Robson Ricardo Costa Vieira

Métodos Heurísticos para Solução do *Order Batching Problem*

Ouro Preto

2018

Robson Ricardo Costa Vieira

Métodos Heurísticos para Solução do *Order Batching Problem*

Texto de qualificação submetido ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Ouro Preto como requisito para obtenção do título de Mestre em Ciência da Computação.

Universidade Federal de Ouro Preto - UFOP

Departamento de Computação

Programa de Pós-Graduação em Ciência da Computação

Orientador: Marco Antonio Moreira de Carvalho

Ouro Preto

2018

Robson Ricardo Costa Vieira
Métodos Heurísticos para Solução do *Order Batching Problem*/ Robson Ricardo Costa Vieira. – Ouro Preto, 2018-
66 p. : il. (algumas color.) ; 30 cm.

Orientador: Marco Antonio Moreira de Carvalho

Dissertação (Mestrado) – Universidade Federal de Ouro Preto - UFOP
Departamento de Computação
Programa de Pós-Graduação em Ciência da Computação, 2018.

1. Palavra-chave1. 2. Palavra-chave2. 2. Palavra-chave3. I. Carvalho, Marco Antonio Moreira de. II. Universidade Federal de Ouro Preto. III. Departamento de Computação. IV. Título

Robson Ricardo Costa Vieira

Métodos Heurísticos para Solução do *Order Batching Problem*

Texto de qualificação submetido ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Ouro Preto como requisito para obtenção do título de Mestre em Ciência da Computação.

Trabalho aprovado. Ouro Preto, 22 de novembro de 2018:

Marco Antonio Moreira de Carvalho
Orientador

Puca Huachi Vaz Penna
Universidade federal de Ouro Preto

Ouro Preto
2018

Resumo

O *Order Batching Problem* (OBP) é um problema de otimização com aplicações práticas que consiste em minimizar a distância percorrida para coletar um conjunto de pedidos de compra em um centro de distribuição de produtos. Para este fim, os pedidos de compra devem ser agrupados em subconjuntos denominados lotes e posteriormente coletados por um único coletor, que possui uma capacidade máxima para realizar esta coleta. O OBP é um problema $\mathcal{NP}$-difícil, representando um desafio tanto para a área acadêmica quanto para as corporações. Este trabalho apresenta a utilização da metaheurística Busca Local Iterada (ou ILS, do inglês *Iterated Local Search*) para abordagem ao OBP. Especificamente, trata-se a etapa de agrupamento, ao passo que a etapa de roteamento utiliza uma política fixa. Os experimentos computacionais preliminares consideraram 2560 instâncias da literatura e compararam os resultados da abordagem proposta aos resultados mais expressivos publicados recentemente na literatura, dentre eles o atual estado da arte. Embora se tratem de resultados preliminares, o ILS demonstrou alguns bons resultados, tendo superado o atual estado da arte em dois subconjuntos de instâncias. Nas demais instâncias, o ILS obteve resultados inferiores, demonstrando que ainda não é competitivo. Sugestões de trabalhos futuros para que este patamar seja atingido são apresentadas.

Palavras-chave: Busca Local Iterada. Order Batching Problem. Agrupamento.

Abstract

The Order Batching Problem (OBP) is an optimization problem with practical applications that consists in minimizing the distance traveled to collect a set of orders in a product distribution center. To this end, orders must be grouped into subsets called batches and then collected by a single picker, which has a maximum capacity to perform such collection. OBP is an $\mathcal{NP}$-hard problem, posing a challenge for both academia and corporations. This work presents the application of the Iterated Local Search (or ILS) metaheuristic to approach the OBP. Particularly, the batching step is addressed, as the routing step uses a fixed policy. Preliminary computational experiments considered 2560 instances of the literature and compared the results of the proposed approach with the most expressive results recently published in the literature, including the current state of the art. Although the results are preliminary, the ILS has shown some good results, outperforming the current state of the art in two subsets of instances. In the other instances, the ILS obtained inferior results, demonstrating that in the current form it is not competitive. Suggestions for future work to reach competitiveness are presented.

Keywords: Iterated Local Search. Order Batching Problem. Batching.

Declaração

Este documento é resultado de meu próprio trabalho, exceto onde referência explícita é feita ao trabalho de outros, e não foi submetida para outra qualificação nesta nem em outra universidade.

Robson Ricardo Costa Vieira

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ABHC	*Attribute-Based Hill Climber*
ALNS	*Adaptive Large Neighborhood Search*
C&W	*Clarke & Wright*
COG	*Centre Of Gravity*
FCFS	*First-Come, First-Served*
GB	*Gigabyte*
GHZ	*Giga-Hertz*
ILS	*Iterated Local Search*
ILST	*Iterated Local Search Algorithm with Tabu*
LNS	*Large Neighborhood Search*
LST	*Local Search Algorithm with Tabu Thresholding*
LU	*Length Unit*
MAA	*Minimum Additional Aisle*
OBP	*Order Batching Problem*
RAM	*Random Access Memory*
RBAS	*Rank-Based Ant System*
SKU	*Stock Keeping Unit*
TS	*Tabu Search*
VNS	*Variable Neighborhood Search*

Sumário

1 Introdução

Os *centros de distribuição* são instalações utilizadas para armazenar e distribuir os mais diversos tipos de produtos comercializados por empresas. Estes possuem um papel fundamental na logística de uma corporação, possibilitando 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.

Entre as atividades internas recorrentes em um centro de distribuição, a *localização* e *coleta* dos produtos armazenados se destacam e são consideradas fatores chave para a eficiência da gestão logística. O processo de localização consiste em determinar a localização física de um determinado produto no centro de distribuição, seja para armazenar este produto ou seja para coletá-lo. O processo de coleta (ou *picking*) consiste em, dada a especificação de um conjunto de pedidos de compra, coletar os produtos específicos localizados no centro de distribuição.

A coleta de produtos é feita com o auxílio de um coletor, que pode ser um dispositivo operado manualmente ou um dispositivo autônomo, seja passivo (como uma esteira) ou ativo (como um robô). Excetuando-se os coletores passivos, há uma capacidade para o armazenamento de produtos pelo coletor, usualmente dada pela quantidade máxima de produtos. A operação do coletor usualmente consiste em entrar no centro de distribuição, realizar uma incursão pelos corredores de prateleiras para coletar produtos e após completar a coleta de todos os produtos, o mesmo se retira do centro de distribuição.

Apesar das inovações tecnológicas empregadas por grandes empresas varejistas, de acordo com Grosse et al. (2014), muitas organizações na Europa Ocidental utilizam centros de distribuição com coleta e separação de pedidos de compra realizados manualmente, devido à facilidade humana de adaptação às mudanças ou imprevistos que podem ocorrer em tempo real. É razoável imaginar que o mesmo ocorra em diferentes continentes.

Na intuito de minimizar os custos associados à coleta de produtos, os pedidos de compra são agrupados em lotes (ou *batches*) de pedidos similares entre si, em um processo denominado agrupamento (ou *batching*). Um único coletor é responsável por coletar todos os produtos de um mesmo lote, em uma única incursão (ou *viagem*) no centro de distribuição. Uma viagem é iniciada na entrada do centro de distribuição, e termina quando o coletor recolher todos os produtos dos pedidos do lote que lhe foi designado e regressar à entrada do centro de distribuição.

Antes da popularização da internet, era difícil para as organizações, mesmo de grande porte, atender todo o território nacional e muito mais raro, atender o mercado internacional. Porém, em tempos atuais, a tecnologia abriu um novo nicho no mercado, o *e-commerce*. Esse novo canal de atendimento foi muito bem aceito pelo mercado, devido a comodidade, preços

atrativos e um aumento da concorrência que permite que o consumidor tenha mais opções.

De acordo com Shamlaty (2000) e Ghiani, Laporte e Musmanno (2004), o 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. As organizações necessitaram se adaptar para se destacarem, de maneira que tornou-se necessário um aumento de eficiência e redução de preços para conseguirem atender a esta nova realidade.

Costa et al. (2004) enfatiza 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. Dentre várias iniciativas possíveis, ressalta-se o posicionamento estratégico dos centros de distribuição e sua gestão otimizada para atender à demanda dos clientes de forma consistente e com um custo viável.

Segundo Petersen (1997) o *leiaute* do centro de distribuição é um dos fatores que influenciam o desempenho e a eficiência da coleta dos produtos referentes aos pedidos. O trabalho contempla um estudo para o *leiaute* de um centro de distribuição que possui n corredores verticais e dois corredores horizontais. Os corredores verticais se conectam aos corredores horizontais, e em cada corredor vertical tem-se prateleiras do lado esquerdo e do lado direito. Essas prateleiras são compostas por unidades de armazenamento conhecidas na literatura como *Stock Keeping Unit* (SKU). A Figura 1 apresenta um exemplo deste leiaute.

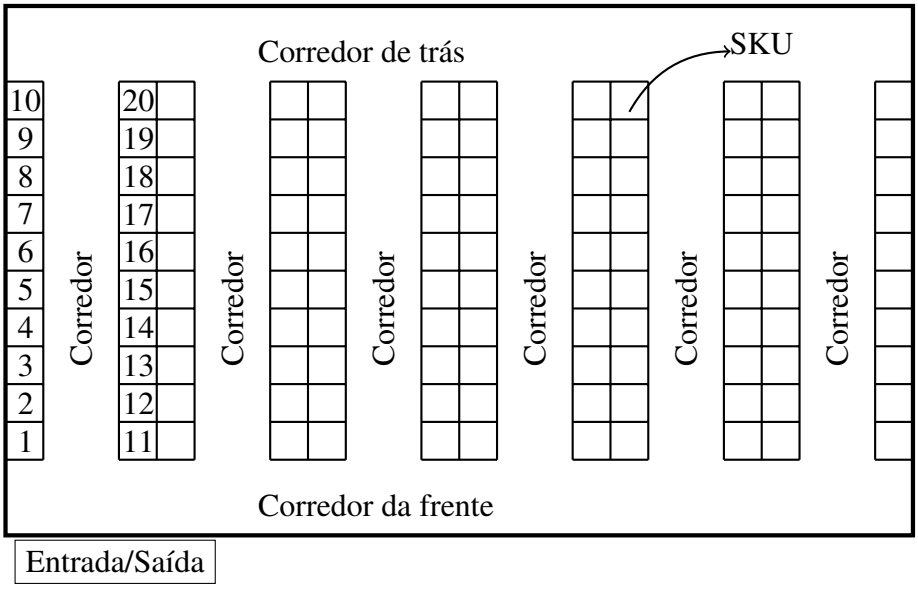


Figura 1 – Ilustração de um centro de distribuição.

Dado um pedido de compras de um cliente, a coleta dos produtos armazenados no centro de distribuição é apontado como o processo que demanda mais esforço e tempo dentre as atividades contidas no processo de armazenamento (TOMPKINS et al., 2003). O problema

de agrupar os pedidos de compra em lotes de maneira a minimizar a distância percorrida pelos coletores ao coletar os respectivos produtos tem sido estudado pela literatura há décadas, sob o nome *Order Batching Problem* (OBP). O agrupamento dos pedidos é um meio para minimizar a distância percorrida pelo coletor para recolher os produtos dos pedidos de compra.

Este problema possui um nível alto de dificuldade para sua resolução de forma geral, e esse fato é comprovado pela publicação de Gademann e Velde (2005), que inclui a prova de que o OBP é um problema que pertence à classe $\mathcal{NP}$-difícil, o que significa que não se conhece algoritmo eficiente para sua solução em tempo determinístico polinomial.

Este trabalho propõe a aplicação da metaheurística Busca Local Iterada (ou *Iterated Local Search*, ILS) para o tratamento do agrupamento de pedidos de compra em lotes no contexto do OBP. A etapa de roteamento para coleta dos lotes na versão do OBP tratada neste trabalho é realizada por uma política fixa, assim como ocorre na literatura.

1.1 Justificativa

Desde o fim da década de 70 e início da década de 80, Gross (1981) defende a ideia de que otimizar a coleta de pedidos é um ponto chave para reduzir custos operacionais de centros de distribuição. Com efeito, Gross (1981) e Hwang, Baek e Lee (1988) estimaram que cerca de 30 a 40% da mão de obra envolvida nos processos de centros de distribuição poderia estar alocada na operação de separação e coleta de pedidos. Entre o final da década de 80 e início da década de 90, Drury (1988) e Coyle et al. (1996) apontaram que o processo de separação e coleta de produtos podia representar até 65% dos custos de operação de um centro de distribuição. Já nos anos 2000, Frazelle e Frazelle (2002) compartilham o ponto de vista de Tompkins et al. (2003), apontando que a tarefa de coleta de pedidos podia absorver até 50% do total dos custos operacionais. Mais recentemente, 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, e que os resultados são benéficos para a sociedade.

De acordo com Tompkins et al. (2003), o tempo de viagem para coletar os produtos consome cerca de 50% de todas as operações necessárias 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 ordens implica na redução proporcional no tempo gasto para coletar esse grupo de ordens (JARVIS; MCDOWELL, 1991).

Adicionalmente, como mencionado, o OBP é um problema que pertence a classe $\mathcal{NP}$-difícil (GADEMANN; VELDE, 2005), representando atualmente um desafio também relevante para a ciência da computação.

1.2 Objetivos

É objetivo geral desta dissertação propor uma abordagem consistente e viável computacionalmente para abordagem do *Order Batching Problem*, que forneça resultados competitivos quando comparado aos resultados existentes na literatura. Compõem os objetivos específicos:

- Realizar pesquisa com o intuito de gerar embasamento teórico e revisão da literatura a respeito do OBP;
- Propor e implementar um método heurístico para abordagem ao OBP;
- Realizar experimentos computacionais utilizando instâncias propostas na literatura; e
- Comparar os resultados obtidos com os resultados publicados de maior relevância na literatura.

1.3 Organização do Texto

O restante deste trabalho está organizado como descrito a seguir. O Capítulo 2 apresenta uma revisão detalhada dos métodos de solução para o OBP, destacando as principais contribuições. Em seguida, no Capítulo 3, o problema abordado neste trabalho é descrito formalmente, bem como o método escolhido para sua abordagem. O desenvolvimento da ILS aplicada à solução do OBP é apresentada em detalhes no Capítulo 4. O Capítulo 5 contempla os experimentos computacionais preliminares realizados e resultados obtidos. O Capítulo 6 indica as atividades futuras para conclusão do trabalho de pesquisa, e finalmente, o Capítulo 7 apresenta a conclusão desta primeira etapa do trabalho de pesquisa.

2 Revisão Bibliográfica

Há décadas o OBP vêm sendo estudado pela comunidade científica, sendo perceptível o avanço obtido neste período. Alguns dos principais trabalhos encontrados sobre o OBP são sumarizados neste capítulo. Em particular, os trabalhos que representam o estado da arte relacionado ao OBP mais recentemente são detalhados.

Estudos iniciais sobre o OBP foram realizados na década de 80 por diversos autores como Elsayed (1981), Elsayed e Stern (1983), Hwang, Baek e Lee (1988), Goetschalckx e Ratliff (1988), Elsayed e Unal (1989). Todos estes trabalhos são caracterizados pela proposta de algoritmos para solução do OBP. Entretanto, os cenários analisados e os resultados obtidos são hoje considerados obsoletos, considerando a definição atual do problema e os cenários reais considerados. Por outro lado, algumas das estratégias empregadas por estes trabalhos são utilizadas até os dias atuais, conforme verificado em trabalhos recentes descritos a seguir.

Por exemplo, Elsayed (1981) propôs quatro algoritmos cuja estratégia se baseia em selecionar pedidos *semente* e a partir dele formar um lote. Esta estratégia consiste em determinar um primeiro pedido e posteriormente mensurar o ganho ao agrupar outros pedidos a ele. A cada instante, o pedido que gerar o maior ganho é adicionado à solução parcial. A métrica utilizada para mensurar o ganho é a similaridade entre as composições do pedido semente e dos demais pedidos. Outro exemplo é devido a Elsayed e Unal (1989), que propuseram a utilização de heurísticas advindas do problema de roteamento de veículos, como heurística de economias de Clarke & Wright (CLARKE; WRIGHT, 1964). Este algoritmo foi utilizado como abordagem para o OBP seguindo o critério de maior economia na distância percorrida ao agrupar pedidos em um mesmo lote.

Outro aspecto relevante e de uso em tempos atuais são as políticas de roteamento do coletor dentro do centro de distribuição de formato retangular. Goetschalckx e Ratliff (1988) aplicaram a *política transversal*, também mencionada na literatura como *S-shape*, e a *política de retorno*. Ambas determinam como o coletor deve se deslocar ao coletar os pedidos de compra e influenciam no custo total da operação. Com base nestas descobertas, novas estratégias de roteamento foram empregadas no OBP na década seguinte, no intuito de criar uma abordagem de roteamento mais eficiente.

Na década de 80 não havia uma descrição clara de como os centros de distribuição estruturavam e organizavam seu estoque de produtos. Na década de 90, pesquisas como as de Gibson e Sharp (1992), Hall (1993), Pan e Liu (1995) e Rosenwein (1996) começaram a considerar novas variáveis, como a organização dos centros de distribuição com corredores e prateleiras. Essa nova perspectiva reflete com maior qualidade a realidade dos centros de distribuição mantidos pelas corporações (GIBSON; SHARP, 1992). Nota-se também o uso de

métricas para determinar o custo para coletar um lote de pedidos considerando a organização dos produtos no estoque e também a política de roteamento.

Dando continuidade aos estudos sobre políticas de roteamento iniciados na década anterior, Hall (1993) descreve e avalia de maneira abrangente as políticas *S-shape*, *mid-point return* e *largest gap return*. No referido trabalho, concluiu-se que as diferentes políticas de fato podem gerar soluções próximas do ótimo em diferentes situações.

Gibson e Sharp (1992) protagonizaram a primeira aplicação da heurística *First-Come*, *First-Served* (FCFS) ao OBP. O FCFS é uma heurística gulosa que considera a ordem de chegada dos pedidos. Na abordagem original, os n primeiros pedidos (ou um número próximo de n) são agrupados em um lote e coletados juntos, independente da localização dos produtos. O valor de n é limitado pela capacidade do coletor. De fácil compreensão e aplicação, tornou-se comum sua aplicação em cenários reais pelo baixo custo da curva de aprendizado pelos coletores.

Pan e Liu (1995) realizaram um estudo comparativo de diferentes critérios para seleção de pedidos semente e para seleção de pedidos a serem agrupados em um mesmo lote. Foram propostos quatro critérios para cada uma das duas decisões, totalizado 16 versões de algoritmos gulosos. Concluiu-se que os melhores critérios são aqueles que levam em consideração o polígono convexo formado pela localização dos produtos no centro de distribuição, de forma que cada produto é um vértice e há uma aresta entre dois produtos coletados em sequência. O pedido de maior área convexa é selecionado como semente, e os pedidos que geram a maior interseção com esta área convexa são agrupados a ele no mesmo lote.

Rosenwein (1996) propôs dois métodos para abordagem do OBP. O primeiro, nomeado como *Centre Of Gravity* (COG) emprega métricas de distância como parâmetro de decisão para o agrupamento dos pedidos. O segundo, nomeado *Minimum Additional Aisle* (MAA), utiliza uma heurística gulosa para realizar o agrupamento dos pedidos. No intuito de mensurar a qualidade dos resultados, foram realizados experimentos computacionais que compararam os métodos propostos aos métodos de Gibson e Sharp (1992), que representava o estado da arte então. O MAA obteve melhores resultados e culminou em uma melhora aproximada de 28% dos resultados obtidos anteriormente.

No final da década de 90, Koster, Poort e Wolters (1999) realizaram uma análise abrangente que permitiu inferirem que os algoritmos de sementes e algoritmos de economias, quando comparados com outras heurísticas, apresentam melhor desempenho para gerarem soluções iniciais. Esta conclusão confirmou as expectativas de alguns autores da década de 80 como exemplo cita-se Elsayed (1981), Elsayed e Unal (1989).

A partir do início do século XXI, nota-se um número maior de trabalhos dedicados a estudar o OBP em comparação às décadas anteriores. Este mesmo período coincide com a popularização da *internet* e um avanço significativo na capacidade dos computadores e resolvedores, em termos de desempenho. Com estes recursos, surge uma maior gama de abordagens para o

OBP, como métodos exatos, heurísticos, metaheurísticos e métodos baseados em mineração de dados. Por exemplo, citam-se os métodos propostos por Chen e Wu (2005), Chen et al. (2005), Gademann e Velde (2005), Hsu, Chen e Chen (2005), Hwang e Kim (2005), Ho e Tseng (2006), Bozer e Kile (2008), Albareda-Sambola et al. (2009).

Neste período, destaca-se o uso de abordagens distintas e diversificadas como propostas de solução para o OBP. Chen e Wu (2005) e Chen et al. (2005) foram pioneiros no uso de técnicas de mineração de dados como abordagem para o agrupamento de pedidos no OBP. Os métodos propostos consistem em criar associações entre os pedidos de compra dos clientes para posteriormente agrupá-los, observando a capacidade máxima do coletor. Apesar disto, até o momento, o uso de técnicas de mineração de dados não são empregadas em muitas pesquisas.

Dentre as várias contribuições do trabalho Gademann e Velde (2005), destaca-se a prova de que o OBP pertence à classe $\mathcal{NP}$-difícil em seu caso geral, em que o número de pedidos de compra por lote é maior do que dois. Outra contribuição importante, em decorrência da prova da complexidade do problema, foi a formalização da definição atual do OBP. Além disso, foi proposto um algoritmo *Branch-and-Price* que utiliza geração de colunas e relaxação Lagrangeana como abordagem para resolver o OBP, juntamente com a aplicação do método *Iterated descent* para obtenção de solução inicial.

É notório neste período o emprego de buscas locais juntamente com variadas heurísticas e metaheurísticas. Albareda-Sambola et al. (2009) investigam o comportamento do método *Variable Neighborhood Search* (VNS) aplicado ao OBP. Foram definidas vizinhanças com base em seis movimentos: (*i*) a transferência de um pedido do lote L_i para outro lote L_j; (*ii*) a transferência de dois pedidos do lote L_i para o lote L_j; (*iii*) a transferência de dois pedidos do lote L_i para os lotes L_j e L_k; (*iv*) a troca de pedidos entre dois lotes L_i e L_j; (*v*) a transferência de dois pedidos dos lotes L_j e L_k para o lote L_i; e (*vi*) uma cadeia de transferências de pedidos entre os lotes L_i, L_j e L_k. Adicionalmente, três políticas de roteamento foram consideradas pelo VNS: *S-shape*, *largest gap* e também uma combinação entre essas duas estratégias, denominada *composite routing strategies*.

Henn et al. (2010) introduzem duas abordagens para o OBP: (*i*) *Iterated Local Search* (ILS); e (*ii*) *Rank-Based Ant System* (RBAS). O ILS é um metaheurística baseada em busca local que consiste em duas fases: (*i*) busca local; e (*ii*) perturbação. Na primeira fase, a busca local parte de uma solução inicial obtida utilizando a heurística FCFS e considera como vizinhança as soluções factíveis (lotes viáveis) obtidas de forma aleatória. A segunda fase consiste em modificar a solução atual e aplicar novamente busca local nesta solução. O RBAS é uma metaheurística evolutiva que combina otimização por colônia de formigas e heurísticas gulosas de economia (CLARKE; WRIGHT, 1964). Inicialmente, cada pedido de compra compõe um único lote. Posteriormente, os lotes são combinados entre si, formando lotes maiores que não excedam a capacidade do coletor. Para cada possível combinação calcula-se a economia possível e também a quantidade de feromônio associada à solução.

Na segunda década do século XXI as pesquisas acerca do OBP se intensificaram. Nota-se o surgimento de pesquisas com objetivos voltados a aplicação prática em organizações. Além disso, nota-se que vários autores promovem com maior empenho a reprodutibilidade dos métodos propostos e a comparação futura com os resultados obtidos, compartilhando bases de instâncias utilizadas nos testes computacionais desenvolvidos em seus trabalhos.

Destaca-se Henn e Wäscher (2012) como principal referência de *benchmark*. Neste trabalho foram geradas mais de duas mil instâncias artificiais, sendo este um dos primeiros trabalhos a disponibilizar a base de instâncias utilizadas em sua pesquisa de forma simplificada e acessível. Neste mesmo trabalho foram propostos os métodos *Attribute-Based Hill Climber* (ABHC) e uma aplicação da Busca Tabu (ou *Tabu Search*, TS). Além disso, foram propostas duas versão da metaheurística ILS aplicadas ao OBP por Henn et al. (2010): (*i*) ILS-1, que consiste no uso do ILS com uma solução inicial fornecida pela heurística FCFS; e (*ii*) ILS-2, que consiste no uso do ILS com uma solução inicial obtida pelo método *Clarke & Wright* (CLARKE; WRIGHT, 1964). O método *Clarke & Wright* (CLARKE; WRIGHT, 1964) e o algoritmo FCFS foram também utilizados para gerar soluções iniciais tanto para o método ABHC quanto para o método TS. O método ABHC, baseado em busca local, obteve melhores resultados quando comparado aos demais métodos, ILS-1, ILS-2, TS.

Öncan (2015) propôs o *Iterated Local Search Algorithm with Tabu* (ILST), que melhorou a qualidade dos resultados disponíveis até então e se tornou o estado da arte até aquele momento. Além disso, foram propostos três modelos de programação inteira mista: um considerando a política de roteamento *S-shape*, um considerando a política *largest gap* e outro considerando a política de ponto médio. Nos experimentos computacionais realizados, o ILST se destacou pela robustez de seu desempenho. Trata-se de uma variação do método *Local Search Algorithm with Tabu Thresholding* (LST). A estrutura básica do LST consiste em aplicar métodos de busca local e também movimentos aleatórios de inserção e troca de pedidos de compra entre lotes até que o limite de iterações seja atingido. O *Tabu Thresholding* é utilizado para guiar a oscilação entre busca local e movimentos aleatórios. O ILST consiste em uma implementação da metaheurística ILS que utiliza o LST como busca local e diferentes métodos de perturbação. Adicionalmente, o ILST permite a geração de soluções inviáveis, diferentemente do LST. Foi demonstrado por meio dos experimentos computacionais que o ILST superou os resultados anteriores da literatura até então. Nos experimentos, foram comparados resultados de diversos trabalhos, dentre eles, os do método ILS-2 proposto por Henn e Wäscher (2012) que era considerado o então estado da arte.

Posteriormente, Muter e Öncan (2015) propuseram um modelo de programação inteira mista baseado em particionamento de conjuntos com geração de colunas para resolver o OBP. Entretanto, após os experimentos computacionais realizados, concluiu-se que para instâncias com lotes acima de 32 pedidos de compra a abordagem não era praticável.

Alguns trabalhos recentes relatam a abordagem do OBP em casos reais com requisitos

específicos, porém, sem a comparação com métodos tradicionais da literatura. Wang, Wang e Mi (2017) investigam uma abordagem para o OBP em um centro de distribuição de um *e-commerce* da China. Os resultados obtidos pela abordagem proposta foram comparadas com as políticas já utilizadas anteriormente na organização, obtendo de cerca 13,45% de melhoria nos resultados em relação à política *First-Come First-Served*, e em torno de 77,88% de melhoria em relação à política *single-order picking*, que consiste em coletar um pedido por incursão no centro de distribuição. Zhao e Yang (2017) relatam a abordagem do OBP para o centro de distribuição da empresa B2C. Para averiguar o impacto que a abordagem proposta no cenário da empresa alvo, foram realizados alguns experimentos confrontando as políticas propostas pelo autor e as adotadas pela organização. Segundo os autores, o resultado apurado demonstra uma melhoria de 59% em relação à política adotada pela empresa que, a FCFS.

Žulj, Kramer e Schneider (2018) proupuseram a aplicação do método *Adaptive Large Neighborhood Search and tabu search* (ALNS/TS) ao OBP. Atualmente, a esta abordagem são atribuídos os melhores resultados para as instâncias encontradas na literatura, portanto, este é considerado o estado da arte. Dada a importância deste trabalho, este é descrito em maior profundidade.

O ALNS (ROPKE; PISINGER, 2006) é um método metaheurístico derivado do *Large Neighborhood Search* (LNS) proposto por Shaw (1998). Dada uma solução inicial viável, o LNS realiza operações de remoção e inserção de elementos da solução para gerar novas soluções. Entretanto, o ALNS realiza estas operações de maneira independente entre si. Para tanto, a metaheurística possui um arcabouço de operadores de inserção e remoção, com estratégias heurísticas variadas e eventualmente divergentes. A combinação desses variados operadores resulta em diversas estratégias de busca local, induzindo uma quantidade maior de vizinhanças diferentes, o que viabiliza maior exploração do espaço de busca. Desta forma, o ALNS se mostra um método robusto. A escolha dos operadores de inserção e remoção em um determinado momento é realizada de forma adaptativa, considerando dados estatísticos anteriores colhidos durante o processo de execução do ALNS. Estes dados refletem o desempenho de cada operador de remoção e inserção. Inicialmente, todos os métodos possuem chances iguais de serem utilizados, entretanto, a cada iteração do ALNS as chances dos operadores são atualizadas com o objetivo de penalizá-los ou gratificá-los de acordo com seu desempenho.

Na implementação do ALNS proposta por Žulj, Kramer e Schneider (2018), o TS é utilizado como mecanismo adicional de busca local, independente dos operadores de inserção e remoção. A solução inicial é gerada pelo método *Clarke & Wright* (CLARKE; WRIGHT, 1964) e seis operadores de remoção e inserção são empregados no ALNS. Após serem aplicados os operadores de inserção e remoção, a solução obtida é submetida a um critério de aceitação adaptativo, baseado no clássico critério do *Simulated Annealing*, porém, com o uso de ruído para aumentar a diversidade da busca.

A remoção é realizada por três operadores: (i) remoção aleatória, que remove q (valor

definido aleatoriamente como um percentual de n) pedidos aleatórios da solução atual; (ii) remoção de pedidos que mais contribuem para a função de avaliação; e (iii) remoção *Shaw*, que define a similaridade entre um pedidos de compra específico e os demais pedidos presentes na solução. A inserção é realizada por outros três operadores: (i) inserção gulosa, cujo é o aumento mínimo do custo para a solução; (ii) 1-arrependimento; e (iii) 2-arrependimento. Ambos (ii) e (iii) são especializações do método k-arrependimento, cujo principio é uma melhoria da inserção gulosa, antecipando o efeito de uma futura inserção. O valor do k-arrependimento descreve a diferença de custo da inserção de um determinado pedido nas k melhores posições da solução, de modo que elementos com maior arrependimento devem ser inseridos primeiro na solução.

A cada iteração do ALNS é aplicado o TS (GLOVER, 1986) com o objetivo explorar mais ainda as vizinhanças da solução atual. O TS utilizado considera a vizinhança *Nshift* and *Nswap* (HENN; WÄSCHER, 2012) e seleciona o melhor vizinho de ambas vizinhanças simultaneamente, mesmo se este possuir qualidade pior do que a solução original. Uma lista tabu é utilizada para inibir a seleção de vizinhos já utilizados. O critério de aspiração utilizado é a determinação de uma nova melhor solução global.

Os experimentos computacionais demonstraram que o ALNS/TS superou os melhores resultados encontrados anteriormente na literatura, tornando-se o estado da arte atual em relação ao OBP. Especificamente, os resultados obtidos pelos métodos ILS-1, ILS-2 e ABHC de Henn (2012) e o ILST de Öncan (2015) foram superados. Foram consideradas as 2560 instâncias de Henn (2012), e adicionalmente foram propostas 800 novas instâncias.

3 Fundamentação Teórica

Este capítulo apresenta de maneira incremental a contextualização e a definição formal do *Order Batching Problem*. Também são apresentados conceitos relacionados, como as estratégias de roteamento que podem ser utilizadas nas abordagens ao OBP. Por fim, é apresentado o conceito da metaheurística ILS, utilizada nesta etapa deste trabalho.

3.1 O *Order Batching Problem*

Na versão abordada neste trabalho, o *Order Batching Problem* considera um centro de distribuição com um número finito de corredores verticais paralelos cuja distância é denotada por w e dois corredores horizontais que delimitam o comprimento dos corredores verticais, denotado por h. Cada corredor vertical possui largura desprezível e é delimitado por prateleiras laterais, divididas em SKUs que são enumeradas em cada corredor. O acesso a cada SKU, para operações de armazenamento ou coleta, é realizado apenas pelo corredor vertical imediatamente à sua frente.

A Figura 2 ilustra o leiaute de um centro de distribuição com seis corredores verticais com dez SKUs do lado esquerdo e dez SKUs do lado direito em cada corredor, que são enumeradas de 1 a 20, dois corredores horizontais e um acesso de entrada e saída do centro de distribuição. Como exemplo, os SKUs do corredor a estão enumerados, em um mesmo padrão que se repete nos demais corredores. O leiaute está organizado de maneira tal que o acesso ao SKU ϕ em destaque é realizado unicamente pelo corredor e. Além disso, caso o coletor esteja em um corredor vertical, este pode acessar simultaneamente os SKUs da prateleira à direita quanto os SKUs da prateleira à esquerda, sem nenhum deslocamento adicional.

Seja $O = \{O_1, O_2, ..., O_n\}$ o conjunto de pedidos de compra, sendo que $O_i = \{p_1, p_2, \ldots, p_m\}$, $i = 1 \ldots n$, é um conjunto não vazio de produtos que representa um pedido de compra i, cujos produtos estão armazenados em um centro de distribuição e as respectivas localizações são conhecidas. A Tabela 1 apresenta um exemplo de composição de pedidos de compra e respectiva localização dos produtos no centro de distribuição. São 4 pedidos de compra (enumerados de O_1 a O_4) e 17 tipos de produtos distintos (enumerados de p_1 a p_{17}), que totalizam 54 produtos pedidos. Cada produto que compõe um pedido de compra é identificado pelo corredor e pelo SKU em que está localizado, considerando-se que cada SKU armazena um único tipo de produto.

A Figura 3 exemplifica a disposição dos produtos que compõem os pedidos de compras do exemplo ilustrado pela Tabela 1, considerando um coletor com capacidade dada por $c = 30$, $h = 10$ e $w = 5$. Assim como Žulj, Kramer e Schneider (2018) e Henn e Wäscher (2012), segue-se a convenção de utilização de unidades de comprimento (ou *Length Units*, LU):

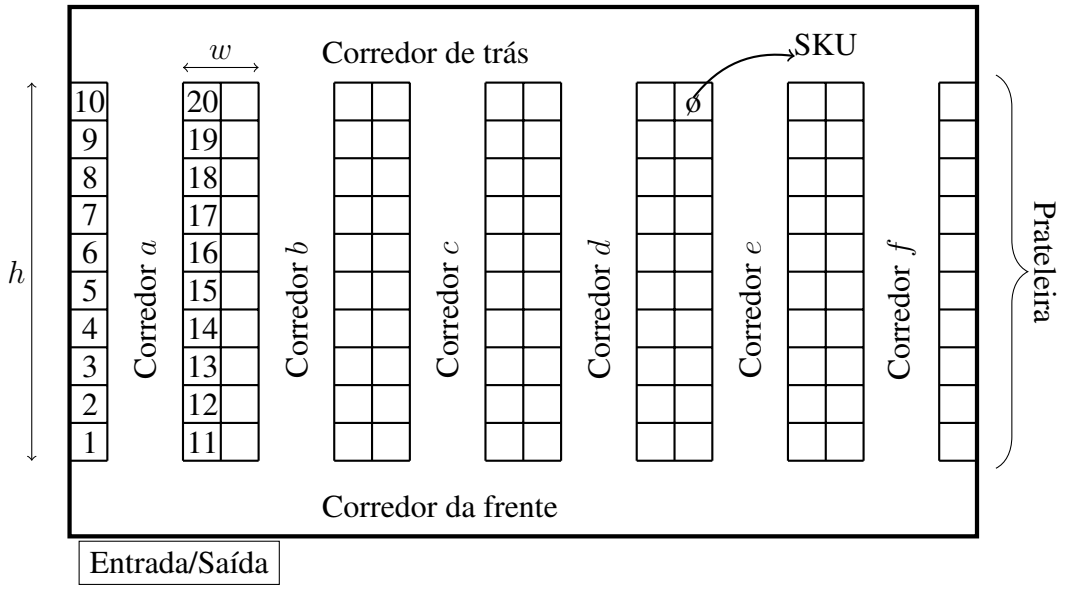


Figura 2 – Exemplo de leiaute de um centro de distribuição.

Tabela 1 – Exemplo de composição de pedidos de compra.

Pedido	Quantidade	Produto	Corredor	SKU
O_1	3	p_1	b	19
	2	p_2	c	8
	1	p_3	f	8
	1	p_4	f	15
O_2	4	p_5	d	7
	1	p_6	f	9
	1	p_7	f	17
O_3	1	p_8	b	4
	5	p_9	c	5
	3	p_{10}	d	3
	4	p_{11}	e	9
	3	p_{12}	f	1
O_4	7	p_{13}	a	1
	3	p_{14}	a	3
	4	p_{15}	a	11
	4	p_{16}	a	17
	7	p_{17}	a	19

- Cada SKU possui o comprimento de uma LU;
- A distância entre dois corredores verticais contíguos é de cinco LU;
- A distância entre o ponto de acesso do centro de distribuição (entrada/saída) e o primeiro SKU do corredor mais À esquerda (i.e., SKU 1 do corredor a) é de $\frac{3}{2}$ LU;
- A distância entre o ponto de acesso do centro de distribuição (entrada/saída) e o corredor horizontal da frente é de $\frac{1}{2}$ LU;

- A distância entre a extremidade de um corredor vertical e corredor horizontal mais próximo é de um LU; e

- A distância entre os dois corredores horizontais é de 12 LU.

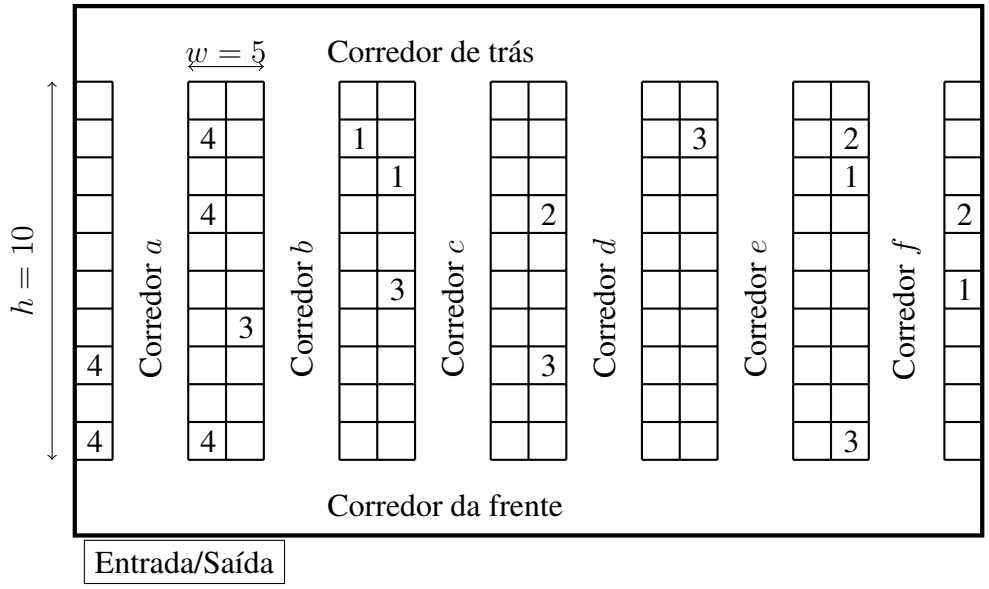


Figura 3 – Localização dos produtos dos pedidos de compra da Tabela 1.

Neste trabalho, considera-se que os coletores são humanos auxiliados por equipamentos para transporte de produtos que possuem capacidade máxima denotada por c. Para a minimização do custo com a coleta, tendo em vista a limitação de capacidade do coletor, deseja-se realizar o agrupamento dos pedidos de compra em *lotes*, tal que cada lote é coletado em uma única incursão no centro de distribuição. Dado o conjunto de n pedidos de compra O conforme definido anteriormente, deseja-se então particioná-lo em lotes B_i, $i \in \{1, \ldots, k\}$, tal que $B_i \subseteq O$, $B_i \cap B_j = \emptyset$ $(i \neq j)$ e $\bigcup_{i=1}^{k} B_i = O$.

O agrupamento dos pedidos de compra em lotes deve respeitar duas outras regras: (i) a cardinalidade de cada lote deve ser menor ou igual a capacidade do coletor; e (ii) os pedidos não podem ser fracionados, i.e., deve-se incluir cada pedido integralmente em somente um lote. Por fim, considera-se que a capacidade do coletor é delimitada pela quantidade de produtos, i.e., cada lote $B_i \in O$ deve conter no máximo c produtos, considerando que cada coletor coleta um lote de produtos por vez.

Considerando o cenário ilustrado pela Figura 3, a Figura 4 apresenta um exemplo de particionamento do conjunto O em lotes, denominados B_1 e B_2. A composição do primeiro lote é dada por $B_1 = \{O_1, O_2, O_3\}$ e a composição do segundo lote é dada por $B_2 = \{O_4\}$. A cardinalidade de cada lote é dada pela soma do número de produtos que cada pedido possui. Desta forma, $|B_1| = 29$ e $|B_2| = 25$, ambos abaixo da capacidade do coletor.

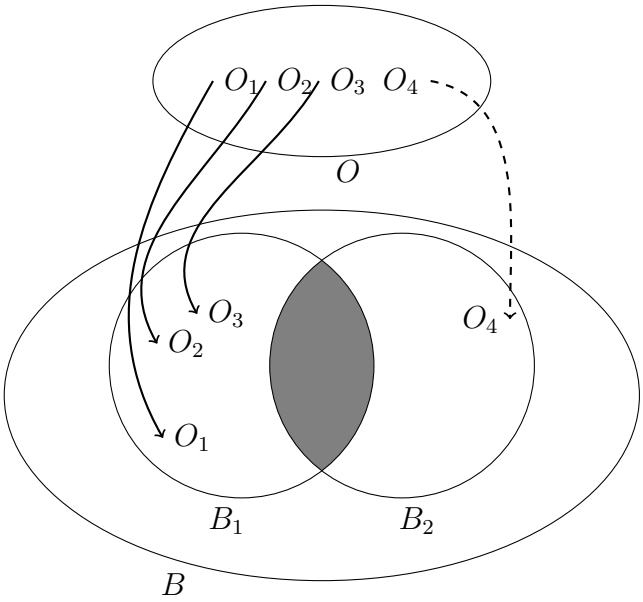


Figura 4 – Exemplo de agrupamento de pedidos em lotes.

O custo de uma solução para o OBP é dada pelo somatório das distâncias percorridas pelo coletor nas rotas realizadas dentro do centro de distribuição ao coletar cada produto de cada lote, desde a entrada até a saída. O custo varia, portanto, de acordo com a estratégia adotada para o *roteamento* de coleta dos itens, definida previamente. A definição da estratégia de roteamento é independente do OBP. A Seção 3.1.1 apresenta duas das estratégias mais amplamente utilizadas na literatura.

Em suma, o OBP consiste em, dados (i) o conjunto de pedidos de compra O, cujos produtos estão armazenados em um centro de distribuição com as respectivas localizações conhecidas; (ii) um coletor de capacidade fixa; e (iii) uma estratégia de roteamento, determinar o agrupamento viável dos pedidos de compra que gere o menor comprimento total de rota para sua coleta. De acordo com Gademann e Velde (2005), este é um problema pertencente à classe $\mathcal{NP}$-Difícil.

3.1.1 Estratégias de roteamento

A estratégia de roteamento no OBP é uma política determinada previamente à determinação dos lotes para posterior coleta. Cada estratégia traz consigo um comportamento imutável, cuja performance pode variar de estratégia para estratégia. Cada rota é iniciada no acesso de entrada/saída do centro de distribuição e termina quando todos os produtos do lote designado a esta incursão forem coletados e o coletor se retirar do centro de distribuição pelo mesmo acesso por onde entrou.

Na literatura são reportadas seis estratégias de roteamento para o OBP, vide Petersen (1997). Entretanto, neste trabalho serão abordadas as estratégias mais populares e também consideradas nos trabalhos que historicamente representam o estado da arte sobre o OBP: (i)

largest gap (HALL, 1993); e (ii) *S-shape* (GOETSCHALCKX; RATLIFF, 1988; HALL, 1993). Segundo Žulj, Kramer e Schneider (2018) estas estratégias minimizam o congestionamento nos corredores, considerando que diferentes coletas são realizadas em paralelo. Além disso, Koster, Poort e Wolters (1999) enfatizam que a execução destas estratégias de percurso é de simples entendimento e tem a adoção pelos funcionários facilitada, tendo em vista a baixa curva de aprendizado.

3.1.1.1 Estratégia *largest gap*

A estratégia de percurso *largest gap* tem como princípio percorrer integralmente somente o primeiro e o último corredores que possuírem produtos que devem ser coletados. Os demais corredores verticais que possuem produtos a serem coletados são acessados de forma parcial, em uma ou nas duas extremidades, conforme a localização dos produtos. Desta forma, a distância não percorrida pelo coletor nestes corredores é máxima. A Figura 5 ilustra um exemplo de rota para a coleta de um lote, obedecendo a política *largest gap*.

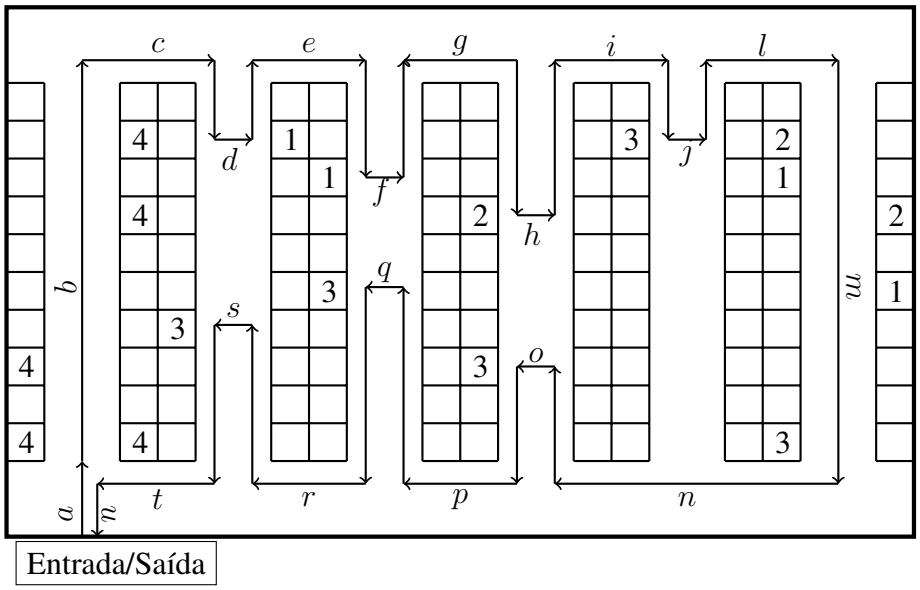


Figura 5 – Exemplo de percurso *largest gap*.

Neste exemplo, a coleta é realizada em 20 passos, enumerados de a à u. A incursão inicia-se no passo (a) e posteriormente o primeiro corredor é percorrido integralmente (b). Para as próximas coletas (d, f, h e j), cada corredor vertical deve ser percorrido o mínimo necessário, portanto, decide-se por acessá-los a partir do corredor horizontal de trás. Na sequência, percorre-se todo o último corredor (m) e novamente cada corredor vertical é percorrido o mínimo necessário nas próximas coletas (o, q e s), sendo acessados pelo corredor horizontal da frente. Por fim, a rota termina com o passos t e u.

Para medir a distância d_t percorrida em uma rota t, conforme a exemplificada, utiliza-se a Equação (3.1). Na referida equação, c_e indica a distância percorrida para entrar no centro de

distribuição, c_s indica a distância percorrida para sair do centro de distribuição, a constante w indica a distância entre dois corredores verticais contíguos, percorridos k vezes, e n_{SKU} representa a quantidade de SKUs percorridos verticalmente.

$$d_t = k \times w + c_e + c_s + n_{SKU} \tag{3.1}$$

A Figura 6 ilustra a rota *largest gap* identificada (a)-(r), para coleta do lote B_1 ilustrado na Figura 4. A rota se inicia pelo ingresso no centro de distribuição pelo ponto de acesso entrada/saída (a) e segue pelo corredor horizontal da frente (b). Na sequência, percorre-se todo o segundo corredor vertical (c), coletando os produtos dos pedidos 1 e 3. A partir de então, a rota segue pelo corredor horizontal de trás o suficiente para obter acesso aos corredores verticais (d, f, h e j) que abrigam produtos que devem ser coletados, (e, g e i). De acordo com a estratégia adotada, os corredores verticais são percorridos o mínimo possível para a coleta dos produtos. O último corredor vertical (l) é também percorrido completamente, a fim de coletar os produtos dos pedidos 1, 2 e 3, além de acessar o corredor horizontal da frente. De maneira análoga à anterior, os corredores verticais são novamente percorridos o mínimo necessário para coleta dos produtos (n e p) e posteriormente a rota se encerra no mesmo acesso em que começou (r). Esta rota possui comprimento de 116 LU, conforme a Equação (3.1).

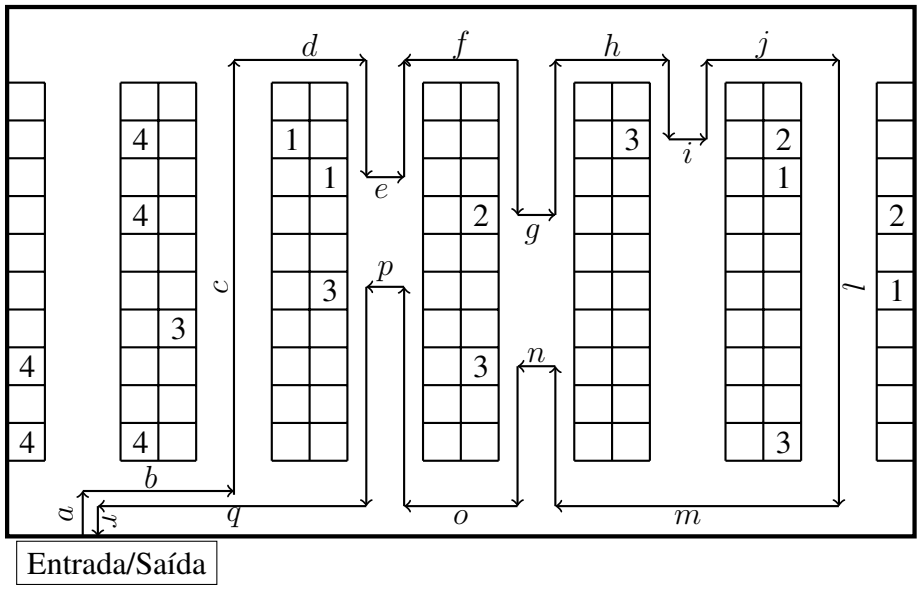


Figura 6 – Rota *largest gap* para coleta do lote B_1.

A Figura 7 ilustra uma rota para a coleta do lote B_2, cujos passos (a)-(e) seguem a politica *largest gap*. A rota é iniciada no acesso de entrada/saída do centro de distribuição (a), e segue pelo corredor mais à esquerda, coletando os produtos do pedido 4. Os SKUs de cada lado do corredor são acessados ao passo em que o coletor se move (b). Ao final da coleta, é realizado o retorno (c) e a rota segue pelo mesmo corredor em direção ao ponto de acesso entrada/saída

do centro de distribuição (d), no qual é finalizada (e). Esta rota tem comprimento de 21 LU. Desta forma, a solução gerada pela coleta dos lotes B_1 e B_2 utilizando-se a estratégia *largest gap* possui custo total 137 LU.

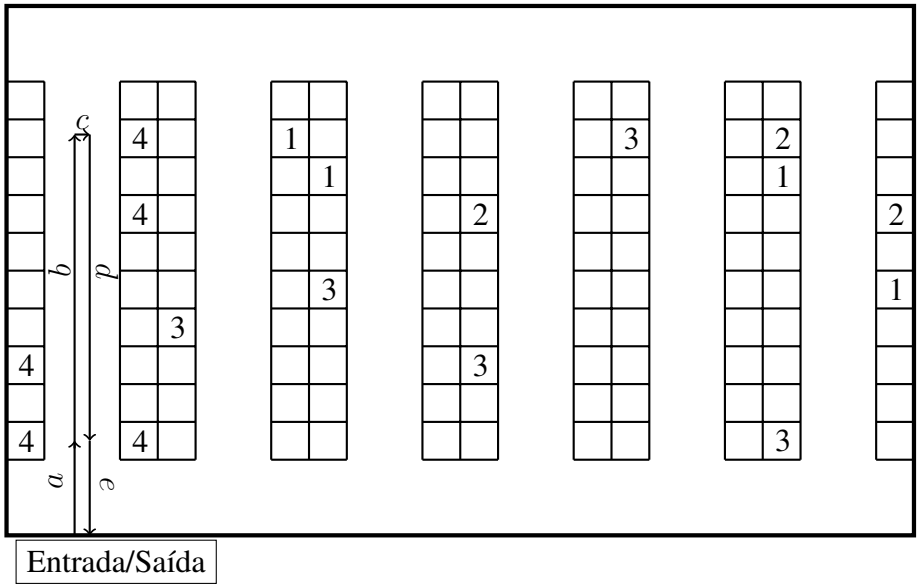


Figura 7 – Rota *largest gap* para coleta do lote B_2.

3.1.1.2 Estratégia *S-shape*

A estratégia *S-shape* (HALL, 1993) tem como princípio percorrer toda a extensão dos corredores em que o coletor entrar. Corredores em que não há coleta a ser realizada não são percorridos. Uma exceção quanto ao percurso completo é realizado em relação ao último corredor vertical da rota: caso este seja acessado pelo corredor horizontal da frente, é possível que este seja percorrido parcialmente e o retorno seja feito pelo próprio corredor vertical.

A Figura 8 ilustra uma rota identificada (a)-(o) que utiliza a estratégia *S-shape* para percorre todo o centro de distribuição. A rota é iniciada pelo acesso de entrada/saída (a), segue pelo corredor mais à esquerda (b) e o percorre até o final. Em seguida, a rota converge à direita (c) e ingressa no próximo corredor (d). O mesmo padrão de percurso é repetido entre os passos (e) e (m), até o retorno ao acesso de entrada/saída (n)-(o).

Para o cálculo da distância total d_t percorrida em uma rota t elaborada pela estratégia *s-shape*, utiliza-se a Equação (3.2). Nesta equação, c_e representa a distância percorrida para entrar no centro de distribuição, c_s representa a distância percorrida para sair do centro de distribuição, i indica o número de corredores verticais que foram percorridos integralmente, j representa o número de transposições de corredores realizadas e u representa o número de SKUs percorridos no corredor que foi explorado parcialmente. Caso não haja corredor percorrido de forma parcial, assume-se $u = 0$.

$$d_t = \sum_{x=1}^{i} h + \sum_{y=1}^{j} w + c_e + c_s + 2u \tag{3.2}$$

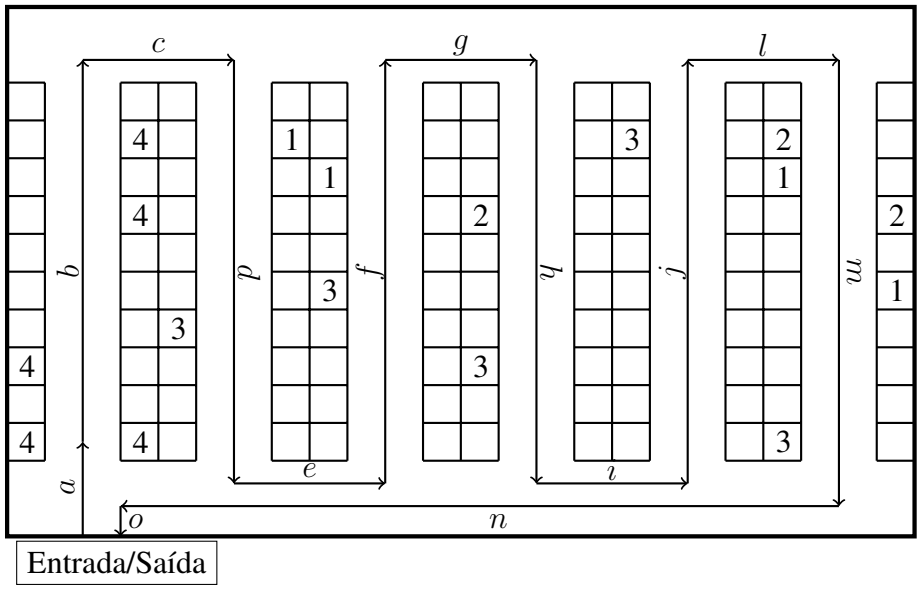


Figura 8 – Exemplo de percurso *S-shape*.

A Figura 9 ilustra uma rota *S-shape* para coleta do lote B_1. A rota identificada (a)-(p) é iniciada no acesso de entrada/saída do centro de distribuição (a). Como não há produto deste lote a ser coletado no primeiro corredor vertical, procede-se ao segundo (b). A partir de então, os corredores são percorridos completamente à medida em que os produtos são coletados (c, e, g e i). Os SKUs de cada lado de um mesmo corredor são acessados simultaneamente, sem percorrer alguma distância adicional. O último corredor não é percorrido completamente (l), dado que não há mais produtos a serem coletados. A rota retorna pelo mesmo corredor (m e n), no intuito de obter acesso ao corredor horizontal da frente e seguir por este até o ponto de acesso entrada/saída (o), finalizando a rota (p). A distância percorrida para coleta deste lote é de 119 LUs.

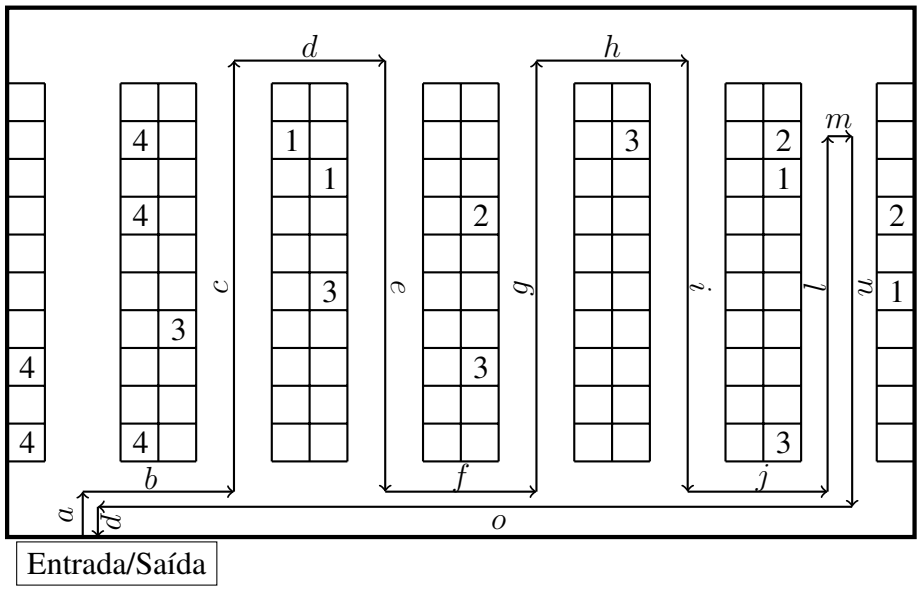


Figura 9 – Rota *S-shape* para coleta do lote B_1.

A coleta do lote B_2 pode ser feita de forma trivial utilizando a estratégia *S-shape*. A rota gerada por esta estratégia é exatamente a mesma gerada pela estratégia *largest gap*, conforme ilustrado na Figura 7. Para a coleta deste lote são percorridos 21 LU, exatamente a mesma distância exigida pela rota *largest gap*. Desta forma, a solução gerada pela coleta dos lotes B_1 e B_2 utilizando-se a estratégia *S-shape* possui custo total 140 LU.

3.2 Busca Local Iterada

A Busca Local Iterada (ou *Iterated Local Search*, ILS) proposta originalmente por Stützle (1998) e novamente por Lourenço, Martin e Stützle (2003), é uma metaheurística que fornece uma maneira de sistematicamente aplicar métodos de busca local. Esta metaheurística, assim como outras, possui dois pilares como base de sua operação:

1. Intensificação da busca, proporcionada pela busca local realizada por um ou mais métodos, que exploram as vizinhanças de uma dada solução em busca de soluções ótimas locais; e

2. Diversificação da busca, característica proveniente das pequenas alterações realizadas nas soluções ótimas locais, na tentativa de escapar das mesmas e possibilitar a exploração de novas regiões do espaço de busca.

Segundo Lourenço, Martin e Stützle (2003), a ILS é uma meta-heurística que gera bons resultados em vários problemas combinatórios. A compreensão do método ILS demanda pouco esforço, e além disso, possui implementação simplificada. A performance deste algoritmo tende a ser consideravelmente boa em relação à outros métodos que envolvem busca local, e está intrinsecamente relacionada a como os suas componentes são implementadas. De acordo com Stützle (1998) há três formas básicas de se implementar intensificação e diversificação no ILS, sendo primordiais para o seu bom desempenho: busca local, perturbação e critério de aceitação de solução.

O componente busca local atua em uma solução intermediária s, explorando suas soluções vizinhas por meio de operações denominadas movimentos e evoluindo-a para uma solução ótima local s^*. Entretanto, somente o uso de métodos de busca local faria com que o método convergisse rapidamente e ficasse preso em ótimos locais (eventualmente não globais) com facilidade. A componente de perturbação possibilita explorar de regiões promissoras do espaço de busca e também escapar de ótimos locais por meio da alteração de elementos de uma dada solução, mesmo que isto implique em piora da avaliação da mesma. Realça-se que o método de perturbação deve encontrar um equilíbrio, tal que seja forte o suficiente para permitir a exploração adequada do espaço de busca, mas não tão forte a ponto de criar um reinício aleatório.

O critério de aceitação de novas soluções geradas é uma tomada de decisão que pode considerar as soluções anteriores e também o histórico da busca local (STÜTZLE, 1998). Deve-se

calibrar a sensibilidade do critério de aceitação, pois soluções ruins quando perturbadas podem levar a uma região não explorada do espaço de busca onde eventualmente reside um ótimo local de melhor qualidade que a melhor solução atual. Por outro lado, a aceitação de muitas soluções ruins pode reduzir drasticamente o desempenho do método, tornando-o impraticável quando aplicado em problemas complexos. O critério de aceitação, portanto, pode agir ao mesmo tempo como componente de intensificação e diversificação.

Segundo Lourenço, Martin e Stützle (2003), a ILS tem a habilidade de fugir dos ótimos locais, mas também contempla uma característica que prejudica a sua performance. A ILS é capaz de gerar uma solução s_2 a partir de uma solução inicial s_1, mas não necessariamente produz a solução s_1 a partir da solução s_2. Consequentemente, parte do espaço de busca pode não ser alcançado pela ILS. No entanto, este aspecto insatisfatório não impede que o método seja eficaz, uma vez que seja atingida uma boa sintonia entre seus componentes.

A Figura 10 ilustra parte de uma execução hipotética da ILS, na qual uma solução inicial s_0 é gerada para um problema de minimização. Em seguida, s_0 é submetida a um procedimento de busca local, o qual gera a melhor solução atual s^*, um ótimo local. Na sequência, aplica-se um procedimento de perturbação em s^* derivando uma nova solução s', com pior valor de função objetivo, porém, localizada em uma região diferente do espaço de busca. Novamente, a solução atual é submetida ao procedimento de busca local, na qual obtém-se o ótimo local (s^*), sendo esta a melhor solução obtida. A metaheurística repete este comportamento até que se atinja um critério de parada.

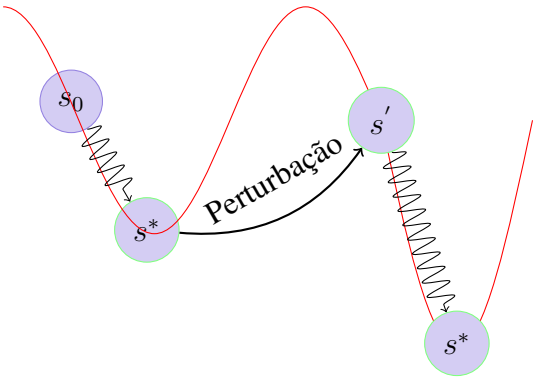


Figura 10 – Possível efeito da perturbação e da busca local na ILS.

A Figura 11 apresenta o fluxograma de execução da ILS padrão. O algoritmo inicia-se com a geração de uma solução inicial. Esta solução é submetida ao método de perturbação, gerando uma nova solução atual. Esta nova solução é então submetida ao método de busca local, gerando um ótimo local da vizinhança adotada. A solução obtida no passo anterior é avaliada pelo critério de aceitação de novas soluções. Caso a solução seja aceita, atualiza-se a solução do ILS. Caso contrário, a solução é descartada. Este processo é repetido até que um critério de parada seja atingido.

A ILS quando aplicada a um problema específico necessita que os componentes solução

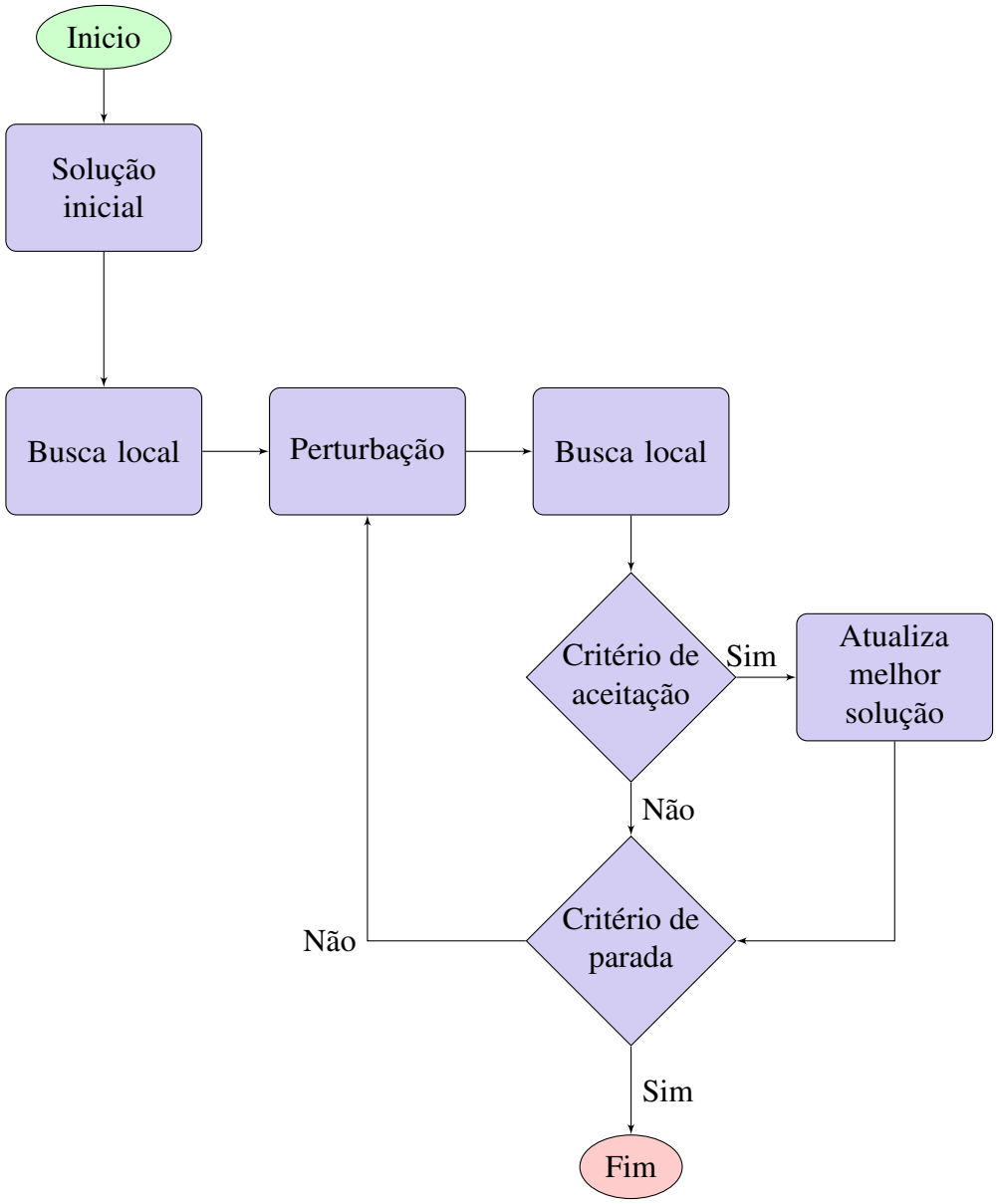


Figura 11 – Modelo de processo do *Iterated Local Search*.

inicial, perturbação, busca local e critério de aceitação sejam adaptados às necessidades do problema específico. A solução inicial, por exemplo, visa gerar um ponto de partida viável para o ILS, e pode ser gerada aleatoriamente ou por alguma heurística. A busca local é estruturada em vizinhanças que são induzidas por movimentos como por exemplo o movimentos de troca e de inserção, tradicionalmente encontrados na literatura. A perturbação é um procedimento que realiza um movimento diferente dos movimentos utilizados na busca local, no intuito de evitar ciclicidade. Por fim, o critério de aceitação, dentre as várias possibilidades, pode ser definido como um critério de descida rápida, no qual somente é aceita uma solução quando esta possuir melhor qualidade que a melhor solução atual ou um critério probabilístico que aceite um percentual de soluções de qualidade inferior a qualidade da melhor solução. O número λ de iterações que o ILS vai executar também deve ser personalizado de acordo com o problema

abordado, podendo ser definido previamente. As adaptações realizadas na ILS neste trabalho para abordar o OBP são descritas no Capítulo 4.

O Algoritmo 1 descreve o comportamento padrão da ILS, conforme proposto por Lourenço, Martin e Stützle (2003). Este algoritmo gera uma solução inicial (linha 2), aplica busca local (linha 3) e a torna a melhor solução encontrada. Em seguida, o laço principal é executado até que um critério de parada seja atingido (linhas 4 a 8). Aplica-se uma perturbação na solução atual (linha 5) e em seguida a solução alterada é submetida a uma busca local, obtendo-se uma nova solução (linha 6). A solução gerada é então submetida ao critério de aceitação, podendo ser aceita ou não (linha 7). Caso seja aceita, a melhor solução é atualizada (linha 7). Caso contrário, não há atualização. Por fim, verifica-se o critério de parada (linha 8). Caso tenha sido atingido, a ILS termina e a melhor solução é retornada (linha 9). Caso contrário, uma nova iteração do laço de repetição é executada.

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

Saída: Solução s^*

1 **início**
2 $s_0 \leftarrow$ geraSoluçãoInicial();
3 $s^* \leftarrow$ buscaLocal(s_0);
4 **repita**
5 $s' \leftarrow$ perturbação(s^*, histórico);
6 $s'' \leftarrow$ buscaLocal(s');
7 $s^* \leftarrow$ criterioAceitação(s^*, s'', histórico);
8 **até** *critério de parada atingido*;
9 **retorna** s^*;
10 **fim**

4 Desenvolvimento

Neste capítulo é detalhada a implementação da metaheurística ILS utilizada como abordagem para o problema OBP. A ILS é um método genérico que pode ser aplicada à vários problemas diferentes, mas para tal, necessita-se de adaptações em suas componentes para que haja um alinhamento estratégico dos componentes que possibilite o funcionamento harmônico e eficaz da metaheurística. Os componentes de geração da solução inicial, busca local, perturbação e critério de aceitação são descritos detalhadamente nas seções seguintes, assim como outros aspectos relevantes da metaheurística.

4.1 Solução inicial

O método para geração de soluções iniciais adotado é a heurística conhecida como Clarke & Wright, ou C&W (CLARKE; WRIGHT, 1964), proposta originalmente para problemas de roteamento de veículos. Aplicada ao OBP, esta heurística considera pedidos de compra como os clientes em um problema de roteamento e utiliza um critério guloso para avaliar a economia gerada a cada pedido inserido em um mesmo lote, que representam as rotas no problema original. A solução inicial é gerada em função da distância necessária para efetuar a coleta de um determinado lote de pedidos. Dada uma lista de pedidos O, a solução inicial considera cada pedido como um lote unitário, realiza combinações destes lotes e seleciona a combinação que oferecer a maior economia. O processo de combinação de lotes é repetido até que nenhuma combinação resulte em uma distância menor para a coleta dos lotes.

O Algoritmo 2, que possui complexidade assintótica $O(n^3)$, demonstra como é executada a heurística C&W aplicada ao OBP. Inicialmente, cria-se um conjunto B de lotes unitários a partir do conjunto de pedidos (linha 2). Em seguida, calcula-se a distância total necessária para efetuar a coleta de todos os lotes do conjunto B (linha 3). A maior economia possível é inicializada com o menor valor possível (linha 4) e após isto, são realizadas as combinações dos lotes tomados dois a dois, mantendo sempre aquela que gerar a maior economia, de acordo com o laço de repetição das linhas $5 - 20$. Um lote B_i é retirado do conjunto B (linha 6) e é combinado com todos os demais lotes (linhas $7 - 19$). O lote B_j é recuperado do conjunto B e soma-se a distância necessária para coletar os lotes B_i e B_j separadamente (linhas 8 e 9). Posteriormente, calcula-se a distância para coletar o lote resultante da união entre B_i e B_j (linha 11). Em seguida é verificado se a união dos lotes gera economia (linha 12). Em caso positivo, calcula-se a distância poupada (linha 13) e verifica-se se esta é melhor combinação até então (linha 14). Em caso positivo, atualiza-se a maior economia e quais são os lotes envolvidos na combinação (linhas $15 - 17$). Após a execução do laço, se houve alguma combinação que gera economia (linha 22), então a combinação de lotes é realizada (linhas 23-25). Por fim, verifica-se

a existência de melhora na iteração atual (linha 27). Caso positivo, o algoritmo é chamado novamente recursivamente (linha 28). Caso contrário o conjunto de lotes B é retornado (linha 30), finalizando o algoritmo.

Algoritmo 2: Método C&W (CLARKE; WRIGHT, 1964).

```
   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      repita
6          B_i ← proximoLote(B);
7          repita
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         até o último elemento do conjunto B;
21     até o último elemento do conjunto B;
22     se C_maiorEconomia ≠ −∞ então
23         excluirLote(l_1, B);
24         excluirLote(l_2, B);
25         incluirLote(Lote_uniao, B);
26     fim
27     se d_t<calculaDistancia(B) então
28         retorna C&W(B);
29     senão
30         retorna B;
31     fim
32 fim
```

A Figura 12 (a)-(f) ilustra um exemplo de geração de solução inicial para uma instância com sete pedidos de compra enumerados de O_1 a O_7, cuja composição é descrita na Tabela 2. Neste exemplo convencionou-se o uso da estratégia de roteamento *s-shape* e a capacidade dos lotes fixada em 30 produtos, considerando um centro de distribuição com o mesmo leiaute apresentado na Figura 2. O número entre parênteses à frente de cada lote denota a distância percorrida para coleta dos mesmos.

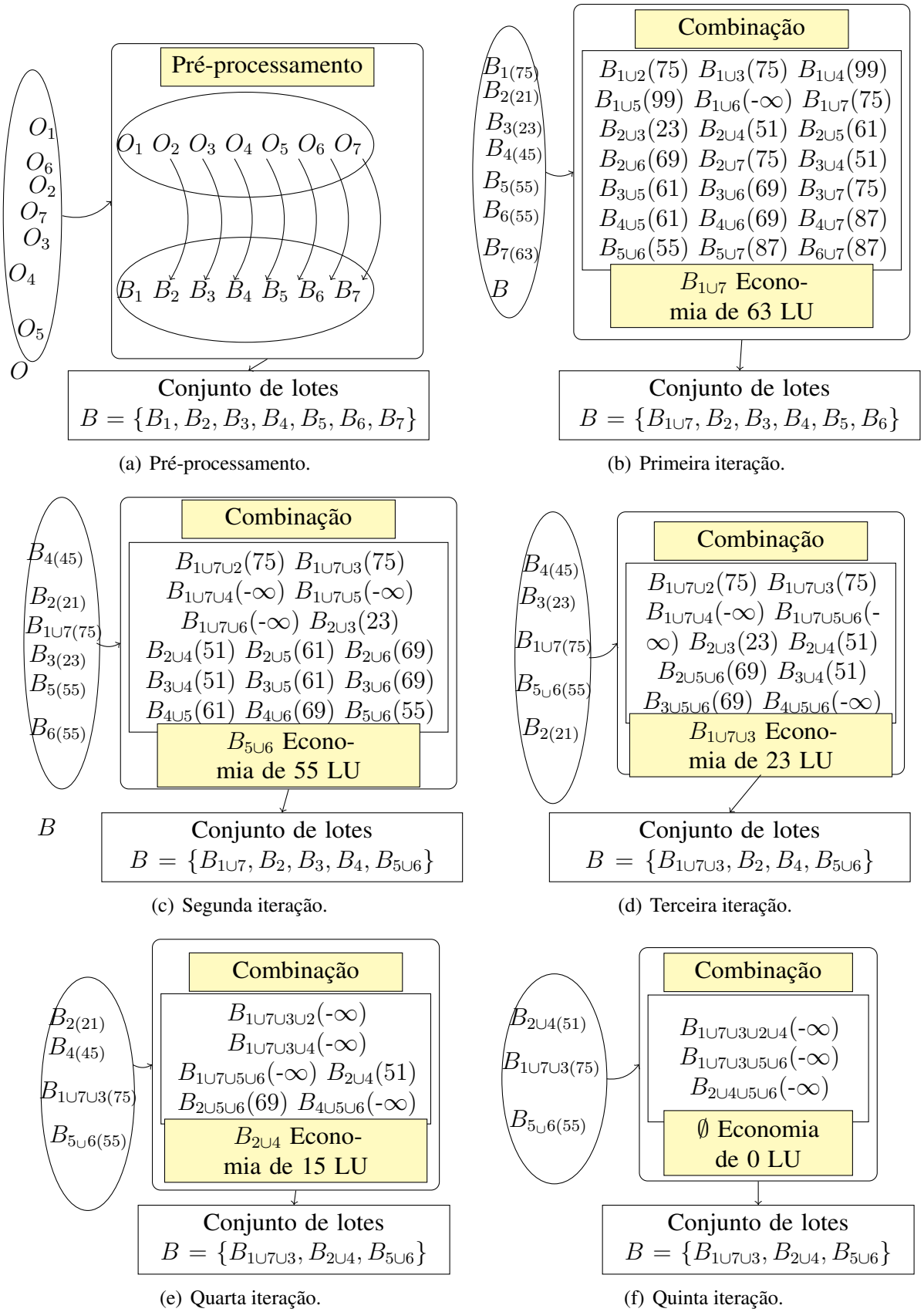
Tabela 2 – Composição dos pedidos de compra utilizados para geração de solução inicial na Figura 12.

Pedido	Quantidade	Produto	Corredor	SKU
O_1	20	p_1	a	1
	3	p_2	f	8
O_2	1	p_3	a	9
O_3	3	p_4	a	10
O_4	3	p_5	b	8
	2	p_6	c	2
O_5	3	p_7	c	5
	1	p_8	d	2
O_6	20	p_9	c	9
	2	p_{10}	d	6
O_7	4	p_{11}	f	5

Inicialmente na Figura 12 (a), tem-se sete lotes unitários, i.e., $B_1 = \{O_1\}$, $B_2 = \{O_2\}$, $B_3 = \{O_3\}$, $B_4 = \{O_4\}$, $B_5 = \{O_5\}$, $B_6 = \{O_6\}$, e $B_7 = \{O_7\}$. Na primeira iteração (b), o algoritmo faz uma combinação de lotes tomados dois a dois e verifica que a combinação dos lotes B_1 e B_7 gera a maior economia de distância, poupando 63 LU. Desta forma, a união destes lotes unitários forma um novo lote $B_{1\cup7}$, ao passo que os lotes B_1 e B_7 são excluídos. A solução corrente é o conjunto B=$\{B_{1\cup7}, B_2, B_3, B_4, B_5, B_6\}$. O processo de combinação de lotes é realizado novamente (c) e a maior economia é dada pela união dos lotes B_5 e B_6 com economia de 55 LU, dando origem ao lote $B_{5\cup6}$ e à extinção dos lotes B_5 e B_6. Tem-se então a solução B=$\{B_{1\cup7}, B_2, B_3, B_4, B_{5\cup6}\}$. Na iteração seguinte ($d$), identifica-se os lotes $B_{1\cup7}$ e B_3 como a combinação mais promissora, promovendo uma economia de 23 LU, logo a solução se torna B=$\{B_{1\cup7\cup3}, B_2, B_4, B_{5\cup6}\}$. Na quarta iteração ($e$) os lotes B_2 e B_4 são combinados, gerando uma economia de 15 LU e a solução B=$\{B_{1\cup7\cup3}, B_{2\cup4}, B_{5\cup6}\}$. Por fim, na última iteração verifica-se que não há nenhuma combinação viável que resulte em economia, logo, o algoritmo termina e a solução final é dada pelos lotes B=$\{B_{1\cup7\cup3}, B_{2\cup4}, B_{5\cup6}\}$, com custo 181 LU.

4.2 Perturbação

A capacidade do ILS diversificar as soluções alcançadas é dependente da componente perturbação. Neste trabalho, utiliza-se o método proposto por Henn et al. (2010). Dado um conjunto B de lotes viáveis, escolhem-se aleatoriamente dois lotes B_i e B_j, que sofrerão operações de inserção e remoção, respectivamente. Um número q é escolhido aleatoriamente do intervalo $(1, 2, \ldots, |B_i|)$, e atua como um limitante superior para o número de operações realizadas. Em outras palavras, serão realizadas q operações de remoção de pedidos no lote B_i e de inserção destes pedidos no lote B_j, desde que a capacidade do lote B_j não seja ultrapassada.

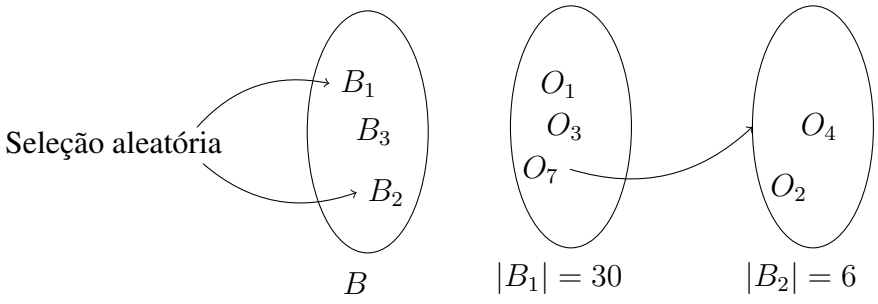


(a) Pré-processamento.
(b) Primeira iteração.
(c) Segunda iteração.
(d) Terceira iteração.
(e) Quarta iteração.
(f) Quinta iteração.

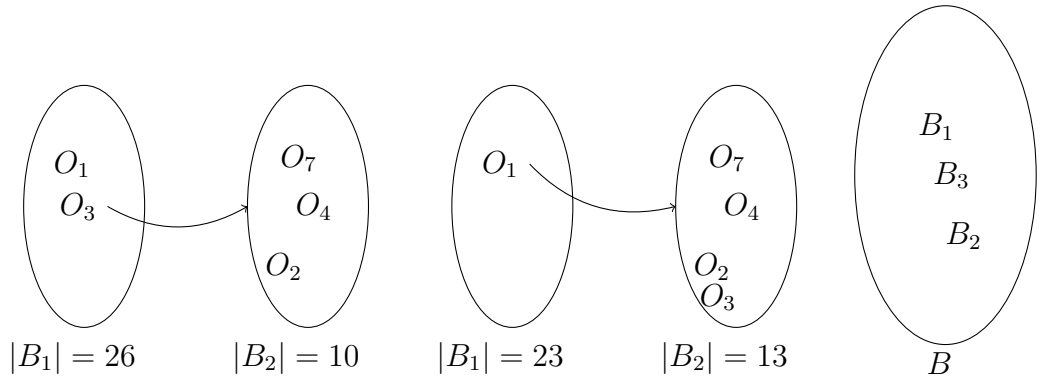
Figura 12 – Exemplo de solução inicial gerada pelo método C&W (CLARKE; WRIGHT, 1964)

Os q pedidos a serem removidos de B_i são selecionados aleatoriamente e incluídos em B_j, sem a realização de nenhuma análise quanto a melhora ou piora do custo da solução, dado que o objetivo é a geração de uma solução diferente da anterior.

A Figura 13 ilustra o comportamento da componente de perturbação, considerando que a capacidade dos lotes é novamente fixada em 30 produtos, e conjunto $B = \{B_1, B_2, B_3\}$ é o mesmo apresentado como solução inicial na Seção 4.1. O método de perturbação recebe um conjunto de três lotes enumerados de B_1 a B_3. Em (a), dois lotes (B_1 e B_2) são selecionados aleatoriamente, bem como o número $q = 3$ de operações a serem realizadas. Em seguida, três operações de remoção em B_1 e inserção em B_2 serão avaliadas, cujos elementos serão selecionados aleatoriamente. Em (b), o pedido O_7 é removido de B_1 e inserido em B_2. Não havendo inviabilidade, a operação é mantida. Em (c), o pedido O_3 é removido de B_1 e inserido em B_2, também mantendo a viabilidade da solução. Por fim, em (d), o pedido O_1 é removido de B_1 e inserido em B_2. A operação implica em inviabilidade, portanto, é desfeita. O lote B_1 se tornou um lote unitário, contendo somente o pedido O_1. A solução após a perturbação é apresentada em (e).



(a) seleção aleatória dos lotes B_1 e B_2. (b) Primeira iteração.



(c) Segunda iteração. (d) Terceira iteração. (e) Quarta iteração

Figura 13 – Exemplo de perturbação de uma solução pelo método de Henn et al. (2010).

O Algoritmo 3, cuja complexidade assintótica é $O(n)$, apresenta o componente de perturbação. Inicialmente, selecionam-se aleatoriamente os lotes B_i e B_j e um número q (linhas $2-4$). Em seguida, cria-se um lote vazio temporário para abrigar os pedidos que não puderem ser inseridos no lote B_j por motivo de inviabilidade (linha 5). Em um laço de repetição (linhas

$6 - 12$) repetido q vezes, seleciona-se um pedido p de forma aleatória do lote B_i (linha 7), exclui-se este pedido do mesmo lote (linha 8) e posteriormente é realizada a tentativa de incluir o pedido no lote B_j (linha 9). Se pedido p exceder a capacidade do lote de destino, então este pedido é inserido no conjunto de pedidos bloqueados (linha 10). Após as operações de inserção e remoção, quaisquer pedidos não inseridos em B_j são inseridos novamente em B_i (linha 13). Caso não hajam pedidos não inseridos e B_i seja vazio, remove-se o lote B_i da solução (linhas $14 - 16$). Por fim, o conjunto atualizado de lotes é retornado (linha 17).

Algoritmo 3: Pseudocódigo da perturbação, pelo método de Henn et al. (2010).

```
   Entrada: Conjunto de lotes B
   Saída: Conjunto de lotes B
1  início
2      B_i ←selecaoAleatoria(B);
3      B_j ←selecaoAleatoria(B);
4      q ←selecaoAleatoria(1, |B_i|);
5      L_pedidosBloqueados ← ∅;
6      repita
7          p ←selecaoAleatoria(B_i);
8          excluiPedido(p, B_i);
9          se não incluiPedido(p, B_j) então
10             incluiPedido(p, L_pedidosBloqueados);
11         fim
12     até atingir q iterações ou atingir a capacidade do lote B_j;
13     união(B_i,L_pedidosBloqueados);
14     se estaVazio(B_i) então
15         excluiLote(B_i, B);
16     fim
17     retorna B;
18 fim
```

4.3 Busca Local

A busca local adotada neste trabalho considera duas vizinhanças aninhadas, baseadas nos movimentos de inserção e troca de pedidos de compra. Dois lotes são selecionados aleatoriamente (B_i e B_j). Inicialmente, são analisados movimentos de inserção de pedidos de B_i em B_j. Entretanto, a inserção pode violar a capacidade do lote B_j. Neste caso, são analisados movimentos de trocas de pedidos entre B_i e B_j. As seções a seguir detalham cada um dos dois movimentos e também fornecem uma visão geral da busca local implementada.

4.3.1 Movimento de Inserção

O movimento de inserção ocorre respeitando duas regras. A primeira é que o lote de destino deve possuir capacidade para abrigar o pedido selecionado; a segunda é que movimentos

que resultem em piora do valor da solução são descartados. Dados dois lotes B_i e B_j, cada um dos pedidos de B_i são removidos individualmente, em ordem aleatória, e inseridos em B_j, caso haja capacidade para tanto. Se houver piora no valor da solução o movimento é desfeito. Caso contrário, o movimento é mantido.

A Figura 14 (b) ilustra o processo de inserção de um pedido em um lote diferente. Dados dois lotes B_2 e B_1, seleciona-se aleatoriamente o pedido O_2 do lote B_2. Em seguida, ocorre a remoção deste pedido no lote B_2 e posteriormente é inserido no lote B_1. Por fim, avalia-se a qualidade e a viabilidade da solução gerada por este movimento. O valor da solução é o mesmo anterior e trata-se de uma solução viável, portanto, o movimento é mantido.

4.3.2 Movimento de Troca

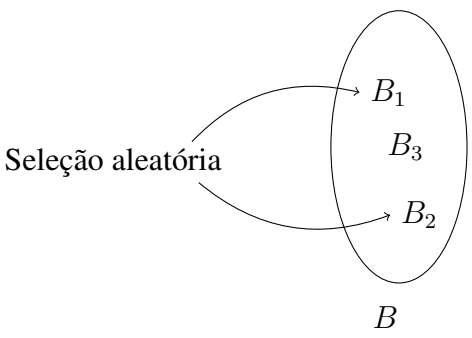
Conforme mencionado, o movimento de troca é realizado após uma tentativa frustrada de aplicação do movimento de inserção. Dados dois lotes B_i e B_j e um pedido de compra O_k, selecionados anteriormente pelo movimento de inserção, o movimento de troca de pedidos seleciona um segundo pedido O_l e avalia a troca entre O_k e O_l. Caso a troca resulte em piora do valor da solução, o movimento é descartado. Caso contrário, o movimento é mantido. Em ambos os casos, o pedido O_k é bloqueado para futuros movimentos de troca e inserção.

A Figura 14 (d) ilustra o processo de troca entre dois lotes B_2 e B_1. Selecionam-se dois pedidos aleatórios, O_3 do lote B_2 e O_1 do lote B_1 e, em seguida, efetua-se a troca dos pedidos. Por fim, este movimento é avaliado quanto à viabilidade. Neste exemplo, houve melhora no custo da solução e a capacidade de ambos os lotes foram respeitadas, logo, a troca é mantida.

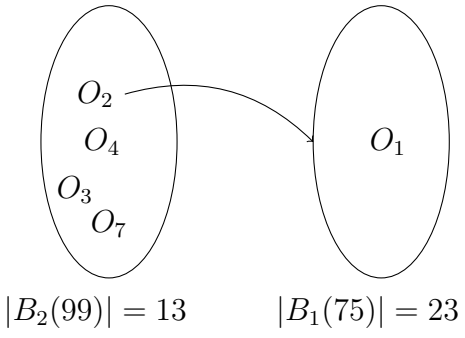
4.3.3 Visão Geral

A Figura 14 (a)-(e) ilustra um exemplo da busca local adotada neste trabalho, utilizando os dois movimentos descritos anteriormente. Novamente, convencionou-se o uso da estratégia de roteamento *s-shape* e a capacidade do coletor fixada em 30 produtos, considerando um centro de distribuição com o mesmo leiaute apresentado na Figura 2. O conjunto de lotes $B = \{B_1, B_2, B_3\}$ é o mesmo apresentado na Seção 4.2, cujos pedidos são descritos na Tabela 2.

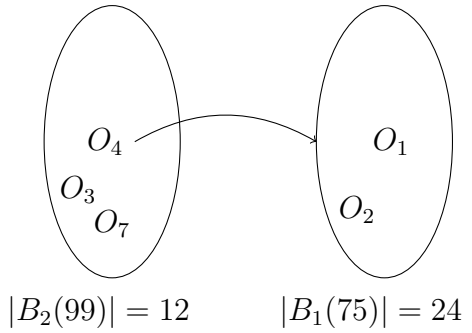
Na Figura 14 (a), dada uma solução $B = \{B_1, B_2, B_3\}$, os lotes B_2 e B_1 são selecionados de forma aleatória. Os números entre parênteses indicam a distância necessária para coletar cada lote e a cardinalidade de cada lote indica o seu número de produtos. A busca local analisa a viabilidade de inserção dos pedidos do lote B_2 no lote B_1. No passo (b) o pedido O_2 é selecionado aleatoriamente em B_2 e posteriormente é inserido no lote B_1. Como não houve piora no custo desta solução, esta inserção é mantida. Em (c), o mesmo procedimento é realizado para o pedido O_4 e resulta em uma nova solução viável. Em (d), o pedido O_3 é selecionado aleatoriamente em B_2, entretanto, sua inserção excede a capacidade do lote B_1. Inicia-se então o processo de troca



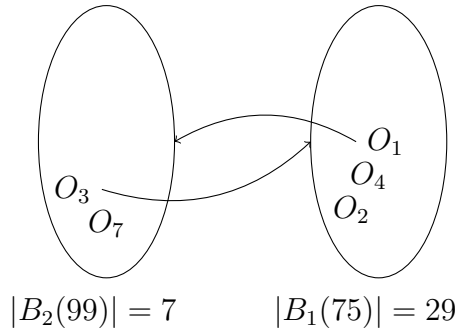
(a) Seleção aleatória dos lotes B_2 e B_1.



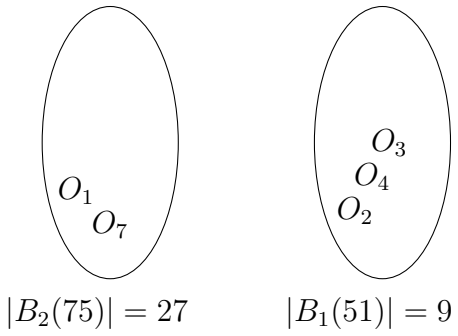
(b) Primeira iteração.



(c) Segunda iteração.



(d) Terceira iteração.



(e) Quarta iteração.

Figura 14 – Exemplo de aplicação da busca local.

e, para tanto, o pedido O_1 é selecionado aleatoriamente em B_1. A troca entre os pedidos O_3 e O_1 é realizada e resulta em uma nova solução viável. Observa-se uma melhora no custo da solução e por isso o movimento de troca é mantido. Por fim, na última iteração (e) seleciona-se o último pedido, O_7, de forma aleatória em B_2 e a sua inserção em B_1 é analisada. Todavia, constata-se uma piora no custo da solução e a inserção é desfeita. Após a execução da busca local, tem-se a solução com o mesmo número de lotes, porém, com um custo de 48 LU menor que a solução original.

O Algoritmo 4 ilustra o comportamento das buscas locais aninhadas. Inicialmente, seleciona-se aleatoriamente dois lotes B_i e B_j (linhas 2 e 3). Em seguida, calcula-se o custo da coleta de ambos os lotes (linha 4). O laço de repetição (linhas 5-27) é executado por $|B_i|$ iterações. Em cada iteração, seleciona-se um pedido p_1 aleatório de B_i (linha 6) e sua inserção em

B_j é avaliada quanto a viabilidade (linha 7). Caso o movimento resulte em uma solução viável, exclui-se o pedido p_1 de B_i (linha 8) e avalia-se o custo da nova solução (linha 9). Caso haja melhora (linha 10), atualiza-se a melhor solução atual (linha 11). Caso contrário, as operações são desfeitas (linhas 13 e 14). No caso de a inserção não ser um movimento viável, seleciona-se aleatoriamente um pedido p_2 de B_j (linha 17) e avalia-se a viabilidade da realização da troca entre os pedidos (linha 18). Se viável, avalia-se o custo da nova solução (linha 19). Caso haja melhora (linha 20), atualiza-se a melhor solução atual (linha 21). Caso contrário, as operações são desfeitas (linha 23). Por fim, é retornado o conjunto de lotes B (linha 28).

Algoritmo 4: Pseudocódigo da busca local.

```
   Entrada: Conjunto de lotes B
   Saída: Conjunto de lotes B aprimorado
 1 início
 2     B_i ←selecaoAleatoria(O);
 3     B_j ←selecaoAleatoria(O);
 4     d ←calculaDistancia(B_i)+calculaDistancia(B_j);
 5     repita
 6         p_1 ←selecaoAleatoria(B_i);
 7         se incluiPedido(p_1, B_j) = verdadeiro então
 8             excluiPedido(p_1, B_i);
 9             d' ←calculaDistancia(B_i)+calculaDistancia(B_j);
10             se d' ≤ d então
11                 d ← d';
12             senão
13                 excluiPedido(p_1, B_j);
14                 incluiPedido(p_1, B_i);
15             fim
16         senão
17             p_2 ←selecaoAleatoria(B_j);
18             se trocaPedido(B_i, p_1, B_j, p_2) = verdadeiro então
19                 d' ←calculaDistancia(B_i)+calculaDistancia(B_j);
20                 se d' ≤ d então
21                     d ← d';
22                 senão
23                     trocaPedido(B_i, p_2, B_j, p_1)
24                 fim
25             fim
26         fim
27     até |B_i| iterações;
28     retorna B;
29 fim
```

4.4 Critério de aceitação

A implementação de ILS proposta neste trabalho segue o princípio de descida rápida. Desta maneira, todas as soluções de custo menor ou igual que a atual são aceitas, ao passo as demais são recusadas. A idéia é explorar a estrutura da ILS que procura por ótimos locais nas diferentes vizinhanças exploradas. Entretanto, conforme indicado no Capítulo 6, diferentes critérios de aceitação serão avaliados na sequência deste trabalho.

4.5 Critério de parada

A ILS, como implementada, não identifica se uma dada solução é ótima, ou mesmo se está próxima a um valor ótimo. Para obter um compromisso entre convergência e tempo de execução, são adotados dois critérios de parada.

O primeiro critério de parada é acionado quando o algoritmo atinge o limite de 5.000 iterações. Este valor foi definido com base em experimentos preliminares e gerou uma boa convergência do ILS. Entretanto, em instâncias menores, esse limite pode gerar computação inútil, dado que o método continua a ser executado mesmo após sua convergência.

Sendo assim, o segundo critério de parada adotado é um gatilho acionado quando o algoritmo atinge o limite de 500 iterações sem nenhuma melhoria na solução atual. A cada iteração da ILS, caso haja melhora na solução, o número de iterações sem melhora recebe valor zero. Caso contrário, o número de iterações sem melhora é incrementado em uma unidade. Ao atingir-se o limite estipulado, a execução do método é suspensa.

5 Experimentos Computacionais

Este capítulo apresenta resultados e análises dos experimentos computacionais preliminares, comparando os resultados iniciais deste trabalho com os principais resultados publicados atualmente na literatura. Além disso, descreve-se as condições técnicas em que o método foi desenvolvido nesta pesquisa foi avaliado.

5.1 Ambiente computacional

Os experimentos computacionais foram realizados em um computador com sistema operacional Linux Ubuntu 15.10, arquitetura 64 *bits*, 8 GB de memória RAM e processador Intel i5-3330 quadcore com frequência 3.00 GHZ. A abordagem proposta neste trabalho foi implementada em c++ e compilada com o gcc 5.2.1, com a opção de otimização -O3.

5.2 Conjunto de instâncias

De acordo com os trabalhos recentes sobre o OBP, assume-se que em cada instância o centro de distribuição é estruturado em um bloco único, contendo dois corredores horizontais denominados corredor horizontal da frente e corredor horizontal de trás. Este centro de distribuição contém dez corredores verticais que ligam o corredor horizontal da frente ao corredor horizontal de trás. Cada um destes dez corredores contém 90 SKUs enumerados de 1 a 90, totalizando 900 SKUs. Em cada corredor, 45 SKUs estão localizados no lado direito dos corredores e as outros 45 SKUs estão dos lado esquerdo. Este leiaute é o mesmo apresentado na Seção 3 e também é utilizado nos trabalhos da literatura mais recentes, tais como Henn et al. (2010), Henn e Wäscher (2012), Žulj, Kramer e Schneider (2018).

O conjunto de instâncias adotado neste trabalho é o mesmo proposto por Henn e Wäscher (2012) e utilizado em vários trabalhos, dentre eles, Öncan (2015), Žulj, Kramer e Schneider (2018). Neste conjunto de instâncias dois cenários de demanda são ilustrados. No primeiro, considera-se que os produtos possuem demanda distribuída uniformemente, denominado UDD. No segundo cenário os produtos possuem demanda associada a uma classificação, denominado CBD:

1. Classe A: Alta frequência. Todos os produtos são localizados no corredor a. Nesta classe, 10% dos produtos correspondem a 52% da demanda total;

2. Classe B: Média frequência. Nesta classe, 30% dos produtos correspondem a 36% da demanda total. Todos os produtos são localizados nos corredores b, c, d; e

3. Classe C: Baixa frequência. Nesta classe, 60% dos produtos correspondem a 12% da demanda total. Estes produtos estão localizados em seis corredores, mais especificamente nos corredores de d à j.

Cada classe possui 40 instâncias. Em todas, a capacidade do coletor pode assumir os valores do conjunto $c = \{30, 45, 60, 75\}$ e a quantidade de pedidos de compra pode assumir os valores do conjunto $n = \{40, 60, 80, 100\}$. Em cada pedido de compra, o número de produtos é selecionado aleatoriamente no intervalo [5, 25]. Este conjunto totaliza 2.560 instâncias.

5.3 Resultados detalhados

A estratégia de roteamento *S-shape* foi utilizada nos experimentos computacionais, cujos resultados são apresentados resumidamente nas tabelas a seguir. Neste sumário de resultados, são apresentados valores médios relativos a dez execuções independentes do ILS. Em cada tabela, tem-se o número de pedidos que serão agrupados em lotes (Pedidos), capacidade do coletor (Capacidade). Os melhores resultados encontrados na literatura (ŽULJ; KRAMER; SCHNEIDER, 2018) (BKS), sendo que os valores sucedidos por * são soluções ótimas obtidas pelo Gurobi. O tempo médio em segundos (t), valor da solução média ($\Delta_{\bar{x}}$), o valor máximo das soluções (Δ_{max}), os valores das melhores soluções (Δ_{min}), e por fim, o desvio padrão (σ). A Tabela 3 apresenta o sumário dos resultados para as instâncias do tipo UDD.

Tabela 3 – Resultados para as instâncias UDD.

Pedidos	Capacidade	BKS	ILS				
			t	$\Delta_{\bar{x}}$	Δ_{max}	Δ_{min}	σ
n=40	c=30	10.462,00*	0,5	30.838,00	41.876,80	19.956,40	4.696,79
	c=45	6.864,00	0,5	15.271,30	19.046,00	12.065,90	1.977,51
	c=60	5.278,00	0,9	15.251,90	19.121,30	11.425,50	1.846,23
	c=75	4.273,00	0,5	15.093,80	18.503,50	9.770,90	1.877,81
n=60	c=30	15.482,00*	0,9	23.547,10	32.256,60	15.522,60	2.908,21
	c=45	10.032,00	1,1	22.480,50	26.542,10	15.720,60	2.449,37
	c=60	7.705,00	1,0	22.715,90	29.781,40	18.743,90	2.328,41
	c=75	6.294,00	1,1	22.871,50	30.373,40	17.932,50	2.868,12
n=80	c=30	20.645,00*	2,3	31.350,60	37.255,60	25.155,20	2.815,87
	c=45	13.328,00	2,2	30972.50	37.928,80	24.368,90	2.785,27
	c=60	10.173,00	2,3	30646.20	35,858,20	21.421,30	3.122,42
	c=75	8.233,00	1,3	30821.40	39.382,10	25.172,70	3.303,06
n=100	c=30	25.540,00*	2,3	39.353,40	46.272,80	32.525,80	3.048,06
	c=45	16.357,00	3,8	37.337,60	44.552,60	32.384,90	2.752,36
	c=60	12.472,00	4,2	37.636,90	43.523,40	31.234,50	2.929,82
	c=75	10.151,00	3,6	37.896,70	45.648,60	31.021,70	3.070,88

Observa-se que no subconjunto de 60 pedidos com capacidade do coletor fixado em 30 produtos o método proposto se aproximou da solução ótima obtendo um gap de 0,01% da solução ótima. Nos demais subconjuntos, o ILS proposto apresenta dificuldade em convergir para a solução ótima, além disso nota-se uma volatilidade alta na diferença das soluções com custo mínimo e máximo. Nota-se também que os resultados demonstram uma alta dispersão. A medida que o número de pedidos vai crescendo nas instâncias, nota-se uma baixa taxa de crescimento no tempo de execução (t), sendo que a taxa de crescimento do número de pedidos não impacta de forma relevante o tempo de execução. O desvio padrão (σ) sofre pouca variação nos valores do desvio padrão mesmo com o aumento do custo das soluções. Os resultados preliminares dos experimentos computacionais para o conjunto de instâncias CBD são apresentados de forma resumida na Tabela 4.

Tabela 4 – Resultados das instâncias CBD.

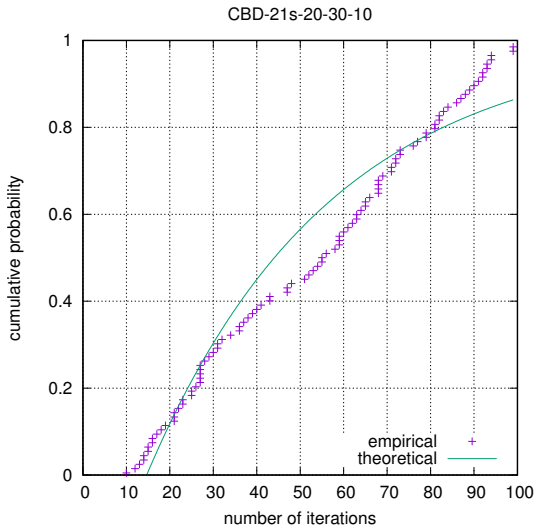
Pedidos	Capacidade	BKS	ILS				
			t	$\Delta_{\bar{x}}$	Δ_{max}	Δ_{min}	σ
n=20	c=30	4.192,00*	0,2	5.923,58	7.930,80	4.192,00	1.190,63
	c=45	2.689,00	0,2	5.567,58	7.519,50	3.266,00	1.149,54
	c=60	2.156,00	0,2	5.668,91	7.765,90	3.926,20	947,50
	c=75	1.743,00	0,2	5.376,22	7.004,90	3.434,60	861,86
n=40	c=30	7.907,00*	0,5	11.312,40	15.065,00	7.967,80	1.716,89
	c=45	5.185,00	0,5	11.490,80	14.146,70	8.386,00	1.454,46
	c=60	3.988,00	0,9	22.793,50	27.105,70	18.130,80	2.544,23
	c=75	3.243,00	0,5	11.069,30	14.323,20	7.039,60	1.442,06
n=50	c=30	10.098,00*	0,7	14.925,70	18.881,00	11.241,80	1.789,60
	c=45	6.462,00	0,7	14.218,20	17.553,20	11.160,50	1.681,22
	c=60	4.988,00	0,7	14.514,90	17.467,60	11.371,80	1.552,66
	c=75	4.077,00	0,7	14.264,70	17.276,30	11.476,40	1.396,52
n=60	c=30	11.609,00*	1,1	17.084,10	23.372,50	11.680,40	2.287,19
	c=45	7.566,00	1,1	16.773,40	19.859,20	11.503,80	1.904,74
	c=60	5.828,00	1,1	16.806,40	18.065,00	17.201,00	339,23
	c=75	4.736,00	1,1	16.764,70	21.236,10	12.658,80	2.108,05

Considerando as instâncias do tipo CBD, nota-se que o método proposto convergiu para a solução ótima, no subconjunto com 20 pedidos e capacidade do coletor fixada em 30 produtos. O método proposto é razoavelmente rápido, no entanto a alta variância nos resultados enfatiza alta dispersão destes resultados. O desvio padrão (σ) possui pouca volatilidade, mesmo com o aumento da distância máxima (Δ_{max}) para a coleta dos lotes. Nota-se que em cada subconjunto desta instância, o *gap* entre a distância máxima (Δ_{max}) e a distância mínima (Δ_{min}) vai diminuindo à medida que a capacidade do coletor (Capacidade) aumenta, em contrapartida o tempo de execução (t) é constante independente da variação da capacidade do coletor, exceto em um subconjunto. O indica que as características das instâncias causam pouco impacto no tempo

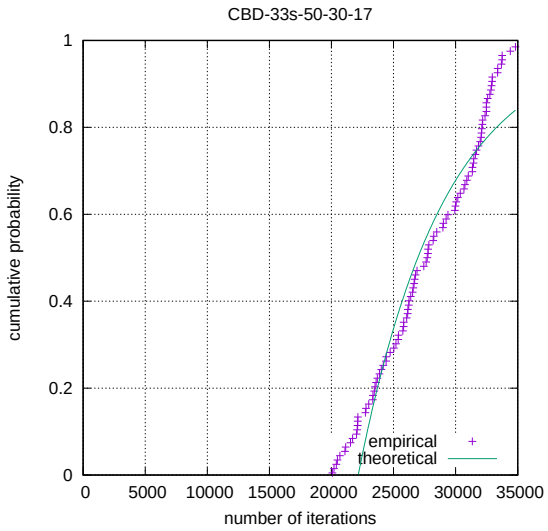
de execução do ILS.

O limite de iterações para o ILS é de 5.000 iterações. A análise de convergência demonstra que este número de iterações é baixo. Para ilustrar a convergência do ILS, para a análise de convergência, selecionou-se duas instâncias de forma aleatória de cada conjunto de instâncias para criar gráficos time-to-target (ttt) (AIEX; RESENDE; RIBEIRO, 2007) vide Figura 15.

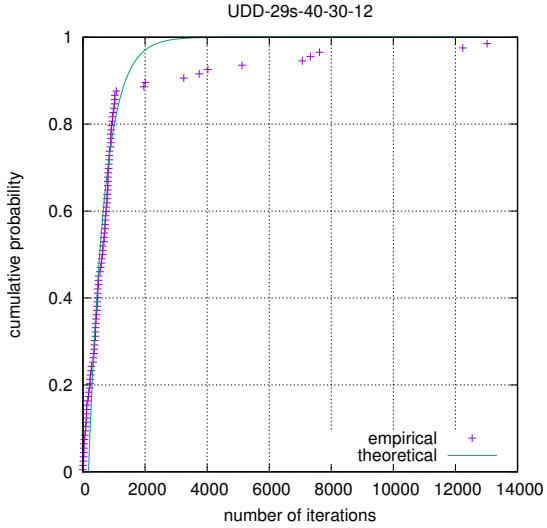
O ILS foi executado 100 vezes de forma independente para cada instância. O número de iterações necessárias, para que o ILS convergisse para uma solução com um gap máximo de 5% da melhor solução divulgada na literatura foi reportado para a elaboração do gráfico ttt. O gráfico ttt emprega uma teoria de que o tempo de execução de um algoritmo se modela à uma distribuição exponencial, desde que o método seja executado uma quantidade suficiente. Em seguida comparou-se a distribuição (*empirical*, denotada pelas cruzes roxas) com a distribuição exponencial (*theoretical*, representada pela linha verde). A Figura 15 (a) mostra que o ILS possui uma probabilidade de mais de 80% de encontrar uma solução com um *gap* máximo de 5% em até 100 iterações para esta instância. Já a Figura 15 (b) mostra que a probabilidade de 80% do ILS encontrar uma solução em aproximadamente 32.000 iterações. A Figura 15 (c) mostra que o ILS tem uma probabilidade de mais de 80% de encontrar uma solução em menos de 2.000 iterações. Na Figura 15 (d) observa-se que existe a probabilidade 80% do ILS encontrar uma solução em aproximadamente 110.000 iterações.



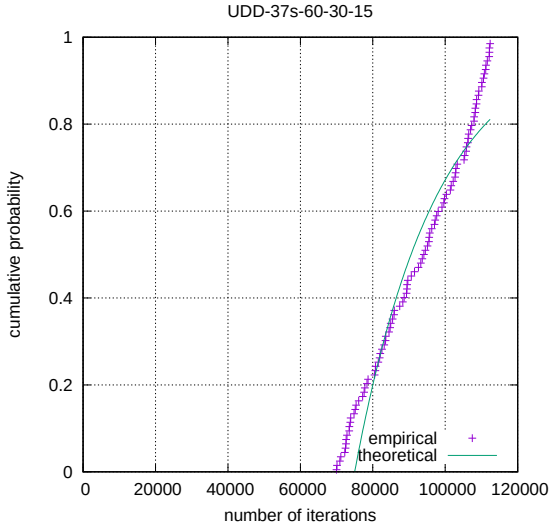
(a) Conjunto de instância CBD.



(b) Conjunto de instância CBD.



(c) Conjunto de instância UDD.



(d) Conjunto de instância UDD.

Figura 15 – Gráfico *Time-to-target plots* de instâncias aleatoriamente selecionadas.

5.4 Comparação com o estado da arte

O atual estado da arte (ŽULJ; KRAMER; SCHNEIDER, 2018) propôs o método ALNS/TS que foi abordado anteriormente no Capítulo 2. Em seus experimentos computacionais, o ALNS/TS convergiu para a melhor solução encontrada na maioria das instâncias. As tabelas a seguir apresentam os resultados do atual estado da arte e os resultados preliminares do ILS proposto. Um resumo dos resultados é apresentado a seguir em duas tabelas, confrontando os resultados do ALNS/TS. Estas tabelas aprentam a quantidade de pedidos (Pedidos), a capacidade do coletor (Capacidade), os melhores resultados encontrados na literatura (ŽULJ; KRAMER; SCHNEIDER, 2018) (BKS), sendo que os valores seguidos por * são soluções ótimas obtidas pelo Gurobi. A média dos melhores resultados (Δ_f), e o tempo de execução em segundos (t). A

Tabela 5 apresenta o sumário dos resultados para o conjunto de instâncias UDD.

Tabela 5 – Comparação de resultados entre ILS e ALNS/TS para as instâncias UDD.

Pedidos	Capacidade	BKS	ALNS/TS		ILS	
			Δ_f	t	Δ_f	t
n=40	c=30	10.462,00*	10.462,00	2,0	19.956,40	0,9
	c=45	6.864,00	6.864,00	2,0	12.065,90	0,4
	c=60	5.278,00	5.278,00	2,0	11.425,50	0,5
	c=75	4.273,00	4.273,00	3,0	9.770,90	0,4
n=60	c=30	15.482,00*	17.030,20	5,0	15.522,60	1,2
	c=45	10.032,00	10.032,00	6,0	15.720,60	1,2
	c=60	7.705,00	7.705,00	7,0	18.743,90	1,1
	c=75	6.294,00	6.294,00	8,0	17.932,50	0,6
n=80	c=30	20.645,00*	22.709,50	12,0	25.155,20	2,3
	c=45	13.328,00	13.328,00	13,0	24.368,90	2,2
	c=60	10.173,00	10.173,00	16,0	21.421,30	2,3
	c=75	8.233,00	8.233,00	17,0	25.172,70	1,3
n=100	c=30	25.540,00*	30.648,00	23,0	32.525,80	2,3
	c=45	16.357,00	16.357,00	23,0	32.384,90	3,8
	c=60	12.472,00	12.472,00	28,0	31.234,50	4,2
	c=75	10.151,00	10.151,00	33,0	31.021,70	3,6

Observa-se nesta tabela que no subconjunto com 60 pedidos e capacidade do coletor fixada em 30 produtos, o ILS obteve melhoria no resultado do ALNS/TS de mais de 8%. Nota-se também que o método manteve um baixo tempo de execução. Em contrapartida o ALNS/TS obteve melhores resultados nas demais instâncias. A Tabela 6 apresenta os resultados para o conjunto de instâncias CBD.

O ILS alcançou bons resultados em algumas instâncias do conjunto CBD. Na Tabela 6, nota-se que no subconjunto de instâncias com 60 pedidos e capacidade do coletor fixada em 30 produtos o ILS obteve uma melhoria no resultado do ALNS/TS de mais de 8% com tempo de execução com aproximadamente um segundo. No subconjunto de instâncias com 20 pedidos e capacidade do coletor fixada em 30 houve empate, no qual os dois métodos alcançaram a solução ótima de 4.192,00 LU e nas demais instâncias o ALNS/TS obteve melhores resultados.

Além disso, aplicou-se testes estatísticos adicionais nos resultados do ILS e ALNS/TS para evidenciar as diferenças destes métodos. O *Shapiro-Wilk Test* (SHAPIRO; WILK, 1965) foi aplicado e confirmou que estes resultados poderiam ser modelados de acordo com uma distribuição normal (W = 0.92917, $p - value$ = 0.03714). Posteriormente, o *Wilcoxon Signed Rank Test* (WOOLSON, 2007) foi aplicado em duas amostradas pareadas com o objetivo de averiguar se existe diferença significativa entre os resultados (V = 488, $p - value$ = 1), no qual ficou evidente que o ALNS/TS possui melhores valores médios.

Tabela 6 – Comparação de resultados entre ILS e ALNS/TS para instâncias CBD.

Pedidos	Capacidade	BKS	ALNS/TS		ILS	
			Δ_f	t	Δ_f	t
n=20	c=30	4.192,00*	4.192,00	0,0	4.192,00	0,4
	c=45	2.689,00	2.689,00	0,0	3.266,00	0,2
	c=60	2.156,00	2.156,00	0,0	3.926,20	0,2
	c=75	1.743,00	1.743,00	1,0	3.434,60	0,2
n=40	c=30	7.907,00*	7.907,00	2,0	7.967,80	0,5
	c=45	5.185,00	5.185,00	2,0	8.386,00	0,5
	c=60	3.988,00	3.988,00	3,0	18.130,80	0,9
	c=75	3.243,00	3.243,00	3,0	7.039,60	0,5
n=50	c=30	10.098,00*	10.098,00	4,0	11.241,80	0,7
	c=45	6.462,00	6.462,00	4,0	11.160,50	0,7
	c=60	4.988,00	4.988,00	5,0	11.371,80	0,7
	c=75	4.077,00	4.077,00	6,0	11.476,40	0,7
n=60	c=30	11.609,00*	12.769,90	5,0	11.680,40	1,1
	c=45	7.566,00	7.566,00	6,0	11.503,80	1,1
	c=60	5.828,00	5.828,00	7,0	17.201,00	1,1
	c=75	4.736,00	4.736,00	9,0	12.658,80	1,1

6 Plano de Atividades Restantes

Na Tabela 7 são apresentadas informações sobre atividades planejadas para os trabalhos futuros da dissertação de mestrado, visando a conclusão deste trabalho de pesquisa.

Tabela 7 – Cronograma de atividades restantes.

Tarefas	Bimestre 1	Bimestre 2	Bimestre 3	Bimestre 4	Bimestre 5	Bimestre 6
Pesquisar sobre métodos de busca local e perturbação	✓	✓	✓			
Pesquisar sobre novas vizinhanças		✓	✓	✓		
Representar o problema utilizando teoria dos grafos			✓	✓		
Pesquisar novos métodos de geração de solução inicial		✓	✓	✓		
Análise do uso de outras estratégias de roteamento			✓	✓		
Aprimorar os métodos de busca local			✓	✓		
Realizar experimentos computacionais			✓	✓		
Analisar experimentos computacionais realizados			✓	✓	✓	
Descrever os experimentos computacionais					✓	✓
Escrever artigo científico					✓	✓
Conclusão do texto da dissertação					✓	✓
Realização do exame de defesa						✓

As atividades se concentram em pesquisar, implementar e analisar novas heurísticas adaptadas à abordagem proposta. Além disso, uma nova representação do problema tratado utilizando-se a teoria de grafos será estudada, uma vez que não se encontra este tipo de representação na literatura. Após a definição e implementação dessas melhorias, novos experimentos computacionais serão executados e analisados. Além da realização deste trabalho de dissertação, objetiva-se a elaboração de um artigo cientifico em congresso nacional, como parte dos requerimentos para obtenção do título de mestre.

7 Conclusão

Neste trabalho é reportada uma abordagem para o *Order Batching Problem* (OBP), um problema provado $\mathcal{NP}$-difícil que modela uma situação prática encontrada em centros de distribuição de produtos. O OBP consiste em minimizar a distância percorrida para coletar um conjunto de pedidos de compra, agrupando-os em subconjuntos denominados lotes. Cada um destes lotes deve obedecer a capacidade máxima do coletor, uma vez que cada lote é coletado em uma única incursão no centro de distribuição por um único coletor. Neste trabalho, foi realizada a revisão da bibliografia sobre o tema, aprofundou-se a fundamentação teórica e propôs-se uma implementação preliminar da metaheurística Busca Local Iterada (ou ILS, do inglês *Iterated Local Search*) com componentes personalizados para o OBP. Os experimentos computacionais preliminares consideraram 2560 instâncias e tiveram como objetivo comparar os resultados iniciais deste trabalho com os mais recentes resultados publicados na literatura. O ILS apresentou um bom tempo de execução, embora a qualidade das soluções geradas seja diversificada. Em duas ocasiões o ILS superou o ALNS/TS, obtendo uma melhoria de mais de 8% em relação ao valor das soluções. Todavia, na maior parte dos casos o ILS gerou resultados inferiores, o que sugere que o método carece de melhorias para se tornar competitivo. Como trabalhos futuros no intuito de aprimorar os resultados obtidos pela implementação preliminar da ILS, destacam-se (i) uma nova representação do problema utilizando-se teoria dos grafos; (ii) novos métodos de geração de soluções iniciais baseados em heurísticas recentes para problemas de roteamento; (iii) novas vizinhanças; (iv) aprimoramento dos métodos de busca local; e (v) análise do uso de outras políticas de roteamento alternativas. Todas as novas implementações serão objeto de experimentos computacionais extensivos, considerando novos conjuntos de instâncias.

Referências

AIEX, R. M.; RESENDE, M. G.; RIBEIRO, C. C. Ttt plots: a perl program to create time-to-target plots. *Optimization Letters*, Springer, v. 1, n. 4, p. 355–366, 2007. Citado na página 54.

ALBAREDA-SAMBOLA, M. et al. Variable neighborhood search for order batching in a warehouse. *Asia-Pacific Journal of Operational Research*, World Scientific, v. 26, n. 05, p. 655–683, 2009. Citado na página 25.

BOZER, Y. A.; KILE, J. W. Order batching in walk-and-pick order picking systems. *International Journal of Production Research*, Taylor & Francis, v. 46, n. 7, p. 1887–1909, 2008. Citado na página 25.

CHEN, M.-C. et al. Aggregation of orders in distribution centers using data mining. *Expert Systems with Applications*, Elsevier, v. 28, n. 3, p. 453–460, 2005. Citado na página 25.

CHEN, M.-C.; WU, H.-P. An association-based clustering approach to order batching considering customer demand patterns. *Omega*, Elsevier, v. 33, n. 4, p. 333–343, 2005. Citado na página 25.

CLARKE, G.; WRIGHT, J. W. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, Informs, v. 12, n. 4, p. 568–581, 1964. Citado 8 vezes nas páginas 11, 23, 25, 26, 27, 41, 42 e 44.

COSTA, S. et al. A importância da logística para o e-commerce: o exemplo da amazon. com. In: *1º Congresso USP de Iniciação Científica em Contabilidade*. [S.l.: s.n.], 2004. Citado na página 20.

COYLE, J. J. et al. *The management of business logistics*. [S.l.]: West publishing company St Paul, MN, 1996. v. 6. Citado na página 21.

DRURY, J. Towards more efficient order picking. *IMM monograph*, The Institute of Materials Management, Cranfield, UK, v. 1, 1988. Citado na página 21.

ELSAYED, E. A. Algorithms for optimal material handling in automatic warehousing systems. *The International Journal of Production Research*, Taylor & Francis, v. 19, n. 5, p. 525–535, 1981. Citado 2 vezes nas páginas 23 e 24.

ELSAYED, E. A.; STERN, R. G. Computerized algorithms for order processing in automated warehousing systems. *The International Journal of Production Research*, Taylor & Francis, v. 21, n. 4, p. 579–586, 1983. Citado na página 23.

ELSAYED, E. A.; UNAL, O. Order batching algorithms and travel-time estimation for automated storage/retrieval systems. *The International Journal of Production Research*, Taylor & Francis, v. 27, n. 7, p. 1097–1114, 1989. Citado 2 vezes nas páginas 23 e 24.

FRAZELLE, E.; FRAZELLE, E. *World-class warehousing and material handling*. [S.l.]: McGraw-Hill New York, 2002. v. 1. Citado na página 21.

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. Citado 3 vezes nas páginas 21, 25 e 32.

GHIANI, G.; LAPORTE, G.; MUSMANNO, R. *Introduction to logistics systems planning and control*. [S.l.]: John Wiley & Sons, 2004. Citado na página 20.

GIBSON, D. R.; SHARP, G. P. Order batching procedures. *European Journal of Operational Research*, Elsevier, v. 58, n. 1, p. 57–67, 1992. Citado 2 vezes nas páginas 23 e 24.

GLOVER, F. Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, Elsevier, v. 13, n. 5, p. 533–549, 1986. Citado na página 28.

GOETSCHALCKX, M.; RATLIFF, H. D. Order picking in an aisle. *IIE transactions*, Taylor & Francis, v. 20, n. 1, p. 53–62, 1988. Citado 2 vezes nas páginas 23 e 33.

GROSS, J. Picking methods may provide key to lower cost warehouse plans. *Industrial Engineering*, INST INDUSTRIAL ENGINEERS 25 TECHNOLOGY PARK/ATLANTA, NORCROSS, GA 30092, v. 13, n. 6, p. 50, 1981. Citado na página 21.

GROSSE, E. H. et al. A simulated annealing approach for the joint order batching and order picker routing problem with weight restrictions. *International Journal of Operations and Quantitative Management*, v. 20, n. 2, p. 65–83, 2014. Citado na página 19.

HALL, R. W. Distance approximations for routing manual pickers in a warehouse. *IIE transactions*, Taylor & Francis, v. 25, n. 4, p. 76–87, 1993. Citado 4 vezes nas páginas 23, 24, 33 e 35.

HENN, S. Algorithms for on-line order batching in an order picking warehouse. *Computers & Operations Research*, Elsevier, v. 39, n. 11, p. 2549–2563, 2012. Citado na página 28.

HENN, S. et al. Metaheuristics for the order batching problem in manual order picking systems. *Business Research*, Springer, v. 3, n. 1, p. 82–105, 2010. Citado 7 vezes nas páginas 11, 25, 26, 43, 45, 46 e 51.

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. Citado 4 vezes nas páginas 26, 28, 29 e 51.

HO, Y.-C.; TSENG, Y.-Y. A study on order-batching methods of order-picking in a distribution centre with two cross-aisles. *International Journal of Production Research*, Taylor & Francis, v. 44, n. 17, p. 3391–3417, 2006. Citado na página 25.

HSU, C.-M.; CHEN, K.-Y.; CHEN, M.-C. Batching orders in warehouses by minimizing travel distance with genetic algorithms. *Computers in Industry*, Elsevier, v. 56, n. 2, p. 169–178, 2005. Citado na página 25.

HWANG, H.; BAEK, W. J.; LEE, M.-K. Clustering algorithms for order picking in an automated storage and retrieval system. *International Journal of Production Research*, Taylor & Francis, v. 26, n. 2, p. 189–201, 1988. Citado 2 vezes nas páginas 21 e 23.

HWANG, H.; KIM, D. Order-batching heuristics based on cluster analysis in a low-level picker-to-part warehousing system. *International Journal of Production Research*, Taylor & Francis, v. 43, n. 17, p. 3657–3670, 2005. Citado na página 25.

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. Citado na página 21.

KEARNEY, E.; KEARNEY, A. Excellence in logistics 2004. *European Logistics Association, Brussels*, 2004. Citado na página 21.

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. Citado 2 vezes nas páginas 24 e 33.

LOURENÇO, H. R.; MARTIN, O. C.; STÜTZLE, T. Iterated local search. In: *Handbook of metaheuristics*. [S.l.]: Springer, 2003. p. 320–353. Citado 3 vezes nas páginas 37, 38 e 40.

MUTER, İ.; ÖNCAN, T. An exact solution approach for the order batching problem. *IIE Transactions*, Taylor & Francis, v. 47, n. 7, p. 728–738, 2015. Citado na página 26.

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. Citado 3 vezes nas páginas 26, 28 e 51.

PAN, C.-H.; LIU, S. A comparative study of order batching algorithms. *Omega*, Elsevier, v. 23, n. 6, p. 691–700, 1995. Citado 2 vezes nas páginas 23 e 24.

PETERSEN, C. G. An evaluation of order picking routeing policies. *International Journal of Operations & Production Management*, MCB UP Ltd, v. 17, n. 11, p. 1098–1111, 1997. Citado 2 vezes nas páginas 20 e 32.

ROPKE, S.; PISINGER, D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, Informs, v. 40, n. 4, p. 455–472, 2006. Citado na página 27.

ROSENWEIN, M. A comparison of heuristics for the problem of batching orders for warehouse selection. *International Journal of Production Research*, Taylor & Francis, v. 34, n. 3, p. 657–664, 1996. Citado 2 vezes nas páginas 23 e 24.

SHAMLATY, R. This is not your father's warehouse. *IIE Solutions*, Institute of Industrial and Systems Engineers (IISE), v. 32, n. 1, p. 30–30, 2000. Citado na página 20.

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. Citado na página 56.

SHAW, P. Using constraint programming and local search methods to solve vehicle routing problems. In: SPRINGER. *International conference on principles and practice of constraint programming*. [S.l.], 1998. p. 417–431. Citado na página 27.

STÜTZLE, T. Local search algorithms for combinatorial problems. *Darmstadt University of Technology PhD Thesis*, Citeseer, v. 20, 1998. Citado na página 37.

TOMPKINS, J. et al. A,(2003) facilities planning. *John Willey & Sons*, 2003. Citado 2 vezes nas páginas 20 e 21.

WANG, Y.; WANG, Z.; MI, S. An order batching clustering algorithm of fixed maximum order number based on order picking system. In: IEEE. *Industrial Economics System and Industrial Security Engineering (IEIS'2017), 2017 4th International Conference on*. [S.l.], 2017. p. 1–6. Citado na página 27.

WOOLSON, R. Wilcoxon signed-rank test. *Wiley encyclopedia of clinical trials*, Wiley Online Library, p. 1–3, 2007. Citado na página 56.

ZHAO, Z.; YANG, P. Improving order-picking performance by optimizing order batching in multiple-cross-aisle warehouse systems: A case study from e-commerce in china. In: IEEE. *Industrial Engineering and Applications (ICIEA), 2017 4th International Conference on*. [S.l.], 2017. p. 158–162. Citado na página 27.

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. Citado 6 vezes nas páginas 27, 29, 33, 51, 52 e 55.