Valores em negrito correspondem a soluções equivalentes às melhores reportada

Os resultados mostram que muitas instâncias alcançam o melhor valor reportado na literatura com um tempo de execução inferior a 1 segundo no recurso computacional utilizado. Os resultados obtidos são equivalentes ou melhores que os resultados reportados pelos métodos TS3 e GLS. Outra observação é que o acúmulo final de caixas para todas as instâncias mostra que o algoritmo proposto obtém em média soluções de qualidade equivalente aos outros métodos: perdendo por uma margem de 0,3% para o BRKGA, 0,2% para GVND e SCH, ganhando por uma margem igualmente pequena de TS3, GLS e HBP. O teste de Wilcoxon mostra que os dados não apresentam significância estatística.

Não podemos fazer nenhum comentário comparativo para o tempo de computação em CPU, pois todos os métodos foram executados em diferentes ambientes computacionais e implementados através de diferentes linguagens de programação. A exceção é o algoritmo MVP para o 3BP. Com o código disponibilizado pelos autores, podemos observar que a Tabela 6 mostra que todos os resultados obtidos pelo GRASP/VND são melhores em um intervalo de tempo 6 vezes menor de computação de CPU.

4. Conclusão

Este trabalho apresentou um algoritmo GRASP/VND simples de ser implementado com um parâmetro calibrável para a resolução do problema de empacotamento tridimensional e bidimensional sem rotação. O método utiliza uma modelagem de áreas máximas para representar o espaço vazio nas caixas. Foram propostas diferentes estratégias de empacotamento para a fase de construção, e diferentes procedimentos para a fase de melhora da iteração GRASP. Através de um teste computacional preliminar foi possível determinar que o GRASP/VND, utilizando a função Best AM para o empacotamento e reempacotamento, obtém as soluções de melhor qualidade. Um teste computacional com 820 instâncias propostas na literatura mostrou que o algoritmo proposto obtém soluções de qualidade equivalente a outros métodos complexos de serem implementados e calibrados da literatura. Além disso, o método apresenta soluções de qualidade média superior com outros métodos relatados. Finalmente, através do teste com o empacotamento tridimensional, foi possível mostrar que o algoritmo proposto neste trabalho gera soluções melhores que o algoritmo MVP com um intervalo 6 vezes menor de tempo de computação.

Referências

Berkey, J. O. e Wang, P. Y. (1987). Two-dimensional finite bin-packing algorithms. *Journal of the operational research society*, 38(5):423–429.

Boschetti, M. A. e Mingozzi, A. (2003a). 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. e Mingozzi, A. (2003b). The two-dimensional finite bin packing problem. part ii: New lower and upper bounds. *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 1(2):135–147.

Crainic, T., Perboli, G., e Tadei, R. (2012). A greedy adaptive search procedure for multi-dimensional multi-container packing problems. https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2012-10.pdf.

Crainic, T. G., Perboli, G., e Tadei, R. (2009). Ts 2 pack: A two-level tabu search for the three-dimensional bin packing problem. *European Journal of Operational Research*, 195(3):744–760.

Faroe, O., Pisinger, D., e Zachariasen, M. (2003). Guided local search for the three-dimensional bin-packing problem. *Informs Journal on Computing*, 15(3):267–283.

Fekete, S. P. e Schepers, J. (2004a). A combinatorial characterization of higher-dimensional orthogonal packing. *Mathematics of Operations Research*, 29(2):353–368.

Fekete, S. P. e Schepers, J. (2004b). A general framework for bounds for higher-dimensional orthogonal packing problems. *Mathematical Methods of Operations Research*, 60(2):311–329.

Gonçalves, J. F. e Resende, M. G. (2013). A biased random key genetic algorithm for 2d and 3d bin packing problems. *International Journal of Production Economics*, 145(2):500–510.

Hansen, P. e Mladenović, N. (2014). Variable neighborhood search. In Burke, E. e Kendall, G., editors, *Search methodologies*, p. 313–337. Springer.

Hifi, M., Negre, S., e Wu, L. (2014). Hybrid greedy heuristics based on linear programming for the three-dimensional single bin-size bin packing problem. *International Transactions in Operational Research*, 21(1):59–79.

Karp, R. M. (1972). Reducibility among combinatorial problems. In Miller, R. e Thatcher, J., editors, *Complexity of Computer Computations*, p. 85–103. Springer.

Liao, C.-S. e Hsu, C.-H. (2013). New lower bounds for the three-dimensional orthogonal bin packing problem. *European Journal of Operational Research*, 225(2):244–252.

Lodi, A., Martello, S., e Vigo, D. (1999a). Approximation algorithms for the oriented two-dimensional bin packing problem. *European Journal of Operational Research*, 112(1):158–166.

Lodi, A., Martello, S., e Vigo, D. (1999b). Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. *INFORMS Journal on Computing*, 11(4):345–357.

Lodi, A., Martello, S., e Vigo, D. (2002). Heuristic algorithms for the three-dimensional bin packing problem. *European Journal of Operational Research*, 141(2):410–420.

Lodi, A., Martello, S., e Vigo, D. (2004). Tspack: a unified tabu search code for multi-dimensional bin packing problems. *Annals of Operations Research*, 131(1-4):203–213.

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

Martello, S., Pisinger, D., Vigo, D., Boef, E. D., e Korst, J. (2007). Algorithm 864: General and robot-packable variants of the three-dimensional bin packing problem. *ACM Transactions on Mathematical Software (TOMS)*, 33(1):1–7.

Martello, S. e Vigo, D. (1998). Exact solution of the two-dimensional finite bin packing problem. *Management science*, 44(3):388–399.

Monaci, M. e Toth, P. (2006). A set-covering-based heuristic approach for bin-packing problems. *INFORMS Journal on Computing*, 18(1):71–85.

Parreño, F., Alvarez-Valdés, R., Oliveira, J., e Tamarit, J. M. (2010). A hybrid grasp/vnd algorithm for two-and three-dimensional bin packing. *Annals of Operations Research*, 179(1):203–220.

Parreño, F., Alvarez-Valdés, R., Tamarit, J. M., e Oliveira, J. (2008). A maximal-space algorithm for the container loading problem. *INFORMS Journal on Computing*, 20(3):412–422.

Resende, M. G. e Ribeiro, C. C. (2016). *Optimization by GRASP*. Springer.