



UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG
CENTRO DE CIÊNCIAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO
CURSO DE MESTRADO EM ENGENHARIA DE COMPUTAÇÃO

Dissertação de Mestrado

**Proposta de Análise de Dados Espaciais Utilizando
Métodos Heurísticos e Técnica de Ensemble: Um estudo
de caso de roteirização de uma frota de veículos**

Timotio Capitão Cubaque

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal do Rio Grande - FURG, como requisito parcial para a obtenção do grau de Mestre em Engenharia de Computação

Orientador: Prof. Dr. Eder Mateus Nunes Gonçalves
Co-orientadora: Profa. Dra. Diana Francisca Adamatti

Rio Grande, 2023

Ficha Catalográfica

C962p Cubaque, Timotio Capitão.

Proposta de análise de dados espaciais utilizando métodos heurísticos e técnica de *Ensemble*: um estudo de caso de roteirização de uma frota de veículos / Timotio Capitão Cubaque. – 2023.

84 f.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-Graduação em Computação, Rio Grande/RS, 2023.

Orientador: Dr. Eder Mateus Nunes Gonçalves.

Coorientadora: Dra. Diana Francisca Adamatti.

1. Problema de roteamento de veículos 2. Problema de roteamento de veículos capacitados 3. Métodos heurísticos 4. Técnica de *Ensemble* I. Gonçalves, Eder Mateus Nunes II. Adamatti, Diana Francisca III. Título.

CDU 004

Catálogo na Fonte: Bibliotecário José Paulo dos Santos CRB 10/2344

DISSERTAÇÃO DE MESTRADO

Proposta de Análise de Dados Espaciais Utilizando Métodos Heurísticos e Técnica de Ensemble: Um estudo de caso de roteirização de uma frota de veículos

Timotio Capitão Cubaque

Banca examinadora:

Documento assinado digitalmente
 **EDUARDO NUNES BORGES**
Data: 06/09/2023 11:56:31-0300
Verifique em <https://validar.iti.gov.br>

Prof. Dr. Eduardo Nunes Borges (FURG)

Documento assinado digitalmente
 **MARILTON SANCHOTENE DE AGUIAR**
Data: 06/09/2023 15:19:25-0300
Verifique em <https://validar.iti.gov.br>

Prof. Dr. Marilton Sanchotene de Aguiar (UFPeI)

Documento assinado digitalmente
 **EDER MATEUS NUNES GONCALVES**
Data: 06/09/2023 10:14:46-0300
Verifique em <https://validar.iti.gov.br>

Prof. Dr. Eder Mateus Nunes Gonçalves (FURG)
Orientador

*Dedico... a minha querida mãe e irmãs.
as minhas sobrinhas meu filho e amigos.*

AGRADECIMENTOS

Agradeço a Deus pela vida, coragem e sabedoria que me deu para resistir os bons e maus momentos que foram surgindo durante este percurso acadêmico. Agradeço aos meus Pais e minhas irmãs por terem sido meus exemplos, que me transmitam amor e ensinamentos para toda vida. Aos meus Orientadores Professor Doutor Eder Mateus Nunes Gonçalves e a Professora Doutora Diana Francisca Adamatti agradeço vos profundamente pela vossa participação e constante apoio, fundamentais para desenvolvimento deste meu trabalho. Os meus agradecimentos vão de maneira particular a Universidade Federal de Rio Grande, em especial seu Magnifico Reitor, sem deixar de lado o Coordenador do PPGComP Professor Doutor Eduardo Nunes Borges e todos docentes do curso da engenharia de computação. E agradeço a todos que de alguma forma direta ou indiretamente contribuíram para realização deste meu trabalho, a minha gratidão.

*Para acompanhar a alta velocidade da tecnologia; precisamos atualizar a
cabeça todo dia , com três palavras: aprender , atualizar e adaptar.*

— DELSON JACINTO VIEIRA

RESUMO

CUBAQUE, Timotio Capitão. **Proposta de Análise de Dados Espaciais Utilizando Métodos Heurísticos e Técnica de Ensemble: Um estudo de caso de roteirização de uma frota de veículos.** 2023. 84 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande - FURG, Rio Grande.

Nos últimos anos, o problema de roteamento de veículos tem atraído cada vez mais interesse entre pesquisadores devido à sua presença em várias situações cotidianas e à grande dificuldade de solução. Esforços significativos têm sido feitos para desenvolver técnicas mais robustas e eficazes, capazes de lidar com a complexidade desse problema. O objetivo central é determinar uma série de rotas para um conjunto de veículos, os quais devem atender a um conjunto de clientes, buscando minimizar o custo total de transporte. O problema de roteamento de veículos capacitado é uma variante do problema de roteamento de veículos, no qual é necessário encontrar um conjunto de rotas para uma frota de veículos homogêneos, de modo que eles possam atender a todos os clientes. Neste contexto, há algumas restrições a serem consideradas: as rotas devem iniciar e terminar no depósito, cada cliente deve ser visitado apenas uma vez por um único veículo e a demanda total em qualquer rota não pode ultrapassar a capacidade do veículo. Devido à sua complexidade, esse tipo de problema é classificado como NP-difícil, o que significa que não existem algoritmos polinomiais conhecidos para resolvê-lo de forma ótima em tempo razoável. Portanto, soluções aproximadas são comumente buscadas utilizando algoritmos heurísticos e meta-heurísticos. O presente estudo investiga a questão da roteirização de uma frota de veículos por meio da análise de dados espaciais, empregando métodos heurísticos e a técnica de *ensemble*. Dois algoritmos de clusterização, *K-means* e DBSCAN, são empregados para agrupar os clientes de acordo com as suas localizações geográficas. A partir dessas abordagens, duas estratégias distintas são desenvolvidas, cada uma explorando um modelo específico de clusterização para agrupar os clientes. Além disso, a heurística do vizinho mais próximo é empregada para gerar soluções iniciais, enquanto o método 2-Opt é aplicado para aprimorar as soluções iniciais obtidas. Nos testes realizados, os resultados obtidos pelas estratégias propostas são comparados entre si e também com abordagem de otimização desenvolvida pela Loggi uma empresa de tecnologia voltada para logística e entregas sob demanda, *Benchmark Loggi BUD*. Os resultados destacam o desempenho positivo de ambas as estratégias, tanto em relação a distância total percorrida pela soma das rotas dos caminhões, quanto no que diz respeito ao equilíbrio das distâncias percorridas entre os caminhões.

Palavras-chave: Problema de Roteamento de Veículos, Problema de Roteamento de Veículos Capacitados, Métodos Heurísticos, Técnica de Ensemble.

ABSTRACT

CUBAQUE, Timotio Capitão. **Proposal for Spatial Data Analysis Using Heuristic Methods and Ensemble Technique: A case study of routing a fleet of vehicles.** 2023. 84 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande - FURG, Rio Grande.

In recent years, the vehicle routing problem has attracted increasing interest among researchers due to its presence in various everyday situations and the great difficulty of solving it. Significant efforts have been made to develop more robust and effective techniques capable of dealing with the complexity of this problem. The central objective is to determine a series of routes for a set of vehicles, which must serve a set of customers, seeking to minimize the total cost of transport. The Capacitated Vehicle Routing Problem is a variant of the Vehicle Routing Problem, in which it is necessary to find a set of routes for a fleet of homogeneous vehicles, so that they can serve all customers. In this context, there are some restrictions to be considered: routes must start and end at the depot, each customer must be visited only once by a single vehicle, and the total demand on any route cannot exceed the capacity of the vehicle. Due to its complexity, this type of problem is classified as NP-hard, which means that there are no known polynomial algorithms to solve it optimally in reasonable time. Therefore, approximate solutions are commonly sought using heuristic and meta-heuristic algorithms. The present study investigates the issue of routing a fleet of vehicles through the analysis of spatial data, using heuristic methods and the *ensemble* technique. Two clustering algorithms, *K-means* and DBSCAN, are employed to group clients according to their geographic locations. From these approaches, two distinct strategies are developed, each one exploring a specific clustering model to group clients. Furthermore, the nearest neighbor heuristic is employed to generate initial solutions, while the 2-Opt method is applied to improve the obtained initial solutions. In the tests carried out, the results obtained by the proposed strategies are compared with each other and also with an optimization approach developed by Loggi, a technology company focused on logistics and on-demand deliveries, *Benchmark Loggi BUD*. The results highlight the positive performance of both strategies, both in relation to the total distance traveled by the sum of the truck routes, and with regard to the balance of the distances traveled between the trucks.

Keywords: Vehicle Routing Problem, Capacitated Vehicle Routing Problem, Heuristic Methods, Ensemble Technique.

LISTA DE FIGURAS

Figura 1	Estrutura ensemble	21
Figura 2	Etapas de construção ensemble	23
Figura 3	Bagging	26
Figura 4	Boosting	28
Figura 5	Stacking	29
Figura 6	Demonstração visual do problema de roteamento de veículos com 5 veículos.	30
Figura 7	Fluxograma gerais das estratégias	50
Figura 8	Fluxograma de etapas do algoritmo k-means	52
Figura 9	Funcionamento de Método Cotovelo	56
Figura 10	Troca 2-Opt entre os trechos 1 - 2 e 3 - 4	58
Figura 11	Fluxograma geral do modelo	60
Figura 12	Método do Cotovelo para instâncias do Rio de Janeiro	64
Figura 13	Método do Cotovelo para instâncias do Distrito Federal	65
Figura 14	Método do Cotovelo para instâncias do Pará	65
Figura 15	Agrupamento por K-Means no Rio de Janeiro	66
Figura 16	Agrupamento por K-Means no Distrito Federal	66
Figura 17	Agrupamento por K-Means no Estado do Pará	67
Figura 18	Agrupamento por DBSCAN no Rio de Janeiro	69
Figura 19	Agrupamento por DBSCAN no Distrito Federal	70
Figura 20	Agrupamento por DBSCAN no Estado do Pará	70
Figura 21	Comparação entre Estratégias A, B e Abordagem da Loggi para instâncias no Rio de Janeiro	73
Figura 22	Comparação entre Estratégias A, B e Abordagem da Loggi para instâncias no Distrito Federal	73
Figura 23	Comparação entre Estratégias A, B e Abordagem da Loggi para instâncias no Pará	74

LISTA DE TABELAS

Tabela 1	variações de PRV	33
Tabela 2	Resumo dos trabalhos apresentados	40
Tabela 3	Métodos de agrupamento	51
Tabela 4	Resultados obtidos por meio da estratégia A para o Rio de Janeiro	67
Tabela 5	Resultados obtidos por meio da estratégia A para o Rio de Janeiro	67
Tabela 6	Resultados obtidos por meio da estratégia A para o Distrito Federal	68
Tabela 7	Resultados obtidos por meio da estratégia A para o Pará	68
Tabela 8	Resultados obtidos por meio da estratégia B para o Rio de Janeiro	71
Tabela 9	Resultados obtidos por meio da estratégia B para o Distrito Federal	71
Tabela 10	Resultados obtidos por meio da estratégia B para o estado do Pará	71
Tabela 11	Resultados do Rio de Janeiro	71
Tabela 12	Resultados do Distrito Federal	72
Tabela 13	Resultados do Pará	72

LISTA DE ABREVIATURAS E SIGLAS

AG	Algoritmo Genético
AM	Aprendizagem de Máquina
BUD	Benchmark for Urban Delivery
BRKGA	Biased Random-Key Genetic Algorithm
B&B	Branch-and-Bound
BT	Busca Tabu
CAS	Colônia de Formigas
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CGLs	Corporate Governance Indicators
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DPA	Desvio Padrão de Amostra
DF	Distrito Federal
ENN	Ensemble of Neural Networks
XGB	Extreme Gradient Boosting
GBM	Gradient Boosting
HHPG	Hiper-heurística de Programação Genética
ILS	Iterated Local Search
JSON	JavaScript Object Notation
L2I	Learn to Improve
ML	Machine Learning
MAPEK	Monitor-Analyse-Plan-Execute over a shared Knowledge
MTSP	Multiple Traveling Salesman Problem
OVRP	Open Vehicle Routing Problem
OR	Operational Research
PR	Pará
PSO	Particle Swarm Optimization

PAC	Probably Approximately Correct
PD	Problema de Designação
PRACI	Problema de Roteamento de Arco Capacitado Incerto
PRI	Problema de Roteamento de Inventário
PRV	Problema de Roteamento de Veículos
PRBC	Problema de Roteamento de Veículos Capacitado
PRVJT	Problema de Roteamento de Veículos com Janela de Tempo
PRVMV	Problema de Roteamento de Veículos com Múltiplas Viagens
PIB	Produto Interno Bruto
PG	Programação Genética
QL	Q-Learning
RF	Randon Florest
RNMP	Redes Neurais Multicamada Perceptron
RL	Reinforcement Learning
RJ	Rio de Janeiro
RPVCD	Routing Problem of Vehicles with Collection and Delivery
RPVP	Routing Problem of Vehicles with Profits
RPVVD	Routing Problem of Vehicles with Various Deposits
SA	Simulated Annealing
SSE	Sum of Squared Errors
SSEWC	Sum of Squared Errors Within cluster
TD	Tree Desicion
WCSS	Within Clusters Sum of Squares

SUMÁRIO

1	Introdução	15
1.1	Contribuições e Justificativas	17
1.2	Problema de Pesquisa	18
1.3	Objetivos	18
1.4	Metodologia	19
1.5	Estrutura do Trabalho	20
2	Embasamento Teórico	21
2.1	Método <i>Ensemble</i>	21
2.1.1	Regras de combinação	22
2.1.2	Diversidade	25
2.1.3	Diferentes métodos <i>ensemble</i>	26
2.2	Problema de Roteamento de Veículos	29
2.2.1	Variações e Especializações do Problema de Roteamento de Veículos	33
2.3	Problema de roteamento de veículos capacitado (PRVC)	34
2.4	Objetivos do Problema de Roteamento de Veículos	36
2.4.1	Balanceamento de Rota	36
2.4.2	Maior Rota	36
2.4.3	Distância Total Percorrida	37
2.5	Trabalhos Relacionados	37
3	Métodos de solução	41
3.1	Métodos exatos	41
3.1.1	<i>Branch-and Bound</i>	42
3.2	Métodos Heurísticos	43
3.2.1	Algoritmo <i>Clarke and Wright</i>	44
3.2.2	Algoritmo de Varredura	46
3.3	Meta-heurísticas	47
3.3.1	Busca Tabu	47
3.3.2	<i>Simulated Annealing</i>	48
4	Algoritmos e Técnicas Utilizados	50
4.1	Agrupamento	50
4.1.1	K-Means	51
4.1.2	DBSCAN	53
4.2	Método Auxiliar	54
4.2.1	Método cotovelo	55

4.3	Geração da solução inicial	56
4.3.1	Algoritmo Vizinho Mais Próximo	56
4.4	Melhoramento de solução inicial	57
4.4.1	Heurística K-Opt	58
4.5	Modelagem das Estratégias Propostas	59
5	Resultados	62
5.1	Conjunto de Dados	62
5.2	Testes	63
5.3	Métricas para avaliação das estratégias	63
5.4	Avaliação das Estratégias nas Instâncias	64
5.4.1	Estratégia baseada no algoritmo <i>K-Means</i>	64
5.4.2	Estratégia baseada no algoritmo DBSCAN	68
6	Conclusão	75
	Referências	77

1 INTRODUÇÃO

O problema de roteamento de veículos é um dos problemas em otimização combinatória com grande relevância científica [86], forma uma ampla classe de algoritmos que, genericamente, resolvem o problema de determinar uma sequência de atendimentos a partir de um ou mais depósitos dispondo de uma frota de veículos.

Apareceu pela primeira vez em um artigo de Dantzig and Ramser [26], onde a primeira abordagem algorítmica foi escrita e aplicada para a entrega de gasolina. Frequentemente, o contexto é o da entrega de mercadorias localizadas em um depósito central para clientes que fizeram pedidos dessas mercadorias.

O problema de roteamento de veículos (PRV) é um importante problema de otimização combinatória no qual o objetivo é encontrar a configuração ótima de rotas para uma frota de veículos que deve entregar algumas mercadorias de uma determinada origem (depósito) para um determinado conjunto de destinos (clientes) [13]. Tal problema é um dos desafios mais complexos na indústria de logística e no transporte de cargas. Soluções eficientes para esse problema são essenciais para minimizar custos e garantir a qualidade do transporte. Esse problema possui um número bastante grande de aplicações práticas nas diferentes áreas como: saúde pública, comércio ou indústria. Pode-se destacar ainda a distribuição de produtos diversos [26], o roteamento de ônibus em um sistema multi-escolar [61], o roteamento em linhas aéreas [59] e o roteamento de caminhões-tanque para coleta de leite [76].

Desde que o problema de roteamento de veículos foi proposto, surgiram diversas variações como, problema de roteamento de veículos com lucros, problema de roteamento de veículos com coleta e entrega, problema de roteamento de veículos com janelas de tempo, problema de roteamento de veículos capacitados, entre outros. Sendo que cada um varia em relação às suas restrições e objetivos.

No cenário de problema de roteamento de veículos capacitado (PRVC), que é o interesse deste trabalho é necessário determinar as rotas a serem percorridas por um conjunto de veículos para entregar as cargas nos clientes, levando em conta as capacidades dos veículos e as demandas de cada cliente. De forma geral, o PRVC visa gerenciar uma frota de veículos para atender a um determinado número de clientes com diferentes de-

mandas. O objetivo é projetar um conjunto de rotas com o menor custo de viagem, todas começando e terminando no depósito. A demanda total de clientes em qualquer rota não deve exceder a capacidade do veículo, e cada cliente é visitado apenas uma vez [42].

Na literatura, diversos métodos foram propostos para resolver o problema de roteamento, incluindo algoritmos evolutivos propostos por Li et al. [52], o Mecanismo Tabu Search apresentado por Bruni et al. [16] e a meta-heurística híbrida BRKGA-QL adaptada por Santos [77].

Laporte [49] mostra uma visão geral das diversas abordagens utilizadas para resolver o problema de roteirização. Estas se desdobram em algoritmos exatos, que encontram a solução ótima para o problema e algoritmos heurísticos, que buscam uma boa solução viável, mas que não é necessariamente a solução ótima.

As técnicas usualmente aplicadas na solução de problema de roteamento variam desde algoritmos com base em heurísticas ou meta-heurísticas, até métodos exatos para problemas de pequeno porte [53].

As abordagens exatas para a resolução do PRV propõe uma forma de enumerar as soluções viáveis a fim de garantir que a melhor solução seja encontrada. A enumeração total geralmente é dispendiosa e impraticável em termos computacionais, o que remete ao estudo de métodos exatos com um certo grau de inteligência para evitar a busca de soluções em regiões do espaço de soluções onde a qualidade não é atrativa. Para isto, utilizam-se técnicas como: *branch and bound* e *branch and cut* [53].

Decorrente da dificuldade computacional de resolver problemas tipicamente combinatoriais de forma exata, como o PRV, algoritmos heurísticos com a finalidade de prover soluções com uma boa qualidade em um tempo hábil foram utilizados.

Segundo Luizelli [57], os métodos heurísticos são, em geral, algoritmos específicos para um determinado problema e geralmente não se aplicam a problemas diferentes, pois tiram proveito de características inerentes a cada problema. Cada método heurístico provê uma alternativa restrita de busca por soluções no espaço de soluções. Por este motivo, os métodos heurísticos não possuem garantia de otimalidade das soluções encontradas nem uma forma de garantir o quão próximos podem estar da solução ótima.

Um dos problemas encontrados na busca de soluções utilizando métodos heurísticos são os mínimos ou máximos locais. Mínimos ou máximos locais são regiões do espaço de busca definidos em vales ou picos, ou seja, regiões onde o custo da função objetivo é o maior (maximização) ou o menor (minimização) dentre as soluções possíveis na vizinhança analisada [57].

As meta-heurísticas, diferentemente das heurísticas, possuem uma definição genérica o suficiente para serem estendidas para a resolução de qualquer problema de otimização numérica ou combinatória. Geralmente se baseiam em comportamentos inteligentes de sistemas naturais e com base nestas características se constituem em mecanismos potencialmente importantes para a busca de boas soluções em problemas complexos, como os

problemas da classe NP-difícil [64].

Portanto, a seleção do algoritmo utilizado na resolução de problema de roteamento é um processo de grande importância. Isso se deve ao fato de que a escolha adequada de um algoritmo pode resultar em uma relação benéfica mais acentuada ao determinar as rotas.

Nessa pesquisa são utilizados dois algoritmos de clusterização, *K-means* e DBSCAN para realizar o agrupamento dos pontos de entrega. No método *K-Means*, os pontos de entrega são divididos em grupos começando com centros aleatórios, e esses centros são iterativamente ajustados com base na proximidade dos pontos. Por outro lado DBSCAN identifica áreas densas entre regiões menos densas, expandindo os grupos a partir de pontos centrais baseado em proximidade. Com base nesses algoritmos, duas estratégias diferentes são desenvolvidas, cada uma delas explorando um dos algoritmos de clusterização mencionados. Além disso, são empregadas as heurísticas do vizinho mais próximo, que é utilizado para gerar uma solução inicial, e o 2-Opt, que é empregado para melhorar a solução inicial obtida. Por fim, emprega a regra de combinador algébrico para a técnica de *ensemble*, de forma a combinar os resultados, obtendo uma solução final.

Ao longo dessa pesquisa foi realizada uma revisão bibliográfica sobre o problema de roteamento de veículos e a técnica de *ensemble*, seguida do desenvolvimento de um modelo computacional usando dados do *Benchmark Loggi BUD*, como base para teste. Os resultados são analisados, interpretados e comparados.

1.1 Contribuições e Justificativas

A gestão do transporte de carga é uma das principais preocupações das empresas, por sua importância operacional e pelo peso nos custos logísticos das empresas. A atividade representa mais da metade dos gastos das empresas com logística, com forte ênfase na distribuição. Os gastos de transportes no Brasil são significativos, mesmo comparativamente a países com ampla área territorial e grande volume de produção [4].

O estudo “Custos Logísticos no Brasil”, da ILOS¹, mostra que os custos logísticos representam 12,2% do PIB brasileiro (Produto Interno Bruto). Esse percentual é bem maior que em outros países. Por exemplo quando comparado com custo logístico nos Estados Unidos que o corresponde a 7,8% do PIB.

A maior parte do custo é formada pelo transporte, que equivale a 6,8% do PIB (R\$ 401 bilhões). Depois vêm estoque (4,5% do PIB, ou R\$ 268 bilhões); armazenagem (0,9% do PIB ou R\$ 53 bilhões); e administrativo (0,5% do PIB, ou R\$ 27 bilhões), segundo Agência CNT Transporte [3].

Considerando a importância de reduzir esses custos em todas as etapas, que englobam desde a produção até a venda de uma mercadoria, o problema de roteamento de veículos

¹ILOS - Especialistas em Logística e Supply Chain é uma empresa referência no Brasil no planejamento, estruturação e implementação de atividades relacionadas à logística e ao Supply Chain.

tornou-se uma área altamente estudada na indústria de logística nas últimas décadas. Pesquisas indicam que um considerável percentual do valor da mercadoria que chega aos clientes é de exclusiva responsabilidade dos gastos obtidos através de sua distribuição. Este valor é estimado no intervalo de 10% a 15% do preço dos produtos. O objetivo das pesquisas em torno dos PRVs é reduzir este percentual de gastos com transporte a um nível mais satisfatório, possibilitando reduções de preço nos produtos finais [90]. Muitos esforços têm sido dedicados para encontrar solução ótima para o problema, e várias técnicas têm sido estudadas e propostas. A crescente busca por soluções eficientes para problemas de otimização combinatória como o PRV, motiva os pesquisadores no sentido de desenvolver algoritmos eficientes dotados de heurísticas sofisticadas, que independente de mudanças no cenário, respondam de maneira satisfatória e com um custo computacional aceitável [53].

Os trabalhos relacionados indicam que existem várias pesquisas sobre problema de roteamento de veículos. Contudo, foram encontrados poucos trabalhos sobre otimização de roteamento de veículos onde é aplicada a técnica de *ensemble*. Assim, pode se destacar a contribuição significativa desta pesquisa na área de logística e distribuição de cargas e para o desenvolvimento de soluções mais eficazes para o problema de roteamento de veículos.

1.2 Problema de Pesquisa

Dado um conjunto de dados geográficos de entrada com coordenadas de latitude, longitude e demanda, como distribuir os dados entre n conjuntos, de modo que o somatório das demandas dos pontos de dados dentro de cada conjunto seja o maior/menor possível e entre os conjuntos seja o mais equilibrado possível em termos de quantidade de pontos distribuídos entre os conjuntos.

1.3 Objetivos

O objetivo geral deste estudo é desenvolver e aplicar uma abordagem para a análise de dados espaciais, utilizando métodos heurísticos e a técnica de *ensemble*, a fim de otimizar o processo de roteirização de uma frota de veículos.

Para tanto, tem-se os seguintes objetivos específicos:

1. Identificar dados relevantes e variáveis críticas para a roteirização, explorando a literatura para identificar as melhores práticas e metodologia em análise de dados espaciais e roteirização de frotas.
2. Implementar algoritmos de agrupamento (*K-means* e DBSCAN) para agrupar pontos de entrega.

3. Interpretar e discutir resultados em termos de eficiência e equilíbrio das rotas.

1.4 Metodologia

A metodologia a ser utilizada durante execução deste trabalho consistiu em:

1. Revisão Bibliográfica:

- Realizar uma revisão bibliográfica sobre o problema de roteamento de veículos, explorando suas variações e propostas para a sua resolução. Além disso, estudar a técnica de *ensemble* e suas aplicações em problemas de otimização.

2. Metodologia de seleção de artigos relacionados:

- O procedimento de busca teve como objetivo identificar estudos a serem considerados para inclusão ou exclusão no conjunto final da revisão, de acordo com a área de interesse. O plano de pesquisa envolveu uma investigação automatizada em várias fontes acadêmicas, incluindo o Portal de Periódicos da CAPES, *Scielo* Brasil, Google Acadêmico e Repositórios das Universidades Federal do Rio Grande e do Rio Grande do Sul. Para garantir a relevância dos estudos selecionados, foram estabelecidos critérios de inclusão, tais como abrangência temporal dos artigos (publicações entre 2018 e 2023) e a apresentação de conceitos relevantes relacionados ao tema da pesquisa.

3. Implementação dos algoritmos:

- Implementar os algoritmos de clusterização *K-means* e DBSCAN para agrupar os pontos de entrega com base em suas localizações geográficas;
- Empregar as heurísticas de vizinho mais próximo para criar soluções iniciais para as estratégias desenvolvidas, e aplicar o método 2-Opt para aprimorar as soluções iniciais obtidas;
- Aplicar a regra de combinação algébrica da técnica de *ensemble*, obtendo uma solução final.

4. Análise e discussão dos resultados:

- Interpretar e discutir os resultados obtidos a partir dos testes, analisando aspectos como a eficiência das soluções, o equilíbrio das distâncias percorridas e a redução dos custos operacionais.

1.5 Estrutura do Trabalho

Este trabalho está estruturado em 6 capítulos. No Capítulo 2, é apresentado o embasamento teórico que serviu como base para fundamentar a proposta do trabalho. O Capítulo 3 descreve detalhadamente as principais técnicas utilizadas para solucionar o problema de roteamento de veículos, incluindo métodos exatos, heurísticas e meta-heurísticas. O Capítulo 4 apresenta algoritmos e técnicas utilizados para resolver problema propostos. O Capítulo 5 apresenta os resultados obtidos por meio da implementação e testes das estratégias propostas. Finalmente, o Capítulo 6 apresenta as conclusões da pesquisa realizada, além de propor trabalhos futuros a serem desenvolvidos.

2 EMBASAMENTO TEÓRICO

O presente capítulo apresenta os conceitos fundamentais que foram utilizados na construção da nova heurística para o Problema de Roteamento de Veículos (PRV). Para isso, é discutida a abordagem *ensemble*, bem como o problema de roteamento de veículos em si.

2.1 Método *Ensemble*

O método de *ensemble* é um paradigma de aprendizado em que propostas alternativas de solução para um problema, denominadas componentes, têm suas saídas individuais combinadas na obtenção de uma solução final.

Na estatística e na computação, os métodos *ensemble* constituem um conjunto de técnicas que combinam múltiplos algoritmos de aprendizagem para obter um desempenho preditivo superior ao de algoritmos individuais [65]. Figura 1 ilustra a estrutura de *ensemble*, onde N classificadores recebem os mesmos dados de entrada e geram resultados individuais, que são combinados em uma única saída. Esses classificadores são chamados de classificadores base, ou também classificadores fracos.

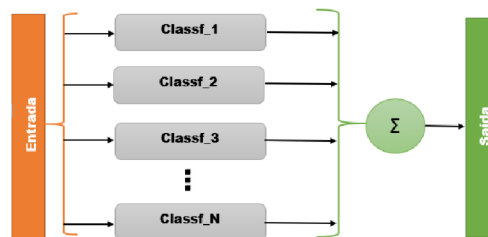


Figura 1: Estrutura ensemble

Tais classificadores normalmente são o resultado de um processo de treinamento, por um algoritmo de classificação, como Árvore de Decisão, Regressão Logística, *Naive Bayes*, Redes Neural Artificial, *Random Forest* ou outros tipos de algoritmos de aprendizagem existentes. A maioria dos métodos *ensemble* utiliza um único algoritmo de classificador singular para produzir aprendizes de base homogêneos, ou seja, aprendizes do mesmo tipo, o que resulta em conjuntos homogêneos. Há também alguns métodos que uti-

lizam diferentes algoritmos de aprendizagem para produzir aprendizes heterogêneos, ou seja, aprendizes de diferentes tipos, levando a conjuntos heterogêneos [8]. Estes métodos serão discutidos na seção 2.1.2.

O paradigma *ensemble* originou-se da proposta feita em Nilsson and Machines [63]. Nestes primeiros trabalhos, a ideia geral era combinar diferentes estimadores para a melhoria final de um sistema de classificação. Quase duas décadas depois, Hansen and Salamon [40], mostraram que a combinação de várias redes neurais artificiais, treinadas separadamente, podem melhorar significativamente a capacidade de generalização do sistema. O argumento principal era que, esses problemas envolvem mais de um mínimo local. Por essa razão, a utilização de uma única rede, embora apresente bons resultados, poderia estar muitas das vezes restrita a esses pontos de mínimo local. Dessa forma, a solução poderia ser potencializada caso outras redes fossem avaliadas em conjunto, ao invés de descartadas após a escolha da melhor delas [31].

Mas foi o trabalho de Schapire que colocou os sistemas de *ensemble* no centro da pesquisa de aprendizado de máquina, quando ele provou que um classificador forte, no sentido provavelmente aproximadamente correto (*PAC – Probably Approximately Correct*), ou seja, que tem alta probabilidade de ser aproximadamente correto, pode ser gerado pela combinação de classificadores fracos por meio de um procedimento que ele chamou de *boosting* [79].

Desde esses trabalhos seminais, a pesquisa em sistemas de *ensemble* se expandiu rapidamente, aparecendo frequentemente na literatura sob muitos designações e diferentes abordagens. A longa lista inclui sistemas de classificadores compostos [27]; mistura de especialistas [43]; generalização empilhada [91]; combinação de vários classificadores [73] [48] [92]; classificador dinâmico seleção [92]; fusão de classificadores [19] [47]; *ensembles* de classificadores [69], entre muitos outros.

A sua construção é feita geralmente em três etapas: geração, seleção e combinação, conforme Figura 2, onde:

- Geração de um conjunto de elementos candidatos a fazerem parte do *ensemble*;
- Seleção dos candidatos que realmente são necessários e que contribuem de alguma forma ao serem inseridos no *ensemble*; e
- Combinação das saídas geradas por esses elementos selecionados (componentes) em uma única saída, que corresponderá ao resultado final do *ensemble*.

2.1.1 Regras de combinação

Geralmente a formação de um grupo de classificadores, onde a previsão final é obtida através da combinação das suas previsões individuais parte do pressuposto de que a união das diversas opiniões, classificadores em um determinado assunto, tende a gerar uma melhor decisão, do que gerada por um único componente. Existem diferentes propostas

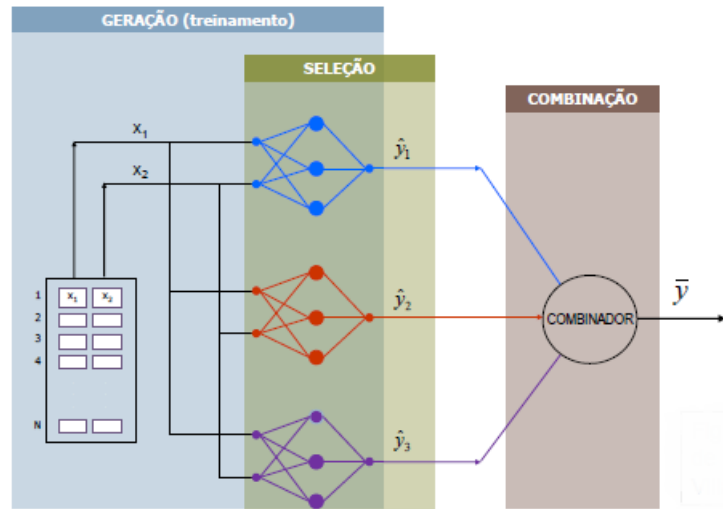


Figura 2: Etapas de construção *ensemble*

na literatura de maneira como a previsão final é obtida. A seguir são apresentadas as propostas de [88] e [93].

2.1.1.1 Classificação de Xu, Krzyzak e Suen

Os autores apresentam três categorias de acordo com níveis de informação disponíveis para o problema de combinação de classificadores:

1. Saída de nível abstrato, em que cada classificador gera um rótulo de classe exclusivo para cada padrão de entrada.
2. Saída de nível de classificação, em que cada classificador gera uma lista de rótulos de classe classificados para cada padrão de entrada.
3. Medida - saída de nível, onde cada classificador gera um vetor de medidas de valor contínuo que podem representar estimativas de probabilidades *a posteriori* de classe ou valores de confiança relacionados à classe que representam o suporte para as possíveis hipóteses de classificação.

Combinando rótulos de classe

Se as saídas dos classificadores combinados estiverem na forma de rótulos de classe, eles poderão ser combinados usando qualquer um dos métodos a seguir [70]:

1. Votação majoritária: nesta abordagem, cada classificador tem voto igual e a classificação mais popular é aquela escolhida pelo conjunto como um todo. A variante mais básica é o voto por pluralidade, onde a classe com mais votos vence. Em geral, temos três versões de votação por maioria, conforme indicado abaixo.
 - (a) Um em que todos os classificadores concordam (votação unânime).

- (b) Previsto por pelo menos um a mais da metade do número de classificadores (maioria simples).
 - (c) Aquele que recebe o maior número de votos, quer a soma desses votos exceda ou não 50% (votação por pluralidade ou apenas votação por maioria).
2. Votação por maioria ponderada se tivermos evidências de que certos especialistas são mais qualificados do que outros, ponderar mais fortemente as decisões desses especialistas qualificados pode melhorar ainda mais o desempenho geral.

Combinando saídas contínuas

A saída contínua é interpretada como o grau de suporte dado a uma classe por um classificador, geralmente aceito como o valor da probabilidade posterior para aquela classe [70]. Combinadores algébricos são usados para mesclar as decisões dos classificadores em formato de saída contínua.

2.1.1.2 Classificação de Puma-Villanueva

Este classificador apresenta quatro maneiras diferentes em como o resultado final é obtido através da combinação das suas saídas individuais, são eles:

- Voto majoritário;
- *Winner-takes-all*;
- Média simples;
- Média ponderada.

Voto majoritário

Na votação majoritário, vários modelos são usados para fazer previsões para cada ponto de dados. As previsões de cada modelo são consideradas como um voto. As previsões que obtemos da maioria dos modelos são usadas como a previsão final [81].

Winner-takes-all

É uma técnica de combinação não-linear e elitista, onde a saída do *ensemble* corresponde à saída do componente que possuir maior certeza.

Média simples

Semelhante à técnica de votação máxima, várias previsões são feitas para cada ponto de dados na média. Nesse método, utiliza-se uma média das previsões de todos os modelos e a utiliza-se para fazer a previsão final. A média pode ser usada para fazer previsões em problemas de regressão ou ao calcular probabilidades para problemas de classificação [81].

Média ponderada

Esta é uma extensão do método de média simples. Todos os modelos recebem pesos diferentes, definindo a importância de cada modelo para a previsão.

2.1.2 Diversidade

O sucesso de um sistema *ensemble* - ou seja, sua capacidade de corrigir os erros de alguns de seus membros - reside diretamente na diversidade dos classificadores que compõem o *ensemble*. Afinal, se todos os classificadores fornecessem a mesma saída, não seria possível corrigir um possível erro. Portanto, classificadores individuais em um sistema de *ensemble* precisam cometer diferentes erros em diferentes instâncias. A intuição, então, é que se cada classificador cometer erros diferentes, uma combinação estratégica desses classificadores pode reduzir o erro total. Especificamente, um sistema *ensemble* precisa de classificadores cujos limites de decisão sejam adequadamente diferentes dos outros. Tal conjunto de classificadores é dito diverso [69].

Segundo Polikar [69], a diversidade de classificadores pode ser alcançada de várias maneiras. Preferencialmente, as saídas dos classificadores devem ser condicionalmente independentes de classe ou, melhor ainda, negativamente correlacionadas. O método mais popular é usar diferentes conjuntos de dados de treinamento para treinar classificadores individuais. Esses conjuntos de dados geralmente são obtidos por meio de técnicas de reamostragem, como *bootstrapping* ou *bagging* (abaixo apresentados em detalhes). Nesses métodos, subconjuntos aleatórios, geralmente com reposição, são extraídos dos dados de treinamento originais. O propósito é garantir que os limites individuais dos classificadores sejam adequadamente diferentes, mesmo quando se utiliza conjuntos de treinamento bastante semelhantes.

Outra abordagem para alcançar a diversidade é o uso de diferentes parâmetros de treinamento para diferentes classificadores. Por exemplo, uma série de redes neurais multicamada perceptron (MLP) pode ser treinada usando diferentes inicializações de peso, número de camadas/nós, metas de erro, etc. O ajuste desses parâmetros permite controlar a instabilidade dos classificadores individuais e, portanto, contribuir à sua diversidade. Alternativamente, tipos totalmente diferentes de classificadores, como MLPs, árvores de decisão, classificadores de vizinhos mais próximos e máquinas de vetores de suporte também podem ser combinados para aumentar a diversidade. Por fim, a diversidade também pode ser alcançada usando diferentes recursos ou diferentes subconjuntos de recursos existentes [69].

2.1.3 Diferentes métodos *ensemble*

A maioria dos métodos *ensemble* utiliza um único algoritmo de classificação para produzir aprendiz de base. Também há alguns métodos *ensemble* que utilizam diferentes algoritmo de classificação para aprendizes fraco ou base. Estas duas possibilidades para criar modelos e como estes modelos serão diferentes um do outro são chamados de método *ensemble* homogêneo e heterogêneo. No *ensemble* homogêneo todos os modelos individuais são construídos usando o mesmo algoritmo de classificação para formar aprendiz de base, as técnicas mais populares deste método são *Bagging* e *Boosting*. A segunda possibilidade de construir vários modelos é construir diferentes modelos de aprendizado de máquina. Cada modelo será diferente, mas usa os mesmos dados de treinamento, neste método a técnica *Stacking* é usado como referência.

2.1.3.1 *Bagging*

O *Bagging* foi um dos primeiros algoritmos utilizados em *ensemble*, proposto por Breiman [15]. Consiste em gerar subconjuntos de entradas através de sorteio com reposição, sobre o conjunto de dados de treinamento original. Cada subconjunto amostrado é utilizado para a construção de um novo classificador. A classificação final é realizada por um sistema de votação. Portanto, o método *bagging* busca reduzir a variância do conjunto através da expansão da quantidade de classificadores, conforme Figura 3. Os métodos de *bagging* podem ser usados tanto para problemas de classificação quanto para problemas de regressão. Se estivermos usando o método de *bagging* do método de classificação, usamos a abordagem de votação por maioria para a previsão final. Considerando que para o tipo de problemas de regressão, tomamos a média de todos os valores previstos pelos modelos individuais [68].



Figura 3: *Bagging*

Algoritmos utilizados em *Bagging*

- O meta-estimador de ensacamento do inglês (*Bagging meta-estimator*) é um algoritmo de agrupamento que pode ser usado para problemas de classificação e regressão. Ele segue a técnica típica de ensacamento para fazer previsões. A seguir estão as etapas para o algoritmo meta-estimador de *bagging*:

1. Subconjuntos aleatórios são criados a partir do conjunto de dados original (*Bootstrapping*).
 2. O subconjunto do conjunto de dados inclui todos os recursos.
 3. Um estimador de base especificado pelo usuário é ajustado em cada um desses conjuntos menores.
 4. As previsões de cada modelo são combinadas para obter o resultado final.
- *Random Forest* é outro algoritmo de aprendizado de máquina *ensemble* que segue a técnica de *bagging*. É uma extensão do algoritmo do estimador de ensacamento. Os estimadores básicos na floresta aleatória são árvores de decisão. Ao contrário do *Bagging* Meta-estimador, o *Random Forest* seleciona aleatoriamente um conjunto de recursos que são usados para decidir a melhor divisão em cada nó da árvore de decisão [81]. Olhando passo a passo, é isso que um modelo de floresta aleatória faz:

1. Subconjuntos aleatórios são criados a partir do conjunto de dados original (inicialização).
2. Em cada nó da árvore de decisão, apenas um conjunto aleatório de características é considerado para decidir a melhor divisão.
3. Um modelo de árvore de decisão é ajustado em cada um dos subconjuntos.
4. A previsão final é calculada pela média das previsões de todas as árvores de decisão.

2.1.3.2 *Boosting*

Boosting foi proposto por SCHAPIRE [79], nesta técnica o treinamento dos classificadores que compõem o *ensemble* deve ser feito de maneira sequencial. A probabilidade de uma dada amostra ser escolhida depende da contribuição desta para o erro dos classificadores já treinados, conforme Figura 4. Nesta abordagem cada classificador é treinado sequencialmente e a distribuição dos dados de treinamento é feita levando em consideração os erros do classificador anterior [9].

Algoritmos mais utilizados em *Boosting*

- O reforço adaptativo ou *AdaBoost* é um dos algoritmos de reforço mais simples. Normalmente, as árvores de decisão são usadas para modelagem. Vários modelos sequenciais são criados, cada um corrigindo os erros do último modelo. O *AdaBoost* atribui pesos às observações que são previstas incorretamente e o modelo subsequente trabalha para prever esses valores corretamente [81].

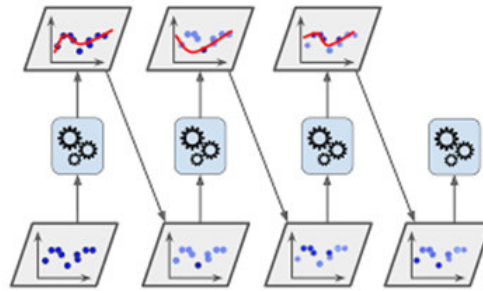


Figura 4: *Boosting*

Abaixo estão as etapas para executar o algoritmo *AdaBoost*:

1. Inicialmente, todas as observações no conjunto de dados recebem pesos iguais.
2. Um modelo é construído em um subconjunto de dados.
3. Usando este modelo, as previsões são feitas em todo o conjunto de dados.
4. Os erros são calculados comparando as previsões e os valores reais.
5. Ao criar o próximo modelo, pesos maiores são dados aos pontos de dados que foram previstos incorretamente.
6. Os pesos podem ser determinados usando o valor de erro. Por exemplo, quanto maior o erro maior é o peso atribuído à observação.
7. Este processo é repetido até que a função de erro não mude, ou o limite máximo do número de estimadores seja atingido.

2.1.3.3 *Stacking*

Aggarwal [2] caracteriza o *Stacking* como um meta-algoritmo de aprendizado de máquina. O meta-aprendizado consiste em aprender em um conjunto de metadados ao invés de aprender em um conjunto de dados original [51]. Na execução da técnica de *Stacking*, um conjunto de n algoritmos, chamados de classificadores base, são treinados na base de dados gerando n modelos. Cada modelo será utilizado para prever as classes dos exemplos de treinamento. Porém, ao invés de utilizar as previsões diretamente para realizar a classificação, as previsões são utilizadas para gerar uma nova representação do exemplo. Uma vez gerada as novas representações dos exemplos, um novo algoritmo de aprendizado é executado utilizando na nova base de dados, na qual as probabilidades informadas por cada algoritmo de primeiro nível se tornam as características a serem avaliadas por esse novo algoritmo (meta-classificador), conforme Figura 5 [2].

Os resultados empíricos [21] mostram que o *Stacking* possui um desempenho robusto que muitas vezes supera o modelo bayesiano médio. Desde a sua introdução [41], *Stacking* foi aplicado com sucesso a uma grande variedade de problemas, como regressão

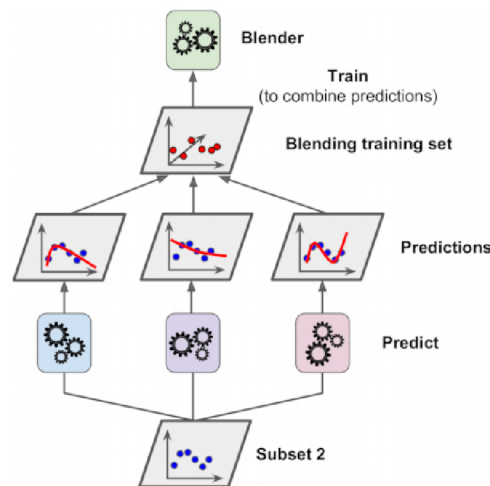


Figura 5: *Stacking*

[15][29], estimativa de densidade [41] e filtragem de spam [74]. Além de usar rótulos de classe previstos de classificadores de primeiro nível. [85] Consideram o uso de probabilidades de classe como características. A vantagem de usar probabilidades condicionais como características é que o conjunto de treinamento do classificador de segundo nível incluirá não apenas previsões, mas também confiança de previsão de classificadores de primeiro nível. Os mesmos autores sugerem ainda o uso de regressão linear multi-resposta, uma variante de regressão linear de mínimos quadrados, como o algoritmo de aprendizado de segundo nível [85].

2.2 Problema de Roteamento de Veículos

O Problema de Roteamento de Veículos em inglês (*Vehicle Routing Problem* ou *VRP*) pode ser considerado como um conjunto organizado de meios que objetiva o atendimento de demandas localizadas nos arcos ou nos vértices de alguma rede de transportes.

Os problemas de roteamento de veículos abordam basicamente a determinação de sequências de visitas que objetivem atender a uma determinada função objetivo. As visitas podem tanto estar associadas às ligações (arestas) ou aos pontos de visitas (nós) do grafo que representa as possíveis conexões entre os pontos de visita (ou pontos de ligações entre as arestas). Os problemas de roteamento de veículos estão entre os mais complexos da área de otimização combinatória. Pelo grande número de variáveis, diversidade de restrições e objetivos apresentados impõem-se o exame de uma cuidadosa taxonomia para seu melhor entendimento [38].

Foi introduzido em 1959 por Dantzig and Ramser [26], com o artigo “*The Truck Dispatching Problem*”, cujo sua fundamentação segundo autores está em definir um conjunto de rotas a serem percorridas por veículos, reduzindo os custos com transportes até determinados consumidores, dado à distância, e obedecendo que:

1. Cada rota começa e termina num ponto inicial (depósito);
2. Todo consumidor é visitado somente uma vez por somente um veículo e
3. A demanda total de qualquer rota não pode ultrapassar capacidade de um veículo.

De acordo com Vieira [87], o Problema de Roteamento de Veículos consiste em determinar um roteiro (conjunto de rotas) a ser seguido por uma frota de veículos, de modo que a demanda de todos os clientes seja satisfeita, e que cada veículo regresse ao depósito de origem ao final do percurso. O objetivo é minimizar o custo total, o tempo do percurso ou distância total do percurso.

A Figura 6 ilustra um exemplo de problema de roteamento de veículos. Na imagem do lado esquerdo tem uma rede com todas as cidades conectadas entre si, um depósito onde estão os itens e a frota dos veículos, também pode ser visto como um cliente que possui uma demanda. Após a execução de um algoritmo de resolução do problema de roteamento de veículos, tem-se uma saída que parecia algo como a imagem no lado direito onde tem cinco rotas diferentes ou seja cinco veículos fazendo entregas e cada um desses veículos passa em uma cidade sem repetir para atender as demandas de 25 clientes geograficamente dispersos.

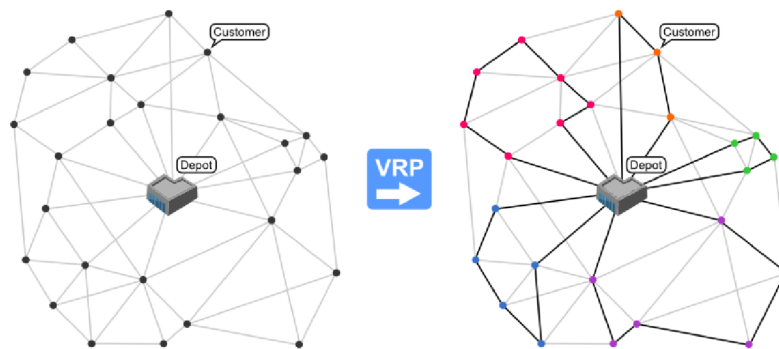


Figura 6: Demonstração visual do problema de roteamento de veículos com 5 veículos.

O PRV é um importante problema de otimização combinatória [1] no qual busca-se encontrar a configuração ideal de rotas para uma frota de veículos que deve realizar entregas a partir de uma origem para um determinado conjunto de clientes [13]. A restrição presente no problema de roteamento de veículos é que cada veículo v possui uma capacidade C_v e o somatório de todas as demandas dos consumidores atendidos por um veículo v não pode ultrapassar C_v . Ele é comprovadamente considerado problema NP-difícil, embora existam algoritmos exatos capazes de encontrar soluções ótimas em um tempo razoável

¹ A Otimização Combinatória é um ramo da ciência da computação e da matemática aplicada que estuda problemas de otimização em conjuntos finitos.

para instâncias relativamente pequenas, esses problemas são computacionalmente difíceis e requerem alto custo computacional.

Segundo explicação de Bastos Junior [7], o problema de roteamento de veículos pode ser definido da seguinte forma: Seja $G = (V, A)$ um grafo onde $V = \{0, \dots, n\}$ é um conjunto de vértices representando localidades (clientes ou cidades) com o depósito localizado no vértice 0, e A é o conjunto de arcos. Cada arco (i, j) , $i \neq j$, é associado a uma matriz de distâncias $C = (C_{ij})$ não negativas. Em alguns contextos, C_{ij} também pode ser interpretado como o custo de viagem ou o tempo de viagem. Quando C é simétrico (isto é, a distância/tempo/custo de i para j é o mesmo de j para i), é conveniente substituir A por um conjunto E de arcos não direcionados. Além disso, assumimos que existem K veículos disponíveis no depósito, onde $K_L \leq K \leq K_U$. Quando $K_L = K_U$ é dito ser fixo. Quando $K_L = 1$ e $K_U = n - 1$, K é dito ser livre. Quando K não é fixo, faz sentido associar um custo fixo ao uso do f veículo.

Como simplificação, Laporte [49] ignorou estes custos, e partiu do princípio de que todos os veículos são idênticos e têm a mesma capacidade Q . O PRV consiste em planejar um conjunto de rotas de menor custo do veículo, de tal forma que:

1. Cada vértice em $V \neq 0$ é visitado apenas uma vez por exatamente um veículo;
2. Todas as rotas se iniciam e terminam no depósito;
3. As seguintes restrições devem ser respeitadas:
 - Restrição de capacidade: a cada vértice $i > 0$ é atribuído um peso não-negativo (ou demanda) d_i e a soma dos pesos de qualquer rota do veículo não pode exceder a capacidade do veículo;
 - O número de vértices em cada rota é limitado a q (este é um caso especial de (a) com $d_i = 1$ para todo $i > 0$ e $Q = q$);
 - Restrição de tempo total: o comprimento de qualquer rota não pode exceder um limite fixado L , sendo este comprimento constituído pelos tempos de viagem c_{ij} e pelos tempos de parada s_i em cada vértice i da rota;
 - Janelas de tempo: o vértice i deve ser visitado dentro do intervalo de tempo $[a_i, b_i]$ e é permitido tempo de espera no vértice i ;
 - Precedência entre pares de vértices: o vértice i pode ter de ser visitado antes do vértice j ;

Pode se encontrar uma série de outras variantes na literatura.

Considere a seguinte formulação:

Variáveis:

$$x_{ijk} = \begin{cases} 1, & \text{se o veículo } k \text{ percorre o arco } (i, j), \forall k \in K, \forall (i, j) \in E \\ 0 & \text{caso contrário} \end{cases}$$

$$\min \sum_{k \in K} \sum_{(ij) \in E} c_{ij} x_{ijk} \quad (1)$$

$$\sum_{k \in K} \sum_{j \in E} x_{ijk} = 1, \forall i \in C \quad (2)$$

$$\sum_{i \in C} d_i \sum_{j \in N} x_{ijk} \leq Q, \forall k \in K \quad (3)$$

$$\sum_{i \in N} \sum_{j \in N} 1_{ij} x_{ij} \leq D, \forall k \in K \quad (4)$$

$$\sum_{j \in N} x_{0jk} = 1, \forall k \in K \quad (5)$$

$$\sum_{i \in N} x_{ihk} - \sum_{j \in N} x_{hjk} = 0, \forall h \in C, \forall k \in K \quad (6)$$

$$\sum_{i \in N} x_{i,n+1,k} = 1, \forall k \in K \quad (7)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ijk} \leq |S| - 1, S \subset C, 2 \leq |S| \leq \lfloor \frac{n}{2} \rfloor, \forall k \in K \quad (8)$$

$$x \in B^{K|E|} \quad (9)$$

A função objetivo (Eq. [1]) representa a minimização do custo total das rotas. As restrições (Eq. [2]) asseguram que cada cliente i é designado a um único veículo. As restrições (Eq. [3]) impõem que a demanda total de cada rota do veículo k não excede a capacidade Q do veículo. As restrições (Eq. [4]) garantem que a duração de cada rota do veículo k não excede o limite D . As restrições (Eq. [5]), (Eq. [6]) e (Eq. [7]) representam restrições de fluxo em redes, que exigem que cada veículo k parta do depósito (nó 0) somente uma vez, deixe o nó h se e somente se entrar neste nó, e retorne ao depósito (nó $n + 1$) somente uma vez. As restrições (Eq. [7]) são redundantes, mas são mantidas no modelo para enfatizar a estrutura de redes. As restrições (Eq. [8]) garantem a não-existência de sub-rotas, e a restrição (Eq. [20]) indica o tipo de variável.

Se o número de veículos no modelo (Eq. [1])-(Eq. [20]) for um parâmetro fixo ou um limitante superior igual a K , remova o arco $(0, n + 1)$. Caso contrário, se o número de veículos é uma variável, atribua um custo c_v a cada veículo usado. Isso é feito impondo-se

$c_{0,n+1} = c_v$. Se c_v é grande, o modelo primeiramente minimiza o número de veículos e, em segundo lugar, minimiza o custo de viagem.

2.2.1 Variações e Especializações do Problema de Roteamento de Veículos

Existem diversas versões e especializações do problema de roteamento de veículos, que variam em relação às suas restrições e objetivos. Algumas das principais versões mais usadas são apresentadas na Tabela 1.

Tabela 1: variações de PRV

Variante	Definição
Problema de Roteamento de Veículos com Lucros (VRPP)	<i>Um problema de maximização onde não é obrigatório visitar todos os clientes. O objetivo é visitar uma vez os clientes maximizando a soma dos lucros arrecadados respeitando o limite de tempo de um veículo.</i>
Problema de roteamento de veículos com coleta e entrega (VRPPD)	<i>Várias mercadorias precisam ser movidas de determinados locais de coleta para outros locais de entrega. O objetivo é encontrar rotas ideais para uma frota de veículos visitar os locais de embarque e desembarque.</i>
Problema de Roteamento de Veículos com LIFO	<i>Semelhante ao VRPPD, exceto que uma restrição adicional é colocada no carregamento dos veículos: em qualquer local de entrega, o item sendo entregue deve ser o item retirado mais recentemente. Esse esquema reduz os tempos de carga e descarga nos locais de entrega, pois não há necessidade de descarregar temporariamente outros itens além daqueles que devem ser deixados.</i>
Problema de Roteirização de Veículos com Janelas de Tempo (VRPTW)	<i>Os locais de entrega possuem janelas de tempo dentro das quais as entregas (ou visitas) devem ser feitas.</i>
Problema de Roteamento de Veículos com Múltiplas Viagens (VRPMT)	<i>Os veículos podem fazer mais de uma rota.</i>
Problema de Roteamento de Veículo Aberto (OVRP)	<i>Os veículos não são obrigados a retornar ao depósito.</i>
Problema de roteamento de veículos com vários depósitos (MDVRP)	<i>Existem vários depósitos a partir dos quais os veículos podem começar e terminar.</i>
Problema de Roteamento de Inventário (IRP)	<i>Os veículos são responsáveis por atender as demandas em cada ponto de entrega.</i>
Problema de Roteamento de Veículo Capacitado (CVRP)	<i>Os veículos têm uma capacidade limitada de carga das mercadorias que devem ser entregues.</i>

2.3 Problema de roteamento de veículos capacitado (PRVC)

O Problema do Roteamento de Veículos Capacitado (PRVC) é uma variação do Problema do Roteamento de Veículos (PRV), como mostrado na Tabela I, em que os veículos possuem uma capacidade limitada para transportar mercadorias. Esse problema é bastante comum em empresas de logística, distribuição e transporte, onde os veículos transportam uma certa quantidade de carga de um local para outro.

De acordo com GOLBARG and LUNA [38], o problema de roteamento de veículos capacitados é relatado com uma frota de m veículos de capacidade W e sediados em ponto garagem. Nesse caso o objetivo é atender a um conjunto de pontos de demanda localizados sobre os vértices do grafo $G = (N, A)$ de modo a minimizar o comprimento total das rotas dos veículos.

No PRVC, é necessário determinar as rotas a serem percorridas por um conjunto de veículos para entregar as cargas nos clientes, levando em conta as capacidades dos veículos e as demandas de cada cliente. O objetivo é minimizar o custo total do transporte, que pode incluir a distância percorrida pelos veículos, o tempo necessário para realizar as entregas, o número de veículos necessários e outros fatores relevantes. Assim como o PRV, o PRVC é um problema NP-difícil.

Laporte [49] formula formalmente, o problema de roteamento de veículos capacitado da seguinte maneira:

Dado um grafo $G = (V, E)$, onde $V = \{v_1, v_2, v_3, \dots, v_n\}$ corresponde aos pontos de demanda e $E = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ o conjunto de arestas entre os pontos de demanda.

A cada cliente $v_i \in V$ existe uma demanda $q_i \in Q$ associada que deve ser suprida por algum veículo. Ao conjunto E está associado uma matriz simétrica de custos não negativos C que representa o custo necessário a percorrer dois nós i e j , sendo $c_{ij} = c_{ji}$. Para tanto se devem respeitar as seguintes restrições:

- Cada cidade $v \in V - 1$ deve ser visitada exatamente uma única vez.
- Todas as rotas devem iniciar e terminar no depósito.
- Atender as restrições de capacidade dos veículos.

De acordo com abordagem de Fisher and Jaikumar [34], a formulação matemática do problema de roteamento de veículos capacitado pode ser expressa de seguinte maneira:

$$\text{Min} \sum_{i,j} \sum_k c_{i,j} x_{ijk} \quad (10)$$

Sujeito a:

$$\sum_k y_{ik} = 1, i = 2 \dots n \quad (11)$$

$$\sum_k y_{ik} = m, i = 1 \quad (12)$$

$$\sum_i q_i y_{ik} \leq Q_k, k = 1 \dots m \quad (13)$$

$$\sum_j x_{ijk} = \sum_j x_{jik} = y_{ik}, i = 1, \dots, n, k = 1, \dots, m \quad (14)$$

$$\sum_j x_{ijk} \leq |S| - 1, \forall S \subseteq \{2, \dots, n\}, k = 1, \dots, m \quad (15)$$

$$y_{ik} \in \{0, 1\}, i = 1, \dots, n, k = 1, \dots, m \quad (16)$$

$$x_{ijk} \in \{0, 1\}, i, j = 1, \dots, n, k = 1, \dots, m \quad (17)$$

Onde:

- x_{ijk} : variável binária que assume valor 1 quando o veículo k visita o cliente j imediatamente após o cliente i , 0 caso contrário.
- y_{ik} : variável binária que assume valor 1 se o cliente i é visitado pelo veículo k , 0 caso contrário.
- q_i : corresponde a demanda do cliente i .
- Q_k : corresponde a capacidade do veículo k .
- c_{ij} : corresponde ao custo de percorrer o trecho que vai do cliente i ao cliente j .

A função objetivo da Eq. [10](#) minimiza a distância total da solução, calculando a soma das distâncias entre pontos de demanda presentes na solução. A restrição da Eq. [11](#) garante que cada ponto de demanda seja visitado exclusivamente por um veículo. A restrição da Eq. [12](#) garante que o depósito receba uma visita de todos os veículos. A restrição da Eq. [13](#) garante que a capacidade dos veículos não seja excedida. A restrição da Eq. [14](#) garante que uma rota tenha término sempre no depósito e não em um ponto de demanda. A restrição da Eq. [15](#) garante a não ocorrência de *subtours* nas rotas. As restrições das Eqs. [16](#) e [17](#) são as restrições de domínio das variáveis de decisão do problema.

2.4 Objetivos do Problema de Roteamento de Veículos

O problema de roteamento de veículos é um desafio fundamental enfrentado em diversas áreas, como logística, transporte e distribuição. Esse problema envolve a tarefa de otimizar a alocação de veículos para entregar um conjunto de itens ou atender a um conjunto de locais, levando em consideração várias restrições e objetivos. Ao longo das décadas, várias variantes do PRV surgiram devido a diferentes contextos e prioridades, resultando em uma ampla gama de objetivos a serem alcançados. As Seções a seguir apresentam uma revisão de alguns dos principais objetivos tratados na literatura em diversas variações do problema de roteamento de veículos.

2.4.1 Balanceamento de Rota

Alguns objetivos são projetados para equilibrar as disparidades entre percursos de rotas. Tais objetivos são frequentemente introduzidos com a finalidade de implementar elementos de equidade nos problemas de roteamento de veículos. Para definir este objetivo é necessário considerar a carga de trabalho nas rotas que podem ser expressadas como o número de cliente visitados, capacidade de carga utilizada, o custo total das rotas, a quantidade de produtos entregues e o tempo necessário [60], uma das questões que motivam o estudo deste problema refere-se a satisfação dos motoristas, uma vez que a carga de trabalho destes deve ser próxima.

Assim, Lee and Ueng [50] incorporam o equilíbrio em uma versão do PRV para melhorar a equidade de trabalho entre os motoristas. A carga de trabalho das rotas considera o tempo de viagem necessário para completar o percurso da rota e o tempo necessário para carregar e descarregar o caminhão. Deste modo, o objetivo é modelado como a minimização da soma das diferenças entre a carga de trabalho de cada rota e a menor carga de trabalho entre todas as rotas. No entanto, um modelo de programação inteira, juntamente com uma heurística são propostas para solucionar o problema.

2.4.2 Maior Rota

De acordo com Matsueda [60], vários pesquisadores propõem formulações do PRV nas quais a minimização da maior rota é considerada objetivo principal. Assim, procuram-se soluções para o problema em que o comprimento de viagem da maior rota seja a menor possível. Essa abordagem é motivado por situações em que a distância considerável entre os locais de coleta de passageiros, por exemplo, resulta em trajetos de ônibus longos e frequentemente incompletos.

O foco em minimizar a maior rota visa garantir uma certa equidade em termos de tempo de viagem no ônibus, comparando o tempo gasto pelo primeiro passageiro ao tempo gasto pelo último passageiro. Essa estratégia também leva em consideração a eficiência operacional e a satisfação do consumidor. Ao encurtar a rota mais longa,

reduzem-se o tempo necessário para manter o depósito central em funcionamento e o tempo de espera do último cliente a ser atendido.

Assim, a minimização da maior rota vai além de simples otimizar a logística; ela aborda os custos operacionais e a melhoria da experiência do cliente. Ao reduzir o percurso da rota mais extensa, essa abordagem atende tanto a aspectos práticos quanto a considerações de equidade e satisfação geral.

2.4.3 Distância Total Percorrida

Grande parte dos trabalhos da literatura, que envolvem o PRV, procuram minimizar os custos com transporte. Estes custos estão diretamente ligados à distância total percorrida pela frota de veículos (custos variáveis). Deste modo, minimizar este objetivo tem sido uma das metas mais abordadas no PRV clássico, bem como em suas variantes [60].

Achuthan et al. [1], por exemplo, procuraram resolver o problema de roteamento de veículos capacitados no qual a única meta avaliada pelos autores considera a minimização da distância total percorrida pelos veículos. Os autores propuseram um método exato em que novos planos de corte baseados em um algoritmo *branch-and-cut* eram utilizados para resolver o problema.

O objetivo é encontrar rotas para os veículos que minimizem os custos totais, como distância percorrida, tempo de viagem ou consumo de combustível.

2.5 Trabalhos Relacionados

O problema de roteamento de veículos tem sido amplamente estudado na última década principalmente na área de logística e transporte de cargas. A sua utilização como alternativa para reduzir os custos da logística é mencionada na literatura em várias pesquisas, a seguir são apresentadas.

Em Kangah et al. [44] é apresentado um método *ensemble* que combina os pontos fortes e fracos da otimização por enxame de partículas (PSO) com operadores de algoritmo genético (GA) como cruzamento e mutação para resolver o problema de roteamento de veículos. Dado que a otimização de enxame de partículas e o algoritmo genético são ambos métodos evolutivos de pesquisa heurística baseados em população, usados em muitos campos, a otimização de enxame de partículas padrão estagna as partículas mais rapidamente e converge prematuramente para soluções sub ótimas que não são garantidas como ótimas locais. Embora tanto o PSO quanto o GA sejam métodos de aproximação para um problema de otimização, esses algoritmos têm suas limitações e benefícios. Neste estudo, são feitas modificações na estrutura algorítmica original do PSO, atualizando-a com alguns operadores AG selecionados para implementar um algoritmo híbrido.

No trabalho de Lu [56] é resolvido o problema de roteamento de veículos capacitado usando técnicas baseadas em aprendizagem de máquina (ML). Os autores propuseram o *Learn to Improve* (L2I), uma solução baseada em aprendizado para CVRP que supera os algoritmos de Pesquisa Operacional (OR) em termos de velocidade de resolução. Especificamente, os autores demonstraram um algoritmo baseado em aprendizado para o CVRP, propondo uma estrutura capaz de dividir os operadores heurísticos em dois grupos diferentes para acelerar a operação e concentrar o Aprendizado por Reforço (RL) naqueles identificados como operadores de melhoria. Por fim, eles apresentaram uma técnica de *ensemble* na qual as regras de RL são ensinadas simultaneamente, gerando melhores resultados com o mesmo custo computacional.

Borowski et al. [13] definam problema de roteamento de arco capacitado incerto (UCARP) como um problema importante com muitas aplicações do mundo real. Um grande desafio na UCARP é lidar com o ambiente incerto de forma eficaz e reduzir o custo do recurso em caso de falhas de rota. A hiper-heurística de programação genética (GPHH) foi aplicada com sucesso para desenvolver automaticamente políticas de roteamento eficazes para tomar decisões em tempo real no processo de roteamento. No entanto, a maioria dos estudos existentes obtém uma única política de roteamento complexa que é difícil de interpretar. Neste artigo, pretendem desenvolver um conjunto de políticas de roteamento mais simples e interpretáveis do que uma única política complexa. Ao considerar as duas propriedades críticas da aprendizagem em conjunto, ou seja, a eficácia de cada elemento do conjunto e a diversidade entre eles, propõem duas novas abordagens de *ensemble GP*, *DivBaggingGP* e *DivNichGP*. *DivBaggingGP* evolui os elementos do conjunto sequencialmente, enquanto *DivNichGP* os desenvolve simultaneamente. Os resultados experimentais mostraram que tanto o *DivBaggingGP* quanto o *DivNichGP* podem obter políticas de roteamento mais interpretáveis do que a única política de roteamento complexa.

Wang et al. [89] consideram que o Problema de Roteamento de Veículos (PRV) é um típico problema de otimização combinatória e tem sido estudado por muitos anos. No entanto, existem poucas pesquisas sobre o Problema de Roteamento de Veículos Dinâmicos com Janela de Tempo (DVRPTW), que é uma extensão do VRP e mais desafiador com a mudança de fatores ambientais, como solicitações estocásticas de clientes. Uma vez que as mudanças acontecem, as rotas devem ser ajustadas para os novos ambientes. Neste artigo, constroem um modelo de otimização multiobjetivo para o DVRPTW e propor um novo algoritmo denominado EL-DMOEA, onde um método de aprendizagem por *ensemble* é investigado para melhorar o desempenho do algoritmo. No EL-DMOEA, para aumentar a diversidade da população e acelerar a convergência, três

estratégias diferentes, ou seja, estratégia de predição baseada na população, estratégia de imigrantes e estratégia aleatória, são empregadas no processo de treinamento de três tipos de modelos básicos, respectivamente. Os resultados experimentais nos *benchmarks* de teste revelam que o algoritmo proposto é eficaz para fazer planos de roteamento promissores.

Silva [80] apresentam o desenvolvimento de uma abordagem híbrida como uma solução para o problema de múltiplos caixeiros viajantes, (do inglês, *multiple Traveling Salesman Problem - mTSP*) aplicado ao escalonamento de rotas para veículos autônomos. Inicialmente, usam o *K-means* como um pré-processamento para gerar rotas que distribuem os locais de entrega entre os carros. Em seguida, essas rotas são definidas como população inicial para os algoritmos bio-inspirados: Algoritmo Genético (GA) e Colônia de Formigas em sua versão (ACS). Esses algoritmos executam um processo evolutivo para encontrar uma rota que minimize a distância geral, mantendo o equilíbrio das rotas individuais de cada carro. Os experimentos foram conduzidos em um sistema de escalonamento de rotas em ambientes virtuais (simulação) e em um estudo de caso no Campus 2 da Universidade de São Paulo. Nos experimentos, foram realizadas comparações das abordagens híbridas, *K-means-GA* e *K-means-ACS* com as suas versões sem pré-processamento, com a geração da população inicial de forma aleatória. Além das comparações com GA e ACS foram realizadas comparações com o algoritmo de Otimização por Enxame de Partículas (PSO). Os resultados apontam que à medida que o número de carros e locais de visita aumentam, as abordagens híbridas superam suas versões clássicas e o PSO. Para avaliação dos resultados foram aplicados um teste não paramétrico *kruskal wallis* seguido de um teste de comparações múltiplas *Dunn-Bonferroni*.

Tabela 2: Resumo dos trabalhos apresentados

Autor	Metodologia usada na abordagem	Estudo de caso
Kangah, et al. (2021)	Apresentam um método ensemble que combina os pontos fortes e fracos da otimização por enxame de partículas com operadores de algoritmo genético	<i>Implementação de um Modelo de Otimização H-PSOGA para Problema de Roteamento de Veículos.</i>
Lu et al. (2020)	Algoritmo baseado em aprendizado para CVRP	<i>Método iterativo baseado em aprendizagem para resolver problemas de rotas de veículos.</i>
Borowski, et al. (2020)	Apresentam novos métodos para resolver o VRP e sua variante mais prática, o CVRP	<i>Apresentam novos algoritmos híbridos, DBSCAN Solver e Solution Partitioning Solver, que usam recozimento quântico para resolver o Problema de Roteamento de Veículos VRP e sua variante CVRP.</i>
Chen et al. (2021)	Usam três algoritmos de conjunto competitivos: RF, AdaBoost e DTs incrementais, para treinar modelos	<i>Utilidade da computação de borda para aprimorar a eficiência e a precisão das abordagens de aprendizado ensemble.</i>
Silva et al. (2022)	Avaliam modelagens de recompensas para o algoritmo de aprendizado usado por um <i>framework</i> multiagente para otimização combinatória	<i>Uma análise da modelagem de recompensa para aprendizado por reforço em uma estrutura multiagente para otimização combinatória.</i>

3 MÉTODOS DE SOLUÇÃO

O problema de roteamento de veículos é conhecido por ser um problema NP-difícil, o que significa que encontrar a solução ótima pode levar um tempo exponencial de acordo com o tamanho da entrada. O PRV pertence à categoria de problemas NP-difíceis [64], que são algoritmos que conjecturam um grande aumento de tempo à medida que aumentam os nós, ou seja, a dificuldade de encontrar a solução aumenta rapidamente à medida que o número de nós cresce.

Portanto encontrar uma solução ótima para problema de roteamento de veículos pode ser bastante desafiadora. Vários métodos para solucionar problema de roteamento de veículos com diferentes abordagens e níveis de complexidade são propostas na literatura. Em geral esses métodos podem ser classificados em Métodos Exatos, Heurísticos e Meta-heurísticos.

Os métodos exatos apresentam sérias dificuldades no que se refere ao tempo de processamento, o que em geral acaba comprometendo a sua utilização na resolução de problemas de ordem de grandeza mais realista. Já as heurísticas são mais rápidas e não apresentam as dificuldades dos métodos exatos, no entanto não permitem garantir que a solução ótima seja alcançada. Segundo Pimentel [67], a principal diferença entre as heurísticas e os métodos exatos reside no fato de, nas heurísticas não existe uma garantia de que a solução ótima do problema seja encontrada, enquanto que os métodos exatos visam a obtenção da sua solução ótima. São discutidos métodos exatos na seção 3.1, heurísticos seção na 3.2 e meta-heurísticos na seção 3.3.

3.1 Métodos exatos

Os Métodos Exatos, para encontrar uma solução ótima de determinado problema, realizam a construção de modelos matemáticos de otimização e a implementação de algoritmos específicos [84]. Assim, a solução somente será considerada ótima se não existir outra solução melhor que a já encontrada, respeitando os critérios de otimização estabelecidos. Para isso, este tipo de método faz a verificação de todas as possibilidades possíveis para resolução do problema resultando no melhor valor encontrado, garantindo assim, o

retorno da melhor solução.

Os métodos exatos se distinguem das heurísticas em um ponto fundamental, possuem a garantia de chegar ao resultado ótimo do problema tratado, mesmo que em tempo exponencial [17]. Isso faz com que os algoritmos exatos sejam potencialmente incapazes, na prática, de resolver problemas da classe NP-difícil, como é o caso de PRVC. Esses métodos fornecem a solução ótima para problema de roteamento de veículos, mas podem ser extremamente lentos para problemas grandes. Eles geralmente envolvem a criação de modelos matemáticos que descrevem o problema de roteamento de veículos e o uso de algoritmos para resolvê-los.

Segundo Lima [53], os métodos exatos possuem tempo polinomial para encontrar a solução ótima e, em geral, resolvem apenas problemas de pequeno porte, que não refletem os problemas reais que geralmente são de grande porte.

Renaud and Boctor [72] afirmam que raramente algoritmos exatos conseguem encontrar a solução ótima para instâncias com tamanho superior a 50 pontos de demanda em um tempo aceitável.

De acordo com Toth and Vigo [86], para grandes problemas de otimização combinatoria torna-se inviável a resolução por métodos exatos, em função do alto custo computacional.

Para Laporte [49], os métodos exatos aplicados na resolução do problema de roteamento de veículos podem ser classificados em três grandes categorias:

1. Métodos de busca direta em árvore: técnicas de atribuição de limite inferior, algoritmo *branch-and-bound*, e árvore geradora central de grau K;
2. Programação dinâmica e programação linear inteira (utilizando como exemplos o particionamento de conjuntos);
3. Algoritmo de geração de colunas: formulação de um índice triplo de fluxo de veículos e a formulação de um índice duplo de fluxo de veículos.

Um número considerável de métodos exatos foi elaborado com vista à resolução de problema de roteamento de veículos, incluindo [6] [11] [66] [33] [5]. Dentre as técnicas exatas para resolução de problema de roteirização, destaca-se *branch-and-bound*, a seguir descrita.

3.1.1 *Branch-and Bound*

O Algoritmo *Branch-And-Bound* (*B&B*) é um modelo combinatório, pois ele se auxilia em outras técnicas para entregar uma solução final, estas são o Modelo Húngaro¹ e o *Branch-And-Bound*.

¹O método húngaro é um algoritmo usado em problemas de alocação quando você deseja minimizar o custo. Ou seja, é usado para encontrar o custo mínimo designando várias pessoas para várias atividades com base no menor custo. Cada atividade deve ser atribuída a uma pessoa diferente.

Segundo Christofides [20], o problema é observado primeiramente como um problema de designação (PD), onde as soluções devolvidas pelo modelo podem ou não ser viáveis ao problema de PCV. Para verificar se uma rota é ou não viável é utilizado o Algoritmo Húngaro, que irá avaliar o conjunto de soluções.

Quando os valores encontrados pelo algoritmo de designação não são viáveis, ou seja, no resultado ótimo da designação são criadas sub-rotas, o modelo variante de *Branch-And-Bound* utiliza um outro modelo como base de auxílio, o método de B&B, que irá criar novas soluções para o problema.

Para DOS REIS [30], o algoritmo *Branch-And-Bound* é um problema de otimização que consiste em criar problemas filhos, a partir do problema original, até que a solução ótima seja encontrada. Cada pai, problema original, só pode gerar dois filhos. Os problemas filhos são chamados de nós, se eles ainda não tiverem sido resolvidos eles são chamados de nós ativos. Para o problema chegar ao fim todos os nós devem ter sido resolvidos ou podados, o processo de poda funciona por três variáveis: poda por qualidade (quando um nó atende a todas as restrições, mas não é a melhor encontrada), poda por infactibilidade (quando ele não tem solução) e poda por otimalidade (quando a solução encontrada atende as restrições e a melhor solução encontrada até o momento), o pseudocódigo do Algoritmo 1 apresenta o *Branch-And-Bound*.

Algorithm 1 Algoritmo *Branch-And-Bound*

1. Criar uma matriz das distâncias entre os pontos;
 2. Aplicar o Modelo Húngaro;
 - (a) Subtraia a menor entrada da linha de todos os outros componentes da linha e repita para todas as linhas;
 - (b) Subtraia a menor entrada da coluna de todos os outros componentes da coluna e repita para toda a coluna;
 - (c) Marque a linha e coluna que só tenha zeros ou números negativos;
 - (d) Teste de otimização;
 - (e) Repita os passos até encontrar uma solução;
 3. A solução não sendo viável, aplique o método de *Branch-And-Bound*;
 - (a) Partindo da solução anterior, crie dois filhos do problema;
 - (b) Encontre os valores dos filhos, se for melhor, selecione como solução;
 - (c) Se não, realize uma poda.
 - (d) Repita o passo (a) até encontrar a melhor solução.
-

3.2 Métodos Heurísticos

As heurísticas são tidas como um conjunto de técnicas que buscam gerar uma boa solução (solução satisfatória) dentro do seu contexto de aplicação, sem a garantia de que

aquela seja a solução ótima do problema objeto de interesse.

Segundo GOLBARG and LUNA [38], as heurísticas têm um papel fundamental na pesquisa do problema de roteamento de veículos. A maioria dos sistemas considerados competitivos na literatura as utilizam em algum momento de suas buscas por soluções de qualidade.

Nicholson [62], define heurística como um procedimento para resolver problemas através de um enfoque intuitivo, em geral racional, no qual a estrutura do problema possa ser interpretada e explorada inteligentemente para obter uma solução razoável.

Sob a ótica da otimização, Lima [53] define uma heurística como um procedimento que busca alcançar soluções com qualidade ótima ou sub ótima em tempo computacional aceitável, entretanto, não garantem a otimalidade da solução.

Cordeau and Laporte [23] classificam métodos heurísticos em três grupos:

1. Heurísticas Construtivas: buscam progressivamente compor uma solução factível, procurando manter o custo desta solução o mais baixo possível.
2. Heurística de Duas Fases: neste modelo busca-se na primeira fase o agrupamento dos clientes (demanda) em rotas factíveis e, somente na segunda fase as rotas são estabelecidas.
3. Métodos de Melhoria: basicamente estes métodos procuram melhorar a solução obtida através de operações de realocação de clientes (demanda). Estes métodos partem de uma solução obtida e procuram aperfeiçoar o resultado, utilizando para isso uma sistemática predefinida.

A seguir são descritos os algoritmos *Clarke and Wright* e algoritmo de varredura, que são muito estudados e usados em PRVC.

3.2.1 Algoritmo *Clarke and Wright*

De acordo com Toth and Vigo [86], o algoritmo *Clarke and Wright* é um dos algoritmos heurísticos mais conhecidos e estudados em PRVC. Foi definido por Clarke and Wright [22], e é baseado no princípio de economia (*savings*). É um método construtivo, ou seja, a solução vai sendo construída a cada passo, sempre se preocupando com as restrições do problema. O princípio do algoritmo se baseia na ideia de que se há duas rotas diferentes $r = (0, \dots, 1, 0)$ e $S = (0, j, \dots, 0)$ que são factíveis de uma fusão em uma nova rota $t = (0, \dots, 1, j, \dots, 0)$, a fusão dessas rotas gera uma economia de custo S_{ij} , definida por:

$$S_{ij} = C(i, 0) + C(0, j) - C(i, j). \quad (18)$$

Duas rotas r e s são factíveis de uma fusão quando a soma das demandas dos clientes das duas rotas não excede a capacidade máxima Q de um veículo:

$$\sum_{i \in r} qi + \sum_{j \in s} qj \leq Q. \quad (19)$$

Existem duas versões do algoritmo *Clarke and Wright*: a versão paralela e a versão sequencial. A diferença entre as versões está na estratégia de seleção e no número de rotas construídas a cada passo do algoritmo. Enquanto a versão sequencial apenas trata de uma rota por vez, a versão paralela permite que mais de uma rota seja construída. O

Algorithm 2 Algoritmo Clarke and Wright

```

1: Entrada: Conjunto de vértices  $V$  e conjunto de arestas  $E$ , capacidade máxima  $Q$  dos
   veículos
2: Saída: Conjunto de rotas
3: routes = createInicialRoutes( $V$ );
4: savingsList = computeSavingsList( $V$ );
5: while  $do$   $i, j \in savingsList$ 
6:   if  $then$   $feasibleMerge(i, j, Q)$ 
7:     routes.add(mergeRoutes( $i, j$ ));
8:   end if
9: end while
10: return routes

```

Algoritmo 2 começa colocando cada cliente i em uma nova rota $r = (0, i, 0)$ que vai até o cliente e retorna para o depósito (Linha 3 do Algoritmo 2). Depois, é criada a lista de $savings(savingslist)$ com a economia obtida entre cada nó $i, j \in V - 0$, ordenada de forma decrescente de economia. O Algoritmo 3 contém o pseudocódigo da função de criação da lista de $savings$. Nos passos das Linhas 5-8 do Algoritmo 2, a lista de $savings$

Algorithm 3 Função computeSavingsList

```

1: Entrada: Conjunto de vértices  $V$ 
2: Saída: Lista de  $savings$ , ordenada por ordem decrescente de economia
3: savingsList = {};
4: while  $do$   $i, j \in V - \{0\}$ 
5:    $S_{ij} = C(i, 0) + C(0, j) - C(i, j)$ ;
6:   savingsList.add( $S_{i,j}$ );
7: end while
8: sortDecreasing(savingsList);
9: return savingsList

```

é iterada e para cada par (i, j) o Algoritmo *CW* tenta fundir as rotas r e s , as quais i e j pertencem respectivamente (Linha 6). A Linha 6 verifica as condições para que duas rotas sejam consideradas factíveis de fusão, já citadas anteriormente:

1. i deve ser o primeiro (ou último) cliente a ser visitado na sua rota r ,
2. j deve ser o último (ou o primeiro) cliente a ser visitado na sua rota s ,

3. r deve ser diferente de s e
4. a capacidade das rotas juntas não deve exceder Q (ver Equação 19).

Quando duas rotas r e s podem ser fundidas, uma nova rota t ligando as rotas r e s é criada. As rotas r e s são eliminadas e a rota t é adicionada na lista de rotas (Linha 7). No final, a lista de rotas criada (*routes*) é retornada pelo algoritmo.

3.2.2 Algoritmo de Varredura

O algoritmo de varredura do inglês (*Sweep Algorithm*) foi proposto por Gillett and Miller [36]. Esta heurística é uma das tantas existentes desenvolvidas que se baseiam no conceito de heurísticas de duas fases conforme a abordagem *cluster-first, route-second*, ou primeiro agrupar e depois rotear.

O método baseia-se na premissa de que as rotas são desenvolvidas preferencialmente entre os pontos de demanda mais próximos (vizinhos) e destaca-se por apresentar uma enorme eficiência e simplicidade na fase de agrupamento [39].

Inicialmente calcula-se as coordenadas polares para todos os pontos de demanda. As coordenadas polares são calculadas em relação ao depósito. Ao final deste procedimento, os pontos de demanda (nós) são ordenados em ordem crescente de acordo com os valores angulares de suas coordenadas. Seleciona-se o ponto de demanda com menor angulação e iterativamente os demais são interconectados até que alguma das restrições seja excedida. Quando uma restrição excede, o procedimento é reiniciado e novamente os pontos de demanda restantes são ligados ao depósito. Desta forma, os pontos são agrupados segundo o critério angular. Como ocorre a conexão dos pontos de demanda a cada iteração, ao final do procedimento tem-se uma solução inicial [57].

Uma possível abordagem é não conectar os pontos de demanda durante a etapa de *clustering*. Neste caso, a segunda etapa do algoritmo deverá necessariamente ser a aplicação de um método que gere soluções sobre o grupo de pontos. Existem ainda abordagens que utilizam na segunda etapa, etapa de *routing*, algoritmo exato.

Após ter construído a solução inicial para o problema, aplica-se procedimentos de busca local no intuito de intensificar e refinar a qualidade da solução gerada. Nesta implementação, utiliza-se o procedimento k-opt, utilizando valor de k igual a 2.

Algorithm 4 Algoritmo de Varredura

```

1: Ler Grafo  $G = (N, A)$ ,  $c_i, j$ ;  $n$  1 depósito central do roteamento
2: Obter as coordenadas polares dos clientes em relação ao depósito;
3: Ordená-las em ordem crescente do seu valor angular;
4:  $F = N \setminus x_1$ ;
5:  $PontaRota = x_1$ ;
6:  $i = 1$ ;
7: while  $do(F \neq \emptyset)$ 
8:    $Rota_i = \{x_i\}$ ;
9:   while  $do(\exists x_s \in F)$  atendendo as condições de viabilidade para:  $Rota_i$ 
10:    Encontrar o vértice  $x_s \in F$  com menor coordenada polar;
11:     $Rota_i = Rota_i \cup x_s$ ;
12:     $PontaRota = x_s$ ;
13:     $F = F \setminus x_s$ ;
14:   end while
15:    $Rota_i = Rota_i \cup \{x_1\}$ ;
16:   Aplicar procedimento de busca local em  $Rota_i$ ;
17:    $i = i + 1$ ;
18:    $PontaRota = x_1$ ;
19: end while

```

3.3 Meta-heurísticas

Existem diversas tentativas de definição formal de meta-heurísticas na literatura. De acordo com Glover [37], uma meta-heurística se refere a uma estratégia principal que guia e modifica outras heurísticas para produzir soluções melhores que estas, que normalmente são geradas para uma busca pela otimalidade local.

Oliveira [64], em trabalho de dissertação, considera que as meta-heurísticas, diferentemente das heurísticas, possuem uma definição genérica o suficiente para serem estendidas para a resolução de qualquer problema de otimização numérica ou combinatória. Geralmente se baseiam em comportamentos inteligentes de sistemas naturais e com base nestas características se constituem em mecanismos potencialmente importantes para a busca de boas soluções em problemas complexos, como os problemas da classe NP-difícil.

As meta-heurísticas são estratégias heurísticas de alto nível, que não atendem a um problema específico e podem ter diferentes aplicações [12]. As mais conhecidas e utilizadas meta-heurísticas são a Busca Tabu (BT), e o *Simulated Annealing* (SA). A seguir uma breve descrição de cada uma destas meta-heurísticas.

3.3.1 Busca Tabu

Proposta por Glover [37], a meta-heurística Busca Tabu consiste em incluir uma estrutura de memória que permita o algoritmo de busca local escapar de soluções ótimas locais. Segundo mesmo autor, a Busca Tabu inclui elementos de memória de curto prazo, que evitam a reversão de movimentos recentes, e uma memória de mais longo prazo, para

reforçar a atração por soluções mais distantes. O algoritmo realiza buscas locais sempre visando encontrar um ótimo local, permitindo movimentos que não sejam de melhoria da solução, para evitar retornar em soluções previamente avaliadas. Os movimentos realizados são armazenados em uma lista tabu. Estes ficam armazenados na lista segundo um critério de tempo (por exemplo número de iterações), e enquanto o movimento estiver registrado na lista tabu, este movimento é considerado tabu, não podendo ser realizado, e impedindo a ciclagem da busca local.

A função objetivo é utilizada para considerar uma solução s que melhore a melhor solução encontrada até o momento, mesmo se o movimento realizado seja um movimento proibido (movimento tabu). No pseudocódigo do Algoritmo 5 é apresentado o procedimento Busca Tabu. Percebe-se que na (linha 1) gera-se uma solução inicial s_0 . Em seguida, a cada iteração o algoritmo explora um subconjunto V da vizinhança $N(s)$ da solução corrente s . A solução s' de V com função objetivo de menor valor torna-se a nova solução corrente mesmo que s' seja pior que s (s' não pode estar armazenado na lista tabu). Esses movimentos são armazenados na lista tabu (linha 9) e permanecem proibidos por número de iterações. Em seguida, é verificado se o valor da nova solução (s) é superior ao valor da melhor solução encontrada até o momento (s^*) (linha 11), caso verdadeiro, então esta possui seu valor atualizado (linha 12). Posteriormente, o contador de iteração é incrementado (linha 13). O procedimento continua até que o critério de parada seja satisfeito.

Algorithm 5 Algoritmo Busca Tabu

```

1: Seja  $s_0$  solução inicial
2:  $s^* \leftarrow s_0$ 
3: Iter  $\leftarrow 0$ 
4: MelhorIter  $\leftarrow 0$ 
5: T  $\leftarrow \emptyset$ 
6: while do(critério de parada não satisfeito)
7:   Seja  $s' \leftarrow s \oplus m$  o melhor elemento de  $V \subseteq N(s)$  tal que o movimento  $m$  não
   seja tabu ( $m \notin T$ ) ou  $s'$  atenda a condição de aspiração ( $f(s') < A(f(s))$ )
8:   Atualise a Lista Tabu T
9:    $s \leftarrow s'$ 
10:  if then  $f(s) < f(s^*)$ 
11:     $s^* \leftarrow s$ 
12:  end if
13:  Iter  $\leftarrow$  Iter + 1
14: end while
15: return  $s^*$ 

```

3.3.2 Simulated Annealing

Simulated Annealing (SA) foi proposta por Kirkpatrick et al. [45]. Trata-se de um procedimento de busca local probabilístico. Seu funcionamento é associado a uma operação

chamada recozimento, relacionada à termodinâmica. Esta operação consiste em aquecer o sistema até atingir uma alta temperatura para então diminuí-la em etapas lentas até que o estado de equilíbrio seja alcançado. Este algoritmo começa sua busca a partir de uma solução inicial (conforme Algoritmo 6 - linha 1). O procedimento consiste em um laço de repetição que gera aleatoriamente, em cada iteração, um único vizinho s' da solução corrente s (linha 6). O critério para aceitar uma solução como a solução atual é feito em função de uma probabilidade determinada de acordo com a qualidade da solução e com a temperatura atual (linhas 7 - 8). Caso o critério de aceitação seja válido, o valor da melhor solução atual (s) recebe o valor da nova solução (s') (linha 9). Em seguida, é verificado se o valor da nova solução (s') é superior ao valor da melhor solução encontrada até o momento (s^*) (linha 10), caso verdadeiro, então esta possui seu valor atualizado (linha 11). Posteriormente, o contador de iteração é incrementado (linha 14) e a temperatura T é modificada pela taxa de resfriamento (linha 16). O algoritmo é encerrado quando a temperatura atual atingir ou estiver abaixo da temperatura mínima.

Algorithm 6 Algoritmo Simulated Annealing

```

1:  $s^* \leftarrow s$ 
2:  $T \leftarrow T_i$ 
3: while  $T > T_f$ 
4:    $IterT \leftarrow 0$ 
5:   while  $IterT < SAm_{ax}$ 
6:      $s' \leftarrow$  gerar aleatoriamente uma solução vizinha  $s' \in N(s)$ 
7:      $(\Delta = f(s') - f(s))$ 
8:     if then  $(\Delta < 0)$  or  $random [0, 1] < e^{-\Delta/T}$ 
9:        $s \leftarrow s'$ 
10:      if then  $(f(s') < f(s^*))$ 
11:         $s^* \leftarrow s'$ 
12:      end if
13:    end if
14:     $IterT \leftarrow IterT + 1$ 
15:  end while
16:   $T \leftarrow T \times (1 - \alpha)$ 
17: end while
18: return  $s^*$ 

```

4 ALGORITMOS E TÉCNICAS UTILIZADOS

Nos capítulos anteriores foram descritos o problema de roteamento de veículos, suas versões e as principais técnicas para sua resolução. Neste capítulo são apresentadas as estratégias utilizadas na construção das heurísticas para resolução do problema proposto. Essa estratégia inclui a clusterização dos pontos de entrega de acordo com suas localizações geográficas, a criação de rotas levando em consideração as restrições impostas, a geração de uma solução inicial utilizando heurística de vizinho mais próximo e a melhoria da solução inicial obtida por meio do algoritmo do vizinho mais próximo. O fluxograma (Figura 7) ilustra processo gerais das estratégias. As Seções a seguir apresentam cada etapa desenvolvida.

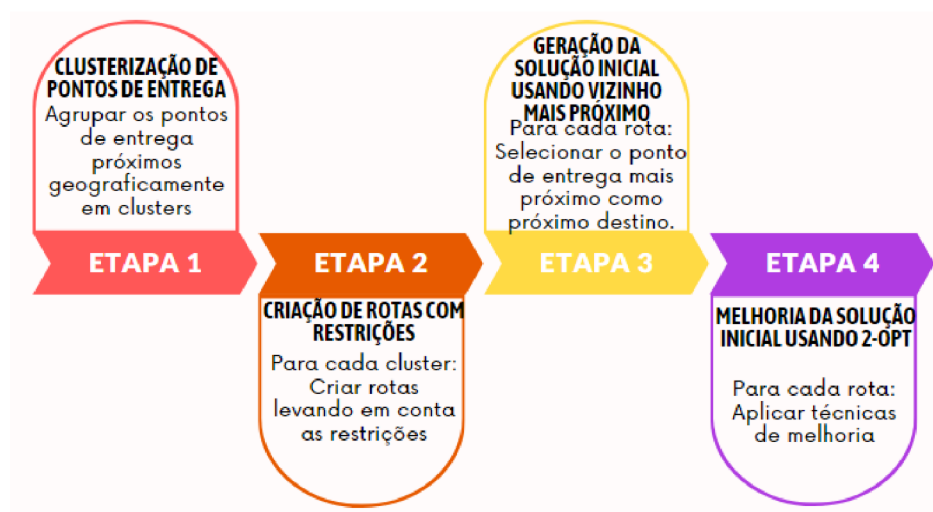


Figura 7: Fluxograma gerais das estratégias

4.1 Agrupamento

Consiste na implementação de técnicas computacionais para separar um conjunto de dados em diferentes grupos com base em suas similaridades.

De acordo com Priy [71], agrupamento é a tarefa de dividir a população ou pontos de dados em vários grupos, de modo que os pontos de dados nos mesmos grupos sejam mais

semelhantes a outros pontos de dados no mesmo grupo e diferentes dos pontos de dados em outros grupos.

Existem vários tipos de métodos de agrupamento, conforme Tabela 3 e a principal diferença entre eles está na abordagem e na estratégia usada para formar os grupos.

Neste contexto, para a presente pesquisa, foram selecionados dois métodos diferentes. O primeiro utiliza o conceito de centroides e é implementado pelo algoritmo *K-Means* e o segundo método é baseado na densidade dos pontos e é executado pelo algoritmo DBSCAN.

Tabela 3: Métodos de agrupamento

Método	Definição
Métodos Baseados em Densidade	<i>Esses métodos consideram os aglomerados como a região densa tendo algumas semelhanças e diferenças com a região densa inferior do espaço. Esses métodos têm boa precisão e a capacidade de mesclar dois clusters.</i>
Métodos de base hierárquica	<i>Os clusters formados neste método formam uma estrutura do tipo árvore baseada na hierarquia. Novos clusters são formados usando o previamente formado. Está dividido em duas categorias Aglomerativo (abordagem de baixo para cima) Divisivo (abordagem de cima para baixo).</i>
Métodos de particionamento	<i>Esses métodos particionam os objetos em k clusters e cada partição forma um cluster.</i>
Métodos baseados em grade	<i>Neste método, o espaço de dados é formulado em um número finito de células que formam uma estrutura semelhante a uma grade.</i>

4.1.1 K-Means

Descrito inicialmente por MacQueen [58], o *k-means* é um algoritmo de agrupamento exclusivo onde cada elemento é atribuído somente a um grupo dentre o conjunto de grupos formados. É um algoritmo iterativo que divide o conjunto de dados não rotulado em k grupos diferentes de forma que cada conjunto de dados pertença a apenas um grupo com propriedades semelhantes.

Neste método de agrupamento, inicia-se definindo o número de grupos que se almeja formar a partir dos dados, esse número inteiro é o valor de k [14].

O *K-Means* tem o objetivo de dividir as observações em k grupos. K é o número de grupos. Esta divisão dos dados entre os grupos tem que ser de forma que os diferentes grupos fiquem mais separados entre eles, enquanto que as observações dentro de cada grupo fiquem mais próximas entre elas. Para isso, é utilizada a soma dos erros quadrados ou *Sum of Squared Errors (SSE)* que busca minimizar a distância entre os pontos e seu centroide. Depois que o modelo é criado, o *K-means* quando é aplicado lembra a média

de cada grupo, também chamada de centroides e, com isso, encontra o centroide mais próximo a cada novo dado. O centroide representa o centro do grupo.

$$j = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - c_j\|^2 \quad (20)$$

Onde:

j = é a função objetiva

k = número de grupos

n = número de casos

x_i = caso i

c_j = centroide para o grupo j

$\|x_i^{(j)} - c_j\|^2$ = função distância

A ideia do algoritmo *K-Means* (também chamado de K-Médias) é fornecer uma classificação de informações de acordo com os próprios dados. Esta classificação, é baseada em análise e comparações entre os valores numéricos dos dados. Desta maneira, o algoritmo automaticamente vai fornecer uma classificação automática sem a necessidade de nenhuma supervisão humana, ou seja, sem nenhuma pré-classificação existente. Por causa desta característica, o *K-Means* é considerado como um algoritmo de mineração de dados não supervisionado.

De acordo com Tan et al. [83], o algoritmo *k-means* pode ser descrito em quatro etapas principais: seleção aleatória de k centroides entre os dados; atribuição de cada ponto ao centroide mais próximo; mudança dos centroides para o centro das amostras que lhe foram atribuídas; e recálculo dos centroides de cada grupo até que ocorra a convergência/estabilidade destes centroides (Figura 8).

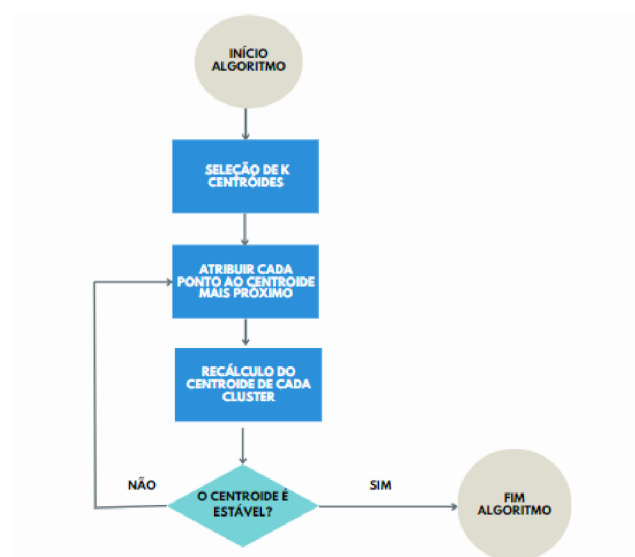


Figura 8: Fluxograma de etapas do algoritmo k-means

É importante entender que, o método *k-means* não garante que os grupos formados converjam para o ótimo global, sendo que muitas vezes a resposta encontrada representa uma solução ótima local. Os resultados são influenciados pela escolha inicial (aleatória) dos centros dos grupos. Outro ponto a salientar é a necessidade de determinar o número de grupos previamente. Para obter bons resultados na prática, é comum executar o algoritmo *k-means* várias vezes com diferentes centroides iniciais [18].

4.1.2 DBSCAN

Um dos algoritmos de agrupamento mais usados e citados é o DBSCAN [32]. Ele requer dois parâmetros (ϵ e $minPts$) e funciona distinguindo pontos centrais, pontos de borda e ruído.

DBSCAN, abreviação do termo *Density Based Spatial Clustering of Application with Noise* (Clusterização Espacial Baseada em Densidade de Aplicações com Ruído) é um método de clusterização não paramétrico baseado em densidade, proposto por Ester et al. [32], que é significativamente efetivo para identificar grupos de formato arbitrário e de diferentes tamanhos, identificar e separar os ruídos dos dados e detectar grupos naturais e seus arranjos dentro do espaço de dados, sem qualquer informação preliminar sobre os grupos. O método requer somente um parâmetro de entrada, mas dá suporte para determinar um apropriado valor para ele. A ideia chave do método DBSCAN é que, para cada ponto de um grupo, a vizinhança para um dado raio contém, no mínimo, certo número de pontos, ou seja, a densidade na vizinhança tem que exceder um limiar. Para entender o método algumas definições específicas listadas a seguir.

- Definição 1. A vizinhança ϵ de um ponto p , denotada por $N_\epsilon(p)$, é definido por $N_\epsilon(p) = \{q \in D | dist(p, q) \leq \epsilon\}$, onde D é um conjunto de pontos e $dist(p, q)$ é uma função de distância por exemplo. Distância euclidiana, entre p e q .
- Definição 2. Um ponto p é um ponto central se $|N_\epsilon(p)| \geq minPts$
- Definição 3. Um ponto p é diretamente alcançável por densidade a partir de um ponto q em relação a ϵ e $minPts$ se $p \in N_\epsilon(q)$ e q é um ponto central.
- Definição 4. Um ponto p é um ponto de fronteira se p é diretamente alcançável por densidade a partir de um ponto central q e $|N_\epsilon(p)| < minPts$
- Definição 5. Um ponto p é alcançável por densidade a partir de um ponto q em relação a ϵ e $minPts$ se houver uma cadeia de pontos $p_1, \dots, p_n$, com $p_1 = q$ e $p_n = p$ de tal modo que p_{i+1} é diretamente alcançável por densidade a partir de p_i
- Definição 6. Um ponto p é conectado por densidade a um ponto q em relação a ϵ e $minPts$ se existe um ponto o tal que ambos, p e q são alcançáveis por densidade a partir de o .

- Definição 7. Seja D um conjunto de pontos. Um *cluster* C em relação a ϵ e $minPts$ é um subconjunto não vazio de D satisfazendo as seguintes condições:

1. $\forall p, q : \text{se } p \in C \text{ e } q \text{ é densidade alcançável de } p \text{ em relação a } \epsilon \text{ e } minPts, \text{ então } q \in C.$
2. $\forall p, q \in C : p \text{ é conectado por densidade a } q \text{ em relação a } \epsilon \text{ e } minPts.$

- Definição 8. Um ponto p é um ruído se não for um ponto central nem um ponto de borda. Isso implica que o ruído não pertence a nenhum *cluster*.

O algoritmo completo para DBSCAN é mostrado nos Algoritmos 7 e 8. Primeiro, DBSCAN começa com um ponto arbitrário e recupera todos os pontos em sua vizinhança ϵ . Se o número de pontos na vizinhança ϵ for suficiente (em relação a $minPts$), um novo grupo será iniciado. Esse grupo será então expandido até recuperar todos os pontos cuja densidade seja alcançável a partir de p . Se o número de pontos na vizinhança ϵ não for suficiente, p será marcado como ruído. A função $RangeQuery(p, \epsilon)$ relata todos os pontos $q \in D$ tais que $dist(p, q) \leq \epsilon$.

Algorithm 7 DBSCAN($D, \epsilon, minPts$)

```

1:  $\{D \text{ é um conjunto de pontos não classificados}\}$ 
2:  $\{\epsilon \text{ é a distância máxima}\}$ 
3:  $\{minPts \text{ é o mínimo de pontos para formar um cluster}\}$ 
4: Inicializar ID do cluster  $C = 0$ 
5: for all cada ponto não classificado  $p \in D$  do
6:    $N_\epsilon(p) = ConsultaIntervalo(p, \epsilon)$ 
7:   if  $|N_\epsilon(p)| \geq minPts$  then
8:     Definir ID do cluster de  $p$  para  $C$ 
9:     ExpandirCluster( $p, N_\epsilon(p), C, \epsilon, minPts$ )
10:     $C < -C + 1$ 
11:   end if
12:   Rotule  $p$  como ruído
13: end for

```

4.2 Método Auxiliar

Após a seleção dos algoritmos *K-MEANS* e DBSCAN como abordagens de agrupamento, é igualmente importante compreender a metodologia de método cotovelo. Esse método desempenha um papel crucial ao determinar o número ideal de *clusters*, denotado como K , o qual é um parâmetro essencial para a configuração inicial do algoritmo *K-Means*. Esta seção fornecerá uma visão abrangente do método cotovelo e como ele auxilia na definição precisa do número ideal de *clusters*.

Algorithm 8 ExpandirCluster($p, PontosVizinhos, C, \epsilon, minPts$)

```

1: for all cada ponto  $q \in PontosVizinhos$  do
2:   if  $q$  não é classificado then
3:      $N_\epsilon(q) = ConsultaIntervalo$ 
4:     if  $|N_\epsilon(p)| \geq minPts$  then
5:        $PontosVizinhos = PontosVizinhos \cup N_\epsilon(q)$ 
6:     end if
7:   end if
8:   if  $q$  não pertence a nenhum cluster then
9:     Defina o ID do cluster de  $q$  como  $C$ 
10:  end if
11: end for

```

4.2.1 Método cotovelo

Kodinariya et al. [46] citam diversos métodos que podem ser utilizados para definir o número de *clusters* no *k-means*, número este passado pelo parâmetro k . Um deles é Método do Cotovelo (traduzido do inglês, *Elbow Method*), considerado pelo autor o método mais antigo. Como o *k-means* calcula a distância das observações até o centroide do agrupamento que elas pertencem, o ideal é que essa distância seja a menor viável.

Matematicamente falando, busca-se uma quantidade de agrupamentos em que a soma dos quadrados intragrupos ou *wcss* (*within-clusters-sum-of-squares*) seja a menor possível [28]. O método de cotovelo é um método que analisa a porcentagem de variância explicada em função do número de *clusters*. Esse método existe com base na ideia de que se deve escolher um número de *clusters* para que a adição de outro *cluster* não forneça uma modelagem muito melhor dos dados. A porcentagem de variância explicada pelos *clusters* é plotada em relação ao número de *clusters*.

De acordo com Sammouda et al. [75] a técnica do critério de cotovelo é um método heurístico aplicado para determinar o número de *clusters* dos pontos de dados em um conjunto de dados. A técnica de cotovelo é utilizada para obter o número ótimo de *clusters* para um conjunto de pontos de dados por ser um método empírico, simples e de fácil implementação. Aplicando o algoritmo de clusterização *k-means*, a técnica do cotovelo plota as variações explicadas através do número de *clusters* e escolhe a curva do cotovelo para obter o número de *clusters*. Depende do cálculo da soma dos erros quadrados dentro do *cluster* de todos os pontos de dados para representar a qualidade de agregação entre ponto de dados no mesmo *cluster* e separação entre *clusters*. A ideia central da técnica do cotovelo é explicada da seguinte forma:

- À medida que o número de *clusters* K aumenta, a separação do modelo é mais distinta e a quantidade de agregação para cada *cluster* aumentará gradualmente; portanto, a soma dos erros quadrados dentro do *cluster* (*Sum of squared errors within cluster*, *WCSS*) se tornará gradual e obviamente menor.

- Quando K é menor que o número real de *clusters*, aumentar o valor de K aumenta significativamente a quantidade de agregação para cada *cluster*. O WCSS será diminuído e, uma vez que K atingirá o número real de *clusters*, a quantidade de agregação alcançada usando K será aumentada. Portanto, o retorno diminuirá rapidamente e o WCSS diminuirá drasticamente. Consequentemente, ele será nivelado quando o K valor continuar aumentando. Isso significa que a relação entre WCSS e K pode ser representada na forma de cotovelo correspondente ao K valor, que é o número de *clusters* verdadeiros para os pontos de exemplo. Por exemplo, na Figura 9, a maior curvatura do cotovelo está em K valor de 5. Portanto, o melhor número de *clusters* será 5.

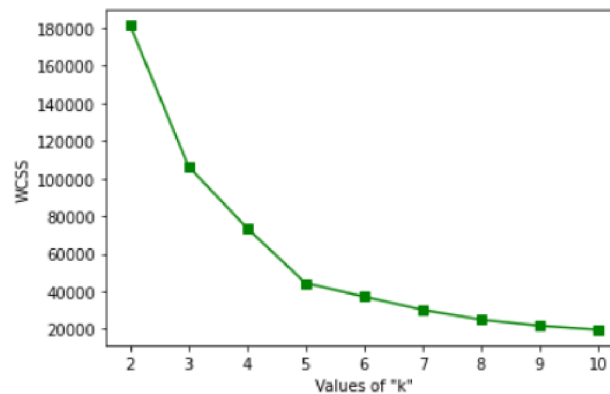


Figura 9: Funcionamento de Método Cotovelo

4.3 Geração da solução inicial

A geração da solução inicial é um passo fundamental em muitos algoritmos de roteamento, onde o objetivo é encontrar a melhor solução possível para um determinado problema. Esse processo envolve a criação de uma solução inicial que servirá como ponto de partida para o algoritmo de roteamento. Na abordagem proposta neste trabalho, a solução inicial é gerada utilizando a heurística de vizinho mais próximo. Essa abordagem constrói uma rota inicial que seja factível, levando em consideração um conjunto de critérios que garantam uma certa eficiência para melhoramento posterior da solução obtida.

4.3.1 Algoritmo Vizinho Mais Próximo

Proposto inicialmente por Cover and Hart [24] o algoritmo vizinho mais próximo consiste na composição da rota com base na inserção sequencial de pontos através de um ponto inicial conforme a menor distância entre este e os seus demais pontos (vizinhos). Após determinar todas as distâncias entre o ponto inicial e os demais pontos do *cluster*, os demais pontos são ordenados de maneira decrescente para possibilitar a escolha e

determinação do nó mais próximo que será designado à rota.

De acordo com Santos and Leal [78], este método é usualmente utilizado para se encontrar uma solução inicial para o problema, solução esta que é então gradativamente melhorada por outras técnicas e modelos. Tem aplicação simples e rápida e pode ser usado para problemas de alta complexidade.

Goldbarg and Luna [39] explicam que esta heurística parte do vértice de origem e adiciona a cada passo o vértice ainda não visitado (Algoritmo 9 - linhas 15 e 16), cuja distância do último vértice visitado seja mínima (linha 10). O procedimento finaliza quando todos os vértices são visitados. Ao final é feita a ligação entre o último vértice e o vértice de origem (linha 19).

Algorithm 9 Algoritmo da Heurística do Vizinho mais próximo

```

1: ENTRADA: Rota;
2: SAÍDA: Rotaminimizada;
3: Primeiro_ponto = Rota[1];
4: Rota_minimizada = Rota[1];
5: Ponto_verificador = Rota[1];
6: I = QuantidadesPontos(Rota);
7: while I > QuantidadePontos(Rotaminimizada)
8:   Menor = 10000000;
9:   for all Rota[K] ≠ Rotaminimizada do
10:     if Distância(Ponto_verificar, Rota[K]) < Menor then
11:       Menor = Distância(Ponto_verificar, Rota[K]);
12:       Ponto_Menor = Rota[K];
13:     end if
14:   end for
15:   Rotaminimizada = Ponto_Menor;
16:   Ponto_verificar = Rota_Menor;
17:   I = I - 1;
18: end while
19: Rotaminimizada = Primeiro_ponto;

```

4.4 Melhoramento de solução inicial

As heurísticas de melhoramento buscam aumentar a eficiência do trajeto, ou seja, refinar uma solução inicial obtida para uma rota a ser seguida, a priori, por um veículo. Começam utilizando uma solução inicial completa, já obtida com o auxílio de um método construtivo ou de métodos aleatórios, onde tentam encontrar uma solução de melhor qualidade na vizinhança [82]. Caso seja encontrada uma solução melhor que a solução corrente, a mesma será substituída. O processo de verificação para melhoria da rota continua até que o critério de parada seja alcançado ou não haja soluções melhores [35]. A melhor classe conhecida de algoritmos é a heurística de trocas de arcos k-opt.

4.4.1 Heurística K-Opt

A heurística k-opt foi proposta por [54], desenvolvida inicialmente para o problema do caixeiro viajante mas pode ser aplicada naturalmente ao problema de roteamento de veículos. Lin and Kernighan [55] descrevem uma forma efetiva de implementar tal método aplicado ao PCV. Tal método é um dos clássicos algoritmos de busca local para o PCV e baseia-se na troca de arcos em soluções inicialmente criadas (através de heurísticas construtivas), onde o número de arcos a serem trocados depende de k. À medida que o parâmetro k aumenta, o procedimento aproxima-se da enumeração total de soluções vizinhas [39].

O procedimento pode ser descrito da seguinte forma: remover k arcos e reinseri-los de todas as maneiras possíveis. Destas reinserções permanece a que possuir o melhor custo associado à função objetivo. Caso não possua custo melhor que o original antes da remoção, escolhe-se outros k arcos para serem removidos. O procedimento é reiniciado até que todas as possibilidades de remoções dos k arcos sejam realizadas.

Segundo Bispo [10], o nome 2-Opt é dado justamente pela troca de duas arestas, valores acima desta abordagem é considerado K-Opt. O autor destaca ainda que é esperado ao crescer o valor de K, uma melhor solução seja encontrada, porém consumindo mais recursos computacionais. Valores de K maiores que 5 são impraticáveis na literatura, visto que o tempo para execução deste procedimento representa tempos exponenciais à medida que K e N crescem. Os valores mais frequentes encontrados na literatura para K são 2 e 3. Neste trabalho optou-se por utilizar o 2-Opt pela sua simplicidade e eficiência computacional.

De acordo com Croes [25], a heurística 2-Opt realiza trocas entre dois arcos com o objetivo de encontrar uma melhoria na solução atual. Caso haja a detecção da melhoria da solução em algumas dessas trocas, esta é assumida como solução atual. Este processo termina quando não for mais possível realizar trocas que melhorem a solução, ou algum limite de parada seja alcançado. Na Figura 10, é ilustrado o processo de troca de arcos para abordagem 2-Opt.

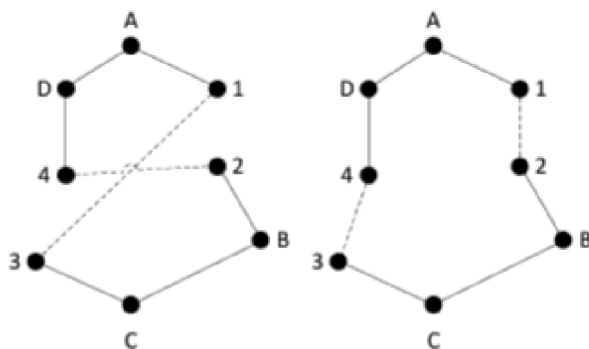


Figura 10: Troca 2-Opt entre os trechos 1 - 2 e 3 - 4

4.5 Modelagem das Estratégias Propostas

As estruturas principais das estratégias propostas são compostas basicamente por seis etapas, conforme apresentado no fluxograma da Figura 11.

Na primeira etapa, algumas etapas de pré-processamento são realizadas para preparar os dados para a clusterização. As colunas LATITUDE e LONGITUDE, contêm coordenadas geográficas representadas como *string* com vírgulas. As vírgulas são substituídas por pontos em ambas as colunas para que as coordenadas sejam tratadas como valores numéricos válidos. Em seguida, as colunas são convertidas para o tipo numérico.

A segunda etapa está relacionada à clusterização dos clientes de acordo com suas localidades geográficas. Conforme mencionado anteriormente, foram propostas duas estratégias, cada uma empregando um modelo específico de método de agrupamento para agrupar os pontos da entrega.

A primeira estratégia (Estratégia baseada no algoritmo *K-means*), utiliza o algoritmo de clusterização *k-means*, que agrupa as observações em *k clusters*, sendo *k* o número de grupos desejado. Em algoritmo de agrupamento, como o *k-means*, é necessário determinar o número adequado de *clusters* para um determinado conjunto de dados. Isso garante que os dados sejam divididos de maneira apropriada e eficiente. Um valor apropriado de *k*, ou seja, o número de *clusters*, ajuda a garantir a granularidade adequada dos grupos e a manter um bom equilíbrio entre a compressibilidade e a precisão dos mesmos. Para determinar o valor ideal de *k*, nas instâncias trabalhadas, utiliza-se o método do cotovelo, que auxilia na definição do número ótimo de grupos. Esse valor de *k*, deve ser passado como parâmetro no início da execução do algoritmo *k-means*.

Na estratégia baseada no DBSCAN, a clusterização dos clientes em grupos é realizada usando o algoritmo DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*). Esse algoritmo é utilizado para agrupar um conjunto de dados espaciais com base em dois parâmetros principais: a distância física de cada ponto e o tamanho mínimo do *cluster*. O DBSCAN identifica regiões densas de pontos no espaço, considerando pontos que estão próximos o suficiente uns dos outros como parte do mesmo *cluster*. O parâmetro ϵ (épsilon), representa a distância máxima que define a vizinhança de cada ponto enquanto que parâmetro *min_samples* define o número mínimo de pontos dentro dessa vizinhança para que o ponto seja considerado como núcleo de um *cluster*. Para calcular a distância entre dois pontos é usada a métrica *haversine*. Essa fórmula calcula a distância entre dois pontos na superfície de uma esfera, levando em conta a curvatura da terra. O algoritmo da árvore de bolas (*BallTree*)¹ é utilizado para otimizar cálculo de distâncias pontuais e encontrar os vizinhos mais próximos dos pontos, auxiliando assim no processo de clusterização.

¹O algoritmo Ball Tree é um método usado para busca de vizinhos mais próximos em espaços multidimensionais. O algoritmo se baseia na ideia de dividir o espaço de dados em regiões esféricas chamadas bolas e construir uma estrutura de árvore para organizar essas bolas de forma hierárquica.

Na terceira etapa, é realizada a seleção de um grupo específico dentre os criados pelo algoritmo de clusterização. O grupo escolhido é armazenado em uma variável, e sua demanda total é calculada. Isso significa que rotas são criadas de forma que a soma das demandas não exceda a capacidade de veículo, que, neste caso, é de **180²**.

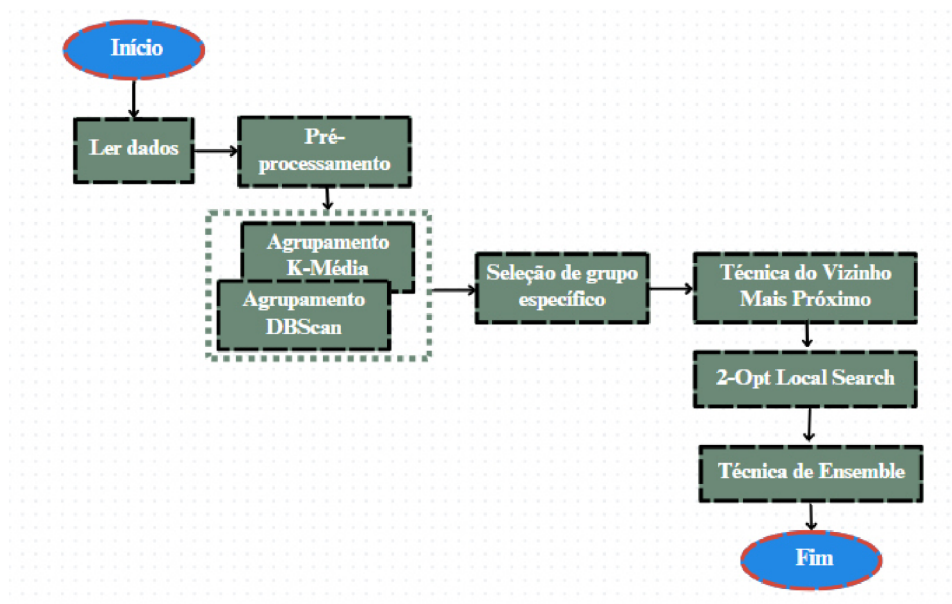


Figura 11: Fluxograma geral do modelo

Na sequência, após da criação das rotas, uma rota inicial é criada. Isso é feito aplicando algoritmo de vizinho mais próximo para encontrar um caminho inicial que visita todos os pontos do grupo específico em ordem de proximidade, partindo e terminando num ponto inicial (depósito), respeitando a restrição de capacidade de veículos e visitando um cliente somente uma vez por somente um veículo. Esta etapa está relacionada à quarta etapa do processo.

Na quinta etapa, é aplicada o algoritmo de melhoramento 2-Opt para aprimorar a solução inicial obtida por meio do algoritmo do vizinho mais próximo. Esse algoritmo gera uma solução inicial viável que serve como ponto de partida. A partir dessa solução inicial, são realizadas trocas com o objetivo de melhorá-la. Em cada etapa, são feitas trocas para reduzir o custo original da rota. Essas trocas são realizadas até que um critério de parada pré-estabelecido seja alcançado, como a ausência de soluções melhores no espaço de trocas definido ou um determinado número de iterações. O objetivo é encontrar um caminho (rota) que minimize a distância total percorrida ao visitar todos os pontos e voltar ao ponto inicial.

Na sexta e última etapa, ocorre a combinação das rotas para obter a solução final. Isso é realizado por meio da regra de combinação *ensemble*, utilizando uma abordagem baseada em combinador algébrico. Esse tipo de combinador é não treinável e opera sobre as

²Valor pré-estabelecido pelo benchmark da Loggi.

saídas contínuas dos classificadores ou, no contexto, das rotas. As saídas são combinadas usando uma expressão algébrica simples, geralmente uma soma.

A função objetivo que representa a soma total das distâncias das rotas pode ser escrita da seguinte maneira:

$$f = \sum_{i=1}^n D_i \quad (21)$$

Onde:

f = é a função objetivo

D_i = distância da rota i

n = número de rotas

i = é a rota

5 RESULTADOS

Este capítulo tem como objetivo avaliar a eficiência das estratégias aplicadas ao problema de roteamento de veículos capacitado. Para isso, são realizados testes comparando o desempenho da abordagem do *benchmark loggi BUD* e das Estratégias propostas. Os resultados dos testes são apresentados e analisados neste capítulo para avaliar a eficácia da técnica e suas vantagens em relação ao modelo original da empresa Loggi, que disponibilizou o banco de dados de testes.

5.1 Conjunto de Dados

O banco de dados utilizado como referência no trabalho é o *Benchmark Loggi BUD*, um conjunto de dados para problemas de grande escala, obtido no repositório *Loggibud*, que simulam o desafio de entregas na última etapa da cadeia logística. Os dados dessa base estão estruturados em dicionários de dados Json, organizados conforme as cidades. Cada arquivo json, contém um nome, a região atendida, as coordenadas do ponto de origem (latitude e longitude), a capacidade dos veículos e os pontos de entrega, constituídos de um identificador e coordenadas do endereço.

No repositório do *benchmark loggiBUD* estão disponíveis dois tipos de instâncias, *delivery-instances* e *cvrp-instances*. Para esta dissertação são trabalhadas as instâncias *cvrp-instances*, as quais representam o clássico problema de roteamento de veículos capacitado (PRVC). Essas instâncias contêm informações sobre a localização do depósito, a capacidade dos veículos da frota, as localizações de cada entrega a ser feita, além das demandas de cada uma das entregas.

O *LoggiBUD* em sua versão inicial disponibiliza um total de 90 instâncias de treinamento e 30 instâncias de avaliação para Rio de Janeiro (RJ), Brasília (DF) e Belém (PA). Sendo cada uma com quantidades de pacotes (entregas) diversos.

Para experimentos realizados neste trabalho foram geradas três instâncias, sendo uma gerada a partir das instâncias do estado do Pará, uma a partir de instâncias do Distrito Federal e uma do Rio de Janeiro.

Em relação ao número de entregas a serem realizadas, a instância do Rio de Janeiro

(RJ) apresenta o maior número, com um total de 4.273 entregas. Na sequência, a instância de Distrito Federal (DF) conta com 987 entregas a serem efetuadas. Por fim, a instância referente ao Pará (PA) possui o menor número de entregas, com um total de 297.

5.2 Testes

Os testes foram realizados utilizando as estratégias aplicadas neste trabalho e foram aplicados em instâncias obtidas no repositório da *LoggiBud*, disponibilizada no seguinte endereço: <https://loggibud.s3.amazonaws.com/dataset.zip>.

As instâncias contêm informações sobre: a localização do depósito, a capacidade dos veículos da frota que é de 180 unidades de carga, as localizações de cada entrega a ser feita e as demandas de cada uma das entregas, variando de 1 a 10 unidades de peso.

Os testes foram executados em um computador pessoal com o sistema operacional *Windows 11* de 64 bits, equipado com 8 GB de memória e um processador Intel(R) Core(TM) i3-1005G1 de 10ª geração. A implementação foi realizada em linguagem de programação *Python* no ambiente *Jupyter Notebook Anaconda3*.

5.3 Métricas para avaliação das estratégias

Para avaliar a qualidade das soluções obtidas pelas estratégias, foram considerados dois objetivos principais: minimizar a distância total percorrida pelos caminhões e alcançar um equilíbrio na distribuição das distâncias percorridas de veículos. Além disso, utilizou-se a medida do desvio padrão da amostra (DPA) em relação à melhor solução conhecida das distâncias percorridas. O desvio padrão da amostra é uma estimativa, com base em uma amostra, de um desvio padrão da população. A variação s^2 e o desvio padrão s da amostra são apresentados nas Eq [22](#) e [23](#).

$$s = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2} \quad (22)$$

$$s^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2 \quad (23)$$

Onde:

s = desvio padrão da amostra

s^2 = variação da amostra

$x_1, \dots, x_N$ = o conjunto de dados de amostra

$\bar{x}$ = valor médio do conjunto de dados de amostra

N = tamanho do conjunto de dados de amostra

5.4 Avaliação das Estratégias nas Instâncias

As estratégias foram avaliadas por meio de experimentos computacionais que envolvem um conjunto de instâncias dos três estados. No que tange à frota de veículos, considerou-se frota homogênea, onde todos os veículos possuem a mesma capacidade. Os parâmetros empregados durante experimentos foram detalhados na seção 4.5.

5.4.1 Estratégia baseada no algoritmo *K-Means*

Nesta estratégia, foi empregado o algoritmo *K-means* para agrupar os clientes. Conforme mencionado anteriormente, no algoritmo *K-means*, é necessário determinar o número adequado de *clusters* para um conjunto de dados específico.

Para escolher o número certo de *clusters*, utiliza-se o ponto de dobra na curva da soma das variâncias dentro do *cluster* em relação ao número de *clusters*. O primeiro ponto de inflexão nessa curva sugere o valor correto de k para qualquer $k > 0$.

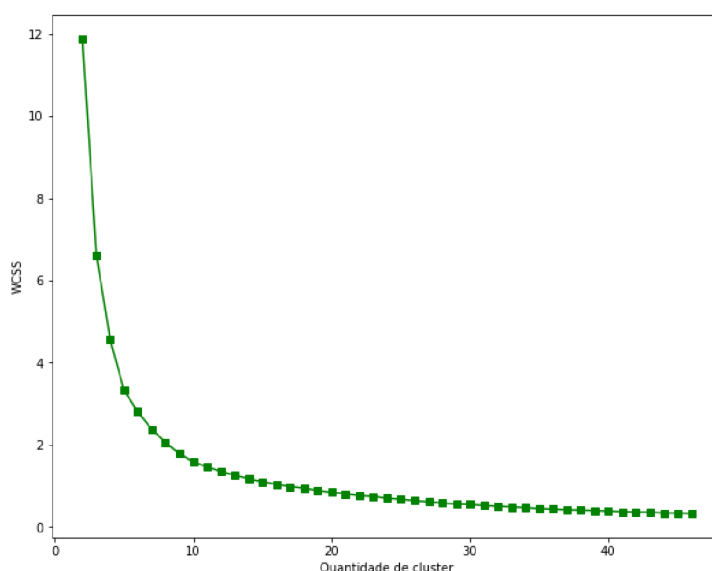


Figura 12: Método do Cotovelo para instâncias do Rio de Janeiro

As Figuras 12, 13 e 14 ilustram os resultados obtidos através da aplicação do método do cotovelo, o qual é empregado para definir o valor adequado do parâmetro k (número de *clusters*). Inicialmente, uma lista vazia é criada para armazenar os valores da inércia (wcss) resultantes do agrupamento. Em seguida, um laço de repetição é utilizado para calcular o valor da inércia para cada k (número de *clusters*), e esses valores são adicionados à lista recém-criada.

Após, há uma estrutura de repetição em forma de laço, a qual calcula o valor da inércia para cada k (número de *clusters*), e acrescenta à lista criada. Enfim, plota-se o gráfico, no qual o eixo horizontal representa o número de *clusters* e eixo vertical representa os valores wcss.

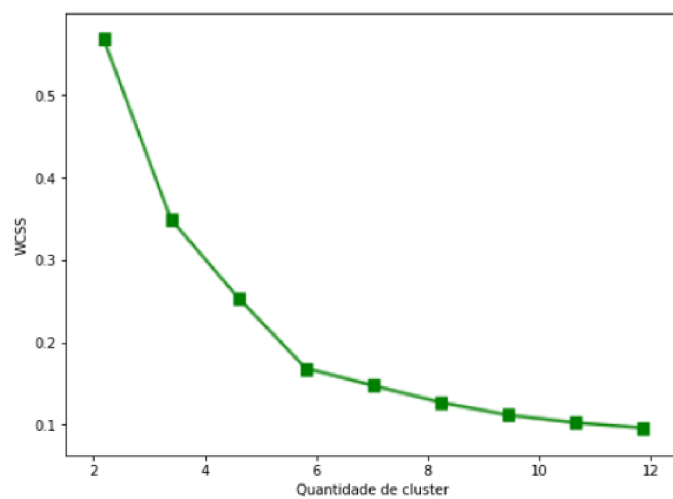


Figura 13: *Método do Cotovelo para instâncias do Distrito Federal*

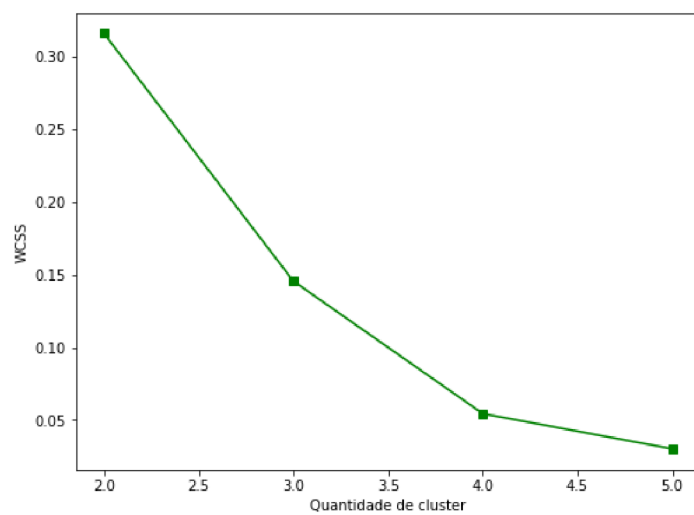


Figura 14: *Método do Cotovelo para instâncias do Pará*

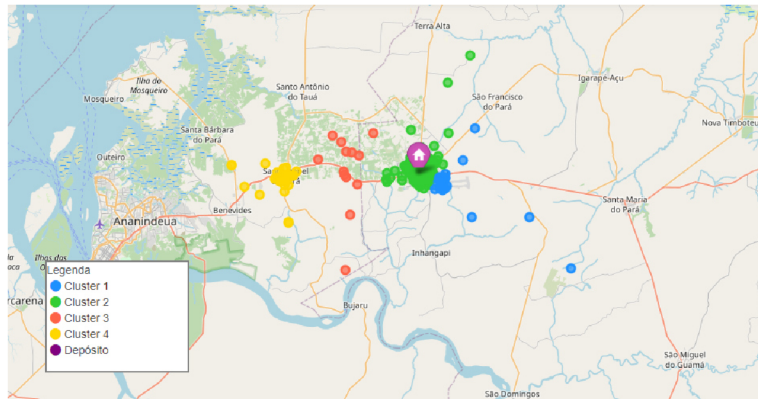


Figura 17: Agrupamento por K-Means no Estado do Pará

171 entregas; o *cluster* 6 tem 453 entregas; o *cluster* 7 tem 458 entregas; o *cluster* 8 tem 549 entregas; o *cluster* 9 tem 1372 entregas; e o *cluster* 10 possui 289 entregas.

Para o Distrito Federal, o *cluster* 1 possui 267 entregas; *cluster* 2 compreende 234 entregas; *cluster* 3 abrange 150 entregas; *cluster* 4 possui 201; *cluster* 5 engloba 13 entregas; e *cluster* 6 tem 121 entregas.

Já para o estado do Pará, *cluster* 1 tem 16 entregas; *cluster* 2 tem 227 entregas; *cluster* 3 abrange 46 entregas; e *cluster* 4 possui 8 entregas.

Os resultados obtidos para as instâncias do Rio de Janeiro, Distrito Federal e Pará a partir dos experimentos computacionais realizados estão apresentados nas Tabelas 4, 5, 6 e 7.

Tabela 4: Resultados obtidos por meio da estratégia A para o Rio de Janeiro

Itens	Grupo_1	Grupo_2	Grupo_3	Grupo_4	Grupo_5
Caminhão	10	9	8	6	6
Demanda	1683	1596	1266	965	976
Distância	1315	1579	1356	1212	875
Pontos de entrega	300	291	228	162	171
Distância média	131.5	175.444	169.5	202	145.833

Tabela 5: Resultados obtidos por meio da estratégia A para o Rio de Janeiro

Itens	Grupo_6	Grupo_7	Grupo_8	Grupo_9	Grupo_10	Total
Caminhão	15	14	17	43	9	137
Demanda	2479	2476	2925	7592	1573	23531
Distância	828	1544	1057	3262	1604	14632
Pontos de entrega	453	458	549	1372	289	4273
Distância média	55.2	110.285	62.176	75.860	178.222	- - -

Tabela 6: Resultados obtidos por meio da estratégia A para o Distrito Federal

Itens	Grupo_1	Grupo_2	Grupo_3	Grupo_4	Grupo_5	Grupo_6	Total
Caminhão	9	8	4	5	1	4	31
Demanda	1439	1268	849	1088	77	632	5353
Distância	1113	908	498	715	112	948	4294
Entregas	267	234	150	201	13	121	986
D. média	123.666	113.5	124.5	143	112	237	- - -

Tabela 7: Resultados obtidos por meio da estratégia A para o Pará

Itens	Grupo_1	Grupo_2	Grupo_3	Grupo_4	Total
Caminhão	1	8	2	1	12
Demanda	81	1289	237	27	1634
Distância	402	921	318	208	1849
Pontos de entrega	16	227	46	8	297
Distância média	402	115.125	159	208	- - -

Para cada estado, são descritas os pontos de entrega (cidades), as distâncias percorridas, as demandas (tamanho da carga) o número necessário de caminhões para realizar as entregas e a distância média das rotas. Nas colunas, são apresentados todos os *clusters* criados pelo algoritmo. A última linha das Tabelas 5, 6 e 7 apresenta a soma dos resultados de cada parte das entregas realizadas.

Conforme observado na Tabela 4, referente ao Distrito Federal com 986 entregas, foram utilizados 31 veículos que percorreram 4.294 quilômetros.

Na Tabela 7 pode-se observar que o estado do Pará realizou 297 entregas para diversos pontos utilizando 12 veículos. Durante as entregas, os veículos percorreram um total de 1.849 quilômetros.

Por outro lado, no Rio de Janeiro, o número de entregas realizadas foi maior, totalizando 4.273, conforme Tabela 4 e 5. Para atender a essa demanda, foram utilizados 137 veículos que percorreram um total de 14.632 Km.

5.4.2 Estratégia baseada no algoritmo DBSCAN

A abordagem adotada nesta estratégia consiste em utilizar o algoritmo DBSCAN para realizar a tarefa de agrupamento de clientes com base em suas coordenadas geográficas. Antes da aplicação do BDSCAN, uma série de etapas de preparação dos dados é executada para assegurar que as colunas de latitude e longitude estejam prontas. Essas etapas incluem a substituição de vírgulas por pontos e a conversão dos valores de formato *string* para números.

O DBSCAN, um algoritmo de agrupamento espacial, opera com base em dois parâmetros fundamentais. O primeiro desses parâmetros é o ϵ (épsilon), que determina

a distância máxima entre dois pontos para que eles sejam considerados vizinhos. Nesse contexto, os valores de ϵ são calculados individualmente para três casos distintos: 1 Km para o Rio de Janeiro; 3 Km para o Distrito Federal e o 3.50 Km para o estado do Pará. Esses valores indicam a proximidade necessária entre os pontos para que sejam considerados como pertencentes a um mesmo *cluster*.

O segundo parâmetro fundamental é o *min_samples*, que define o tamanho mínimo de pontos necessários para formar um *cluster* válido. Quaisquer pontos que não se enquadrem nesse critério são categorizados como ruído. Os valores escolhidos para o *min_samples* variam de acordo com contexto geográfico: 10 para o Rio de Janeiro; 5 para o Distrito Federal; e 5 para o estado do Pará. Essa diferenciação leva em consideração a densidade esperada de pontos em cada região. Ao contrário do *k-means*, o DBSCAN não exige que especifique o número de *clusters* antecipadamente ele os determina automaticamente com base nos parâmetros ϵ e *min_samples*.

A métrica de haversine e o algoritmo da *árvore de bolas* são usados para calcular grandes distâncias circulares entre pontos. É importante observar que os valores dos ϵ e as coordenadas são convertidas em radianos, devido à necessidade da métrica haversine do *scikit-learn* de operar com unidades de radianos.

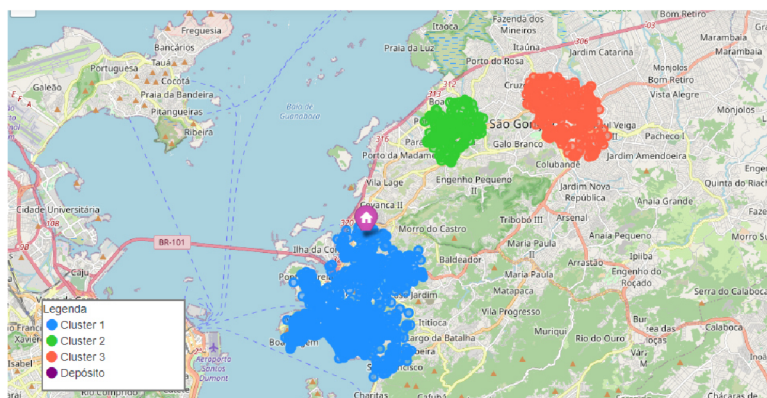


Figura 18: Agrupamento por DBSCAN no Rio de Janeiro

As Figuras 18, 19 e 20 ajudam na compreensão da distribuição geográfica dos *clusters*. Conforme pode-se notar, somente os pontos que se encontram dentro da distância definida pelo ϵ são selecionados para compor os *clusters*, sendo descartados aqueles que não atendam a essa premissa.

Do total de 4.273 entregas programadas para o Rio de Janeiro, o algoritmo selecionou 3.285 para inclusão nos *clusters*, enquanto 988 pontos foram descartados por não atenderem aos critérios. Para o Distrito Federal, onde inicialmente estavam previstas 986 entregas, 960 foram agrupados, e 26 pontos foram considerados ruído e excluídos. No estado do Pará, dos 297 pontos previstos, 27 foram identificados como ruído e não incluídos no agrupamento.

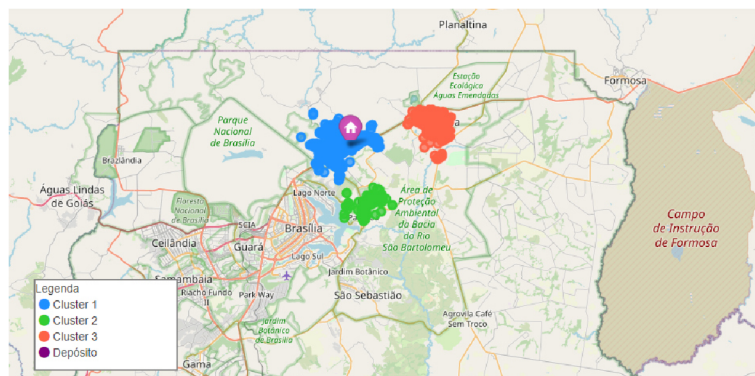


Figura 19: Agrupamento por DBSCAN no Distrito Federal

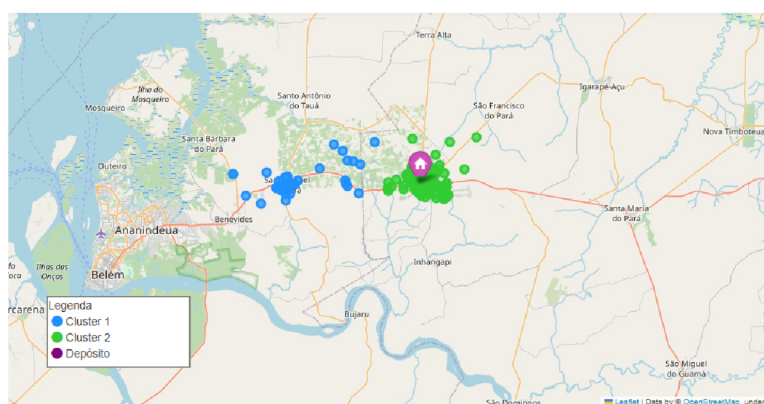


Figura 20: Agrupamento por DBSCAN no Estado do Pará

Conforme observado, este método é altamente sensível aos dois parâmetros mencionados. Se um valor muito pequeno for atribuído ao *min_samples* e ϵ , o algoritmo tende a formar um número grande de grupos e em caso de *min_samples* for definido como 1, é possível que o número de grupos seja equivalente a cardinalidade do conjunto de dados, não fazendo sentido executar um algoritmo de análise de agrupamento para definir cada elemento como um grupo. Outra dificuldade que pode surgir na definição dos parâmetros ocorre quando o valor de ϵ ser muito grande. Isso resulta na formação de apenas um único grupo, devido todos os elementos do conjunto de dados serem alcançáveis por densidade a partir de um único elemento, ou muito pequeno de modo a não permitir que os elementos sejam alcançáveis por densidade e consequentemente rotular todos os elementos como ruído.

Os resultados obtidos para as instâncias do Rio de Janeiro, Distrito Federal e Pará estão apresentados nas Tabelas [8](#), [9](#) e [10](#), respectivamente. Essas tabelas exibem informações como as distâncias percorridas, as demandas (tamanho da carga) o número necessário de caminhões para realizar as entregas e distância média de caminhões. Cada coluna representa um grupo e seus respectivos resultados.

Na Tabela [8](#), referente ao Rio de Janeiro, pode-se observar que, para um total de 3.285

Tabela 8: Resultados obtidos por meio da estratégia B para o Rio de Janeiro

Itens	Grupo_1	Grupo_2	Grupo_3	Total
Caminhão	58	8	12	78
Demanda	10287	1429	2144	13860
Distância	7087	833	2344	10264
Pontos de entrega	2182	482	621	3285
Distância média	122.189	104.125	195.333	- - -

Tabela 9: Resultados obtidos por meio da estratégia B para o Distrito Federal

Itens	Grupo_1	Grupo_2	Grupo_3	Total
Caminhão	17	5	8	30
Demanda	2855	788	1397	5040
Distância	2433	820	1201	4454
Pontos de entrega	551	144	265	960
Distância média	143.117	164	150.125	- - -

Tabela 10: Resultados obtidos por meio da estratégia B para o estado do Pará

Itens	Grupo_1	Grupo_2	Total
Caminhão	8	2	10
Demanda	1267	186	1453
Distância	1092	1009	2101
Pontos de entrega	230	40	270
Distância média	136.5	504.5	- - -

entregas, foram utilizados 78 caminhões, os quais percorreram uma distância total de 10.264 quilômetros.

No caso do Distrito Federal, como apresentado na Tabela 9, os caminhões percorreram 4.454 quilômetros para realizar 960 entregas.

Já no estado do Pará, conforme demonstrado na Tabela 10, onde havia 270 pontos de entrega, a distância total percorrida foi de 2.101 quilômetros.

Tabela 11: Resultados do Rio de Janeiro

ITENS	Estratégia A	Estratégia B	Abordagem da Loggi
Caminhão	137	78	145
Demanda	23531	13860	23531
Distância	14632	10264	16985
Pontos de entrega	4273	3285	4273
Desvio Padrão	627.735	3263.221	755.367

Nas Tabelas 11, 12 e 13, são apresentados os resultados consolidados para as estratégias em três estados. Ao considerar critérios como a qualidade da solução, em relação

Tabela 12: Resultados do Distrito Federal

ITENS	Estratégia A	Estratégia B	Abordagem da Loggi
Caminhão	31	30	32
Demanda	5353	5040	5353
Distância	4294	4454	4902
Pontos de entrega	986	960	986
Desvio Padrão	363.493	843.085	536.415

Tabela 13: Resultados do Pará

ITENS	Estratégia A	Estratégia B	Abordagem da Loggi
Caminhão	12	10	10
Demanda	1634	1453	1634
Distância	1849	2101	1520
Pontos de entrega	297	270	297
Desvio Padrão	315.981	58.689	457.570

aos objetivos preconizados, fica evidente que tanto Estratégias baseadas nos algoritmos *K-means* quanto ao DBSCAN apresentam soluções de alta qualidade.

Ao analisar as instâncias do Rio de Janeiro, conforme indicado na Tabela [11](#), percebe-se que a estratégia baseada no algoritmo DBSCAN demonstrou um desempenho superior em relação ao número de caminhões utilizados e à distância total percorrida. É importante ressaltar, porém, que a estratégia DBSCAN não utilizou o mesmo número de instâncias que a Estratégia *K-Means* e a abordagem da Loggi.

No caso da estratégia baseada no algoritmo *K-Means*, observa-se um benefício tanto em termos do número de caminhões quanto da distância total percorrida, quando comparada à abordagem da Loggi.

A Tabela [12](#) mostra os resultados para as instâncias do Distrito Federal. A estratégia baseada no algoritmo *K-Means* obteve um desempenho superior em relação ao números de caminhões e à distância total percorrida, quando comparada com a abordagem da Loggi. Em relação à estratégia baseada no algoritmo DBSCAN, é importante destacar que pontos de entrega eram diferentes, o que pode ter influenciado na comparação dos resultados.

Quanto aos resultados do estado do Pará, a abordagem da Loggi apresentou um desempenho melhor tanto no número de caminhões como na distância percorrida, em relação às estratégias propostas. Esses resultados estão apresentados na Tabela [13](#).

As Figuras [21](#), [22](#) e [23](#) ilustram uma comparação entre os itens das soluções obtidas por cada uma das estratégias e abordagem da Loggi nos testes conduzidos para as instâncias dos três estados em análise. A partir destas figuras, torna-se evidente que a estratégia baseada no algoritmo *K-Means*, exibe os menores desvios padrão para as instâncias do Rio de Janeiro e do Distrito federal. Isso sugere que ela é mais consistente

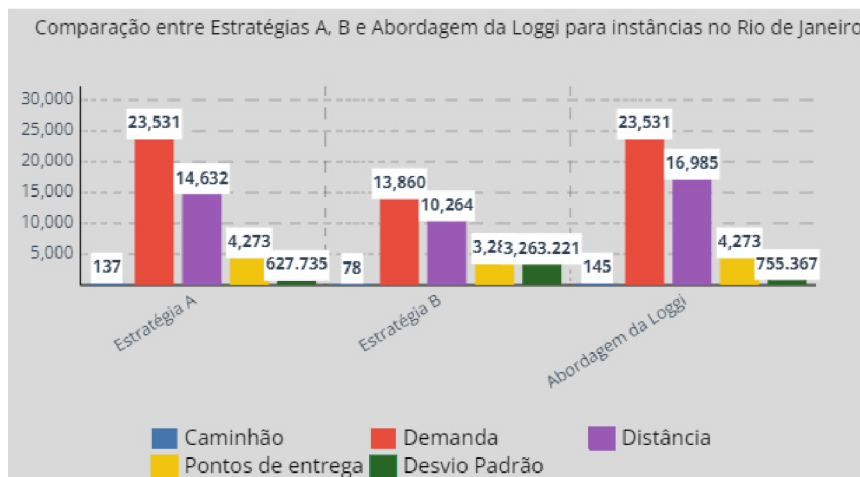


Figura 21: *Comparação entre Estratégias A, B e Abordagem da Loggi para instâncias no Rio de Janeiro*

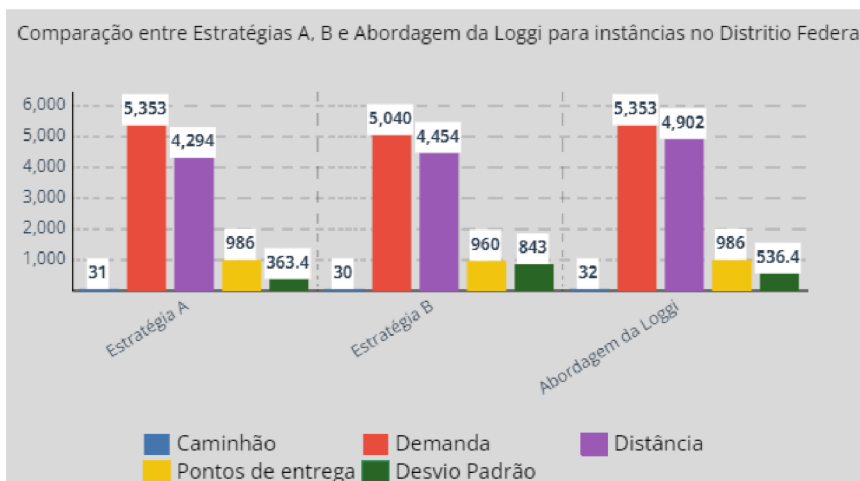


Figura 22: *Comparação entre Estratégias A, B e Abordagem da Loggi para instâncias no Distrito Federal*

e previsíveis aos resultados das instâncias destes dois estados.

Na Figura 23, destaca-se que a estratégia baseada no algoritmo DBSCAN, apresenta um desvio padrão menor, indicando uma distribuição mais consistente e previsível dos resultados às instâncias do Pará. É importante reiterar que quando menor for o valor do desvio padrão, mais eficientes foram as entregas e isso aponta para uma distribuição mais eficaz.

Como mencionado na etapa final da seção de modelagem das estratégias, a regra de combinador *ensemble*, baseada na combinação algébrica de soma, é uma técnica empregada para unificar os resultados individuais de cada rota otimizada, visando a obtenção de uma medida global da eficiência das rotas. Essa regra atua como o mecanismo central para combinar os valores individuais de distância percorrida em todas as rotas, resultando

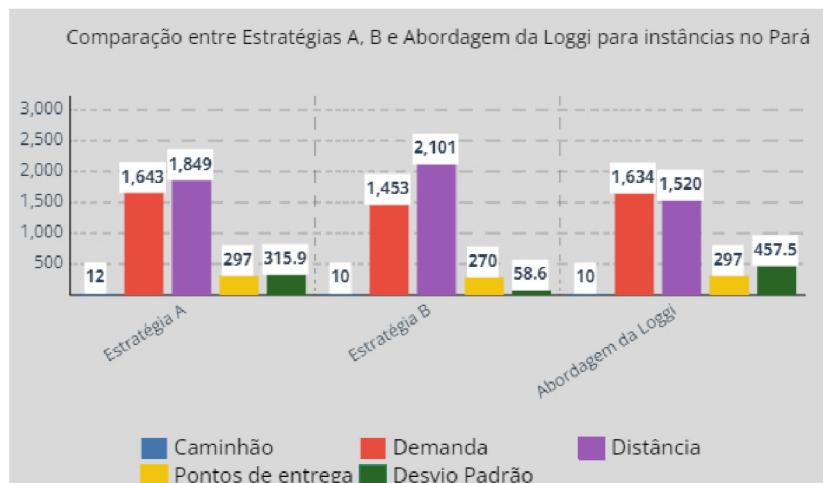


Figura 23: Comparação entre Estratégias A, B e Abordagem da Loggi para instâncias no Pará

em uma distância total que reflete o desempenho geral das estratégias adotadas.

O processo de aplicação dessa regra pode ser esclarecido da seguinte maneira:

Inicialmente, após a etapa de otimização, são obtidas as distâncias individuais de cada rota, representando as distâncias percorridas por cada veículo. A regra de combinador ensemble entra em ação, onde as distâncias individuais de todas as rotas são somadas através de uma expressão algébrica. A distância total obtida através da soma das distâncias individuais se torna uma medida unificada de desempenho. Quanto menor for o valor da distância total, mais eficientes serão as rotas em termos de minimização da distância percorrida.

6 CONCLUSÃO

Este trabalho teve como objetivo desenvolver e aplicar uma abordagem para a análise de dados espaciais, utilizando métodos heurísticos e técnica de *ensemble* para otimizar o processo de roteirização de uma frota de veículos.

Para alcançar esse objetivo, foram propostas duas estratégias distintas, cada uma empregando um modelo de clusterização para agrupar pontos de entrega (clientes) em grupos. Além disso, uma revisão bibliográfica foi realizada para destacar a importância do PRVC no campo da otimização combinatória. Essa revisão também apontou vários métodos exatos, heurísticos e meta-heurísticos que têm sido empregados para abordar esse desafiador problema.

A aplicação dos algoritmos de clusterização *K-means* e DBSCAN permitiu um agrupamento eficiente dos clientes, fornecendo informações essenciais para a criação de estratégias de roteirização. A implementação das duas estratégias distintas, cada uma centrada em um modelo específico de clusterização, demonstrou a flexibilidade da metodologia proposta.

Com base nos experimentos realizados, foi possível concluir que tanto estratégia baseada no algoritmo *K-Means* quanto a estratégia baseada no algoritmo DBSCAN apresentaram bons resultados para o PRVC. Elas responderam de forma satisfatória à minimização da distância total percorrida pelos caminhões, ao equilíbrio na distribuição das distâncias percorridas e ao desvio padrão.

Assim, pode-se dizer que tanto o objetivo geral quanto os objetivos específicos desta pesquisa foram atingidos.

No entanto, para estender essa pesquisa e contribuir ainda mais para o campo, várias oportunidades de trabalho futuro podem ser consideradas. Entre elas, destacam-se:

- Exploração de algoritmos de otimização meta-heurística, a consideração de restrições adicionais, como janelas de tempo, e a validação rigorosa com dados do mundo real;
- Ampliar as abordagens para incluir a coleta de itens, em adição às entregas;
- Realizar uma avaliação abrangente das estratégias propostas em cenários de larga

escala, com um grande número de clientes e veículos. Isso permitirá identificar eventuais desafios ou limitações que podem surgir quando a complexidade do problema aumenta;

- Dependendo da geografia da entrega na mesma cidade, cada parte dos clusters será utilizada em uma técnica de cluster diferente. Outra parte das entregas será associada à outra técnica de cluster, permitindo-nos oferecer o melhor serviço possível para esta cidade. O ensemble combinará esses elementos para otimizar nossos resultados;
- Implementar e testar as estratégias desenvolvidas em um ambiente real, como um aplicativo de entrega ou frota de veículos em operação.

Em síntese, este estudo oferece uma perspectiva valiosa sobre a roteirização de frotas de veículos através de análise de dados espaciais e métodos heurísticos. Ao abordar desafios contemporâneos e fornecer direções para pesquisas futuras, espera-se que este trabalho contribua para a melhoria das operações logísticas e de transporte.

REFERÊNCIAS

- [1] Achuthan, N. R., Caccetta, L., and Hill, S. P. (2003). An improved branch-and-cut algorithm for the capacitated vehicle routing problem. *Transportation Science*, 37(2):153–169.
- [2] Aggarwal, C. C. (2014). A survey of stream classification algorithms. *Data classification: algorithms and applications*, 245(1):30.
- [3] Agência CNT Transporte (2016). Custo logístico consome 12,7% do pib do brasil. [urlhttps://www.cnt.org.br/agencia-cnt/custo-logistico-consome-12-do-pib-do-brasil](https://www.cnt.org.br/agencia-cnt/custo-logistico-consome-12-do-pib-do-brasil).
- [4] Atual, P. A. C. T. (2023). Transporte de carga. [urlhttps://www.ilos.com.br/web/solucoes-por-tema/solucoes-por-tema-transportes/](https://www.ilos.com.br/web/solucoes-por-tema/solucoes-por-tema-transportes/).
- [5] Augerat, P., Naddef, D., Belenguer, J., Benavent, E., Corberan, A., and Rinaldi, G. (1995). Computational results with a branch and cut code for the capacitated vehicle routing problem.
- [6] Bard, J. F., Kontoravdis, G., and Yu, G. (2002). A branch-and-cut procedure for the vehicle routing problem with time windows. *Transportation Science*, 36(2):250–269.
- [7] Bastos Junior, F. C. D. (2020). Problema de roteamento de veículos-um estudo de caso em uma empresa de distribuição de jornais e revistas.
- [8] Bertoni, M. A. (2021). Aplicação do método conjunto stacking do classificador floresta de caminhos ótimos para o problema de detecção de intrusão. pages 43–107.
- [9] Bibimoune, M., Elghazel, H., and Aussem, A. (2013). *An empirical comparison of supervised ensemble learning approaches*. Disponível em: https://scholar.google.com.br/scholar?hl=pt-BR&as_dt=0
- [10] Bispo, R. C. (2018). *Planejador de roteiros turísticos: uma aplicação do problema do Caixeiro viajante na cidade do Recife*. Brasil.

- [11] Blasum, U. and Hochstättler, W. (2000). Application of the branch and cut method to the vehicle routing problem. *Zentrum für Angewandte Informatik Köln Technical Report zpr2000-386*.
- [12] Blum, C. and Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM computing surveys (CSUR)*, 35(3):268–308.
- [13] Borowski, M., Gora, P., Karnas, K., Błajda, M., Król, K., Matyjasek, A., Burczyk, D., Szewczyk, M., and Kutwin, M. (2020). *New hybrid quantum annealing algorithms for solving vehicle routing problem*. Springer.
- [14] Bramer, M. (2007). *Principles of data mining*. Springer.
- [15] Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.
- [16] Bruni, M., Guerriero, F., and Beraldi, P. (2014). Designing robust routes for demand-responsive transport systems. *Transportation Research Part E: Logistics and Transportation Review*, 70:1–16.
- [17] César Brandão de Oliveira, H. (2007). Um modelo híbrido estocástico para tratamento do problema de roteamento de veículos com janela de tempo.
- [18] Chen, R. and Tan, Y. (2021). A multi-branch ensemble agent network for multi-agent reinforcement learning. *International Conference on Data Mining and Big Data*, pages 485–498.
- [19] Cho, S.-B. and Kim, J. H. (1995). Combining multiple neural networks by fuzzy integral for robust classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(2):380–384.
- [20] Christofides, L. N. (1975). Supply Variables in Term Structure Equations. *Canadian Journal of Economics*, 8(2):276–281.
- [21] Clarke, C. (2003). Catia v 5 r 10: Staking new ground automation in process automation. *Desktop Engineering*, 8(8):30–32.
- [22] Clarke, G. and Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581.
- [23] Cordeau, J.-F. and Laporte, G. (2002). Tabu search heuristics for the vehicle routing problem. *published*, pages 1–17.
- [24] Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

- [25] Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations research*, 6(6):791–812.
- [26] Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management science*, 6(1):80–91.
- [27] Dasarathy, B. V. and Sheela, B. V. (1979). A composite classifier system design: Concepts and methodology. *Proceedings of the IEEE*, 67(5):708–713.
- [28] de Freitas, N. C. A. (2023). *Inteligência de negócios e análise de dados*. Editora Senac São Paulo.
- [29] Di Bartolomeo, M., Bajetta, E., Buzzoni, R., Mariani, L., Carnaghi, C., Somma, L., Zilembo, N., and Di Leo, A. (1996). Clinical efficacy of octreotide in the treatment of metastatic neuroendocrine tumors: a study by the italian trials in medical oncology group. *Cancer: Interdisciplinary International Journal of the American Cancer Society*, 77(2):402–408.
- [30] DOS REIS, D. C. S. (2007). Um algoritmo branch and bound para o problema da alocação ótima de monitores de qualidade de energia elétrica em redes de transmissão. *Juiz de Fora, MG*.
- [31] Enes, K. B., Villela, S. M., and Neto, R. F. (2016). *Uma abordagem baseada em Perceptrons balanceados para geração de ensembles e redução do espaço de versões*. Master's thesis, Universidade Federal de Juiz de Fora, [Online
- [32] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. 96(34):226–231.
- [33] Ferreira, K. M. (2022). Problema de roteamento de veículos com restrições de empacotamento bidimensional e entrega fracionada para a minimização da emissão de dióxido de carbono.
- [34] Fisher, M. L. and Jaikumar, R. (1981). A generalized assignment heuristic for vehicle routing. *Networks*, 11(2):109–124.
- [35] FRAGA, M. C. P. (2006). *Uma metodologia híbrida colônia de formigas–busca tabu–reconexão por caminhos para resolução do problema de roteamento de veículos com janelas de tempo*. Dissertação de Mestrado em Modelagem Matemática e Computação, Centro Federal
- [36] Gillett, B. E. and Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations research*, 22(2):340–349.

- [37] Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, 13(5):533–549.
- [38] GOLBARG, M. C. and LUNA, H. P. L. (2000). Otimização combinatória e programação linear. *Editora CAMPUS, Rio de Janeiro*.
- [39] Goldberg, M. C. and Luna, H. P. L. (2005). *Otimização combinatória e programação linear: modelos e algoritmos*. Elsevier.
- [40] Hansen, L. K. and Salamon, P. (1990). Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001.
- [41] H.Wolpert, D. (1992). Stacked generalization. *sciencedirect*. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0893608005800231>., 5(2):241–259.
- [42] Jiang, H., Lu, M., Tian, Y., Qiu, J., and Zhang, X. (2022). An evolutionary algorithm for solving capacitated vehicle routing problems by using local information. *Applied Soft Computing*, 117:108431.
- [43] Jordan, M. I. and Jacobs, R. A. (1994). Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214.
- [44] Kangah, J. K., Appati, J. K., Darkwah, K. F., and Soli, M. A. T. (2021). Implementation of an h-psoga optimization model for vehicle routing problem. *International Journal of Applied Metaheuristic Computing (IJAMC)*, 12(3):148–162.
- [45] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. (1983). Optimization by simulated annealing. *Science*, 220:671 – 680.
- [46] Kodinariya, T. M., Makwana, P. R., et al. (2013). Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95.
- [47] Kuncheva, L. I., Bezdek, J. C., and Duin, R. P. (2001). Decision templates for multiple classifier fusion: an experimental comparison. *Pattern recognition*, 34(2):299–314.
- [48] Lam, L. and Suen, C. Y. (1995). Optimal combinations of pattern classifiers. *Pattern Recognition Letters*, 16(9):945–954.
- [49] Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European journal of operational research*, 59(3):345–358.
- [50] Lee, T.-R. and Ueng, J.-H. (1999). A study of vehicle routing problems with load-balancing. *International Journal of Physical Distribution & Logistics Management*, 29(10):646–657.

- [51] Lemke, C., Budka, M., and Gabrys, B. (2015). Metalearning: a survey of trends and technologies. *Artificial intelligence review*, 44(1):117–130.
- [52] Li, S., Gong, W., Yan, X., Hu, C., Bai, D., Wang, L., and Gao, L. (2019). Parameter extraction of photovoltaic models using an improved teaching-learning-based optimization. *Energy Conversion and Management*, 186:293–305.
- [53] Lima, S. J. D. A. (2015). *Otimização do problema de roteamento de veículos capacitado usando algoritmos genéticos com heurísticas e representações cromossômicas alternativas*. Universidade Nove de Julho.
- [54] Lin, S. (1965). Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44(10):2245–2269.
- [55] Lin, S. and Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516.
- [56] Lu, Hao e Zhang, X. e. Y. S. (2020). *A learning-based iterative method for solving vehicle routing problems*. International conference on learning representations.
- [57] Luizelli, M. C. (2012). Problema de roteamento de veículos aplicado ao problema de despacho de ordens de serviço.
- [58] MacQueen, J. (1967). Classification and analysis of multivariate observations. *University of California Los Angeles LA USA*, pages 281–297.
- [59] Marsten, R. E., Muller, M. R., and Killion, C. L. (1979). Crew planning at flying tiger: a successful application of integer programming. *Management Science*, 25(12):1175–1183.
- [60] Matsueda, L. C. O. (2015). Análise e otimização do problema de roteamento de veículos com muitos objetivos e janelas de tempo flexíveis.
- [61] Newton, R. M. and Thomas, W. H. (1974). Bus routing in a multi-school system. *Computers & Operations Research*, 1(2):213–222.
- [62] Nicholson, T. (2007). *Optimization in industry: Optimization techniques*. Transaction Publishers.
- [63] Nilsson, N. and Machines, L. (1965). Foundations of trainable pattern classifying systems. *McGraw-Hill, New York OBrien RM (2007) A caution regarding rules of thumb for variance ination factors. Qual Quant*, 41(1):3–673.
- [64] Oliveira, H. C. B. d. (2007). Um modelo híbrido estocástico para tratamento do problema de roteamento de veículos com janela de tempo. *Repositório Digital da UFPE*, pages 38–121.

- [65] Opitz, D. and Maclin, R. (1999). Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198.
- [66] Pereira, D. L. (2010). Heurísticas e algoritmo exato para o problema de roteamento de veículos com coleta e entrega simultâneas. *Universidade Federal de Minas Gerais*.
- [67] Pimentel, C. (2005). Algoritmos de partição e geração de colunas para dimensionamento de lotes de produção.
- [68] Polamuri, S. (2020). *Ensemble methods bagging vs boosting difference*. Disponível em: <https://dataaspirant.com/ensemble-methods-bagging-vs-boosting-difference/t-1599488265664>.
- [69] Polikar, R. (2009). *Ensemble learning*. *Scholarpedia* 4 (1): 2776. Disponível em: doi:10.4249/scholarpedia.2776.
- [70] Polikar, R. (2012). Ensemble machine learning. *Springer*, pages 1–34.
- [71] Priy, S. (2013). Clustering in machine learning. *GeeksforGeeks*.
- [72] Renaud, J. and Boctor, F. F. (2002). A sweep-based algorithm for the fleet size and mix vehicle routing problem. *European Journal of Operational Research*, 140(3):618–628.
- [73] Rogova, G. (2008). *Combining the results of several neural network classifiers*. Springer.
- [74] Sakkis, G., Androutopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., and Stamatopoulos, P. (2001). Stacking classifiers for anti-spam filtering of e-mail. *arXiv preprint cs/0106040*, pages 44–50.
- [75] Sammouda, R., El-Zaart, A., et al. (2021). An optimized approach for prostate image segmentation using k-means clustering algorithm with elbow method. *Computational Intelligence and Neuroscience*, 2021.
- [76] Sankaran, J. K. and Ubgade, R. R. (1994). Routing tankers for dairy milk pickup. *Interfaces*, 24(5):59–66.
- [77] Santos, A. G. (2022). Uma meta-heurística adaptativa aplicada ao problema de roteamento de veículos capacitados: estudo de caso de uma transportadora de pacotes.
- [78] Santos, R. L. and Leal, J. (2006). Uma aplicação de algoritmos de colônias de formigas em problemas de roteirização de veículos com janelas de tempo.
- [79] Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, 5(2):197–227.

- [80] Silva, C. E. d. (2020). *Coordenação de Múltiplos Veículos Autônomos de Entrega Usando K-Means e Algoritmos Bio-Inspirados*. UFB Disponível em: [ur-https://repositorio.ufu.br/handle/123456789/29807](https://repositorio.ufu.br/handle/123456789/29807).
- [81] Singh, A. (2018). *A Comprehensive Guide to Ensemble Learning with Python codes*. Disponível em: <https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/>.
- [82] Siqueira, R. (2017). *Comparação de metodos heurísticos para otimização de rotas de distribuição no município de Cascavel-PR*. Universidade Tecnológica Federal do Paraná.
- [83] Tan, P.-N., Steinbach, M., and Kumar, V. (2016). *Introduction to data mining*. Pearson Education India.
- [84] Temponi, E. C. C. (2007). Uma proposta de resolução do problema de corte bidimensional via abordagem metaheurística. 100p. *Dissertação (Mestrado em modelagem matemática computacional)–Centro Federal de Educação Tecnológica de Minas Gerais, Belo Horizonte-MG*.
- [85] Ting, K. M. and Witten, I. H. (1999). Issues in stacked generalization. *Journal of artificial intelligence research*, 10:271–289.
- [86] Toth, P. and Vigo, D. (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1-3):487–512.
- [87] Vieira, H. (2013). Metaheurística para a solução de problemas de roteamento de veículos com janela de tempo. campinas: Unicamp, 2013. 108p. *Dissertação (Mestrado em Matemática Aplicada)–Instituto de Matemática ... Disponível em: https://www.ime.unicamp.br/chico/tese_heloisa.pdf*.
- [88] Villanueva, W. J. P. (2006). *Comitê de máquinas em predição de séries temporais*. Faculdade de Engenharia Elétrica e de Computação.
- [89] Wang, F., Liao, F., Li, Y., Yan, X., and Chen, X. (2021). An ensemble learning based multi-objective evolutionary algorithm for the dynamic vehicle routing problem with time windows. *Computers & Industrial Engineering*, 154:107131.
- [90] Wikipédia, a. e. l. (2022). *Problema de roteamento de veículos capacitados*. Wikimedia. Disponível em: https://en.wikipedia.org/wiki/Vehicle_routing_problem.
- [91] Wolpert, D. H. (1992). Stacked generalization. *Neural networks*, 5(2):241–259.

- [92] Woods, K. S., Bowyer, K., and Kegelmeyer, W. P. (1996). Combination of multiple classifiers using local accuracy estimates. *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 391–396.
- [93] Xu, L., Krzyzak, A., and Suen, C. Y. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE transactions on systems, man, and cybernetics*, 22(3):418–435.