Capítulo **5**

Modelo de programação por restrições para o problema de empacotamento ortogonal tridimensional

Oliviana Xavier do Nascimento[1] Thiago Alves de Queiroz[1]
Liliane de Azevedo Oliveira[1]

Resumo: O Problema de Empacotamento Ortogonal em três dimensões (largura, altura e profundidade) tem por objetivo decidir se todas as caixas de um conjunto cabem dentro de um recipiente sem que elas se sobreponham. Para este problema, o presente trabalho apresenta um modelo de programação por restrições que o resolve. O modelo não busca pela melhor solução para o problema e sim por uma que seja viável. Além disso, utiliza-se um mecanismo, baseado nos pontos de discretização, para diminuir o número de variáveis do modelo que é testado sobre várias instâncias da literatura.

Palavras-chave: Problema de Empacotamento Ortogonal Tridimensional. Programação por Restrições. Pontos de Discretização.

Introdução

Os problemas de empacotamento visam, como o próprio nome sugere, empacotar itens dentro de recipientes. Tais problemas podem ser tratados em uma ou mais dimensões e sob variados escopos. Dentre estes problemas há um tipo que se propõem a decidir se um conjunto de itens pode ser empacotado sem que os itens ocupem a mesma posição no recipiente, ou seja, sem que ocorra sobreposição,

1 Universidade Federal de Goiás – UFG. Regional Catalão, Unidade Acadêmica Especial de Matemática e Tecnologia. Contato: olivianaxn@gmail.com, lilianeazevedoliveira@hotmail.com, taq@ufg.br.

com os itens inteiramente contidos no recipiente e com os itens empacotados com os seus lados paralelos aos lados do recipiente. Este é chamado de Problema de Empacotamento Ortogonal (OPP).

O OPP e suas versões bi- e tridimensional são problemas recorrentes no contexto industrial. Eles aparecem no carregamento de mercadorias em contêineres, caminhões e outros veículos de carga, no empilhamento de caixas sobre paletes, no corte de bobinas de papel, placas de madeira (indústria moveleira) ou placas e barras de aço (indústria siderúrgica). Isso justifica o estudo do OPP e de suas versões do ponto de vista aplicado.

No OPP a decisão com relação a um conjunto de itens ser empacotado é tomada verificando se existe, para a cada item, um ponto no recipiente no qual ele é empacotado satisfazendo as restrições impostas. Se estes pontos existirem, há uma solução viável para o caso em questão, se não, ele é considerado inviável.

Neste artigo, o OPP é tratado na sua versão tridimensional (3OPP), que considera largura, altura e profundidade e os itens como sendo caixas. O 3OPP é resolvido por meio de um modelo de programação por restrições, sendo antes realizado um teste para verificar se o volume total das caixas excede o volume do recipiente. Além disso, foram utilizados os pontos de discretização de Herz (1972) para diminuir o número de variáveis do problema.

A técnica de programação por restrições se fundamenta em variáveis e restrições. As variáveis representam as decisões do problema e as restrições determinam os valores que tais variáveis podem receber. Assim, esta técnica procura uma atribuição de valores para as variáveis que satisfaça todas as restrições definidas. Sobre a programação por restrições, Tavares (2000) pontuou que ela é uma técnica que se mostrou eficaz para a resolução de problemas intratáveis quando se recorreu a outras técnicas.

No que diz respeito a complexidade computacional, o 3OPP é NP-completo, o que significa a pouca provável existência de um algoritmo determinístico cujo o tempo de solução seja uma função polinomial do tamanho da entrada (GAREY; JOHNSON, 1979). Assim, usar a programação por restrições para resolver o problema é uma alternativa perante às formulações inteiras que geralmente demandam muito tempo para encontrar uma solução.

Em uma formulação inteira para o 3OPP, as variáveis estão relacionadas com todas as posições onde os itens podem ser empacotados e o método de resolução percorre uma árvore de enumeração para achar uma solução viável. Já em um modelo de programação por restrições, observa-se diretamente o domínio assumido para as variáveis que representam as posições para empacotar os itens. Diferente do método de resolução usado pela programação inteira, na programação por restrições não se lida com variáveis reais.

Na literatura o 3OPP foi resolvido por Fekete, Schepers e Veen (2007). Tais autores propuseram uma abordagem válida tanto para duas quanto para três dimensões, sendo baseada em grafos de intervalo utilizados para obter classes de empacotamentos viáveis. Clautiaux et al. (2008) e Mesyagutov, Scheithauer e Belov (2012) resolveram a versão bidimensional do OPP usando programação por restrições e os resultados foram bons quando comparados com outros resultados da literatura.

Este artigo está organizado da seguinte maneira: na Seção 1, o 3OPP é caracterizado, além de ser definido o teste de viabilidade, o modelo de programação por restrições e a estrutura do algoritmo usado para resolvê-lo. A Seção 2 mostra como foram realizados os experimentos computacionais, em que instâncias foram adaptadas da literatura, além de apresentar e discutir os resultados. Por fim, apresentam-se as conclusões e direções para trabalhos futuros.

1 O Problema de Empacotamento Ortogonal Tridimensional

O Problema de Empacotamento Ortogonal Tridimensional, o 3OPP, considera como instância I um par (V, R), no qual $V = \{1, \ldots, m\}$ é um conjunto de caixas que devem ser empacotadas em R, sendo $R = (L, P, A)$ um recipiente de largura L, profundidade P e altura A. Cada caixa i de V possui largura l_i, profundidade p_i e altura a_i, com $l_i, p_i, a_i \in \mathbb{N}$. O sistema de referência adotado se dá em termos dos eixos x, y e z, que representam, respectivamente, largura, profundidade e altura.

Dada uma instância do 3OPP, o objetivo é verificar se todos as caixas de V podem ser empacotadas em R sem que ocorra sobreposição entre quaisquer caixas, além disso as caixas devem estar todas contidas em R e devem ser organizadas de maneira ortogonal aos lados de R. Se isso é verificado, então há uma solução para a instância I.

1.1 Teste de inviabilidade

De acordo com Belov et al. (2009), se o volume total dos itens excede o volume do recipiente, o empacotamento não é possível. Ainda de acordo os autores, esta verificação consiste em um limitante para o problema e permite verificar rapidamente se o empacotamento é inviável.

Isso é importante, pois, não existindo empacotamento possível e tal fato podendo ser identificado de maneira bastante rápida, evita-se a resolução do modelo de programação por restrições.

1.2 Modelo de programação por restrições

A principal restrição envolvendo o 3OPP é a que garante a não sobreposição das caixas. Em um modelo de programação por restrições para o problema, esta restrição pode ser definida sobre as variáveis X_i, Y_i *e* Z_i. Elas representam o ponto (X_i, Y_i, Z_i) onde a caixa *i* é empacotada com relação a seu canto inferior frontal esquerdo. A essas variáveis são associados os respectivos domínios: $Dom\ (X_i) = \{0, 1, 2, \ldots, L - l_i\}$, $Dom\ (Y_i) = \{0, 1, 2, \ldots, P - p_i\}$ *e* $Dom\ (Z_i) = \{0, 1, 2, \ldots, A - a_i\}$. Tais domínios garantem que as caixas não ultrapassem os limites definidos pelas dimensões do recipiente.

Os domínios podem ser reduzidos usando os pontos de discretização de Herz (1972), obtidos a partir da combinação cônica entre as dimensões dos itens. Com este mecanismo, pontos redundantes para empacotar os itens podem ser descartados dos domínios das variáveis de decisão. O modelo de programação por restrições possui, então, uma restrição definida para cada par de caixas *i* e *j*, qual seja:

$$\begin{aligned} X_i + l_i \leq X_j \vee X_j + l_j \leq X_j \vee \\ Y_i + p_i \leq Y_j \vee Y_j + p_j \leq Y_i \vee \\ Z_i + a_i \leq Z_j \vee Z_j + a_j \leq Z_i \end{aligned} \qquad (1)$$

A restrição (1) garante que a caixa *i* é empacotada a esquerda, atrás ou abaixo da caixa *j* (e vice-versa). Deste modo, a restrição (1), uma vez definida para todo par de caixas *i* e *j*, evita que as caixas se sobreponham no recipiente.

1.3 Algoritmo proposto

O algoritmo para resolver o 3OPP primeiro considera uma instância para o problema e computa os pontos de discretização de Herz (1972), que contribuem para diminuir o número de variáveis. Em seguida, realiza-se o teste de inviabilidade, baseado no volume das caixas e do recipiente. Caso o volume total das caixas seja maior que o volume do recipiente, não existe solução para a instância considerada. Caso ocorra o contrário, o modelo de programação por restrições é chamado para resolver o problema e, se existe um conjunto de pontos (X_i, Y_i, Z_i) tais que as caixas são empacotadas sem gerar sobreposição com quaisquer outras caixas, diz-se que há uma solução viável para a instância, caso contrário, a instância é considerada inviável. A Figura 5.1 ilustra o algoritmo.

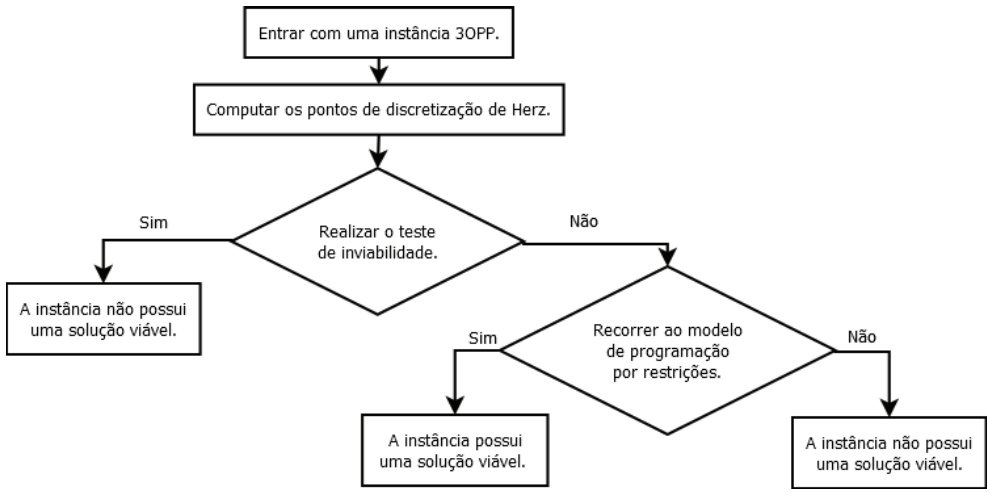

Figura 5.1 Algoritmo para resolver o 3OPP.

2 Experimentos Computacionais

O algoritmo apresentado para o 3OPP foi codificado na linguagem C++ e usando as classes e bibliotecas do IBM ILOG CPLEX Optimization Studio (IBM, 2014). Os experimentos ocorreram em um computador com processador Intel Core i5-3570 de 3,40 GHz, 8 GB de memória RAM e sistema operacional Linux.

Os experimentos foram realizados com instâncias contendo 10 e 20 caixas, recipientes de tamanho 10, 20 e 30 e caixas pequenas e grandes em relação ao tamanho do recipiente. Estas instâncias foram usadas por Junqueira, Morabito e Yamashita (2012) para resolver o problema de Carregamento em um Único Contêiner. Como aqui o problema tratado é o 3OPP, foram considerados, nas instâncias, somente os parâmetros que dizem respeito a este problema, que são: o número de itens, o tamanho do recipiente e o tamanho dos itens.

Na identificação das instâncias, n10, n20 e n30 representam que o tamanho do recipiente considerado na instância possui tamanho 10, 20 e 30, respectivamente; m10 e m20 representam o número de itens, sendo eles 10 e 20; b2d1 representa que as caixas são pequenas em relação ao recipiente e b2d2 que as caixas são grandes.

As Tabelas de 1 a 5 trazem os resultados para um conjunto de 50 instâncias adaptadas para o 3OPP, nas quais as caixas são consideradas pequenas em relação ao tamanho do recipiente. Elas mostram o nome da instância, se existe uma solução viável para a instância e o tempo, em segundos, gasto para obter a resposta. Os resultados obtidos mostram que todas as instâncias são viáveis, sendo resolvidas muito rapidamente.

Tabela 5.1 Resultados para instâncias com recipiente de tamanho 10 e 10 caixas pequenas.

Instância	Viável?	Tempo (s)
n10m10b2d1-01	Sim	0,012
n10m10b2d1-02	Sim	0,012
n10m10b2d1-03	Sim	0,013
n10m10b2d1-04	Sim	0,012
n10m10b2d1-05	Sim	0,011
n10m10b2d1-06	Sim	0,011
n10m10b2d1-07	Sim	0,013
n10m10b2d1-08	Sim	0,011
n10m10b2d1-09	Sim	0,013
n10m10b2d1-10	Sim	0,012

Tabela 5.2 Resultados para instâncias com recipiente de tamanho 10 e 20 caixas pequenas.

Instância	Viável?	Tempo (s)
n10m20b2d1-01	Sim	0,031
n10m20b2d1-02	Sim	0,024
n10m20b2d1-03	Sim	0,023
n10m20b2d1-04	Sim	0,048
n10m20b2d1-05	Sim	0,029
n10m20b2d1-06	Sim	0,025
n10m20b2d1-07	Sim	0,029
n10m20b2d1-08	Sim	0,032
n10m20b2d1-09	Sim	0,031
n10m20b2d1-10	Sim	0,034

Tabela 5.3 Resultados para instâncias com recipiente de tamanho 20 e 10 caixas pequenas.

Instância	Viável?	Tempo
n20m10b2d1-01	Sim	0,014
n20m10b2d1-02	Sim	0,012
n20m10b2d1-03	Sim	0,016
n20m10b2d1-04	Sim	0,013
n20m10b2d1-05	Sim	0,01
n20m10b2d1-06	Sim	0,015
n20m10b2d1-07	Sim	0,015
n20m10b2d1-08	Sim	0,012
n20m10b2d1-09	Sim	0,012
n20m10b2d1-10	Sim	0,012

Tabela 5.4 Resultados para instâncias com recipiente de tamanho 20 e 20 caixas pequenas.

Instância	Viável?	Tempo
n20m20b2d1-01	Sim	0,033
n20m20b2d1-02	Sim	0,036
n20m20b2d1-03	Sim	0,093
n20m20b2d1-04	Sim	0,033
n20m20b2d1-05	Sim	0,03
n20m20b2d1-06	Sim	0,041
n20m20b2d1-07	Sim	0,052
n20m20b2d1-08	Sim	0,049
n20m20b2d1-09	Sim	0,025
n20m20b2d1-10	Sim	0,034

Tabela 5.5 Resultados para instâncias com recipiente de tamanho 30 e 10 caixas pequenas.

Instância	Viável?	Tempo
n30m10b2d1-01	Sim	0,014
n30m10b2d1-02	Sim	0,011
n30m10b2d1-03	Sim	0,012
n30m10b2d1-04	Sim	0,011
n30m10b2d1-05	Sim	0,013
n30m10b2d1-06	Sim	0,011
n30m10b2d1-07	Sim	0,015
n30m10b2d1-08	Sim	0,015
n30m10b2d1-09	Sim	0,013
n30m10b2d1-10	Sim	0,013

As Tabelas de 6 a 10 trazem os resultados para um conjunto de 30 instâncias para o 3OPP, nas quais as caixas são consideradas grandes em relação ao tamanho do recipiente. Elas mostram o nome da instância, se existe uma solução viável e o tempo, em segundos, gasto para obter a resposta. Os resultados obtidos mostram que a maioria dessas instâncias são inviáveis, ou seja, não é possível empacotar todas as caixas.

Observa-se ainda que o tempo gasto para decidir que as instâncias são inviáveis foi menor do que o tempo gasto para mostrar a viabilidade do empacotamento. Isto se deve ao teste de inviabilidade, que evita resolver o problema pelo modelo de programação por restrições para os casos em que as caixas são grandes e excedem o volume do recipiente, assim resultando em um tempo menor de resolução.

Tabela 5.6 Resultados para instâncias com recipiente de tamanho 10 e 10 caixas grandes.

Instância	Viável?	Tempo
n10m10b2d2-01	Não	0,111
n10m10b2d2-02	Não	0,012
n10m10b2d2-03	Sim	0,018
n10m10b2d2-04	Sim	0,015

Continua

Tabela 5.6 Resultados para instâncias com recipiente de tamanho 10 e 10 caixas grandes. *(Continuação)*

Instância	Viável?	Tempo
n10m10b2d2-05	Sim	0,013
n10m10b2d2-06	Sim	0,022
n10m10b2d2-07	Sim	0,017
n10m10b2d2-08	Não	0,071
n10m10b2d2-09	Não	0,077
n10m10b2d2-10	Não	0,019

Tabela 5.7 Resultados para instâncias com recipiente de tamanho 10 e 20 caixas grandes.

Instância	Viável?	Tempo
n10m20b2d2-01	Não	<0,001
n10m20b2d2-02	Não	<0,001
n10m20b2d2-03	Não	<0,001
n10m20b2d2-04	Não	<0,001
n10m20b2d2-05	Não	<0,001
n10m20b2d2-06	Não	<0,001
n10m20b2d2-07	Não	<0,001
n10m20b2d2-08	Não	<0,001
n10m20b2d2-09	Não	0,011
n10m20b2d2-10	Não	<0,001

Tabela 5.8 Resultados para instâncias com recipiente de tamanho 20 e 10 caixas grandes.

Instância	Viável?	Tempo
n20m10b2d2-01	Não	<0,001
n10m20b2d2-02	Não	0,002
n10m20b2d2-03	Não	<0,001
n10m20b2d2-04	Não	<0,001

Contunua

Tabela 5.8 Resultados para instâncias com recipiente de tamanho 20 e 10 caixas grandes. *(Continuação)*

Instância	Viável?	Tempo
n10m20b2d2-05	Não	<0,073
n10m20b2d2-06	Não	<0,001
n10m20b2d2-07	Não	<0,108
n10m20b2d2-08	Não	<0,001
n10m20b2d2-09	Não	<0,001
n10m20b2d2-10	Não	<0,001

Tabela 5.9 Resultados para instâncias com recipiente de tamanho 30 e 10 caixas grandes.

Instância	Viável?	Tempo
n30m10b2d2-01	Não	<0,001
n30m10b2d2-02	Não	<0,001
n30m10b2d2-03	Não	<0,001
n30m10b2d2-04	Não	<0,001
n30m10b2d2-05	Não	<0,001
n30m10b2d2-06	Não	<0,001
n30m10b2d2-07	Não	<0,001
n30m10b2d2-08	Não	<0,001
n30m10b2d2-09	Não	<0,001
n30m10b2d2-10	Não	<0,001

Tabela 5.10 Resultados para instâncias com recipiente de tamanho 30 e 10 caixas grandes.

Instância	Viável?	Tempo
n30m20b2d2-01	Não	<0,001
n30m20b2d2-02	Não	<0,001
n30m20b2d2-03	Não	<0,001
n30m20b2d2-04	Não	<0,001

Contunua

Tabela 5.10 Resultados para instâncias com recipiente de tamanho 30 e 10 caixas grandes. *(Continuação)*

Instância	Viável?	Tempo
n30m20b2d2-05	Não	<0,001
n30m20b2d2-06	Não	<0,001
n30m20b2d2-07	Não	<0,001
n30m20b2d2-08	Não	<0,001
n30m20b2d2-09	Não	<0,001
n30m20b2d2-10	Não	<0,001

Conclusões

Neste artigo foi usada uma abordagem para resolver o 3OPP que inicialmente checa se o volume total das caixas é maior do que o volume do recipiente, em um teste para detectar rapidamente casos inviáveis. Caso o volume total das caixas seja menor ou igual ao volume do recipiente, significa que a instância pode ser viável e isto é checado, em seguida, por meio de um modelo de programação por restrições. O modelo considera o domínio das variáveis reduzido por meio dos pontos de discretização de Herz (1972).

Os resultados computacionais foram satisfatórios, pois foi possível resolver todas as instâncias em pouco tempo computacional. Eles confirmam que realizar o teste de inviabilidade é um mecanismo que torna o modelo mais rápido para as instâncias inviáveis. Para os casos resolvidos neste artigo, a diferença de tempo entre os casos viáveis e inviáveis é bem pequena. Mas, dependendo da instância, realizar o teste de inviabilidade pode representar uma diferença de tempo considerável. Quanto ao modelo de programação por restrições, este se mostrou eficaz na resolução das instâncias consideradas, uma vez que retornou com a solução em pouco tempo de processamento.

Agradecimentos

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

Referências

BELOV, G. et al. One-dimensional relaxations and lp bounds for orthogonal packing. *International Transactions in Operational Research*, v.16, p. 745 – 766, 2009.

CLAUTIAUX, F. et al. A new constraint programming approach for the orthogonal packing problem. *European Journal of Operational Research*, v. 95, n.3, p. 944-959, 2008.

FEKETE, S. P.; SCHEPERS, J; VEEN, J.C. van der. An exact algorithm for higher-dimensional orthogonal packing. *Operations Research*, v. 55, n. 3, p. 569-587, 2007.

GAREY, M. R.; JOHNSON, D. S. *Computers and intractability: a guide to the theory of NP-completeness*. San Francisco: Freeman, 1979.

HERZ, J. C. A recursive computational procedure for two-dimensional stock-cutting. *IBM Journal of Research Development*, p. 462-469, 1972.

JUNQUEIRA, L.; MORABITO, R.; YAMASHITA, D. S. Three-dimensional container loading models with cargo stability and load bearing constraints. Computers & Operations Research, v. 39, n. 1, p. 74 – 85, 2012.

MESYAGUTOV, M.; SCHEITHAUER, G.; BELOV, G. Lp bounds in various constraint programming approaches for orthogonal packing. *European Journal of Operational Research*, v. 39, n. 10, p. 2425-2438, 2012.

TAVARES, J. A. *Geração de configurações de sistemas industriais com o recurso à tecnologia das restrições e computação evolucionária*. Tese (Doutorado) — Tese de doutorado em Informática. Universidade de Minho, Braga, Portugal, 2000.