57º SBPO
SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
Gramado • RS
DE 5 A 9 DE OUTUBRO DE 2025

SELEÇÃO INTELIGENTE DE HEURÍSTICAS PARA O PROBLEMA DO EMPACOTAMENTO COM BINS VARIÁVEIS E CONFLITOS: UMA ABORDAGEM BASEADA EM ÁRVORE DE DECISÃO

Filipe de Medeiros Santos, Gilberto Farias de Sousa Filho,
Lucídio dos Anjos Formiga Cabral
Centro de Informática – Universidade Federal da Paraíba (UFPB)
João Pessoa – PB – Brasil
filipe.medeiros@academico.ufpb.br, {gilberto,lucidio}@ci.ufpb.br

RESUMO

Neste artigo, analisamos o Problema do Empacotamento com *Bins* Variáveis e Conflitos (VSBPPC). Esse problema consiste em alocar itens de tamanhos específicos em *bins* com capacidades e custos variados, respeitando restrições de conflito entre pares de itens, com o objetivo de minimizar o custo total dos *bins* utilizados. Dada sua aplicabilidade prática em contextos como alocação de recursos, planejamento de produção e distribuição logística, e motivados pela elevada complexidade computacional das abordagens existentes, apresentamos aprimoramentos ao algoritmo de busca em grandes vizinhanças (LNSA), que incluem variações na heurística construtiva e a integração de modelos de aprendizagem de máquina para orientar as decisões do algoritmo. A abordagem proposta foi comparada ao LNSA presente na literatura, demonstrando uma melhoria média de 0,085%, enquanto reduziu o tempo de execução para aproximadamente um terço do necessário pelo algoritmo original.

PALAVRAS CHAVE. Problema do empacotamento, Algoritmo de busca em vizinhança larga, Algoritmos heurísticos, Árvore de decisão.

Tópicos. Otimização combinatória, Meta-heurística, Aprendizagem de máquina.

ABSTRACT

In this article, we analyze the Variable-Sized Bin Packing Problem with Conflicts (VSBPPC). This problem involves allocating items of specific sizes into bins with varying capacities and costs, while respecting conflict constraints between item pairs, with the goal of minimizing the total cost of the bins used. Given its practical relevance in contexts such as resource allocation, production planning, and logistics distribution, and motivated by the high computational complexity of existing approaches, we present enhancements to the Large Neighborhood Search Algorithm (LNSA), including variations in the constructive heuristic and the integration of machine learning models to guide the algorithm's decisions. The proposed approach was compared to the LNSA described in the literature, demonstrating an average performance improvement of 0.085% while reducing execution time to approximately one-third of that required by the original algorithm.

KEYWORDS. Bin packing problem. Large neighborhood search algorithm. Heuristic algorithms. Decision tree.

Paper topics. Combinatorial optimization. Metaheuristic. Machine learning.

1. Introdução

O Problema do Empacotamento (BPP), conforme apresentado por Coffman et al. [1978], é um problema clássico na área de otimização combinatória que consiste em alocar um conjunto de tarefas $T = \{t_1, \ldots, t_s\}$, cada uma com tamanho l_t, em um conjunto N de *bins* idênticos, cada um com capacidade fixa. O objetivo principal é minimizar a quantidade de *bins* utilizados, assegurando que a soma dos tamanhos das tarefas em cada *bin* não ultrapasse a sua capacidade.

No Problema do Empacotamento com Conflitos (BPPC), introduz-se um grafo de conflitos $G = (V, E)$, em que cada vértice $v_t \in V$ representa uma tarefa $t \in T$ e as arestas $(i, j) \in E$ indicam que as tarefas i e j não podem ser alocadas no mesmo *bin* (Maiza e Radjef [2011]). Já no Problema do Empacotamento com *Bins* Variáveis (VSBPP), considera-se um conjunto de tipos de *bins* $B = \{1, 2, \ldots, m\}$, sendo C_k a capacidade do *bin* do tipo $k \in B$. Nesse caso, o objetivo é minimizar o custo total associado ao uso desses *bins* (Friesen e Langston [1986]).

Este trabalho aborda o Problema do Empacotamento com *Bins* Variáveis e Conflitos (VSBPPC), uma versão generalizada que une as complexidades do BPPC e VSBPP. Nessa variante, além de selecionar adequadamente os *bins* para minimizar o custo total, é necessário alocar cuidadosamente as tarefas conflitantes em *bins* separados. Para modelar o problema via programação linear inteira, define-se as variáveis de decisão binária $y_{pk} = 1$ se o *bin* p do tipo k é utilizado na solução e $x_{tp} = 1$ se a tarefa t é alocada no *bin* p. O modelo a seguir foi proposto por Ekici [2023]:

$$\text{Min} \quad VSBPPC = \sum_{p \in N} \sum_{k \in B} C_k \cdot y_{pk} \tag{1}$$

$$\text{s.t.} \quad \sum_{k \in B} y_{pk} \leq 1, \qquad \forall p \in N, \tag{2}$$

$$\sum_{p \in N} x_{tp} = 1, \qquad \forall t \in T, \tag{3}$$

$$\sum_{t \in T} l_t \cdot x_{tp} \leq \sum_{k \in B} C_k \cdot y_{pk}, \qquad \forall p \in N, \tag{4}$$

$$x_{ip} + x_{jp} \leq \sum_{k \in B} y_{pk}, \qquad \forall (i, j) \in E, \forall p \in N. \tag{5}$$

A função objetivo (1) busca minimizar o custo total associado à utilização dos *bins*. As restrições em (2) garantem que, no máximo, um tipo de *bin* seja atribuído a cada *bin* p e as restrições em (3) garantem que cada tarefa $t \in T$ seja alocada exatamente a um *bin*. Já as restrições (4) asseguram que a capacidade de cada *bin* p não seja excedida, enquanto as restrições (5) impedem que tarefas conflitantes sejam alocadas no mesmo *bin*.

Este problema revela-se mais abrangente, permitindo modelar diversos cenários reais de grande relevância. Uma aplicação prática, ilustrada na Figura 1, refere-se ao seu emprego no contexto de sistemas de armazenamento distribuído. Considere o cenário em que múltiplos arquivos precisam ser alocados em uma rede de computadores com o objetivo de minimizar o custo total do armazenamento, levando-se em conta que:

1. Dispõe-se de uma rede heterogênea de computadores, cada um com diferentes capacidades de armazenamento e especificações;

2. Implementa-se uma estratégia de tolerância a falhas, o que envolve a duplicação de arquivos críticos e a criação de conflitos entre as cópias, que precisam ser armazenadas em máquinas separadas para garantir a confiabilidade do sistema.

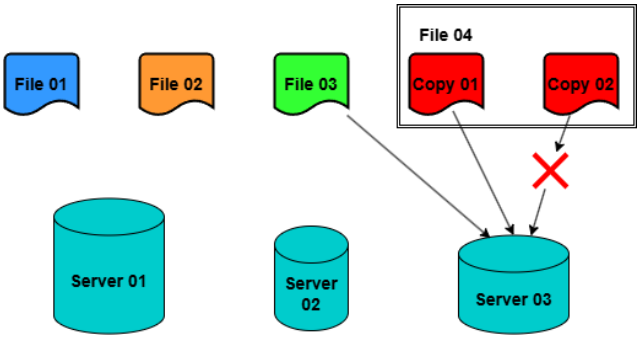

Figura 1: Aplicação prática do VSBPPC. Fonte: Acervo do autor.

Este trabalho busca superar as limitações dos métodos existentes para resolver o VSBPPC, especialmente a meta-heurística LNSA proposta por Ekici [2023], cuja eficiência decai conforme a complexidade do problema aumenta. Assim, propõem-se melhorias que aprimoram a qualidade das soluções e reduzem o tempo computacional. Os principais objetivos são:

- Desenvolver novas abordagens, incluindo variações na decisão gulosa presente na literatura e uma heurística construtiva inspirada no método *best fit decreasing* (Johnson [1974]);
- Integrar modelos de aprendizagem de máquina, como árvores de decisão, ao algoritmo de otimização para orientar a escolha das heurísticas construtivas;
- Melhorar a qualidade das soluções obtidas, reduzindo o tempo de execução.

Este artigo está organizado da seguinte forma: primeiramente, são revisados os trabalhos relacionados. Em seguida, são propostas variações heurísticas e uma nova heurística construtiva, acompanhadas da descrição do treinamento de um modelo de árvore de decisão para orientar sua seleção. Por fim, apresentam-se os resultados computacionais comparando as abordagens propostas com os métodos de referência, seguidos pelas conclusões e sugestões para trabalhos futuros.

2. Trabalhos relacionados

Conforme destacado por Coffman et al. [1978], o problema do empacotamento pertence à classe dos problemas NP-difíceis. Essa classificação implica que, em casos gerais, a resolução por algoritmos exatos é computacionalmente inviável, motivando o estudo e desenvolvimento de técnicas heurísticas que permitam obter soluções próximas a ótima de maneira eficiente.

Neste contexto, Johnson [1974] apresenta e analisa heurísticas clássicas, como o *First Fit Decreasing* (FFD) e o *Best Fit Decreasing* (BFD). O autor demonstra, por meio de análises teóricas, que esses métodos, ao listar os itens em ordem não crescente de tamanho e aplicar regras simples de alocação, garantem um desempenho assintótico no pior caso limitado a 1,22 vezes o valor ótimo.

Ao abordar o VSBPP, Friesen e Langston [1986] argumentam que essa variante também é NP-difícil, por ser redutível ao BPP. Diante disso, o estudo propõe três algoritmos aproximativos, inspirados nas heurísticas *Next Fit* (NF) e FFD, que apresentam garantias teóricas de desempenho no pior caso, com razões assintóticas de 2, 3/2 e 4/3 em relação à solução ótima.

No contexto do BPPC, Maiza e Radjef [2011] introduzem o grafo de conflitos estendido $G_{\text{ext}}(V, E_{\text{ext}})$, uma generalização do grafo de conflitos original. Seja C_k a capacidade máxima do maior tipo de *bin* e l_i e l_j os tamanhos dos itens i e j, o conjunto de arestas E_{ext} é definido por $E_{\text{ext}} = E \cup \{(i, j) : i, j \in V \text{ e } l_i + l_j > C_k\}$. Os autores propõem seis heurísticas que combinam técnicas de empacotamento como FFD, BFD e MBS, com estratégias de busca de cliques maximais.

Em um contexto mais próximo ao problema abordado neste trabalho, Epstein et al. [2011] introduzem o *Online Variable-Sized Bin Packing with Conflicts*, no qual os itens e o grafo de conflitos são previamente conhecidos, enquanto *bins* de tamanhos variáveis chegam de forma sequencial. Entre as estratégias, destacam-se o método *Color-and-Pack*, que combina técnicas de coloração de grafos e empacotamento guloso, e adaptações de algoritmos clássicos, entre eles o FFD.

Dentre os avanços recentes, Ekici [2023] apresenta uma abordagem inovadora para o VSBPPC, propondo novos limites inferiores e um algoritmo de busca em grandes vizinhanças (LNSA, do inglês *Large Neighborhood Search Algorithm*). O autor também descreve o limite inferior, denominado LB^{MRS}, originalmente proposto por Maiza *et al.* (2016), que foi adotado para avaliar os resultados das abordagens propostas e comparar os métodos ao longo do artigo.

Em contraste com as abordagens existentes na literatura, que focam em adaptações de heurísticas clássicas (Friesen e Langston [1986]) e soluções voltadas para subclasses específicas do problema (Epstein et al. [2011]), este trabalho propõe aprimoramentos à meta-heurística LNSA aplicada ao VSBPPC (Ekici [2023]), incorporando um algoritmo de classificação para selecionar a heurística construtiva mais adequada com base nas características da instância.

3. Heurísticas para o VSBPPC

Nesta seção, apresentamos aprimoramentos propostos para o LNSA adaptado ao VSBPPC por Ekici [2023], incluindo uma variação na decisão gulosa e uma nova heurística construtiva.

3.1. Avaliação da solução

Seja $\mathcal{P} = \{1, 2, \ldots, P\}$ o conjunto de índices dos *bins* abertos, $\overline{V}_p$ o conjunto de tarefas que são alocados no *bin* $p \in \mathcal{P}$, e k_p o tipo do *bin* p, cujo valor é definido por $k_p = \arg\min_{k \in B}\{C_k \mid \sum_{j \in \overline{V}_p} l_j \leq C_k\}$. O conjunto de tarefas já alocados é representado por $\overline{V} = \bigcup_{p \in \mathcal{P}} \overline{V}_p$. Sendo assim, uma solução viável $s =< (\overline{V}_1, k_1), \ldots, (\overline{V}_P, k_P) >$ será avaliada pela função

$$f(s) = \sum_{p \in \mathcal{P}} C_{k_p}.$$

3.2. Heurística construtiva HC_1

A heurística construtiva gulosa proposta por Ekici [2023] é guiada pela função A_{tp} que avalia a alocação de uma tarefa $t \notin \overline{V}$ em um *bin* p. A cada iteração, seleciona-se a tarefa t^* e o *bin* p^* que minimizam o custo A_{tp}, ou seja, $(t^*, p^*) = argmin_{t \in V \setminus \overline{V}, p \in \{1,..,P+1\}}\{A_{tp}\}$, respeitando as restrições de conflito, ou seja, $\{j \in \overline{V}_{p^*} \mid (t^*, j) \in E\} = \emptyset$. Este procedimento atualiza $\overline{V}_{p^*} = \overline{V}_{p^*} \cup \{t^*\}$ e continua até que $\overline{V} = V$. A chave desta heurística está na função gulosa

$$A_{tp} = \begin{cases} \min_{k \in B}\{\frac{C_k - C_{k_p}}{l_t} \mid l_t + \sum_{j \in \overline{V}_p} l_j \leq C_k\}, & \text{se } p \in \mathcal{P} \\ \min_{k \in B}\{\frac{C_k}{l_t} \mid l_t \leq C_k\}, & \text{se } p = P + 1. \end{cases}$$

Quando $p = P + 1$, A_{tp} mede o custo de adicionar a tarefa t em um novo bin com o tipo $k \in B$ de menor custo, e quando $p \in \mathcal{P}$, A_{tp} avalia o menor custo de adicionar t em um *bin* p já aberto.

Quando a tarefa t é adicionada em um *bin* $p \in \mathcal{P}$ sem alterar seu tipo k_p, ou seja, $l_t + \sum_{j \in \overline{V}_p} l_j \leq C_{k_p}$, então $A_{tp} = 0$. Como isto ocorre muitas vezes durante a construção de uma solução, propõe-se um critério de desempate, ou seja, sempre que $A_{tp} = 0$ então recalculamos

$$A_{tp} = \lambda \cdot (1 - \frac{l_t + \sum_{j \in \overline{V}_p} l_j}{C_{k_p}}), \tag{6}$$

com $\lambda = 10^{-3}$, valorizando assim a tarefa t que consiga ocupar mais do espaço livre no bin p.

Para ilustrar o critério de desempate proposto, considere a instância do problema representada na Figura 2, com dois tipos de *bins*: um com capacidade de 100 e outro com capacidade de 120. Sem o critério de desempate, a heurística HC alocaria inicialmente a tarefa 0 no *bin* 0, o que inviabilizaria a alocação posterior da tarefa 1 sem causar um aumento no valor da função objetivo.

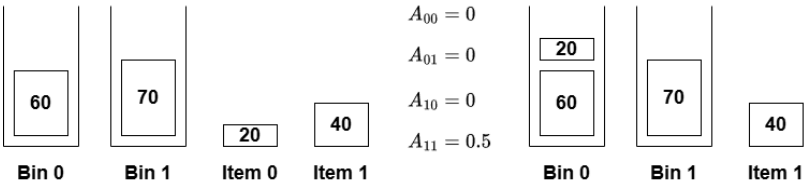

Figura 2: Escolha gulosa da heurística original. Fonte: Acervo do autor.

Ao aplicar o critério de desempate, observa-se que a tarefa 1 é alocada no *bin* 0, preenchendo-o completamente. Dessa forma, na iteração seguinte, torna-se viável alocar a tarefa 0 no *bin* 1 sem provocar um incremento no valor da função objetivo, conforme ilustrado na Figura 3.

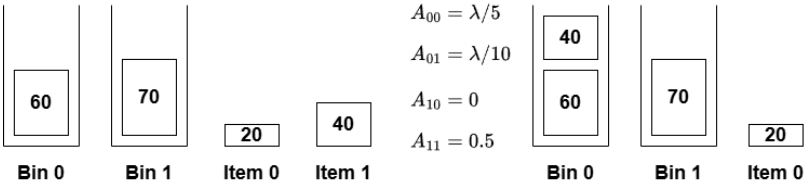

Figura 3: Escolha gulosa da heurística HC1. Fonte: Acervo do autor.

3.3. Heurística construtiva HC_2

Neste trabalho, propomos uma segunda heurística construtiva, que segue o mesmo processo iterativo da heurística HC_1, mas redefine o custo A_{tp} inspirada no método *Best Fit* descrito por Johnson [1974]. Para $p \in \{1, \ldots, P+1\}$, o novo custo A_{tp} é definido como:

$$A_{tp} = \min_{k \in B} \{ \frac{C_k}{l_t + \sum_{j \in \overline{V}_p} l_j} \mid l_t + \sum_{j \in \overline{V}_p} l_j \leq C_k \}. \tag{7}$$

Pela equação (7) o custo A_{tp} atinge seu valor mínimo igual a 1 quando a inserção da tarefa t preencher completamente o espaço livre do *bin* p com tipo k, valorizando assim esta escolha.

3.4. Meta-heurística *large neighborhood search algorithm* (LNSA)

Ekici [2023] propõe uma meta-heurística LNSA para o VSBPPC. O algoritmo começa com a construção de uma solução inicial viável e, a cada iteração, destrói parte da solução e realoca as tarefas dos *bins* eliminados, gerando uma nova solução viável. Após a construção da nova solução, aplica-se um procedimento de busca local dividido em duas fases para refiná-la. A cada iteração, o algoritmo avalia a solução gerada e armazena a melhor encontrada até o momento.

Na primeira fase da busca local, aplica-se um movimento de vizinhança que transfere uma tarefa de seu *bin* atual para outro, respeitando as restrições de conflito. Após a transferência atualiza-se os tipos dos *bins* associados, determinando o de menor custo que acomode as tarefas. Se não houver melhora na solução mas a maior relação entre espaço preenchido e custo entre os *bins* aumentar, este movimento é aceito com a expectativa de melhorias em iterações seguintes.

Na segunda fase da busca local, procura-se trocar tarefas entre dois *bins*, respeitando as restrições de conflito e adotando os mesmos critérios de atualização e aceitação da primeira fase. As duas fases são aplicadas sobre a solução atual de forma sequencial até que ela não possa mais ser melhorada. Propomos a adoção das heurísticas construtivas HC_1 e HC_2, tanto na fase de construção da solução inicial quanto na reparação da solução destruída, o que resulta em duas variantes da meta-heurística LNSA, denominadas $LNSA - HC_1$ e $LNSA - HC_2$.

4. Árvore prescritiva para o VSBPPC

O desempenho das heurísticas HC_1 e HC_2, integradas à meta-heurística LNSA, varia de acordo com as características das instâncias do problema. Para lidar com essa variabilidade, inspirados na abordagem proposta por Vilas Boas et al. [2021], investigamos o uso de árvores de decisão para recomendar a heurística mais promissora com base nas propriedades da instância.

Nesta seção, apresentamos uma definição formal da adoção de árvores prescritivas para otimização. Em seguida, descrevemos o treinamento dessas árvores e introduzimos um modelo que aplica iterativamente as predições da árvore ao longo da execução do algoritmo de otimização.

4.1. Árvore prescritiva para otimização

Considere $X = \{x_1, \ldots, x_n\}$ o conjunto de instâncias de entrada para um problema de otimização P. Seja $\Phi(x) : \mathcal{X} \to \mathbb{R}^d$ uma função que computa d estatísticas descritivas para cada instância $x \in X$, e $\mathcal{A} = \{a_1, \ldots, a_m\}$ o conjunto de m algoritmos aproximativos para P. Para cada algoritmo $a \in \mathcal{A}$, define-se $f_a(x) : \mathcal{X} \to \mathbb{R}$ como a função objetivo associada à melhor solução encontrada pelo algoritmo a para a instância x.

Seja $\mathcal{D} = \mathbb{R}^d$ o espaço que contém os pontos $\{\Phi(x) \mid x \in X\}$. A metodologia proposta neste trabalho particiona $\mathcal{D}$ em L subconjuntos disjuntos $(\mathcal{L}_1, \ldots, \mathcal{L}_L)$. Dentro de cada parte l, as previsões têm como objetivo minimizar, de forma conjunta, a função f_a associada ao algoritmo $a \in \mathcal{A}$, designado para otimizar as soluções das instâncias $x \in X$ que satisfazem $\Phi(x) \in \mathcal{L}_l$.

O particionamento $(\mathcal{L}_1, \ldots, \mathcal{L}_L)$ baseia-se em classes de hipóteses construídas por modelos de árvore de decisão, denotados como $\Psi_\theta^{(tree)}$. Estas hipóteses são construídas com um conjunto de hiper-parâmetros θ e define "regras de divisão", caracterizadas como uma sequência hierárquica de funções disjuntivas $h_\phi : \mathcal{D} \to \mathbb{R}$. Uma representação formal dessa estrutura é definida como

$$\min_{(\mathcal{L}_1,..,\mathcal{L}_L)\in\Psi_\theta^{(tree)}} \left\{ \sum_{l=1}^{L} \left(\min_{a\in\mathcal{A}} \left\{ \sum_{x\in X|\Phi(x)\in\mathcal{L}_l} f_a(x) \right\} \right) \right\}.$$

4.2. Heurística para árvore prescritiva

Propomos uma metodologia de duas fases para o treinamento do modelo. Na primeira fase construímos uma base de dados descritiva D das instâncias de X que servirá, na segunda fase, para treinar e validar a hipótese $T \in \Psi_\theta^{(tree)}$ construída a partir de um modelo de árvore de decisão.

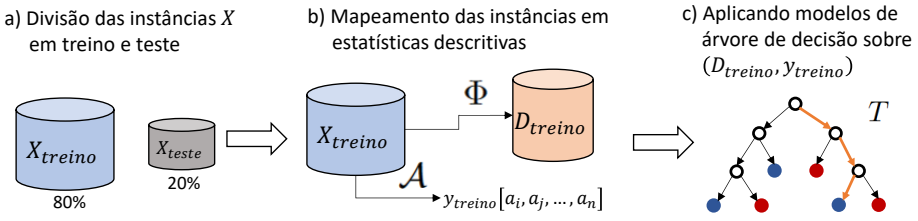

Figura 4: Metodologia da heurística de construção da árvore prescritiva para algoritmos aproximativos.
Fonte: Acervo do autor.

A Figura 4 ilustra a metodologia proposta. As instâncias de X são divididas aleatoriamente em X_{treino}, com 80% das instâncias, e X_{teste}, com 20% (Figura 4(a)). Em seguida, definimos $D_{treino} = \{\Phi(x) \mid x \in X_{treino}\}$, $D_{teste} = \{\Phi(x) \mid x \in X_{teste}\}$ e o vetor de rótulos y, no qual, para cada amostra $d_i \in D$, associada à instância $x_i \in X$, temos que $y_i = argmin_{a \in \mathcal{A}}\{f_a(x_i)\}$. Por fim, a base (D_{treino}, y_{treino}) é fornecida para o modelo de árvore de decisão (Figura 4(c)).

O principal hiper-parâmetro θ de definição da complexidade de uma árvore é seu número de ramificações. Quanto mais ramificações a árvore tem, maior é a sua capacidade de aprender detalhes sobre os dados, o que pode levar ao *overfitting*. Sendo assim, utilizamos o algoritmo de regularização *Minimal Cost-Complexity Pruning* (Breiman et al. [1984]), que realiza uma busca exaustiva pelo valor do parâmetro α que minimiza a função $R_\alpha(T_\alpha) = R(T_\alpha) + \alpha \cdot |T_\alpha|$, procurando ajustar T_α aos dados, equilibrando os erros de classificação $R(T_\alpha)$ e o tamanho $\alpha \cdot |T_\alpha|$ da árvore.

4.3. Aplicação iterativa da árvore prescritiva

Propõe-se uma abordagem alternativa que aplica iterativamente a árvore prescritiva ao LNSA, com o objetivo de minimizar o impacto dos erros de classificação da árvore. Essa abordagem explora as probabilidades associadas aos nós folha, evitando restringir as decisões a predições discretas. Assim, promove uma adaptação dinâmica ao longo da otimização, permitindo ajustes mais refinados ao comportamento das instâncias e incentivando a diversificação das soluções.

Na abordagem tradicional, ilustrada na Figura 5(a), a árvore é aplicada apenas no início da execução do LNSA, com o propósito de selecionar a heurística mais adequada para cada região do espaço $\mathcal{D}$. Essa estratégia determina diretamente qual meta-heurística deve ser utilizada e, para tal, utilizamos tanto a árvore não regularizada T quanto a árvore regularizada T_α.

Adicionalmente, propomos uma aplicação iterativa da árvore regularizada, que denominamos T_α^i, ao longo da execução do LNSA, conforme ilustrado na Figura 5(b). Nessa abordagem, no início da execução, a árvore fornece, a partir do nó folha correspondente às características da instância a ser otimizada, as probabilidades associadas a cada heurística. Em cada iteração, a heurística corrente é selecionada com base nesses valores, por meio de um teste probabilístico.

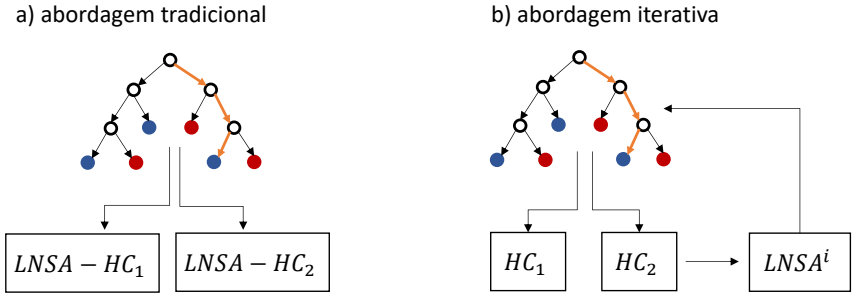

Figura 5: Abordagens para utilização da árvore. Fonte: Acervo do autor.

5. Resultados computacionais

Nesta seção, apresentamos os resultados dos experimentos computacionais para avaliar as abordagens propostas para o VSBPPC. Começamos com a descrição das métricas de avaliação utilizadas, seguida pela análise do desempenho das heurísticas para o LNSA e, por fim, discutimos o uso de árvores prescritivas na seleção da heurística mais adequada.

O limite inferior foi implementado em Python, utilizando a biblioteca python-mip e o solver CBC, enquanto os algoritmos foram desenvolvidos em C++. Todas as implementações foram realizadas e testadas em um notebook de 64 bits, com processador Intel Core i5 e 8 GB de RAM.

5.1. Métricas de avaliação

Para avaliar a eficácia das abordagens propostas, utilizamos duas métricas principais. A primeira avalia a proximidade da solução obtida em relação ao limite inferior. Seja f_a o valor da solução obtida por determinado algoritmo e $f_{LB^{MRS}}$ o limite inferior, o *gap* é calculado por $Gap_a = \frac{f_a - f_{LB^{MRS}}}{f_a} \times 100\%$. A segunda métrica mede o percentual de melhoria entre os $gaps$ das soluções propostas em relação ao LNSA, sendo calculada pela fórmula $\Delta_a = Gap_{LNSA} - Gap_a$.

Para avaliar o desempenho dos modelos de aprendizagem de máquina, utilizamos as métricas E_{in}, E_{out}, precisão e revocação. A taxa de erro no treinamento (E_{in}) mede a proporção de previsões incorretas nos dados utilizados para o treinamento. Por sua vez, a taxa de erro no teste (E_{out}) reflete a proporção de previsões incorretas em dados não vistos durante o treinamento.

A precisão avalia a proporção de exemplos classificados como positivos que realmente pertencem à classe positiva. Considerando VP (verdadeiros positivos) e FP (falsos positivos), é definida como Precisão $= \frac{\text{VP}}{\text{VP+FP}} \times 100\%$. A revocação, por outro lado, avalia a proporção de exemplos positivos corretamente identificados pelo modelo em relação ao total de exemplos positivos existentes. Considerando FN (falsos negativos), sua fórmula é dada por Revocação $= \frac{\text{VP}}{\text{VP+FN}} \times 100\%$.

5.2. Meta-heurísticas LNSA para o VSBPPC

Os experimentos apresentados avaliam o algoritmo original LNSA e as abordagens propostas, $LNSA-HC_1$ e $LNSA-HC_2$. Os resultados incluem os *gaps* percentuais, o tempo médio de execução dos algoritmos e os percentuais de melhoria alcançados pelas heurísticas propostas. A análise foi conduzida utilizando o conjunto X_{teste}, composto por 144 instâncias, e considerou as variáveis: quantidade de itens $|V|$, tamanho dos itens s e densidade do grafo de conflitos d.

Conforme apresentado na Tabela 1, os valores médios indicam que as heurísticas propostas superaram o LNSA, com melhorias mais expressivas à medida que o número de itens aumenta. Entre as abordagens avaliadas, o algoritmo $LNSA - HC_1$ apresentou o melhor resultado no conjunto de instâncias com 200 itens, embora com uma melhoria média de apenas 0,11%.

Tabela 1: Comparação entre as heurísticas construtivas para diferentes quantidades de itens.

	$LNSA$		$LNSA-HC_1$			$LNSA-HC_2$		
$\|V\|$	$Gap(\%)$	$\bar{t}(s)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$
100	4,36	0,92	**4,32**	0,39	0,04	4,34	**0,19**	0,02
200	3,10	6,61	**2,99**	3,79	0,11	3,02	**2,61**	0,08

Conforme apresentado na Tabela 2, as heurísticas propostas demonstram melhor desempenho à medida que a variabilidade nos tamanhos dos itens aumenta. Para itens com tamanho mínimo de 1 unidade, o $LNSA - HC_1$ obteve melhoria média de 0,16%. Em contrapartida, para itens com tamanho mínimo de 50 unidades, observa-se uma leve deterioração no desempenho.

Tabela 2: Comparação entre as heurísticas construtivas para diferentes tamanhos de itens.

	$LNSA$		$LNSA-HC_1$			$LNSA-HC_2$		
s	$Gap(\%)$	$\bar{t}(s)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$
1 - 100	4,60	4,71	**4,44**	2,52	0,16	4,48	**1,73**	0,12
20 - 100	3,14	4,60	**3,08**	2,42	0,06	3,10	**1,67**	0,04
50 - 100	**3,44**	1,98	3,46	1,33	-0,02	3,45	**0,81**	-0,01

A Tabela 3 revela que, para densidades no intervalo de 0,7 a 0,9, as heurísticas propostas apresentaram os resultados mais expressivos em relação ao algoritmo original. No entanto, observa-se que, para d = 0,95, o desempenho do $LNSA - HC_2$ apresenta uma piora substancial, exibindo um comportamento distinto das demais densidades.

Tabela 3: Comparação entre as heurísticas construtivas para diferentes densidades do grafo de conflitos.

	$LNSA$		$LNSA - HC_1$			$LNSA - HC_2$		
d	$Gap(\%)$	$\bar{t}(s)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$
0,00	0,54	2,83	**0,53**	1,50	0,01	0,54	**0,58**	0,00
0,10	0,55	3,33	0,53	1,42	0,02	**0,52**	**0,75**	0,03
0,20	0,50	3,67	0,48	1,58	0,02	**0,47**	**0,83**	0,03
0,30	0,55	3,67	0,54	1,67	0,01	**0,53**	**0,83**	0,02
0,40	0,67	4,17	**0,62**	1,75	0,05	0,63	**1,08**	0,04
0,50	0,86	3,75	0,81	1,83	0,05	**0,80**	**1,25**	0,06
0,60	1,01	3,83	0,92	2,08	0,09	**0,92**	**1,67**	0,09
0,70	1,57	4,33	1,45	2,42	0,12	**1,45**	**1,92**	0,12
0,80	1,98	3,83	**1,83**	2,50	0,15	1,86	**1,92**	0,12
0,90	4,12	4,08	3,97	2,08	0,15	**3,97**	**1,83**	0,15
0,95	8,41	3,25	**8,32**	3,33	0,09	8,54	**2,00**	-0,13
0,99	23,99	4,42	**23,91**	2,92	0,08	23,92	**2,17**	0,07

Os valores médios apresentados nas três tabelas revelam um impacto positivo, embora modesto, das heurísticas propostas no desempenho do algoritmo original. Além disso, observa-se um aumento significativo na eficiência, com uma redução de aproximadamente dois terços no tempo de execução necessário para o LNSA.

Em termos de melhoria média global, o $LNSA - HC_1$ apresentou um ganho de 0,075%, enquanto o $LNSA - HC_2$ obteve 0,05% em relação ao LNSA. No entanto, no que diz respeito ao tempo de execução, o $LNSA - HC_2$ destacou-se como a heurística mais eficiente, com um tempo médio de 1,4 segundos, representando uma redução de 2,37 segundos em relação ao LNSA.

5.3. Árvores prescritivas para o VSBPPC

Para a construção das árvores prescritivas para o VSBPPC, definimos o conjunto de algoritmos $\mathcal{A} = \{LNSA - HC_1, LNSA - HC_2\}$ e dividimos as 720 instâncias de X em X_{treino}, com 576 instâncias (80%), e X_{teste}, com 144 instâncias (20%).

A função descritiva das instâncias de X é definida como $\Phi(x) : \mathcal{X} \rightarrow \mathbb{R}^3$, cuja saída corresponde à tripla (n, l_{min}, d), onde n é a quantidade de tarefas, $l_{min} = min_{t \in T}\{l_t\}$ é o tamanho da menor tarefa e $d = \frac{2 \cdot |E|}{|V|^2 - |V|}$ é a densidade do grafo de conflitos. Utilizou-se Φ sobre X para construir a base descritiva D e o conjunto de algoritmos $\mathcal{A}$ para criar o vetor de rótulos y.

Para o treinamento das árvores de decisão, utilizou-se o algoritmo CART (Breiman et al. [1984]) em $(D_{\text{treino}}, y_{\text{treino}})$, gerando duas árvores: uma não regularizada (T) e outra regularizada (T_α). A árvore T possui 91 nós, inviabilizando sua exibição, enquanto T_α é ilustrada na Figura 6.

Os resultados das árvores de decisão estão sintetizados na Tabela 4. Nota-se que a árvore sem regularização possui a menor taxa de erro de treinamento, contudo, isso ocorre às custas de um erro significativamente maior no conjunto de teste. A árvore regularizada demonstra maior consistência nos conjuntos de treino e teste, alcançando acurácias de 82,3% e 84,7%, respectivamente.

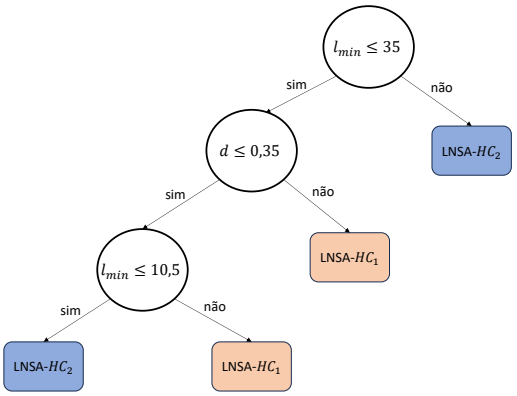

Figura 6: Modelo de árvore regularizada treinada. Fonte: Acervo do autor.

Tabela 4: Comparação do treinamento dos modelos de árvore de decisão.

	T	T_α
E_{in} (%)	**13,76**	17,69
E_{out} (%)	20,83	**15,28**
Precisão (%)	79,00	**86,00**
Revocação (%)	79,00	**86,00**

5.4. Meta-heurísticas guiadas pela árvore prescritiva

As Tabelas 5, 6 e 7 apresentam uma análise comparativa do desempenho das variantes do LNSA guiadas pela árvore prescritiva: $LNSA - T$, $LNSA - T_\alpha$ e $LNSA - T_\alpha^i$, que utilizam as predições das árvores de decisão T, T_α e T_α^i. Esses experimentos foram realizados adotando as mesmas instâncias, métricas e variáveis discutidas na seção 5.2.

Os experimentos mostram que o $LNSA - T$ apresentou desempenho inferior ao LNSA, indicando o sobreajuste da árvore não regularizada T. Em contrapartida, a abordagem tradicional com a árvore prescritiva T_α combinou o melhor desempenho do $LNSA - HC_1$ com a maior eficiência do $LNSA - HC_2$, e o $LNSA - T_\alpha^i$, ao aplicar iterativamente as prescrições da árvore T_α, superou o $LNSA - T_\alpha$, tanto em termos de qualidade média das soluções quanto em eficiência.

Tabela 5: Resultado dos LNSAs guiados pelas árvores prescritivas para diferentes quantidades de itens.

	$LNSA - T$			$LNSA - T_\alpha$			$LNSA - T_\alpha^i$		
$\|V\|$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$
100	7,10	0,33	-2,74	**4,29**	0,26	0,07	4,31	**0,14**	0,05
200	4,98	3,33	-1,88	3,02	2,54	0,08	**2,98**	**2,39**	0,12

Os experimentos mostram que os comportamentos do $LNSA - T_\alpha$ e do $LNSA - T_\alpha^i$ seguiram um padrão semelhante ao observado nas meta-heurísticas $LNSA - HC_1$ e $LNSA - HC_2$, com as quais essas variantes foram treinadas. Para essas abordagens, o desempenho foi superior à medida que aumentava a quantidade de itens nas instâncias do problema, diminuía o tamanho mínimo dos itens e o intervalo de densidade do grafo de conflitos se situava entre 0,7 e 0,9.

A Tabela 8 resume as métricas em relação ao LNSA das quatro meta-heurísticas propostas,

Tabela 6: Resultado dos LNSAs guiados pelas árvores prescritivas para diferentes tamanhos de itens.

	$LNSA-T$			$LNSA-T_\alpha$			$LNSA-T_\alpha^i$		
s	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$
1 - 100	8,18	2,29	-3,58	4,42	1,52	0,18	**4,40**	**1,46**	0,20
20 - 100	5,65	2,40	-2,51	3,09	1,81	0,05	**3,07**	**1,58**	0,07
50 - 100	4,29	0,85	-0,89	**3,45**	0,88	-0,01	3,46	**0,75**	-0,02

Tabela 7: Resultado dos LNSAs guiados pelas árvores prescritivas para diferentes densidades d.

	$LNSA-T$			$LNSA-T_\alpha$			$LNSA-T_\alpha^i$		
d	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$
0,00	1,54	0,83	-1,00	**0,53**	**0,67**	0,01	0,53	0,75	0,01
0,10	1,90	1,00	-1,35	**0,52**	0,92	0,03	0,53	**0,83**	0,02
0,20	1,71	1,00	-1,21	**0,47**	1,00	0,03	0,48	**0,75**	0,02
0,30	2,16	1,58	-1,61	0,53	1,17	0,02	**0,51**	**1,00**	0,04
0,40	2,61	1,50	-1,94	**0,62**	**1,17**	0,05	0,63	1,33	0,04
0,50	2,66	1,67	-1,80	**0,80**	1,17	0,06	0,81	**1,08**	0,05
0,60	3,19	2,08	-2,18	0,92	1,33	0,09	**0,92**	**1,00**	0,09
0,70	3,92	2,33	-2,35	1,42	1,50	0,15	**1,41**	**1,50**	0,16
0,80	5,41	2,33	-3,43	**1,82**	**1,83**	0,16	1,83	1,92	0,15
0,90	8,50	1,92	-4,38	3,94	2,00	0,18	**3,88**	**1,33**	0,24
0,95	13,01	3,08	-4,60	8,37	1,75	0,04	**8,32**	**1,75**	0,09
0,99	25,88	2,67	-1,89	23,92	2,33	0,07	**23,89**	**1,92**	0,10

incluindo o $LNSA^*$, que representa um algoritmo capaz de executar as meta-heurísticas $LNSA-HC_1$ e $LNSA-HC_2$, selecionando sempre a melhor entre as soluções fornecidas.

Tabela 8: Sumarização dos resultados obtidos pelas diferentes abordagens propostas.

$\mathcal{A}$	$Gap(\%)$	$\bar{t}(s)$	$\Delta(\%)$
$LNSA-HC_1$	3,655	2,090	0,075
$LNSA-HC_2$	3,68	1,400	0,050
$LNSA-T_\alpha$	3,655	1,400	0,075
$LNSA-T_\alpha^i$	3,645	**1,265**	0,085
$LNSA^*$	**3,640**	3,490	0,090

Verifica-se que o $LNSA^*$ estabelece um limite superior, alcançando o melhor resultado com uma melhoria de 0,09% em relação ao LNSA. Considerando que a árvore regularizada T_α obteve uma acurácia de 84,72% no teste, o desempenho de 0,075% do $LNSA-T_\alpha$ é consistente com o obtido pelo $LNSA^*$, correspondendo a aproximadamente 83,33% de sua melhoria média.

Observa-se que o $LNSA-T_\alpha^i$ supera o desempenho do $LNSA-T_\alpha$ e se aproxima do resultado do $LNSA^*$. Esse algoritmo extrapola os melhores resultados das heurísticas propostas, aperfeiçoando o desempenho superior do $LNSA-HC_1$ e a maior eficiência do $LNSA-HC_2$.

6. Considerações finais e trabalhos futuros

Este trabalho analisa o problema do empacotamento com *bins* variáveis e conflitos (VSBPPC) e propõe aprimoramentos ao LNSA. As melhorias incluem novas heurísticas construtivas e a integração de modelos de aprendizagem de máquina para guiar a seleção de heurísticas construtivas.

Os resultados computacionais mostraram um desempenho ligeiramente superior à literatura, com reduções significativas no tempo de execução. Além disso, o estudo evidenciou o sucesso da aplicação de modelos de aprendizagem de máquina para guiar as decisões ao longo do algoritmo de otimização, permitindo uma adaptação mais precisa às características das instâncias.

No entanto, apesar dos avanços alcançados, algumas limitações persistem, apontando para diversas oportunidades de investigação futura. Uma possibilidade de melhoria seria integrar técnicas de aprendizado por reforço, o que permitiria aprimorar o LNSA em outras partes do algoritmo, superando a limitação associada à necessidade de dados rotulados. Além disso, o estudo de extensões do VSBPPC que considerem incertezas nos tamanhos dos itens pode possibilitar a modelagem de cenários mais realistas, alcançando variações nas demandas ou margens de erro nas medições.

Essas direções abrem caminho para o aprimoramento tanto da eficácia quanto da aplicabilidade do VSBPPC, expandindo seu alcance para novos domínios e cenários do mundo real. Por fim, espera-se que as meta-heurísticas, aliadas a técnicas de aprendizagem de máquina, desempenhem um papel cada vez mais decisivo na solução de problemas de otimização complexos.

Referências

Breiman, L., Friedman, J., Stone, C., e Olshen, R. (1984). *Classification and Regression Trees*. Taylor & Francis. ISBN 9780412048418.

Coffman, E. G., Jr., Garey, M. R., e Johnson, D. S. (1978). An application of bin-packing to multiprocessor scheduling. *SIAM Journal on Computing*, 7(1):1–17.

Ekici, A. (2023). A large neighborhood search algorithm and lower bounds for the variable-sized bin packing problem with conflicts. *European Journal of Operational Research*, 308(3):1007–1020. ISSN 0377-2217.

Epstein, L., Favrholdt, L. M., e Levin, A. (2011). Online variable-sized bin packing with conflicts. *Discrete Optimization*, 8(2):333–343.

Friesen, D. e Langston, M. (1986). Variable sized bin packing. *SIAM Journal on Computing*, 15: 222–230.

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

Maiza, M. e Radjef, M. S. (2011). Heuristics for solving the bin-packing problem with conflicts. *Applied Mathematical Sciences (Ruse)*, 5.

Vilas Boas, M. G., Santos, H. G., Merschmann, L. H. d. C., e Vanden Berghe, G. (2021). Optimal decision trees for the algorithm selection problem: integer programming based approaches. *International Transactions in Operational Research*, 28(5):2759–2781.