UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

EDUARDO MATOSINHOS FLORINDA
Orientador: Prof. Dr. Marco Antonio Moreira de Carvalho

ITERATED LOCAL SEARCH PARA SOLUÇÃO DO *ORDER BATCHING PROBLEM*

Ouro Preto, MG
2023

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

EDUARDO MATOSINHOS FLORINDA

ITERATED LOCAL SEARCH PARA SOLUÇÃO DO *ORDER BATCHING PROBLEM*

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Marco Antonio Moreira de Carvalho

Ouro Preto, MG
2023

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E BIOLOGICAS
DEPARTAMENTO DE COMPUTACAO

FOLHA DE APROVAÇÃO

Eduardo Matosinhos Florinda

Iterated Local Search para Solução do Order Batching Problem

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 20 de Março de 2023.

Membros da banca

Marco Antonio Moreira de Carvalho (Orientador) - Doutor - Universidade Federal de Ouro Preto
Joubert de Castro Lima (Examinador) - Doutor - Universidade Federal de Ouro Preto
Leonardo Cabral da Rocha Soares (Examinador) - Doutor - Instituto Federal do Sudeste de Minas Gerais

Marco Antonio Moreira de Carvalho, Orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 20/03/2023.

Documento assinado eletronicamente por **Marco Antonio Moreira de Carvalho**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 20/03/2023, às 10:41, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0493034** e o código CRC **DF260398**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591692 - www.ufop.br

Resumo

O *order batching problem* (OBP) é um problema de otimização com aplicações práticas provado NP-difícil, representando um desafio tanto para a área acadêmica quanto para a indústria. O objetivo do OBP consiste em minimizar a distância percorrida para coletar um conjunto de pedidos de compra em um centro de distribuição de produtos. Desta forma, os pedidos devem ser agrupados em lotes para posterior coleta. O coletor responsável por esta operação possui uma capacidade máxima, e portanto os lotes formados não devem exceder esta capacidade. Este trabalho apresenta uma implementação da metaheurística busca local iterada adaptada para solução do OBP. Foram conduzidos experimentos computacionais, utilizando instâncias conhecidas na literatura, a fim de comparar os resultados obtidos com os resultados do atual estado da arte. Uma análise estatística indicou que o método proposto superou o estado da arte em termos de qualidade da solução em algumas das instâncias. Nos casos em que os resultados obtidos pareceram ser piores, a análise estatística demonstrou que não houve diferença significativa entre a qualidade das soluções reportadas pelos métodos comparados.

Palavras-chave: busca local iterada. *order batching problem*. *warehouse management*.

Abstract

The order batching problem (OBP) is an optimization problem with practical applications proved as NP-hard, representing a challenge for both academia and industry. The goal of the OBP is to minimize the distance traveled to collect a set of customer orders in a warehouse. In this way, orders must be grouped into batches for later collection. The collector responsible for this operation has a maximum capacity and therefore the batches formed must not exceed it. This work presents an implementation of the iterated local search (ILS) metaheuristic adapted to the OBP solution. Computational experiments were conducted using known instances in the literature to compare the results obtained with the results of the current state of the art method. A statistical analysis indicated that the proposed method surpassed the state of the art in terms of solution quality in some instances. In cases where the results obtained seemed to be worse, the statistical analysis showed that there was no significant difference between the methods.

Keywords: *iterated local search. order batching problem. warehouse management.*

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

ABCOMM	associação brasileira de comércio eletrônico
ABHC	*attribute-based hill climbing*
ALNS	*adaptive large neighborhood search*
ALNS/TS	*adaptive large neighborhood search and tabu search*
C&W	*Clarke and Wright*
CBD	*class-based demand*
ILS	*iterated local search*
ILST	*iterated local search algorithm with tabu thresholding*
JOBPRP	*joint order batching and picker routing problem*
MSE	melhor solução encontrada
OBP	*order batching problem*
OOBP	*online order batching problem*
PLIM	programação linear inteira mista
PRP	*picker routing problem*
RBAS	*rank based ant system*
SKU	*stock keeping unit*
TS	*tabu search*
UDD	*uniformly distributed demand*
VNS	*variable neighborhood search*

Lista de Símbolos

α	Letra grega Alpha - representa o percentual que uma solução pode ser pior que a melhor solução encontrada para ser aceita.
λ	Letra grega Lambda - representa o percentual de aplicação do método de busca local.
ω	Letra grega Omega - representa o percentual de aplicação do método de perturbação.

Sumário

1 Introdução

O *e-commerce* é um modelo de negócio que vem crescendo de forma exponencial em diferentes setores do comércio. A pandemia da COVID-19 pelo coronavírus SARS-CoV-2 impulsionou este crescimento, impondo aos empreendedores a necessidade de adequar seus negócios ao novo contexto de consumo (SILVA et al., 2021). Praticidade, funcionamento 24 horas, monitoração dos hábitos dos consumidores, identificação dos produtos mais procurados, horários de pico de acesso, períodos do ano em que as vendas aumentam, além da diminuição da distância entre cliente-fornecedor são algumas de suas vantagens. Com o advento da pandemia esse cenário se tornou ainda mais vantajoso. Segundo Souza (2022), o *e-commerce* teve um aumento significativo a partir do primeiro semestre de 2020, em virtude do fechamento de lojas físicas e do aumento de novos consumidores.

Os centros de distribuição são responsáveis por armazenar e distribuir diversos tipos de produtos comercializados por empresas. Estes possuem um papel fundamental na logística, já que possibilitam maior eficiência e flexibilidade para responder às necessidades de demanda do mercado. A logística de um centro de distribuição compreende o recebimento de produtos a partir de diferentes fornecedores, o armazenamento e a organização dos produtos e sua distribuição, dentre outras atividades. Estes fatos ressaltam a importância de minimizar os custos de operações dos centros de distribuição a fim de lidar com o grande volume de operações deste modelo. É possível pontuar três problemas de planejamento que se destacam nesse contexto:

1. Determinar a localização de cada produto no centro de distribuição;
2. Agrupar os pedidos de compra em lotes;
3. Determinar a rota de coleta para cada lote.

O primeiro problema consiste em determinar a localização física de um determinado produto no centro de distribuição, seja para armazenar este produto ou seja para coletá-lo, conhecido na literatura como *storage assignment problem*. O segundo problema trata do agrupamento de pedidos de compra em lotes, conhecido na literatura como *order batching problem* (OBP). O OBP pode ser resumido da seguinte maneira: dado um conjunto de pedidos de compra, um coletor de capacidade limitada e uma política de roteamento, deve-se agrupar os pedidos em lotes, de tal forma que a distância necessária para coletá-los seja minimizada e que a restrição de capacidade do coletor seja respeitada. Já o terceiro problema consiste em determinar a ordem de coleta dos produtos que compõem cada lote, conhecido na literatura como *picker routing problem* (PRP).

A coleta de pedidos (ou *picking*) é uma função crucial do centro de distribuição. O objetivo é coletar os produtos em seus respectivos locais de armazenamento, a fim de satisfazer as demandas dos clientes. Ter uma coleta eficiente resulta em um atendimento satisfatório ao cliente, com um prazo pequeno entre a data do pedido até a data da entrega, minimizando os custos de operações. De todas as operações de um centro de distribuição, a coleta de pedidos é considerada uma das mais custosas. Segundo Frazelle (2016), até 50% dos custos operacionais de um centro de distribuição podem ser atribuídos a esta função.

Esta monografia trata do OBP, o qual provou ser fundamental para a eficiência das operações de um centro de distribuição (KOSTER; ROODBERGEN; VOORDEN, 1999). Um bom planejamento nesta etapa implica na redução da duração total da coleta de pedidos. Como consequência, os tempos de processamento e custos de mão de obra também são reduzidos.

Neste trabalho foi implementada a metaheurística denominada busca local iterada (ILS, sigla para *iterated local search*) adaptada para solução do OBP. A ILS possui componentes que permitem explorar uma porção ampla das possíveis soluções do problema. A capacidade do método em fornecer bons resultados depende diretamente das estratégias adotadas para cada componente. Como o foco é no agrupamento dos pedidos de compra em lotes, foram utilizadas políticas de roteamento pré-definidas conhecidas na literatura para tratar a etapa da coleta.

1.1 Justificativa

O problema considerado neste trabalho possui ampla aplicação prática em centros de distribuição de produtos. Além da aplicabilidade prática, trata-se de um problema pertencente a classe NP-Difícil (GADEMANN; VELDE, 2005), o que significa que não se conhece algoritmo eficiente para sua solução em tempo determinístico polinomial, possuindo, portanto, uma grande relevância teórica.

Adicionalmente, segundo pesquisa realizada pelo Movimento Compre&Confie em parceria com a Associação Brasileira de Comércio Eletrônico (ABCOMM), o *e-commerce* brasileiro faturou 56,8% a mais nos oito primeiros meses de 2020 em comparação com o mesmo período do ano de 2019 (Movimento Compre&Confie, ABCOMM, 2022). Este crescimento do *e-commerce* impulsionou um aumento substancial da concorrência, e o acesso à uma grande gama de ofertas fez surgir um tipo de cliente crítico e seletivo, que cobra mais eficiência no atendimento. O presidente da ABCOMM, Mauricio Salvador, afirmou que empresas com operação restrita ao ambiente físico estão em uma situação de desvantagem ampla e correm sérios riscos de sobrevivência (Movimento Compre&Confie, ABCOMM, 2022). Diante disto, fica evidente que a eficiência dos centros de distribuição é um grande diferencial para lidar com este crescimento do *e-commerce*.

Kearney e Kearney (2004) revelam que o custo total de um centro de distribuição pode chegar à 25% dos custos logísticos de uma empresa, fortalecendo a premissa de que é necessário

reduzir os custos desse setor. ALVES et al. (2005) também enfatizam o sistema de distribuição como um fator determinante para o sucesso ou fracasso de uma empresa. Desta maneira, as empresas que conseguem minimizar os custos com o sistema de distribuição podem obter vantagem competitiva sobre a concorrência.

De acordo com Tompkins et al. (2010), o tempo utilizado pelo coletor na coleta de todos os pedidos de compra corresponde a aproximadamente 50% do tempo total de todas as operações essenciais para a coleta dos produtos. Adotando a premissa de que o coletor se locomove a uma velocidade constante, podemos inferir que a minimização da distância necessária para coletar um grupo de pedidos implica na redução proporcional no tempo gasto para coletar esse grupo de pedidos (JARVIS; MCDOWELL, 1991).

1.2 Objetivos

De maneira geral, o objetivo deste trabalho é elaborar um método robusto para solução do *order batching problem* que forneça resultados competitivos, em qualidade de solução e em tempo de execução, quando comparados com os resultados expressos na literatura. São objetivos específicos:

- Realizar pesquisa para geração de embasamento teórico e revisão bibliográfica sobre o tema tratado;
- Implementar a metaheurística busca local iterada aplicada ao *order batching problem*;
- Avaliar o método implementado utilizando instâncias propostas na literatura;
- Comparar os resultados obtidos com aos do método detentor do atual estado da arte.

1.3 Organização do trabalho

O restante deste trabalho está organizado como segue. O Capítulo 2 introduz os primeiros trabalhos encontrados na literatura sobre o tema, bem como uma revisão da literatura de trabalhos que utilizam as mesmas instâncias do OBP consideradas nesta pesquisa. A descrição formal do problema e a fundamentação teórica são apresentadas no Capítulo 3. No Capítulo 4 descreve-se o método proposto para solução do OBP. O Capítulo 5 relata os resultados obtidos pelo método proposto considerando instâncias conhecidas da literatura. Por fim, no Capítulo 6 a conclusão é apresentada.

2 Revisão bibliográfica

Há décadas, operações em centros de distribuição vêm sendo discutidas na literatura de pesquisa operacional. Fisher (1948) publicou o primeiro livro a fazer importantes observações sobre os procedimentos necessários neste contexto, como alocação e coleta de produtos, definição de leiaute, separação de pedidos, procedimentos de empacotamento e controle de estoque, entre outros. Neste estudo, são realizadas sugestões e recomendações de melhoria para estas operações, que resultam em economias operacionais substanciais, e adicionalmente, forneceram o embasamento teórico necessário para elaboração de pesquisas científicas neste tema a partir de então.

Há relatos sobre o OBP na literatura desde a década de 1970, haja visto que Armstrong, Cook e Saipe (1979) o formulam como um problema de programação inteira mista, com o objetivo de organizar os pedidos de compra em lotes, de forma a minimizar o tempo total de processamento. O centro de distribuição considerado neste estudo é de um conselho provincial de bebidas alcoólicas Canadense. Aproximadamente 30 pedidos de compra eram processados diariamente por este centro de distribuição, variando em tamanho de 200 a 6000 itens, sendo que o volume de entrega diário variava entre 18000 a 35000 itens. A partir disto, nota-se a necessidade de considerar um leiaute padronizado para centros de distribuição, no intuito de possibilitar o reuso de metodologias de operação e também a comparação de resultados entre os diferentes estudos existentes na literatura.

Devido às diversas possibilidades para valores de área, distâncias e leiaute de centros de distribuição encontradas na literatura e em contextos reais, Gademann e Velde (2005) propuseram uma definição padrão para estes componentes do OBP. Este padrão foi muito bem aceito pela comunidade científica, uma vez que há diversos trabalhos publicados seguindo este padrão, o que tornou possível a comparação dos métodos propostos. A partir desta versão padrão, também surgiram variações que contemplam características específicas presentes em algumas aplicações, como a versão *online* deste problema (OOBP, sigla para *online order batching problem*) e o OBP integrado ao *picker routing problem* (JOBPRP). Nesta monografia, a versão padrão do OBP foi adotada, e esta é descrita em detalhes no Capítulo 3.

Apesar de existir uma definição padrão para o OBP, havia uma variedade de instâncias não compartilhadas entre pesquisadores, as quais se referiam a cenários reais ou fictícios. Henn et al. (2010), com o propósito de padronizar instâncias para o OBP, geraram um conjunto de 1600 instâncias, incentivando pesquisadores a utilizá-las a fim de possibilitar uma comparação justa dos métodos propostos na literatura, bem como evidenciar a evolução dos mesmos. Estas instâncias são também adotadas nesta monografia. Na sequência, descreve-se como esta versão do OBP foi tratada por diferentes autores na literatura, dada a definição padronizada de ambos,

OBP e conjunto de instâncias.

No passado, para o OBP, há uma predominância de heurísticas construtivas tradicionais propostas como métodos de solução. Estas heurísticas podem ser distinguidas em três grupos: algoritmos baseados em regras de prioridade (*priority rule-based algorithms*), algoritmos de semente (*seed algorithms*) e algoritmos de economia (*savings algorithms*) (WÄSCHER, 2004). Posteriormente, devido ao espaço de soluções do OBP ser muito grande, abordagens utilizando-se meta-heurísticas passaram a ser comuns na literatura. Henn et al. (2010) introduzem abordagens para a solução deste problema através de duas metaheurísticas: a primeira é uma ILS e a segunda uma colônia de formigas baseada em classificação (RBAS, sigla para *rank based ant system*). Foi demonstrado que os métodos propostos são superiores aos métodos de solução clássicos mencionados anteriormente. Ambas abordagens obtiveram resultados semelhantes no quesito qualidade de solução, no entanto a preferência é pela ILS, uma vez que esta metaheurística exigiu menos tempo de processamento para fornecer soluções de qualidade equiparável quando comparada com a RBAS.

Henn e Wäscher (2012) ampliaram o conjunto padrão de instâncias de 1600 para 2560 instâncias. Adicionalmente, foram sugeridas duas abordagens baseadas no princípio de busca tabu (TS, sigla para *tabu search*). A primeira é uma TS clássica, e a segunda é a *attribute-based hill climbing* (ABHC), a qual requer um número menor de parâmetros. Foi demonstrado que ambas as abordagens geram soluções superiores às abordagens existentes anteriormente na literatura. Além disso, o tempo de processamento necessário para ambas abordagens foi menor quando comparado com os outros métodos existentes. Os trabalhos apresentados na sequência também utilizaram as instâncias propostas por Henn e Wäscher (2012).

Öncan (2015) aborda o OBP a partir de três políticas de roteamento diferentes. Porém, neste trabalho é mencionada apenas a versão com a política *s-shape* (descrita na Seção 3.2.1), constante na versão padronizada do OBP e que permite a comparação com os demais trabalhos da literatura. Este foi o primeiro trabalho a apresentar formulações de programação linear inteira mista (PLIM) para a versão padrão do OBP. Além do mais, foi desenvolvido um ILS com *tabu thresholding* (ILST). Nos experimentos realizados, comparou-se o desempenho da ILST com a implementação de ILS proposta por Henn et al. (2010). De acordo com os resultados obtidos nos experimentos computacionais, foi observado que a ILST supera a ILS em termos de qualidade de solução e tempo de processamento.

Koch e Wäscher (2016) introduzem duas implementações de algoritmos genéticos para solução do OBP, a primeira orientada a itens e a segunda orientada a grupos. A aplicação de ambas as implementações resultou em uma redução significativa da duração total da jornada dos coletores de pedidos com tempos de computação considerados razoáveis. No entanto, a aplicação do algoritmo genético orientado a grupos resultou em melhorias maiores que as do algoritmo orientado a itens. Os resultados mostram que o novo algoritmo genético orientado a grupos é preferível para o OBP e que o algoritmo fornece soluções de alta qualidade.

Menéndez et al. (2017) implementaram uma busca de vizinhança variável (VNS, sigla para *variable neighborhood search*) em dois estágios. Nos experimentos computacionais realizados foi possível notar que o método proposto é competitivo quando comparado aos métodos de solução relatados na literatura. Adicionalmente, foi proposto um procedimento para construir soluções iniciais de alta qualidade e diversificadas, com base em uma *elite candidate list* dentro do método de busca local. A solução inicial é então melhorada com a aplicação da VNS em que os métodos de diversificação e de melhoria consideram uma exploração aninhada de duas vizinhanças diferentes, retornando um ótimo local em relação a ambas. Finalmente, o ótimo local é aprimorado com uma estratégia de pós-otimização baseada em uma pesquisa geral de vizinhança variável. No entanto, neste trabalho foi considerado um subconjunto das instâncias consideradas nesta monografia, além de utilizar uma política de roteamento diferente, portanto, seus resultados não serão utilizados para comparação de qualidade com o método proposto.

Žulj, Kramer e Schneider (2018) desenvolveram um algoritmo híbrido de busca adaptativa de grande vizinhança (ALNS, sigla para *adaptive large neighborhood search*) e busca tabu, denotada como ALNS/TS. A vantagem desta abordagem híbrida é que ela combina as capacidades de diversificação do ALNS e a capacidade de intensificação da TS. Em experimentos computacionais, avaliou-se o desempenho da hibridização e foi demonstrado que o ALNS/TS supera claramente a ALNS e TS ambos aplicados isoladamente. Este método é o primeiro ALNS e a primeira metaheurística híbrida projetada para o OBP. Foi comparado o desempenho do ALNS/TS com todos os métodos publicados anteriormente na literatura que utilizaram a versão padrão do OBP (HENN; WÄSCHER, 2012; ÖNCAN, 2015; KOCH; WÄSCHER, 2016).

O ALNS/TS superou todos os métodos em relação à qualidade média da solução e tempo de execução. Para as instâncias mais relevantes, com um número maior de pedidos de compra e maiores capacidades do coletor, se tornam evidentes as vantagens do método quando comparado com os métodos existentes. Além disso, foi gerado um conjunto de instâncias do OBP de grandes dimensões para avaliação do método e disponibilizado para futuras comparações. Essas instâncias baseiam-se na estrutura das instâncias de referência propostas por Henn e Wäscher (2012). Nesta monografia, os resultados obtidos serão comparados com Žulj, Kramer e Schneider (2018), incluindo o conjunto de instâncias de grandes dimensões, considerado o estado da arte atual para a versão padrão do OBP.

Yang, Zhao e Guo (2020) abordam uma variante do OBP, a qual define que um item pode possuir mais que uma localização dentro do centro de distribuição. São propostos dois métodos para o OBP: *tabu search batching algorithm* e *branch-and-price batching algorithm*. O *branch-and-price batching algorithm* é aplicável apenas para problemas de pequenos lotes. A solução inicial para a busca tabu é obtida usando o *greedy seed batching algorithm*. No entanto, para comparação da eficiência do método proposto, este foi adaptado para a versão padrão do OBP, a fim de comparar os resultados com o trabalho correspondente ao estado da arte Žulj, Kramer e Schneider (2018). Em termos de qualidade de solução, o *gap* médio foi de 2,6%, mas

no quesito tempo de processamento o método se mostrou competitivo.

3 Fundamentação teórica

Este capítulo apresenta a definição formal do OBP, bem como o funcionamento das políticas de roteamento adotadas neste trabalho. Por fim, é apresentada a ideia geral da metaheurística ILS, utilizada também neste trabalho.

3.1 *Order batching problem*

O OBP padrão considera um centro de distribuição retangular, com 10 corredores verticais paralelos delimitados por dois corredores horizontais, além de um acesso de entrada e saída do centro de distribuição. Cada corredor vertical se situa entre prateleiras laterais divididas em unidades de armazenamento conhecidas na literatura como *stock keeping unit* (SKU). A localização de cada tipo de produto é representada por um SKU. Cada SKU possui o seu próprio identificador e uma dimensão constante. Para a coleta dos produtos ser possível, o coletor precisa estar no corredor vertical imediatamente à frente dos seus respectivos SKUs. Caso haja dois SKUs, em um mesmo corredor vertical paralelos entre si, é possível realizar a coleta em ambos simultaneamente sem custo adicional. A Figura 3.1 ilustra um leiaute de um centro de distribuição que segue estas especificações.

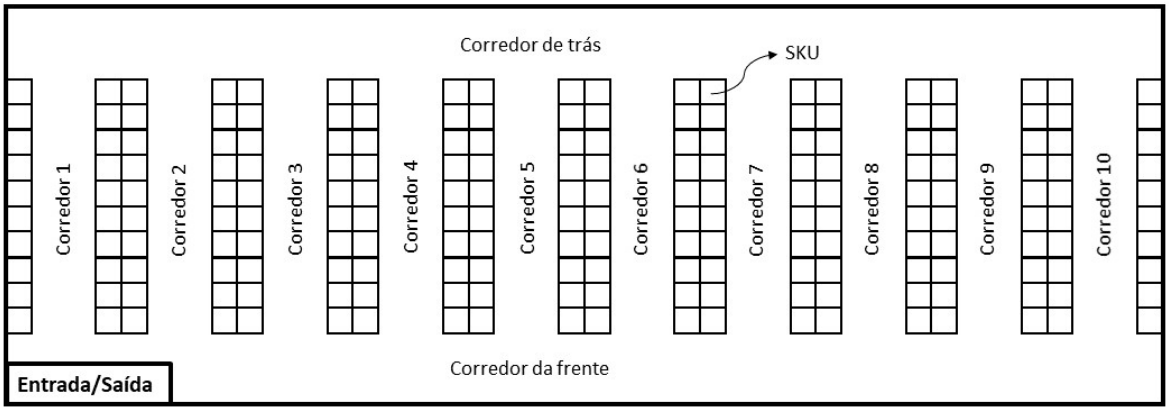


Figura 3.1 – Exemplo de leiaute de um centro de distribuição

Dado um conjunto de pedidos de compra $O = \{O_1, O_2, ..., O_n\}$, sendo cada pedido composto por um conjunto de produtos $O_i = \{p_1, p_2, ..., p_m\}$, $i = 1...m$, armazenados em um centro de distribuição com suas respectivas localizações conhecidas, o objetivo do OBP é agrupar os pedidos de compra em lotes de tal forma que cada lote seja coletado em uma única incursão no centro de distribuição. Ademais, as seguintes regras devem ser respeitadas: (i) a quantidade total de produtos de cada lote deve ser menor ou igual a capacidade do coletor; e (ii) os pedidos

de compra não podem ser fracionados, ou seja, considerar um pedido em um lote implica que todos os produtos que o compõem devem ser coletados na mesma incursão.

O custo de uma solução do OBP é dada pela soma das distâncias percorridas pelo coletor ao coletar todos os lotes formados, levando em consideração uma política fixa de roteamento. Estas distâncias correspondem a distância entre os corredores horizontais, a distância entre um corredor vertical e um corredor horizontal, a distância entre os corredores verticais, a distância entre a entrada/saída do centro de distribuição e o SKU mais próximo, além da distância entre a entrada/saída e o corredor horizontal da frente.

Cada incursão deve ser iniciada e finalizada na entrada/saída do centro de distribuição. Dado um conjunto de pedidos a serem coletados em um centro de distribuição, definir a rota de coleta a ser realizada constitui o PRP. Neste trabalho, como o foco é a resolução do OBP e não do PRP. Assim, foram adotadas políticas de roteamento fixas para esta etapa as quais também foram utilizadas por outros autores para abordagem do OBP padrão.

3.2 Políticas de roteamento

A política de roteamento influencia no custo da solução do OBP, uma vez que é a partir dela que se define a sequência em que cada produto pertencente a um lote será coletado, interferindo na distância percorrida pelo coletor.

Nesta monografia, adotam-se duas políticas de roteamento: *largest gap* (HALL, 1993) e *s-shape* (GOETSCHALCKX; RATLIFF, 1988). Estas estratégias foram utilizadas por Žulj, Kramer e Schneider (2018) viabilizando uma comparação justa do método proposto com o atual estado da arte do OBP padrão. Segundo Koster, Poort e Wolters (1999) a execução destas políticas de roteamento são de simples entendimento, tendo uma baixa curva de aprendizado e uma alta adoção pelos funcionários. Além disso, Žulj, Kramer e Schneider (2018) enfatizam que estas estratégias minimizam o congestionamento nos corredores, viabilizando que diferentes coletas sejam realizadas em paralelo. A seguir o funcionamento destas políticas é apresentado.

3.2.1 *S-shape*

As rotas baseadas na política *s-shape* respeitam as especificações a seguir. O coletor inicia uma incursão a partir da entrada do centro de distribuição, identificando quais são os corredores que possuem produtos a serem coletados, iniciando a coleta pelo corredor mais à esquerda. Todo corredor que possui pelo menos um produto a ser coletado é percorrido integralmente. Já os corredores que não possuem produtos a serem coletados são ignorados pelo coletor. Desta maneira, percorrer cada corredor implica que o coletor acesse alternadamente os corredores frontal e traseiro do centro de distribuição. Caso a incursão pelo corredor mais à direita se inicie pelo corredor frontal do centro de distribuição, o coletor deve percorrer até a localização do último produto a ser coletado e retornar pelo mesmo corredor, não sendo necessário percorrê-lo

integralmente. Após coletar todos os produtos, o coletor deve se direcionar à entrada/saída do centro de distribuição.

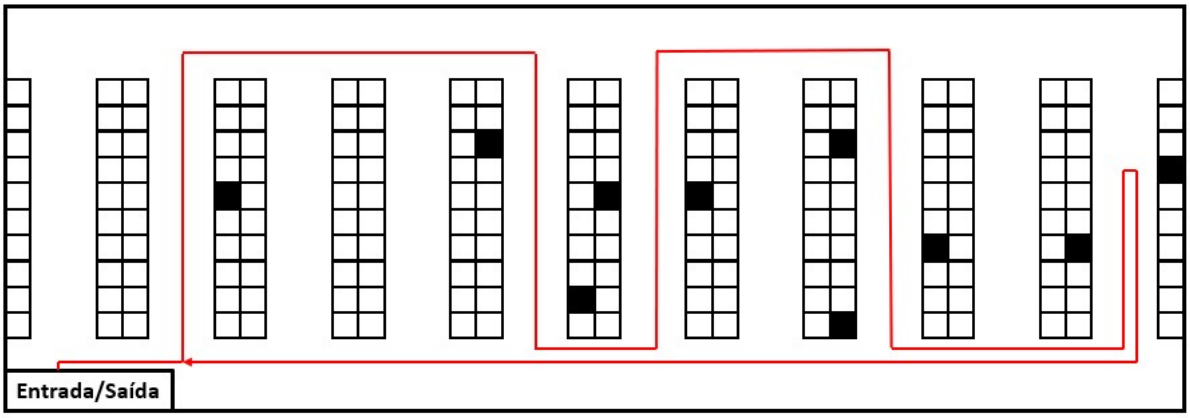


Figura 3.2 – Exemplo de funcionamento da política *s-shape*

A Figura 3.2 ilustra o funcionamento desta política. Nesta figura, os SKUs destacados são aqueles em que há produtos a serem coletados. Note que todos os corredores que possuem produtos a serem coletados são percorridos integralmente, com exceção do corredor mais à direita, que é percorrido somente até a localização do último produto a ser coletado. Caso a incursão do corredor mais à direita fosse feita a partir do corredor de trás do centro de distribuição, este corredor também seria percorrido integralmente. Após a coleta de todos os produtos, o coletor retorna à entrada/saída do centro de distribuição.

3.2.2 *Largest gap*

Nas rotas baseadas na política *largest gap*, o coletor inicia uma incursão pela entrada do centro de distribuição, identificando quais os corredores que possuem produtos a serem coletados, pois os demais são desconsiderados. O corredor mais à esquerda e o corredor mais à direita são percorridos integralmente. Para os demais corredores, deve-se calcular o maior *gap* de forma a evitá-lo. O *gap* pode ser definido pela: (i) distância entre quaisquer pontos de coleta; (ii) distância entre o corredor traseiro e o ponto de coleta mais próximo deste; ou (iii) distância entre o corredor frontal e o ponto de coleta mais próximo deste. Desta forma, o coletor não percorre esta maior lacuna definida pelo *gap*. O corredor mais à esquerda é percorrido integralmente, acessando o corredor traseiro do centro de distribuição. A partir de então o coletor deve acessar os demais corredores, caso o *gap* não corresponda ao caso (ii), fazendo a incursão parcial por cada corredor e retornando ao corredor traseiro. O corredor mais à direita também é percorrido integralmente, acessando o corredor frontal do centro de distribuição. A partir de então o coletor deve acessar os demais corredores, que ainda possuem itens a serem coletados, cujo o *gap* não corresponda ao caso (iii), fazendo a incursão parcial por cada corredor e retornando ao corredor frontal. Após a coleta de todos os produtos o coletor retorna à entrada/saída do centro de distribuição.

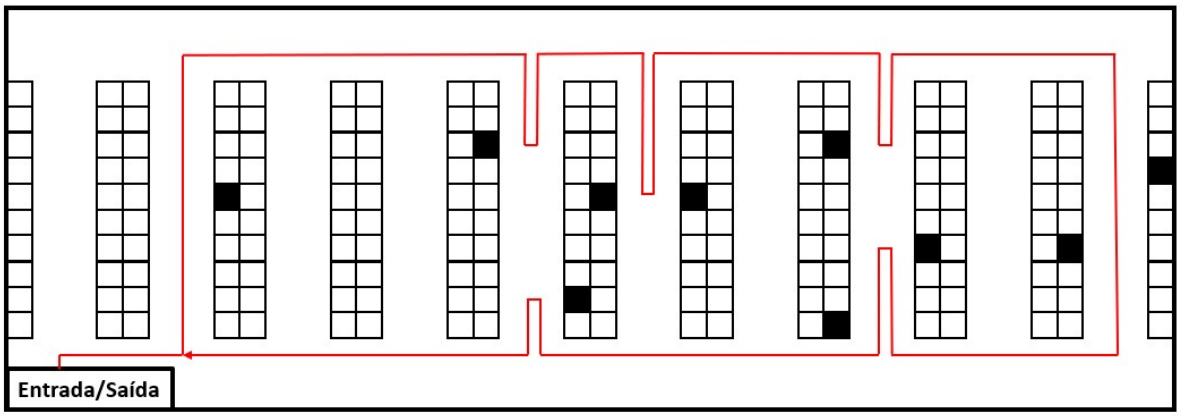


Figura 3.3 – Exemplo de funcionamento da política *largest gap*

A Figura 3.3 ilustra o funcionamento desta política. O corredor mais à esquerda e mais à direita são percorridos integralmente, já os demais são percorridos parcialmente evitando-se percorrer a maior lacuna. Após a coleta de todos os produtos o coletor retorna à entrada/saída do centro de distribuição.

3.3 *Iterated local search*

A ILS (LOURENÇO; MARTIN; STÜTZLE, 2003) é uma metaheurística comumente utilizada para resolução de problemas de otimização combinatória. O grande desafio em resolver problemas desta natureza é que o número de soluções possíveis geralmente é muito grande. Portanto, torna-se imprescindível a utilização de técnicas eficientes para tratar esses problemas na prática.

Como o OBP se enquadra nessa classe de problemas, foi desenvolvida nesta monografia uma ILS adaptada ao problema, a fim de se obter resultados competitivos em relação aos resultados obtidos por outros métodos existentes na literatura. Esta metaheurística possui grande capacidade de explorar o espaço de soluções, uma vez que seus componentes permitem escapar de mínimos locais, caso o problema seja de minimização, sendo possível encontrar uma boa solução em um tempo razoável de processamento.

O sucesso da ILS depende diretamente de seus componentes: busca local, solução inicial, perturbação e critério de aceitação. O funcionamento do método se dá pela amostragem de ótimos locais obtidos pelo processo de intensificação, através da busca local, e diversificação, através da perturbação e pelo critério de aceitação. A intensificação consiste em explorar a região atual de busca, e a diversificação consiste em mudar a região de busca. Os componentes serão descritos a seguir.

3.3.1 Busca local

A busca local consiste em explorar o espaço de busca, definido por todas as soluções possíveis, através de vizinhanças. Uma vizinhança pode ser definida como um conjunto de soluções que podem ser alcançadas entre si através de um determinado movimento.

Dada uma solução, o objetivo de uma busca local é retornar um ótimo local da vizinhança adotada. Um ótimo local é a melhor solução em um subespaço do espaço de busca. O método escolhido para esse procedimento deve ser forte o suficiente para fornecer soluções de boa qualidade e leve o suficiente para que não seja necessário muito recurso computacional para a execução do método.

3.3.2 Solução inicial

A solução inicial pode ser gerada a partir de qualquer método, como a utilização de heurísticas construtivas ou até mesmo utilizando-se a aleatoriedade, desde que as restrições do problema sejam respeitadas. No entanto, soluções de baixa qualidade necessitam de maior esforço computacional para que seja possível aprimorá-las a partir de um método de busca local. Por isto, a escolha de um bom método para geração da solução inicial é mais pertinente, já que este esforço computacional é reduzido.

3.3.3 Perturbação

O objetivo da perturbação é aumentar a diversidade de soluções exploradas. Em um problema de minimização, este método permite que a ILS escape de mínimos locais, uma vez que a busca local não possui esta capacidade, possibilitando explorar outras regiões do espaço de soluções do problema. O método escolhido não deve ser capaz de desfazer uma operação da busca local, uma vez que seu intuito é diversificar a solução a fim de evitar ciclicidade e desperdício de processamento. Adicionalmente, a intensidade da perturbação deve ser forte o suficiente para que seja possível escapar de ótimos locais, e ao mesmo tempo fraca o suficiente para que seja possível guardar características do ótimo local atual.

3.3.4 Critério de aceitação

O critério de aceitação é utilizado para decidir a partir de qual solução continuará a exploração, visto que a cada iteração da ILS um ótimo local é retornado após os procedimentos de perturbação e busca local. Este ótimo local pode corresponder a melhor solução encontrada ou não. É o critério de aceitação que determina se esta solução será considerada ou não na próxima iteração do método.

Um critério de aceitação comumente utilizado para a ILS é aceitar uma solução somente se ela for melhor que a melhor solução encontrada. No entanto, soluções que são piores que a

melhor solução também podem ser aceitas com uma pequena probabilidade. Isto evita a utilização constante da melhor solução e consequentemente uma rápida estagnação das soluções avaliadas (KAMPKE; ARROYO; SANTOS, 2011).

3.3.5 Algoritmo ILS

A ILS é estruturada como apresentado no Algoritmo 1. Primeiramente, gera-se uma solução inicial viável s_0 (linha 2), a qual é submetida ao procedimento de busca local, obtendo-se a solução vizinha s (linha 3). Esta é considerada a melhor solução até este momento. Em seguida, um laço de repetição (linhas 4 a 8) é executado enquanto o critério de parada não for atendido. A partir de então, aplica-se uma perturbação em s (linha 5), obtendo uma solução intermediária s'. Esta é submetida ao método de busca local, produzindo a solução s'' (linha 6). O critério de aceitação determina de qual solução a próxima perturbação será aplicada, s ou s'' (linha 7). Este procedimento repete-se até que o critério de parada seja satisfeito, o qual pode ser limitado, por exemplo, por tempo ou número de iterações.

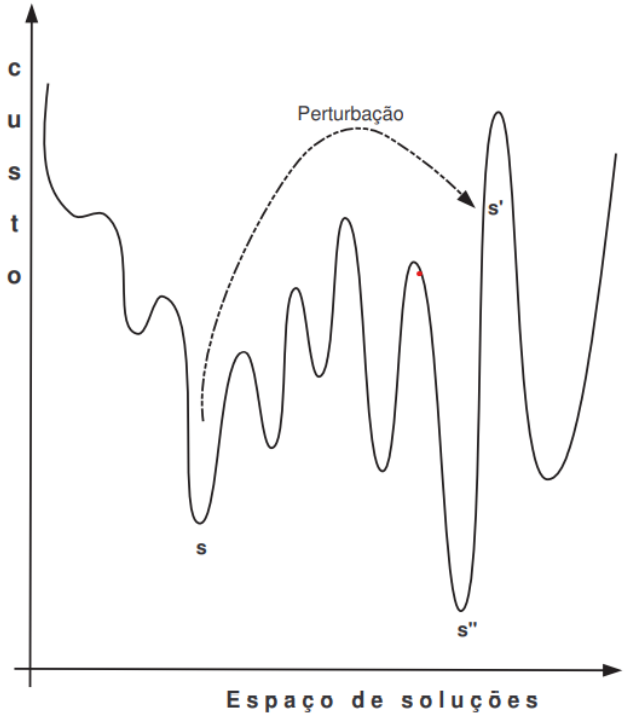
Algoritmo 1: ILS (LOURENÇO; MARTIN; STÜTZLE, 2003).

```
1  início
2      s_0 ← SolucaoInicial();
3      s ← BuscaLocal(s_0);
4      enquanto critério de parada não for satisfeito faça
5          s' ← Perturbacao(s);
6          s'' ← BuscaLocal(s');
7          s ← CriterioAceitacao(s, s'');
8      fim
9      retorna s;
10 fim
```

A Figura 12 ilustra o funcionamento da ILS. Seja um problema de minimização, e dada uma solução s, a qual corresponde a um ótimo local, aplica-se um procedimento de perturbação a fim de escapar deste ótimo local, resultando em uma solução intermediária s'. Apesar de s' não ser uma solução tão boa em relação a s, ao aplicar o método de busca local em s' é produzido um novo ótimo local s''. Caso s'' atenda aos critérios de aceitação, tem-se que a busca prossegue a partir dessa solução.

Saída: Solução atual s

Figura 3.4 – Representação esquemática do funcionamento da ILS.



11

12 Fonte: Souza (2009).

4 Desenvolvimento

Neste capítulo é descrita a implementação da ILS proposta. Para que esta metaheurística forneça bons resultados, é imprescindível um alinhamento estratégico de seus componentes para um funcionamento harmônico do método. Os componentes de geração da solução inicial, busca local, perturbação e critério de aceitação são descritos detalhadamente nas seções seguintes, bem como o critério de parada adotado.

4.1 Solução inicial

Foi adotada a heurística *Clarke and Wright ii* (C&W(ii)) como método de geração da solução inicial. Esta heurística foi desenvolvida primeiramente para solução do problema de roteamento de veículos. No entanto, em Koster, Poort e Wolters (1999), foi adaptada também para solução do OBP.

O funcionamento do método é descrito a seguir. Primeiramente, todos os pedidos de compras são considerados como lotes unitários. Na sequência, é calculada a economia gerada ao se coletar dois lotes simultaneamente em uma mesma incursão, respeitando a capacidade do coletor. Este procedimento é realizado para todos os pares de lotes que podem se unir. A união que resultar na maior economia é efetuada. Com isto, é obtido um novo lote e todo este processo deve se repetir até que não haja mais uniões que geram economia possíveis de serem realizadas.

O Algoritmo 2 apresenta os passos da heurística C&W(ii) aplicada ao OBP. Primeiramente, cria-se um conjunto B de lotes unitários a partir do conjunto de pedidos de compras (linha 2). Na sequência, é calculada a distância necessária para efetuar a coleta de todos estes lotes (linha 3). Logo após, cria-se uma variável global para armazenar a maior economia, a qual é inicializada com o menor valor possível (linha 4). Em seguida, as combinações dos lotes tomados dois a dois são realizadas, calculando a economia obtida para cada combinação, de acordo com o laço de repetição das linhas $5-20$. O objetivo é unir o par de lotes que resulta na maior economia. Na linha 6 um lote B_i é selecionado do conjunto B, o qual é combinado com todos os demais lotes (linhas $7-19$). O lote B_j representa, a cada iteração do laço da linha 7, um lote do conjunto B que seja diferente de B_i (linha 8). Calcula-se a distância necessária para coletar ambos os lotes B_i e B_j separadamente (linha 9). Depois disto, caso a união de ambos os lotes não viole a restrição de capacidade do coletor, é calculada a distância necessária para coletar o lote resultante desta união (linhas 10 e 11). Logo após, é verificada se esta união gera economia (linha 12). Em caso positivo, esta economia é calculada (linha 13). Posteriormente, verifica-se se a união destes lotes resultam na melhor economia obtida até então (linha 14). Em caso positivo, atualiza-se a variável referente a maior economia e armazenam-se os lotes pertencentes a esta combinação (linhas 15-17). Após a execução do laço de repetição mais externo, verifica-se se houve alguma

combinação que gerou melhora na solução (linha 22). Em caso positivo, a união que resultou na maior economia é realizada (linhas 23-25). Enquanto houver uniões possíveis e que geram economia o algoritmo é chamado recursivamente (linha 26). Caso não haja o conjunto de lotes formado é retornado, interrompendo a execução deste algoritmo (linha 28).

Algoritmo 2: C&W(ii) (KOSTER; POORT; WOLTERS, 1999).

Entrada: Conjunto de pedidos O
Saída: Conjunto de lotes B

```
1  início
2      B ← geraLotes(O);
3      d_t ← calculaDistancia(B);
4      C_maiorEconomia ← −∞;
5      para cada elemento do conjunto B faça
6          B_i ← proximoLote(B);
7          para cada elemento do conjunto B faça
8              B_j ← B_{i+1};
9              d_soma ← calculaDistancia(B_i)+calculaDistancia(B_j);
10             Lote_uniao ← uniao(B_i, B_j);
11             d_uniao ← calculaDistancia(Lote_uniao);
12             se d_uniao<d_soma então
13                 d_economia ← d_soma − d_uniao;
14                 se d_economia>C_maiorEconomia então
15                     C_maiorEconomia ← d_economia;
16                     l_1 ← B_i;
17                     l_2 ← B_j;
18                 fim
19             fim
20         fim
21     fim
22     se C_maiorEconomia ≠ −∞ então
23         excluirLote(l_1, B);
24         excluirLote(l_2, B);
25         incluirLote(Lote_uniao, B);
26         retorna C&W(B);
27     senão
28         retorna B;
29     fim
30     retorna B;
31 fim
```

4.2 Busca local

São propostos dois métodos de busca local. O primeiro considera uma vizinhança baseada no movimento de reinserção de pedidos. Já o segundo considera uma vizinhança baseada no movimento de troca de pedidos. Ambas as vizinhanças são tradicionalmente encontradas na literatura. Para cada iteração da ILS, ambos os métodos são utilizados. Primeiramente aplica-se a busca local por reinserção e na sequência a busca local por troca. Essa estratégia foi adotada devido aos resultados obtidos em experimentos preliminares.

São realizados apenas os movimentos que respeitam as restrições do problema, neste caso a capacidade do coletor, e que resultam em uma solução com custo melhor que a da atual. Caso estas condições não sejam satisfeitas o movimento não é realizado. Além do mais, há um parâmetro λ correspondente ao percentual de aplicação da busca local, o qual determina a intensificação da busca e que será definido a partir de experimentos computacionais. Ambos os procedimentos adotados não exploram toda a vizinhança da solução atual, evitando-se uma pesquisa exaustiva pelo melhor vizinho. Como consequência, não há garantia de que a solução obtida seja um ótimo local. A seguir é apresentado em detalhes cada busca local.

4.2.1 Busca local por reinserção

Esta busca local utiliza-se de movimentos de reinserções para melhorar uma solução. A partir deste procedimento, é possível encontrar soluções vizinhas por meio de remoções de pedidos dado um lote origem e inserção dos mesmos dado um lote destino. A escolha destes lotes é feita aleatoriamente e quantas inserções serão realizadas vai depender do par de lotes em questão. Pois, seja B_i o lote origem e B_j o lote destino, identifica-se quais os pedidos de B_i podem ser inseridos em B_j sem que haja violação na restrição de capacidade do coletor. Estes pedidos compõem a lista de candidatos. Todos os elementos desta lista são avaliados quanto a possibilidade de serem inseridos em B_j.

Características das instâncias são levadas em consideração para guiar esta busca local. Em específico, a similaridade de corredores entre os pedidos de compra. Como um pedido é composto por vários produtos, os quais possuem suas localizações no centro de distribuição conhecidas, é possível saber em quais corredores o coletor deve percorrer para que cada pedido seja coletado integralmente. Esta informação possibilita a ordenação da lista de candidatos, de acordo com a similaridade de corredores entre cada elemento da lista e o conjunto de pedidos que constituem o lote destino. Aqueles que são mais semelhantes possuem maior probabilidade de serem selecionados para uma possível inserção.

O Algoritmo 3 apresenta o pseudocódigo desta busca local. O laço de repetição das linhas $2-18$ é executado até que sejam realizadas λ reinserções. Inicialmente, seleciona-se dois lotes B_i e B_j aleatoriamente (linhas $3-4$). Em seguida, armazena-se em uma lista todos os pedidos de B_i que são possíveis de serem inseridos em B_j, dada a restrição de capacidade do coletor (linha 5).

Já na linha 6, esta lista é ordenada de acordo com a similaridade de corredores entre os elementos de L e os pedidos de B_j. O laço de repetição mais interno percorre todos os elementos de L, com o propósito de efetuar reinserções de pedidos entre B_i e B_j (linhas $7-17$). Na linha 8 define-se qual pedido será removido de B_i, visto que aqueles com maior semelhança de corredores em relação ao lote B_j possuem maior probabilidade de serem selecionados. A reinserção do pedido selecionado é realizada (linhas $9-10$). Verifica-se então se o movimento efetuado resulta em melhora na solução atual e se não é violada a restrição de capacidade do coletor (linha 11). Em caso positivo, o movimento é mantido, visto que este pedido é desconsiderado para as próximas iteração do laço mais interno (linha 12) e a execução do algoritmo avança para a linha 17 (linha 13). Em caso negativo, o movimento é desfeito (linhas $15-16$).

Algoritmo 3: Busca local por reinserção.

Entrada: Conjunto de lotes B e percentual de aplicação λ
Saída: Conjunto de lotes B

```
1  início
2      repita
3          B_i ← loteAleatorio(B);
4          B_j ← loteAleatorio(B);
5          L ← candidatos(B_i, B_j);
6          ordenaSimilaridadeCorredores(L, B_j);
7          repita
8              O_i ← ordemProbabilidade(L);
9              excluiPedido(O_i, B_i);
10             incluiPedido(O_i, B_j);
11             se quantidadeProdutos(B_j) ≤ capacidade do coletor E houve melhora na
                solução atual então
12                 excluiPedido(O_i, L);
13                 continue;
14             senão
15                 excluiPedido(O_i, B_j);
16                 incluiPedido(O_i, B_j);
17             fim
18         até que todos os elementos de L sejam visitados;
19     até realizar λ reinserções;
20     retorna B;
21 fim
```

4.2.2 Busca local por troca

Esta busca local utiliza-se de movimentos de trocas para melhorar uma solução. O funcionamento deste procedimento pode ser descrito da seguinte maneira: selecionam-se dois lotes B_i e B_j aleatoriamente, e um número h, o qual corresponde à quantidade de pedidos do menor lote. Logo, são avaliadas h trocas entre os pedidos de B_i e B_j. Aquelas que não violam a restrição de capacidade do coletor são consideradas para uma possível troca.

De forma análoga ao procedimento anterior, esta busca local também é guiada pela similaridade de corredores. Neste caso, são ordenados todos os pedidos de B_i, de tal forma que os primeiros elementos sejam os mais semelhantes quando comparados com o conjunto de pedidos que constituem B_j. Desta forma, os primeiros elementos também possuem maior probabilidade de serem selecionados para uma possível troca. Já para determinar o pedido de B_j que será selecionado para a troca, utiliza-se a aleatoriedade.

O Algoritmo 4 apresenta o pseudocódigo desta busca local. O laço de repetição das linhas $2-27$ é executando até que sejam realizadas λ trocas. Inicialmente, seleciona-se dois lotes aleatoriamente B_i e B_j (linhas $3-4$). Na sequência, verifica-se qual destes lotes possui menos pedidos e quantos são, armazenando esse valor na variável h (linhas $5-9$). Em seguida, os pedidos de B_i são ordenados de acordo com a similaridade de corredores entre cada um e o conjunto de pedidos que constituem B_j. O laço interno (linhas $11-26$) é executado por h iterações. Na linha 12 define-se o pedido de B_i que será trocado, visto que aqueles com maior semelhança de corredores em relação ao lote B_j possuem maior probabilidade de serem selecionados. Já na linha 13 seleciona-se aleatoriamente um pedido de B_j para a troca. Em seguida, a troca é realizada (linhas $14-17$). Avalia-se se este é um movimento viável e se resultou em melhoria na solução atual (linhas 18). Em caso positivo, o movimento é mantido avançando a execução do algoritmo para a linha 26 (linha 19). Em caso negativo, a troca é desfeita (linhas $21-24$). Em ambos os casos, o par de pedidos avaliados são bloqueados para as próximas h iterações.

Algoritmo 4: Busca local por troca.

Entrada: Conjunto de lotes B e percentual de aplicação λ

Saída: Conjunto de lotes B

```
1  início
2      repita
3          B_i ← loteAleatorio(B);
4          B_j ← loteAleatorio(B);
5          se |B_i| < |B_j| então
6              h ← |B_i|;
7          senão
8              h ← |B_j|;
9          fim
10         ordenaSimilaridadeCorredores(B_i, B_j);
11         repita
12             O_i ← ordemProbabilidade(B_i);
13             O_j ← ordemAleatoria(B_j);
14             excluiPedido(O_i, B_i);
15             excluiPedido(O_j, B_j);
16             incluiPedido(O_j, B_i);
17             incluiPedido(O_i, B_j);
18             se quantidadeProdutos(B_i) ≤ capacidade do coletor E
                quantidadeProdutos(B_j) ≤ capacidade do coletor E houve melhora na
                solução atual então
19                 continue;
20             senão
21                 excluiPedido(O_i, B_j);
22                 excluiPedido(O_j, B_i);
23                 incluiPedido(O_i, B_i);
24                 incluiPedido(O_j, B_j);
25             fim
26         até h;
27     até realizar λ trocas;
28     retorna B;
29 fim
```

4.3 Perturbação

O componente de perturbação adotado modifica a solução atual por meio de movimentos aleatórios de trocas. O método funciona da seguinte maneira: dado o conjunto B que representa

todos os lotes, escolhem-se aleatoriamente dois lotes B_i e B_j e um número q, o qual varia no intervalo (1, 2, ..., $|B_i|$) caso B_j possua mais pedidos de compras que B_i, ou no intervalo (1, 2, ..., $|B_j|$) caso contrário. Este número representa quantas tentativas de trocas serão realizadas entre esses lotes. É permitido apenas uma troca entre cada par de lotes. O percentual de aplicação da perturbação é um parâmetro definido a partir de experimentos computacionais e que representa a intensidade do método, ou seja, quantas trocas devem ser realizadas na solução atual.

Os pedidos de compra que serão selecionados de cada lote para tentativa de troca são definidos de maneira aleatória. A troca será realizada nos casos em que a quantidade de produtos de cada lote não exceda a capacidade do coletor. Caso contrário, o movimento é descartado e será escolhido outro par de pedidos enquanto não atingir as q tentativas de trocas previamente definidas. Caso não se realize nenhuma troca entre B_i e B_j, outro par de lotes é escolhido e este processo se repete até que se efetue a quantidade de trocas definidas pelo percentual de aplicação adotado. Como o objetivo da perturbação é fornecer uma solução diferente que a solução atual, não é analisada a qualidade da solução obtida após a aplicação da perturbação.

O Algoritmo 5 apresenta como é o funcionamento do componente de perturbação adotado. Inicialmente, são selecionados dois lotes B_i e B_j aleatoriamente (linhas $3-4$). Logo após, identifica-se se B_i possui menos pedidos de compras que B_j (linha 5). Em caso positivo, gera-se um número aleatório q no intervalo (1, 2, ..., $|B_i|$) (linha 6). Em caso negativo, gera-se um número aleatório q no intervalo (1, 2, ..., $|B_j|$) (linha 8). Este número q indica quantas trocas serão tentadas entre B_i e B_j (linhas $10-25$). Escolhe-se um pedido de compra de cada lote aleatoriamente (linhas $11-12$). Na sequência, a troca destes pedidos é efetuada (linhas $13-16$). Verifica-se então se este é um movimento viável, ou seja, se a quantidade de produtos de cada lote não exceda a capacidade do coletor (linha 17). Em caso positivo, a troca é mantida, e a partir do comando *break* o laço de repetição mais interno é interrompido (linha 18). Consequentemente, a execução do programa continua na próxima linha após o laço, verificado se as ω trocas previamente definidas foram realizadas (linha 26). Em caso positivo, a execução do método é suspensa, já em caso negativo, um novo par de lotes é selecionado e repete-se todo este processo que foi especificado, retornando-se ao início do laço de repetição mais externo (linha 3). No entanto, note que caso a condição da linha 17 não seja satisfeita, a troca proposta é desfeita (linha $20-23$). Desta forma, verifica-se se a condição do laço mais interno foi satisfeita, ou seja, se houve q tentativas de trocas entre os lotes B_i e B_j (linha 25). Em caso positivo, o processo deve ser repetido enquanto não sejam efetuadas as ω trocas, e enquanto houver trocas possíveis de serem realizadas, retornando a execução do algoritmo para a linha 3. Em caso negativo, o laço mais interno é executado novamente (linhas $10-25$), neste momento selecionando um novo par de pedidos de B_i e B_j para a tentativa de uma nova troca.

Algoritmo 5: Perturbação.

Entrada: Conjunto de lotes B e percentual de aplicação ω
Saída: Conjunto de lotes B

início
  repita
    $B_i \leftarrow$ loteAleatorio(B);
    $B_j \leftarrow$ loteAleatorio(B);
    se $|B_i| < |B_j|$ **então**
      $q \leftarrow$ geraNumero($1, |B_i|$);
    senão
      $q \leftarrow$ geraNumero($1, |B_j|$);
    fim
    repita
      $O_i \leftarrow$ ordemAleatoria(B_i);
      $O_j \leftarrow$ ordemAleatoria(B_j);
      excluiPedido(O_i, B_i);
      excluiPedido(O_j, B_j);
      incluiPedido(O_j, B_i);
      incluiPedido(O_i, B_j);
      se *quantidadeProdutos(*B_i*)* $\leq$ *capacidade do coletor* E
       quantidadeProdutos(B_j*)* $\leq$ *capacidade do coletor* **então**
        break;
      senão
        excluiPedido(O_i, B_j);
        excluiPedido(O_j, B_i);
        incluiPedido(O_i, B_i);
        incluiPedido(O_j, B_j);
      fim
    até q;
  até *realizar* ω *trocas*;
  retorna B;
fim

4.4 Critério de aceitação

O critério de aceitação adotado considera soluções melhores que a atual e também soluções piores até α porcento em relação a melhor solução encontrada (s^*). Esta probabilidade é responsável pela diversificação de soluções e é definida através dos experimentos computacionais. Desta forma, não se aceitam somente soluções que melhoram a solução atual, aceitam-se também

soluções estritamente piores que a melhor solução encontrada. Caso este critério não seja satisfeito a solução é descartada e a exploração sucede a partir de s^*.

O Algoritmo 6 apresenta o pseudocódigo deste componente. Seja s a solução a qual sucede a exploração da ILS, s^* a melhor solução encontrada, e s'' uma solução obtida após perturbar e refinar s, o funcionamento do componente se dá pelo seguinte: verifica-se se o custo da solução s'' é melhor que o da melhor solução encontrada ou pior em até α por cento (linha 2). Em casa positivo, a busca da ILS sucede a partir de s'' (linha 3). Em caso negativo, a busca sucede a partir de s^* (linha 5). Note que é o parâmetro α que define o quão pior s'' pode ser em relação a s^* para que esta solução seja aceita. O valor de α foi determinado a partir de experimentos preliminares em um porcento.

Algoritmo 6: Critério de aceitação.

Entrada: Percentual de aceitação α

```
1 início
2     se f(s'') ≤ (f(s*) × α) então
3         s ← s'' ;
4     senão
5         s ← s*;
6     fim
7 fim
```

4.5 Critério de parada

O critério de parada determina a condição que deve ser satisfeita para interromper a execução da ILS. Isto se deve ao fato de que uma metaheurística não identifica se uma dada solução é um ótimo global ou se está próxima disto. Logo, é de suma importância adotar um critério de parada para se obter um equilíbrio entre convergência e tempo de execução. Neste trabalho um dos critérios adotados é um limite máximo de iterações para a ILS, que forneça bons resultados com tempos de processamentos aceitáveis quando comparados com os resultados existentes na literatura. Este limite será definido a partir de experimentos computacionais.

Entretanto, para instâncias menores, este limite pode gerar desperdício de processamento, uma vez que o método pode convergir com poucas iterações e continuar a ser executado sem melhora no resultado. Diante disto, é adotado um segundo critério de parada, o qual considera um limite de iterações sem melhora em relação a melhor solução encontrada. É necessário que somente um dos critérios definidos seja satisfeito para que a execução do método seja suspensa.

5 Experimentos computacionais

Neste capítulo são apresentados os resultados dos experimentos computacionais realizados. Todos os experimentos foram realizados em um computador com processador *Intel i7 Octa Core* de 2.9 GHz com 16 GB RAM sob o sistema operacional Ubuntu 22.4.1 LTS. Os códigos dos métodos propostos foram escritos em C++, compilados com g++ 11.3.0 e a opção de otimização -O3.

5.1 Experimentos preliminares

A ILS aplicada ao OBP possui três principais parâmetros que influenciam diretamente no desempenho do algoritmo: porcentagem de aplicação de busca local (λ), porcentagem de aplicação de perturbação (ω), número máximo de iterações e número máximo de iterações sem melhoria na solução. Para a escolha dos melhores valores foi utilizado o *irace* (LÓPEZ-IBÁÑEZ et al., 2016), uma ferramenta capaz de fornecer a melhor configuração para um algoritmo dado um conjunto de parâmetros e instâncias. Para os parâmetros foram pré-selecionados alguns valores conforme apresentado na Tabela 5.1, em que a coluna *Parâmetros* indica qual parâmetro foi analisado. Na coluna *Valores* é apresentado o conjunto dos diferentes valores possíveis para cada parâmetro. Foi utilizado 10% do conjunto de instâncias adotado nos experimentos, selecionados aleatoriamente, de forma a representar todo o conjunto.

Tabela 5.1 – Conjunto de parâmetros testados no *irace*.

Parâmetros	**Valores**
Busca Local (%)	60, 80, 100
Perturbação (%)	1, 3, 5, 10
Max iterações	3000, 4000, 5000, 6000
Iterações sem melhoria	800, 1000, 1200, 1500

O *irace* apresentou a seguinte configuração como a de melhor desempenho: Busca Local = 100%, Perturbação = 1%, Max iterações = 4000 e Iterações sem melhoria = 1200.

5.2 Conjuntos de instâncias

Segue-se a convenção de utilização de unidades de comprimento (LU, sigla para *lenght unit*) assim como em Žulj, Kramer e Schneider (2018), para definir as dimensões físicas do centro de distribuição adotado neste trabalho. Adicionalmente, o centro de distribuição possui 10 corredores de 90 SKUs cada, sendo 45 SKUs de cada lado de cada corredor. Logo, no total

são considerados 900 SKUs. Por fim, caracteriza-se um centro de distribuição pelas distâncias internas:

- O comprimento de cada SKU é de 1 LU.
- A distância entre a entrada do centro de distribuição e o primeiro SKU do corredor mais à esquerda é de 1,5 LU.
- A distância entre a extremidade de cada corredor vertical e o corredor horizontal (transição entre ambos) mais próximo é de 1 LU.
- A distância entre dois corredores verticais é de 5 LU.
- A distância entre os dois corredores horizontais é de 47 LU.

As instâncias utilizadas neste trabalho são um subconjunto das instâncias propostas em Henn et al. (2010) e o conjunto proposto em Henn e Wäscher (2012), sendo os mesmos adotados em Žulj, Kramer e Schneider (2018). No conjunto de instâncias de Henn e Wäscher (2012) dois cenários de demanda são ilustrados. No primeiro, considera-se que os produtos possuem demanda distribuída uniformemente, denominado *uniformly distributed demand* (UDD). No segundo, denominado *class-based demand* (CBD), considera-se que os produtos possuem demanda associada às seguintes classes:

- Alta frequência. Todos os produtos são localizados no corredor 1. Nesta classificação, 10% dos produtos correspondem a 52% da demanda total;
- Média frequência. Nesta classificação, 30% dos produtos correspondem a 36% da demanda total. Todos os produtos são localizados nos corredores 2, 3, 4; e
- Baixa frequência. Nesta classificação, 60% dos produtos correspondem a 12% da demanda total. Estes produtos estão localizados em seis corredores, mais especificamente nos corredores de 5 à 10.

Cada classe, com exceção das instâncias de grandes dimensões criadas por Žulj, Kramer e Schneider (2018), é composta por 40 instâncias. Uma classe é definida pela tupla (p, c), sendo p número de pedidos de compra e c a capacidade do coletor. Para cada pedido de compra, o número de produtos é selecionado aleatoriamente no intervalo [5, 25].

Já para as instâncias de grandes dimensões geradas por Žulj, Kramer e Schneider (2018), também baseadas na estrutura das instâncias propostas por Henn e Wäscher (2012), podemos definir as seguintes classes:

- (200, 6);

- (200, 9);
- (200, 12);
- (200, 15);
- (300, 6);
- (400, 6);
- (500, 6); e
- (600, 6).

Para cada pedido de compra, o número de produtos é selecionado aleatoriamente no intervalo [1, 5]. Para este conjunto, cada classe contém 10 instâncias. Todas as instâncias estão disponíveis para *download* [1].

5.3 Comparação com ao estado da arte

Os resultados obtidos pela ILS proposto foram comparados com o método ALNS/TS proposto por Žulj, Kramer e Schneider (2018), cujos resultados são os melhores para os conjuntos de instâncias adotados e representam o estado da arte atualmente. Dada a utilização de componentes de aleatoriedade pelos algoritmos, foram realizadas 10 execuções independentes para cada uma das instâncias. Já em Žulj, Kramer e Schneider (2018) é considerada apenas uma execução. Os resultados detalhados para os experimentos realizados podem ser acessados no *link* [2].

Cada tabela está organizada da seguinte maneira: a coluna *Pedidos* indica a quantidade de pedidos de compra dada cada classe de instância. A coluna *Capacidade* representa a capacidade do coletor. A coluna *MSE* representa o valor da melhor solução encontrada até o momento na literatura. Os valores de MSE sucedidos por $*$ referem-se a soluções ótimas obtidas pelo resolvedor exato Gurobi (Gurobi Optimization, LLC, 2023). O melhor resultado obtido por cada método é representado na coluna *S**. A média dos resultados encontrados nas 10 execuções são representados na coluna *S*, e o tempo de execução médio, expresso em segundos, se encontra na coluna *T*. Para melhor efeito de comparação de resultados, o *gap* (ou distância percentual) entre a melhor solução existente e a melhor solução encontrada por cada método é também apresentado na tabela e calculado conforme a Equação 5.1. Ademais, para que seja possível checar a robustez do método, o desvio padrão entre as soluções encontradas após as 10 execuções, para cada classe de instância, pode ser observado na coluna *desvio*.

$$100 \times \frac{S^* - MSE}{MSE} \tag{5.1}$$

[1] https://www.dropbox.com/sh/13k18o8eslr0n2d/AAASkg8v0fAqVvT0xtfKXAzha?dl=0
[2] https://drive.google.com/drive/folders/1xmz79TZLNSl5tHQcqTALYVAv-wb_iPNB?usp=sharing

Têm-se que os valores apresentados nas colunas MSE e *gap* correspondem a média, uma vez que cada classe é composta por várias instâncias individuais. Já para a coluna T, para cada instância individual, é considerado o tempo médio dada as 10 execuções e apresentado nas tabelas a média destas médias. Além disso, os tempos de execução do ALNS/TS foram normalizados, dado que que diferentes computadores foram usados para realizar os experimentos. Para tornar os tempos comparáveis, foi utilizado a pontuação Passmark (PM, consulte o *link* [3]) de um único núcleo dos processadores usados (pontuação PM do Intel Core i7, 3,5 gigahertz: 2071; pontuação PM do Intel Core i7, 2,9 gigahertz: 2917). Uma comparação totalmente justa dos tempos de execução não é possível, visto que as linguagens de programação e os sistemas operacionais são diferentes. Por fim, na última linha, é apresentado a média dos resultados de cada coluna.

As Tabelas 5.2 e 5.3 apresentam os resultados correspondentes aos subconjuntos de instâncias CBD/*largest gap* e UDD/*largest gap*, respectivamente. Os valores de MSE para esses grupos foram os mesmos obtidos por Žulj, Kramer e Schneider (2018), uma vez que se observou uma discrepância na interpretação da política de roteamento *largest gap* em comparação com outros autores. Isto foi possível visto que as soluções ótimas encontradas eram melhores do que às encontradas na literatura. Para as soluções ótimas, somente a média arredondada foi disponibilizada, não sendo possível obter a informação decimal.

No subconjunto CBD/*largest gap*, o ILS proposto superou o ALNS/TS em relação à solução em 14 das 16 classes, sendo que dez desses resultados se tornaram os novos valores de MSE. No subconjunto UDD/*largest gap*, a ILS obteve melhor desempenho em relação à solução em 13 classes, sendo que dez desses resultados se tornaram os novos valores de MSE na literatura. Em relação ao tempo, considerando a média geral, observou-se uma melhora de aproximadamente 43% e 24% para os respectivos subconjuntos, embora essa seja uma comparação aproximada. Como *gap* máximo dos dois subconjuntos, tem-se para o ALNS/TS o valor de 0,34%, já para a ILS 0,12. Já o *gap* médio do método proposto foi de 0,02% para ambos subconjuntos, enquanto para o ALNS/TS foi de 0,19 e 0,16. O maior valor de desvio padrão representa 0,004% em relação a solução média, o que demonstra a robustez do método.

[3] https://www.passmark.com

Tabela 5.2 – Resultados para as instâncias CBD/*largest gap*.

Pedidos	**Capacidade**	**MSE**	ALNS/TS			ILS				
			S*	***gap***	**T**	**S***	**S**	**desvio**	***gap***	**T**
40	30	6940,00*	6947,45	0,11	3,55	6945,85	6953,29	6,19	0,08	1,97
	45	5048,05	5048,05	0,00	5,68	5049,58	5068,57	16,90	0,03	4,82
	60	3945,18	3945,18	0,00	9,47	3946,20	3965,91	17,10	0,03	7,62
	75	3396,30	3396,78	0,01	13,08	3396,30	3416,56	17,12	0,00	10,63
60	30	10671,00*	10686,75	0,15	11,81	10675,53	10684,97	8,41	0,04	4,90
	45	7367,68	7377,23	0,13	19,96	7367,68	7395,91	21,39	0,00	13,23
	60	5744,63	5749,10	0,08	30,14	5744,63	5782,55	25,56	0,00	20,20
	75	4959,35	4965,15	0,12	36,23	4959,35	4998,12	29,53	0,00	24,70
80	30	13737,00*	13773,78	0,27	23,96	13741,38	13760,48	20,37	0,03	9,90
	45	9537,80	9558,60	0,22	42,79	9537,80	9580,23	29,09	0,00	25,48
	60	7726,43	7752,88	0,34	63,50	7726,43	7781,44	36,01	0,00	36,94
	75	6452,95	6464,03	0,17	83,53	6452,95	6502,49	31,64	0,00	50,21
100	30	17095,00*	17138,08	0,25	47,32	17110,58	17135,41	20,30	0,09	18,22
	45	11797,80	11830,55	0,28	81,51	11797,80	11847,78	37,96	0,00	44,33
	60	9393,08	9422,88	0,32	117,00	9393,08	9445,70	36,95	0,00	65,37
	75	7826,25	7837,40	0,14	157,04	7826,25	7886,37	37,92	0,00	86,45
Média		**8227,40**	**8243,37**	**0,19**	**46,66**	**8229,46**	**8262,86**	**24,53**	**0,02**	**26,56**

Tabela 5.3 – Resultados para as instâncias UDD/*largest gap*.

Pedidos	**Capacidade**	**MSE**	ALNS/TS			ILS				
			S*	***gap***	**T**	**S***	**S**	**desvio**	***gap***	**T**
40	30	9679,00*	9681,50	0,03	3,67	9681,90	9687,59	7,53	0,03	2,51
	45	7111,33	7111,33	0,00	5,83	7113,05	7140,24	22,85	0,02	5,85
	60	5598,13	5602,55	0,08	8,74	5598,13	5625,82	20,88	0,00	9,07
	75	4813,60	4815,23	0,03	12,71	4813,60	4838,74	17,17	0,00	14,15
60	30	14873,00*	14891,73	0,13	12,06	14877,08	14883,72	7,06	0,03	5,87
	45	10381,00	10399,55	0,18	18,26	10381,00	10421,67	28,41	0,00	15,07
	60	8165,83	8184,43	0,23	27,37	8165,83	8213,10	32,92	0,00	23,41
	75	7089,60	7100,58	0,15	33,39	7089,60	7136,89	31,54	0,00	35,53
80	30	19223,00*	19275,45	0,27	25,30	19234,13	19260,36	21,53	0,06	12,62
	45	13536,83	13569,05	0,24	40,64	13536,83	13602,44	45,05	0,00	33,04
	60	11089,00	11121,28	0,29	61,18	11089,00	11147,82	40,98	0,00	50,37
	75	9269,50	9289,35	0,21	77,29	9269,50	9325,87	37,18	0,00	64,62
100	30	23898,00*	23961,75	0,27	52,07	23925,98	23954,59	19,73	0,12	22,49
	45	16904,15	16933,50	0,17	79,42	16904,15	16975,38	49,52	0,00	55,17
	60	13511,18	13541,13	0,22	108,23	13511,18	13597,07	55,79	0,00	79,53
	75	11244,23	11244,23	0,00	139,53	11250,75	11321,19	46,76	0,06	105,52
Média		**11649,21**	**11670,16**	**0,16**	**44,11**	**11652,60**	**11695,78**	**30,30**	**0,02**	**33,43**

As Tabelas 5.4 e 5.5 apresentam, respectivamente, os resultados dos subconjuntos de instâncias CBD/*s-shape* e UDD/*s-shape*. No primeiro subconjunto, o método proposto obteve um melhor desempenho em 11 classes em comparação com ao estado da arte, sendo que 10 desses resultados são os novos valores de MSE na literatura. Em relação ao tempo médio, o ALNS/TS foi superior, tendo gasto aproximadamente 45% do valor gasto pela ILS. Quanto à qualidade da solução, a ILS performou melhor com um *gap* médio de 0,03% contra 0,10%.

Para o segundo subconjunto de instâncias, o método proposto apresentou melhores resultados em 7 classes, sendo que apenas 3 desses se tornaram os novos valores de MSE na literatura. Embora o ALNS/TS tenha apresentado um melhor desempenho nesse subconjunto, o *gap* médio foi bastante próximo, sendo de 0,03% contra 0,08%. Em relação ao tempo médio, mais uma vez houve uma vantagem para o método proposto por Žulj, Kramer e Schneider (2018), que exigiu cerca de 45% do tempo gasto pela ILS. Conclui-se, portanto, que para a política de roteamento *s-shape*, o ALNS/TS possui um tempo de execução menor. No que diz respeito à robustez do método proposto, o desvio padrão máximo considerando estes dois subconjuntos equivale a 0,005% em relação a solução média.

Tabela 5.4 – Resultados para as instâncias CBD/*s-shape*.

Pedidos	**Capacidade**	**MSE**	ALNS/TS			ILS				
			S*	***gap***	**T**	**S***	**S**	**desvio**	***gap***	**T**
20	30	4192,13*	4192,13	0,00	0,21	4192,58	4192,87	0,80	0,01	0,21
	45	2688,90	2688,90	0,00	0,25	2693,08	2701,77	7,67	0,16	0,63
	60	2155,68	2156,43	0,03	0,33	2155,68	2162,95	6,62	0,00	1,12
	75	1742,70	1743,03	0,02	0,41	1742,70	1750,36	10,69	0,00	1,31
40	30	7907,00*	7907,38	0,00	1,39	7911,75	7918,37	5,51	0,06	1,03
	45	5184,25	5184,75	0,01	1,50	5184,25	5220,19	27,14	0,00	2,92
	60	3985,93	3987,95	0,05	1,85	3985,93	4019,31	27,27	0,00	4,65
	75	3243,48	3243,48	0,00	2,20	3248,05	3279,20	24,39	0,14	7,77
50	30	10097,70*	10097,28	0,00	2,51	10101,93	10106,86	5,84	0,04	1,73
	45	6448,80	6461,83	0,20	2,94	6448,80	6497,79	32,00	0,00	6,03
	60	4970,58	4987,50	0,34	3,29	4970,58	5013,50	30,62	0,00	9,25
	75	4054,93	4076,75	0,54	4,07	4054,93	4092,90	27,47	0,00	11,98
60	30	11609,00*	11615,03	0,05	3,82	11611,35	11624,01	11,03	0,02	2,69
	45	7558,00	7566,03	0,11	4,40	7558,00	7619,25	44,27	0,00	8,68
	60	5821,30	5828,20	0,12	5,12	5821,30	5875,25	36,91	0,00	11,94
	75	4731,08	4736,05	0,11	6,18	4731,08	4783,29	35,23	0,00	16,95
Média		**5399,46**	**5404,54**	**0,10**	**2,53**	**5400,75**	**5428,62**	**20,84**	**0,03**	**5,55**

Tabela 5.5 – Resultados para as instâncias UDD/*s-shape*.

Pedidos	**Capacidade**	**MSE**	ALNS/TS			ILS				
			S*	***gap***	**T**	**S***	**S**	**desvio**	***gap***	**T**
40	30	10461,83*	10464,60	0,03	1,32	10464,45	10472,90	10,11	0,03	1,24
	45	6863,55	6863,55	0,00	1,22	6872,58	6917,56	34,49	0,13	3,56
	60	5277,80	5277,80	0,00	1,63	5279,40	5313,61	26,25	0,03	5,48
	75	4272,70	4272,70	0,00	1,87	4276,88	4307,47	22,72	0,10	6,93
60	30	15482,05*	15493,43	0,07	3,64	15487,23	15504,24	13,45	0,03	3,05
	45	10031,50	10031,50	0,00	4,02	10038,73	10105,52	45,97	0,07	9,36
	60	7704,90	7704,90	0,00	4,64	7710,33	7769,27	36,64	0,07	13,91
	75	6288,18	6293,75	0,09	5,35	6288,18	6335,78	30,23	0,00	18,66
80	30	20644,65*	20670,90	0,13	8,58	20657,33	20683,00	16,66	0,06	5,75
	45	13328,45	13328,45	0,00	9,23	13350,50	13447,45	66,98	0,17	19,14
	60	10171,23	10172,60	0,01	11,19	10171,23	10232,28	43,50	0,00	28,80
	75	8226,63	8232,53	0,07	12,22	8226,63	8286,38	38,30	0,00	36,50
100	30	25539,70*	25578,45	0,15	16,04	25574,75	25599,39	19,80	0,14	10,38
	45	16357,45	16357,45	0,00	16,46	16397,78	16495,62	66,42	0,25	27,88
	60	12472,30	12472,30	0,00	20,14	12484,95	12558,42	47,16	0,10	47,13
	75	10151,35	10151,35	0,00	23,25	10161,25	10236,34	50,93	0,10	71,80
Média		**11454,64**	**11460,39**	**0,03**	**8,80**	**11465,13**	**11516,58**	**35,60**	**0,08**	**19,35**

Em Žulj, Kramer e Schneider (2018), foram realizadas modificações nas constantes de leiaute do centro de distribuição a fim de permitir a comparação dos resultados com Öncan (2015). Em particular, considerou-se uma distância de 0,5 LU da entrada do centro de distribuição até o corredor horizontal frontal e uma distância de 1 LU dos corredores horizontais até o corredor vertical mais próximo. Realizadas estas adaptações, foi realizado um novo experimento a fim de comparar os resultados obtidos nos subconjuntos CBD/*s-shape* e UDD/*s-shape*. Estes resultados são apresentados nas Tabelas 5.6 e 5.7.

No subconjunto CBD/*s-shape*, em relação ao *gap* médio, a ILS obteve 0,10% contra 0,17% do ALNS/TS. O método proposto teve melhor desempenho em 12 das 20 classes, obtendo 9 novos valores de MSE na literatura. No subconjunto UDD/*s-shape*, o método ALNS/TS foi superior com um *gap* médio de 0,02% em comparação com 0,16% do método ILS. Além disso, o método superior teve melhor desempenho apenas em 4 classes das 20, sendo estes resultados os novos valores MSE na literatura. Em relação ao desvio padrão, para ambos subconjuntos, o desvio máximo foi de 71,13 o que equivale a 0,004% em relação a solução média, confirmando a robustez da ILS. Para ambos os subconjuntos o ALNS/TS teve um tempo de execução menor, equivalente a 45% e 48%, respectivamente, do tempo gasto pela ILS.

Tabela 5.6 – Resultados para as instâncias *Öncan* CBD/*s-shape*.

Pedidos	**Capacidade**	**MSE**	ALNS/TS			ILS				
			S*	***gap***	**T**	**S***	**S**	**desvio**	***gap***	**T**
20	30	4133,85*	4133,85	0,00	0,22	4134,25	4134,31	0,19	0,01	0,22
	45	2649,95	2649,95	0,00	0,24	2654,78	2665,45	8,96	0,18	0,64
	60	2123,20	2124,00	0,04	0,30	2125,50	2131,41	4,83	0,11	1,07
	75	1716,60	1717,08	0,03	0,32	1717,88	1724,00	7,59	0,07	1,44
40	30	7797,30*	7799,33	0,03	1,12	7799,33	7807,15	8,16	0,03	1,17
	45	5101,20	5107,35	0,12	1,23	5113,53	5148,70	27,43	0,24	2,94
	60	3908,20	3926,35	0,46	1,56	3927,55	3961,84	25,74	0,50	5,31
	75	3172,80	3192,48	0,62	1,82	3196,33	3233,69	24,53	0,74	7,67
60	30	11448,43*	11453,45	0,04	4,23	11452,28	11462,32	9,76	0,03	3,12
	45	7454,70	7465,33	0,14	4,12	7454,70	7513,48	42,42	0,00	8,79
	60	5733,63	5747,38	0,24	4,69	5733,63	5793,73	38,09	0,00	13,76
	75	4664,48	4666,90	0,05	5,54	4664,48	4713,65	32,94	0,00	19,11
80	30	15394,88*	15415,53	0,13	8,94	15407,23	15420,98	10,50	0,08	5,69
	45	9898,68	9912,35	0,14	9,09	9898,68	9960,68	46,27	0,00	16,53
	60	7527,70	7543,93	0,22	9,99	7527,70	7592,09	42,21	0,00	24,31
	75	6119,35	6129,95	0,17	12,84	6119,35	6172,97	33,52	0,00	35,52
100	30	18850,65*	18875,78	0,13	15,43	18869,75	18892,62	19,90	0,10	10,13
	45	12081,33	12105,63	0,20	16,47	12081,33	12166,36	53,17	0,00	30,33
	60	9251,43	9284,55	0,36	19,02	9251,43	9323,60	48,96	0,00	40,67
	75	7535,15	7556,80	0,29	19,77	7535,15	7602,66	46,72	0,00	55,13
Média		**7328,17**	**7340,40**	**0,17**	**6,85**	**7333,24**	**7371,08**	**26,59**	**0,10**	**14,18**

Tabela 5.7 – Resultados para as instâncias *Öncan* UDD/*s-shape*.

Pedidos	**Capacidade**	**MSE**	ALNS/TS			ILS				
			S*	***gap***	**T**	**S***	**S**	**desvio**	***gap***	**T**
20	30	5565,18*	5565,18	0,00	0,27	5568,98	5570,06	1,24	0,07	0,24
	45	3486,88	3486,88	0,00	0,28	3515,13	3527,65	11,51	0,81	0,68
	60	2738,90	2738,90	0,00	0,34	2744,75	2752,48	7,39	0,21	1,08
	75	2227,28	2227,28	0,00	0,39	2231,48	2237,41	7,77	0,19	1,34
40	30	10293,95*	10296,63	0,03	1,21	10305,63	10310,74	7,02	0,11	1,21
	45	6744,45	6744,45	0,00	1,31	6770,30	6811,01	33,53	0,38	3,35
	60	5187,08	5187,08	0,00	1,55	5195,28	5231,51	26,83	0,16	5,52
	75	4199,85	4199,85	0,00	1,68	4209,40	4242,97	25,00	0,23	7,14
60	30	15233,78*	15241,33	0,05	3,72	15254,05	15267,91	11,91	0,13	3,02
	45	9877,40	9877,40	0,00	4,04	9879,15	9946,37	49,20	0,02	9,04
	60	7583,83	7583,83	0,00	5,00	7592,43	7646,93	36,10	0,11	14,92
	75	6192,70	6196,35	0,06	5,69	6192,70	6236,85	30,05	0,00	20,83
80	30	20316,38*	20321,80	0,03	8,88	20346,38	20365,79	15,29	0,15	6,02
	45	13139,48	13139,48	0,00	9,49	13153,23	13240,17	61,32	0,10	18,39
	60	10007,80	10013,30	0,05	11,15	10007,80	10072,81	43,61	0,00	30,75
	75	8103,28	8113,95	0,13	12,16	8103,28	8158,37	37,82	0,00	38,45
100	30	25132,30*	25168,23	0,14	16,22	25180,10	25205,42	18,04	0,19	10,17
	45	16101,40	16101,40	0,00	16,80	16129,85	16237,69	71,13	0,18	31,40
	60	12277,13	12277,13	0,00	20,23	12293,83	12362,28	45,04	0,14	46,39
	75	10001,35	10001,65	0,00	23,60	10001,35	10076,52	50,58	0,00	72,09
Média		**9720,52**	**9724,10**	**0,02**	**7,20**	**9733,75**	**9775,05**	**29,52**	**0,16**	**16,10**

Para as instâncias de grande dimensão, o método ILS obteve resultados superiores tanto em qualidade de solução quanto em tempo de execução. Em todas as classes de instâncias, o método proposto obteve desempenho melhor, gerando novos valores de MSE para este subconjunto. Além disso, o tempo médio gasto pela ILS foi de apenas 0,13% do tempo médio gasto pelo ALNS/TS. O desvio padrão máximo foi de 0,001%, o que confirma novamente a robustez da ILS. Estes resultados indicam que o método proposto é mais recomendado para cenários reais, nos quais o número de pedidos de compra tende a aumentar. Estes resultados estão apresentados na Tabela 5.8.

Tabela 5.8 – Resultados para as instâncias *large* UDD/*s-shape*.

Pedidos	**Capacidade**	**MSE**	ALNS/TS			ILS				
			S*	***gap***	**T**	**S***	**S**	**desvio**	***gap***	**T**
200	6	18358,20	18417,20	0,32	158,88	18358,20	18409,11	39,99	0,00	37,85
	9	12805,50	12917,30	0,87	180,47	12805,50	12916,70	60,86	0,00	63,58
	12	10520,20	10681,90	1,54	212,61	10520,20	10621,05	63,44	0,00	86,42
	15	8834,30	9024,80	2,16	231,06	8834,30	8958,95	75,61	0,00	107,17
300	6	28277,50	28461,50	0,65	542,38	28277,50	28352,35	54,31	0,00	90,55
400	6	37139,50	37479,40	0,92	1251,95	37139,50	37240,17	67,34	0,00	174,92
500	6	46209,70	46689,60	1,04	2430,09	46209,70	46368,12	95,86	0,00	275,70
600	6	55362,00	55888,60	0,95	4018,90	55362,00	55522,36	98,89	0,00	370,50
Média		**27188,36**	**27445,04**	**1,06**	**1128,29**	**27188,36**	**27298,60**	**69,54**	**0,00**	**150,84**

5.3.1 Análise Estatística

Foi conduzido o teste de *Shapiro-Wilk* (SHAPIRO; WILK, 1965) para verificar a normalidade dos dados em todo o conjunto de instâncias utilizadas nesta monografia. Os resultados indicaram que os dados são distribuídos normalmente, com um intervalo de confiança de 95%. A seguir os resultados do teste para cada subconjunto de instâncias e para cada método.

- CBD/*largest gap*:
  ILS: $W = 0,93608; p\text{-}value = 0,3038$
  ALNS/TS: $W = 0,93627; p\text{-}value = 0,3058$
- UDD/*largest gap*:
  ILS: $W = 0,94068; p\text{-}value = 0,3574$
  ALNS/TS: $W = 0,94063; p\text{-}value = 0,3567$
- CBD/*s-shape*:
  ILS: $W = 0,93424; p\text{-}value = 0,2843$
  ALNS/TS: $W = 0,93473; p\text{-}value = 0,2894$
- UDD/*s-shape*:
  ILS: $W = 0,91343; p\text{-}value = 0,1322$
  ALNS/TS: $W = 0,91307; p\text{-}value = 0,1304$
- *Öncan* CBD/*s-shape*:
  ILS: $W = 0,92128; p\text{-}value = 0,1049$
  ALNS/TS: $W = 0,92083; p\text{-}value = 0,1028$

- *Öncan* UDD/*s-shape*:
  ILS: $W = 0,92114; p\text{-}value = 0,1042$
  ALNS/TS: $W = 0,92127; p\text{-}value = 0,1048$

- *large* UDD/*s-shape*:
  ILS: $W = 0,90898; p\text{-}value = 0,3469$
  ALNS/TS: $W = 0,90718; p\text{-}value = 0,3347$

Seja o nível de significância predeterminado de 0,05. O *p-value* maior que o nível de significância implica que os dados são distribuídos normalmente. Logo, é viável a aplicação de métodos paramétricos para a tomada de decisão sobre a rejeição ou não da hipótese nula e, portanto, tornando possível uma avaliação estatística da superioridade de um método sobre o outro. Para comparar estatisticamente os métodos considerados, formulam-se as seguintes hipóteses:

- Hipótese nula: Os resultados não são significativamente diferentes.
- Hipótese alternativa: Os resultados são significativamente diferentes.

O teste paramétrico *Student's t-test* (STUDENT, 1908) foi aplicado para verificar se existe uma diferença significativa entre os resultados reportados pelos métodos comparados. A seguir os resultados do teste para cada subconjunto de instâncias.

- CBD/*largest gap*:
  O teste indicou que há diferença significativa entre os resultados comparados (t = -4,4644; df = 15; p = 0,0002272),

- UDD/*largest gap*:
  O teste indicou que há diferença significativa entre os resultados comparados (t = -4,6788; df = 15; p = 0,0001484),

- CBD/*s-shape*:
  O teste indicou que há diferença significativa entre os resultados comparados (t = -1,9217; df = 15; p = 0;03693),

- UDD/*s-shape*:
  O teste indicou que não há diferença significativa entre os resultados comparados (t = 1,4713; df = 15; p = 0,9191),

- *Öncan* CBD/*s-shape*:
  O teste indicou que não há diferença significativa entre os resultados comparados (t = 0,51652; df = 19; p = 0,6943),

- *Öncan* UDD/*s-shape*:
  O teste indicou que não há diferença significativa entre os resultados comparados (t = 3,8985; df = 19; p = 0,9995),

- *large* UDD/*s-shape*:
  O teste indicou que há diferença significativa entre os resultados comparados (t = -4,2087; df = 7; p = 0,001996),

Para os subconjuntos em que há diferença significativa entre os resultados comparados o método proposto foi superior em relação a qualidade da solução. Destes, somente no subconjunto CBD/*s-shape* foi observado um tempo de execução inferior do método proposto em comparação com o método ALNS/TS. Esses resultados confirmam que a ILS é competitivo superando o estado da arte anterior, estabelecendo-se como ao novo estado da arte.

6 Conclusão

Nesta monografia foi proposta uma abordagem para solução do *order batching problem*. O OBP é um problema provado NP-difícil, e portanto sua resolução possui grande relevância teórica. Ademais, este problema modela uma situação prática comum nos centros de distribuição que impacta diretamente na redução de custos deste setor. Logo, são imprescindíveis métodos eficientes para tratar tal. O OBP consiste em minimizar a distância percorrida para coletar um conjunto de pedidos de compra, visto que cada pedido é composto por um conjunto de produtos. Estes pedidos devem ser agrupados em subconjuntos denominados lotes. A quantidade de produtos de cada lote não deve violar a capacidade máxima do coletor, responsável por coletá-los. Adicionalmente, cada lote deve ser coletado em uma única incursão, não sendo possível fracioná-lo. Neste trabalho foi realizada uma revisão bibliográfica sobre o tema e proposta uma implementação da metaheurística busca local iterada com componentes personalizados para o OBP. O método proposto foi descrito em detalhes no intuito de gerar embasamento teórico. Através de experimentos computacionais, realizou-se uma comparação da eficiência do método proposto em relação ao estado da arte atual. Utilizou-se o teste paramétrico *Student's t-test* para avaliar se algum dos métodos é superior em relação ao outro. Em alguns subconjuntos, o resultado do teste indicou que não há diferença significativa nos resultados obtidos, porém, em outros subconjuntos, o método proposto apresentou uma qualidade média de solução superior. Especificamente, para a política de roteamento *largest gap* e para instâncias de grande dimensão, o método ILS demonstrou vantagens claras em relação ao método ALNS/TS, tanto em termos de qualidade de solução quanto em tempo de execução. Destaca-se que, para instâncias de grande dimensão, o método proposto apresentou desempenho superior em todas as classes de instâncias, alcançando novas melhores soluções encontradas na literatura. Adicionalmente, a robustez do método proposto foi comprovada para todos os subconjuntos, utilizando-se o desvio padrão. Portanto, a superioridade do método proposto em relação ao ALNS/TS foi evidenciada, o que o torna o novo estado da arte.

Referências

ALVES, C. S.; CHAVES, R. P.; PENTEADO, I. M.; COSTA, S. A. D. *A Importância da Logística para o e-Commerce: O Exemplo da Amazon. com*. 2005.

ARMSTRONG, R. D.; COOK, W. D.; SAIPE, A. L. Optimal batching in a semi-automated order picking system. *Journal of the operational research society*, Taylor & Francis, v. 30, n. 8, p. 711–720, 1979.

FISHER. *Warehouse Operations of Service Wholesale Druggists*. Bureau of Business Research, College of Commerce and Administration, Ohio State Univ., 1948. (Bureau of Business Research monograph). Disponível em: <https://books.google.com.br/books?id=bkUUAQAAMAAJ>.

FRAZELLE, E. H. *World-class warehousing and material handling*. McGraw-Hill Education: McGraw-Hill Education, 2016.

GADEMANN, N.; VELDE, S. Order batching to minimize total travel time in a parallel-aisle warehouse. *IIE transactions*, Taylor & Francis, v. 37, n. 1, p. 63–75, 2005.

GOETSCHALCKX, M.; RATLIFF, H. D. Order picking in an aisle. *IIE transactions*, Taylor & Francis, v. 20, n. 1, p. 53–62, 1988.

Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2023. Disponível em: <https://www.gurobi.com>.

HALL, R. W. Distance approximations for routing manual pickers in a warehouse. *IIE transactions*, Taylor & Francis, v. 25, n. 4, p. 76–87, 1993.

HENN, S.; KOCH, S.; DOERNER, K. F.; STRAUSS, C.; WÄSCHER, G. Metaheuristics for the order batching problem in manual order picking systems. *Business Research*, Springer, v. 3, n. 1, p. 82–105, 2010.

HENN, S.; WÄSCHER, G. Tabu search heuristics for the order batching problem in manual order picking systems. *European Journal of Operational Research*, Elsevier, v. 222, n. 3, p. 484–494, 2012.

JARVIS, J. M.; MCDOWELL, E. D. Optimal product layout in an order picking warehouse. *IIE transactions*, Taylor & Francis, v. 23, n. 1, p. 93–102, 1991.

KAMPKE, E. H.; ARROYO, J. E. C.; SANTOS, A. G. dos. Grasp reativo e iterated local search aplicado ao problema de programação de tarefas em máquinas paralelas com tempos de preparação dependentes da sequência e de recursos. In: . XXXI Congresso da Sociedade Brasileira de Computação (CSBC'2011): XXXI Congresso da Sociedade Brasileira de Computação (CSBC'2011), 2011.

KEARNEY, E.; KEARNEY, A. Excellence in logistics 2004. *European Logistics Association, Brussels*, 2004.

KOCH, S.; WÄSCHER, G. A grouping genetic algorithm for the order batching problem in distribution warehouses. *Journal of Business Economics*, Springer, v. 86, n. 1, p. 131–153, 2016.

KOSTER, M. D.; POORT, E. S. Van der; WOLTERS, M. Efficient orderbatching methods in warehouses. *International Journal of Production Research*, Taylor & Francis, v. 37, n. 7, p. 1479–1504, 1999.

KOSTER, R. d.; ROODBERGEN, K. J.; VOORDEN, R. v. Reduction of walking time in the distribution center of de bijenkorf. In: *New trends in distribution logistics*. New trends in distribution logistics: Springer, 1999. p. 215–234.

LÓPEZ-IBÁÑEZ, M.; DUBOIS-LACOSTE, J.; CÁCERES, L. P.; BIRATTARI, M.; STÜTZLE, T. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, Elsevier, v. 3, p. 43–58, 2016.

LOURENÇO, H. R.; MARTIN, O. C.; STÜTZLE, T. Iterated local search. In: *Handbook of metaheuristics*. Handbook of metaheuristics: Springer, 2003. p. 320–353.

MENÉNDEZ, B.; PARDO, E. G.; ALONSO-AYUSO, A.; MOLINA, E.; DUARTE, A. Variable neighborhood search strategies for the order batching problem. *Computers & Operations Research*, Elsevier, v. 78, p. 500–512, 2017.

Movimento Compre&Confie, ABCOMM. *Faturamento do e-commerce cresce 56,8% neste ano e chega a R$ 41,92 bilhões*. 2022. Disponível em: <https://abcomm.org/noticias/faturamento-do-e-commerce-cresce-568-neste-ano-e-chega-a-r-4192-bilhoes/>. Acesso em: 13 de outubro de 2022.

ÖNCAN, T. Milp formulations and an iterated local search algorithm with tabu thresholding for the order batching problem. *European Journal of Operational Research*, Elsevier, v. 243, n. 1, p. 142–155, 2015.

SHAPIRO, S. S.; WILK, M. B. An analysis of variance test for normality (complete samples). *Biometrika*, JSTOR, v. 52, n. 3/4, p. 591–611, 1965.

SILVA, W. M. d.; MORAIS, L. A. d.; FRADE, C. M.; PESSOA, M. F. Digital marketing, e-commerce and pandemia: a bibliographic review on the brazilian panorama. *Research, Society and Development*, v. 10, n. 5, p. e45210515054, May 2021. Disponível em: <https://rsdjournal.org/index.php/rsd/article/view/15054>.

SOUZA, M. d. F. R. d. *O comércio eletrônico durante a pandemia do coronavírus: uma análise acerca do grau de confiabilidade dos consumidores*. Dissertação (B.S. thesis), 2022.

SOUZA, M. J. F. *Inteligência Computacional para Otimização, Notas de aula, 2009/1*. Ouro Preto, 2009.

STUDENT. The probable error of a mean. *Biometrika*, JSTOR, p. 1–25, 1908.

TOMPKINS, J. A.; WHITE, J. A.; BOZER, Y. A.; TANCHOCO, J. M. A. *Facilities planning*. John Wiley & Sons: John Wiley & Sons, 2010.

WÄSCHER, G. Order picking: a survey of planning problems and methods. In: *Supply chain management and reverse logistics*. Supply chain management and reverse logistics: Springer, 2004. p. 323–347.

YANG, P.; ZHAO, Z.; GUO, H. Order batch picking optimization under different storage scenarios for e-commerce warehouses. *Transportation Research Part E: Logistics and Transportation Review*, Elsevier, v. 136, p. 101897, 2020.

ŽULJ, I.; KRAMER, S.; SCHNEIDER, M. A hybrid of adaptive large neighborhood search and tabu search for the order-batching problem. *European Journal of Operational Research*, Elsevier, v. 264, n. 2, p. 653–664, 2018.