UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

Juan Camilo Fonseca Galindo

Mineração de Dados Aplicada a Problemas de Last-Mile na Logística

Belo Horizonte, MG
Dezembro/2021

Juan Camilo Fonseca Galindo

Mineração de Dados Aplicada a Problemas de Last-Mile na Logística

Tese de Doutorado submetida à banca examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica. Área de concentração: **Engenharia de Computação e Telecomunicações**.

Co-orientador: Prof. Dr. Cristiano Leite de Castro

Orientador: Prof. Dr. André Paim Lemos

Belo Horizonte, MG

Dezembro/2021

G157m

Galindo, Juan Camilo Fonseca.
Mineração de dados aplicada a problemas de last-mile na logística [recurso eletrônico] / Juan Camilo Fonseca Galindo. – 2021.
1 recurso online (xi, 89 f. : il., color.) : pdf.

Orientador: André Paim Lemos.
Coorientador: Cristiano Leite de Castro.

Tese (doutorado) Universidade Federal de Minas Gerais, Escola de Engenharia.

Bibliografia: f. 75-89.
Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Logística - Teses. 3. Comércio eletrônico - Teses. 4. Mineração de dados (Sistemas de recuperação da informação) - Teses. I. Lemos, André Paim. II. Castro, Cristiano Leite de. III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pela Bibliotecária Roseli Alves de Oliveira CRB/6 2121
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG.

"Mineração de Dados Em Trajetórias Aplicado Em Problemas de Last-mile"

Juan Camilo Fonseca Galindo

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica.

Aprovada em 06 de dezembro de 2021.

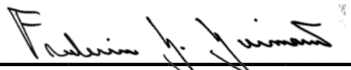
Por:

Prof. Dr. André Paim Lemos
DELT (UFMG)

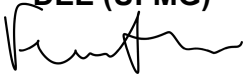
Cristiano Leite de Castro:03680288603

Digitally signed by Cristiano Leite de Castro:03680288603
Date: 2021.12.09 14:15:34 -03'00'

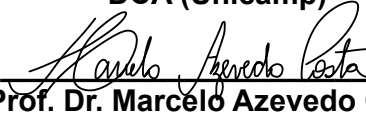
Prof. Dr. Cristiano Leite de Castro
DEE (UFMG)



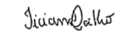
Prof. Dr. Frederico Gadelha Guimarães
DEE (UFMG)



Prof. Dr. Fernando Antônio Campos Gomide
DCA (Unicamp)



Prof. Dr. Marcelo Azevedo Costa
DEP (UFMG)



Prof. Dr. Ticiana Linhares Coelho da Silva
Instituto Universidade Virtual (UFC)

Dedico mi título de "Doutor em Engenharia Elétrica"a mis padres y mi abuelita Teresa Quiroz, por su apoyo constante y amor incondicional.

Agradecimentos

A Deus por dar-me a oportunidade de realizar meu Doutorado no Brasil, especialmente na UFMG.

A minha família, especialmente a meus pais pelo seu amor constante.

A meu orientador, André Paim Lemos, e o meu co-orientador, Cristiano Leite de Castro, por sua paciência e seu suporte profissional.

A Gabriela Surita, pelo tempo, conhecimento, confiança, e amizade proporcionada.

A meus amigos, Jaime Arturo Dulce Galindo e Juan José Quiroz Omaña.

À Loggi, em especial Davi Reis e Bruno Fonseca, pela confiança, espaço e oportunidade de desenvolver a minha pesquisa.

Aos professores e colegas do LITC: Frederico Coelho, Luiz Carlos Bambirra, José Maia Neto e Honovan Rocha, pelo tempo e conhecimento compartilhado. Agradeço especialmente ao professor Antônio Braga por seu compromisso no grupo de pesquisa e por permitir-me desenvolver meu trabalho de doutorado.

Á CAPES pelo incentivo financeiro concedido, imprescindível para a dedicação integral ao doutorado.

Resumo

Este trabalho propõe a aplicação de técnicas de mineração de dados em problemas de logística de *e-commerce*. Inicialmente, é proposta uma metodologia para solução do problema de roteirização de *last-mile* em entregas de *e-commerce*. A proposta é baseada em um sistema multiagente que usa técnicas de mineração de dados em trajetórias para extrair padrões territoriais e usá-los na criação dinâmica de rotas de *last-mile*. Em seguida, o problema de atribuição de rotas de *last-mile* para mensageiros e o problema de detecção de rotas anômalas são abordados a partir de um algoritmo de agrupamento incremental baseado em uma mistura de densidades. A abordagem incremental foi proposta como solução na logística de *e-commerce* devido ao grande volume das entregas observados nos últimos anos, tornando inviável o uso de técnicas treinadas em bateladas. O algoritmo de agrupamento proposto é baseado no framework TEDA, o qual divide o problema de clusterização em dois subproblemas: *micro-clusters* e *macro-clusters* representados por estruturas de dados que favorecem o armazenamento eficiente em memória e permitem a sua escalabilidade em grandes bases de dados. As metodologias propostas foram avaliadas em bases de dados de trajetórias reais de uma empresa de logística brasileira. O banco de dados usado nos testes contém dezenas de milhares de pacotes entregues em milhares de rotas, comprovando a eficiência das abordagens devido aos seus baixos custos computacionais. As abordagens propostas foram comparadas com algoritmos estado-da-arte, comprovando o desempenho, robustez e eficiência, especialmente em grandes volumes de dados devido aos seus baixos custos computacionais.

Palavras-chave: Logística de *E-commerce*, Problema de roteirização dinâmica de veículos capacitados com clientes estocásticos, algoritmo de agrupamento incremental, mineração de dados em trajetórias, fluxos de dados contínuos.

Abstract

This work proposes the application of trajectory data mining techniques to e-commerce logistics problems. Initially, a methodology for solving the last-mile routing problem in e-commerce deliveries is proposed. The proposal is based on a multi-agent system that uses trajectory data mining techniques to extract territorial patterns and use them in the dynamic creation of last-mile routes. Next, the problem of assigning last-mile routes to messengers and the problem of detecting an online outlier in last-mile routes are addressed using an evolving clustering algorithm based on mixture of densities. The evolving approach was proposed as a solution in e-commerce logistics due to the large volume of deliveries observed in recent years, making the use of techniques trained in batches infeasible. The proposed clustering algorithm is based on the TEDA framework, which divides the clustering problem into two sub-problems: micro-clusters and macro-clusters represented by data structures that favor efficient memory storage and enables scalability in large databases. The proposed methodologies were evaluated in databases of real trajectories of a Brazilian logistics company. The database used in the tests contains tens of thousands of packages delivered on thousands of routes, proving the efficiency of the approaches due to their low computational costs. The proposed approaches were compared with state-of-the-art algorithms, proving their performance, robustness and efficiency, especially in large data volumes due to their low computational costs.

Keywords: E-commerce Logistics, Dynamic Capacitated Vehicle Routing Problem with Stochastic Customers, Trajectory Data Mining, Big Data, Evolving Clustering Algorithm, Stream Data Mining.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

Nos últimos anos, os avanços na tecnologia tem permitido adquirir e armazenar grandes bases de dados, gerando como desafio o desenvolvimento de novos métodos capazes de extrair conhecimento oculto com o propósito de melhorar a análise e interpretação a partir destes dados disponíveis. Abonyi and Feil (2007) descrevem o processo de descobrir conhecimento em bases de dados KDD (do inglês *knowledge discovery in dataset*) em cinco etapas principais: seleção, pré-processamento, transformação, mineração e, avaliação e interpretação. Estas etapas são apresentadas na Figura 1.1, onde as setas pretas representam o fluxo da evolução da informação em cada uma das etapas. Portanto, se a informação obtida na avaliação e interpretação não é relevante ou não compreensível o processo pode voltar nas etapas anteriores com a finalidade de melhorar os resultados obtidos. Isto é representado através das setas pontilhadas na figura. O propósito geral de cada uma destas etapas é:

- Seleção: a primeira etapa no processo de KDD consiste na coleta dos dados e em seguida compreendê-los para identificar problemas de qualidade. Logo, criar uma percepção nos dados ou detectar conjuntos interessantes para realizar hipótese da informação oculta.

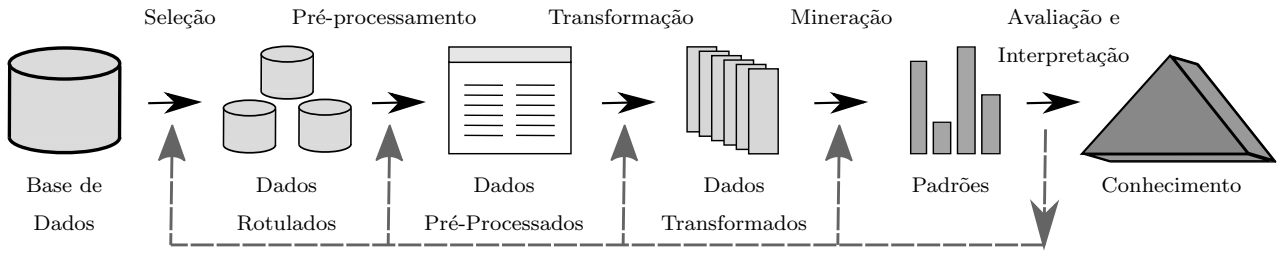


Figura 1.1: Processo para descobrir o conhecimento em base de dados (Abonyi and Feil, 2007; Fayyad et al., 1996).

- Pré-processamento: esta é a etapa de preparação dos dados. Para este propósito são utilizadas técnicas para selecionar atributos, transformar e deletar dados incompletos (*data cleaning*), remover o ruído presente nos dados, entre outras.

- Transformação: são utilizadas técnicas de redução de dimensionalidade e projeção dos dados visando encontrar características úteis para representar os dados. *Artificial neural networks, cluster analysis* e *neuro-fuzzy systems* são utilizados parar reduzir o número de variáveis e encontrar uma representação invariante dos dados (Abonyi and Feil, 2007; Witten et al., 2016).

- Mineração de dados ou *Data Mining*: nessa etapa precisa-se selecionar o algoritmo apropriado para a procura dos padrões nos dados. São definidos os parâmetros do modelo, tempo de execução, critério de avaliação e algoritmo de otimização a utilizar. As principais tarefas dos algoritmos utilizados são o agrupamento, sumarização, regressão e classificação dos dados, para tanto são utilizadas regras de classificação, árvores ou representações espaciais.

- Avaliação e interpretação: nessa etapa é consolidado o conhecimento descoberto nos dados a partir do critério do especialista.

O aumento da penetração da Internet e o fácil acesso ao pagamento *online* criaram um ambiente favorável para vendas de comércio eletrônico[1]. No entanto, esse novo mercado também trouxe desafios significativos para a logística de entrega, principalmente a demanda por entregas no mesmo dia de baixo custo.

[1]Foundation News, Ecommerce Foundation (accessed December 15, 2020), `https://retailx.net/product/ecommerce-report-brazil-2019/`

A tarefa principal da logística de entrega consiste em transportar um pacote desde o embarcador até o cliente que comprou o pacote. Esta tarefa pode-se caracterizar em sub-problemas categorizados em três etapas, *First-mile*, *middle-mile* e *last-mile*. *First-mile* consiste em coletar o pacote do embarcador e transportar para uma instalação, onde é realizado o processo de separação dos pacotes entre veículos para ser transportados para outras cidades ou instalações mais próximos do cliente. O transporte entre as instalações é conhecido como *middle-mile*. Na instalação mais próxima do destinatário, os pacotes são consolidados em rotas, as quais são percorridas por mensageiros, para finalmente, entregar os pacotes ao cliente final. Esta ultima etapa de transporte, é conhecida como *last-mile*. O problema consiste em como separar os pacotes em veículos e consolidar os pacotes nas rotas com o objetivo de minimizar custos e reduzir o tempo da entrega dos pacotes.

O Brasil teve um aumento exponencial nas vendas de comércio eletrônico em 2019, especialmente em produtos eletrônicos, casa, decoração, saúde, cosméticos e perfumaria[2]. O país tem uma população de mais de 200 milhões e uma extensão territorial continental (maior que $8,5$ milhões de km^2) com grandes diferenças socioeconômicas entre as regiões. A porcentagem da população de *e-shoppers* em 2019 era de $50,6\%$, com mais de 80% deles concentrados nas regiões sul e sudeste [3]. Os entregadores costumam se especializar nessas regiões, realizando entregas *express* em locais remotos com duração superior a seis dias e com custos superiores a 20 dólares por entrega [4], quando possível.

Consequentemente, este cenário gera a necessidade do desenvolvimento de técnicas que aproveitem a disponibilidade de grandes volumes de dados históricos de entregas para extrair conhecimento e aplicá-lo em modelos capazes de realizar as tarefas de separação e consolidação visando minimizar o custo e o tempo das entregas. Dados de entregas pode ser representados por trajetórias das rotas percorridas pelos mensageiros, portanto, o uso de mineração de dados em trajetórias possibilitam a extração de conhecimento explícito da cidade e do comportamento da população que pode ser usado na modelagem dos algoritmos. Os principais desafios da logística de *e-commerce* no Brasil são:

- o Brasil é um país com uma grande extensão territorial e com grandes diferenças socioeconômicas entre as regiões. Isto torna ineficiente o uso de técnicas desenvolvidas em outros países, precisando construir modelos capazes de extrair conhecimento das especificidades do comportamento da população.

[2]WEBSHOPPERS, ebit a Nielsen Company (accessed December 15, 2020), `https://www.fecomercio.com.br/public/upload/editor/ws38_vfinal.pdf`

[3]Foundation News, Ecommerce Foundation (accessed December 15, 2020), `https://retailx.net/product/ecommerce-report-brazil-2019/`

[4]Correios Brasil (accessed December 15, 2020), `http://www2.correios.com.br/sistemas/precosPrazos/`

- O aumento da quantidade de pacotes vendidos diariamente pelas empresas de *e-commerce* e a necessidade de que estes sejam entregues no menor tempo possível, impossibilita o armazenamento destes pacotes limitando o uso de técnicas de otimização estáticas para a construção das rotas das entregas. Este cenário requer o uso de técnicas para análise de *Big Data* e fluxos de dados contínuos capazes de distribuir os pacotes entre os mensageiros de forma eficiente, para isso o modelo tem que conseguir prever a chegada de futuros pacotes baseado no histórico das entregas realizadas.

- Devido à extensão territorial e à quantidade de cidades do Brasil, não é viável ter automação na separação dos pacotes em todas as instalações. Portanto, muitos dos processos na separação são realizados manualmente, o que torna o processo susceptível a erros. Estes erros impactam diretamente o custo de entrega do pacote já que podem representar misturar pacotes entre cidades, instalações ou rotas de *last-mile*. Consequentemente, é indispensável ter uma etapa de detecção de *outlier* capaz de identificar as falhas operacionais, prevendo caminhos não desejados dos pacotes e finalmente reduzindo o custo e tempo de entrega.

1.2 Objetivos

O objetivo principal deste trabalho é desenvolver metodologias baseadas em técnicas de mineração de dados em trajetórias capazes mitigar as lacunas presentes na logística de *e-commerce* no Brasil. Os procedimentos propostos devem possuir a capacidade de:

- processar pacotes de forma *online*, removendo a etapa de armazenamento presente na maioria de modelos de roteirização.

- possuir tempo de processamento baixo, para poder ser acoplados com *hardware* usados na logística de entrega.

- ser capaz extrair conhecimento das especificidades do comportamento da população e das características geográficas da cidade.

- trabalhar de forma evolutiva, ou seja, conseguir se adaptar se existirem mudanças nas entregas, detectadas no banco de dados.

Duas metodologias são propostas nesse trabalho. A primeira pretende resolver o problema de roteirização de *last-mile* em entregas de *e-commerce*. A proposta apresenta um sistema multiagente que usa técnicas de mineração de dados em trajetórias para extrair padrões territoriais e usá-los na criação dinâmica de rotas de *last-mile*. O problema pode ser definido como Problema

de Roteirização Dinâmica de Veículos Capacitados com Clientes Estocásticos (DCVRPSC do inglês *Dynamic Capacitated Vehicle Routing Problem with Stochastic Customer*) (Huang et al., 2018; Bent and Van Hentenryck, 2004; Van Hemert and La Poutré, 2004; Van Hentenryck et al., 2010). Porém, essa metodologia difere dos demais modelos propostos na literatura:

- o problema de roteirização, que é NP-HARD, é resolvido como uma heurística com tempo de execução linear dadas especificações físicas do sistema *Warehouse* e não em função da quantidade de pacotes processados, o qual é apropriado para cenários de *Big Data*, presentes em entrega de produtos *e-commerce*.
- a heurística é construida no contexto de sistemas multiagentes, o qual permite a separação dinâmica dos pacotes, removendo a etapa de armazenamento.
- a solução usa técnicas de mineração de dados em trajetórias históricas, trazendo conhecimento intrínseco do comportamento das população e das especificidades geográficas da região.

A segunda metodologia aborda os problemas de distribuição de rotas entre os mensageiros e a detecção de rotas anômalas a partir de um algoritmo de agrupamento incremental de trajetórias. Essa metodologia permite separar o problema de roteirização do problema de atribuição de rotas aos mensageiros. Isto faz que as duas etapas sejam realizadas de forma independente em lugares diferentes na malha logística. Assim, essa metodologia de modelagem difere dos métodos existentes:

- o algoritmo é incremental, portanto consegue se adaptar às mudanças no comportamento do mercado, como a abrangência nas cidades e abertura de novos clientes.
- o problema de agrupamento é dividido em dois subproblemas: *micro-clusters* e *macro-clusters* representados por estruturas de dados que favorecem o armazenamento eficiente em memória e permitem a sua escalabilidade em grandes bases de dados.
- o algoritmo possui uma etapa de detecção de *outlier* permitindo monitorar e detectar rotas anômalas antes de serem expedidas, representando uma redução de custo para a empresa já que estes pacotes podem ser enviados de uma forma eficiente em outras rotas.

Por fim, este trabalho tem por objetivo mostrar que as abordagens propostas são promissoras para a implementação em empresas de logística de *e-commerce*. Os experimentos foram avaliados em bases de dados de trajetórias reais de uma empresa de logística brasileira. As abordagens propostas foram comparadas com algoritmos estado-da-arte, comprovando o desempenho, robustez e eficiência, especialmente em grandes volumes de dados devido aos seus baixos custos computacionais.

1.3 Trabalhos Publicados

As seguintes publicações foram realizadas ao longo da realização deste trabalho:
Publicação em periódico científico indexado:

- Fonseca-Galindo, J.C.; Surita G C; Neto, J.M; de Castro C.L. and Lemos, A.P. **"A Multi-Agent System for Solving the Dynamic Capacitated Vehicle Routing Problem with Stochastic Customers using Trajectory Data Mining"**, Expert Systems with Applications, 2022.

- Neto, J.M; Junior C. A.; Guimarães F. G.; de Castro C.L.; Lemos, A.P. Fonseca-Galindo, J.C. and Cohen M. G. **"Evolving clustering algorithm based on mixture of typicalities for stream data mining"**, Future Generation Computer Systems, 2020.

Publicações em congressos nacionais:

- Neto, J.M; Fonseca-Galindo, J.C.; de Castro C.L. and Lemos, A.P. **"Algoritmos de Map-Matching Online para Processamento de Trajetórias de Veículos: Um Estudo Comparativo"**, XIII Simpósio Brasileiro de Automação Inteligente, 2017.

- Fonseca-Galindo, J.C.; Neto, J.M; de Castro C.L. and Lemos, A.P. **"Modelo Logístico para *Map-Matching Online* de Trajetória de Veículos"**, XIII Congresso Brasileiro de Inteligência Computacional, 2017.

Publicação em congressos nacionais realizado neste período, mas não relacionada ao tópico da Tese:

- Fonseca-Galindo, J.C.; Torres, L.B.; Lacerda G.R.; Neto, P.C. and Braga A.P. **"CML-Simplex: uma abordagem de programação linear para classificadores incrementais de margem larga"**, XIII Congresso Brasileiro de Inteligência Computacional, 2017.

1.4 Organização do Texto

Após essa introdução, o capítulo 2 realiza uma breve revisão do presente estado-da-arte das técnicas utilizadas nas etapas do processo de descobrimento de conhecimento em bases de dados de trajetórias (Figura 1.1). Seguidamente, a mesma seção introduz o modelo de entrega de uma empresa brasileira especializada em logística de *e-commerce*, apresentando os três problemas os quais se pretende resolver com as duas metodologias propostas nos seguintes capítulos.

No capítulo 3 é apresentada a abordagem de um sistema multi-agente para resolver o *Dynamic Capacitated Vehicle Routing Problem with Stochastic Customers* usando mineração de dados em trajetórias, o algoritmo foi desenvolvido para a roteirização de *last-mile* em logística de e-commerce.

No capítulo 4 é descrita a segunda metodologia proposta, nesse trabalho é realizada uma prova de conceito do uso de um modelo de agrupamento incremental o qual pretende solucionar o problema de gerenciamento de trajetórias de *last-mile* entre os mensageiros e a detecção de rotas anômalas na roteirização.

Finalmente, o Capítulo 5 conclui o trabalho, resumindo suas contribuições e propondo os tópicos a serem desenvolvidos como continuidade.

Capítulo 2

Referencial Teórico

2.1 Introdução

Esse capítulo realiza uma revisão dos principais trabalhos propostos na literatura na extração de conhecimento em dados de trajetórias. O objetivo dessa revisão não é detalhar todas as possíveis aplicações nem todos os modelos propostos nessa área, mas sim introduzir, descrever e discutir como diferentes autores formulam os problemas e usam bases de dados para poder extrair o conhecimento e assim, usar-los em diferentes aplicações.

As duas abordagens apresentadas nesse trabalho são aplicadas a problemas presentes na logística de *e-commerce* de empresas do Brasil. Assim, esse capitulo também apresenta o modelo logístico usado em uma empresa brasileira. Em seguida são apresentados os problemas que serão considerados neste trabalho.

A seção 2.2 apresenta o processo de descoberta do conhecimento proposto por Abonyi and Feil (2007) aplicado em base de dados (KDD), e faz uma revisão da literatura em cada uma das suas etapas aplicado em trajetórias. Em seguida, na 2.3 são apresentados os problemas atuais na logística de *e-commerce* aplicado em uma empresa brasileira e, em seguida, realizada uma revisão da literatura em VRP *Fully Dynamic* (DRVP) com consumidores estocásticos aplicados a problemas de *Big Data* e em técnicas de mineração de dados de trajetórias aplicados em grandes volumes de dados que podem ser usados no problema de *last-mile*.

2.2 KDD aplicado em Trajétorias

Abonyi and Feil (2007) apresentam um processo de descoberta do conhecimento em bases de dados (KDD), esse processo foi apresentado na Figura 1.1 do capítulo anterior. O KDD é dividido em 5 etapas: seleção, pré-processamento, transformação e mineração de dados, avaliação e interpretação. Nas seguintes sub-seções são apresentados os principais trabalhos em cada uma destas etapas aplicados em dados de trajetórias.

2.2.1 Bases de Dados

Uma trajetória pode ser definida como a amostragem do deslocamento de um objeto através do espaço-tempo. Os avanços tecnológicos em telemetria facilitaram a obtenção da localização de um objeto, permitindo armazenar trajetórias de veículos, pessoas, animais ou fenômenos naturais. Formalmente, $p = \{x, y, t\}$ representa a localização espaço-temporal de um objeto, onde x e y são os ângulos da latitude e longitude que representam a posição sobre a Terra referenciada em relação a Linha do Equador e ao Meridiano de Greenwich, respectivamente, e t é o instante do tempo que foi realizada a amostra. Sensores modernos oferecem medidas extras como a altitude z do objeto, geralmente em referência ao nível médio do mar, e o ângulo de direção (*heading*) do objeto com relação ao norte h, complementando a representação da localização de um objeto como $p = \{x, y, z, h, t\}$. Portanto, uma trajetória é uma sequência de pontos de localização, $T = \{p_1, p_1, ..., p_n\}$, onde n é a quantidade de pontos que descrevem a trajetória.

Wu et al. (2018) e Zheng (2015) classificaram as bases de dados de trajetórias em ativas e passivas, dependendo de como foram armazenados os dados. Esta classificação foi utilizada para contextualizar o tipo de base de dados deste trabalho. Outro tipo de informação muito utilizada na área é o mapa digital da rede rodoviária ou *road network*, que é um grafo que representa a distribuição espacial das ruas em uma região determinada. Os tipos de trajetórias e o mapa digital são explicados detalhadamente na continuação desta seção.

Base de dados de trajetórias ativas

Na atualidade o uso de redes sociais (Facebook, Instagram, Twitter, etc.) permite compartilhar informação junto com a localização do usuário. Além disso, permite associar as informações com *tags* de pontos de interesse (POI) como lojas, parques, restaurantes, shoppings, cidades ou países. A união temporal dessa informação pode ser representada por uma trajetória, onde a localização é associada com um POI.

A imagem da esquerda da Figura 2.1 apresenta este tipo de trajetória, através de setas azuis pontilhadas são mostrados os POI's visitados pelos usuários. Com a informação disponibilizada

pelos perfis dos usuários é possível gerar um grafo da correlação dos usuários, apresentado no quadro superior da direita da Figura 2.1, onde as conexões das arestas representam a união de usuários similares. Da mesma forma, baseado em informação *a priori* da região é possível gerar o grafo de correlação dos POI's, apresentado no quadro inferior da direita da Figura 2.1, sendo por exemplo maior a correlação entre uma academia e um centro esportivo do que uma academia e uma igreja. Este tipo de informação é muito utilizada em sistemas de recomendação (Liu and Wang, 2017; Ravi and Vairavasundaram, 2016; Bao et al., 2015).

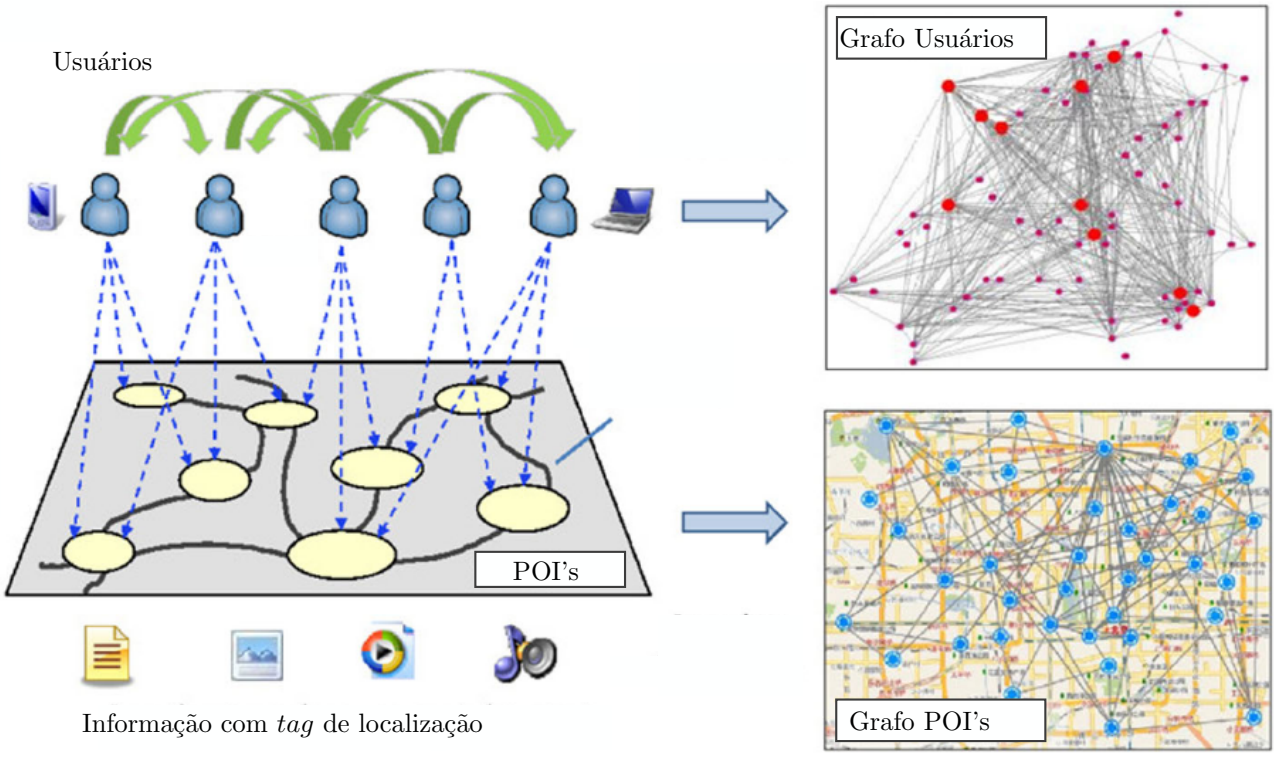
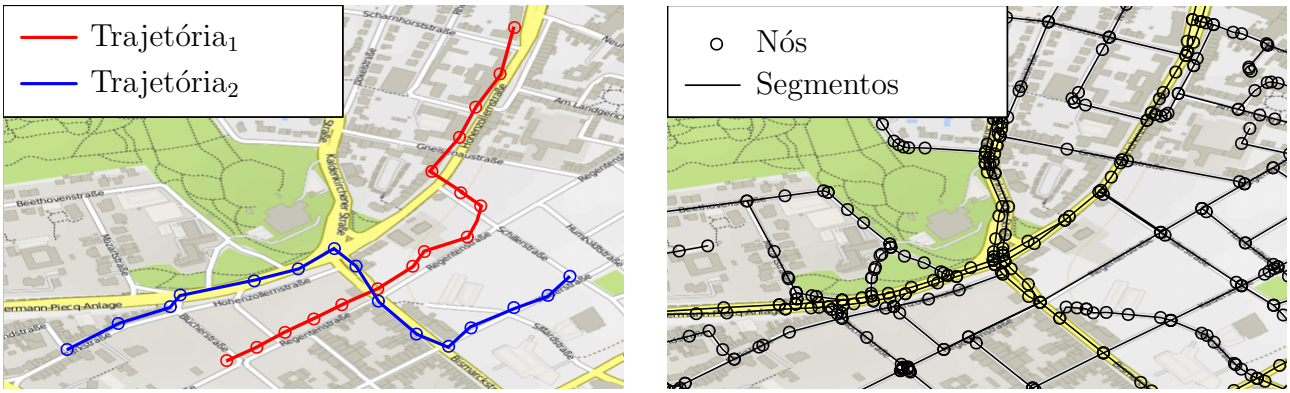


Figura 2.1: Base de dados de trajetórias ativas (Bao et al., 2015).

Dados de trajetórias passivas

O desenvolvimento de sistemas de posicionamento global tem permitido que muitos objetos em movimento sejam equipados com dispositivos que armazenam a informação da posição. A amostragem feita por esses dispositivos gera uma sequência de localização espaço-temporal que é interpretada como uma trajetória passiva, diferente das trajetórias ativas que são representadas por POI's extraídos de informação publicada em redes sociais. A Figura 2.2a representa este

tipo de trajetória, onde os círculos vermelhos e azuis representam as amostras realizadas pelo sistema de posicionamento do objeto 1 e 2, respectivamente. A união temporal destas amostras, representadas pelas linhas vermelhas e azuis, correspondem às trajetórias passivas de cada um dos objetos.



a) b)

Figura 2.2: a) Base de dados de trajetórias passivas, b) mapa digital.

Mapa Digital

O mapa digital representa a rede rodoviária de uma região na forma de um grafo, no qual os segmentos correspondem às ruas e os nós às interseções entre ruas (Zheng, 2015), conforme ilustrado na Figura 2.2b. Os mapas digitais podem conter informação de elementos da rede como o número de pistas, direção, limite máximo de velocidade e tipo de carro autorizado para transitar nas ruas (Kang et al., 2017). O mapa digital tem sido amplamente utilizado no preprocessamento de dados de trajetórias (Gong et al., 2018; Liu et al., 2017b; Hashemi and Karimi, 2016, 2014), onde o objetivo principal desses trabalhos é casar as trajetórias com o mapa digital para melhorar a qualidade dos dados.

2.2.2 Pré-processamento dos Dados

Dispositivos móveis incluindo celulares, relógios, pulseiras e veículos possuem um papel importante na vida diária. Muitos destes dispositivos incluem sistemas de navegação inercial, geralmente fornecidos por sensores como o GPS ou o GLONASS. No entanto, precisa-se considerar a imprecisão inerente à localização dos sensores, uma vez que o sinal torna-se pouco confiável quando está operando perto de obstáculos, por causa das reflexões multi-caminho.

Outros problemas consistem na exposição a interferências, especialmente quando estiverem funcionando em frequências civis (Fonseca-Galindo and Lemos, 2016). Por esse motivo, faz-se necessária uma etapa de pré-processamento dos dados.

Zheng (2015) estabelece cinco grupos de técnicas básicas para o pré-processamento de bases de dados de trajetórias: *noise filtering*, técnicas de filtragem para reduzir o ruído presente nos sensores; *stay point detection*, técnicas utilizadas para detectar POIs visitados pelos usuários, tais como shoppings, estações de serviço e atrações para turistas; compressão de trajetórias, técnicas que reduzem a quantidade de pontos que representam uma trajetória, isto quando sensores de alta resolução geram grandes quantidades de pontos de localização que não são necessários em muitas aplicações; segmentação de trajetórias, técnicas que segmentam ou dividem trajetórias baseadas em intervalos de tempo ou em sua forma espacial; *map-matching* que são técnicas que convertem os pontos de coordenadas de latitude e longitude em sequência de segmentos de ruas, ou em pontos sobre os segmentos das ruas. Esta técnica utiliza o mapa digital para melhorar a informação das trajetórias e reduzir erros dos sensores.

2.2.3 Transformação e Mineração de Dados

Após a implementação da etapa de pré-processamento, precisa-se adequar a base de dados para a implementação correta das técnicas de mineração de dados. As metologias utilizadas para a transformação dos dados, dependem das técnicas de mineração, deste modo, estas duas etapas serão apresentadas de forma conjunta. Feng and Zhu (2016) definem quatro tarefas principais em *trajectory data mining*: agrupamento, classificação, mineração de padrões e descoberta de conhecimento. Alguns dos trabalhos mais representativos na mineração de dados de trajetórias são:

Agarwal et al. (2018) apresentam *subtrajectory clustering*, um modelo de agrupamento de subsequências de trajetórias. O objetivo é capturar as partes ou segmentos de trajetórias compartilhadas em uma base de dados, assumindo que cada trajetória é uma concatenação de um pequeno conjunto de caminhos, com possíveis intervalos entre eles. Agarwal *et al.* utiliza a distância discreta Fréchet (Aronov et al., 2006b) como medida de similaridade entre duas trajetórias. Os experimentos foram realizados em bases de dados reais e sintéticas, avaliando a robustez dos algoritmos em regiões muito densas e com presença de variações, demostrando que o algoritmo pode ser implementado em ITS (Sistemas de Transporte Inteligentes) pelo fato de poder ser implementado em forma paralela.

da Silva et al. (2016a) propôs um método de agrupamento *online* para trajetórias. Este método visa buscar grupos de trajetórias, chamados *micro-cluster*, que representam a relação entre o objeto e o movimento. Para isso, a autora estipula que o objeto pode aparecer, desaparecer, manter ou atualizar suas trajetórias em uma janela de tempo.

Sun and Ban (2018) propõem um modelo de classificação de trajetórias baseado em uma SVM multiclasse (*Support Vector Machine*). O objetivo do autor é a identificação do tipo de veículo usando dados do sistema de posicionamento global (carro, caminhão pequeno e grande). A acurácia de classificação obtida nesse trabalho é relativamente baixa (75%), possuindo como maior desafio o problema de desbalanceamento de dados.

Qin et al. (2018) desenvolveram um modelo de mineração espaço-temporal para descobrir padrões de rotina em pessoas utilizando trajetórias criadas por *smarthphones*. Estes padrões tem dois aspectos de informação: Como são os padrões típicos de deslocamento das pessoas? Quanto os seus comportamentos variam de dia para dia? Neste trabalho foram utilizadas trajetórias ativas, o objetivo principal foi criar um modelo de predição da probabilidade de uma pessoa estar em um POI, em função do horário e do dia da semana.

da Silva et al. (2016b) apresentaram um framework para descobrir padrões de mobilidade e adaptação em um fluxo de dados de subtrajetórias. Um algoritmo incremental é proposto para capturar a evolução de *micro-cluster*. Eles definem um *micro-cluster* como uma estrutura que representa a relação entre objetos em movimento.

Li et al. (2015a) propuseram um método para a criação de um mapa digital baseado na informação gerada pelo usuário. O método utiliza técnicas de mineração de dados e ferramentas de processamento de linguagem natural em dados de trajetórias e dados *geotagged*[1] de meios sociais para criar um mapa digital.

Lv et al. (2016) desenvolveram um *hierarchical clustering algorithm* para resolver o problema de reconhecimento de POI utilizando características espaço-temporais e sequências dos POI's visitados nas trajetórias.

Boukhechba et al. (2015) propuseram um algoritmo para o reconhecimento *online* de atividades realizadas por pessoas a partir de dados de GPS. O autor estabelece três tipos de padrões presentes em uma trajetória: *stop concept* que representa características estacionárias, o *Moving Activity* ou conjunto de POI em uma trajetória que poderiam representar uma atividade e o *Moves* que representa o movimento entre os POI. Estes tipos de padrões são usados pelo algoritmo para inferir, de forma incremental, a probabilidade de uma pessoa estar fazendo algo interessante e assim atribuir uma atividade especifica, por exemplo, turismo para uma pessoa que visita museus e praças ou esporte para uma pessoa que vai a academia e centros esportivos.

Giannotti et al. (2011) realizam um estudo da complexidade da mobilidade humana consultando e explorando dados de trajetórias. O autor apresenta um sistema de mineração e consulta chamado M-atlas, o qual tenta responder perguntas como: quais são os padrões frequentes de viagens das pessoas? Como grandes eventos influenciam a mobilidade? Como prever áreas de tráfego denso no futuro próximo? Como caracterizar engarrafamentos? Giannotti *et al.* expõem as metodologias de transformação e mineração de dados, além da avaliação e exploração dos resultados, utilizados no sistema.

[1]POI's rotulados com localização, por exemplo: endereço de restaurante, praças ou centros comerciais.

2.2.4 Interpretação e Avaliação

Nesta etapa, o conhecimento extraído a partir das trajetórias já deve que estar disponível para que o especialista possa aplicá-lo em uma tarefa específica. A extração de conhecimento em trajetórias tem ajudado o desenvolvimento de aplicações, tais como: predição de condições de tráfego (Katrakazas et al., 2016); planejamento de rotas (Spichkova et al., 2015); estimação do tempo de viagens baseado no histórico de trajetórias (Sanaullah et al., 2016); gerenciamento de frotas (Thong et al., 2007); recomendação de caronas (Cruz et al., 2015); recomendação de rotas para motoristas de táxis (Chen et al., 2017b; Liu et al., 2017a); sistemas para o compartilhamento de trajetórias em taxis (Ma et al., 2015); distribuição de pontos controle para melhorar o serviço e a cobertura de ambulâncias (Li et al., 2015b); sistemas de recomendação para carga de veículos elétricos (Liu et al., 2017a); mecanismos e aplicativos dedicados a descobrir *bugs* e erros nos mapas digitais de OSM (OpenStreetMap) (Basiri et al., 2016).

Wang et al. (2018) desenvolveram um sistema de recomendação de zonas de procura de passageiros para taxista. O sistema inicialmente implementa um algoritmo de agrupamento no mapa digital, em seguida os grupos são classificados utilizando um modelo *Extreme Learning Machine* para avaliar a potencial procura de passageiros nas regiões.

Sun and Ban (2018) propuseram um modelo de *machine learning* para identificar múltiplas classes de veículos em dados de GPS, incluindo carros de passageiros, caminhões e caminhões multi-reboque. A acurácia apresentada pelo classificador foi do 75%, pelo fato de caminhões de um reboque e multi reboque terem padrões de mobilidade semelhantes. Este trabalho apresenta um desafio em bases de dados desbalanceadas, pois a quantidade de dados de carros de passageiros é maior que as dos caminhões.

Chen et al. (2017a) propuseram um sistema econômico de entrega de encomendas aproveitando os percursos realizados pelos taxistas, sem degradar a qualidade das viagem para os passageiros. Primeiramente, são identificados os caminhos mais curtos da entrega baseado em dados históricos de trajetórias, nesse passo não é considerado o tempo da viagem nem a relação origem-destino. Em seguida, utiliza-se um algoritmo adaptativo *on-line*, usando os caminhos e o tempo de viagem como referência, para encontrar de forma iterativa os caminhos de entrega em tempo real e, dessa forma, direcionar a roteirização para a entrega dos pacotes.

Liu and Wang (2017) apresentaram um sistema para detectar comunidades em uma região baseado em trajetórias. O sistema combina informação das bases de dados de trajetórias com matrizes de similaridade dos usuários construídas a partir de informação de múltiplas fontes. O objetivo é identificar um conjunto de objetos a partir do comportamento individual e coletivo de motoristas. Uma comunidade é um conjunto de objetos cuja proximidade ou similaridade de movimento é provavelmente uma manifestação de alguma interação mútua subjacente ou relacionamento compartilhado.

Bao et al. (2017) propuseram uma abordagem para desenvolver planos de construção de ciclovias com base em dados de trajetórias de bicicleta do mundo real. Utilizam uma função

objetivo flexível para ajustar o benefício entre a cobertura do número de usuários e a duração de suas trajetórias.

Maruseac and Ghinita (2016) focaram na preservação da privacidade dos usuários das trajetórias. É proposto um método que preserva a privacidade para a representação de padrões de viagens. A solução proposta consiste em um algoritmo de amostragem baseado no mecanismo exponencial de privacidade diferencial (McSherry and Talwar, 2007), a qual usa informação do mapa digital para aumentar sua acurácia.

2.3 Logística de *e-commerce* no Brasil

Nos últimos anos, o mercado de *e-commerce* tem aumentado devido ao ambiente favorável resultante do aumento ao acesso de Internet e facilidade de pagamento *online*. No entanto, também trouxe desafios significativos para a logística de entrega, principalmente a demanda por entregas no mesmo dia de baixo custo. Em países como Brasil, os desafios aumentam devido às características geográficas do país, sua grande extensão territorial e distribuição socioeconômica do país. As metodologias propostas nesse trabalho visam resolver problemas presentes na logística de *e-commerce* no Brasil, portanto, nessa seção é apresentado o modelo de entrega de uma empresa brasileira e, seguidamente, expõem-se os problemas abordados ao longo do trabalho.

Loggi é uma empresa brasileira de logística de entregas expressas baseada em um modelo de economia compartilhada como Uber, Rappi ou Glovo. Mensageiros independentes usam a plataforma online para aceitar e fazer entregas de *last-mile* com o seus próprios veículos. O principal desafio da Loggi é criar uma rede logística sustentável para entregas do mesmo dia em todo o país.

O modelo de entregas *e-commerce* usado pela Loggi é apresentado na Figura 2.3, o qual esta dividido em três etapas: *first-mile*, *middle-mile* e *last-mile*. No *first*, os pacotes são coletados nos centros de distribuição das companhias *e-commerce* e são enviados para o *Distribution Center* (DC). No DC da Loggi, os pacotes são agrupados em *unit loads* baseados em critérios distintos, como prioridade, mercadorias perigosas, e *Expedition Center* (EC) destino. No *middle-mile*, pacotes são transferidos do DC para ECs. Esta transferência é realizada por caminhões ou aviões, dependendo da localização do EC alvo. Nesta etapa são tomadas as decisões de corte ou despacho, processo conhecido na literatura como ondas entre DCs e ECs (Klapp et al., 2018). Por fim, no *last-mile*, os pacotes agrupados em cargas unitárias correspondentes aos roteiros de entrega são recolhidos pelos motoristas autônomos nos ECs, em carros, vans ou motocicletas, e entregues aos consumidores finais.

Com o objetivo de reduzir o tempo e complexidade do processamento dos pacotes nos ECs, Loggi realiza a roteirização de pacotes no DC. Pacotes são agrupados em *unit loads* correspondentes às rotas de entrega, ou last-mile, para cada uma das cidades servidas pela

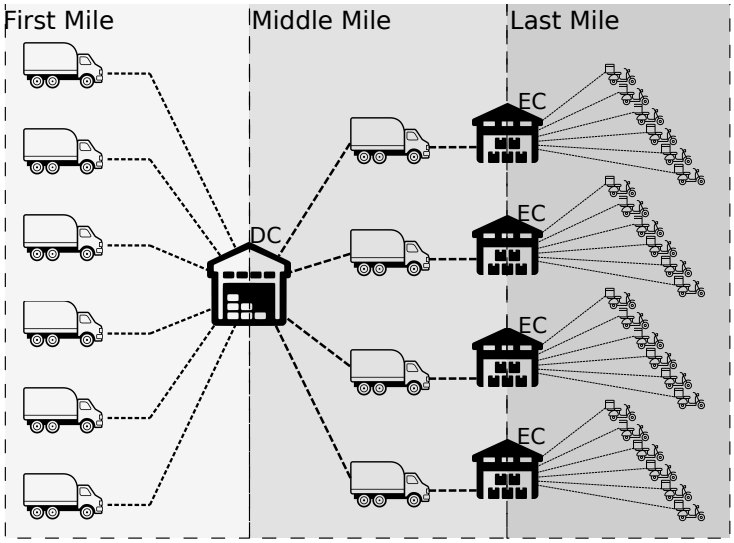


Figura 2.3: Modelo de entrega usado na Loggi.

Loggi no DC, em seguida são transferidos para o os respetivos ECs. Este problema de criação de rotas de *last-mile* é modelado como *Vehicle Routing Problem* (VRP) (Braekers et al., 2016b; Laporte, 1992). Alguns dos desafios presentes na etapa de *Last-Mile* do modelo logístico de *e-commerce* aplicado na Loggi são:

- **Roteirização de Veículos no *Last-Mile***:

  o modelo de logística de *e-commerce* requer que as entregas sejam de baixo custo e no menor tempo possível, isto significa que etapas nas quais o pacote não realiza movimentação na malha de processamento precisam ser removidas. Uma destas etapas é o armazenamento necessário para a implementação do VRP. Isso se deve ao fato de que o VRP resolve um problema estático e precisa de toda a informação para otimizar as rotas percorridas pelos mensageiros. Ademais, VRP é um problema NP-Hard, o que torna ele inviável para grandes volumes de pacotes. Portanto, para a separação eficiente dos pacotes, exige-se a implementação de um modelo dinâmico. Além disso, empresas de logística usam *hardware* especializado nas etapas de separação que requerem um tempo de processamento limitado, portanto a complexidade dos modelos implementados precisa ser baixa.

- **Gerenciamento de rotas de *Last-Mile***:

  o modelo de entrega usado na Loggi, separa o processo de roteirização realizado nos DCs do processo de expedição das rotas feitos nos ECs. Além disso, a Loggi tem um modelo de economia compartilhada, o que significa que as rotas são expostas em um sistema

computacional e aceitas pelo mensageiro. Portanto, é necessário realizar o casamento apropriado entre mensageiro e rota, tentando maximizar a probabilidade da entrega dos pacotes. Este problema pode-se ser modelado como um problema de classificação de trajetórias, rotas de *last-mile*, entre mensageiros com maior afinidade.

- **Detecção de rotas anômalas de *Last-Mile*** :

  o fato da etapa de roteirização ser realizado em instalações diferentes às de expedição, e o alto volume de pacotes processados nos CDs, favorece erros operacionais que representam o envio errado de rotas nos CEs ou a inserção de pacotes errados entre as rotas. Estes erros aumentam o custo de entrega dos pacotes e podem significar uma perda no prazo de entrega dos pacotes. Portanto, ter uma etapa de detecção de anomalia nas rotas criadas possibilita a identificação desses erros e minimiza o mau desempenho na logística de entrega.

As metodologias propostas nesse trabalho visam resolver estes três desafios. A **Roteirização de Veículos no *Last-Mile*** é abordado no Capitulo 3 modelado como um sistema multiagente que usa técnicas de mineração de dados em trajetórias para extrair padrões territoriais e usá-los na criação dinâmica de rotas de *last-mile*. O **Gerenciamento de rotas de *Last-Mile*** e a **Detecção de rotas de *Last-Mile* anômalas** são abordadas com a implementação de um algoritmo de agrupamento incremental apresentado no Capítulo 4.

O problema de roteirização de Veículos no *Last-Mile* Particularmente, consiste em um VRP *Fully Dynamic* (DRVP) com consumidores estocásticos aplicados a problemas de *Big Data*. Por este motivo, nas seguentes sub-seções é apresentada uma revisão da literatura centrada nos DVRP com clientes estocásticos, técnicas de mineração de dados de trajetórias aplicados em problemas com grandes volumes de dados e a contextualização do problema de roteirização de last-mile da Loggi. Finalmente é apresentada a revisão da literatura no problema de gerenciamento e detecção de *outliers* no *Last-Mile*.

2.3.1 DVRP com Clientes Estocásticos

O objetivo do VRP é minimizar a distância percorrida pelos veículos para entrega/coleta de produtos e é amplamente adotado em problemas de transporte. No entanto, as empresas de transporte têm necessidades específicas, como prioridades de entrega, janelas de tempo, disponibilidade de recursos, incerteza na demanda, tempo ou clientes. Esses motivos têm representado a necessidade de variações deste problema na literatura (Braekers et al., 2016a). Algumas variações tradicionais são o VRP capacitado (CVRP) (Laporte, 2009), o VRP com janelas de tempo ou *time windows* (VRPTW) (Dixit et al., 2019), o VRP dinâmico (Pillac et al., 2013), o problema de roteamento com *cross-docking* (Maknoon and Laporte, 2017), entre outros.

Na formulação de VRP dinâmica (DVRP), também conhecida como VRP em tempo real ou online, os dados de entrada não estão totalmente disponíveis no início do problema. Ao contrário, são revelados ao longo do tempo. As soluções para este problema na literatura são classificadas de acordo com o tipo de informação dinâmica, por exemplo, novas solicitações de clientes, demandas, tempos de serviço e tempos de viagem. No entanto, em aplicações como o abastecimento de supermercados ou estações de serviço, existe um componente estocástico para o problema conhecido de antemão. O problema apresentado neste trabalho é definido como um VRP Estocástico e Dinâmico (SDVRP) ou, mais especificamente, um DVRP com Clientes Estocásticos ou DVRP *with Stochastic Customers*. As informações sobre os pacotes só são conhecidas quando chegam ao CD. No entanto, podemos usar dados históricos sobre os pacotes entregues como um proxy estocástico para os que chegam.

O SDVRP é dividido em duas etapas. A primeira é a fase de planejamento, na qual as decisões pré-processadas são construídas com base em dados históricos. A seguir está a fase de execução que usa as decisões pré-processadas e novos eventos que ocorrem para criar descrições *online* das rotas finais (Bernardo and Pannek, 2018; Ritzinger et al., 2016). Existem diferentes metodologias para essas etapas: alguns métodos comuns para a etapa de planejamento são a construção de um *pool* com todas as soluções possíveis e o uso de algoritmos de seleção de rotas na etapa de execução. Bent and Van Hentenryck (2004) propuseram uma solução para o VRP dinâmico com janelas de tempo e clientes estocásticos, onde o objetivo é maximizar o número de clientes atendidos. A fase de planejamento usa uma abordagem de múltiplos cenários (MSA) que gera e resolve cenários que incluem solicitações estáticas e dinâmicas. Na fase de execução, a função de consenso seleciona o plano mais semelhante do *pool* atual. Barbucha and Jędrzejowicz (2009) baseou sua proposta em um sistema MA. No estágio de planejamento, seu método usou *Gillett and Miller's sweep algorithm* para criar as rotas iniciais; cada rota é um agente que representa um veículo. Na fase de execução, as solicitações são avaliadas pelos agentes que estão em um ambiente que muda dinamicamente. A principal contribuição é a viabilidade da utilização de um sistema MA em um VRP, conseguindo tirar proveito de diversos recursos, como a autonomia dos agentes, a capacidade de aumentar a eficiência computacional através da paralelização e a possibilidade de utilização de um ambiente distribuído.

Outras metodologias foram propostas para resolver o SDVRP. Joe and Lau (2020) modelou o problema de roteamento de veículos com janelas de tempo e clientes parcialmente estocásticos como um Processo de Decisão Markov baseado em rota e propôs uma abordagem que combina *Deep Reinforcement Learning*, incluindo *neural networks-based Temporal-Difference learning* com repetição de experiência para aproximar o *value function*, utilizando *Simulated Annealing*. Huang et al. (2018) propôs um algoritmo baseado em áreas urbanas de alta densidade populacional. Por meio de um *two-echelon logistics system*, eles introduziram um algoritmo de otimização de fluxo em quarteirões pré-estabelecidos. O trabalho é baseado em sistemas de entrega urbana do mundo real na China, com foco no crescimento do volume de entrega e lidar com as variações do volume de entrega do dia a dia. Ulmer et al. (2019a) propôs uma heurística

de programação dinâmica aproximada offline-online para gerar políticas de roteamento dinâmico para o problema de roteamento de veículo único com solicitações de serviço estocásticas (VRPSSR). Eles incorporam o *value function approximation* (VFA) *offline* em um algoritmo de distribuição online, fornecendo uma condição suficiente para garantir que uma política de distribuição baseada em VFA tenha um desempenho pelo menos tão bom quanto a política VFA sozinha. Tal como o nosso problema, este problema contém uma componente estocástica espacial e temporal, e as suas análises foram baseadas na avaliação dos clientes com pedidos atrasados e na melhoria das políticas propostas na *Over Myopic Policy*. Infelizmente, eles não discutiram a qualidade, o design e o custo total das rotas geradas.

Um número considerável de autores discutindo o VRP com consumidores estocásticos também inclui algumas condições especiais, tornando seus problemas mais adequados para problemas de entrega no mesmo dia. Por exemplo, Ulmer et al. (2019b) propôs roteamento de entrega no mesmo dia; no entanto, os veículos podem integrar solicitações dinâmicas em rotas de entrega, aproveitando a devolução de depósitos preventivos. Em outras palavras, um veículo após o despacho pode retornar ao depósito para carregar novos pedidos que ocorram dentro de algum horizonte de tempo. Eles se propõem a usar informações estocásticas para projetar novas solicitações e, assim, minimizar o custo de entrega por meio de um modelo de processo de decisão de Markov. Ulmer et al. (2018) enviou um veículo com rota de serviço para coleta de pacotes na área de serviço. As solicitações de coleta ocorrem dinamicamente durante o dia e são desconhecidas antes de sua solicitação real; portanto, seu objetivo é atingir o maior número de solicitações confirmadas usando heurísticas de orçamento de tempo antecipado usando métodos de programação dinâmica aproximada.

2.3.2 Mineração de dados de trajetórias

Rotas de *last-mile* podem ser classificadas como trajetórias passivas, uma vez que um sistema de posicionamento global em smartphones do motorista de *last-mile* as gera. Porém, se considerarmos a saída do VRP como uma sequência de pontos, as localizações de cada pacote a ser entregue, podemos classificá-los como rotas ativas, e cada ponto será definido como um POI. Estudos desenvolvidos em Lv et al. (2019); Vahedian Khezerlou et al. (2019); Li et al. (2018); Fu et al. (2017); Chen et al. (2011); Cao et al. (2005); Mamoulis et al. (2004) dividem a área alvo como um conjunto de células que representam regiões dentro de um mapa. Assim, uma rota pode ser representada por sequências de POI, células ou áreas. No problema abordado neste trabalho, a região onde os pacotes podem ser entregues é normalmente conhecida. Portanto, todas as células possíveis nas quais os pacotes podem ser entregues também já são conhecidas, de forma que técnicas de mineração de conjuntos de itens frequentes podem ser aplicadas considerando cada item como um POI ou célula visitada.

Lee et al. (2009) propôs um algoritmo de mineração baseado em grafos, nesse trabalho as trajetórias foram mapeadas em um grafo e técnicas de mineração de dados foram usadas para extrair padrões de trajetória frequentes. Eles aproveitam as adjacências da representação em grafo para reduzir o espaço; seus resultados mostraram resultados melhores do que métodos baseados em Apriori e baseados em PrefixSpan. Essa propriedade não pode ser usada em nosso problema porque os pacotes em uma rota podem fazer parte de regiões não adjacentes. Fu et al. (2017) propôs uma técnica para mineração de padrões de rotas frequentes. Em primeiro lugar, partição de trajetória, extração de localização, simplificação de dados e descoberta de segmento comum são usados para resumir dados de trajetória, converter essas trajetórias em sequências temporais de segmento comum (STS) e gerar conjuntos de *1-frequent itemsets*. Em seguida, um algoritmo de mineração de padrões baseado em relacionamento de adjacência espaço-temporal é proposto. Os autores também mostraram que algoritmos de mineração de padrão de sequência para dados de transação, por exemplo, mineração de regras de associação (Apriori, FP-*Tree*) e mineração de padrão de sequência (GSP, PrefixSpan), não podem ser aplicados diretamente aos dados de trajetória resumidos. Esses algoritmos possuem alta complexidade de tempo e não consideram a contiguidade espacial e a continuidade temporal entre os itens. Porém, para a implementação proposta de nosso trabalho, não é necessário manter a continuidade temporal entre a entrega dos pacotes na construção dinâmica das rotas, visto que os pacotes que chegam ao centro de distribuição não possuem um padrão temporal. Portanto, essas técnicas podem ser utilizadas.

Porém, Apriori é um algoritmo que usa *breadth-first search*, o que faz com que o tempo de execução aumente exponencialmente com o aumento do número de conjuntos de itens. Além disso, Apriori verifica todas as transações para cada iteração; portanto, devido ao consumo de memória, sua aplicação torna-se inviável para grandes bancos de dados, como no nosso problema. Han et al. (2000) propôs um algoritmo que usa uma estrutura de árvore de padrões frequente (FP-*Tree*) para realizar uma pesquisa em profundidade e, assim, extrair o conjunto completo de padrões frequentes por crescimento de fragmento de padrão. O principal ganho dessa proposta é que o tempo de execução aumenta linearmente com o número de conjuntos de itens. Li et al. (2008) propôs uma versão distribuída do algoritmo FP-*growth* (PFP). O PFP cria partições computacionais para executar grupos de tarefas de mineração em máquinas independentes, eliminando dependências computacionais e comunicação. Este algoritmo foi selecionado para mineração de dados de trajetória em nosso trabalho.

2.3.3 Roteirização de Veículos no *Last-Mile*

O problema de roteirização de Veículos no *Last-Mile* consiste na criação de rotas para entregar um conjunto de pacotes é conhecido como Problema de Roteirização de Veículos (ou VRP do inglês *Vehicle Routing Problem*) (Braekers et al., 2016b; Laporte, 1992). VRP é

usualmente resolvido por um algoritmo de otimização estática, o que significa que toda a informação dos pacotes para ser entregues (localização de destino, volume, peso entre outros) são necessários antes de começar a execução. Na prática, isto significa que para usar um algoritmo de otimização estática, é necessário armazenar todos os pacotes que serão roteirizados, executar o algoritmo de otimização, e dividí-los nas rotas de last-mile. Porém, dado que o número de pacotes processados no DC por dia pode ser centenas de milhares e o objetivo da companhia é entregar no mesmo dia, armazenar todos estes no DC é inviável. Isso pode requerer um grande espaço de armazenamento, e incrementar o tempo e complexidade do processamento de pacotes.

Para resolver esses problemas, a Loggi implementa um algoritmo VRP dinâmico (Ritzinger et al., 2016; Larsen et al., 2002), isto é, um algoritmo de otimização incremental que pode processar pacotes um por vez, sem requerer armazenamento.

A solução de roteamento usada pela Loggi é formalmente descrita considerando as seguintes restrições:

- Os veículos usados para *last-mile* tem limite de capacidade máxima homogênea, considerando peso, volume e quantidade de pacotes. Este problema é conhecido como *Capacitated* VRP (CVRP) (Laporte, 2009).

- A companhia contrata mensageiros *last-mile* independentes, por essa motivo, as rotas criadas não precisam retornar para a origem. Isso é conhecido como *Open* VRP (OVRP) (Cao et al., 2014; Li et al., 2007; Schrage, 1981).

- Rotas de *last-mile* são criadas no DC e expedidas em vários ECs. Isso é conhecido como *Multi-depot* VRP (VRPMD) (Nucamendi-Guillén et al., 2021; de Oliveira et al., 2016; Baldacci et al., 2013) ou *Two-Echelon vehicle routing problem* (2E-VRP) (Vidović et al., 2016; Hemmelmayr et al., 2012; Crainic et al., 2009).

- Pacotes não podem ser armazenados no DC, devido que representa tempo que o pacote não é movimentado, além de aumentar o custo já que é necessário ter espaço disponível para a armazenagem. Em outras palavras, o algoritmo é um *Online* VRP (Jaillet and Wagner, 2008). Rotas são criadas incrementalmente quando os pacotes chegam no DC. Isso pode ser definido como *dynamic* VRP (DVRP) (Psaraftis et al., 2016; Pillac et al., 2013; Psaraftis, 1988).

- Pacotes não são conhecidos até chegar no DC. Porém, baseado em dados históricos, a distribuição de probabilidade dos pacotes por localização pode ser calculada e usada para a construção de melhores rotas. Isso representa informação estocástica dos consumidores ou VRP *Stochastic Consumers* (Huang et al., 2018; Bent and Van Hentenryck, 2004; Van Hemert and La Poutré, 2004; Van Hentenryck et al., 2010).

- Características particulares do Brasil, tais como população, extensão territorial e a quantidade de cidades a serem atendidas (em torno do 5700), requerem que seja modelado como um problema de *Big Data*.

Algumas soluções propostas para VRP com as características mencionadas: Ausiello et al. (2001) exibiram o uso de um algoritmo *online* TSP (*Traveling Salesman Problem*) (Gutin and Punnen, 2007) como possível solução para coleta e entrega no mesmo dia *same-day pickup and deliveries*. Nessa implementação, rotas são criadas incrementalmente usando modelos TSP probabilísticos, que aproveitam as informações históricas das entregas já realizadas. A solução considera restrições tais como veículo capacitivo e rotas abertas. Não obstante, essa solução não é aplicável diretamente no problema da Loggi visto que as otimizações são somente para uma rota, em outros termos ela resolve um TSP e não um VRP.

Zhong et al. (2007) apresentaram um modelo para entrega de produtos *e-commerce*. O principal objetivo é construir um modelo realista para melhorar o planejamento territorial e o processo de despacho de veículos. Ao considerar as localizações e demandas aleatórias dos clientes e manter a familiaridade do motorista com suas áreas de serviço, tal solução oferece rotas flexíveis com motoristas mais familiares, resultando em melhor serviço com menor custo.

Huang et al. (2018) expuseram uma proposta para um *two-echelon logistics system* para entregas de *e-commerce*. O primeiro passo nesta implementação foi selecionar uma boa localização para entregar em uma área urbana ou centro de distribuição em uma cidade. Para isso, pacotes são transportados para centros satélites, nos quais são despachados para serem entregues no seu destino final. Os autores apresentaram uma estratégia de limitação de entregas usando células, e considera duas formulações para as restrições de conectividade, ambas inspiradas em conceitos de fluxo de commodities múltiplas. Estes modelos podem ser uma solução viável para o problema exposto neste trabalho, no entanto, esses modelos são baseados na entrega para clientes em áreas urbanas densamente povoadas, característica não presente em todas as cidades do Brasil.

A arquitetura Multiagente (MA) para *Dynamic* VRP (Thangiah et al., 2001; Barbucha and Jędrzejowicz, 2009) tem mostrado bons resultados graças a diversos recursos tipicamente observados em sistemas com múltiplos agentes, como a autonomia dos agentes, a capacidade de aumentar a eficiência computacional por meio da paralelização e a possibilidade de uso de um ambiente distribuído. Neste caso, cada agente representa um *unit load* e resolvem problemas como capacidade e fechamento dos *unit loads* de forma independente. No entanto, para um alto dinamismo, o algoritmo proposto por Barbucha and Jędrzejowicz (2009) não é escalável. Além disso, os testes foram realizados para bancos de dados com no máximo 200 consumidores, impedindo sua implementação para o problema exposto neste trabalho, que pode ter mais de centenas de milhares de consumidores diariamente.

A maioria dos trabalhos encontrados na literatura são testados em pequenas bases de dados (Bujel et al., 2018). Para lidar com grandes volumes de dados, ou *Big Data*, alguns artigos

usaram técnicas de *Big Data* no VRP (Bertsimas et al., 2019; Wang et al., 2016; Kytöjoki et al., 2007). Ao trabalhar com bancos de dados grandes e reais, uma das técnicas utilizadas para diminuir o custo computacional é dividir a área de entrega em regiões fixas, conhecido como *Territory-Based* VRP (Zhong et al., 2007). No *Territory-Based* VRP, uma rota é uma sequência de regiões visitadas pelo veículo, que pode ser representada pela união de dados espaço-temporais (Lv et al., 2019; Fu et al., 2017). Vários trabalhos focam no reconhecimento de padrões em dados espaço-temporais aplicados como um diferencial para diferentes aplicações, como previsão do tempo de viagem (Nakata and Takeuchi, 2004), previsão de eventos (Vahedian Khezerlou et al., 2019), detecção de anomalias na geolocalização (Shih et al., 2016), sistemas de previsão de rota (Chen et al., 2011) e previsão de localização (Wu et al., 2018). A mineração de dados de trajetória (Zheng, 2015) é uma área emergente que pode trazer aplicações e melhorias para algoritmos de VRP.

2.3.4 Gerenciamento e Detecção de *Outliers* no *Last-Mile*

O gerenciamento de rotas de *Last-Mile* consiste em distribuir as rotas já construídas entre os mensageiros, maximizando a probabilidade do pacote ser entregue. Huang et al. (2018) demostraram que esta probabilidade cresce quando existe uma zonalidade nas entregas realizadas pelos mensageiros, já que eles conhecem as características geográficas da região e aprendem o comportamento da população. Portanto, o desafio é construir grupos de trajetórias que consigam extrair informação territorial, e assim, serem usados no EC para fazer um casamento das rotas criadas no DC com os entregadores. Contudo, o agrupamento de trajetórias torna-se mais complexo devido à natureza dos dados. Estes são normalmente coletados por diferentes sensores o que geram divergência na frequência da amostragem, isto faz com que a mesma trajetória possa ser representadas de múltiplas formas, por este motivo, muitos autores concentram os estudos em propor métricas de similaridade entre as trajetórias que sejam robustas para diferentes representações, por exemplo, a distância Hausdorff (Sim et al., 1999; Huttenlocher et al., 1993), distância LCSS (Robinson, 1990), distância DTW Kruskal (1983), distância Fréchet (Agarwal et al., 2018; Har-Peled and Raichel, 2014; Aronov et al., 2006a) distância baseada no *road network* (da Silva et al., 2020), métrica de similaridade proposta para *Model-Driven Matching* (Sankararaman et al., 2013), entre outros. Além disso, dado que as trajetórias representarem o deslocamento de um objeto, podem existir similaridade em apenas alguns trechos, para resolver este problema alguns autores, como Agarwal et al. (2018); Shein and Puntheeranurak (2018), realizam o agrupamento dividindo a trajetórias em sub-trajetórias e procurando padrões nestas.

O segundo problema consiste na detecção de rotas anômalas ou *outliers*, uma vez que no processo de separação de pacotes podem acontecer erros operacionais que reflitam no envio de pacotes para CEs errados ou na criação de rotas fora do padrão causada pela mistura de pacotes

entre rotas. A detecção oportuna desses erros acarreta em uma diminuição de custo na entrega dos pacotes, pois estes erros são identificados antes dos pacotes serem enviados para os CEs ou serem expedidos. Por outro lado, a distribuição de pacotes pode mudar ao longo do tempo, já que novos mercados podem emergir ou a expansão da empresa pode mudar a abrangência dos CEs, portanto, é necessária a implementação de um algoritmo incremental capaz de adaptar-se à estas mudanças. Porém, como foi discutido em Liu et al. (2021), existe uma diferença entre a tarefa de agrupamento e a tarefa de detecção de *outliers*. A detecção de *outliers* pode ser definida como uma etapa na análise de agrupamento, mas ao incluir dados anômalos como entradas de um algoritmo de agrupamento, estes podem danificar as estruturas que definem os grupos. Por exemplo, poucos *outliers* destroem facilmente a estrutura dos clusters derivadas do algoritmo K-means (MacQueen et al., 1967) e podem gerar distribuições incoerentes a partir do modelo de mistura de Gaussianas. Visto que, trajetórias contendo *outliers* podem ser construídas devido a erros operacionais, é necessário implementar um algoritmo de agrupamento que seja robusto a *outliers*.

No cenário atual, a quantidade crescente de fluxo de dados ou *data streams* exige o desenvolvimento de algoritmos de agrupamento com a capacidade de descobrir novos padrões no ambiente *online*. *Data streams* geralmente vêm de ambientes não estacionários, sem informações anteriores sobre a distribuição de dados ou sem conhecimento da quantidade de grupos. Além disso, a distribuição de dados, bem como o número de grupos, podem mudar com o tempo. Assim, em contraste com os métodos em lote ou aplicados em janelas de tempo, algoritmos para agrupar *data streams online* processam um elemento por vez, atualizam a estrutura dos grupos sempre que necessário e armazenam na memória um pequeno número de elementos ou protótipos como uma representação dos clusters. No dias atuais, o agrupamento *online* tornou-se uma ferramenta importante para muitas aplicações, como descoberta de conhecimento (Gama, 2010), detecção de falhas de processo (Lemos et al., 2011; Costa et al., 2015), sistemas de recomendação e detecção de anomalias ou *outliers* (Angelov, 2014b). A maior parte dos algoritmos de *data streams* aplicados a dados de trajetória que tem sido propostos na literatura centralizam os estudos na análise em tempo real de dados de mobilidade que permitam detecção *online* de regularidade ou anomalias que podem ser usadas em aplicações como sistemas de monitoramento que melhorem as condições de tráfego, predição do clima ou em sistemas de vigilância nas cidades, entre outros (da Silva et al., 2020; Shein and Puntheeranurak, 2018; Lan et al., 2017; da Silva et al., 2016a; Laxhammar and Falkman, 2014).

Várias abordagens têm sido propostas para agrupar *data streams* Silva et al. (2013). A versão incremental do *k-means* (Pham et al., 2004) é um algoritmo simples, mas eficiente, baseado na distância. Embora *k-means* incrementais apresentem baixo custo computacional, ele requer algum conhecimento prévio sobre o problema, por exemplo, o número de grupos k que devem ser conhecidos antecipadamente. Além disso, os métodos baseados em distância geralmente assumem que os grupos vêm da mesma distribuição (ou forma) conforme a métrica de distância usada.

Outros algoritmos incrementais mais avançados, primeiramente dividem *data streams* em *micro-grupos* ou *micro-clusters* e, em seguida, encontram os grupos finais com base em *micro-clusters* que compartilham propriedades comuns. Esta abordagem tem se mostrado eficaz para agrupamento incremental de *data streams* (Ackerman and Dasgupta, 2014) e está sendo amplamente utilizada neste contexto. O algoritmo Clustream (Aggarwal et al., 2003) segue esta abordagem usando amostras dentro de uma janela de tempo para criar e atualizar *micro-clusters* incrementalmente, no entanto, como *k-means* incrementais, Clustream tem problemas ao lidar com clusters não esféricos. Algoritmos baseados em densidade como DBstream (Hahsler and Bolaños, 2016), Denstream (Cao et al., 2006) podem encontrar automaticamente o número real de grupos em cada etapa e criar grupos de formas arbitrárias, mas o grande número de parâmetros livres torna a busca pelo melhor modelo muito difícil para diferentes aplicações (Silva et al., 2013). Estes modelos de agrupamento foram implementados e testados em dados atemporais e d-dimensionais. Para que eles possam ser aplicados ao problema de trajetórias exposto nesse trabalho é necessária uma etapa de preprocessamento de dados e/ou uma adaptação para o uso de uma métrica de similaridades em trajetórias.

O MicroTEDAclus (Maia et al., 2020) é um algoritmo de agrupamento evolutivo, proposto para abordar essas lacunas. Semelhante a CEDAS (Hyde et al., 2017), este algoritmo divide o problema em *micro-clusters* e *macro-clusters*. Os *micro-clusters* são baseados no conceito de Análise de Dados de Tipicidade e Excentricidade (TEDA) (Angelov, 2014a). Em primeiro lugar, o algoritmo atualiza *micro-clusters* que têm sua variância limitada por um limite superior que cresce dinamicamente (σ_0) à medida que mais dados são usados no cálculo TEDA. Em seguida, a estrutura do *macro-cluster* é atualizada. Cada *macro-cluster* é composto por um conjunto de *micro-clusters* conectados que se interceptam ou sobrepõem. Em seguida, filtramos a estrutura do *macro-cluster* para ativar apenas seus *micro-clusters* mais densos e estimamos a densidade de um *macro-cluster* por uma soma ponderada das tipicidades de seus *micro-clusters* conectados, como em um modelo de mistura de densidade. Finalmente, o algoritmo atribui um novo ponto de dados ao *macro-cluster* que possui a maior associação conforme a pontuação de mistura de tipicidade. Este método foi comparado aos métodos estado-da-arte sobre diversas bases de dados e obteve resultados similares, porém sem a necessidade um ajuste de parâmetros. O algoritmo de agrupamento evolutivo tem as seguintes características:

- um procedimento de passagem única para atualizar protótipos de *cluster* e estimar suas densidades, uma amostra de cada vez;
- geração de grupos com formas e distribuições arbitrárias (ou formas) que chegam em uma ordem arbitrária;
- o método possui dois parâmetros a ser definido pelo usuário;
- a saída não é apenas a atribuição do cluster, mas também o grau de pertinência (ou de pertencimento) de um novo ponto de dados a todos os clusters existentes.

Capítulo 3

Roteirização de Veículos no *Last-Mile*

A modelagem proposta neste capitulo visa resolver o problema de roteirização de Veículos no *Last-Mile* exposto na Seção 2.3 e na Sub-Seção 2.3.3, aplicado ao problema de roteamento enfrentado por Loggi. Este trabalho propõe um algoritmo VRP baseado em um sistema MA que usa técnicas de mineração de dados de trajetória para diminuir o custo computacional e permitir o uso de informações históricas de rotas passadas para melhorar o desempenho do algoritmo. O modelo MA proposto é um algoritmo de *stream* VRP, possibilitando a geração de rotas sem a necessidade de armazenamento de pacotes para otimização, aumentando a capacidade de processamento por meio da resolução de problemas de *Big Data*. O sistema MA modela cada rota como um agente. Cada rota possui características próprias, como capacidade homogênea e rotas abertas. Além disso, nosso trabalho utiliza uma versão distribuída do algoritmo FP-*growth* (Li et al., 2008), que trata o problema de mineração como *Big Data*, possibilitando o uso de computação distribuída para minerar grandes bancos de dados de forma eficiente.

Os experimentos foram realizados em registros de entrega Loggi. Comparamos nossa proposta com algoritmos de *benchmark* para problemas DCVRP com consumidores estocásticos. Adicionalmente, realizamos uma análise empírica para avaliar a robustez e confiabilidade das soluções alcançadas pelo método proposto. Nesta análise, comparamos nossa metodologia dinâmica com uma heurística de pesquisa local comumente usada na indústria para resolver VRPs estáticos. A heurística estática foi escolhida porque simula as condições ótimas do problema em que todas as informações dos pacotes são conhecidas com antecedência, removendo a informação estocástica do problema. Assim, podemos contrastar as soluções de ambos os métodos (o nosso e a heurística estática) para validar características desejadas importantes de qualquer solução de VRP, como robustez a variações nas entradas e desempenho geral.

O restante do capítulo está organizado da seguinte forma. Descrevemos a formulação do problema na Seção 3.1. Definimos formalmente o algoritmo proposto para resolver o Problema de

Roteirização Dinâmica de Veículos Capacitados com Clientes Estocásticos a partir de técnicas de Mineração de Dados de Trajetória e Sistema MA na Seção 3.2. Finalmente, apresentamos um extenso estudo computacional comparando diferentes estratégias de entrega na 3.3.

3.1 Formulação do Problema

O Problema de Roteamento de Veículos (VRP) vem sendo estudado há mais de 60 anos (Braekers et al., 2016a), tornando-se um dos mais importantes problemas de otimização combinatória. Golden et al. (2008) definiu VRP como: seja um grafo direcionado $G = (V, A)$, onde $V = \{0, 1, ..., n\}$ é o conjunto de $n + 1$ nós e A é o conjunto de arcos. O nó 0 é a origem ou depósito e os outros n são os clientes que requerem unidades de fornecimento q_i do depósito. Uma frota de veículos que fica no depósito é utilizada para abastecer os clientes. A frota de veículos é formada por m diferentes tipos de veículos, com $M = \{1, 2, ..., m\}$, e cada tipo $k \in M$, m_k veículos tendo uma capacidade Q_k e um custo fixo F_k. Além disso, para cada arco $(i, j) \in A$, tem-se um custo de roteamento não negativo $c^k_{i,j}$ para cada tipo de veículo $k \in M$. Portanto, o VRP calcula as viagens ótimas para visitar todos os consumidores que utilizam a frota de veículos, ou seja, minimizando a distância percorrida pelos veículos para visitar todos os consumidores, partindo e terminando nos depósitos. A viagem ou rota é um circuito simples em G definido como $R = (i_1, i_2, ..., i_{|R|})$, com $i_1 = i_{|R|} = 0$ percorrido por um veículo $k \in M$.

O problema abordado neste trabalho é um problema de roteamento de veículos capacitados totalmente dinâmicos com clientes estocásticos; no entanto, é baseado no modelo de logística de *e-commerce* da Loggi. O problema deve atender às características:

- Cliente: é a pessoa que receberá o pacote de *e-commerce*. O ith consumidor corresponde ao nó V_i do gráfico G na definição do VRP. V_i é definido como uma localização geográfica com latitude (lat_i) e longitude (lng_i).
- Pacote (p_i): produto a ser entregue ao cliente i. Cada pacote contém informações sobre a localização V_i do cliente correspondente e características como volume (v_i) e peso (w_i).
- Rota (T_j): seqüência de pacotes a serem entregues aos respectivos clientes. $T_j = \{p_1, p_2, p_3, ..., p_q\}$.
- Veículo: transporte utilizado para realizar uma rota. Cada tipo de veículo k tem algumas restrições de capacidade, como o volume máximo Q_{kv}, o peso máximo Q_{kw} e o número máximo de pacotes que podem ser entregues Q_{kp}.
- *Unit load* (U): um conjunto de itens agrupados em contêineres de transporte que podem ser movidos facilmente com uma caixa, saco, porta-paletes ou caminhão (Laundrie, 1986).

Em nosso caso, uma *unit load* é um conjunto de pacotes atribuídos a uma rota. A rota será realizada por um veículo k, portanto a *unit load* deve respeitar as restrições de capacidade do veículo selecionado (Q_{kv}, Q_{kw} e Q_{kp}).

- *Warehouse system* (S): elemento de hardware ou processo (ou uma combinação de ambos) que permite a triagem e armazenamento intermediário de mercadorias entre duas etapas sucessivas de uma cadeia de abastecimento (Boysen et al., 2019). Em nosso caso, podemos definir um sistema de warehouse como um conjunto limitado de cargas unitárias $S = \{U_1, U_2, U_3, ..., U_s\}$ e uma lógica de classificação usada para atribuir cada novo pacote de entrada a um dos *unit loads*. Quando uma unidade atinge sua capacidade máxima (ou atinge qualquer outra *regra de fechamento*, como um tempo limite), ela é fechada e substituída por uma nova *unit load*. A *unit load* fechada está pronta para ser transferida para o próximo estágio da cadeia de abastecimento.

- Centro de expedição (CE): armazém ou depósito correspondente à origem das rotas de *last-mile*. Cada CE possui um sistema de depósito e localização geográfica, ou seja, latitude e longitude.

O Problema de Roteamento de Veículos Capacitados Dinâmicos com Clientes Estocásticos (DCVRP-SC) é uma variação estocástica e dinâmica do Problema de Roteamento de Veículos Capacitados (CVRP). Essa é uma maneira eficiente de modelar sistemas de roteamento de entrega onde encomendas e correspondências chegam horas extras durante o procedimento de triagem. Nessa situação, o depósito de entrega pode operar de maneira simplificada. Sempre que um novo lote de encomendas chega, eles são imediatamente atribuídos aos veículos de entrega. Essa formulação não permite esperar para atribuir um cliente a um veículo ou trocar de cliente entre os veículos porque isso viola a natureza simplificada da operação. As demandas futuras dos clientes são consideradas estocásticas. Os clientes estocásticos produzem uma demanda esperada caracterizada por uma probabilidade de chegada e pode mudar durante a janela de solução.

O problema abordado neste trabalho consiste em separar os pacotes de encomenda (um a um) entre um conjunto definido de *unit loads* de um *Warehouse system*, minimizando a distância percorrida pelos veículos para entregar todos os pacotes. Uma *unit load* será removida do *Warehouse system* e transformada em uma rota quando os pacotes dentro dela excederem qualquer um dos limites de capacidade máxima do veículo. Ao final do processo, cada *unit load* do *Warehouse system* será transformada em uma rota de veículos, garantindo o embarque de todos os pacotes. Por se tratar de um problema baseado no modelo de Loggi, não há restrição quanto ao tamanho da frota de veículos disponíveis, e todos os veículos possuem os mesmos limites de capacidade, ou seja, uma frota homogênea.

A Figura 3.1 apresenta um exemplo do Problema de Roteamento de Veículos Capacitados Dinâmicos com Clientes Estocásticos abordado neste trabalho. Na esquerda da imagem, são

apresentados a localização dos pacotes e do depósito ou *Expedition Center*; na direita o estado *Warehouse system* com 6 *Unit Loads* disponíveis. Defina-se como estado (S), cada configuração do *Warehouse system* dada uma sequência de pacotes de entrada. O S_1 apresenta o estado do primeiro pacote confirmado, esse pacote é direcionado para o $UnitLoad_1$ dada alguma política de separação. Seguidamente, chega o segundo pacote sendo configurado o S_2, onde o segundo pacote é inserido no $UnitLoad_3$, a distribuição de pacotes continua até chegar no S_6, onde algum dos critérios de fechamento de *Unit Load* é cumprido, sendo removidos os pacotes do *Warehouse system*, criada com eles a Rota_1. Dessa forma, dada uma sequência de pacotes de entrada em um *Warehouse system* são geradas dinamicamente rotas para entregá-los e o desafio é otimizar o critério da política de separação, visando minimizar a distância total de todas as rotas construídas.

3.2 Metodologia

Esta seção apresenta a metodologia proposta e os algoritmos de benchmark usados para avaliar o desempenho da proposta.

3.2.1 Metodologia Proposta

Este trabalho apresenta uma lógica de classificação *stream* para um *warehouse system*. A entrada do algoritmo é um fluxo de pacotes e as saídas são *unit loads*. As decisões de classificação são realizadas de forma incremental, ou seja, para cada novo pacote de entrada, a lógica o atribui a um *unit load*.

A plataforma multiagente usada nesse trabalho foi proposta por Barbucha and Jedrzejowicz (2008). A metodologia é baseado no middleware JABAT (JADE-Based A-Team), desenvolvida para resolver problemas difíceis de otimização combinatória (Barbucha et al., 2009). JABAT é funtamentado no paradigma de equipe assíncrona (A-Team do inglês *asynchronous team*), introduzido inicialmente por Talukdar et al. (1998). Este paradigma já foi aplicado para resolver o Problema de Roteamento de Veículos (Barbucha and Jedrzejowicz, 2008; Barbucha and Jędrzejowicz, 2009; Thangiah et al., 2001). Porém, nossa proposta utiliza técnicas de mineração de dados de trajetória para extrair informações estocásticas da distribuição de pacotes e, assim, melhorar a aposta feita pelos agentes, mantendo características como processamento paralelo e independência dos agentes.

A Figura 3.2 apresenta a estrutura do sistema MA proposto. Para cada novo pacote recebido, o agente PackageManager coleta as informações do pacote (destino, volume e peso) e as distribui entre todos os agentes UnitLoad. Cada agente UnitLoad é responsável por um *unit load*. O UnitLoad faz uma aposta em cada novo pacote distribuído pelo PackageManager

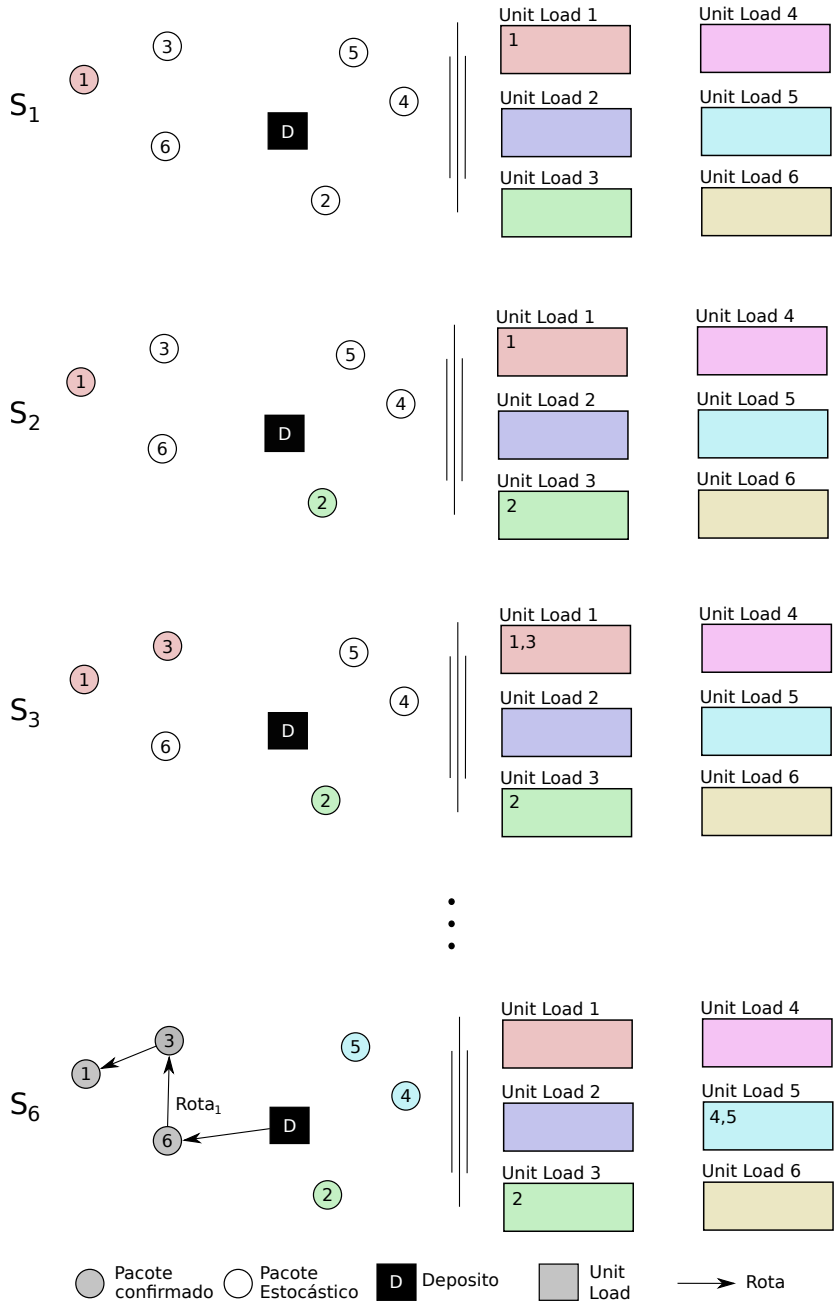


Figura 3.1: Exemplo do Problema de Roteamento de Veículos Capacitados Dinâmicos com Clientes Estocásticos (DCVRP-SC).

usando o estado atual do *unit load* correspondente. Finalmente, o PackageManager atribui o pacote ao *unit load* associado à aposta mais alta. As apostas são estimadas com base em uma combinação convexa da semelhança entre o pacote e os pacotes já atribuídos ao *unit load*; e algum estado interno do agente UnitLoad representando rotas históricas com uma alta probabilidade de recorrência. O estado interno de cada agente UniLoad é calculado antes do início do processo de classificação. As restrições de capacidade de *unit load* são verificadas para cada novo pacote, e se a eventual inclusão de um novo pacote violar alguma restrição, a aposta correspondente é 0. O agente UnitLoadManager recebe informações sobre o pacote e o *unit load* selecionada pelo PackageManager. Em seguida, ele insere o pacote no *unit load* e decide se fecha ou não. Se o *unit load* for fechada, o UnitLoadManager executa um algoritmo TSP usando o conjunto de pacotes atribuídos ao *unit load* e cria a rota de *last-mile* correspondente.

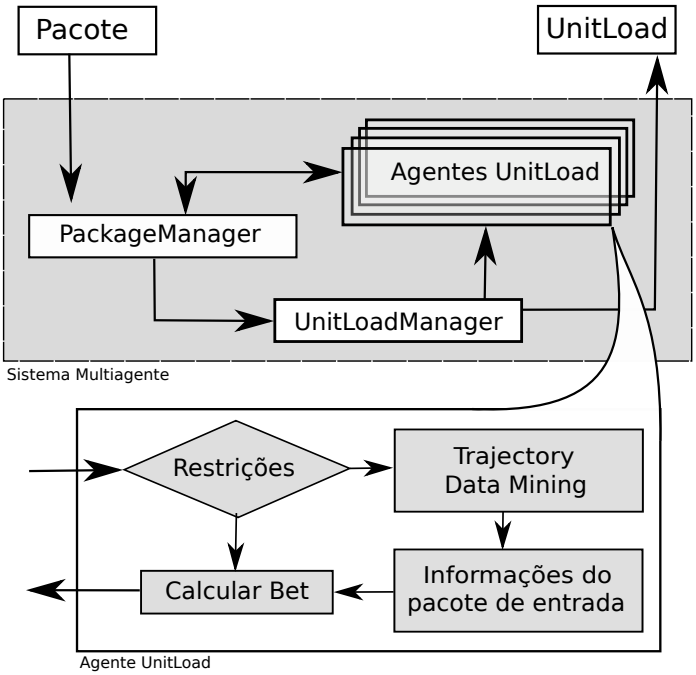


Figura 3.2: Estrutura da arquitetura Multiagente para o método proposto.

O sucesso da nossa proposta foca-se em calcular com eficiência a aposta de um agente UnitLoad. A aposta é definida como a combinação convexa de um valor estimado via mineração de dados de trajetória ($bet_{DataMining}$) e outro estimado usando apenas as informações reais dos pacotes no *unit load* ($bet_{Distance}$), conforme definido abaixo, onde $\rho \in [0, 1]$:

$$bet = \rho * bet_{DataMining} + (1 - \rho) * bet_{Distance} \tag{3.1}$$

3.2.2 Mineração de dados de trajetorias

Uma rota ou trajetória pode ser definida como a amostra do espaço-tempo do deslocamento de um objeto. Formalmente, $p = \{x, y, t\}$ representa a localização espaço-temporal de um objeto, onde x e y são os ângulos de latitude e longitude que representam a posição na Terra referenciada com o Equador e hora média de Greenwich, respectivamente, e t é o instante de tempo que obteve a amostra. Portanto, uma trajetória é uma sequência de pontos de localização, $T = \{p_1, p_2, ..., p_n\}$, onde n é o número de pontos que descrevem a trajetória. Uma rota é então uma trajetória, e poderíamos usar técnicas de mineração de dados de trajetória para extrair padrões territoriais e usá-los na criação dinâmica de rotas.

Regras de associação foram introduzidas em Agrawal and Srikant (1994) como um método para descobrir relações interessantes entre variáveis em grandes bancos de dados, usados principalmente em supermercados. Por exemplo, a regra {carne, carvão} ⇒ {cerveja} encontrada nos dados de vendas de um supermercado indica que se um cliente comprar carne e carvão juntos, ele provavelmente irá comprar cervejas. Essas informações podem ser utilizadas para maximizar as vendas de um supermercado, por exemplo, na localização de produtos ou estratégias promocionais. Em correlação com as trajetórias, os métodos de regras de associação aplicados às trajetórias extraem informações como $\{\text{area}_1, \text{area}_2\} \Rightarrow \{\text{area}_3\}$, o que significa que se um mensageiro entrega pacotes em $area_1$ e $area_2$, então ele entregará um pacote em $area_3$. O antecedente da regra é a área que contém os pacotes já na *unit load* e o consequente da regra, as áreas que contêm os pacotes que provavelmente farão parte da mesma rota. Esta interpretação é o principal pressuposto deste trabalho, ou seja, com base em dados históricos de trajetórias, é possível encontrar relações entre pacotes utilizados para a separação dinâmica de rotas. Além disso, uma vez encontradas as regras de associação, elas podem ser armazenadas em uma estrutura de dados *hashtable*, onde a chave é a regra e a saída é o suporte, permitindo uma separação rápida e eficiente dos pacotes.

A lógica de classificação proposta é dividida em duas fases. A primeira é a fase de planejamento, na qual são realizados o pré-processamento e a extração das regras de associação. Em seguida, na fase de execução, na qual as regras de associação extraídas de dados históricos e informações em tempo real sobre novos pacotes recebidos são utilizadas para estimar a aposta que cada agente UniLoad fará em um novo pacote.

Fase de planejamento

Esta fase é realizada *offline* e precisa ser atualizada periodicamente para adaptar as informações extraídas devido a mudanças de comportamento do consumidor ou mudanças nas regras de negócios. A etapa de planejamento é a tarefa de criar regras de associação usadas para calcular a aposta que cada agente UnitLoad coloca em novos pacotes recebidos. Esta fase é dividida em duas etapas. A primeira etapa realiza o pré-processamento dos dados, onde

cada rota histórica a ser considerada é transformada em uma sequência de regiões geográficas de granulação grossa. Na segunda etapa, o algoritmo FP-*growth* é usado para construir uma FP-*tree* que representa conjuntos de itens frequentes, ou seja, áreas frequentes que são visitadas por rotas históricas. As regras de associação são extraídas dos nós desta árvore.

Neste trabalho, propomos uma etapa de pré-processamento que mapeia uma sequência de pontos em uma sequência de áreas geográficas denominadas células. Este processo é realizado para reduzir a granularidade das informações a serem processadas pelo algoritmo FP-*growth*. O desafio é representar a Terra em áreas contínuas computacionalmente eficientes para fazer eficientes consultas. O Google fornece a biblioteca de geometria S2 (Eric et al., 2011), que faz uma projeção esférica da forma da Terra, permitindo-nos mapear todos os pontos do planeta usando matemática perfeita. Além disso, o S2 foi projetado para ter um excelente desempenho em grandes conjuntos de dados geográficos. A geometria S2 inicialmente divide a Terra em seis células com uma área média de $85011012,19km^2$, e então gera uma árvore de profundidade 30, onde cada nível é um quarto da área da célula do nó de base, sendo $0,74cm^2$ a área média do nível 30 células. Desta forma, as rotas podem ser representadas por áreas de diferentes níveis, e cada nível define diferentes tipos de padrão extraídos. Assim, a primeira etapa da fase de planejamento, mapeia cada trajetória histórica em uma sequência de células S2, dada uma granularidade desejada ajustada pela seleção de um nível apropriado da S2 *tree*. A etapa de pré-processamento consiste em mapear trajetórias em áreas geográficas, nós usamos a libraria geometria S2, porém, librarias como H3 do Uber[1] (Eric et al., 2011) baseada em *Geodesic discrete global grid systems* (Sahr et al., 2003) também pode ser utilizada.

A Figura 3.3 mostra um exemplo de duas trajetórias (T_1 em azul e T_2 em vermelho) mapeadas em células S2 de diferentes níveis. Esta figura mostra o efeito do ajuste da granularidade das informações expressas por uma trajetória usando níveis distintos da árvore S2. Quanto mais alto o nível (a), maior é a similaridade entre as rotas, mas os padrões extraídos são mais específicos nos níveis mais baixos. As trajetórias resultantes para cada cenário são representadas na Equação 3.2. Para níveis com áreas maiores, há uma semelhança maior entre as rotas, uma vez que a maioria das células são iguais, células A_4, A_5, A_6 na Figura 3.3a. Ainda assim, há uma perda de precisão na informação apresentada, célula C_{24} da Figura 3.3c. Células de níveis diferentes mapeiam trajetórias em padrões diferentes, portanto, recomendamos o uso de níveis diferentes nas trajetórias de pré-processamento.

[1]H3, Uber (acessado em 15 de março de 2021), `https://eng.uber.com/h3/`

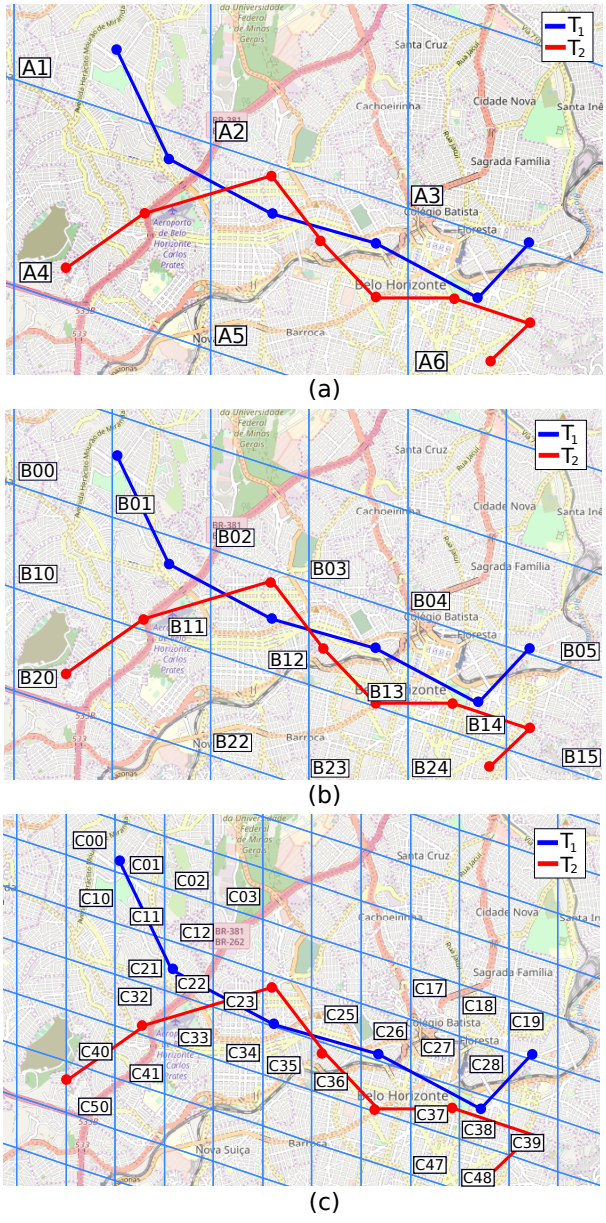


Figura 3.3: Etapa de pré-processamento de trajetórias. A trajetória azul (T_1) e a trajetória vermelha (T_2) mapeadas nas celulas S2 com diferentes niveis. Figura (a) com nível 11 (A), (b) com nível 12 (B), e (c) com nível 13 (C). O mapeamento é representado na Equação 3.2.

$$
\begin{aligned}
&T_1 = \{p_{11}, p_{12}, p_{13}, p_{14}, p_{15}, p_{16}\} \\
&T_2 = \{p_{21}, p_{22}, p_{23}, p_{24}, p_{25}, p_{26}, p_{27}, p_{28}\} \\
&\text{Level}_A : \begin{cases} T_1 = \{A_1, A_4, A_5, A_6, A_3\} \\ T_2 = \{A_4, A_5, A_6\} \end{cases} \\
&\text{Level}_B : \begin{cases} T_1 = \{B_{01}, B_{11}, B_{12}, B_{13}, B_{14}, B_{05}\} \\ T_2 = \{B_{20}, B_{11}, B_{12}, B_{13}, B_{14}, B_{15}, B_{24}\} \end{cases} \\
&\text{Level}_C : \begin{cases} T_1 = \{C_{01}, C_{22}, C_{24}, C_{26}, C_{28}, C_{19}\} \\ T_2 = \{C_{50}, C_{32}, C_{24}, C_{36}, C_{37}, C_{39}, C_{48}\} \end{cases}
\end{aligned}
\tag{3.2}
$$

O armazenamento de grandes base de dados de compras (como em supermercados, sites, restaurantes, etc.) impulsiona o desenvolvimento de técnicas capazes de prever compras futuras. Assim, agrupar itens para aumentar a probabilidade de eles serem adquiridos. Técnicas de mineração de dados são usadas para descobrir conjuntos de itens frequentes. Este problema é visto como regras de associação de mineração (Leskovec et al., 2014). Da mesma forma, em nosso problema, uma trajetória pode ser modelada como uma transação (compra realizada), onde as células visitadas podem ser representadas como itens comprados. Assim, o segundo passo é implementar o FP-*growth* nas trajetórias transformadas em uma sequência de células (ou itens) e extrair as regras de associação nos nós da FP-*tree* construída.

FP-*growth* foi introduzido em Han et al. (2000) como um algoritmo para extrair regras de associação em base de dados que contém transações, como base de dados de rotas. Da mesma forma que o Apriori utiliza o princípio da monotonicidade para realizar sua busca por regras de associação, garantindo que não tenha que varrer todas as possibilidades de combinações de conjuntos de itens, o que teria um custo proibitivo e exponencial. FP-*growth* cria uma estrutura de árvore compacta chamada árvore de padrão frequente ou FP-*tree* do inglês *frequent pattern tree*, que modera o problema *multiscan* e melhora a geração do conjunto de itens candidato.

Para explicar a implementação do FP-*growth*, selecionamos cinco trajetórias e implementamos a etapa de pré-processamento nas células S2 com nível 12, apresentadas na Equação 3.4. O objetivo é encontrar o conjunto de itens frequentes no nível 12, dadas essas trajetórias. As trajetórias visitam apenas o subconjunto de células $L_1 = \{B_{01} : 1, B_{02} : 1, B_{05} : 1\}$ da célula B total no nível 12. A primeira etapa de crescimento de FP é contar o número de rotas que entregam pacotes nas células L_1 (3.5), para ordenar em ordem decrescente (3.6), e podar este vetor usando um parâmetro de entrada chamado suporte, que indica com que frequência ele aparece no conjunto de dados, definido como

$$
\text{Support(X)} = \frac{|X \subseteq t, t \in T|}{|T|}, \tag{3.3}
$$

onde t é o número de trajetórias que visitaram o conjunto de células X, e T é o número total de transações. O suporte usado neste exemplo foi 50%, portanto, itens com menos de três contagens foram removidos.

$$\begin{aligned} T_1 &= \{B_{01}, B_{11}, B_{12}, B_{13}, B_{14}, B_{05}\} \\ T_2 &= \{B_{20}, B_{11}, B_{12}, B_{13}, B_{14}, B_{15}, B_{24}\} \\ T_3 &= \{B_{20}, B_{21}, B_{22}, B_{23}, B_{13}, B_{14}, B_{15}\} \\ T_4 &= \{B_{10}, B_{11}, B_{12}, B_{02}\} \\ T_5 &= \{B_{12}, B_{13}, B_{14}, B_{15}\} \end{aligned} \tag{3.4}$$

$$L_1 : \begin{cases} B_{01}:1, B_{02}:1, B_{05}:1, B_{10}:1, B_{11}:3, B_{12}:4, \\ B_{13}:4, B_{14}:4, B_{15}:3, B_{20}:2, B_{21}:1, B_{22}:1, \\ B_{23}:1, B_{24}:1 \end{cases} \tag{3.5}$$

$$L_1' : \{B_{12}:4, B_{13}:4, B_{14}:4, B_{11}:3, B_{15}:3\}. \tag{3.6}$$

Dado o conjunto de itens frequentes L_1', o segundo é construir o FP-*tree* com base no conjunto de trajetórias fornecidas. A FP-*tree* é mostrada na Figura 3.4, sua construção é feita através de cada uma das trajetórias:

- Os itens de T_1 são ordenados com base em L_1', gerando assim $T_1' = \{B_{12}, B_{13}, B_{14}, B_{11}, B_{01}, B_{05}\}$, onde B_{12} está vinculado como filho à raiz, o nó raiz é considerado nulo. B_{13} está vinculado a B_{12}, B_{14} a B_{13}, B_{11} a B_{14}, B_{01} para B_{11}, e B_{05} para B_{05}, conforme mostrado na Figura 3.4. Cada nó é marcado com o número de caminhos que passam por ele, na primeira instância todos são marcados com 1.

- Da mesma forma que T_1, a trajetória T_2 é atualizada em $T_2' = \{B_{12}, B_{13}, B_{14}, B_{11}, B_{15}, B_{20}, B_{24}\}$. Como a conexão B_{12} para root já existe, a contagem no nó B_{12} é aumentada em 2 e continuamos com as seguintes conexões. Desta forma, B_{13}, B_{14}, B_{11} também tem uma contagem de 2 e B_{15}, B_{20}, B_{24} são conectados à árvore. B_{15} a B_{11}, B_{20} a B_{15} e B_{24} a B_{20}, estes nós são marcados com 1.

- Em semelhança para T_1 e T_2, a trajetória T_3 é atualizada em $T_3' = \{B_{13}, B_{14}, B_{15}, B_{20}, B_{21}, B_{22}, B_{23}\}$. Porém, como nesta trajetória o item mais frequente não é B_{12}, B_{13} está vinculado como um filho à raiz 2 os outros nós estão conectados em sequência, cada um é marcado com 1.

- As trajetórias T_4 e T_5 foram inseridas na árvore da mesma forma que T_2 e T_3. Portanto, construímos a FP-*tree* da Figura 3.4 usando T_1, T_2, T_3, T_4 e T_5, e suporte de 50%.

O último passo é extrair itens frequentes da FP-*tree* com base no item L_1'. O último item em L_1' é B_{15}, isso ocorre em 3 *branch* $\{B_{12}, B_{13}, B_{14}, B_{11}, B_{15} : 1\}$, $\{B_{12}, B_{13}, B_{14}, B_{15} : 1\}$, e $\{B_{13}, B_{14}, B_{15} : 1\}$. Portanto, considerando B_{15} como sufixo, os caminhos de prefixo serão $\{B_{12}, B_{13}, B_{14}, B_{11} : 1\}$, $\{B_{12}, B_{13}, B_{14} : 1\}$, e $\{B_{13}, B_{14} : 1\}$. Isso forma a base do padrão condicional. Imediatamente, precisamos calcular o padrão condicional de base considerando a

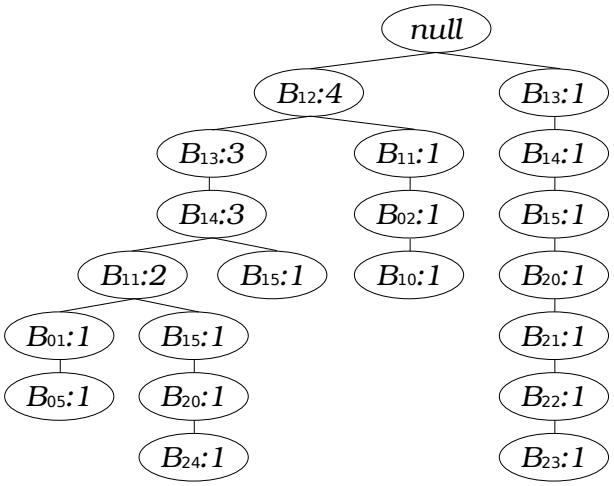


Figura 3.4: FP-*tree* construído com as trajetórias 3.4.

FP-*tree* construída, que são os prefixos comuns com maior suporte para suporte limitante, para B_{15} é $\langle B_{13} : 3, B_{14} : 3\rangle$. Finalmente, todas as combinações de padrões frequentes são calculadas. O processo de extração de itens frequentes é mostrado na Tabela 3.1.

Tabela 3.1: Mineração do FP-*tree*.

Item	Padrão base condicional	FP-*tree* condicional	Padrões Frequentes gerados
B_{15}	$\{\{B_{12}, B_{13}, B_{14}, B_{11} : 1\}, \{B_{12}, B_{13}, B_{14} : 1\}, \{B_{13}, B_{14} : 1\}\}$	$\{B_{13}, B_{14} : 3\}$	$\{B_{13}, B_{15} : 3\}, \{B_{14}, B_{15} : 3\}, \{B_{13}, B_{14}, B_{15} : 3\}$
B_{14}	$\{\{B_{12}, B_{13} : 3\}, \{B_{13} : 3\}\}$	$\{B_{12}, B_{13} : 3\}, \{B_{13} : 3\}$	$\{B_{12}, B_{14} : 3\}, \{B_{13}, B_{14} : 4\}$ $\{B_{13}, B_{14}, B_{15} : 3\}$
B_{13}	$\{\{B_{12} : 3\}\}$	$\{B_{12} : 3\}$	$\{B_{12}, B_{13} : 3\}$

O processo de extração de regras de associação (Piatetsky-Shapiro, 1991; Agrawal et al., 1993) consiste em encontrar, em um banco de dados, relações interessantes entre duas ou mais variáveis. Para um melhor entendimento, utilizamos o banco de dados T definido na Equação 3.4, cada trajetória é mapeada em ambas as células $S2$ nos níveis 12 e está usando o FP-*growth* para extrair conjunto de itens frequentes representados em um FP-*growth*, exibido na Tabela 3.1. Dado um conjunto de itens frequente $\{B_{13}, B_{14}, B_{15}\}$, pode construir as regras de associação: $B_{13}, B_{14} \rightarrow B_{15}$, $B_{13}, B_{15} \rightarrow B_{14}$, e $B_{14}, B_{15} \rightarrow B_{13}$. Na primeira regra B_{13}, B_{14} são o antecedente e B_{15} é o consequente.

No entanto, essas regras precisam ser validadas usando suporte (Equação 3.3) e confiança. A confiança é a probabilidade condicional de que uma transação contenha Y, uma vez que contém X, definido como

$$\text{Confidence}(X \mid Y) = \frac{\text{support}(X \cup Y)}{\text{support}(X)}. \tag{3.7}$$

Para extrair regras de associação em um conjunto de dados, em relação a quaisquer dois conjuntos de itens (X e Y), deve-se criar a regra desejada e calcular o suporte dos conjuntos de itens envolvidos. Uma vez calculada, pode-se calcular a confiança da regra usando a Equação 3.7, para cada um dos níveis mapeados. As regras de associação geralmente atendem a um nível de confiança mínimo de β e um nível de suporte mínimo de α, determinado para cada problema, de acordo com os requisitos desejados. O problema de encontrar regras de associação é que o número de conjuntos de itens cresce exponencialmente com o número de itens no banco de dados. Portanto, os valores α e β definem a profundidade do algoritmo de mineração e seu custo computacional.

Fase de execução

Nesta fase, para cada novo pacote recebido a ser atribuído a um *unit load*, cada agente UnitLoad calcula uma aposta a ser feita. A aposta é definida pela Equação 3.1 sendo composta por dois componentes. O primeiro, $bet_{DataMining}$, é baseado em uma regra de associação extraída de dados históricos e atribuída ao agente durante a fase de planejamento. Dadas as propriedades do novo pacote, o agente calcula sua confiança de regra de associação usando os pacotes já atribuídos como o antecedente da regra. Em outras palavras, suponha que queiramos calcular a aposta da inserção do ponto p_{16} na sub-trajetória de T_1, $T_{11} = \{p_{11}, p_{12}, p_{13}, p_{14}, p_{15}\}$, mostrado na Figura 3.3 e Equação 3.2. Primeiro, calcula-se a aposta de T_{11} e p_{16} nos níveis A, B e C, conforme pode ser visto em 3.8, e em seguida, a média de aposta para diferentes mapeamentos define a aposta do agente UnitLoad. A média foi usada porque representa uma ponderação da semelhança entre duas rotas. Pegando o exemplo anterior, $bet_{DataMining}$ conforme definido na Equação 3.9.

$$\begin{aligned} bet_a &= \text{Confidence}(A_1, A_4, A_5, A_6 \mid A_3) \\ bet_b &= \text{Confidence}(B_{01}, B_{11}, B_{12}, B_{13}, B_{14} \mid B_{05}) \\ bet_c &= \text{Confidence}(C_{01}, C_{22}, C_{24}, C_{26}, C_{28} \mid C_{19}) \end{aligned} \tag{3.8}$$

$$bet_{DataMining} = average(bet_A, bet_B, bet_C) \tag{3.9}$$

Os níveis A, B e C são selecionados baseado na distribuição das rotas no espaço, quanto mais alto o nível, menor a granularidade e maior é a similaridade entre as rotas, e níveis menores os padrões extraídos são mais específicos.

3.2.3 Informação de distância

Se a aposta foi baseada apenas no primeiro componente da Equação 3.1, um novo pacote recebido não poderia corresponder a nenhuma das regras de associação existentes, ou seja, nenhum agente colocaria uma aposta diferente de zero nele. Uma solução para esse problema seria rejeitar o pacote e apresentá-lo ao sistema posteriormente. No entanto, essa não é uma solução eficiente, pois rejeitar um grande conjunto de pacotes diminui a eficiência operacional. Para evitar este cenário, conforme discutido anteriormente, a aposta é definida como uma combinação convexa de dois componentes: um baseado em regras de associação e outro baseado na distância mínima entre o pacote de entrada p_p e todos os pacotes em cada trajetória $T_v = p_1, p_2, ..., p_n$. Este componente é denominado ($bet_{distance}$) e é definido como:

$$bet_{distance} = \frac{1}{1 + e^{\gamma(d_{min}(T_v) - \delta)}} \tag{3.10}$$

Os parâmetros *gamma* e *delta* da função logística podem ser encontrados usando *cross-validation*. $d_{min}(T_v)$ é a distância mínima entre o pacote de entrada, p_p, e todos os outros pacotes já atribuídos a uma dada trajetória T_v:

$$d_{min}(T_v) = min(d(p_1, p_p), ..., d(p_m, p_p)) \tag{3.11}$$

$d(p_x, p_y)$ é uma métrica que representa a distância entre dois pontos, por ex. Distância de Manhattan, distância terrestre em linha reta ou distância territorial baseado na rede rodoviária (Aggarwal, 2015), entre outras.

3.2.4 Algoritmos *Benchmark*

Para estimar a eficácia de nossa abordagem na solução de DCVRP *with stochastic consumers*, comparamos seus resultados com dois métodos de *Benchmark* amplamente utilizados na literatura: o algoritmo guloso que resolve um algoritmo VRP dinâmico, e a Abordagem de Cenários Múltiplos proposta para o VRP parcialmente dinâmico com clientes estocásticos. Os algoritmos tiveram que ser adaptados ao nosso problema, e estas modificações são descritas a seguir.

Algoritmo Guloso

O algoritmo guloso ou *greedy* (Ritzinger et al., 2016; Larsen et al., 2002) é a abordagem tradicional para VRP dinâmico. Na implementação original, este resolve VRP parcialmente dinâmico. Existe um conjunto de requisições iniciais e novas requisições que precisam ser inseridas dinamicamente nos veículos ao longo do tempo. A inserção pode permitir ou não o

reencaminhamento das encomendas entre as rotas, podendo os veículos ser capacitados. Portanto, o algoritmo possui duas etapas, uma estática em que implementa um CVRP para as solicitações iniciais. A segunda etapa otimiza novamente a solução inicial para incluir as novas solicitações na solução obtida anteriormente. A segunda fase é implementada cada vez que um conjunto de novas requisições chega ao depósito. No entanto, nosso problema é totalmente dinâmico; isso significa que a primeira etapa não foi implementada porque não há requisições iniciais.

O algoritmo é executado no *Warehouse system* e os novos pacotes são inseridos um a um. Assim, a etapa de reotimização sempre busca o melhor caminho para inserir o novo pacote, consertando os pacotes da solução anterior. Além disso, como não é permitido redirecionar pacotes e inserir um pacote por vez, o problema do CVRP equivale a calcular o custo de inserir o pacote em cada rota e selecionar o veículo de menor custo. Nesse caso, o custo é definido pela diferença na distância da rota com e sem pacote. O problema de calcular a rota de distância mínima dado um ponto é equivalente a um TSP, então usamos *the heuristic guided local search* (Voudouris and Tsang, 2003) para escapar do mínimo local. O custo de inserção do pacote de entrada P_p na rota ou trajetória de cada veículo T_v é dado por

$$cost_{greedy}(T_v, P_p) = d_T(TSP(T_{v,p})) - d_T(TSP(T_v)) \tag{3.12}$$

onde $T_{v,p}$ é a união de T_v pontos e o novo ponto P_p, e d_T é a distância da rota traçada pelo TSP.

Para fazer uma comparação justa entre o algoritmo guloso e nossa proposta, o algoritmo guloso foi implementado usando uma adaptação da estrutura do sistema MA proposta, como pode ser visto na Figura 3.5. As tarefas executadas pelos agentes PackageManager, UnitLoad, UnitLoadManager são as mesmas. A única diferença é que a aposta a ser estimada para cada UnitLoad agora é baseada no custo da inserção gulosa (Equação 3.12) e não nas trajetórias das técnicas de mineração. Nesse caso, o agente PackageManager seleciona o agente UnitLoad que possui o menor custo de inserção.

Abordagem de Cenários Múltiplos

A Abordagem de Cenários Múltiplos (ou MSA do ingles *The Multiple Scenario Approach*) foi proposta por Bent and Van Hentenryck (2004) para resolver um VRP parcialmente dinâmico com clientes estocásticos e janelas de tempo; no entanto, relaxamos as restrições das janelas de tempo para o nosso problema. Este algoritmo não possui uma fase de treinamento. Sua fase de execução encontra a rota ideal em cada ponto de decisão por meio da amostragem de cenários antecipados e do cálculo da decisão que retorna o menor custo total médio entre as amostras.

Da mesma forma que o algoritmo guloso, foi implementado utilizando uma adaptação da estrutura do sistema MA proposta, conforme pode ser visto na Figura 3.6. O agente PackageManager coleta as informações do pacote e as distribui entre todos os agentes de cenário. Cada

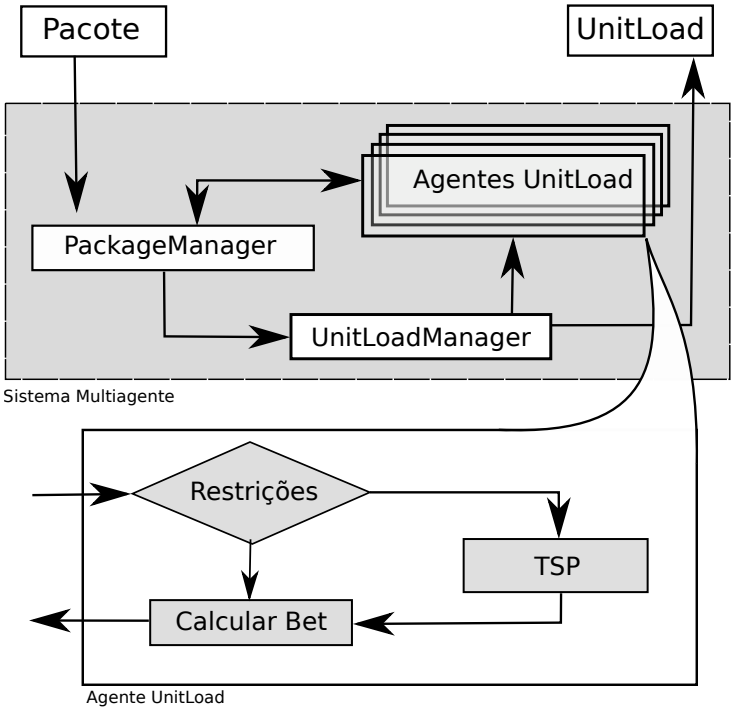


Figura 3.5: Estrutura da arquitetura Multiagente para o algoritmo guloso.

agente de cenário recebe o novo pacote, os dados dos pacotes armazenados no *unit load* do *Warehouse system*, as informações contidas nos agentes UnitLoad, e carrega um cenário estocástico para executar um CVRP com a restrição de fixar os pacotes já separados nos *unit loads*. Um cenário estocástico é um conjunto de requisições futuras obtidas por amostragem de suas distribuições de probabilidade. Visto que este trabalho assume que a distribuição geográfica do pacote não muda significativamente ao longo de uma sequência de dias, a distribuição de probabilidade é representada por um conjunto de pacotes anteriores de tamanho N_s.

Em seguida, o PackageManager seleciona o cenário de menor custo e atribui o pacote à *unit load* associada à solução desse cenário. O custo de cada cenário é a soma das distâncias percorridas por todos os veículos da solução CVRP. Empregamos o método *Path Most Constrained Arc* da biblioteca OR-Tools[2], que é uma versão adaptada do *nearest neighbors* (Cook, 27 Dec. 2011) que melhor satisfaz as restrições de sub-roteamento fixas. Finalmente, o agente UnitLoadManager implementa a mesma tarefa que a estrutura MA proposta; recebe informações sobre o pacote e a *unit load* selecionada pelo PackageManager e a seguir insere o pacote na *unit load*,

[2]OR-Tools (7.2), Google (acessado em 15 de dezembro de 2020), `https://developers.google.com/optimization/`

verificando se deve ou não fechá-la. Trabalhamos em cenários de 12 em nossa implementação, e o tamanho do conjunto de pacotes anterior era $N_s = 200$. Além disso, paralelizamos 6 threads para minimizar o tempo computacional. Os pacotes estocásticos foram carregados sem peso e volume para não exceder as capacidades do VRP.

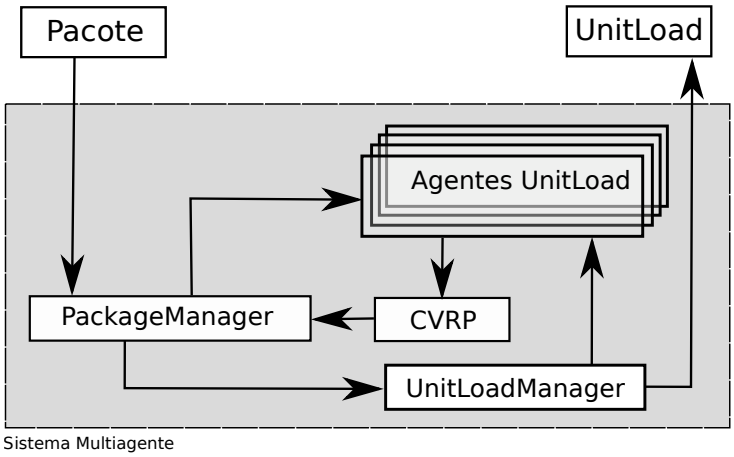


Figura 3.6: Estrutura da arquitetura Multiagente para MSA.

3.3 Resultados experimentais

Esta seção apresenta os experimentos conduzidos para validar o desempenho da abordagem proposta. Inicialmente, descrevemos o conjunto de dados utilizado juntamente com um teste estatístico para apoiar a hipótese de que existem correlações temporais entre as distribuições espaciais dos pacotes durante os dias da semana. A seguir, apresentamos os detalhes da configuração experimental. Em seguida, avaliamos a eficácia de nossa proposta em resolver DCVRP com informações estocásticas em duas situações. No primeiro, comparamos nossa proposta com os algoritmos *benchmark* descritos na Seção anterior 3.2.4. Posteriormente, comparamos nossa metodologia dinâmica com uma heurística de pesquisa local comumente usada na indústria para resolver VRPs estáticos. A heurística estática foi escolhida porque simula as condições ótimas do problema em que todas as informações dos pacotes são conhecidas com antecedência. Assim, podemos contrastar as soluções de ambos os métodos para validar requisitos desejados de qualquer solução de VRP, como robustez a variações nas entradas e desempenho geral.

3.3.1 Dataset

O algoritmo de roteamento proposto foi avaliado usando uma base de dados históricos fornecido pela Loggi. O banco de dados corresponde a $136,000$ pacotes entregados em 2019 a partir de 3 ECs localizados na cidade de Belo Horizonte, Brasil. O conjunto de dados foi dividido em um subconjunto treinamento e um de teste. O conjunto de treinamento tem duas partições distintas: $100,000$ pacotes foram usados para construir regras de associação na mineração de dados de trajetória e $1,500$ foram usados para selecionar os hiperparâmetros γ e δ da metodologia proposta, Equação 3.10. O conjunto de teste é composto por $34,000$ pacotes. Cada pacote é representado por sua localização de destino (latitude e longitude), peso, volume e CE de partida (depósito). Os percursos foram realizados em motocicletas e estão limitados pela capacidade máxima deste veículo, ou seja, volume máximo de 110 litros, peso máximo de 20 kg e número máximo de 25 pacotes. O sistema de almoxarifado possui 28 *unit loads* para cada CE ou origem; este número é baseado nas restrições operacionais da Loggi. O fechamento da *unit load* é realizado quando algum dos valores for igual ou superior a 80% do limite do veículo. A mesma configuração do *Warehouse system* foi usada nos métodos *baselines*. Para o MSA, foi usado um subconjunto de $100,000$ pacotes como informação estocástica.

3.3.2 Suposição Estatística

O sistema Multi Agente proposto neste trabalho assume que a distribuição geográfica dos destinos dos pacotes não muda significativamente ao longo de uma sequência de dias. Para confirmar essa hipótese, aplicamos o *the Kernel Density-Based Global Two-Sample Comparison Test* (Duong et al., 2012) usando um conjunto de dados históricos de pacotes entregues ao longo de uma semana. Este é um teste não paramétrico e assintoticamente normal proposto para comparar a distribuição de dados n-dimensionais que representam morfologias celulares.

O teste foi implementado para verificar se a distribuição espacial dos pacotes durante a semana de março era semelhante. A hipótese nula é $H_0 : F_l \equiv F_m$, onde F_l, F_m são as respectivas densidades diárias de pacotes de um par de dias em uma determinada semana. Tabela 3.2 apresenta os $p-values$ a um nível de significância de 5%. Como pode ser observado na Tabela 3.2, não há evidências estatísticas para rejeitar a hipótese nula, de que a distribuição espacial dos pacotes é semelhante ao longo dos dias da semana. A Figura 3.7 mostra a distribuição geográfica dos pacotes na mesma semana de março, ilustrando graficamente os resultados obtidos no teste.

3.3.3 Configuração Experimental

Os experimentos foram implementados em Python usando OR-Tools (Perron and Furnon, 2019), que é um software de código aberto para otimização, especializado em problemas como

Tabela 3.2: p-value do *kernel density based global two-sample comparison test* para uma semana em Março de 2019.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Mon	1	0.462	0.377	0.394	0.430	0.401	0.252
Tue		1	0.495	0.467	0.528	0.549	0.405
Wed			1	0.440	0.459	0.466	0.245
Thu				1	0.536	0.457	0.336
Fri					1	0.528	0.355
Sat						1	0.273
Sun							1

roteamento de veículos, fluxos, programação inteira e linear e programação por restrição. A implementação do algoritmo FP-*growth* é baseada no PySpark. PySpark é um *wrapper* do python para Spark (Spark, 2018), uma estrutura de código aberto para computação distribuída, permitindo a aplicação de mineração de dados de trajetória em grandes conjuntos de dados. Usamos a biblioteca S2 (Eric et al., 2011) do Google para selecionar áreas na fase de pré-processamento das trajetórias.

O ajuste dos hiperparâmetros γ e δ de $bet_{distance}$ foi realizado usando validação cruzada k-*fold* (Stone, 1974). A Equação 3.13 mostra a faixa de valores testados para esses hiperparâmetros durante o processo de validação cruzada. As constantes $\frac{10}{2000}$ e 5000 são fatores de escala típicos para esses parâmetros.

$$\begin{aligned} \gamma &: \tfrac{10}{2000} * [0, 0.1, ..., 0.9, 1] \\ \delta &: 5000 * [0, 0.1, ..., 0.9, 1] \end{aligned} \tag{3.13}$$

Conforme mencionado anteriormente, a abordagem proposta pretende gerar as rotas de forma dinâmica, supondo que os pacotes cheguem em um fluxo contínuo.

As rotas criadas usando VRP estático em lotes de pacotes gerados a partir dos subconjuntos de treinamento foram usadas para executar regras de associação. O método *Path Most Constrained Arc* da biblioteca OR-Tools[3] foi empregado, assim como no MSA (veja a Seção 3.2.4). Consideramos um depósito no centro da cidade e dividimos os pacotes de treinamento em 20 lotes de $2,000$ pacotes cada. Assim, mapeamos o conjunto de $6,006$ rotas em áreas de

[3]OR-Tools (7.2) Google (acessado em 15 de dezembro de 2020), https://developers.google.com/optimization/

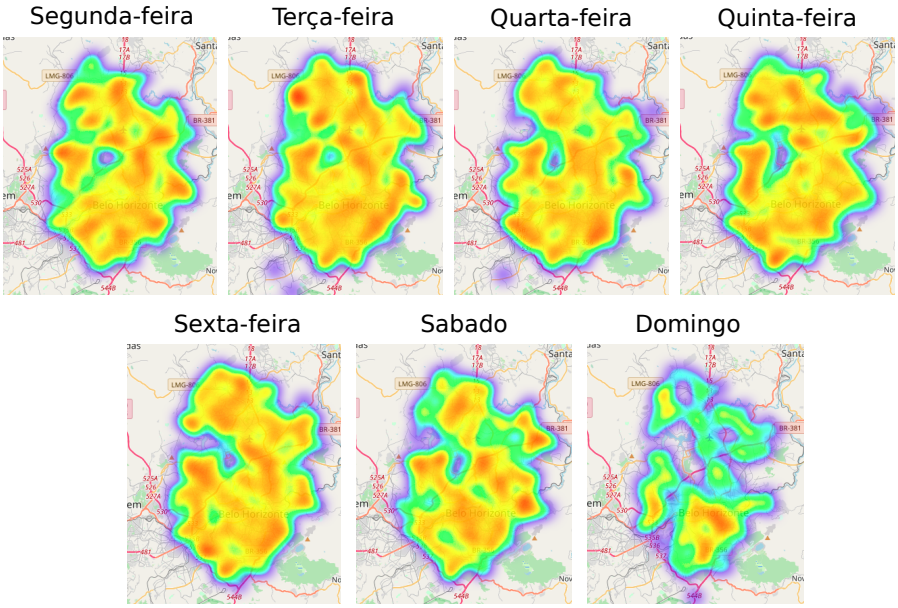


Figura 3.7: *Heatmap* da distribuição dos pacotes em uma semana do Março de 2019.

células S2 de tamanhos diferentes, conforme ilustrado anteriormente na Figura 3.3. A tabela 3.3 mostra as características das células S2 e o número de regras de associação extraídas. Esta configuração foi usada nos testes a seguir.

3.3.4 Comparação com outras heurísticas VRP dinâmicas

O VRP é um problema de otimização em que o objetivo é minimizar a distância (d) percorrida por todos os veículos. Porém, nosso problema se aplica ao modelo de negócio da Loggi, em que os motoristas preferem rotas com mais pacotes e a frota de veículos operacionalmente não é infinita devido ao modelo de economia compartilhada. Portanto, além da distância percorrida (d), consideramos também o número de rotas e pacotes por rota como métricas adicionais para comparação dos resultados obtidos.

O modelo proposto e os algoritmos de *benchmark* (MSA e guloso) foram testados em 10,000 pacotes do conjunto de dados de teste, foi selecionado um sub-conjunto do conjunto total de teste, devido ao custo computacional que do MSA. Os resultados são apresentados na Tabela 3.4, e as taxas de desempenho em relação ao VRP estático são mostradas na Tabela 3.5. Em comparação com o algoritmo guloso, pode-se ver que a distância total percorrida é significativamente menor, apesar de haver mais algumas rotas. O método proposto apresenta uma melhoria de aproximadamente 43% na distância total, com apenas um aumento de 1% nas rotas e paco-

Tabela 3.3: Características das células S2 usadas e o número de regras de associação extraidas com FP-*growth* em cada nível.

Celulas		Regras de Associação
Nível	Área média (Km^2)	
11	20.27	691
12	5.07	4541
13	1.27	17844
14	0.32	72080
15	0.08	174147

tes. Essa mineração de dados de trajetória pode extrair informações estocásticas para melhorar os resultados em um problema totalmente dinâmico. Também comparamos nossos resultados com o método MSA, um algoritmo consolidado na literatura para resolver o DVRP com clientes estocásticos, mas adaptado ao nosso problema específico. Os resultados do MSA foram melhores do que o algoritmo guloso, mas piores do que o algoritmo proposto. O algoritmo proposto foi aproximadamente 24% melhor do que o MSA, com apenas um aumento de 1% nas rotas e pacotes.

Tabela 3.4: Resultados experimentais.

Modelo	Distância (km)	Rotas	Pacotes	Tempo de Execução (min)
Proposta	7.65e+03	804	12.44	2.75
Guloso	13.22e+03	774	12.91	46.82
MSA	10.44e+03	795	12.57	6675.57

Nossa abordagem implementa um DCVRP com clientes estocasticos, que permite a separação de pacotes utilizando hardware já consagrado na indústria, como, por exemplo, o *Warehouse system*. Uma contribuição significativa apresentada por nosso método foi o uso de técnicas de mineração de dados para resolver um problema NP-*Hard* utilizando uma heurística com tempo de execução linear nas especificações físicas do *Warehouse system*. Em outras palavras, a complexidade do nosso modelo é $\mathcal{O}(n_{CE} * n_U * (1 + n_{Up}))$, onde n_{CE} é o número de CEs, n_U é o número de *unit loads* do *Warehouse system*, $(1 + n_{Up})$ é o custo computacional do algoritmo de mineração de dados de trajetória e o pacote para o cálculo de distâncias em agentes UnitLoad.

Tabela 3.5: Resultados experimentais em relação ao modelo proposto (*benchmark*/proposta).

Modelo	Distância	Rotas	Pacotes
Guloso	0.573	1.043	0.959
MSA	0.736	1.011	0.989

A respeito disso, uma vez que as regras de associação podem ser armazenadas em uma *hashtable* e a operação *compute bet* tem $\mathcal{O}(1)$ complexidade; e como a operação compute bet precisa obter a distância entre um determinado pacote e todos os outros pacotes em cada unidade de carga, portanto $\mathcal{O}(n_{Up})$. Portanto, concluímos que a complexidade do modelo não depende do número de pacotes processados. Dessa forma, podemos fornecer estabilidade para problemas com grandes quantidades de pacotes (*Big Data Problems*), como é o problema exposto neste trabalho. Os resultados relacionados ao tempo de execução em minutos apresentados na Tabela 3.4 mostram grande melhora na complexidade do tempo da solução proposta em comparação com os outros algoritmos de *benchmark*.

Os resultados experimentais demonstram a viabilidade do modelo proposto como uma possível solução para o problema de roteamento dinâmico de veículos com capacidade e clientes estocásticos. Ainda assim, não é possível analisar os padrões extraídos nas trajetórias. Para tanto, a Figura 3.8 mostra as 10 primeiras rotas criadas pelo modelo proposto e algoritmos de *benchmark*. O mapa à esquerda mostra as rotas geradas pelo método proposto, o mapa central se refere ao algoritmo guloso e as rotas de mapa à direita foram criadas pelo MSA.

Conforme esperado na proposta, foram criadas rotas específicas para locais remotos, juntando pacotes de uma mesma área, devido ao agrupamento implícito das rotas representadas nas células S2. Além disso, é possível perceber as rotas em forma de raio com o centro no CE, característica das rotas abertas, o que não otimiza o retorno ao CE. Por fim, conforme mostrado na Figura 3.8c, o algoritmo MSA também possui duas características mencionadas anteriormente, ou seja, algumas rotas, como a preta e a vinho tinto, contêm pacotes de regiões específicas. No entanto, ele também cria rotas como a roxa, que é uma rota longa que entrega pacotes por todo o caminho.

3.3.5 Comparação com uma heurística estática

Em pequenos *dataset*, o problema de entrega da *last-mile* geralmente é modelado como um VRP estático. Nesse contexto, o número de clientes costuma ser pequeno e todas as informações necessárias costumam ser conhecidas com antecedência. Porém, em configurações dinâmicas de *big data*, como a abordada neste trabalho, o número de clientes a serem atendidos e o dinamismo do mercado tornam esse problema solucionável usando apenas algoritmos dinâmicos.

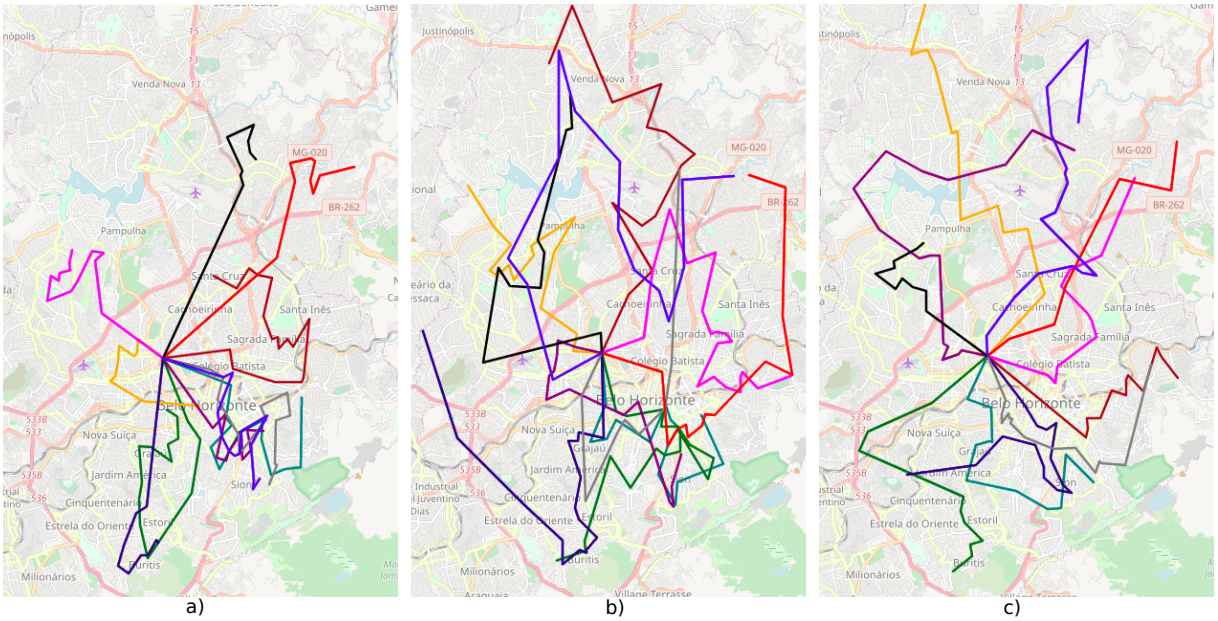


Figura 3.8: Primeiras 10 rotas criadas pelo modelo proposto (a), algoritmo guloso (b), e MSA (c).

Nesta seção, comparamos nossa metodologia dinâmica proposta com uma heurística estática implementada na biblioteca de ferramentas OR-Tools (heurística construtiva *Path Most Constrained Arc* e heurística *guided local search*), que é uma das bibliotecas mais usadas na indústria para resolver VRPs estáticos (Bergmann et al., 2020). Embora as heurísticas estáticas geralmente não sejam adequadas para resolver problemas como o abordado neste trabalho, realizamos algumas comparações para validar empiricamente características desejadas importantes de qualquer solução de VRP, como robustez a variações nas entradas e desempenho geral.

Criamos 30 cenários de teste, cada um considerando $2,500$ pacotes escolhidos aleatoriamente do conjunto de teste. Usamos um número reduzido de pacotes neste experimento porque a heurística estática usada no experimento não converge para tamanhos de amostra maiores. As Figuras 3.9.a e 3.9.b apresentam o tempo de execução e a distância total para os dois métodos. Adicionalmente, a 3.9.c mostra a diferença entre os resultados, ou seja, para cada cenário testado, a razão dos resultados obtidos pelo modelo proposto e a heurística estática.

A distância total do algoritmo proposto foi 33% pior do que a heurística do OR-Tool. Este era um resultado esperado, pois a heurística estática tem acesso a todas as informações sobre o problema antes do início de sua execução, enquanto na configuração dinâmica utilizada pela heurística dinâmica a informação é apresentada de forma incremental durante a execução da heurística.

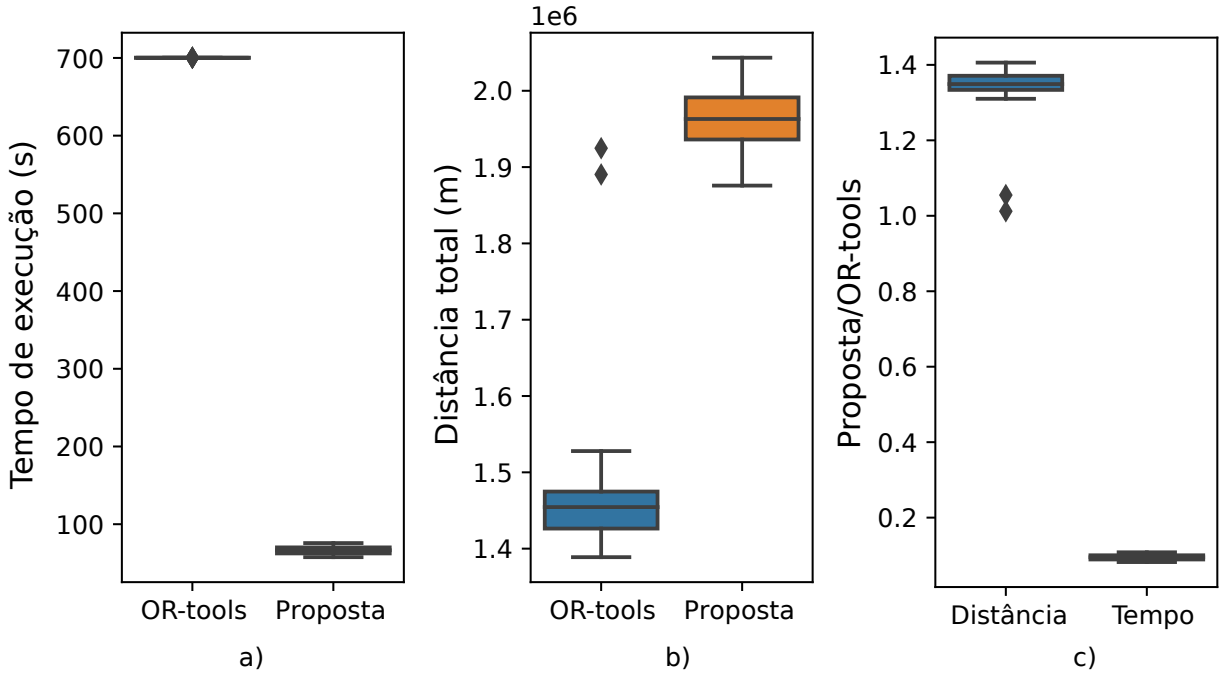


Figura 3.9: A diferença entre a distância total e o tempo de execução entre o método proposto e o VRP estático.

O método proposto foi 91% mais rápido do que a heurística do OR-Tool; isso também era esperado porque a complexidade de tempo da heurística estática usada no OR-Tool depende do número de pacotes, enquanto nossa abordagem é determinística com uma complexidade de tempo que não depende disso (a complexidade de tempo da metodologia proposta foi apresentada na seção anterior seção).

O resultado da distância total da heurística estática é superior à nossa proposta. Porém, como mencionado anteriormente, seu uso é inviável para o processamento de grandes volumes de pacotes. Além da complexidade de tempo significativamente maior, também aumenta o custo operacional do armazém. As heurísticas estáticas exigem todas as informações sobre o problema antes do início de sua execução (ou seja, volume, peso, destino de cada pacote). Para coletar essas informações geralmente é necessário reservar uma grande área do *warehouse* para armazenamento dos pacotes, criando lotes que são processados por uma heurística estática. A heurística dinâmica proposta não exige nenhuma informação prévia, ela processa os pacotes de forma incremental, à medida que chegam, não exigindo a criação de lotes e reduzindo significativamente o custo do *warehouse*.

Em seguida, a robustez do método proposto foi avaliada empiricamente para mostrar que o resultado do modelo proposto não está condicionado à ordem das instâncias no conjunto de

dados de teste. Para isso, 30 cenários foram gerados embaralhando os mesmos $2,500$ pacotes, e cada um dos conjuntos de dados resultantes foi usado como entradas do método proposto e do método VRP estático.

A robustez de um modelo é medida pela variância das métricas. Usamos o teste Levene com nível de significância de $0,05$ para verificar se as variâncias da distância total e do tempo de execução são iguais em ambos os modelos. A hipótese nula é $H_0 : S_1^2 \equiv S_2^2$ onde S_1^2 e S_2^2 são as variâncias da amostra para o método proposto e a heurística VRP estática.

Em relação à distância total, o *p-value* foi de $0,07$, o que confirma que não há evidências estatísticas para rejeitar a hipótese nula, mostrando que a abordagem proposta é tão robusta quanto implementar a heurística construtiva *Path Most Constrained Arc* e a heurística *guided local search* do OR-Tools. Porém, para o tempo de execução, o *p-value* foi $1,73e-06$ o que indica que há evidências estatísticas para rejeitar a hipótese nula; neste caso, a variação do tempo de execução do método proposto foi de $39,05$ segundos, enquanto do VRP estático foi de $616,66$ segundos. Isso é explicado pela natureza determinística do modelo proposto, em comparação com a implementação estática do VRP usando uma heurística construtiva e a heurística *guided local search* com um limite de tempo. Os histogramas da Figura 3.10 mostram os dados obtidos nas simulações. O valor da mediana foi subtraído dos resultados para obter uma visualização clara da diferença nas variâncias.

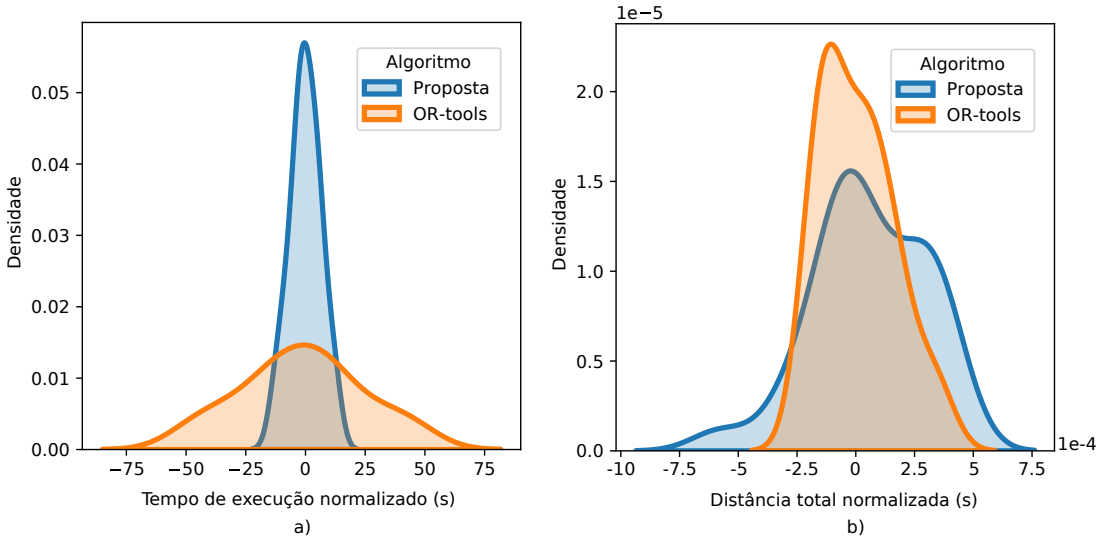


Figura 3.10: Distância total e tempo de execução entre o VRP proposto e estático para cenários de 30 gerados pelo embaralhamento dos mesmos $2,500$ pacotes.

Capítulo 4

Gerenciamento e Detecção de *Outliers* no *Last-Mile*

4.1 Introdução

Este capítulo apresenta uma prova de conceito da utilização de um algoritmo de agrupamento incremental aplicado aos problemas do gerenciamento e a detecção de rotas de *Last-Mile* anômalas exposto na Seção 2.3 e na Sub-Seção 2.3.4.

A abordagem proposta neste Capítulo refere-se à adaptação do MicroTEDAclus para o agrupamento de trajetórias, visando resolver os dois problemas na logística de *e-commerce* apresentados. O objetivo é criar grupos de rotas construídas no *cross-docking* e estes grupos podem ser atribuídos para um grupo de mensageiros especializados em uma dada região. O uso de um algoritmo de agrupamento evolutivo para esse problema, além de fazer o agrupamento das trajetórias mantendo a zonalidade nos grupos, traz vantagens como:

- se existirem mudanças na abrangência das entregas, os *macro-clusters* e *micro-clusters* conseguem-se adaptar automaticamente;
- grandes empresas de logística entregam diariamente centenas de milhares de pacotes, armazenar os grupos em *micro-clusters* representados pelo centroide, variancia e número de amostras, permitem a sua escalabilidade devido ao baixo consumo de memória e baixo custo de processamento;
- erros operacionais podem fazer que sejam construídas rotas não desejadas, que represen-

tam um custo alto devido que inserção errada de pacotes pode gerar rotas não otimizadas por conseguinte, mais longas. O uso do MicroTEDAclus como detecção de *outliers* permitiria identificar e reprocessar estes pacotes, representando uma diminuição no custo final da entrega destes pacotes.

O restante do capítulo está organizado da seguinte forma. Na Seção 4.2 apresentação do algoritmo de detecção de anomalias TEDA e o algoritmo de agrupamento MicroTEDAclus. Na Seção 4.3, apresentamos a metodologia proposta da adaptação do MicroTEDAclus para usar com dados de trajetórias. Finalmente, na Seção 4.4, são discutidos os experimentos realizados.

4.2 Referencial Teórico

4.2.1 TEDA

O algoritmo proposto é baseado nos conceitos do *framework* TEDA (Angelov, 2014a). TEDA é um algoritmo para detecção de anomalias que modela incrementalmente uma distribuição de dados com base apenas na proximidade entre as amostras de dados.

TEDA é baseado no conceito de proximidade cumulativa. Dado um vetor de entrada d-dimensional $\boldsymbol{x_k} \in \mathbb{R}^d$, no *timestamp* k, a proximidade cumulativa $\pi(.)$ de $\boldsymbol{x}_k$ em relação a todas as amostras de dados existentes, é calculado como

$$\pi_k(\boldsymbol{x}) = \sum_{i=1}^{k} d(\boldsymbol{x}_k, \boldsymbol{x}_i) \ , \tag{4.1}$$

onde $d(\boldsymbol{a}, \boldsymbol{b})$ é a distância entre os pontos de dados $\boldsymbol{b}$, k é o timestamp quando o dado $\boldsymbol{x}$ é amostrado.

De $\pi_k(\boldsymbol{x})$ calculamos a excentricidade $\xi_k(\boldsymbol{x})$, que é uma medida da dissimilaridade entre o ponto de dados $\boldsymbol{x}_k$ em relação a todos os pontos de dados recebidos até o timestamp k

$$\xi_k(\boldsymbol{x}) = \frac{2\pi_k(\boldsymbol{x})}{\sum_{i=1}^{k} \pi_k(\boldsymbol{x}_i)} \ , \ \sum_{i=1}^{k} \pi_k(\boldsymbol{x}_i) > 0 \ , \ k > 2. \tag{4.2}$$

Para o caso da distância euclidiana, a excentricidade pode ser calculada recursivamente como segue Angelov (2014b)

$$\xi(\boldsymbol{x}_k) = \frac{1}{k} + \frac{(\boldsymbol{\mu}_k - \boldsymbol{x}_k)^T(\boldsymbol{\mu}_k - \boldsymbol{x}_k)}{k\sigma_k^2}, \tag{4.3}$$

onde $\boldsymbol{\mu}_k$ e σ_k^2 são a média e a variância respectivamente, que também podem ser atualizados recursivamente

$$\boldsymbol{\mu}_k = \frac{k-1}{k}\boldsymbol{\mu}_{k-1} + \frac{\boldsymbol{x}_k}{k} \ , \ k \geq 1 \ , \ \boldsymbol{\mu}_1 = \boldsymbol{x}_1. \tag{4.4}$$

$$\sigma_k^2 = \frac{k-1}{k}\sigma_{k-1}^2 + \frac{1}{k-1}\|\boldsymbol{x}_k - \boldsymbol{\mu}_k\|^2 \ , \ \sigma_1^2 = 0. \tag{4.5}$$

A tipicidade $\tau(\boldsymbol{x}_k)$ é o dual da excentricidade e representa o quão típico é um ponto de dados arbitrário $\boldsymbol{x}_k$ em relação a todos os pontos de dados recebidos até o timestamp k

$$\tau(\boldsymbol{x}_k) = 1 - \xi(\boldsymbol{x}_k) \ , \ k \geq 2. \tag{4.6}$$

Os conceitos de tipicidade e excentricidade são representados na Figura 4.1 (Bezerra et al., 2016). O ponto de dados "A" está mais distante do conjunto de dados do que o ponto de dados "B", portanto "A" tem maior excentricidade e menor tipicidade do que "B".

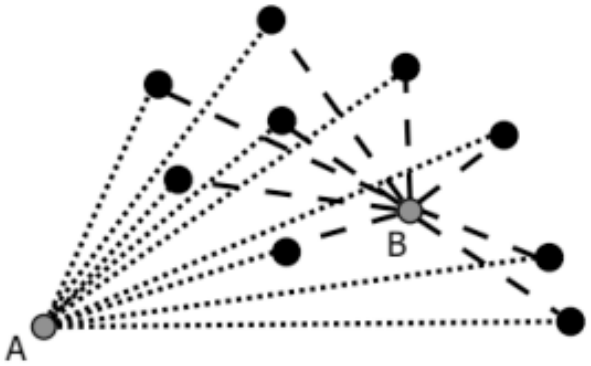


Figura 4.1: Conceitos de tipicidade e excentricidade no TEDA Bezerra et al. (2016).

A excentricidade normalizada $\zeta(\boldsymbol{x}_k)$ e a tipicidade $t(\boldsymbol{x}_k)$ podem ser obtidas da seguinte forma

$$\zeta(\boldsymbol{x}_k) = \frac{\xi(\boldsymbol{x}_k)}{2} \tag{4.7}$$

$$t(\boldsymbol{x}_k) = \frac{\tau(\boldsymbol{x}_k)}{k-2} \tag{4.8}$$

A excentricidade normalizada $\zeta(\boldsymbol{x}_k)$ é usada para definir um limite baseado na bem conhecida desigualdade de Chebyshev para detecção de valores discrepantes (Saw et al., 1984). A

condição expressa por (4.9) define se a amostra real $\boldsymbol{x}_k$ está "$m\sigma$" longe da média, onde m é um valor constante que define quantos padrões desvios distantes da média de uma amostra de dados devem ser para ser considerada um *outlier*. Na maioria dos casos, o valor de m é definido como 3 (Bezerra et al., 2016; Kangin et al., 2016; Costa et al., 2016; Kangin and Angelov, 2015).

$$\zeta_k(\boldsymbol{x}) > \frac{m^2+1}{2k},\ m > 0 \tag{4.9}$$

Apesar de sua robustez para detecção de anomalias, o TEDA precisa ser adaptado para agrupar dados não ordenados. o MicroTEDAclus propõe um algoritmo de agrupamento baseado em *micro-clusters* TEDA criados usando restrições alteradas dinamicamente na variância, MicroTEDAclus é detalhado na próxima sub-seção.

4.2.2 MicroTEDAclus

Da mesma forma que o CEDAS (Hyde et al., 2017), o algoritmo de agrupamento MicroTEDAclus (Maia et al., 2020) é dividido em duas etapas. A primeira etapa realiza a atualização dos *micro-clusters*. Nesta etapa, criamos e atualizamos *micro-clusters* com base em uma versão restrita do algoritmo TEDA-Cloud (Bezerra et al., 2016). A segunda etapa realiza a atualização dos *macro-clusters*, em que os *macro-clusters* conectados são unidos em grupos. Finalmente, a densidade de cada *macro-cluster* é estimada e uma nova amostra de dados é atribuída ao *macro-cluster* para o qual possui o maior grau de adesão.

Definição de Limite de *Outlier*

A excentricidade $\xi(\boldsymbol{x}_k)$ é uma medida estatística da dissimilaridade entre amostras de dados, portanto, é mais confiável quanto mais dados (maiores de k) são usados em seu cálculo. Por exemplo, quando temos um pequeno número de amostras de dados, é difícil dizer com precisão que $\boldsymbol{x}_k$ está $m\sigma$ longe da média porque não temos uma boa estimativa da média e desvio padrão, portanto, é melhor um valor pequeno de m e, conseqüentemente, um limite de *outlier* mais restritivo. m foi definido anteriormente no TEDA como um valor constante que define quantos padrões desvios distantes da média de uma amostra de dados devem ser para ser considerada um *outlier*. Por outro lado, conforme o número de amostras de dados usadas para calcular o TEDA aumenta, pode-se ter mais certeza de dizer que $\boldsymbol{x}_k$ esta $m\sigma$ longe da média e do valor de m pode ser maior. Nesse sentido, um *micro-cluster* TEDA é uma versão ligeiramente modificada das nuvens de dados, apresentadas em Bezerra et al. (2016), que têm sua variância restringida de modo a evitar que um *cluster* cresça excessivamente e englobe todos os dados de *clusters* separados. Maia et al. (2020) aplicaram esta restrição no parâmetro m, para o limite de *outlier*, definida na Equação (4.10).

$$m(k) = \frac{3}{1 + e^{-0.007(k-100)}} \tag{4.10}$$

Assim, em vez de ter um valor constante de m, temos m em função de k. As características da função empírica $m(k)$ são indicadas a seguir:

1. $m(k)$ começa em 1, porque pode-se facilmente ver a partir de (4.7) que para $k = 2$ a excentricidade normalizada $\zeta(\boldsymbol{x}_2)$ sempre será igual a $0,5$, desta forma, o valor de m deve ser pelo menos igual a 1, caso contrário, cada ponto de dados será um cluster. Então, conforme mais dados são adquiridos, o valor de m cresce até 3, onde satura.

2. $m(k)$ satura em 3 quando $k \approx 1,000$. Em (4.9) o valor do limite começa em diferentes posições dependendo do valor de m. No entanto, quando k se torna alto, o limite para todos os valores de m é quase uma constante. Portanto, definimos $k \approx 1,000$ como o valor no qual a função empírica $m(k)$ satura perto de 3.

Os parâmetros de $m(k)$ foram projetados para obedecer às características acima. A Figura 4.2 mostra como $m(k)$ se comporta para diferentes valores de k. Embora a Figura 4.2 represente uma função contínua para facilitar a visualização, na prática $m(k)$ é discreto porque k representa timestamps discretos.

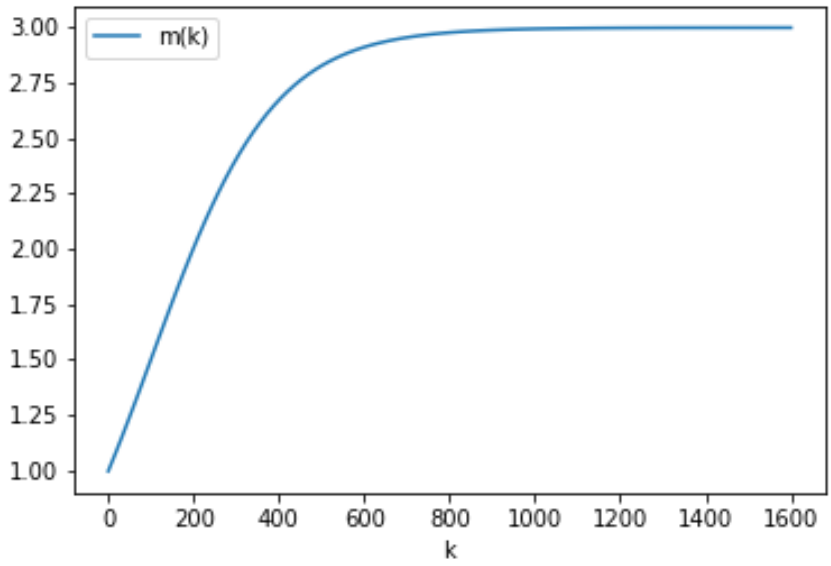


Figura 4.2: A função empírica $m(k)$.

Micro-Clusters

Um *micro-cluster* é uma estrutura de dados granular com um parâmetro *outlier* definido automaticamente m pela função $m(k)$. Seja $\mathfrak{m}_i$, $i = 1 \ldots n$ o conjunto de *micro-clusters*. Um protótipo de $\mathfrak{m}_i$ é definido pelos seguintes parâmetros, atualizados toda vez que um novo ponto de dados é amostrado:

- S_k^i: número de amostras;
- $\boldsymbol{\mu}_k^i$: centroide;
- $(\sigma_k^i)^2$: variança;
- $\xi^i(\boldsymbol{x}_k)$, $\zeta^i(\boldsymbol{x}_k)$: excentricidade e excentricidade normalizada;
- $\tau^i(\boldsymbol{x}_k)$, $t^i(\boldsymbol{x}_k)$: tipicidade e tipicidade normalizada;
- v_i: fator de vida;
- $D_k^i = \frac{1}{\zeta^i(\boldsymbol{x}_k)}$: densidade;
- $m_k^i(S_k^i)$: parâmetro de limite de *outlier*;

Quando a primeira amostra de dados $\boldsymbol{x}_1$ chega, o primeiro *micro-cluster* $\mathfrak{m}_1$ é criado com os seguintes parâmetros:

$$n = 1 \ , \ v^1 = 1 \ , S_1^1 = 1 \ , \ \boldsymbol{\mu}_1^1 = \boldsymbol{x}_1 \ , \ (\sigma_1^1)^2 = 0 \tag{4.11}$$

onde n é o número de *micro-clusters*. Observe que apenas alguns parâmetros são calculados quando $S_k^i = 1$ porque a tipicidade e a excentricidade só podem ser calculadas com 2 ou mais amostras de dados.

Quando uma nova amostra $\boldsymbol{x}_k$ chega no *timestamp* $k > 1$, o algoritmo calcula a tipicidade e excentricidade de $\boldsymbol{x}_k$ para todos os *micro-clusters* existentes pela Equação (4.6) e Equação (4.3), respectivamente e verifique se $\boldsymbol{x}_k$ é um *outlier* ou não:

$$\zeta^i(\boldsymbol{x}_k) > \frac{m_k^i(S_k^i)^2 + 1}{2S_k^i} \tag{4.12}$$

$$m_k^i(S_k^i) = \frac{3}{1 + e^{-0.007(S_k^i - 100)}} \tag{4.13}$$

Para o cálculo do TEDA, pelo menos 2 amostras de dados são necessárias, então usando a condição expressa por (4.9) com $m \geq 1$ a segunda amostra do *micro-cluster* $\mathfrak{m}_i$ nunca será considerada um *outlier*, mesmo que esteja distante da primeira $\mathfrak{m}_i$. Essa propriedade é indesejável quando as primeiras amostras estão distantes uma das outras e pode resultar em *micro-clusters* muito grandes que não modelam regiões densas adequadamente no espaço de dados. Assim, adicionamos um parâmetro r_0 para limitar a variância de cada *micro-cluster* quando $S_k^i = 2$ para evitar que um *cluster* cresça indefinidamente. Este parâmetro modifica a condição *outlier* expressa pela Equação (4.9) apenas quando $k = 2$:

$$\left[\zeta_2^i(\boldsymbol{x}_2) > \frac{(m^i(2))^2 + 1}{4}\right] \textbf{ AND } \left[(\sigma_2^i)^2 > r_0\right] \tag{4.14}$$

Em seguida, uma das duas condições pode ocorrer:

Condição 1: $\boldsymbol{x}_k$ não é um *outlier* para pelo menos um *micro-cluster*, **então** atualiza-se todos os *micro-clusters* para os quais esta condição vale.

$$
\begin{aligned}
S_k^i =& S_{k-1}^i + 1 \\
\boldsymbol{\mu}_k^i =& \frac{S_k^i - 1}{S_k^i} \boldsymbol{\mu}_{S_k^i - 1} + \frac{\boldsymbol{x}_k}{S_k^i} \\
(\sigma_k^i)^2 =& \frac{S_k^i - 1}{S_k^i} (\sigma_{k-1}^i)^2 + \frac{1}{S_k^i - 1} \left(\frac{2\|\boldsymbol{x}_k - \boldsymbol{\mu}_k^i\|}{d} \right)^2 \\
\xi^i(\boldsymbol{x}_k) =& \frac{1}{S_k^i} + \frac{2(\boldsymbol{\mu}_k^i - \boldsymbol{x}_k)^T (\boldsymbol{\mu}_k^i - \boldsymbol{x}_k)}{S_k^i (\sigma_k^i)^2 d} \\
\upsilon_k^i =& 1
\end{aligned}
\tag{4.15}
$$

onde d é a dimensionalidade do conjunto de dados.

Condição 2: $\boldsymbol{x}_k$ é um *outlier* para todos os *micro-clusters* existentes **então** crie um *micro-cluster*.

$$
n = n + 1;\ \upsilon_k^n = 1\ ,\ S_k^n = 1;\ \boldsymbol{\mu}_k^n = \boldsymbol{x}_k;\ (\sigma_k^n)^2 = 0 \tag{4.16}
$$

O procedimento de atualização do *micro-cluster* é detalhado no Algoritmo 1.

Os *micro-clusters* para o conjunto de dados da Figura 4.3(a) são ilustrados na Figura 4.3(b).

Para todos os *micro-clusters* que não foram atualizados é usado a remoção descrita em Hyde et al. (2017), no qual é usado um decaimento linear simples (Equação 4.17), onde ρ é definido pelo dinamismo do modelo.

$$
\upsilon_k^i = \upsilon_{k-1}^i - \rho \tag{4.17}
$$

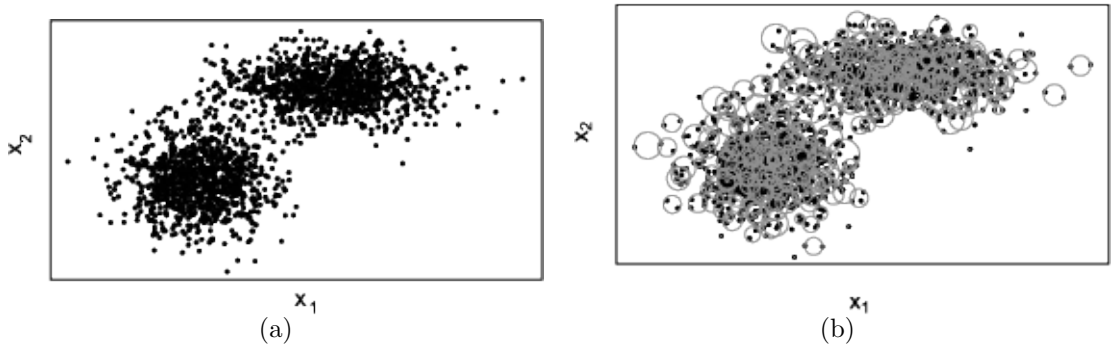
Macro-Clusters

Com os *micro-clusters* para o *timestamp* k, o algoritmo de atualização dos *macro-clusters* encontra grupos de *micro-clusters* que se cruzam. Este procedimento é semelhante à forma como o algoritmo CEDAS Hyde et al. (2017) encontra *macro-clusters*, atualizando um gráfico de interseção e agrupando os *micro-clusters* desconectados. O grafo de interseção é gerado a partir da matriz de adjacência cujas dimensões são iguais ao número de *micro-clusters*. Cada elemento na matriz é definido como 1 se dois *micro-clusters* se cruzarem e 0 caso contrário. A condição para verificar se dois *micro-clusters* se cruzam é definida como:

$$
dist(\boldsymbol{\mu}_k^i, \boldsymbol{\mu}_k^j) < 2(\sigma_k^i + \sigma_k^j)\ ,\ \forall i \neq j \tag{4.18}
$$

Algoritmo 1: Atualização *Micro-cluster* baseado na distância euclidiana

Input: $\boldsymbol{x}_k$, r_0
Output: $\mathfrak{m}_i$, $i = 1, 2, \ldots, n$
begin
  while *Nova amostra k disponível* **do**
    if $k == 1$ **then**
      Defina $\mathfrak{m}_1$ parâmetros como na Equação 4.11;
    else
      $flag \leftarrow true$;
      for $i = 1 : n$ **do**
        $\mathfrak{m} \leftarrow \mathfrak{m}_i$;
        if $S_k^i == 2$ **then**
          $outlier \leftarrow$ condição Equação 4.14;
        else
          $outlier \leftarrow$ condição Equação 4.12;
        end
        if $outlier == false$ **then**
          Atualiza $\mathfrak{m}_i$ com Equação 4.15;
          $flag \leftarrow false$;
        end
      end
      if $flag == true$ **then**
        Criar um novo *micro-cluster* com Equação 4.16;
      end
    end
  end
end



(a) (b)

Figura 4.3: (a) Conjunto de dados (b) Micro-clusters.

Usando a condição acima para um conjunto de dados sobrepostos, como na Figura 4.3, é claro que todos os *micro-clusters* podem ser conectados criando apenas um grande *macro-cluster*. Para evitar esse problema, aplicamos um filtro para ativar ou não os *micro-clusters* com base em sua densidade. Seja $\mathfrak{M}_j = \{\mathfrak{m}_1^j, \mathfrak{m}_2^j, \ldots, \mathfrak{m}_l^j\},\ j = 1, 2, \ldots, N\ jth$ *macro-cluster* composto por um conjunto de l *micro-clusters* conectados. O conjunto de *micro-clusters* ativos do *macro-cluster* $\mathfrak{M}_j$ são aqueles para os quais a densidade D_k^l é maior ou igual à densidade média calculada sobre todos os *micro-clusters* que pertencem a $\mathfrak{M}_j$:

$$active(\mathfrak{m}_l^j) = D_k^l \geq mean(D_k^l),\ l = 1, \ldots, |\mathfrak{M}_j| \tag{4.19}$$

O efeito da aplicação deste filtro é que os *micro-clusters* em regiões de baixa densidade serão inativados, enquanto aqueles em regiões de alta densidade ficarão ativos. Em outras palavras, presumimos que as regiões de baixa densidade indicam separação entre *macro-clusters* sobrepostos, portanto, o filtro removerá dos cálculos de *macro-clusters* os *micro-clusters* que não representam o padrão principal no *timestamp* k. Um *micro-clusters* desativado se tornará ativo quando novas amostras são recebidas, e um *macro-clusters* se ativará novamente quando a sua densidade dos seus *micro-clusters* aumenta e passa o limiar do promédio da densidade de todos os *macro-clusters* (Equação 4.19), a partir de novas amostras.

Finalmente, a estimativa de densidade de cada *macro-cluster* é calculada como uma soma das tipicidades normalizadas de seus *micro-clusters* ativos, ponderada por sua densidade normalizada, como um modelo de mistura de densidade, conforme discutido em Angelov (2014a):

$$\mathcal{T}_j(\boldsymbol{x}_k) = \sum_{l \in \mathfrak{M}_j} w_k^l t_k^l(\boldsymbol{x}_k) \tag{4.20}$$

$$w_k^l = \frac{D_k^l}{\sum\limits_{l \in \mathfrak{M}_j} D_k^l} \tag{4.21}$$

Um novo ponto de dados $\boldsymbol{x}_k$ é atribuído ao *macro-cluster* para o qual ele tem a maior mistura de pontuação de tipicidade $\mathcal{T}_j(\boldsymbol{x}_k)$. O procedimento detalhado para calcular *macro-clusters* é apresentado em Algoritmo 2.

Um exemplo de como a definição do *macro-cluster* funciona é ilustrado na Figura 4.4. A Figura 4.4(a) mostra os *micro-clusters* dispostos em dois *macro-cluster* diferentes (azul e vermelho). Na Figura 4.4(b), as regiões de densidade são destacadas de acordo com a mistura de tipicidade. A Figura 4.4(c) mostra as estruturas finais do *cluster* identificadas pelo algoritmo no conjunto de dados.

Portanto, a cada novo ponto de dados $\boldsymbol{x}_k$ MicroTEDAclus atualiza os *micro-clusters*, recalcula os *macro-clusters* e retorna o cluster para o qual $\boldsymbol{x}_k$ é mais compatível com relação à mistura de tipicidade $\mathcal{T}_j(\boldsymbol{x}_k)$.

Algoritmo 2: Atualização de *Macro-clusters* baseado na distância euclidiana

Input: $\boldsymbol{x}_k$, $\mathfrak{m}$

Output: Grau de pertinência de $\boldsymbol{x}_k$ para cada *macro-cluster*

begin

  while *Nova amostra disponível* **do**

    $\mathfrak{M} \leftarrow$ Grupo de *micro-clusters* que se interceptam conforme a condição expressa na Equação 4.18;

    Encontre o conjunto de *micro-clusters* de cada *macro-cluster* $\mathfrak{M}_j,\ j = 1, 2, \ldots, N$ de acordo com a Equação 4.19;

    Calcule $\mathcal{T}_j(\boldsymbol{x}_k)$ para $j = 1, 2, \ldots N$ de acordo com a Equação 4.20;

    Atribua $\boldsymbol{x}_k$ ao cluster j que tem o maior $\mathcal{T}_j(\boldsymbol{x}_k)$;

  end

end

4.3 Metodologia

Nesse trabalho é proposta uma metodologia de agrupamento incremental de trajetórias e com uma etapa de detecção de *outliers*, a qual pretende resolver os dois problemas na logística de *e-commerce* expostos na introdução do capítulo. A metodologia é composta por duas etapas, o pré-processamento de dados que transforma cada rota em um vetor binário, e a implementação do MicroTEDAclus usando a similaridade cosseno.

4.3.1 Etapa de pré-processamento

No problema exposto, trajetórias são definidas pela localização dos pacotes p entregues em cada uma das rotas T construídas pelo algoritmo de roteirização nos CDs. A etapa de pré-processamento exposta nessa metodologia é a mesma utilizada no capítulo anterior (Seção 3.2.2), em que cada sequência de pacotes é mapeada em uma sequência de áreas geográficas de diferentes tamanhos denominados células. Da mesma forma que os experimentos realizados no capítulo anterior, nessa etapa usamos a biblioteca de geometria S2, do Google. Portanto, cada rota construída no CD com n pacotes é mapeada em um vetor binário de dimensão fixa m, onde m representa a quantidade de células S2 selecionadas na região. O valor no vetor é 1

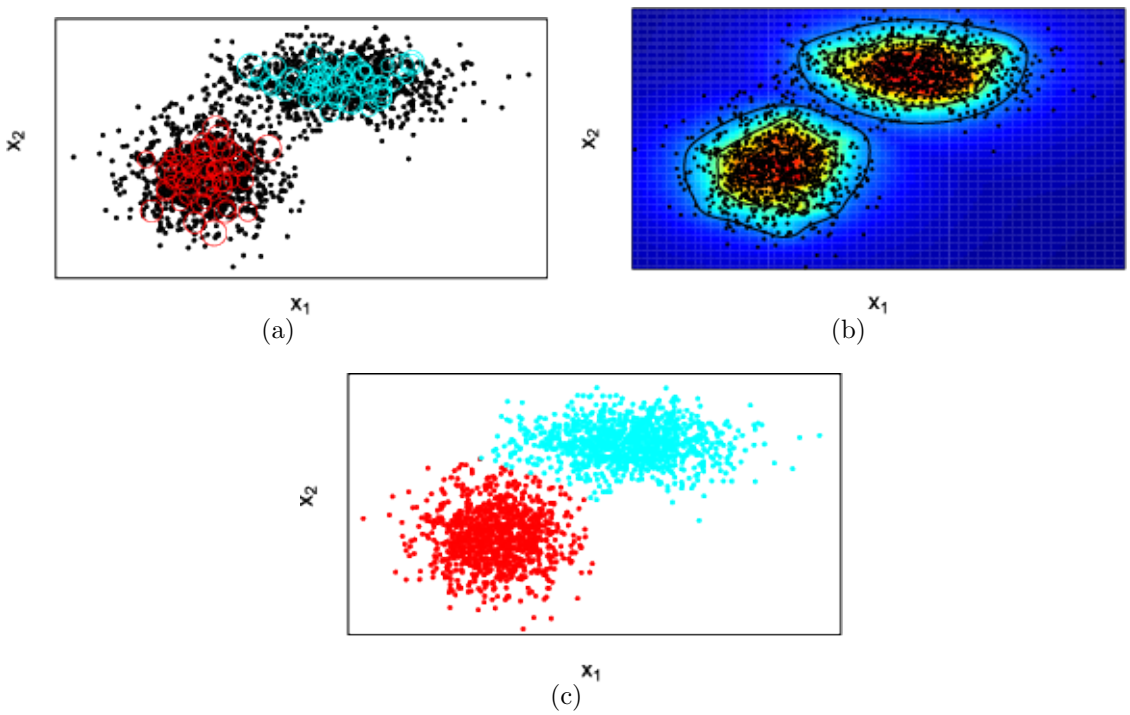


Figura 4.4: (a) *Macro-Clusters*. (b) Mistura de tipicidade. (c) Assignação do *cluster*.

se tem pelo menos um pacote entregue na região S2 que representa essa dimensão, ou 0 caso contrário.

Uma rota de *last-mile* realizada por um mensageiro pode ser representada pela sequência de pontos da localização dos pacotes entregues, descrita na trajetória T_1 (Equação 4.22), mostrada pela linha azul na Figura 3.3. Aplicando a etapa de pre-processamento exposta na Seção 3.2.2, podemos projetar T_1 nos espaços em regiões S2 A, B e C da Figura 3.3. Dessa forma, T_1 pode ser representado pelo conjunto de regiões S2 que contem as localizações de entrega dos pacotes, e pelo conjunto binário que representa espaço total de da cidade, cada uma representada por T_{1_A}, T_{1_B} e T_{1_C} na Tabela 4.1. O valor no vetor é 1 se pelo menos um pacote é entregue na região, caso contrario é 0. Finalmente, cada rota é representada pela concatenação destes vetores binarias, Equação 4.23.

$$T_1 = \{p_{11}, p_{12}, p_{13}, p_{14}, p_{15}, p_{16}\} \tag{4.22}$$

$$T_1 = \{T_{1_A}, T_{1_B}, T_{1_C}\} = \{1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, ..., 0\} \tag{4.23}$$

Tabela 4.1: Mapeamento da trajetória T_1 nos espaços em regiões S2 A, B e C da Figura 3.3.

	Espaço S2	T_1 projetado no espaço S2	Vetor Binario de T_1 no espaço S2
T_{1_A}	$\{A_1, A_2, A_3, A_4, A_5, A_6\}$	$\{A_1, A_4, A_5, A_6, A_3\}$	$\{1, 0, 0, 1, 1, 1, 1\}$
T_{1_B}	$\{B_{00}, B_{01}, B_{02}, B_{03}, ..., B_{24}\}$	$\{B_{01}, B_{11}, B_{12}, B_{13}, B_{14}, B_{05}\}$	$\{0, 1, 0, 0, ..., 0\}$
T_{1_A}	$\{C_{00}, C_{01}, C_{02}, C_{03}, ..., C_{48}\}$	$\{C_{01}, C_{22}, C_{24}, C_{26}, C_{28}, C_{19}\}$	$\{0, 1, 0, 0, ..., 0\}$

4.3.2 MicroTEDAclus

O MicroTEDAclus (Maia et al., 2020) foi proposto inicialmente baseado na distância euclidiana como métrica de similaridade nos dados. Porém, esta métrica de similaridade não pode ser usada na nossa abordagem, devido à projeção das trajetórias em um espaço desconexo representadas por regiões S2, por exemplo, T_1 na Equação 4.23. Por este motivo, propomos o uso da similaridade cosseno para o agrupamento das trajetórias. Assim, foi necessário realizar algumas adaptações ao algoritmo original do MicroTEDAclus.

A Equação 4.12, usada para a detecção de *outlier*, é baseada no cálculo de excentricidade fundamentada na distância euclidiana. Por essa razão, foi adaptada esta condição de detecção de uma rota anômala para a condição definida na Equação 4.24. Angelov (2014b) expõe a análise detalhada da detecção de *outliers* proposta pelo TEDA.

$$\left[(\boldsymbol{x}_k - \boldsymbol{\mu}_k^i)^2 > (m_k^i)^2 \sigma_2^i\right] \tag{4.24}$$

Dado que $\boldsymbol{x}_k$ não é um *outlier* para pelo menos um *micro-cluster*, a **Condição 1** da Equação 4.15 foi modificada para 4.25.

$$\begin{aligned}
S_k^i =& S_{k-1}^i + 1 \\
\boldsymbol{\mu}_k^i =& \frac{S_k^i - 1}{S_k^i} \boldsymbol{\mu}_{S_k^i - 1} + \frac{1}{S_k^i} \frac{\boldsymbol{x}_k}{\|\boldsymbol{x}_k\|}; \|\boldsymbol{x}_k\| \neq 0 \\
(\sigma_k^i)^2 =& \frac{S_k^i - 1}{S_k^i} (\sigma_{k-1}^i)^2 + \frac{1}{S_k^i - 1} \left(\frac{2\|\boldsymbol{x}_k - \boldsymbol{\mu}_k^i\|}{d} \right)^2 \\
\xi^i(\boldsymbol{x}_k) =& \frac{1 - \boldsymbol{\mu}_k^i \frac{\boldsymbol{x}_k}{\|\boldsymbol{x}_k\|}}{1 + k \boldsymbol{\mu}_k^i \boldsymbol{\mu}_k^i}; \|\boldsymbol{x}_k\| \neq 0
\end{aligned} \tag{4.25}$$

Por consequência, os Algoritmos 3 e 4 detalham a atualização dos *macro-clusters* e *micro-*

clusters, respetivamente.

Algoritmo 3: Atualização *Micro-cluster* baseado na similaridade cosseno

Input: $\boldsymbol{x}_k$, r_0
Output: $\mathfrak{m}_i,\ i = 1, 2, \ldots, n$
begin
  while *Nova amostra disponível* **do**
    if $k == 1$ **then**
      Defina $\mathfrak{m}_1$ parâmetros como na Equação 4.11;
    else
      $flag \leftarrow true$;
      for $i = 1 : n$ **do**
        $\mathfrak{m} \leftarrow \mathfrak{m}_i$;
        $outlier \leftarrow$ condição Equação 4.24;
        if $outlier == false$ **then**
          Atualiza $\mathfrak{m}_i$ com Equação 4.25;
          $flag \leftarrow false$;
        end
      end
      if $flag == true$ **then**
        Criar um novo Create a new *micro-cluster* com Equação 4.16;
      end
    end
  end
end

Algoritmo 4: Atualização de *Macro-clusters* baseado na similaridade cosseno

Input: $\boldsymbol{x}_k$, $\mathfrak{m}$
Output: Grau de pertinência de $\boldsymbol{x}_k$ para cada *macro-cluster*
begin
  while *Nova amostra disponível* **do**
    $\mathfrak{M} \leftarrow$ Grupo de *micro-clusters* que se interceptam conforme a condição expressa na Equação 4.18;
    Encontre o conjunto de *micro-clusters* de cada *macro-cluster* $\mathfrak{M}_j,\ j = 1, 2, \ldots, N$ de acordo com a Equação 4.19;
    Calcule $\mathcal{T}_j(\boldsymbol{x}_k)$ para $j = 1, 2, \ldots N$ de acordo com a Equação 4.20;
    Atribua $\boldsymbol{x}_k$ ao cluster j que tem o maior $\mathcal{T}_j(\boldsymbol{x}_k)$;
  end
end

4.4 Resultados experimentais

Nesse trabalho apresentamos a adaptação do algoritmo de agrupamento evolutivo denominado MicroTEDAclus aplicado em trajetórias. Os experimentos foram conduzidos em problemas de logística de *last-mile* no e-commerce, o primeiro o gerenciamento de rotas entre os mensageiros e o segundo detecção de *outliers* nas rotas históricas. A avaliação do desempenho escalabilidade e complexidade computacional do MicroTEDAclus já foi realizada em (Maia et al., 2020). Por este motivo, os experimentos realizados foram análise dos resultados obtidos na implementação em dados de trajetórias, avaliando os resultados sobre os problemas na logística de *e-commerce*. Os experimentos foram realizados em dados de trajetória reais, baseado no modelo de entregas da Loggi. As seguintes subseções apresentam o conjunto de dados usado nos experimentos, o primeiro experimento que demostra o uso do algoritmo proposto para gerenciamento de rotas no *last-mile* e o segundo teste onde analisamos a detecção de *outliers* nas rotas construídas.

4.4.1 *Dataset*

Nesses experimentos, usamos o mesmo *dataset* dos experimentos realizados no capítulo anterior (Seção 3.3.1), estes dados foram fornecidos pela Loggi. No experimento anterior, os pacotes do *dataset* de treinamento foram roteirizados em 6434 rotas aplicando o VRP estático em lotes. Para isso, foi usada a heurística construtiva *Path Most Constrained Arc* e metaheurística *guided local search* da biblioteca OR-Tools[1]. Finalmente, do *dataset* de rotas sintético de 6434, foram utilizadas 4182 referentes aos pacotes de dois ECs para avaliar o MicroTEDAclus em trajetórias. Além disso, o parâmetro r_0 foi definido como 0.15 e ρ como 0.001. Estes parâmetros foram calculados de forma empírica, o valor do r_0 depende da distribuição espacial das rotas, e nosso caso deve ser configurado dependendo da cidade e da densidade de pacotes na região. O fator de esquecimento depende do dinamismo do mercado.

4.4.2 Gerenciamento de Rotas no *Last-mile*

No modelo de entrega proposto pela Loggi as rotas de *Last-mile* são criadas no DC e precisam ser expedidas no EC. Por este motivo, precisa-se de um modelo capaz de atribuir as rotas entre os mensageiros. Além disso, nos últimos anos o mercado de e-commerce tem apresentado um crescimento exponencial, resultando em um dinamismo na distribuição de pacotes e o volume de pacotes que precisam ser entregues. Por esse motivo, os experimentos são realizados em duas

[1]OR-Tools (7.2) Google (acessado em 15 de dezembro de 2020), `https://developers.google.com/optimization/`

etapas. A primeira etapa do experimento consiste em analisar visualmente grupos gerados pelo algoritmo MicroTEDAclus, baseado nos estudos realizados por Huang et al. (2018), procurando consistência espacial dos grupos para aumentar a probabilidade dos pacotes serem entregues. A segunda etapa pretende explorar a adaptação dos grupos quando existem mudanças na abrangência dos centros de expedição.

No primeiro experimento, foi aplicado MicroTEDAclus em 3000 rotas das 4182 consideradas. O MicroTEDAclus identificou 323 *micro-clusters* e, no conjunto de *micro-clusters* foram encontrados 20 *macro-clusters*. Baseado na densidade dos *macro-clusters*, 110 dos 323 *micro-clusters* foram ativos. Na Figura 4.5 foram plotados 9 *macro-clusters* dos 20 encontrados, o motivo de não plotar todos os *macro-clusters* foi porque a visualização de 20 cores diferentes não ficou entendível. Cada *macro-clusters* é representado por uma amostra dos pacotes das rotas dos *micro-clusters* ativos de cada *macro-clusters*. Na figura, observa-se que existe consistência espacial nos grupos, validando a hipótese que as trajetórias separadas nos grupos pelo algoritmo podem ser usadas para selecionar o mensageiro, o qual será ofertada a rota. No entanto, dividir a cidade em 20 mensageiros pode não ser possível devido ao volume diário de pacotes entregues na cidade, portanto, podemos analisar o comportamento dos *micro-clusters* para fazer uma classificação mais granular das rotas entre os mensageiros.

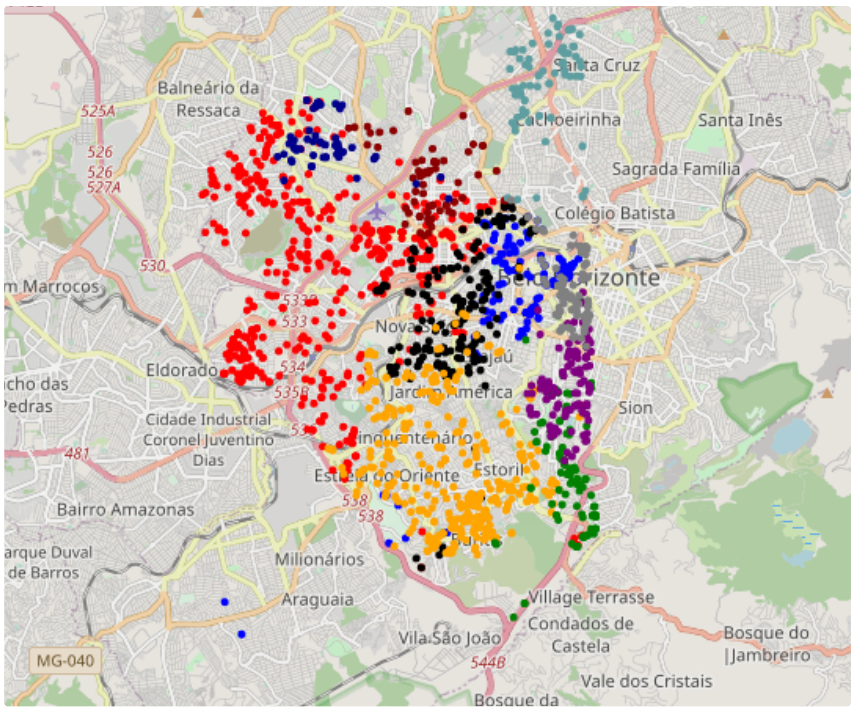


Figura 4.5: *Macro-clusters* gerados pelas rotas de *last-mile* em Belo Horizonte, Brasil.

Cada *macro-clusters* é uma união de um conjunto de *micro-clusters* ativos, para uma análise mais detalhada do comportamento do MicroTEDAclus foram plotados os *micro-clusters* que compõem um *macro-cluster* na Figura 4.6. Diferente à Figura 4.5, os *micro-clusters* foram representados pelas rotas agrupadas nos *micro-clusters*. O desenho das rotas foi realizado usando o serviço do OSRM[2]. Na Figura 4.6 a. são apresentados os três micro-clusters de um *macro-cluster*, seguidamente, para uma melhor visualização dos *micro-cluster* estes foram plotados separadamente nas Figuras 4.6 b, 4.6 c e 4.6 d. Como se esperava, existe uma sobreposição entre os *micro-cluster* que contem o *macro-cluster*, da mesma forma que na Figura 4.4(c) a.

Os resultados dos *macro-cluster* e *micro-cluster*, apresentados nas Figuras 4.5 e 4.6, respectivamente evidenciam, a possibilidade de usar os grupos para atribuir as rotas construídas no DC entre mensageiros, oferecendo independência operacional entre as instalações, desacoplando a etapa de criação de *last-mile* da etapa de alocação de rotas a mensageiros. Além disso, a atribuição baseada nos grupos resultantes do agrupamento permite atribuir mensageiros em zonas recorrentes sem precisar de intervenção manual de um especialista, isto representa redução de tempo na expedição e redução de custos.

No segundo experimento, queremos validar a adaptação dos grupos encontrados pelo MicroTEDAclus quando existem mudanças na abrangência da agência. Para esse experimento, foram criados os grupos de um EC e começamos enviar rotas de um EC próximo. Os resultados são apresentados na Figura 4.7.Na Figura 4.7a são apresentados 20 *macro-clusters* construídos a partir de 1500 rotas de um EC. Seguidamente, são inseridas sequencialmente lotes de 400 trajetórias de um EC vizinho (Figuras 4.7b, 4.7c e 4.7d), até chegar no cenário da e 4.7d, em que todos os *macro-clusters* pertencem ao segundo EC.

Os resultados do segundo experimento demostraram o comportamento incremental do MicroTEDAclus, os *macro-cluster* conseguiram-se adaptar as mudanças de comportamento nas trajetórias devido às atualizações na abrangência da cidade. Em outras palavras, conseguimos ter uma adaptação das áreas para manter o casamento rota-mensageiro sem precisar da intervenção de um especialista.

4.4.3 Detecção de *outliers* no *Last-mile*

Os processos de separação e consolidação de pacotes, na maioria dos casos, são realizados manualmente. Posto isto, o modelo é susceptível a erros, quando pacotes são colocados erradamente entre as rotas. Identificar estes erros antes da expedição reflete diretamente no custo, já que podem ser reprocessados e direcionado na rota correta de entrega. Por esta razão, direcionamos os experimentos para explorar a etapa de detecção de *outliers* do MicroTEDAclus para dois casos em específico, o primeiro misturar pacotes entre rotas, e o segundo misturar rotas entre centros de expedição.

[2]Open Source Routing Machine (acessado em 09 de novembro de 2021), `http://project-osrm.org/`

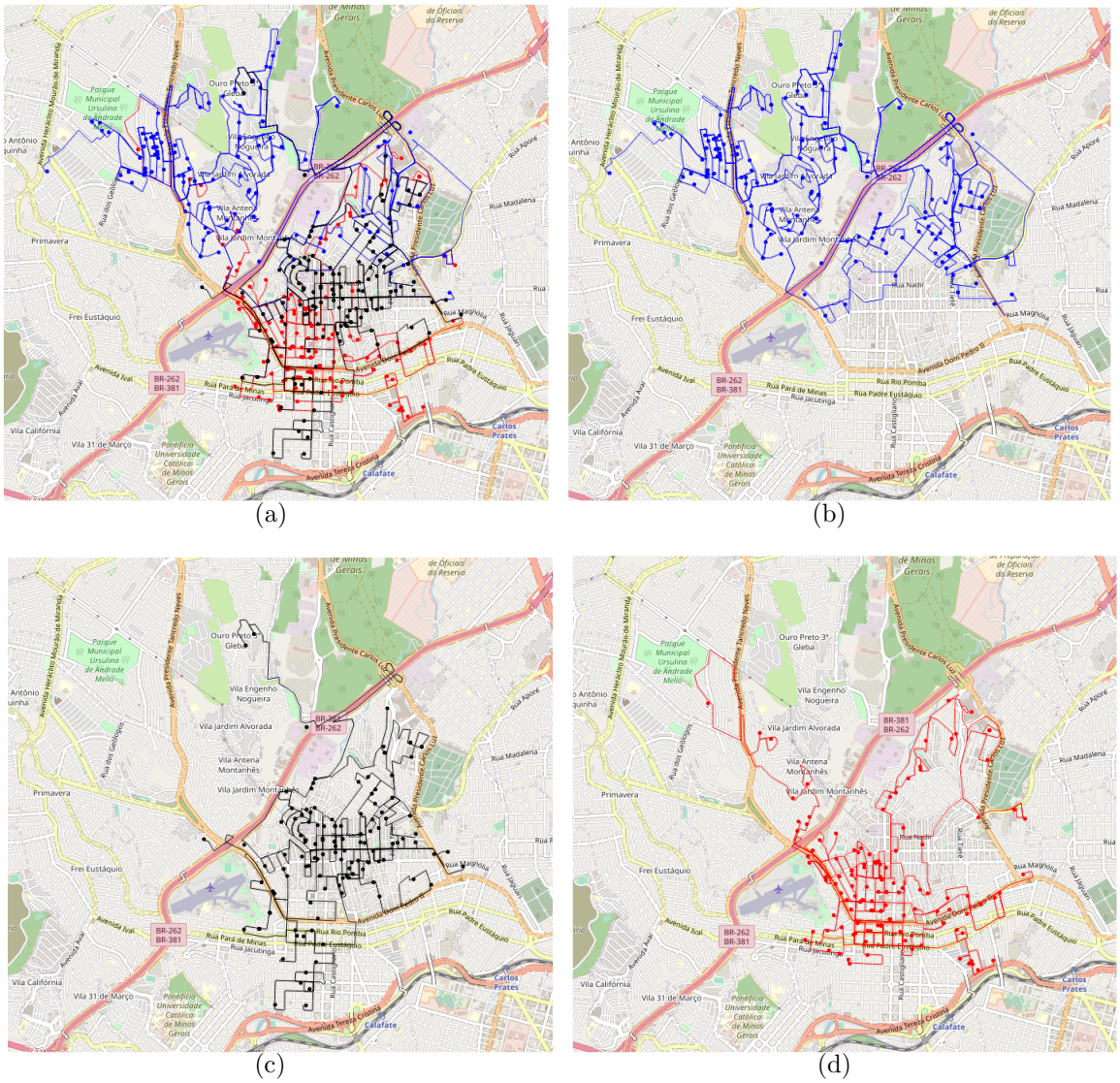


Figura 4.6: *Micro-clusters* de um *macro-cluster* gerado pelas rotas de *last-mile* em Belo Horizonte, Brasil. (a) Todos os *micro-clusters* do *macro-clusters*. (b) *Micro-cluster* 1. (c) *Micro-cluster* 2. (d) *Micro-cluster* 3.

Para cada rota de entrada no MicroTEDAclus, este aplica uma etapa de detecção de *outlier*, definida na Seção 4.2.2. Portanto, determina-se como *outlier* toda rota que estiver m vezes do desvio padrão de todos os *micro-cluster* existentes, onde m é calculado pela Equação 4.10. Assim, quando um conjunto de rotas *outlier* começa ser frequente, estas viram um *micro-cluster*

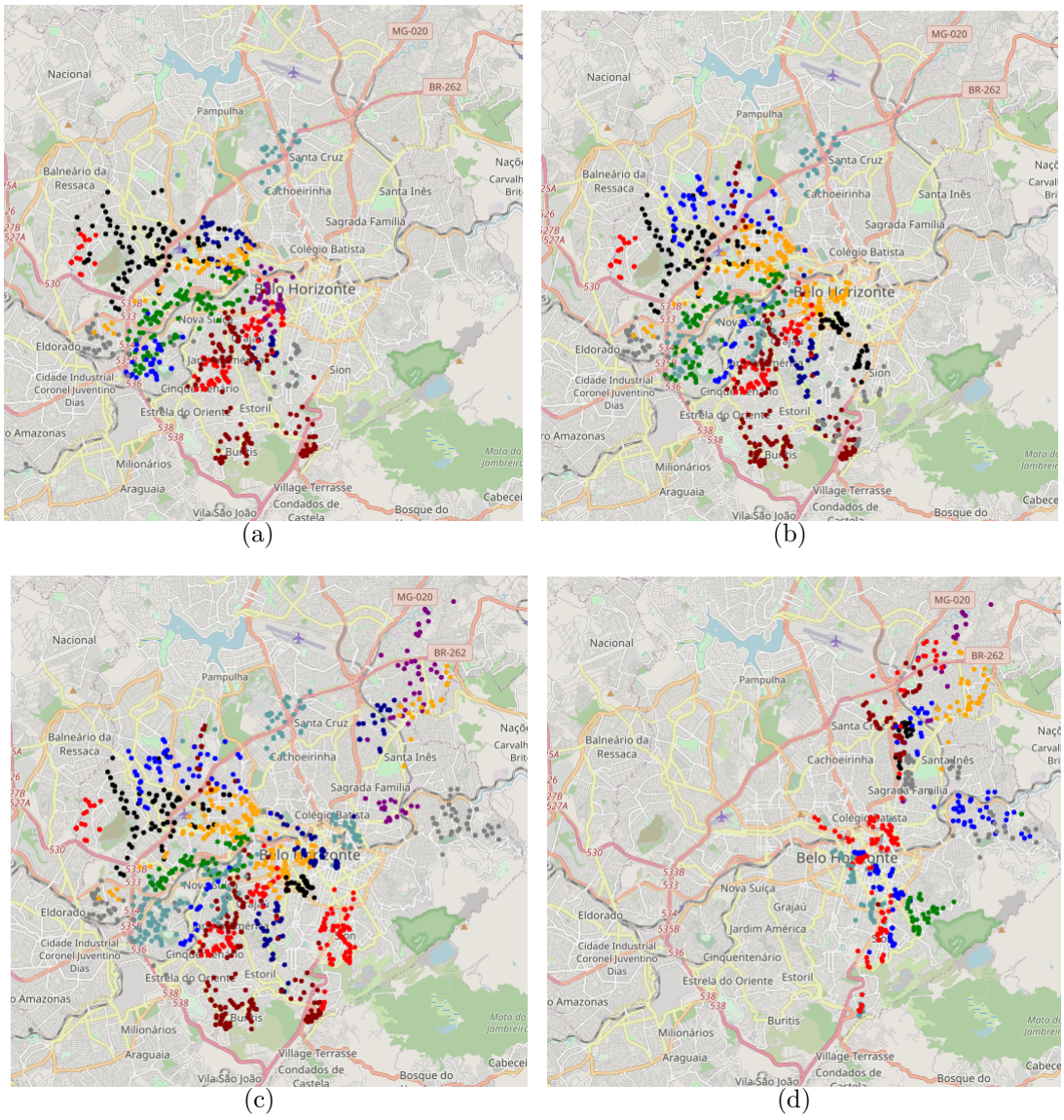


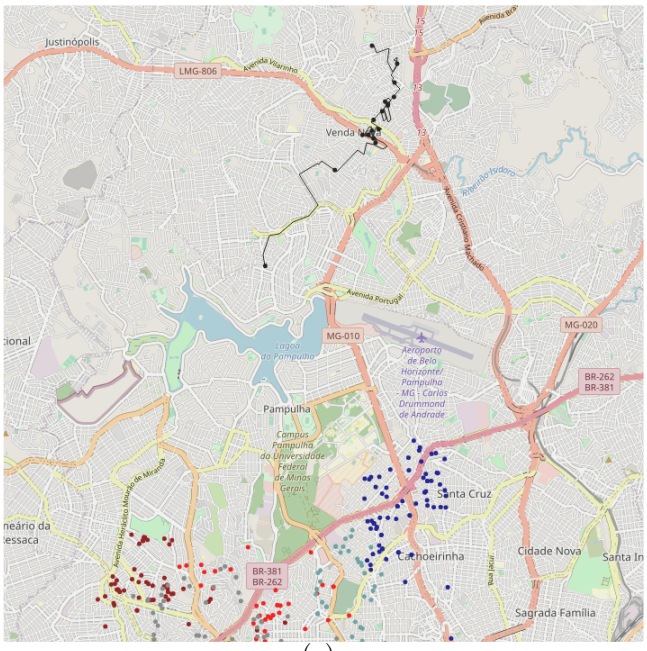
Figura 4.7: Dinamismo do MicroTEDAclus em rotas de *las-mile* em Belo Horizonte, Brasil. Adaptação dos *Macro-clusters* construídos pelas rotas expedidas em um EC, para as rotas construídas no EC vizinho, a sequência da adaptação é (a), (b), (c) e (d).

e dessa, o *MicroTEDAclus* consegue diferenciar a detecção de rotas anômalas e de mudança de abrangência.

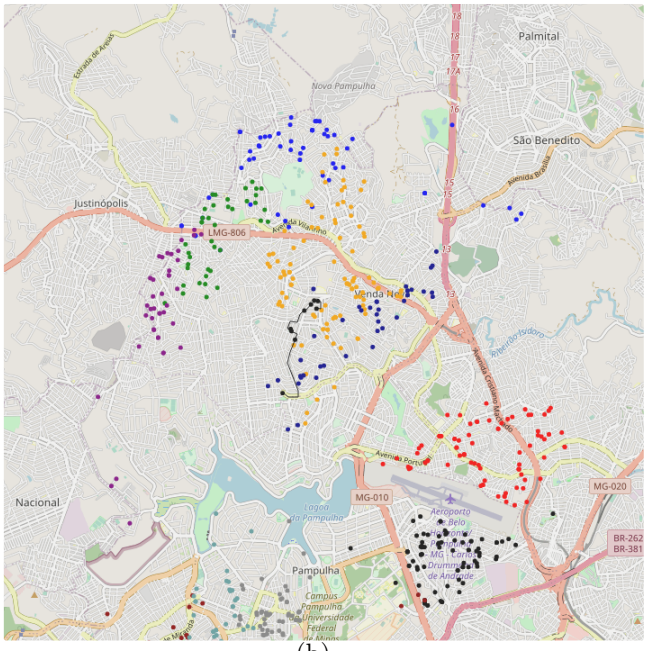
No primeiro experimento, queremos validar se o algoritmo consegue identificar quando uma

rota é enviada de forma errada para um EC, nesse caso a rota não estaria na mesma região dos *macro-cluster* do EC. Para isso, implementamos o MicroTEDAclus para um conjunto de trajetórias de um dos EC, e intencionalmente, inserimos uma rota de outro EC para validar que o modelo conseguisse identificar que essa rota é um *outlier*. Na Figura 4.8a, é apresentado o resultado do experimento, o modelo foi capaz de identificar como *outlier* a rota preta, que não faz parte do conjunto de rotas e pacotes da região do EC. Nesse caso, ela pode ser direcionada para o EC certo e expedir ela com um custo menor.

O segundo experimento consiste em identificar cenários em que operador mistura pacotes entre as rotas do mesmo EC. Nesse caso, deve ser gerada uma rota anômala que percorre dois *macro-clusters*. A Figura 4.8b apresenta um exemplo do caso exposto. A rota representada pela linha preta foi identificada como *outlier* pois não faz parte de nenhum dos grupos identificados anteriormente. Como resultado, conseguimos demostrar dois casos pontuais onde a implementação do MicroTEDAclus aplicado em rotas de *last-mile* conseguiria identificar comportamentos anômalos na operação. Isto impacta diretamente em redução de custo da empresa, já que estes pacotes podem ser identificados e reprocessados antes de ser expedidos.



(a)



(b)

Figura 4.8: Tipos de *outliers* encontrados pelo MicroTEDAclus em trajetórias (a) Rota pertence à abrangência de outro EC. (b) Rota construída entre dois *macro-clusters*.

Capítulo 5

Conclusões e Perspectivas

O crescimento no mercado de *e-commerce*, o aumento da penetração da internet e o fácil acesso ao pagamento *online* tem gerado desafios na logística da entrega. Os problemas consistem em entregar pacotes rapidamente com custo baixo, conseguido-se adaptar as mudanças do mercado e do comportamento da população. Nesse trabalho, foram realizadas duas propostas que usam técnicas de mineração de dados focadas em problemas de *last-mile*.

A primeira proposta, apresentada no Capítulo 3, é uma metodologia para a solução do CVRP dinâmico e estocástico com foco em problemas logísticos reais de *e-commerce*. Nesse problema, os pacotes a serem entregues chegam dinamicamente e precisam ser separados rapidamente sem armazenamento. Modelamos o problema a partir desses requisitos como um fluxo de dados de pacotes de entrada e rotas de saída. Nossa solução pode lidar com cenários de *Big Data*, permitindo a separação de muitos pacotes de forma eficiente. O problema foi modelado em um *Warehouse system*, onde os pacotes são divididos dinamicamente entre uma quantidade fixa de *unit loads*. Os *unit loads* são fechados com base em critérios de operação, comumente realizados por empresas de logística, principalmente *e-commerce*. Além disso, o problema foi modelado como um sistema multiagente que traz vantagens como a autonomia dos agentes, a capacidade de aumentar a eficiência computacional por meio de computação paralela e a possibilidade de usar um ambiente distribuído. Cada agente representa um *unit loads* e consegue resolver problemas como as restrições de capacidade e a necessidade de fechar cargas unitárias de forma independente. O algoritmo proposto utiliza padrões territoriais extraídos de técnicas de mineração de dados de trajetória para melhorar a aposta dos agentes, combinando informações estocásticas ao modelo. O método proposto para a extração desses padrões usa uma melhoria do FP-*growth*, possibilitando a distribuição do processamento nas máquinas, permitindo-lhes trabalhar de forma eficiente com grandes volumes de dados. Até o momento, nenhum trabalho foi encontrado na literatura que modele uma heurística usando

técnicas de mineração de dados e um sistema multiagente para resolver o problema DCVRP com consumidores estocásticos. Assim, a principal contribuição deste método está em lidar com um problema logístico de grande escala, relatando uma solução inovadora que leva em consideração padrões de trajetória extraídos de dados históricos para que seja possível absorver as limitações impostas por tais cenários; que estão relacionados ao grande número de pacotes e ao custo operacional do armazém.

A segunda metodologia proposta, é uma prova de conceito do uso de um modelo de agrupamento incremental para solucionar o problema de gerenciamento de trajetórias de *last-mile* entre mensageiros e detecção de rotas anômalas. Os resultados mostraram que MicroTEDAclus consegue criar grupos que podem ser usados para atribuir as rotas de *last-mile* entre os mensageiros. Além disso, foi avaliado o dinamismo do algoritmo para se adaptar as mudanças de abrangência que podem existir devido ao aumento de pacotes entregues, mudança nos embarcadores, ou clientes atendidos. Finalmente, os resultados dos experimentos realizados na detecção de *outlier* demostraram que o algoritmo é capaz de identificar erros operacionais representados na criação anômala de rotas. A identificação apropriada destes erros antes da expedição dos pacotes representa uma redução no custo de entrega, já que estes podem ser reprocessados e enviados em rotas mais convenientes. O algoritmo armazena a informação em estrutura de dados denominadas *micro-clusters* e *macro-clustes* que permite representar grandes volumes de trajetórias de forma compacta. Isso pode ser visto como uma vantagem, principalmente para o caso de uso da logística do Brasil, a qual tem mais de 5000 cidades e uma população maior de 200 milhões de habitantes. Dessa forma, a principal contribuição da segunda metodologia é expor o uso da adaptação de uma técnica de agrupamento incremental para resolver lacunas emergentes na logística *e-commerce*. O uso da proposta representa uma diminuição direta no custo da entrega de *last-mile*, já que a seleção de rotas em zonas recurrentes para os mensageiros, faz que eles virem especialistas na região, e por tanto aumenta a probabilidade do pacote ser entregue, além disso, o algoritmo se acopla em modelos de entrega que a roteirização e a expedição de rotas são realizadas em instalações diferentes, eliminando a dependência entre estas duas tarefas.

5.1 Propostas de Continuidade

Nesse trabalho foi explorado o uso de técnicas de mineração de dados em dois problemas na logística de *e-commerce* do Brasil. O primeiro na roteirização de *last-mile* e o segundo no gerenciamento e detecção rotas anômalas no *last-mile*. No entanto, existem outros desafios como a roteirização de *first-mile*, *middle-mile*, localização de centros de expedição e centros de expedição, que ainda continuam abertos (Loggi, 2021).

Como proposta de continuidade do primeiro trabalho, sugerimos explorar formas de modelar outros tipos de incertezas típicas do problema, como o tempo de viagem estocástico e dinâ-

mico. Além disso, pretendemos identificar diferentes configurações como robustez no número de agentes, que impacta diretamente na qualidade das rotas e no encerramento dos *unit loads*; isso possibilitaria um melhor aproveitamento da ocupação dos veículos.Também pretendemos explorar outros padrões nas rotas que possam ser usados para melhorar a aposta do agente. Outros estudos, como o apresentado em Soeanu et al. (2020), exploraram estratégias de mitigação de risco que podem levar à falha na entrega. Trabalhos futuros devem explorar a inclusão de recursos de risco nas células para criar regras de associação que considerem a probabilidade de falha na entrega do pacote. Além disso, o método usa heurísticas customizadas para selecionar e fechar as rotas. Acreditamos que o uso de heurísticas aprendidas de dados pode melhorar potencialmente essas duas fases.

Na segunda parte, validamos a prova de conceito do uso do MicroTEDAclus em trajetórias aplicado em problemas no *last-mile*, como proposta de continuidade é realizar testes reais na operação, em colaboração com a empresa de logística de *e-commerce* Loggi.

Referências Bibliográficas

J. Abonyi and B. Feil. *Cluster Analysis for Data Mining and System Identification*. Springer Science & Business Media, 2007. ISBN 978-3-7643-7987-2. doi: 10.1007/978-3-7643-7988-9. URL http://link.springer.com/10.1007/978-3-7643-7988-9.

M. Ackerman and S. Dasgupta. Incremental clustering: The case for extra clusters. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 307–315. Curran Associates, Inc., 2014.

P. K. Agarwal, K. Fox, K. Munagala, A. Nath, J. Pan, and E. Taylor. Subtrajectory clustering: Models and algorithms. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, SIGMOD/PODS '18, pages 75–87, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-4706-8. doi: 10.1145/3196959.3196972. URL http://doi.acm.org/10.1145/3196959.3196972.

C. C. Aggarwal. *Data Mining: The Textbook*. Springer Publishing Company, Incorporated, 2015. ISBN 3319141414.

C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, VLDB '03, pages 81–92. VLDB Endowment, 2003. ISBN 0-12-722442-4.

R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc. ISBN 1-55860-153-8. URL http://dl.acm.org/citation.cfm?id=645920.672836.

R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. *SIGMOD Rec.*, 22(2):207–216, June 1993. ISSN 0163-5808. doi: 10.1145/170036.170072. URL http://doi.acm.org/10.1145/170036.170072.

P. Angelov. Outside the box: an alternative data analytics framework. *Journal of Automation Mobile Robotics and Intelligent Systems*, 8(2):29–35, 2014a.

P. Angelov. Anomaly detection based on eccentricity analysis. In *2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, pages 1–8, Dec 2014b. doi: 10.1109/EALS.2014.7009497.

B. Aronov, S. Har-Peled, C. Knauer, Y. Wang, and C. Wenk. Fréchet distance for curves, revisited. In Y. Azar and T. Erlebach, editors, *Algorithms – ESA 2006*, pages 52–63, Berlin, Heidelberg, 2006a. Springer Berlin Heidelberg. ISBN 978-3-540-38876-0.

B. Aronov, S. Har-Peled, C. Knauer, Y. Wang, and C. Wenk. Fréchet Distance for Curves, Revisited. In Y. Azar and T. Erlebach, editors, *Algorithms – ESA 2006*, pages 52–63, Berlin, Heidelberg, 2006b. Springer Berlin Heidelberg. ISBN 978-3-540-38876-0.

G. Ausiello, E. Feuerstein, S. Leonardi, L. Stougie, and M. Talamo. Algorithms for the on-line travelling salesman. *Algorithmica (New York)*, 29(4):560–581, 2001. ISSN 01784617. doi: 10.1007/s004530010071. URL `https://link.springer.com/article/10.1007/s004530010071`.

R. Baldacci, A. Mingozzi, R. Roberti, and R. W. Calvo. An exact algorithm for the two-echelon capacitated vehicle routing problem. *Operations Research*, 61(2):298–314, mar 2013. ISSN 0030364X. doi: 10.1287/opre.1120.1153. URL `https://pubsonline.informs.org/doi/abs/10.1287/opre.1120.1153`.

J. Bao, Y. Zheng, D. Wilkie, and M. Mokbel. Recommendations in location-based social networks: a survey. *GeoInformatica*, 19(3):525–565, jul 2015. ISSN 1573-7624. doi: 10.1007/s10707-014-0220-8. URL `https://doi.org/10.1007/s10707-014-0220-8`.

J. Bao, T. He, S. Ruan, Y. Li, and Y. Zheng. Planning bike lanes based on sharing-bikes' trajectories. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 1377–1386, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4887-4. doi: 10.1145/3097983.3098056. URL `http://doi.acm.org/10.1145/3097983.3098056`.

D. Barbucha and P. Jedrzejowicz. Multi-agent platform for solving the dynamic vehicle routing problem. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pages 517–522, 2008. doi: 10.1109/ITSC.2008.4732573.

D. Barbucha and P. Jędrzejowicz. Agent-based approach to the dynamic vehicle routing problem. In *Advances in Intelligent and Soft Computing*, volume 55, pages 169–178. Springer Verlag, 2009. ISBN 9783642004865. doi: 10.1007/978-3-642-00487-2_18. URL `https://link.springer.com/chapter/10.1007/978-3-642-00487-2{_}18`.

D. Barbucha, I. Czarnowski, P. Jedrzejowicz, E. Ratajczak-Ropel, and I. Wierzbowska. *e-JABAT – An Implementation of the Web-Based A-Team*, pages 57–86. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-88071-4. doi: 10.1007/978-3-540-88071-4_4. URL https://doi.org/10.1007/978-3-540-88071-4_4.

A. Basiri, P. Amirian, and P. Mooney. Using Crowdsourced Trajectories for Automated OSM Data Entry Approach. *Sensors*, 16(9):1510, sep 2016. ISSN 1424-8220. doi: 10.3390/s16091510. URL http://www.mdpi.com/1424-8220/16/9/1510.

R. W. Bent and P. Van Hentenryck. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52(6):977–987, nov 2004. ISSN 0030364X. doi: 10.1287/opre.1040.0124. URL https://pubsonline.informs.org/doi/abs/10.1287/opre.1040.0124.

F. M. Bergmann, S. M. Wagner, and M. Winkenbach. Integrating first-mile pickup and last-mile delivery on shared vehicle routes for efficient urban e-commerce distribution. *Transportation Research Part B: Methodological*, 131:26–62, 2020. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2019.09.013. URL https://www.sciencedirect.com/science/article/pii/S0191261518310166.

M. Bernardo and J. Pannek. Robust Solution Approach for the Dynamic and Stochastic Vehicle Routing Problem. *Journal of Advanced Transportation*, 2018, 2018. ISSN 20423195. doi: 10.1155/2018/9848104.

D. Bertsimas, P. Jaillet, and S. Martin. Online vehicle routing: The edge of optimization in large-scale applications. *Operations Research*, 67(1):143–162, jan 2019. ISSN 15265463. doi: 10.1287/opre.2018.1763. URL https://pubsonline.informs.org/doi/abs/10.1287/opre.2018.1763.

C. G. Bezerra, B. S. J. Costa, L. A. Guedes, and P. P. Angelov. A new evolving clustering algorithm for online data streams. In *Evolving and Adaptive Intelligent Systems (EAIS), 2016 IEEE Conference on*, pages 162–168. IEEE, 2016.

M. Boukhechba, A. Bouzouane, B. Bouchard, C. Gouin-Vallerand, and S. Giroux. Online recognition of people's activities from raw gps data: Semantic trajectory data analysis. In *Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '15, pages 40:1–40:8, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3452-5. doi: 10.1145/2769493.2769498. URL http://doi.acm.org/10.1145/2769493.2769498.

N. Boysen, R. de Koster, and F. Weidinger. Warehousing in the e-commerce era: A survey, sep 2019. ISSN 03772217.

K. Braekers, K. Ramaekers, and I. V. Nieuwenhuyse. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99:300–313, 2016a. ISSN 0360-8352. doi: https://doi.org/10.1016/j.cie.2015.12.007. URL `http://www.sciencedirect.com/science/article/pii/S0360835215004775`.

K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse. The vehicle routing problem: State of the art classification and review, sep 2016b. ISSN 03608352.

K. Bujel, F. Lai, M. Szczecinski, W. So, and M. Fernandez. Solving High Volume Capacitated Vehicle Routing Problem with Time Windows using Recursive-DBSCAN clustering algorithm. *ArXiv*, dec 2018. URL `http://arxiv.org/abs/1812.02300`.

E. Cao, M. Lai, and H. Yang. Open vehicle routing problem with demand uncertainty and its robust strategies. *Expert Systems with Applications*, 41(7):3569 – 3575, 2014. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2013.11.004. URL `http://www.sciencedirect.com/science/article/pii/S0957417413009044`.

F. Cao, M. Ester, W. Qian, and A. Zhou. Density-based clustering over an evolving data stream with noise. In *In 2006 SIAM Conference on Data Mining*, pages 328–339, 2006.

H. Cao, N. Mamoulis, and D. W. Cheung. Mining frequent spatio-temporal sequential patterns. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 82–89, 2005. ISBN 0769522785. doi: 10.1109/ICDM.2005.95.

C. Chen, D. Zhang, X. Ma, B. Guo, L. Wang, Y. Wang, and E. Sha. crowddeliver: Planning city-wide package delivery paths leveraging the crowd of taxis. *IEEE Transactions on Intelligent Transportation Systems*, 18(6):1478–1496, June 2017a. ISSN 1524-9050. doi: 10.1109/TITS.2016.2607458.

L. Chen, M. Lv, Q. Ye, G. Chen, and J. Woodward. A personal route prediction system based on trajectory data mining. *Information Sciences*, 181(7):1264–1284, apr 2011. ISSN 00200255. doi: 10.1016/j.ins.2010.11.035.

P. Chen, H. Lv, S. Gao, Q. Niu, and S. Xia. A real-time taxicab recommendation system using big trajectories data. *Wireless Communications and Mobile Computing*, 2017:1–18, 2017b. ISSN 1530-8669. doi: 10.1155/2017/5414930. URL `https://www.hindawi.com/journals/wcmc/2017/5414930/`.

W. J. Cook. *In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation*. Princeton University Press, Princeton, 27 Dec. 2011. ISBN 978-1-4008-3959-9. doi: https://doi.org/10.1515/9781400839599. URL `https://www.degruyter.com/princetonup/view/title/507955`.

B. S. J. Costa, P. P. Angelov, and L. A. Guedes. Fully unsupervised fault detection and identification based on recursive density estimation and self-evolving cloud-based classifier. *Neurocomputing*, 150:289 – 303, 2015. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2014.05.086. Bioinspired and knowledge based techniques and applications The Vitality of Pattern Recognition and Image Analysis Data Stream Classification and Big Data Analytics.

B. S. J. Costa, C. G. Bezerra, L. A. Guedes, and P. P. Angelov. Unsupervised classification of data streams based on typicality and eccentricity data analytics. In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 58–63, July 2016. doi: 10.1109/FUZZ-IEEE.2016.7737668.

T. G. Crainic, N. Ricciardi, and G. Storchi. Models for evaluating and planning city logistics systems. *Transportation Science*, 43(4):432–454, oct 2009. ISSN 15265447. doi: 10.1287/trsc.1090.0279. URL `https://pubsonline.informs.org/doi/abs/10.1287/trsc.1090.0279`.

M. O. Cruz, H. Macêdo, and A. Guimarães. Grouping similar trajectories for carpooling purposes. In *2015 Brazilian Conference on Intelligent Systems (BRACIS)*, pages 234–239, Nov 2015. doi: 10.1109/BRACIS.2015.36.

T. L. C. da Silva, K. Zeitouni, and J. A. F. d. Macêdo. Online clustering of trajectory data stream. In *2016 17th IEEE International Conference on Mobile Data Management (MDM)*, volume 1, pages 112–121, June 2016a. doi: 10.1109/MDM.2016.28.

T. L. C. da Silva, K. Zeitouni, J. A. F. d. Macêdo, and M. A. Casanova. A framework for online mobility pattern discovery from trajectory data streams. In *2016 17th IEEE International Conference on Mobile Data Management (MDM)*, volume 1, pages 365–368, June 2016b. doi: 10.1109/MDM.2016.65.

T. L. C. da Silva, F. Lettich, J. A. F. de Macêdo, K. Zeitouni, and M. A. Casanova. Online clustering of trajectories in road networks. In *2020 21st IEEE International Conference on Mobile Data Management (MDM)*, pages 99–108, 2020. doi: 10.1109/MDM48529.2020.00031.

F. B. de Oliveira, R. Enayatifar, H. J. Sadaei, F. G. Guimarães, and J.-Y. Potvin. A cooperative coevolutionary algorithm for the multi-depot vehicle routing problem. *Expert Systems with Applications*, 43:117 – 130, 2016. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2015.08.030. URL `http://www.sciencedirect.com/science/article/pii/S0957417415005771`.

A. Dixit, A. Mishra, and A. Shukla. Vehicle routing problem with time windows using metaheuristic algorithms: A survey. In N. Yadav, A. Yadav, J. C. Bansal, K. Deep, and J. H. Kim, editors, *Harmony Search and Nature Inspired Optimization Algorithms*, pages 539–546, Singapore, 2019. Springer Singapore. ISBN 978-981-13-0761-4.

T. Duong, B. Goud, and K. Schauer. Closed-form density-based framework for automatic detection of cellular morphology changes. *Proceedings of the National Academy of Sciences of the United States of America*, 109(22):8382–8387, may 2012. ISSN 00278424. doi: 10.1073/pnas.1117796109. URL `https://www.pnas.org/content/109/22/8382https://www.pnas.org/content/109/22/8382.abstract`.

V. Eric, R. Jesse, E. Eric, S. Robert, B. Julien, and M. Tom. S2-geometry (0.9.0), 2011. URL `https://s2geometry.io/`.

U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37, Mar. 1996. doi: 10.1609/aimag.v17i3.1230. URL `https://ojs.aaai.org/index.php/aimagazine/article/view/1230`.

Z. Feng and Y. Zhu. A survey on trajectory data mining: Techniques and applications. *IEEE Access*, 4:2056–2067, 2016. ISSN 2169-3536. doi: 10.1109/ACCESS.2016.2553681.

J. C. Fonseca-Galindo and A. P. Lemos. Proposta de um método de extração de características aplicado ao problema de estimação da posição de um vant em navegação autônoma. *Anais do XXI Congresso Brasileiro de Automática*, pages 3139–3144, 2016.

Z. Fu, Z. Tian, Y. Xu, and K. Zhou. Mining Frequent Route Patterns Based on Personal Trajectory Abstraction. *IEEE Access*, 5:11352–11363, 2017. ISSN 21693536. doi: 10.1109/ACCESS.2017.2712703.

J. Gama. *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 1st edition, 2010. ISBN 1439826110, 9781439826119.

F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli, C. Renso, S. Rinzivillo, and R. Trasarti. Unveiling the complexity of human mobility by querying and mining massive trajectory data. *The VLDB Journal*, 20(5):695–719, oct 2011. ISSN 1066-8888. doi: 10.1007/s00778-011-0244-8. URL `http://link.springer.com/10.1007/s00778-011-0244-8`.

B. L. Golden, S. Raghavan, and E. A. Wasil. *The vehicle routing problem: latest advances and new challenges*, volume 43. Springer Science & Business Media, 2008. ISBN 978-0-387-77778-8.

Y. Gong, E. Chen, X. Zhang, L. M. Ni, and J. Zhang. Antmapper: An ant colony-based map matching approach for trajectory-based applications. *IEEE Transactions on Intelligent Transportation Systems*, 19(2):390–401, Feb 2018. ISSN 1524-9050. doi: 10.1109/TITS.2017.2697439.

G. Gutin and A. P. Punnen, editors. *The Traveling Salesman Problem and Its Variations*, volume 12 of *Combinatorial Optimization*. Springer US, Boston, MA, 2007. ISBN 978-0-387-44459-8. doi: 10.1007/b101971. URL `http://link.springer.com/10.1007/b101971`.

M. Hahsler and M. Bolaños. Clustering data streams based on shared density between micro-clusters. *IEEE Transactions on Knowledge and Data Engineering*, 28(6):1449–1461, June 2016. ISSN 1041-4347. doi: 10.1109/TKDE.2016.2522412.

J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 29(2):1–12, jun 2000. ISSN 01635808. doi: 10.1145/335191.335372. URL `http://portal.acm.org/citation.cfm?doid=335191.335372`.

S. Har-Peled and B. Raichel. The fréchet distance revisited and extended. *ACM Transactions on Algorithms (TALG)*, 10(1):1–22, 2014.

M. Hashemi and H. A. Karimi. A critical review of real-time map-matching algorithms: Current issues and future directions. *Computers, Environment and Urban Systems*, 48: 153–165, nov 2014. ISSN 01989715. doi: 10.1016/j.compenvurbsys.2014.07.009. URL `https://linkinghub.elsevier.com/retrieve/pii/S0198971514000908`.

M. Hashemi and H. A. Karimi. A weight-based map-matching algorithm for vehicle navigation in complex urban networks. *Journal of Intelligent Transportation Systems*, 20 (6):573–590, nov 2016. ISSN 1547-2450. doi: 10.1080/15472450.2016.1166058. URL `https://www.tandfonline.com/doi/full/10.1080/15472450.2016.1166058`.

V. C. Hemmelmayr, J. F. Cordeau, and T. G. Crainic. An adaptive large neighborhood search heuristic for Two-Echelon Vehicle Routing Problems arising in city logistics. *Computers and Operations Research*, 39(12):3215–3228, dec 2012. ISSN 03050548. doi: 10.1016/j.cor.2012.04.007.

Y. Huang, M. Savelsbergh, and L. Zhao. Designing logistics systems for home delivery in densely populated urban areas. *Transportation Research Part B: Methodological*, 115:95–125, sep 2018. ISSN 01912615. doi: 10.1016/j.trb.2018.07.006.

D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993. doi: 10.1109/34.232073.

R. Hyde, P. Angelov, and A. MacKenzie. Fully online clustering of evolving data streams into arbitrarily shaped clusters. *Information Sciences*, 382:96–114, 2017.

P. Jaillet and M. R. Wagner. Online vehicle routing problems: A survey. *Operations Research/ Computer Science Interfaces Series*, 43:221–237, 2008. ISSN 1387666X. doi: 10.1007/978-0-387-77778-8_10. URL `https://link.springer.com/chapter/10.1007/978-0-387-77778-8{_}10`.

W. Joe and H. C. Lau. Deep reinforcement learning approach to solve dynamic vehicle routing problem with stochastic customers. *Proceedings of the International Conference on Automated Planning and Scheduling*, 30(1):394–402, Jun. 2020. URL `https://ojs.aaai.org/index.php/ICAPS/article/view/6685`.

W. Kang, S. Li, W. Chen, K. Lei, and T. Wang. Online map-matching algorithm using object motion laws. In *2017 ieee 3rd international conference on big data security on cloud (bigdatasecurity), ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids)*, pages 249–254, May 2017. doi: 10.1109/BigDataSecurity.2017.31.

D. Kangin and P. Angelov. Evolving clustering, classification and regression with teda. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pages 1–8. IEEE, 2015.

D. Kangin, P. Angelov, and J. A. Iglesias. Autonomously evolving classifier tedaclass. *Information Sciences*, 366:1 – 11, 2016. ISSN 0020-0255. doi: http://dx.doi.org/10.1016/j.ins.2016.05.012.

C. Katrakazas, M. A. Quddus, and W.-H. Chen. Real-time classification of aggregated traffic conditions using relevance vector machines. 2016.

M. A. Klapp, A. L. Erera, and A. Toriello. The Dynamic Dispatch Waves Problem for same-day delivery. *European Journal of Operational Research*, 271(2):519–534, dec 2018. ISSN 03772217. doi: 10.1016/j.ejor.2018.05.032.

J. B. Kruskal. An overview of sequence comparison: Time warps, string edits, and macromolecules. *SIAM review*, 25(2):201–237, 1983.

J. Kytöjoki, T. Nuortio, O. Bräysy, and M. Gendreau. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers and Operations Research*, 34(9):2743–2757, sep 2007. ISSN 03050548. doi: 10.1016/j.cor.2005.10.010.

R. Lan, Y. Yu, L. Cao, P. Song, and Y. Wang. Discovering evolving moving object groups from massive-scale trajectory streams. In *2017 18th IEEE International Conference on Mobile Data Management (MDM)*, pages 256–265, 2017. doi: 10.1109/MDM.2017.42.

G. Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3):345–358, jun 1992. ISSN 03772217. doi: 10.1016/0377-2217(92)90192-C. URL https://linkinghub.elsevier.com/retrieve/pii/037722179290192C.

G. Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, oct 2009. ISSN 15265447. doi: 10.1287/trsc.1090.0301. URL https://pubsonline.informs.org/doi/abs/10.1287/trsc.1090.0301.

A. Larsen, O. Madsen, and M. Solomon. Partially dynamic vehicle routing - Models and algorithms. *Journal of the Operational Research Society*, 53(6):637–646, may 2002. ISSN 01605682. doi: 10.1057/palgrave.jors.2601352. URL https://link.springer.com/article/10.1057/palgrave.jors.2601352.

J. F. Laundrie. *Unitizing goods on pallets and slipsheets*, volume 52. US Department of Agriculture, Forest Service, Forest Products Laboratory, 1986.

R. Laxhammar and G. Falkman. Online learning and sequential anomaly detection in trajectories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1158–1173, 2014. doi: 10.1109/TPAMI.2013.172.

A. J. Lee, Y.-A. Chen, and W.-C. Ip. Mining frequent trajectory patterns in spatial-temporal databases. *Information Sciences*, 179(13):2218–2231, 2009.

A. Lemos, F. Gomide, and W. Caminhas. Multivariable gaussian evolving fuzzy modeling system. *Fuzzy Systems, IEEE Transactions on*, 19(1):91–104, 2011.

J. Leskovec, A. Rajaraman, and J. D. Ullman. *Mining of Massive Datasets*. Cambridge University Press, Cambridge, 2014. ISBN 9781139924801. doi: 10.1017/CBO9781139924801. URL http://ebooks.cambridge.org/ref/id/CBO9781139924801.

F. Li, B. Golden, and E. Wasil. The open vehicle routing problem: Algorithms, large-scale test problems, and computational results. *Computers and Operations Research*, 34(10):2918–2930, oct 2007. ISSN 03050548. doi: 10.1016/j.cor.2005.11.018.

H. Li, Y. Wang, D. Zhang, M. Zhang, and E. Y. Chang. PFP: Parallel FP-growth for query recommendation. In *RecSys'08: Proceedings of the 2008 ACM Conference on Recommender Systems*, pages 107–114, New York, New York, USA, 2008. ACM Press. ISBN 9781605580937. doi: 10.1145/1454008.1454027. URL http://portal.acm.org/citation.cfm?doid=1454008.1454027.

J. Li, Q. Qin, J. Han, L.-A. Tang, and K. H. Lei. Mining Trajectory Data and Geotagged Data in Social Media for Road Map Inference. *Transactions in GIS*, 19(1):1–18, feb 2015a. ISSN 13611682. doi: 10.1111/tgis.12072. URL `http://doi.wiley.com/10.1111/tgis.12072`.

X. Li, C. Yu, L. Ju, J. Qin, Y. Zhang, L. Dou, and S. Yuqing. Position prediction system based on spatio-temporal regularity of object mobility. *Information Systems*, 75:43–55, jun 2018. ISSN 03064379. doi: 10.1016/j.is.2018.02.004.

Y. Li, Y. Zheng, S. Ji, W. Wang, L. H. U, and Z. Gong. Location selection for ambulance stations: A data-driven approach. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '15, pages 85:1–85:4, New York, NY, USA, 2015b. ACM. ISBN 978-1-4503-3967-4. doi: 10.1145/2820783.2820876. URL `http://doi.acm.org/10.1145/2820783.2820876`.

D. Liu, D. Weng, Y. Li, J. Bao, Y. Zheng, H. Qu, and Y. Wu. Smartadp: Visual analytics of large-scale taxi trajectories for selecting billboard locations. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):1–10, Jan 2017a. ISSN 1077-2626. doi: 10.1109/TVCG.2016.2598432.

H. Liu, J. Li, Y. Wu, and Y. Fu. Clustering with outlier removal. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2369–2379, 2021. doi: 10.1109/TKDE.2019.2954317.

S. Liu and S. Wang. Trajectory community discovery and recommendation by multi-source diffusion modeling. *IEEE Transactions on Knowledge and Data Engineering*, 29(4):898–911, April 2017. ISSN 1041-4347. doi: 10.1109/TKDE.2016.2637898.

X. Liu, K. Liu, M. Li, and F. Lu. A st-crf map-matching method for low-frequency floating car data. *IEEE Transactions on Intelligent Transportation Systems*, 18(5):1241–1254, May 2017b. ISSN 1524-9050. doi: 10.1109/TITS.2016.2604484.

Loggi. loggibud: Loggi benchmark for urban deliveries. *GitHub repository*, 2021.

M. Lv, L. Chen, Z. Xu, Y. Li, and G. Chen. The discovery of personally semantic places based on trajectory data mining. *Neurocomput.*, 173(P3):1142–1153, Jan. 2016. ISSN 0925-2312. doi: 10.1016/j.neucom.2015.08.071. URL `https://doi.org/10.1016/j.neucom.2015.08.071`.

M. Lv, L. Chen, T. Chen, D. Zeng, and B. Cao. Discovering individual movement patterns from cell-id trajectory data by exploiting handoff features. *Information Sciences*, 474:18–32, feb 2019. ISSN 00200255. doi: 10.1016/j.ins.2018.09.033.

S. Ma, Y. Zheng, and O. Wolfson. Real-time city-scale taxi ridesharing. *IEEE Transactions on Knowledge and Data Engineering*, 27(7):1782–1795, July 2015. ISSN 1041-4347. doi: 10.1109/TKDE.2014.2334313.

J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

J. Maia, C. A. Severiano, F. G. Guimarães, C. L. de Castro, A. P. Lemos, J. C. Fonseca Galindo, and M. Weiss Cohen. Evolving clustering algorithm based on mixture of typicalities for stream data mining. *Future Generation Computer Systems*, 106:672–684, 2020. ISSN 0167-739X. doi: https://doi.org/10.1016/j.future.2020.01.017. URL `https://www.sciencedirect.com/science/article/pii/S0167739X19312786`.

Y. Maknoon and G. Laporte. Vehicle routing with cross-dock selection. *Computers & Operations Research*, 77:254 – 266, 2017. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2016.08.007. URL `http://www.sciencedirect.com/science/article/pii/S0305054816302027`.

N. Mamoulis, M. Hadjieleftheriou, H. Cao, Y. Tao, G. Kollios, and D. W. Cheung. Mining, indexing, and querying historical spatiotemporal data. In *KDD-2004 - Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 236–245, New York, New York, USA, 2004. Association for Computing Machinery. ISBN 1581138881. doi: 10.1145/1014052.1014080. URL `http://portal.acm.org/citation.cfm?doid=1014052.1014080`.

M. Maruseac and G. Ghinita. Differentially-private mining of representative travel patterns. In *2016 17th IEEE International Conference on Mobile Data Management (MDM)*, volume 1, pages 272–281, June 2016. doi: 10.1109/MDM.2016.48.

F. McSherry and K. Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103, Oct 2007. doi: 10.1109/FOCS.2007.66.

T. Nakata and J. I. Takeuchi. Mining traffic data from probe-car system for travel time prediction. In *KDD-2004 - Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 817–822, New York, New York, USA, 2004. Association for Computing Machinery (ACM). ISBN 1581138881. doi: 10.1145/1014052.1016920. URL `http://portal.acm.org/citation.cfm?doid=1014052.1016920`.

S. Nucamendi-Guillén, A. G. Padilla, E. Olivares-Benitez, and J. M. Moreno-Vega. The multi-depot open location routing problem with a heterogeneous fixed fleet. *Expert Systems with Applications*, 165:113846, 2021.

L. Perron and V. Furnon. Or-tools (7.2), 2019. URL `https://developers.google.com/optimization/`.

D. T. Pham, S. S. Dimov, and C. D. Nguyen. An incremental k-means algorithm. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 218(7):783–795, 2004. doi: 10.1243/0954406041319509.

G. Piatetsky-Shapiro. Discovery, analysis and presentation of strong rules. In G. Piatetsky-Shapiro and W. J. Frawley, editors, *Knowledge Discovery in Databases*, pages 229–248. AAAI Press, 1991.

V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, feb 2013. ISSN 03772217. doi: 10.1016/j.ejor.2012.08.015.

H. N. Psaraftis. Dynamic vehicle routing problems. *Vehicle routing: Methods and studies*, 16: 223–248, 1988. URL `http://www.martrans.org/documents/2008/rst/dvrppsaraftis88.pdf`.

H. N. Psaraftis, M. Wen, and C. A. Kontovas. Dynamic vehicle routing problems: Three decades and counting. *Networks*, 67(1):3–31, 2016. doi: https://doi.org/10.1002/net.21628. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/net.21628`.

T. Qin, W. Shangguan, G. Song, and J. Tang. Spatio-temporal routine mining on mobile phone data. *ACM Trans. Knowl. Discov. Data*, 12(5):56:1–56:24, June 2018. ISSN 1556-4681. doi: 10.1145/3201577. URL `http://doi.acm.org/10.1145/3201577`.

L. Ravi and S. Vairavasundaram. A collaborative location based travel recommendation system through enhanced rating prediction for the group of users. *Computational intelligence and neuroscience*, 2016:1–28, 2016. ISSN 1687-5265. doi: 10.1155/2016/1291358. URL `http://www.hindawi.com/journals/cin/2016/1291358/`.

U. Ritzinger, J. Puchinger, and R. F. Hartl. A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research*, 54(1):215–231, jan 2016. ISSN 1366588X. doi: 10.1080/00207543.2015.1043403. URL `https://www.tandfonline.com/doi/abs/10.1080/00207543.2015.1043403`.

M. T. Robinson. The temporal development of collision cascades in the binary-collision approximation. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 48(1):408–413, 1990. ISSN 0168-583X. doi: https://doi.org/10.1016/0168-583X(90)90150-S. URL `https://www.sciencedirect.com/science/article/pii/0168583X9090150S`.

K. Sahr, D. White, and A. J. Kimerling. Geodesic discrete global grid systems. *Cartography and Geographic Information Science*, 30(2):121–134, 2003.

I. Sanaullah, M. Quddus, and M. Enoch. Developing travel time estimation methods using sparse GPS data. *Journal of Intelligent Transportation Systems*, 20(6):532–544, 2016. doi: 10.1080/15472450.2016.1154764. URL `https://doi.org/10.1080/15472450.2016.1154764`.

S. Sankararaman, P. K. Agarwal, T. Mølhave, J. Pan, and A. P. Boedihardjo. Model-driven matching and segmentation of trajectories. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL 13, pages 234–243, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450325219. doi: 10.1145/2525314.2525360. URL `https://doi.org/10.1145/2525314.2525360`.

J. G. Saw, M. C. K. Yang, and T. C. Mo. Chebyshev inequality with estimated mean and variance. *The American Statistician*, 38(2):130–132, 1984. ISSN 00031305.

L. Schrage. Formulation and structure of more complex/realistic routing and scheduling problems. *Networks*, 11(2):229–232, 1981. ISSN 00283045. doi: 10.1002/net.3230110212. URL `http://doi.wiley.com/10.1002/net.3230110212`.

T. T. Shein and S. Puntheeranurak. Incremental clustering approach for evolving trajectory data stream. In *2018 International Electrical Engineering Congress (iEECON)*, pages 1–4, 2018. doi: 10.1109/IEECON.2018.8712334.

D. H. Shih, M. H. Shih, D. C. Yen, and J. H. Hsu. Personal mobility pattern mining and anomaly detection in the GPS era. *Cartography and Geographic Information Science*, 43 (1):55–67, jan 2016. ISSN 15450465. doi: 10.1080/15230406.2015.1010585. URL `https://www.tandfonline.com/doi/abs/10.1080/15230406.2015.1010585`.

J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. d. Carvalho, and J. a. Gama. Data stream clustering: A survey. *ACM Comput. Surv.*, 46(1):13:1–13:31, July 2013. ISSN 0360-0300. doi: 10.1145/2522968.2522981.

D.-G. Sim, O.-K. Kwon, and R.-H. Park. Object matching algorithms using robust hausdorff distance measures. *IEEE Transactions on Image Processing*, 8(3):425–429, 1999. doi: 10.1109/83.748897.

A. Soeanu, S. Ray, J. Berger, A. Boukhtouta, and M. Debbabi. Multi-depot vehicle routing problem with risk mitigation: Model and solution algorithm. *Expert Systems with Applications*, 145:113099, 2020. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2019.113099. URL `http://www.sciencedirect.com/science/article/pii/S0957417419308164`.

A. Spark. Apache spark. *Retrieved January*, 17:2018, 2018.

M. Spichkova, M. Simic, and H. Schmidt. Formal model for intelligent route planning. *Procedia Computer Science*, 60(1):1299–1308, 2015. ISSN 18770509. doi: 10.1016/j.procs.2015.08.196.

M. Stone. Cross-Validatory Choice and Assessment of Statistical Predictions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(2):111–133, jan 1974. doi: 10.1111/j.2517-6161.1974.tb00994.x. URL `https://rss.onlinelibrary.wiley.com/doi/full/10.1111/j.2517-6161.1974.tb00994.x`.

Z. Sun and X. J. Ban. Identifying multiclass vehicles using global positioning system data. *Journal of Intelligent Transportation Systems*, 22(1):1–9, 2018. doi: 10.1080/15472450.2017.1358623. URL `https://doi.org/10.1080/15472450.2017.1358623`.

S. Talukdar, L. Baerentzen, A. Gove, and P. De Souza. Asynchronous Teams: Cooperation Schemes for Autonomous Agents. *Journal of Heuristics*, 4(4):295–321, 1998. ISSN 13811231. doi: 10.1023/A:1009669824615. URL `https://link.springer.com/article/10.1023/A:1009669824615`.

S. R. Thangiah, O. Shmygelska, and W. Mennell. An agent architecture for vehicle routing problems. In *Proceedings of the ACM Symposium on Applied Computing*, pages 517–521, New York, New York, USA, mar 2001. Association for Computing Machinery. ISBN 1581132875. doi: 10.1145/372202.372445. URL `http://portal.acm.org/citation.cfm?doid=372202.372445`.

S. T. S. Thong, C. T. Han, and T. A. Rahman. Intelligent fleet management system with concurrent gps amp; gsm real-time positioning technology. In *2007 7th International Conference on ITS Telecommunications*, pages 1–6, June 2007. doi: 10.1109/ITST.2007.4295849.

M. W. Ulmer, D. C. Mattfeld, and F. Köster. Budgeting time for dynamic vehicle routing with stochastic customer requests. *Transportation Science*, 52(1):20–37, 2018. doi: 10.1287/trsc.2016.0719. URL `https://doi.org/10.1287/trsc.2016.0719`.

M. W. Ulmer, J. C. Goodson, D. C. Mattfeld, and M. Hennig. Offline-online approximate dynamic programming for dynamic vehicle routing with stochastic requests. *Transportation Science*, 53(1):185–202, 2019a. doi: 10.1287/trsc.2017.0767. URL `https://doi.org/10.1287/trsc.2017.0767`.

M. W. Ulmer, B. W. Thomas, and D. C. Mattfeld. Preemptive depot returns for dynamic same-day delivery. *EURO Journal on Transportation and Logistics*, 2019b. ISSN 21924384. doi: 10.1007/s13676-018-0124-0.

A. Vahedian Khezerlou, X. Zhou, L. Tong, Y. Li, and J. Luo. Forecasting Gathering Events through Trajectory Destination Prediction: a Dynamic Hybrid Model. *IEEE Transactions*

on Knowledge and Data Engineering, pages 1–1, aug 2019. ISSN 1041-4347. doi: 10.1109/tkde.2019.2937082.

J. I. Van Hemert and J. A. La Poutré. Dynamic routing problems with fruitful regions: Models and evolutionary computation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3242:692–701, 2004. ISSN 16113349. doi: 10.1007/978-3-540-30217-9_70. URL `https://link.springer.com/chapter/10.1007/978-3-540-30217-9{_}70`.

P. Van Hentenryck, R. Bent, and E. Upfal. Online stochastic optimization under time constraints. *Annals of Operations Research*, 177:151–183, 2010. doi: 10.1007/s10479-009-0605-5.

M. Vidović, B. Ratković, N. Bjelić, and D. Popović. A two-echelon location-routing model for designing recycling logistics networks with profit: Milp and heuristic approach. *Expert Systems with Applications*, 51:34–48, 2016.

C. Voudouris and E. P. K. Tsang. *Guided Local Search*, pages 185–218. Springer US, Boston, MA, 2003. ISBN 978-0-306-48056-0. doi: 10.1007/0-306-48056-5_7. URL `https://doi.org/10.1007/0-306-48056-5_7`.

G. Wang, A. Gunasekaran, E. W. Ngai, and T. Papadopoulos. Big data analytics in logistics and supply chain management: Certain investigations for research and applications, jun 2016. ISSN 09255273.

R. Wang, C. Chow, Y. Lyu, V. C. S. Lee, S. Kwong, Y. Li, and J. Zeng. Taxirec: Recommending road clusters to taxi drivers using ranking-based extreme learning machines. *IEEE Transactions on Knowledge and Data Engineering*, 30(3):585–598, March 2018. ISSN 1041-4347. doi: 10.1109/TKDE.2017.2772907.

I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

R. Wu, G. Luo, J. Shao, L. Tian, and C. Peng. Location prediction on trajectory data: A review. *Big Data Mining and Analytics*, 1(2):108–127, jun 2018. ISSN 20960654. doi: 10.26599/BDMA.2018.9020010. URL `https://ieeexplore.ieee.org/document/8336847/`.

Y. Zheng. Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology*, 6(3):1–41, may 2015. ISSN 21576904. doi: 10.1145/2743025. URL `http://dl.acm.org/citation.cfm?doid=2764959.2743025`.

H. Zhong, R. W. Hall, and M. Dessouky. Territory planning and vehicle dispatching with driver learning. *Transportation Science*, 41(1):74–89, feb 2007. ISSN 15265447. doi: 10.1287/trsc.1060.0167. URL `https://pubsonline.informs.org/doi/abs/10.1287/trsc.1060.0167`.