DECOMPOSIÇÃO DO PROBLEMA DE EMPACOTAMENTO BIDIMENSIONAL

Charbel Daher Boulos[1] (IC), Pedro Hokama (PQ)[2], Mário Cesar San Felice (PQ)[3]
[1]*Instituto de Engenharia de Sistemas e Tecnologia da Informação - Universidade Federal de Itajubá*
[2]*Instituto de Matemática e Computação - Universidade Federal de Itajubá*
[3]*Departamento de Computação - Universidade Federal de São Carlos*

Palavras-chave: Algoritmo, Descarregamento, Empacotamento, Restrições.

Introdução

No cotidiano das empresas, diversos problemas são analisados com o objetivo de melhorar o desempenho em diferentes áreas de atuação. Um exemplo comum é o desafio de carregar contêineres ou caminhões com produtos destinados à entrega ou armazenamento. Diante desse contexto, surge o Problema de Empacotamento Bidimensional Ortogonal (Two-dimensional Orthogonal Packing Problem - 2D-OPP) (BOSCHETTI, MINGOZZI, HADJICONSTANTINOU, 2002).

Além disso, quando um determinado item é retirado, não é desejado que seja necessário rearranjar os demais itens remanescentes no contêiner. Dessa forma, a ordem de entrega de determinados produtos pode alterar a sequência de carregamento do caminhão, configurando o Problema de Empacotamento Bidimensional Ortogonal com restrições de descarregamento (Two-dimensional Orthogonal Packing Problem with Unloading Constraints - 2D-OPP-UL). Este trabalho concentra-se no estudo desse problema.

O 2D-OPP-UL tem como entradas um contêiner C com altura H e largura W, um conjunto de itens I o qual cada item i tem sua altura h_i, sua largura w_i e o índice de precedência de descarregamento p_i. O objetivo do problema é decidir se há ou não uma solução de empacotamento, ou seja, deseja-se saber se os itens do conjunto I são empacotáveis no contêiner C respeitando os limites, a regra da não-sobreposição, onde dois itens i e j não podem se sobrepor numa mesma área, e a ordem de descarregamento, em que, para todo par de itens i e j, se $p_i < p_j$ então i deve poder ser retirado do contêiner em um único movimento na direção da saída do mesmo (representada em seu topo) antes do item j, sem ser obstruído.

Metodologia

Neste trabalho seguindo a ideia de Côté, Gendreau e Potvin (2014), o 2D-OPP-UL será abordado fazendo a decomposição do problema em duas partes, na primeira parte o 2D-OPP-UL é relaxado para o Problema do Empacotamento Unidimensional Contínuo em Contêineres (One-dimensional Contiguous Bin Packing Problem - 1CBP). Nesta relaxação, consideramos uma sequência de H bins unidimensionais, onde cada bin é denominado por H_i, todos de tamanho W. Para o conjunto de itens I, cada item i será dividido em k partes unitárias e cada parte é denominada por i_k, possuindo uma largura w_i. Os itens devem ser empacotados de maneira contínua, ou seja, se o item i foi empacotado no bin l, então o k-ésimo pedaço do item i deve ser empacotado no bin $l + k - 1$. Uma solução para o 1CBP ainda não é uma solução para o problema original, é preciso garantir as demais restrições já conhecidas do 2D-OPP, que são a de não sobreposição, a do limite de largura e altura do contêiner. Também será necessário adicionar a restrição de descarregamento (UL). A verificação se essas restrições podem ser atendidas é feita em uma segunda fase chamada x-check. Se com a solução atual do 1CBP não for possível atender essas restrições, a solução é eliminada e uma nova deve ser gerada. Caso seja encontrada uma solução para este problema, então foi obtido uma solução para o 2D-OPP-UL.

Para atacar o 1CBP foi utilizado a Programação Linear Inteira (*Integer Linear Programing* - ILP) e para auxiliar a resolução foi utilizado a Programação por restrições (*Constraint Programming* - CP). A maneira como isso foi feito será explicado abaixo.

- Programação Linear Inteira

Este tipo de programação tem esse nome, pois trabalha com um conjunto de variáveis inteiras, um conjunto de restrições (que são inequações lineares) e uma função objetivo que se deseja maximizar ou minimizar (que é uma expressão linear). Caso todas as restrições propostas sejam respeitadas atendendo a meta da função objetiva, então os valores atribuídos às variáveis formam uma solução para este problema (WOLSEY, 2020).

Antes de entender como a função objetiva e as restrições foram moldadas, é preciso apresentar algumas variáveis para a compreensão deste cenário. O conjunto de possíveis pontos que de empacotamento será P, de maneira mais específica, P_i^H é o conjunto de possíveis pontos que o item i pode ser empacotado no eixo vertical, este advém do domínio em Y (será mostrado como é obtido posteriormente). P^Hserá o conjunto com os pontos de todos os itens no eixo vertical, ou seja, é a união dos conjuntos P_i^H. Por fim, será chamado de t_{iy} o conjunto de variáveis binárias que são 1 quando o canto inferior do item i está no bin y e 0 quando não.

Para atender as condições do problema foram estabelecidas as seguintes restrições (CÔTÉ, GENDREAU, POTVIN, 2014):

$$\sum_{y \in P_i^H} t_{iy} = 1,\ i \in I$$

$$t_{iy} \in \{0,1\},\ i \in I,\ y \in P_i^H$$

Esta restrição garante que todo item será empacotado em apenas uma posição, já que o canto inferior esquerdo do item deve estar em apenas uma posição y.

$$\sum_{I \in I} \sum_{y \in P_i^H} w_i t_{iy} \leq W,\ y \in P^H$$

Já esta restrição garante que o somatório dos itens empacotados respeitará o limite de largura do bin.

$$\sum_{t_{iy} \in S'} t_{iy} \leq |S'| - 1,\ S' \in S$$

Por fim, essa restrição elimina soluções do 1CBP que a combinação de itens e coordenadas y não são viáveis considerando as demais restrições do 2D-OPP-UL. Essa solução será chamado de S'. Esse conjunto surge ao longo do problema como um conjunto de variáveis iguais a 1 no 1CBP tal que o x-check é inviável (CÔTÉ, GENDREAU, POTVIN, 2014). Por exemplo, na Figura 1 um subconjunto seria $S' = \{t^v_{1,0}, t^v_{2,0}, t^v_{3,1}\}$.

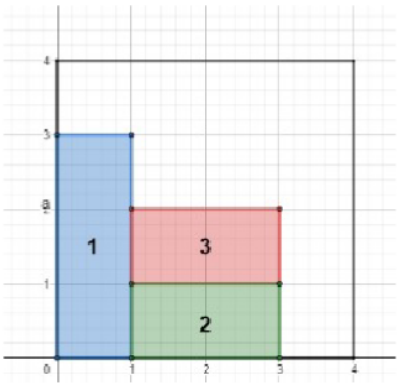

Figura 1 - Figura da solução v do 1CBP utilizado para compreender como é formado o conjunto S'.

Então, toda vez que for encontrada uma solução inviável no algoritmo do x-check é criado um subconjunto de S chamado de Conjunto minimal inviável ou simplesmente MIS (do inglês, *minimal infeasible set*). O intuito de encontrar este conjunto é obter o conjunto que de fato torna aquela solução inviável. O x-check será melhor explicado com a programação por restrições. Para formar este conjunto serão considerados todos os itens de I e serão removidos um item de cada vez ordenados do menor para o maior, o algoritmo então realiza o teste sem um determinado item, caso encontre uma solução viável para o x-check, o item volta para o conjunto e então o próximo item é removido. O procedimento acontece até considerar todos os itens, assim encontra-se um conjunto minimal de itens (e posições y) que quando empacotados juntos tornam o problema inviável.

Por ser um problema de decisão, uma função objetiva não é necessária. Entretanto, resultados preliminares mostraram que várias soluções do 1CBP deixavam os itens que seriam descarregados depois nos bins superiores, o que gerava muitas soluções inviáveis. Para contornar esse problemas, inserimos uma função objetivo que visa minimizar a altura dos itens que tivessem a menor precedência para serem descarregados, ou seja, que tenham a necessidade de ser removidos depois. Portanto, a função objetiva foi modelada da seguinte forma:

$$Min \sum_{i \in I} \sum_{y \in P_i^H} (p_i + 1) \times (y + 1) \times t_{iy}$$

- **Programação por restrições**

O CP é um paradigma da programação que trabalha com

um conjunto de variáveis as quais cada uma tem um conjunto finito de possíveis valores, denominado por D (domínio), e com uma ou mais restrições que podem reduzir o domínio de cada variável, ou seja, impedir combinações de determinados valores. Neste caso foi utilizado para cada item i a variável X para estabelecer a posição do canto i esquerdo do item no eixo das larguras e Y a posição do i canto inferior no eixo das alturas. Inicialmente, o domínio dessas variáveis são $D(X_i) = \{0,\dots, W - w_i\}$ para largura e $D(Y_i) = \{0,\dots, H - h_i\}$ para altura. Esses domínios garantem que o empacotamento não ultrapasse o limite de C. Além de garantir que os itens respeitarão os limites de C, também será garantido que, caso um item i tenha ordem de descarregamento igual a de um item j, então nenhum item i será sobreposto por outro j, pela seguinte restrição:

$$[X_i + w_i \leq X_j]\ ou\ [X_j + w_j \leq X_i]\ ou$$
$$[Y_i + h_i \leq H_j]\ ou\ [Y_j + h_j \leq Y_i]$$

No entanto, caso o item i tenha maior ordem de descarregamento, a restrição será dada por:

$$[X_i + w_i \leq X_j]\ ou\ [X_j + w_j \leq X_i]\ ou\ [Y_j + h_j \leq Y_i]$$

Neste caso, é garantido que o item i está à direita ou à esquerda, ou acima do item j, assim, garantindo que o item i será removido antes do item j.

O x-check é o algoritmo de *callback*[1] que visa achar, baseado em uma solução advinda do 1CBP, um conjunto de posições no eixo horizontal (ou também no eixo x, daí o nome) para ser usado em um 2D-OPP-UL de modo que os itens não se sobreponham e respeitem a restrição de descarregamento.

Para a formulação deste problema vamos assumir t^v como a solução binária do 1CBP com t^v_{iy} sendo y a posição no eixo vertical em que o item i está nesta solução v, dessa forma, adicionamos a restrição:

$$Y_i = y$$

Então, com essa solução do 1CBP, o algoritmo tentará empacotar a partir do canto inferior esquerdo e testar se as restrições não sobreposição e de descarregamento são respeitadas. Caso as restrições sejam respeitadas e o algoritmo encontre uma maneira de empacotar os itens em todos os H_i bins do 1CBP, então foi encontrado uma solução para o 2D-OPP-UL.

[1] É uma função de retorno a uma determinada ação, neste caso, a solução do 1CBP.

Resultados e discussão

Para testar o algoritmo feito foram utilizadas instâncias da literatura (HOKAMA, MIYAZAWA, XAVIER, 2016) as quais foram obtidas a partir dos testes que foram baseados em um conjunto de 60 instâncias do 2L-CVRP, ou também, *Capacitated Vehicle Routing Problem with Loading Constraints* (IORI, SALAZAR-GONZÁLEZ, VIGO, 2007), adaptadas a partir do CVRP original (REINELT, 1991). Essas instâncias do 2D-OPP-UL foram geradas executando um algoritmo exato para o 2L-CVRP. Sempre que o algoritmo encontrava uma rota que não podia ser resolvida pelo modelo baseado no paradigma da programação por restrições (CLAUTIAUX et al., 2008) em menos de 1 segundo, essa rota era salva como uma nova instância de 2D-OPP-UL, resultando em um total de 296 instâncias adicionais.

Estas 296 instâncias foram geradas para o 2D-OPP-UL e foram testadas utilizando 4 modelos diferentes. Cada modelo teve um tempo limite de 120 segundos para tentar encontrar uma solução para cada instância. Caso o modelo não encontre uma solução no tempo limite, então essa instância é dada como não resolvida e tenta resolver outra. Na Tabela 1 é possível observar os resultados obtidos em cada modelo.

Tabela 1 - Resultados que mostram o desempenho de cada modelo ao tentar resolver as 296 instâncias com o limite de tempo.

Classe	Instâncias resolvidas	Aproveitamento
I	23	7,77%
II	38	12,84%
III	56	18,92%
IV	64	21,62%

Fonte: Autor

A classe I é o modelo sem nenhuma melhoria. A classe II é o modelo com a aplicação do Conjunto minimal inviável (MIS). A classe III é o modelo com o uso da função objetiva. E por fim a classe IV é o modelo com o uso do MIS e da função objetiva.

Conclusões

Portanto, ao final deste trabalho foi possível compreender um pouco mais sobre os problemas 1CBP e x-check e, assim, entender como eles de alguma maneira podem contribuir para encontrar soluções para o 2D-OPP-UL. Além disso, também foi possível perceber a partir dos resultados que foram evidenciados na Tabela 1 que a utilização das melhorias do Conjunto minimal inviável adjunto do uso da função objetiva melhorou de maneira considerável o algoritmo para encontrar soluções para o 2D-OPP-UL.

Ademais, fica como uma possível continuação do trabalho implementar os Padrões (BOULOS, HOKAMA, SAN FELICE, 2023) para reduzir o domínio e, assim, consequentemente diminuir ainda mais o tempo para encontrar uma solução para o 2D-OPP-UL.

Agradecimentos

Ao Programa Institucional de Bolsas de Iniciação Científica (PIBIC) UNIFEI/UNIÃO pelo apoio financeiro.

Gostaria de agradecer também ao meu professor orientador Pedro Hokama que sempre me auxiliou e me deu suporte no desenvolvimento do trabalho e me ajudou a aprimorar diversas habilidades pessoais.

Referências

BOSCHETTI, Marco A.; MINGOZZI, Aristide; HADJICONSTANTINOU, Eleni. New upper bounds for the two-dimensional orthogonal non-guillotine cutting stock problem. **IMA Journal of Management Mathematics**, v. 13, n. 2, p. 95-119, 2002.

CLAUTIAUX, François et al. A new constraint programming approach for the orthogonal packing problem. **Computers & Operations Research**, v. 35, n. 3, p. 944-959, 2008.

CÔTÉ, Jean-François; GENDREAU, Michel; POTVIN, Jean-Yves. An exact algorithm for the two-dimensional orthogonal packing problem with unloading constraints. **Operations Research**, v. 62, n. 5, p. 1126-1141, 2014.

BOULOS, Charbel Daher; HOKAMA, Pedro Henrique Del Bianco; SAN FELICE, Mário César. Padrões para o Problema do Empacotamento Bidimensional. **Revista dos Trabalhos de Iniciação Científica**, 2023.

HOKAMA, Pedro; MIYAZAWA, Flávio K.; XAVIER, Eduardo C. A branch-and-cut approach for the vehicle routing problem with loading constraints. **Expert Systems with Applications**, v. 47, p. 1-13, 2016.

IORI, Manuel; SALAZAR-GONZÁLEZ, Juan-José; VIGO, Daniele. An exact approach for the vehicle routing problem with two-dimensional loading constraints. **Transportation science**, v. 41, n. 2, p. 253-264, 2007.

REINELT, Gerhard. TSPLIB—A traveling salesman problem library. **ORSA journal on computing**, v. 3, n. 4, p. 376-384, 1991.

WOLSEY, Laurence A. **Integer programming**. John Wiley & Sons, 2020.