

GENERATE-AND-SOLVE aplicada ao Problema de Seleção de Pedidos

Rommel Dias Saraiva^a, Maria Rita Rocha Mota^b, Napoleão Nepomuceno^c, Raphael Kramer^b

^{a,c}Universidade de Fortaleza, ^aNúcleo de Ciência de Dados e Inteligência Artificial, ^cPrograma de Pós-Graduação em Informática Aplicada

^{a,c}Av. Washington Soares, 1321, bairro Edson Queiroz, Fortaleza - CE, Brasil, CEP: 60811-905

^bUniversidade Federal de Pernambuco, Departamento de Engenharia de Produção

^bAv. da Arquitetura, s/n – Cidade Universitária, Recife - PE, Brasil, CEP: 50740-550

{rommelds, napoleaovn}@unifor.br

{maria.mrrm, raphael.kramer}@ufpe.br

RESUMO

Neste trabalho, abordamos o Problema de Seleção de Pedidos (PSP), um problema de Otimização Combinatória proposto pelo MERCADO LIVRE durante o primeiro desafio de otimização realizado pela Sociedade Brasileira de Pesquisa Operacional (SOBRAPO). Em suma, o objetivo é maximizar a razão da quantidade de itens coletados por quantidade de corredores selecionados, respeitando limites mínimo e máximo de itens. Aplicamos o *framework* híbrido GENERATE-AND-SOLVE, que combina um Algoritmo Genético de Chaves Aleatórias Viciadas (BRKGA) como gerador de instâncias reduzidas e um modelo de Programação Linear Inteira para resolvê-las. Nossa abordagem explora subconjuntos promissores de corredores para trabalhar com um modelo linear, reduzir a complexidade de resolução do problema e garantir soluções de alta qualidade. Experimentos computacionais com vinte instâncias mostraram que a heurística convergiu para a solução ótima da abordagem exata em cinco casos e que, nos demais casos, ambos apresentaram resultados próximos, demonstrando assim a eficácia do método proposto.

PALAVRAS CHAVE. Programação Inteira, Algoritmos híbridos, Problema de Seleção de Pedidos.

MH – Metaheurísticas; L&T – Logística e Transportes

ABSTRACT

This work addresses the Order Selection Problem (OSP), a Combinatorial Optimization problem proposed by MERCADO LIVRE during the first optimization challenge organized by the Brazilian Society of Operational Research (SOBRAPO). In summary, the objective is to maximize the ratio of the number of items collected to the number of selected aisles, while respecting minimum and maximum item constraints. We applied the hybrid *Generate-and-Solve* framework, which combines a Biased Random-Key Genetic Algorithm (BRKGA) as a reduced-instance generator with an Integer Linear Programming model to solve them. Our approach explores promising subsets of aisles to work with a linear model, reduce the problem's computational complexity, and ensure high-quality solutions. Computational experiments with twenty instances showed that the heuristic converged to the optimal solution found by the exact approach in five cases, and in the remaining instances, both approaches yielded similar results, thus demonstrating the effectiveness of the proposed method.

KEYWORDS. Integer Programming. Hybrid algorithms. Order Selection Problem.

MH – Metaheuristics; L&T – Logistics and Transport

1. Introdução

Problemas de Otimização Combinatória (OC) visam encontrar uma solução ótima a partir de um conjunto finito, mas tipicamente grande, de soluções viáveis. A maioria dos problemas de OC é NP-Difícil [Gary e Johnson, 1979], o que significa que não existe um algoritmo de tempo polinomial conhecido capaz de resolver todas as instâncias desses problemas de forma exata. Assim, à medida que o tamanho da instância cresce, algoritmos aproximativos, heurísticas e meta-heurísticas são empregados para encontrar soluções viáveis em tempo computacional aceitável.

Os problemas de OC surgem em diversas áreas, como: roteamento (*e.g.*, o Problema do Caixeiro Viajante [Applegate, 2006] e o Problema de Roteamento de Veículos [Toth e Vigo, 2002]), escalonamento (*e.g.*, o Problema de *Job Shop* [Carlier e Pinson, 1989] e o Problema de *Flow Shop* [Stützle et al., 1998]) e o Problema da Mochila [Dantzig, 1957]. Esses são alguns exemplos de problemas fundamentais da literatura, cuja importância está relacionada com a possibilidade deles servirem de base conceitual para o entendimento e estruturação de outros modelos de programação, como é o caso do Problema da Mochila que pode compor outros problemas como o de Corte e Empacotamento [Gilmore e Gomory, 1965] e o de Seleção e Separação de Pedidos em Armazéns [Hackman et al., 1990; Gu et al., 2010]. O Problema da Mochila busca selecionar um subconjunto de itens visando maximizar o valor total (benefício) sujeito a restrições como a capacidade da mochila. Essa lógica é diretamente aplicável a cenários de seleção de pedidos em armazéns, em que é necessário escolher quais pedidos atender, respeitando restrições como capacidade de transporte, prazos ou prioridades.

No processo de seleção e coleta de pedidos, estão envolvidas decisões críticas para a gestão operacional de armazéns como a definição do selecionador responsável pela atividade de coleta, o momento de início da rota e o trajeto percorrido. Essas escolhas podem representar cerca de 60% do tempo total gasto nas operações do armazém. Nesse contexto, a otimização desses processos pode gerar um impacto significativo na redução dos custos operacionais, além de melhorar o desempenho de etapas subsequentes da cadeia de suprimentos, como a entrega dos produtos corretos aos clientes no prazo adequado [Pardo et al., 2024].

Na pesquisa de Gu et al. [2010] os autores ressaltam a associação entre esses problemas (Problema da Mochila e Seleção de Pedidos) e propõem um algoritmo *branch-and-bound* modificado baseado em aproximação externa para a resolução de um problema de alocação de itens na área de *picking*, visando maximizar o benefício desta área ao equilibrar os custos de separação de pedidos e de reabastecimento interno. Os autores comparam a eficiência (tempo para a obtenção da otimalidade) dos resultados dessa abordagem exata com os resultados obtidos ao ser considerada a abordagem heurística proposta na pesquisa de Hackman et al. [1990]. Gu et al. [2010] ressaltam que foi possível obter, na maioria dos casos, soluções ótimas em curto tempo computacional e que o método proposto por eles demonstra ser uma boa alternativa ao método heurístico.

Pinto et al. [2023] apresentam, em uma pesquisa de natureza descritiva, que a abordagem de Problemas de Seleção de Pedidos pode variar de acordo com os sistemas de separação utilizados, as regras de negócio e as especificidades de cada armazém. Além disso, os autores classificam os sistemas de *picking* de acordo com políticas de coleta e métodos de otimização, sendo eles: Agrupamento de Pedidos (*Order Batching*), Métodos de Roteamento (*Routing Methods*), Definição de Layout (*Layout Definition*) e Designação de Armazenamento (*Storage Designation*). Para cada uma dessas categorias, os autores expõem as pesquisas relacionadas e os métodos de otimização mais utilizados, sendo possível observar que a utilização de heurísticas se sobressai frente à resolução dos problemas de forma exata, dada a complexidade computacional.

Esta pesquisa aborda o Problema de Seleção de Pedidos (PSP) anunciado pelo *Shipping Optimization Team* (SOT) do MERCADO LIVRE no primeiro desafio de otimização da empresa,

feito em colaboração com a Sociedade Brasileira de Pesquisa Operacional (SOBRAPO). O objetivo busca maximizar a quantidade de itens coletados por corredor visitado do armazém, sujeito a limites mínimo e máximo de itens a serem recolhidos. Para a resolução desse problema, recorremos à metodologia híbrida GENERATE-AND-SOLVE [Nepomuceno et al., 2008], que envolve um método exato e uma meta-heurística. Realizamos experimentos computacionais em instâncias disponibilizadas publicamente. Comparamos os resultados obtidos pela metodologia híbrida com os obtidos por meio de modelagem de Programação Matemática e com os apresentados por outras equipes e divulgados pela organização do Desafio.

O restante deste artigo está estruturado da seguinte forma. Na Seção 2, têm-se a descrição e a modelagem matemática do PSP. Na Seção 3, é explicado o funcionamento de um Algoritmo Genético de Chaves Aleatórias Viciadas. Na Seção 4, é apresentado o funcionamento da metodologia híbrida GENERATE-AND-SOLVE e como ela foi aplicada no PSP. Na Seção 5, têm-se os resultados dos experimentos computacionais. Por fim, na Seção 6, são apresentadas considerações finais deste trabalho.

2. Descrição do Problema

No MERCADO LIVRE, quando um pedido é realizado por parte de um cliente, ele entra em um *backlog* de pedidos da empresa. A etapa subsequente envolve a coleta no armazém dos itens correspondentes, seguida pela embalagem e transporte. A coleta individual de itens, ou seja, pedido a pedido, tende a ser menos eficiente do que a coleta de múltiplos pedidos organizados em grupos denominados *waves*, termo que utilizaremos ao longo deste trabalho para nos referirmos a tais agrupamentos. Para maximizar a produtividade da operação de coleta, é necessário que os pedidos que irão compor a próxima *wave* sejam selecionados adequadamente. O foco do desafio proposto pelo MERCADO LIVRE em conjunto com a SOBRAPO é a concepção de algoritmos para resolver o problema de encontrar uma *wave* ótima para maximização da produtividade da operação de coleta [Souza, 2021].

Esse problema, chamado de Problema de Seleção de Pedidos (PSP), tem como entrada um conjunto de pedidos O , um conjunto de itens I e um conjunto de corredores de coleta A . Cada pedido $o \in O$ tem um subconjunto de itens $I_o \subseteq I$. Sabe-se que $I = \bigcup_{o \in O} I_o$. Tem-se também $A_i \subseteq A$ como o subconjunto de corredores contendo pelo menos uma unidade do item i , para todo $i \in I$. Sabe-se que $A = \bigcup_{i \in I} A_i$.

Além disso, considere u_{oi} como o número de unidades do item i solicitado pelo pedido o , para todo $o \in O$ e $i \in I$. Considere também e_{ai} como o número de unidades do item i disponíveis no corredor a , para todo $a \in A$ e $i \in I$. Por fim, sejam LB e UB , respectivamente, o limite inferior e o limite superior do tamanho da *wave* (mensurado em quantidade de itens).

O objetivo do PSP é coletar um subconjunto de pedidos (isto é, uma *wave*) $O' \subset O$ do *backlog* e um subconjunto de corredores (locais de coleta) $A' \subset A$ que maximizem a produtividade do processo de coleta, dada por:

$$\frac{\sum_{o \in O'} \sum_{i \in I_o} u_{oi}}{|A'|} \quad (1)$$

Há algumas condições necessárias a serem seguidas no processo de coleta de uma *wave* como, por exemplo, o número total de itens que devem ser coletados. Esse número deve estar entre um limite inferior (LB) e superior (UB), ambos inclusos. Além disso, os corredores selecionados devem ter armazenamento suficiente para cada um dos itens incluídos nos pedidos da *wave*. Assim, qualquer par (O', A') que satisfaça essas condições constitui uma solução viável para o PSP, tornando a *wave* O' viável para a operação.

Modelamos o PSP por meio de um modelo de Programação Não-Linear Inteira (PNLI). Para isso, consideramos as famílias de variáveis:

$$x_o = \begin{cases} 1, & \text{se } o \text{ é um pedido selecionado para a wave,} \\ 0, & \text{caso contrário.} \end{cases}$$

para todo pedido $o \in O$. Além disso, temos:

$$y_a = \begin{cases} 1, & \text{se } a \text{ é um corredor visitado no processo de coleta da wave,} \\ 0, & \text{caso contrário.} \end{cases}$$

para todo corredor $a \in A$. O modelo de PNLI para o PSP é dado por:

$$(PSP) \text{ maximizar } \frac{\sum_{o \in O} \sum_{i \in I_o} u_{oi} x_o}{\sum_{a \in A} y_a} \quad (2)$$

$$\text{sujeito à: } \sum_{o \in O} u_{oi} x_o \leq \sum_{a \in A} e_{ai} y_a, \quad \forall i \in I \quad (3)$$

$$\sum_{o \in O} \sum_{i \in I_o} u_{oi} x_o \geq LB \quad (4)$$

$$\sum_{o \in O} \sum_{i \in I_o} u_{oi} x_o \leq UB \quad (5)$$

$$x_o \in \{0, 1\}, \quad \forall o \in O \quad (6)$$

$$y_a \in \{0, 1\}, \quad \forall a \in A \quad (7)$$

A função objetivo (2), não-linear, maximiza a produtividade da operação de coleta. As restrições (3) asseguram que os corredores selecionados tenham armazenamento suficiente para cada um dos itens incluídos nos pedidos da wave. As restrições (4) e (5) impõem, respectivamente, um limite inferior e superior de itens a serem considerados na wave. Por fim, as restrições (6) e (7) determinam o domínio das variáveis de decisão.

3. Algoritmo Genético de Chaves Aleatórias Viciadas

O Algoritmo Genético de Chaves Aleatórias Viciadas (BRKGA, do inglês *Biased Random-Key Genetic Algorithm*) é uma meta-heurística que utiliza chaves aleatórias para representar soluções de problemas de otimização. Proposto por Gonçalves e Resende [2011], o BRKGA é uma variação dos algoritmos genéticos tradicionais e tem sido largamente aplicado na literatura de OC com o intuito de encontrar boas soluções viáveis, como em problemas de empacotamento [Gonçalves e Resende, 2012], cobertura [Ruiz et al., 2015], design de redes [Andrade et al., 2015], dentre vários outros presentes na literatura de OC.

Um BRKGA clássico inicia a partir da geração de uma população de indivíduos, cada qual representado por um vetor de chaves aleatórias (números reais entre 0 e 1), chamado de cromossomo. Uma função de decodificação, dependente do problema, recebe como entrada o cromossomo e determina uma solução para o problema de otimização, cuja qualidade é mensurada com uma função de *fitness*. Quanto maior é o valor de retorno dessa função, melhor é a qualidade da solução.

Para a geração seguinte $k + 1$, divide-se a população da geração k em dois grupos: elite (melhores soluções) e não-elite (restante), como mostra a Figura 1. Copiam-se os indivíduos do grupo elite para a geração $k + 1$ e geram-se novos indivíduos aleatoriamente (mutantes). Os demais

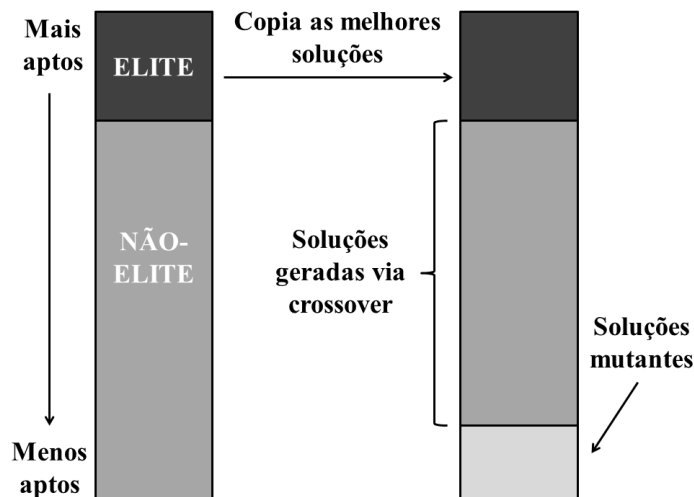


Figura 1: Transição de gerações no BRKGA

indivíduos da geração $k + 1$ são gerados via *crossover* (ver Figura 2) entre um indivíduo elite e um não-elite, com um viés para as características do indivíduo elite a partir de uma probabilidade p_A dada como entrada (na figura, com valor 0,7). Esse passo a passo se segue até um critério de parada preestabelecido ser alcançado, como número máximo de gerações ou quantidade de gerações sem melhoria na qualidade da solução.

4. GENERATE-AND-SOLVE

Denominadas de meta-heurísticas híbridas por Dumitrescu e Stützle [2003], metodologias que combinam métodos exatos e heurísticos são vistas como poderosas ferramentas de resolução de problemas de OC, já que exploram as melhores características de cada uma das abordagens no intuito de obter soluções de boa qualidade para os problemas tratados.

Dentro desse contexto está inserida a metodologia híbrida GENERATE-AND-SOLVE (GS), introduzida por Nepomuceno et al. [2008]. Como mostra a Figura 3, o GS faz uso de duas estruturas conceitualmente distintas: o Decodificador de Instâncias Reduzidas (DIR) e o Gerador de Instâncias Reduzidas (GIR). No DIR, implementa-se um método exato para resolver um problema de otimização (por exemplo, *solver* de PLI). Já no GIR, tem-se uma meta-heurística (por exemplo, BRKGA) que gera instâncias reduzidas do problema original. Iterativamente, o GIR gera uma

Cromossomo-pai A	0.22	0.77	0.43	0.95
Cromossomo-pai B	0.36	0.15	0.81	0.54
Número aleatório	0.58	0.89	0.78	0.25
$p_A = 0.7$	<	>	>	<
Cromossomo-filho	0.22	0.15	0.81	0.95

Figura 2: Cruzamento no BRKGA

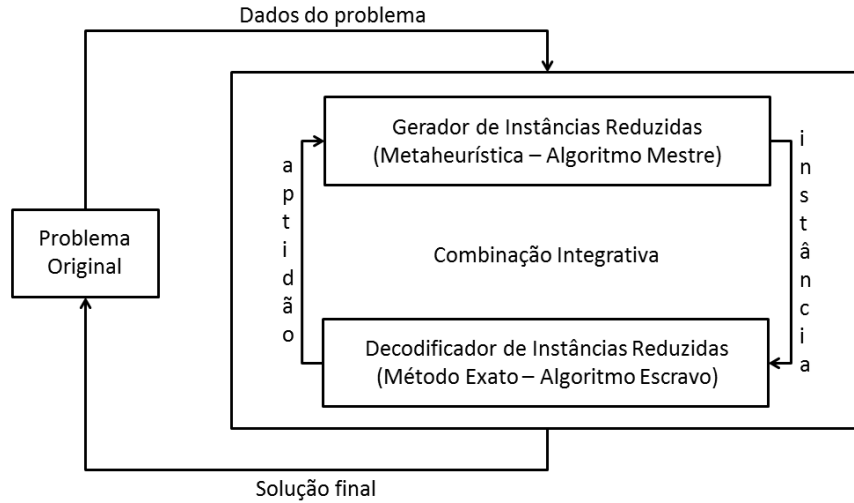


Figura 3: Metodologia híbrida GENERATE-AND-SOLVE

subinstância de um problema de otimização que, quando submetida ao DIR, recebe um valor de aptidão (ou *fitness*) associado. Com esse mecanismo de interações entre GIR e DIR, espera-se que a meta-heurística gere instâncias reduzidas promissoras para o problema original, preservando subestruturas da solução que levam a um *fitness* de alta qualidade.

A literatura de OC aponta que o GS já fora aplicado em problemas de corte [Saraiva et al., 2013; Santos e Nepomuceno, 2022], empacotamento [Saraiva et al., 2019] e redes de sensores [Coudert et al., 2010, 2011; Pinheiro et al., 2012], para citar alguns. Neste trabalho, recorreremos ao GS para resolver o PSP. Para isso, descreveremos nas seções seguintes como instâncias reduzidas são representadas no GIR, bem como são resolvidas no DIR.

4.1. Gerador de Instâncias Reduzidas

O modelo (2)–(7) que concebemos para o PSP, formulado na Seção 2, é não-linear. Dessa forma, não se pode modelar e resolver o problema em *solvers* de otimização linear como IBM ILOG CPLEX e OR-TOOLS [Perron e Furnon, 2024].

No entanto, observe que, caso sejam fixados valores para as variáveis y_a (com $a \in A$), isto é, um subconjunto $A' \subset A$ de corredores a serem obrigatoriamente considerados na solução, teríamos um modelo linear com somente variáveis x_o (com $o \in O$). Esse modelo seria responsável por encontrar a *wave* ótima $O' \subset O$ para o subconjunto de corredores A' tal que a produtividade fosse maximizada. Isso corresponde a um subproblema do problema original. O desafio consiste em encontrar quais corredores são promissores para maximizar a produtividade da coleta de pedidos.

Para essa escolha de corredores, recorreremos ao BRKGA, onde cada cromossomo é um vetor v de $|A|$ elementos como se segue.

$$\begin{bmatrix} v_1 & v_2 & v_3 & \dots & v_{|A|-1} & v_{|A|} \end{bmatrix}$$

Um valor $v_i \in [0, 1]$, com $i = 1, \dots, |A|$, onde i corresponde as posições do vetor, ou seja, a cada elemento do cromossomo, cuja probabilidade determina a decisão de se considerar ou não o respectivo corredor na solução. Caso o valor contido em v_i seja menor ou igual a uma probabilidade p_E dada como entrada, então o corredor associado à posição i estará na solução do PSP.

Para o subconjunto de corredores A' em consideração, o modelo, agora linear, encontrará a *wave* ótima $O' \subset O$ e, como custo da função objetivo, uma produtividade correspondente ao *fitness* do cromossomo. Obviamente, o valor p_E deve ser calibrado com cuidado (ver Seção 4.2).

Valores elevados tendem a levar a produtividades baixas, enquanto valores baixos tendem a levar a produtividades mais satisfatórias, porém também a inviabilidades.

Para fins de exemplificação, suponha que existam um total de $|A| = 5$ corredores como entrada e que $p_E = 0.3$. O seguinte cromossomo leva a um subproblema do problema original:

0.21	0.94	0.09	0.70	0.68
------	------	------	------	------

Nele, teremos o subconjunto de corredores $A' = \{1, 3\}$ para o qual uma *wave* $O' \subset O$ e o respectivo valor de *fitness* serão identificados após a resolução do modelo, agora linear. Com essa estratégia, trabalhamos com um modelo linear e reduzimos a complexidade de resolução do problema.

Ao final do processo evolutivo do BRKGA, esperamos que a população de soluções tenha convergido para um conjunto de indivíduos de alta qualidade, capazes de oferecer subconjuntos A' satisfatórios para maximizar a produtividade de coleta da *wave*.

4.2. Avaliação de corredores

A avaliação dos corredores em uma instância do PSP é fundamental para guiar adequadamente o processo evolutivo do BRKGA. Conforme discutido na seção anterior, cada cromossomo possui cardinalidade $|A|$, que pode variar significativamente de acordo com a entrada do problema. Em alguns casos, esse valor pode ser suficientemente elevado a ponto de prejudicar o desempenho do algoritmo. Para mitigar esse efeito, propomos funções que atribuem um *score* $s_a \in [0, 1]$ a cada corredor $a \in A$. Quanto mais próximo de 1 for o *score* de um corredor, maior a probabilidade de que ele esteja presente na solução.

A primeira função de *score* utiliza uma matriz $S_{|A| \times |O|}$, em que cada elemento S_{ao} representa o percentual de itens do pedido o que são atendidos pelo corredor situado na posição a . O *score* de cada corredor é obtido pela média dos valores de sua respectiva linha na matriz.

Na segunda função de *score*, para cada elemento $a \in A$, identificamos quantos pedidos do conjunto O são completamente atendidos por tal corredor. Ao final dessa contagem, dividimos o valor pelo total de pedidos.

Já para a terceira função *score*, optamos por determinar um valor equivalente para cada corredor. Testes preliminares indicaram que $s_a = 0.1$, para todo $a \in A$, levou a soluções melhores do que quando comparamos com outros valores fixos testados.

As funções de *score* são então usadas da seguinte forma: para cada cromossomo inicializado, escolhemos, de maneira uniforme, uma das funções. Para cada corredor $a \in A$, geramos um número aleatório no intervalo real $[0, 1]$. Se esse número for menor ou igual ao *score* s_a daquela função sorteada, então o corredor será adicionado no subconjunto A' de corredores.

4.3. Decodificador de Instâncias Reduzidas

A partir de um conjunto fixo A' de corredores, implementamos no DIR um modelo de PLI que visa encontrar a *wave* ótima que maximize a produtividade. O modelo é como segue:

$$(PSP_{\text{reduzido}}) \text{ maximizar } \frac{\sum_{o \in O} \sum_{i \in I_o} u_{oi} x_o}{|A'|} \quad (8)$$

$$\text{sujeito à: } \sum_{o \in O} u_{oi} x_o \leq \sum_{a \in A'} e_{ai}, \quad \forall i \in I \quad (9)$$

e restrições (4), (5), (6)

A função objetivo (8), agora linear, maximiza a produtividade para $|A'|$ corredores. Já as restrições (9) asseguram que o subconjunto A' de corredores possui disponibilidade para atender a demanda de itens da *wave*. Esse modelo de PLI também leva em consideração as restrições de limite inferior (4) e superior (5) da *wave*, além do domínio das variáveis de decisão (6) referentes aos pedidos a serem escolhidos.

Uma visão geral sobre o método desenvolvido neste trabalho pode ser visualizado na Figura 4 que apresenta o fluxo das ações executadas na metodologia híbrida proposta.

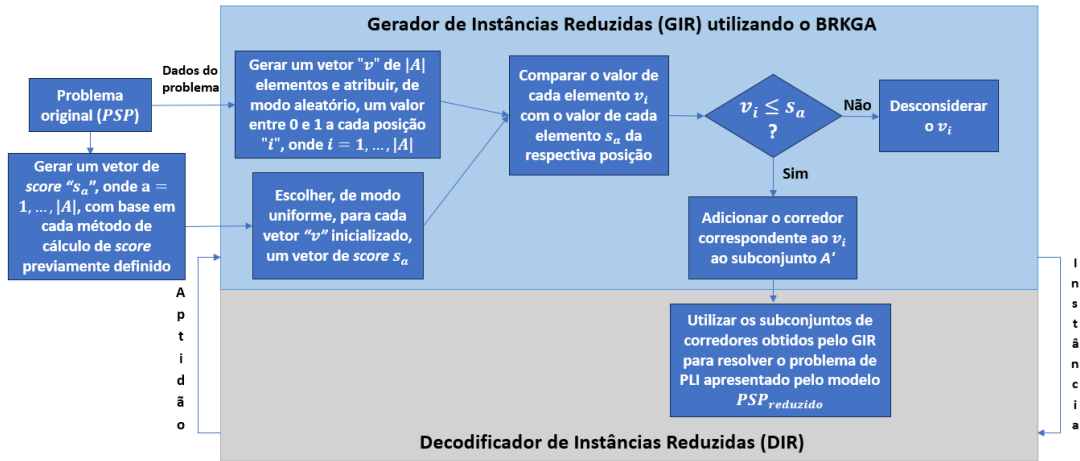


Figura 4: Detalhamento da metodologia GENERATE-AND-SOLVE para o PSP

5. Experimentos computacionais

Por orientação das regras do desafio, implementamos o BRKGA do GIR em linguagem de programação Java, bem como adotamos um tempo limite de $t = 10$ minutos para a execução de cada caso de teste. Um total de 20 instâncias foram usadas nos experimentos as quais foram disponibilizadas pela organização do desafio e podem ser acessadas em: <https://github.com/mercadolibre/challenge-sbpo-2025>.

Para calibrar os parâmetros do BRKGA, realizamos testes empíricos considerando uma combinação de valores com o tamanho da população $popSize \in \{50, 100, 200, 500\}$, a porcentagem de indivíduos elite $topPerc \in \{0.15, 0.20, 0.25, 0.30\}$ e a porcentagem de indivíduos mutantes $botPerc \in \{0.10, 0.15, 0.20\}$. O número de gerações $nGen$ foi definido com um valor muito grande, sendo o critério de parada o limite de tempo. Após teste preliminares, adotamos $popSize = 50$, $topPerc = 0.20$ e $botPerc = 0.20$. Adotamos também $p_A = 0.7$, como sugerem estudos do BRKGA.

Com o intuito de resolver os subproblemas de PLI gerados pelo GIR, utilizamos no DIR o otimizador linear OR-TOOLS, versão 9.11, com parâmetros *default*. Os testes foram executados em uma máquina Intel Pentium i7, 3.60 GHz, e 16 GB de memória RAM. Além do uso dessa metodologia híbrida, o modelo de PNLI (2)–(7) foi resolvido de forma exata com o *solver* Gurobi, versão 12.0.1, para fins de análise comparativa de ambos os métodos.

Para avaliar a qualidade das respostas obtidas pelo método proposto neste trabalho, em comparação com aquelas geradas pela resolução exata do problema original modelado via PNLI, construímos a Tabela 1, que está organizada em quatro blocos principais de colunas: “Instância”, “Modelo Exato”, “GS” e “BKS” (sigla para *Best Known Solution*), os quais apresentam, respectivamente:

- As características da instância, incluindo o identificador dela, a quantidade de pedidos, a quantidade de itens e a quantidade de corredores, respectivamente, nas colunas *nome*, $|O|$, $|I|$ e $|A|$;
- Os resultados da resolução exata do problema de PNLI que contemplam: o custo z_{LB} da melhor solução viável encontrada e o valor do limite dual (*upper bound*) z_{UB} encontrado ao serem realizados experimentos com limites de tempo de 600s, 1800s e 3600s; e os valores dos *gaps* computados para cada tempo de execução;
- Os resultados da abordagem híbrida proposta, apresentados na coluna *GS* que inclui: o custo da *melhor* solução encontrada, a *média* do custo obtido em 10 execuções realizadas e o desvio-padrão do custo na coluna *desvio*;
- Soluções divulgadas pela organização do Desafio, oriundas dos melhores resultados obtidos pelas equipes participantes.

Como se trata de um problema de maximização, a melhor solução viável encontrada é representada pelo *Lower Bound* (*LB*), enquanto o *Upper Bound* (*UB*) corresponde ao limite teórico da melhor solução (valor relaxado) encontrada durante o respectivo experimento. Desse modo, o *gap* foi calculado como: a diferença do *UB* pelo *LB* dividido pelo *UB*.

Com base nos resultados apresentadas na Tabela 1, observa-se que foi possível encontrar a solução ótima para 5 instâncias e que, para estes casos, a abordagem heurística convergiu para o mesmo valor. Além disso, observa-se que as soluções apresentadas pelo “Modelo Exato” e pelo “GS” foram muito próximas as do “BKS” e que, em várias instâncias, foi possível obter o mesmo valor nos três casos.

Nota-se também que, mesmo ocorrendo o aumento do limite de tempo para a realização dos experimentos, não houve uma melhora significativa na qualidade das soluções obtidas na abordagem exata, como pode ser visto ao serem comparados os valores dos *gaps* nos experimentos. Isso pode estar relacionado com a elevada complexidade computacional devido à não linearidade e ao domínio combinatório das variáveis, dificultando a exploração do espaço de busca.

Destaca-se ainda que não foi possível obter o *gap*_{600s} para a *instance_0014*, pois o *solver* não conseguiu explorar a árvore de busca o suficiente, nesse limite de tempo, para realizar o processo de relaxação do modelo e, conseqüentemente, obter um *upper bound*. No entanto, nota-se que, nos demais experimentos para essa instância (e para as demais), o *solver* conseguiu retornar esse valor, possibilitando o cálculo do *gap*.

Por fim, com relação as respostas obtidas pela abordagem heurística, nota-se que, de modo geral, as execuções apresentaram soluções homogêneas, com baixa variabilidade, à exceção de *instance_0006* e *instance_0007* cujo desvio-padrão foi mais acentuado.

6. Conclusão

Neste trabalho, abordamos o Problema de Seleção de Pedidos (PSP), proposto no contexto do desafio de otimização promovido pelo MERCADO LIVRE em parceria com a SOBRAPO. Para tratá-lo, recorremos ao *framework* híbrido GENERATE-AND-SOLVE, que integra um Algoritmo Genético de Chaves Aleatórias Viciadas (BRKGA) para geração de instâncias reduzidas com um modelo de Programação Linear Inteira para resolvê-las com eficiência.

Os resultados computacionais obtidos nas 20 instâncias demonstraram a viabilidade e a qualidade da abordagem proposta, sendo possível encontrar soluções ótimas ou viáveis em tempo computacional restrito (até 10 minutos). Também foi possível observar que, ao ser considerado o mesmo tempo limite de 600s para ambas as abordagens, os resultados heurísticos já se aproximaram dos obtidos pelo método exato, sugerindo que o método proposto pode ser uma estratégia eficaz frente à abordagem exata, principalmente por conseguir reduzir a complexidade do problema original decorrente da não linearidade e da natureza das variáveis de decisão.

Tabela 1: Resultados comparativos entre modelo exato e metodologia GS

Instância				Modelo Exato										GS		BKS
nome	O	I	A	zLB_{600s}	zUB_{600s}	gap _{600s} (%)	zLB_{1800s}	zUB_{1800s}	gap _{1800s} (%)	zLB_{3600s}	zUB_{3600s}	gap _{3600s} (%)	melhor	média	desvio	melhor
instance.0001	61	155	116	15,00	15,00	0,0%	15,00	15,00	0,0%	15,00	15,00	0,0%	15,00	15,00	0,00	15,00
instance.0002	7	7	33	2,00	2,00	0,0%	2,00	2,00	0,0%	2,00	2,00	0,0%	2,00	2,00	0,00	2,00
instance.0003	82	246	124	12,00	26,67	55,0%	12,00	26,50	54,72%	12,00	26,50	54,72%	12,00	11,75	0,80	12,00
instance.0004	16	59	91	3,50	3,50	0,0%	3,50	3,50	0,0%	3,50	3,50	0,0%	3,50	3,43	0,18	3,50
instance.0005	2625	6407	161	154,19	2.197,50	92,98%	154,19	2.197,50	92,98%	155,43	2.197,50	92,93%	176,88	146,49	22,86	177,88
instance.0006	10341	7089	184	691,00	3.892,00	82,25%	691,00	3.892,00	82,25%	691,00	3.892,00	82,25%	680,00	364,08	162,68	691,00
instance.0007	8320	5747	180	392,25	3.847,00	89,80%	392,25	3.847,00	89,80%	392,25	2.622,00	85,04%	389,00	296,36	83,22	392,25
instance.0008	2185	5831	168	162,94	1.420,00	88,53%	162,94	1.420,00	88,53%	162,94	1.420,00	88,53%	160,00	137,37	26,45	162,94
instance.0009	70	222	304	4,19	51,00	91,79%	4,19	51,00	91,79%	4,19	49,00	91,45%	4,42	3,81	0,52	4,42
instance.0010	1602	3689	383	16,57	1.746,00	99,05%	16,57	1.746,00	99,05%	16,63	1.746,00	99,05%	16,00	13,68	2,81	16,79
instance.0011	1029	2784	375	14,25	2.027,00	99,30%	14,46	1.611,00	99,10%	14,46	1.049,00	98,62%	14,87	13,28	0,93	16,85
instance.0012	133	337	342	11,25	88,50	87,29%	11,25	88,50	87,29%	11,25	88,50	87,29%	11,25	9,55	1,36	11,25
instance.0013	8375	7525	413	105,33	4.583,00	97,70%	105,33	4.583,00	97,70%	105,33	4.583,00	97,70%	102,00	70,71	15,30	117,38
instance.0014	12402	10974	413	179,55	NaN	-	181,64	7.739,00	97,65%	181,64	7.739,00	97,65%	178,00	94,20	24,42	181,64
instance.0015	7367	6633	402	144,67	3.679,00	96,07%	144,67	3.679,00	96,07%	144,67	3.679,00	96,07%	145,00	78,09	23,98	149,33
instance.0016	1108	1051	88	85,00	228,67	62,83%	85,00	228,67	62,83%	85,00	197,33	56,93%	85,00	84,70	0,92	85,00
instance.0017	417	411	83	36,50	36,50	0,00%	36,50	36,50	0,00%	36,50	36,50	0,00%	36,50	26,50	0,00	36,50
instance.0018	2682	2309	90	117,20	602,50	80,55%	117,20	602,50	80,55%	117,20	401,67	70,82%	117,20	116,21	1,28	117,20
instance.0019	2257	2104	134	202,00	499,50	59,56%	202,00	499,50	59,56%	202,00	499,50	59,56%	202,00	193,03	15,84	202,00
instance.0020	5	5	5	5,00	5,00	0,0%	5,00	5,00	0,0%	5,00	5,00	0,00%	5,00	5,00	0,00	5,00

Como trabalhos futuros, propomos o refinamento das funções de *score* na seleção de corredores, a linearização do modelo para tentar resolver o problema de forma exata, e um algoritmo de decomposição para instâncias desafiadoras.

Referências

- Andrade, C. E., Resende, M. G., Zhang, W., Sinha, R. K., Reichmann, K. C., Doverspike, R. D., e Miyazawa, F. K. (2015). A biased random-key genetic algorithm for wireless backhaul network design. *Applied Soft Computing*, 33:150–169.
- Applegate, D. L. (2006). *The traveling salesman problem: a computational study*, volume 17. Princeton university press.
- Carlier, J. e Pinson, É. (1989). An algorithm for solving the job-shop problem. *Management science*, 35(2):164–176.
- Coudert, D., Nepomuceno, N., e Rivano, H. (2010). Power-efficient radio configuration in fixed broadband wireless networks. *Computer Communications*, 33(8):898–906.
- Coudert, D., Nepomuceno, N., e Tahiri, I. (2011). Energy saving in fixed wireless broadband networks. In *International Conference on Network Optimization*, p. 484–489. Springer.
- Dantzig, G. B. (1957). Discrete-variable extremum problems. *Operations Research*, 5(2):266–288. URL <https://doi.org/10.1287/opre.5.2.266>.
- Dumitrescu, I. e Stützle, T. (2003). Combinations of local search and exact algorithms. In *Workshops on Applications of Evolutionary Computation*, p. 211–223. Springer.
- Gary, M. R. e Johnson, D. S. (1979). Computers and intractability: A guide to the theory of NP-completeness.
- Gilmore, P. C. e Gomory, R. E. (1965). Multistage cutting stock problems of two and more dimensions. *Operations Research*, 13(1):94–120.
- Gonçalves, J. F. e Resende, M. G. (2011). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525.
- Gonçalves, J. F. e Resende, M. G. (2012). A parallel multi-population biased random-key genetic algorithm for a container loading problem. *Computers & Operations Research*, 39(2):179–190.
- Gu, J., Goetschalckx, M., e and, L. F. M. (2010). Solving the forward-reserve allocation problem in warehouse order picking systems. *Journal of the Operational Research Society*, 61(6):1013–1021.
- Hackman, S. T., Rosenblatt, M. J., e and, J. M. O. (1990). Allocating items to an automated storage and retrieval system. *IIE Transactions*, 22(1):7–14.
- Nepomuceno, N., Pinheiro, P., e Coelho, A. L. (2008). A hybrid optimization framework for cutting and packing problems: case study on constrained 2d non-guillotine cutting. *Recent Advances in Evolutionary Computation for Combinatorial Optimization*, p. 87–99.

- Pardo, E. G., Gil-Borrás, S., Alonso-Ayuso, A., e Duarte, A. (2024). Order batching problems: Taxonomy and literature review. *European Journal of Operational Research*, 313(1):1–24. ISSN 0377-2217. URL <https://www.sciencedirect.com/science/article/pii/S0377221723001534>.
- Perron, L. e Furnon, V. (2024). OR-Tools. URL <https://developers.google.com/optimization/>.
- Pinheiro, P. R., Coelho, A. L. V., Aguiar, A. B., e Sobreira Neto, A. d. M. (2012). Towards aid by generate and solve methodology: application in the problem of coverage and connectivity in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 8(10):790459.
- Pinto, A. R. F., Nagano, M. S., e Boz, E. (2023). A classification approach to order picking systems and policies: Integrating automation and optimization for future research. *Results in Control and Optimization*, 12:100281. ISSN 2666-7207. URL <https://www.sciencedirect.com/science/article/pii/S2666720723000838>.
- Ruiz, E., Albareda-Sambola, M., Fernández, E., e Resende, M. G. (2015). A biased random-key genetic algorithm for the capacitated minimum spanning tree problem. *Computers & Operations Research*, 57:95–108.
- Santos, J. V. S. e Nepomuceno, N. (2022). Computational performance evaluation of column generation and Generate-and-Solve techniques for the one-dimensional cutting stock problem. *Algorithms*, 15(11):394.
- Saraiva, R., Nepomuceno, N., e Pinheiro, P. (2019). A two-phase approach for single container loading with weakly heterogeneous boxes. *Algorithms*, 12(4):67.
- Saraiva, R. D., Nepomuceno, N. V., e Pinheiro, P. R. (2013). The Generate-and-Solve framework revisited: generating by simulated annealing. In *Evolutionary Computation in Combinatorial Optimization: 13th European Conference, EvoCOP 2013, Vienna, Austria, April 3-5, 2013. Proceedings 13*, p. 262–273. Springer.
- Souza, C. C. (2021). Primeiro desafio Mercado Livre de otimização. URL <https://medium.com/mercadolivre-tech/desafio-mercado-livre-de-otimiza%C3%A7%C3%A3o-3a4009607ee3>.
- Stützle, T. et al. (1998). An ant approach to the flow shop problem. In *Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT'98)*, volume 3, p. 1560–1564. Verlag Mainz, Wissenschaftsverlag Aachen.
- Toth, P. e Vigo, D. (2002). *The vehicle routing problem*. SIAM.