Heurísticas para o Problema do Empacotamento Colorido*

Renan F.F. da Silva[1], **Yulle G.F. Borges**[1], **Rafael C.S. Schouery**[1]

[1]Instituto de Computação - Universidade Estadual de Campinas (Unicamp)
Av. Albert Einstein, 1251 - Cidade Universitária, Campinas - SP, 13083-852

r223989@dac.unicamp.br, glebbyo@ic.unicamp.br, rafael@ic.unicamp.br

Abstract. *The Colorful Bin Packing Problem (CBPP) is a generalization of the Bin Packing Problem (BPP) where, given a set of items with a size and a color, we must pack the items in bins of limited capacity, minimizing the number of bins used and satisfying the constraint that two items of the same color cannot be packed side by side in the same bin. In this article, we propose the adaptation of heuristics from BPP to CBPP along with new heuristics for the problem. We also propose a Variable Neighborhood Search based heuristic for the CBPP. The results indicate that our approach is capable of finding good solutions to large instances of the problem.*

Resumo. *O Problema do Empacotamento Colorido (CBPP) é uma generalização do Problema do Empacotamento (BPP) onde, dado um conjunto de itens com um tamanho e uma cor, devemos empacotar os itens em recipientes de capacidade limitada, minimizando a quantidade de recipientes utilizados e satisfazendo a restrição que dois itens de mesma cor não podem ser empacotados lado a lado em um mesmo recipiente. Neste artigo, propomos a adaptação de heurísticas do BPP para o CBPP acompanhado de algumas heurísticas novas para o problema. Propomos também uma heurística para o CBPP baseada em* Variable Neighborhood Search. *Os resultados indicam que a nossa abordagem é capaz de encontrar boas soluções para instâncias grandes do problema.*

1. Introdução

O Problema do Empacotamento Colorido (*Colorful Bin Packing Problem* – CBPP) é uma generalização do Problema do Empacotamento (*Bin Packing Problem* – BPP). No CBPP, dado um conjunto de itens $I = \{1, ..., n\}$, onde cada $i \in I$ tem um tamanho $0 < s_i \leq L$ e uma cor $c_i \in \mathbb{N}$, devemos empacotar I em recipientes $\{B_1, B_2, ..., B_k\}$ para algum k, todos de capacidade L, minimizando a quantidade k de recipientes utilizados e satisfazendo a restrição que dois itens de mesma cor não podem ser empacotados lado a lado em um mesmo recipiente. Isto é, para todo $1 \leq r \leq k$, temos que $\sum_{i \in B_r} s_i \leq L$ e existe uma permutação π de B_r tal que, para todo i e j em B_r, se $c_i = c_j$, então $\pi(i) \neq \pi(j) + 1$. Notamos que a permutação π não precisa ser considerada diretamente, pois se f for a cor com maior frequência em B_r, é possível provar que $|\{i : i \in B_r \wedge c_i = f\}| \leq \lceil |B_r|/2 \rceil$ é uma condição necessária e suficiente para a existência da permutação π.

*Estudo financiado pelos processos 2015/11937-9 e 2020/06511-0, Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), e processos 144257/2019-0, 308689/2017-8, 311039/2020-0 e 425340/2016-3, Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

O CBPP foi proposto independentemente em [Dósa and Epstein 2014] e em [Böhm et al. 2014], sendo estudado nesses artigos e em publicações posteriores apenas do ponto de vista de algoritmos online e de algoritmos de aproximação.

Tanto o BPP como o CBPP são problemas NP-difíceis, ou seja, não admitem algoritmo exato polinomial, a menos que P = NP. Visto isso, o uso de meta-heurísticas é sustentado como uma alternativa para se encontrar soluções de boa qualidade nestes problemas. Para o BPP já há bons resultados com o uso de meta-heurísticas, como em [Fleszar and Hindi 2002] com a *Variable Neighborhood Search* (VNS), e em [Loh et al. 2008] com a *Weight Annealing*. Nesse artigo, apresentamos heurísticas iniciais para o CBPP cujas soluções são posteriormente melhoradas pelo algoritmo VNS proposto em seguida.

2. Heurísticas Iniciais

Nós implementamos um total de quatro heurísticas iniciais. As duas primeiras são adaptações das heurísticas do BPP chamadas *Minimum Bin Slack'* e *Best Fit Decreasing*. Já as duas últimas são heurísticas novas, propostas por nós, chamadas *Hard BFD* e *Two By Two*. A seguir apresentamos essas heurísticas em maiores detalhes.

Na heurística *Minimum Bin Slack'* (MBS') proposta por [Fleszar and Hindi 2002], a cada passo criamos, dentre os itens restantes, um recipiente com o subconjunto de itens que melhor preenche o recipiente sem violar as restrições de cores e que contém o maior item. Já na heurística *Best Fit Decreasing* (BFD) proposta por [Johnson et al. 1974], nós processamos os itens em ordem não-crescente de tamanho e empacotamos o item atual no recipiente que ficar com a menor capacidade residual sem violar as restrições de cores. Se o item não puder ser empacotado em nenhum recipiente, então criamos um novo. A heurística *Hard BFD* é uma adaptação da BFD na qual antes de criar um recipiente novo, encontramos, para cada item j de cor diferente do item atual i, qual é o recipiente existente r mais cheio no qual i e j podem ser empacotados juntos. Se existir tal recipiente r, para algum j, então escolhemos o j que gerar o recipiente mais preenchido.

Por fim, na heurística *Two by Two* é criado um recipiente r e fazemos movimentos que minimizam uma função f. No primeiro movimento escolhemos um item para colocar em r, e em cada passo seguinte buscamos o melhor entre colocar um ou dois itens em r. Se não houver mais movimentos válidos, criamos um novo recipiente. Considere, então, um movimento m, isto é, um conjunto de até dois itens a serem empacotados. Seja S^m o conjunto de itens não empacotados excluindo m, e sejam $S^m_f \subseteq S^m$ o conjunto de itens da cor mais frequente em S^m e $S^m_n = S^m \setminus S^m_f$. A função f prioriza minimizar $(\alpha_m/L)^2 + |S| \cdot \left(\frac{|S_n|}{|S_f|} - \frac{|S^0_n|}{|S^0_f|}\right)^2$, onde S^0_f é o conjunto de itens da cor mais frequente de I, $S^0_n = I \setminus S^0_f$ e α_m é capacidade residual do recipiente r após empacotar m. Isto é, buscamos movimentos que, após realizados, minimizam a capacidade residual do recipiente e buscam manter a razão original entre o tamanho do conjunto de itens de cor mais frequente e o tamanho do seu conjunto complementar, de forma a não deixar o conjunto de itens a serem empacotados tão "desbalanceado".

3. VNS

A VNS é uma meta-heurística proposta em [Mladenović and Hansen 1997] e é composta por um conjunto de vizinhanças $\{\mathcal{N}_1, ..., \mathcal{N}_k\}$, uma função *Shake* e uma função objetivo.

Função objetivo Como objetivo primário buscamos minimizar a quantidade de recipientes utilizados, e como objetivo secundário buscamos a menor ordem lexicográfica no vetor das capacidades residuais dos recipientes utilizados em ordem não-decrescente.

Vizinhanças Dado uma solução viável x, uma vizinhança é o conjunto de soluções viáveis alcançáveis a partir de x ao se realizar um movimento de um certo tipo. Um elemento de tal conjunto é chamado de vizinho de x. Nossa VNS utiliza os seguintes movimentos: *Move Item*, *Swap Items*, *Swap And Move To In* e *Move Two to One*. O movimento *Move Item* move um item de um recipiente para outro. Já o movimento *Swap Items* executa a troca entre dois itens de recipientes diferentes. No movimento *Swap And Move To In* selecionamos três itens i, j, k de recipientes distintos, trocamos i com j, e, por fim, movemos k para o recipiente de i. Ademais, o movimento *Move Two to One* seleciona dois itens de recipientes distintos para mover para um terceiro recipiente.

Shake O *Shake* é um algoritmo que faz movimentos aleatórios, buscando por meio disso escapar de ótimos locais. O nosso *Shake* escolhe executar uma de duas sub-rotinas com igual probabilidade. Na primeira sub-rotina são realizados 20 movimentos entre recipientes distintos, e o movimento pode ser ou *Move Item* ou *Swap Items*, sendo exigida a viabilidade da solução final em ambos os casos. Já na segunda sub-rotina são selecionados dois recipientes, sendo então reempacotados os itens dos mesmos usando a heurística *Best Fit*. Sendo a *Best Fit* análoga à BFD, com a diferença que a ordem de processamento utilizada é uma permutação qualquer dada, no nosso caso, uma permutação aleatória.

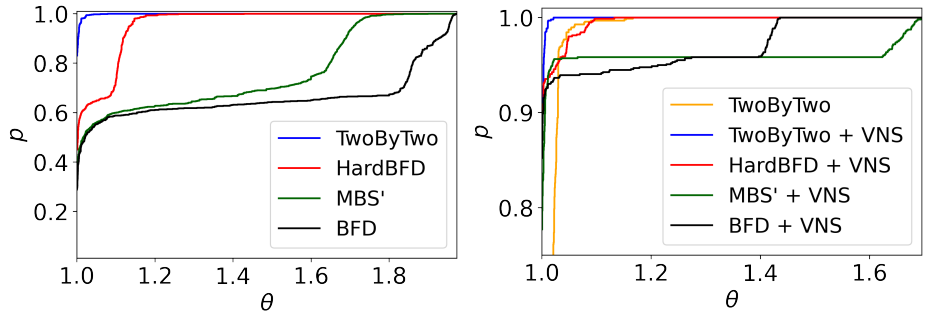
Algoritmo As vizinhanças são nomeadas de N_1 até N_4 na ordem que foram apresentadas. A VNS começa com $k = 1$ e recebe uma solução viável x construída com alguma das heurísticas iniciais apresentadas. A cada etapa fazemos uma busca local em x com a vizinhança $\mathcal{N}_k$, mudando sempre para o melhor (em termos da função objetivo) vizinho da solução atual, até atingirmos uma solução x' que é ótima local. Se x' for melhor que x, então fazemos $x \leftarrow x'$ e $k \leftarrow 1$, do contrário fazemos $k \leftarrow k + 1$. Esse processo é executado até $k = 5$. Neste ponto a rotina *Shake* é executada para gerar uma perturbação aleatória na solução, sendo em sequência iniciado uma nova iteração da VNS com $k = 1$.

4. Resultados Práticos

A seguir mostraremos a comparação entre as heurísticas iniciais com e sem a VNS, em 3 conjuntos de instâncias e com um tempo limite de 60 segundos.

No primeiro conjunto a capacidade do recipiente $L \in \{501, 100001\}$, o tamanho dos itens são inteiros uniformemente escolhidos em $[\frac{L}{10}, \frac{4L}{5}]$, a quantidade de itens $n \in \{102, 201, 402, 801, 1602, 3201, 6402, 12801\}$ e a quantidade de cores $Q \in \{2, 5, 10, 20\}$, sendo a cor de cada item também escolhida uniformemente ao acaso. Além disso, há 5 instâncias para cada possibilidade de escolha desses parâmetros.

O segundo conjunto é análogo ao anterior, com a diferença que o tamanho dos itens são inteiros uniformemente escolhidos em $[1, \frac{L}{4}]$. Por fim, no terceiro conjunto os itens são obtidos particionando, de forma independente, $\frac{n}{3}$ recipientes em 3 partes, com cada parte no intervalo $[\frac{L}{4}, \frac{L}{2}]$, sendo isso feito uniformemente ao caso. Além disso, os dois itens mais pesados de cada recipiente terão a cor 0, e o outro item terá cor 1. O restante dos parâmetros é escolhido semelhante aos outros conjuntos, com a diferença que há 20 instâncias de cada tipo. Note que nesse último conjunto o valor ótimo é $\frac{n}{3}$.



(a) Heurísticas Inicias sem o VNS **(b) Heurísticas Iniciais com o VNS**

Figura 1. Comparação de desempenho das heurísticas iniciais.

Na Figura 1 utilizamos um *performance profile* para comparar o desempenho das heurísticas. Em um primeiro momento, calculamos, para cada instância, qual a razão entre o valor da solução encontrada pelo algoritmo e o valor da melhor solução encontrada entre todos os algoritmos (chamada de referência). Com isso, produzimos um gráfico em que um ponto (p, θ) indica a proporção acumulada p das instâncias que o algoritmo foi no máximo θ vezes pior que a referência.

Na Figura 1a podemos ver que em 50% das instâncias há diferenças significativas na qualidade das soluções encontradas entre as heurísticas, as quais ocorrem exatamente nas instâncias com duas cores. Em particular, note que a BFD em mais de 30% das instâncias utiliza pelo menos 1,8 vezes a quantidade de recipientes gastos pela *Two by Two*, o que nos mostra o quão vantajoso pode ser explorar as restrições de cores.

Por fim, podemos ver na Figura 1b a melhoria advinda da VNS. Nas instâncias com até 6402 itens ela consegue diminuir drasticamente a diferença entre as heurísticas iniciais. Além disso, utilizando a combinação *Two by Two* + VNS, temos que no terceiro conjunto as soluções encontradas com até 6402 itens utilizam no máximo 1 recipiente a mais que a solução ótima, e no máximo 2 recipientes a mais com 12801 itens.

Referências

Böhm, M., Sgall, J., and Veselý, P. (2014). Online colored bin packing. *Proceedings of the 12th International Workshop Approximation and Online Algorithms*, pages 35–46.

Dósa, G. and Epstein, L. (2014). Colorful bin packing. *Proceedings of the 14th Scandinavian Symposium and Workshops on Algorithm Theory*, pages 170–181.

Fleszar, K. and Hindi, K. S. (2002). New heuristics for one-dimensional bin-packing. *Computers & Operations Research*, 29(7):821–839.

Johnson, D., Demers, A., Ullman, J., Garey, M., and Graham, R. (1974). Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM J. Comput.*, 3:299–325.

Loh, K.-H., Golden, B., and Wasil, E. (2008). Solving the one-dimensional bin packing problem with a weight annealing heuristic. *Computers & Operations Research*, 35(7):2283–2291.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100.