Heurísticas para o Problema do Empacotamento Colorido

Candidato: Tiago Domingos Almeida Souza
Orientador: Rafael Crivellari Saliba Schouery

Resumo

O Problema do Empacotamento, em que buscamos particionar um conjunto de itens com pesos de forma que a soma dos pesos dos itens em cada parte seja menor ou igual a um número inteiro dado, minimizando o tamanho do número de partes, é um problema extensamente estudado na computação. Trata-se de um problema NP-difícil, ou seja, é improvável que exista algoritmo polinomial para o mesmo e por isso, são usados algoritmos heurísticos e de aproximação para gerar uma solução próxima da ótima. Uma das generalizações recentes do Problema do Empacotamento é o problema do Empacotamento Colorido, em que cada item além de um peso, tem uma cor e dois itens adjacentes em um recipiente não podem ter a mesma cor. Essa variante tem aplicações como a alocação de programas em uma grade de televisão.

Nesse projeto, estudaremos o Problema do Empacotamento Colorido e algoritmos meta-heurísticos que podem ser aplicados ao problema, como o *VNS*, *BRKGA*, *Busca Tabu* e *GRASP*, além de heurísticas do tipo Any-Fitting para o mesmo.

Como iniciação científica, esse projeto visa também introduzir o candidato na área de pesquisa científica e a complementação de sua formação na área da Algoritmos, bem como a produção de um texto com os resultados estudados que sirva de base para outros pesquisadores da área.

1 Introdução e Justificativa

O ramo da Ciência da Computação que estuda problemas de otimização em um domínio discreto (minimizar ou maximizar uma certa função que recebe um vetor como parâmetro) é chamado de otimização combinatória. Nesta área se concentra muitos problemas com aplicações logísticas ou industriais, como por exemplo o Problema do Empacotamento, que tem aplicações como alocação de programas em blocos do disco rígido, corte de bobinas e empacotamento de caixas em galpões. Nesse projeto estudaremos uma variante do Problema de Empacotamento, o Problema do Empacotamento Colorido, quem tem como uma de suas aplicações a alocação de uma grade de programas televisivos (Böhm et al. [1]). Estudaremos

esse problema por uma ótica heurística e meta-heurística, esta que foi pouco explorada na literatura.

No Problema do Empacotamento (*BPP*, do inglês *Bin Packing Problem*), temos infinitos recipientes com capacidade C e um conjunto $X = \{w_1, \ldots, w_n\}$ de n itens, onde w_i é o peso do i-ésimo item e $w_i \leq C$, queremos particionar X em partes $A_1, \ldots, A_M$ tal que a soma dos pesos dos itens em cada parte seja menor que C e o número de partes seja mínimo. O problema foi proposto em 1939 por L. V. Kantorovich [2] em um trabalho contendo diversos problemas relacionados a demandas industriais da época. O BPP é um caso particular do Problema do Corte de Estoque (*CSP*, do inglês *Cutting Stock Problem*), no qual temos para cada item i uma quantidade d_i de cópias que devem ser empacotadas. No BPP temos que $d_i = 1$ para todo item i.

Nesse projeto, estudaremos uma variante do BPP conhecida como Problema do Empacotamento Colorido (*CBPP*, do inglês *Colored Bin Packing Problem*), onde similarmente ao BPP, temos infinitos recipientes com capacidade C e um conjunto $X = \{(w_1, b_1), \ldots, (w_n, b_n)\}$ de n itens, w_i é o peso e b_i a cor do i-ésimo item, onde $b_i \in Q$ e $w_i \leq C$, sendo Q o conjunto de cores disponíveis. Queremos particionar X em conjuntos ordenados $A_1, \ldots, A_M$ tal que a soma dos elementos em cada parte seja menor que C e seja $A_i = \{(p_1, h_1), \ldots, (p_k, h_k)\}$, temos que $h_i \neq h_{i+1}$ para $1 \leq i < k$, ou seja, nenhum par de elementos adjacentes dentro de uma parte tem a mesma cor. O CBPP foi introduzido por J. Balogh et al. [3] para o caso com $|Q| = 2$ sob o nome de Black and White Bin Packing e posteriormente estendido por Alsarhan et al. [4] para $|Q| \geq 2$.

Tanto o BPP quanto o CBPP são problemas NP-difíceis (mostrado por R. M. Karp [5]), logo, não podem ser resolvidos em tempo polinomial no tamanho da entrada a menos que $P = NP$. Sustenta-se o uso de heurísticas e meta-heurísticas para conseguir soluções boas em tempo computacional razoável.

Esse projeto de iniciação científica visa complementar a formação do candidato, que atualmente cursa o primeiro ano do curso de Engenharia de Computação na Universidade Estadual de Campinas (UNICAMP), foi medalhista de Ouro em 2018 na Olimpíada Brasileira de Informática, e selecionado para compor o time que representará o Brasil na Olimpíada Internacional de Informática (*IOI*) e na Competição Ibero-Americana de Informática e Computação (*CIIC*) em 2019. Como sua primeira iniciação científica, o projeto visa introduzi-lo ao trabalho de pesquisa na área de algoritmos e otimização combinatória.

2 Abordagens de Pesquisa

A seguir, são apresentados conceitos e técnicas de projeto de algoritmos relevantes para este projeto de iniciação científica.

2.1 Heurísticas

Heurísticas são técnicas e estratégias algorítmicas aplicadas para resoluções de problemas específicos, em geral elas retornam em tempo computacional razoável uma solução boa e por isso são usadas para problemas NP-difíceis, como o BPP e CBPP.

2.1.1 Any-Fit

Os algoritmos do tipo AF (do inglês, *Any-Fit*), são algoritmos online para o BPP que podem ser estendidos para o CBPP. Os algoritmos de AF são bastante similares, um algoritmo de AF usa uma estratégia Λ para empacotar os itens. Seja P uma permutação dos itens a serem empacotados, o algoritmo empacotará os itens um a um na ordem da permutação segundo a estratégia de empacotamento Λ. Seja B o conjunto de recipientes abertos, procuramos o recipiente aberto em B que melhor satisfaz Λ e empacotaremos o item nele caso possível. Caso contrário, abrimos um novo recipiente no qual o item será empacotado.

Dizemos que um algoritmo A é uma α-aproximação se A executa em tempo polinomial e, para qualquer instância I, A tem uma solução de valor $A(I)$ que, assumindo um problema de minimização, $A(I) \leq \alpha \cdot OPT$, onde OPT é a solução ótima para a instância I e $\alpha \geq 1$, além disso, dizemos que um algoritmo A é uma α-aproximação assintótica se para qualquer instância I, $A(I) \leq \alpha \cdot OPT + c$, onde c é uma constante. O fator α é chamado de razão de aproximação do algoritmo. Alguns algoritmos do tipo AF são α-aproximações

O Next Fit é um algoritmo do tipo AF. Neste, a estratégia de empacotamento é sempre tentar empacotar um item no último recipiente aberto. Caso não seja possível empacotar o item, seja pela restrição de peso ou pela restrição de cores, no último recipiente, abrimos um novo recipiente no qual o item será colocado. Para o BPP temos que o Next Fit é uma 2-aproximação (Coffman et al. [6]), já para o CBPP temos que a solução do Next Fit no pior caso é, no máximo, três vezes a resposta ótima da instância (Böhm et al. [7]). O Next Fit pode ser implementado em complexidade de tempo $\mathcal{O}(n)$ para o BPP e CBPP.

O First Fit é também um algoritmo do tipo AF. Neste, a estratégia de empacotamento consiste em tentar empacotar um item no primeiro recipiente aberto da lista B tal que o empacotamento não quebre a restrição do problema. Caso não seja possível empacotar o item, seja pela restrição de peso ou pela restrição de cores, abriremos um novo recipiente no qual o item será colocado. Para o BPP, o First Fit é uma 1.7-aproximação (Dósa e Sgall [8]), já para o CBPP o First Fit não admite razão de aproximação (Böhm et al. [7]). O First Fit tem, de acordo com a literatura, a complexidade de tempo no pior caso de $\mathcal{O}(n \log n)$ para o BPP (D. S. Johnson [9]), porém não foi encontrada na literatura limite para a complexidade de tempo no pior caso do First Fit para o CBPP.

O Best Fit e Worst Fit são algoritmos similares do tipo AF. Para o Best Fit a estratégia de empacotamento consiste em empacotar um item no recipiente aberto que menos terá espaço livre após o item ser colocado lá e que não quebre a restrição do problema, já para o Worst Fit, a estratégia de empacotamento consiste em empacotar um item no recipiente aberto que mais terá espaço livre após o item ser colocado lá e que não quebre a restrição

do problema. Em ambos vale que caso houver empate, o recipiente mais à esquerda da lista que satisfaz a estratégia será escolhido. Caso não seja possível empacotar o item em nenhum recipiente aberto, seja pela restrição de peso ou pela restrição de cores, abriremos um novo recipiente no qual o item será colocado. Para o BPP, o Best Fit é uma 1.7-aproximação e o Worst Fit é uma 2-aproximação (Boyar et al. [10]), já para o CBPP, o Best Fit não admite razão de aproximação (Böhm et al. [7]). Tanto o Worst Fit quanto o Best Fit tem, de acordo com a literatura, a complexidade de tempo no pior caso de $\mathcal{O}(n \log n)$ para o BPP (D. S. Johnson [9]), porém não foi encontrada na literatura limite para a complexidade de tempo no pior caso do Best Fit ou Worst Fit para o CBPP.

Os algoritmos do tipo *AFD* (do inglês, *Any Fit Decreasing*) são variações offline dos AF. Neles, ordenamos a permutação P em ordem decrescente do peso dos itens e aplicamos algum algoritmo do tipo AF. No BPP temos que o First Fit Decreasing e o Best Fit Decreasing são uma $\frac{11}{9}$-aproximação assintótica e também que o Next Fit Decreasing é uma 1.2899-aproximação assintótica (Coffman et al. [6]). Não foi encontrado na literatura análise de algoritmos do tipo AFD para o CBPP.

2.2 Meta-heurísticas

Meta-heurísticas são técnicas gerais aplicadas em problemas de otimização combinatória (aqueles que admitem espaço de busca discreto) para achar uma solução suficientemente boa. Tais técnicas não garantem uma solução boa e nem tempo computacional razoável, entretanto são usados em problemas de otimização combinatória por serem um bom método de explorar o espaço de busca.

2.2.1 Busca local

Busca local é um método heurístico de encontrar um ótimo local em um domínio discreto. Temos estados representando soluções do problema e, para cada estado, um valor relacionado gerado a partir de uma função avaliadora f. Temos também uma vizinhança definida a partir de uma função g. O conjunto de estados é percorrido utilizando g usando certa estratégia definida pela meta-heurística aplicada.

A Busca Tabu (proposta por Fred Glover [11]) é uma meta-heurística de busca local que faz uso do conceito de lista tabu, uma lista com transições que estão proibidas. Com isto o algoritmo prossegue para um estado na vizinhança que tenha o maior valor para a função avaliadora e que a transição do estado atual para esse não está na lista tabu. Se o valor desse novo estado for melhor que o estado atual adicionaremos o estado atual a lista tabu. A lista tabu tem tamanho máximo de α (uma constante escolhida), caso este seja atingido removeremos os elementos mais antigos dela.

2.2.2 GRASP

O *GRASP* (do inglês, *Greedy Randomized Adaptative Search Procedure*) é uma meta-heurística semi-gulosa. Seja S um estado e $f(S)$ uma função que avalia o valor de um dado estado. No GRASP temos um estado S_b tal que $f(S_b)$ foi o melhor valor para f encontrado (considerando S_b aleatório no começo). A cada iteração todos os elementos candidatos (no caso do BPP e CBPP, os itens) serão avaliados usando uma função gulosa, e escolheremos os $\beta > |S_b|$ elementos melhores avaliados, formando uma lista restrita de candidatos que será usada para a escolha de $|S|$ valores aleatórios para formar um vetor de estado S_w. Posteriormente achamos o ótimo local (aplicando alguma meta-heurística de busca local) S_n alcançado a partir de S_w, atualizamos S_b para S_n se $f(S_b) \geq f(S_n)$ (assumindo que o problema seja de minimização). Como o GRASP forma gulosamente a lista restrita de candidatos, mas usa de aleatorização para formar o estado a partir dela, ele é considerado semi-guloso. Esta meta-heurística foi inicialmente descrita por T.A. Feo e M.G.C. Resende [12].

2.2.3 VNS

O *VNS* (do inglês, *Variable Neighborhood Search*) é uma meta-heurística relacionada a técnicas de busca local. A estratégia empregada pelo VNS é de explorar vizinhanças cada vez maiores para tentar escapar de um ótimo local. No começo, temos um estado E aleatoriamente gerado, seja B o melhor valor para uma função avaliadora f que encontramos na nossa busca, inicialmente $B = f(E)$, em cada iteração do VNS, seja S um elemento gerado aleatoriamente da vizinhança de E e S' o ótimo local alcançável a partir de S, se o valor de S' for melhor que o de E, atualizaremos E para S' e B para $f(S')$. Usamos de alguma meta-heurística de busca local como rotina do VNS. A quantidade de iterações que o VNS fará depende do quão rápido conseguiremos encontrar o ótimo local. O VNS foi proposto por P. Hansen e N. Mladenović [13].

2.2.4 BRKGA

O *BRKGA* (do inglês, *Biased Random Key Genetic Algorithm*) é uma meta-heurística da subclasse de algoritmos genéticos. Tais algoritmos são inspirados na seleção natural, onde temos uma população de indivíduos, conhecidos como cromossomos (que são os estados do nosso espaço de busca), sendo cada cromossomo representado por uma lista de genes (que representa suas características). Cada membro da população evolui usando de conceitos presentes na evolução, como a seleção, que mantêm uma porcentagem da população que é mais apta (que apresenta maior qualidade em uma função avaliadora), o *crossing-over*, que decompõe dois ou mais indivíduos distintos da população e recombina eles (mistura de estados) e a mutação, que muda aleatoriamente um indivíduo da população (alteração do estado).

Um BRKGA é um algoritmo genético para problemas envolvendo sequências, definimos uma *Random Key* como um número real aleatório p entre 0 e 1, em um BRKGA, cada gene de um individuo é representado por uma codificação $H = ((p_1) \dots (p_n))$, onde p_i é uma Random Key, definimos como $D(x)$ um decodificador que recebe um cromossomo e devolve um valor que corresponde ao valor da solução do problema codificada pelo cromossomo.

Temos uma nova operação evolutiva chamada cruzamento. No cruzamento escolhemos dois indivíduos a e b que serão os pais e geramos um novo individuo c que será o filho. Para cada gene de c definiremos de qual parente ele foi herdado com o lançamento de uma moeda viciada (considerando que a probabilidade de a ser escolhido é α e que $\alpha > 0.5$). No BRKGA não existe mutação, entretanto temos mutantes, indivíduos totalmente randômicos que foram adicionados a população. Para cada individuo da população, criaremos dois conjuntos, o da elite, que tem os indivíduos com a decodificação de melhor valor para uma função avaliadora e o da não-elite, que contêm o restante dos indivíduos. Para uma nova iteração do BRKGA, a população é gerada usando de toda a elite anterior, alguns elementos aleatórios da não-elite anterior e de cruzamentos onde os pais são um membro a da elite e um membro b do restante. O algoritmo foi proposto por Mauricio G. C. Resende e José Fernando Gonçalves [14]

3 Objetivos

O objetivo do projeto é o estudo do problema do Empacotamento Colorido e problemas correlatos, com foco em algoritmos heurísticos. Todos os estudos publicados sobre o CBPP são com foco em algoritmos online e de aproximação. Com esse projeto, pretendemos estudar as heurísticas e como elas podem ser aplicadas ao CBPP, além de seus desempenhos e tempos de execução. As demonstrações e métodos estudados serão reunidas e organizadas de forma clara e didática em um relatório técnico ao fim do projeto.

Além disso, como iniciação científica o projeto introduzirá o candidato a técnicas e métodos de pesquisa, além de complementar sua formação na área de Ciência da Computação, aprofundando seu conhecimento na área de otimização combinatória, heurísticas e técnicas de projeto e análise de algoritmos.

4 Plano de Trabalho e Cronograma

Os três primeiros meses do projeto serão dedicados ao estudo da bibliografia existente e estudo sobre algoritmos do tipo Any-Fit para o CBPP, suas implementações, análise da complexidade dos mesmos e pesquisa sobre um limite para a complexidade de tais algoritmos para o Problema do Empacotamento Colorido.

Dedicaremos após isso dois meses para cada meta-heurística (Busca Tabu, GRASP, VNS,

BRKGA). Para cada uma das delas, haverá um estudo sobre a mesma, um ciclo de desenvolvimento e teste de implementações e uma fase de avaliação final usando de instâncias da literatura.

O último mês do projeto será dedicado a compilação dos resultados da pesquisa e produção do relatório final.

Atividades \ Mês	1º	2º	3º	4º	5º	6º	7º	8º	9º	10º	11º	12º
Revisão bibliográfica e algoritmos clássicos	•	•	•									
Busca Tabu				•	•							
GRASP						•	•					
VNS								•	•			
BRKGA										•	•	
Relatório final												•
Escrita do relatório científico	•	•	•	•	•	•	•	•	•	•	•	•

5 Materiais e Métodos

Durante o projeto, o candidato estudará artigos importantes sobre o Problema do Empacotamento e sobre o Problema do Empacotamento Colorido, sendo o acesso a esses fornecidos gratuitamente pela Unicamp, além de livros sobre heurísticas relevantes ao Problema do Empacotamento, disponibilizados pela biblioteca do Instituto de Matemática, Estatística e Computação Cientifica.

O candidato também implementará heurísticas relevantes ao projeto, usara de instâncias da literatura e de ferramentas como a biblioteca Pandas e matplotlib de Python para comparar a eficiência de suas implementações com as presentes na literatura e a biblioteca *BPP-lib* de C++ que contêm diversas implementações de códigos relevantes ao BPP, além de instâncias do BPP para comparação de resultados.

Além disso, serão realizadas reuniões quinzenais entre o candidato, o coorientador e o orientador para averiguar o andamento do projeto e discutir o conteúdo aprendido.

6 Forma de análise dos resultados

Haverá reuniões quinzenais entre o candidato e o orientador para averiguar o andamento do projeto e o entendimento do conteúdo estudado. O candidato apresentará o conteúdo estudado formalmente e os resultados obtidos para o orientador.

Ao fim do projeto os resultados da pesquisa serão compilados em um relatório técnico final, mostrando que o candidato aprendeu o conteúdo e é capaz de apresentá-lo formalmente, além disso, o candidato também fará um relatório técnico para o Instituto de Computação da UNICAMP descrevendo o que foi utilizado e o que foi aprendido durante o projeto.

Referências

[1] Martin Böhm, Jiří Sgall, and Pavel Veselý. Online colored bin packing. 2014.

[2] L. V. Kantorovich. Mathematical methods of organizing and planning production. *Manage. Sci.*, 6(4):366–422, July 1960.

[3] J. Balogh, J. Békési, G. Dosa, H. Kellerer, and Z. Tuza. Black and white bin packing. lecture notes in computer science, 131–144. 2013.

[4] Hamza Alsarhan, Davin Chia, Ananya Christman, Shannia Fu, and Yanfeng Jin. Bin packing with multiple colors. 2015.

[5] Ronald V. Book. Karp, Richard m. . reducibility among combinatorial problems. complexity of computer computations, 1972, pp. 85–103. *J. symb. log.*, 40(04):618–619, December 1975.

[6] E. G. Cofman, Jr. M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing: A survey. 2013.

[7] Martin Böhm, György Dósa, Leah Epstein, Jiří Sgall, and Pavel Veselý. Colored bin packing: Online algorithms and lower bounds. *Algorithmica*, 80(1):155–184, November 2016.

[8] György Dósa and Jiří Sgall. First fit bin packing: A tight analysis. 2013.

[9] David S. Johnson. Fast algorithms for bin packing. *Journal of Computer and System Sciences*, 8(3):272–314, June 1974.

[10] Joan Boyar and Lene M. Favrholdt. The relative worst order ratio for online algorithms. *TALG*, 3(2):12–es, May 2007.

[11] Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, January 1986.

[12] Thomas A Feo and Mauricio G.C Resende. A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8(2):67–71, April 1989.

[13] P. Hansen and N. Mladenović. An introduction to variable neighborhood search. 1998.

[14] José Fernando Gonçalves and Mauricio G. C. Resende. Biased random-key genetic algorithms for combinatorial optimization. 2010.