UNIVERSIDADE FEDERAL DE OURO PRETO

ALGORITMOS META-HEURÍSTICOS PARA O PROBLEMA *DIAL-A-RIDE*

André Luyde da Silva Souza

Orientadores: Puca Huachi Vaz Penna
Marcone Jamilson Freitas Souza

Ouro Preto
Junho de 2019

UNIVERSIDADE FEDERAL DE OURO PRETO

UFOP

decom departamento de computação

ALGORITMOS META-HEURÍSTICOS PARA O PROBLEMA *DIAL-A-RIDE*

André Luyde da Silva Souza

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Ciência da Computação, da Universidade Federal de Ouro Preto, como parte dos requisitos necessários à obtenção do título de Mestre em Ciência da Computação.

Orientadores: Puca Huachi Vaz Penna
Marcone Jamilson Freitas Souza

Ouro Preto
Junho de 2019

S895a Souza, André Luyde da Silva .
Algoritmos meta-heurísticos para o problema dial-a-ride [manuscrito] / André Luyde da Silva Souza. - 2019.
viii, 50f.: il.: tabs.

Orientador: Prof. Dr. Puca Huachi Vaz Penna.
Coorientador: Prof. Dr. Marcone Jamilson Freitas Souza.

Dissertação (Mestrado) - Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Biológicas. Departamento de Computação. Programa de Pós-Graduação em Ciência da Computação.
Área de Concentração: Ciência da Computação.

1. Transporte urbano. 2. Trânsito urbano. 3. Problema de roteamento de veículos. I. Penna, Puca Huachi Vaz . II. Souza, Marcone Jamilson Freitas . III. Universidade Federal de Ouro Preto. IV. Titulo.

CDU: 004.023

Catalogação: www.sisbin.ufop.br

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UFOP

ATA DE DEFESA DE DISSERTAÇÃO

Aos 12 dias do mês de junho do ano de 2019, às 10:00 horas, nas dependências do Departamento de Computação (Decom), foi instalada a sessão pública para a defesa de dissertação do mestrando **André Luyde da Silva Souza**, sendo a banca examinadora composta pelo Prof. Dr. Puca Huachi Vaz Penna (Presidente - UFOP), pelo Andre Gustavo dos Santos (Membro - Externo), pelo Prof. Dr. Tulio Angelo Machado Toffolo (Membro - UFOP) e pelo Prof. Dr. Marcone Jamilson Freitas Souza (Co-Orientador - UFOP). Dando início aos trabalhos, o presidente, com base no regulamento do curso e nas normas que regem as sessões de defesa de dissertação, concedeu ao mestrando 40 minutos para apresentação do seu trabalho intitulado "Algoritmos Meta-Heurísticos para o Problema Dial-A-Ride". Terminada a exposição, o presidente da banca examinadora concedeu, a cada membro, um tempo máximo de 30 minutos para perguntas e respostas ao candidato sobre o conteúdo da dissertação, na seguinte ordem: Primeiro, Prof. Andre Gustavo dos Santos; segundo, Prof. Tulio Angelo Machado Toffolo; terceiro, Prof. Marcone Jamilson Freitas Souza; quarto, Prof. Puca Huachi Vaz Penna. Dando continuidade, ainda de acordo com as normas que regem a sessão, o presidente solicitou aos presentes que se retirassem do recinto para que a banca examinadora procedesse à análise e decisão, anunciando, a seguir, publicamente, que o mestrando foi aprovado por unanimidade, sob a condição de que a versão definitiva da dissertação deva incorporar todas as exigências da banca, devendo o exemplar final ser entregue no prazo máximo de 6 (seis) meses à Coordenação do Programa. Para constar, foi lavrada a presente ata que, após aprovada, vai assinada pelos membros da banca examinadora e pelo mestrando. Ouro Preto, 12 de junho de 2019.

Prof. Dr. Puca Huachi Vaz Penna
Presidente

Prof. Dr. Marcone Jamilson Freitas Souza

Andre Gustavo dos Santos

Prof. Dr. Tulio Angelo Machado Toffolo

André Luyde da Silva Souza
Mestrando

Resumo da Dissertação apresentada à UFOP como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

ALGORITMOS META-HEURÍSTICOS PARA O PROBLEMA *DIAL-A-RIDE*

André Luyde da Silva Souza

Junho/2019

Orientadores: Puca Huachi Vaz Penna
Marcone Jamilson Freitas Souza

Programa: Ciência da Computação

Este trabalho trata do problema *Dial-a-Ride*, que consiste em fazer rotas para veículos com a finalidade de transportar pacientes de diferentes locais para realizar exames médicos em unidades de tratamento de saúde. O *Dial-a-Ride* é uma extensão do Problema de Roteamento de Veículos, possuindo características do Problema de Roteamento de Veículos com Janela de Tempo e do Problema de Roteamento de Veículos com Coleta e Entrega, combinados com restrições relativas aos pacientes. O trabalho considera a forma estática do problema e utiliza dados obtidos da Prefeitura Municipal de Ouro Preto-MG para modelagem e contextualização do problema. Para resolvê-lo, propõe-se dois algoritmos heurísticos, MS-VNS1 e VNS2, ambos baseados na meta-heurística *Variable Neighborhood Search* (VNS). O primeiro, MS-VNS1, é guiado pela meta-heurística *Multi-Start* tendo como busca local o VNS. O segundo, por sua vez, é guiado apenas pelo VNS. Nos dois algoritmos o método de busca local do VNS é o procedimento heurístico *Randomized Variable Neighborhood Descent* (RVND), o qual usa os movimentos de realocação, troca e cruzamento para explorar o espaço de soluções do problema. Os resultados computacionais foram obtidos pela aplicação dos algoritmos em um conjunto de instâncias da literatura e comparados com os das melhores soluções desta variante do problema. Apesar de simples, os algoritmos desenvolvidos foram capazes de encontrar a solução ótima para algumas instâncias e soluções de boa qualidade para as demais. Os algoritmos também foram testados em um conjunto de instâncias criadas a partir de dados fornecidos pela Prefeitura Municipal de Ouro Preto-MG. Ambos se mostraram capazes de atender as demandas da cidade de Ouro Preto de forma automatizada, proporcionando ao setor de transporte da prefeitura uma ferramenta que possibilita reduzir

os custos com o transporte de pacientes e diminuir a alocação de funcionários para cumprir essa atividade.

Sumário

Capítulo 1

Introdução

Segundo Lutz *et al.* (2008), o aumento da expectativa de vida e o declínio da taxa de natalidade vêm resultando no envelhecimento da população. Os autores apresentaram um provável aumento na velocidade do envelhecimento da população nas próximas décadas e um contínuo envelhecimento da população ao longo do século. Esse fato tem aumentado a necessidade de melhoria no setor de transporte de pessoas com dificuldade de locomoção. De acordo com Cordeau & Laporte (2007), isso tem ocorrido principalmente nos países ocidentais em função da necessidade de melhorar os serviços de saúde em relação ao transporte.

Na cidade de Ouro Preto – MG, o setor de transporte da Prefeitura Municipal de Ouro Preto (PMOP), diariamente, leva pacientes e funcionários da área de saúde para a região metropolitana de Belo Horizonte – MG. Este serviço de transporte tem por finalidade a realização de exames médicos, tratamentos, consultas, entre outros serviços relacionados à saúde. Tendo isso em vista, a PMOP percebeu a necessidade de melhoria no serviço prestado. Analisando-se os dados fornecidos pelo setor de transporte da PMOP, esse serviço pode ser classificado como um problema de otimização. Sendo este, uma variante do Problema de Roteamento de Veículos (VRP, *Vehicle Routing Problems*) denominada Problema *Dial-a-Ride* (DARP, *Dial-a-Ride Problem*).

Segundo Toth & Vigo (2002), o VRP consiste em determinar um conjunto de rotas a serem executadas por uma frota de veículos, com o intuito de atender determinadas requisições feitas por um conjunto de usuários. Diversas variantes do VRP vêm sendo estudadas ao longo dos anos e uma delas é o DARP. Esse problema consiste no planejamento de rotas para um conjunto de veículos com a finalidade de transportar usuários de um determinado ponto a outro. Do ponto de vista de modelagem, pode-se considerar que o DARP é uma generalização de algumas variantes VRPs. Ele pode ser considerado como a combinação do Problema de Roteamento de Veículos com Coleta e Entrega (PDVRP, *Pickup and Delivery Vehicle Routing Problem*) e do Problema de Roteamento de Veículos com Janela de Tempo (VRPTW,

Vehicle Routing Problem with Time Windows). A resolução do problema consiste em encontrar um conjunto de rotas com custo mínimo, que transporte uma determinada quantidade de usuários de um local a outro, respeitando-se as restrições definidas pelo problema. No entanto, o DARP difere destes problemas no que diz respeito à perspectiva humana, pois o conforto e a comodidade dos usuários são levados em consideração (Cordeau & Laporte, 2007).

Na literatura é possível encontrar algumas variantes do DARP. Ele pode ser classificado em relação ao tipo de frota de veículos, tipo de requisições, quantidade de depósitos e tipo de usuários. Em relação às requisições, ele pode ser classificado como estático ou dinâmico. No estático, todas as requisições são informadas *a priori* e no dinâmico as requisições são realizadas no decorrer do trajeto e as rotas são atualizadas em tempo real. Para a quantidade de depósitos considera-se o DARP com um único depósito (garagem) ou o multi depósitos, de maneira que a variante com garagem única, todos os veículos são concentrados em um único local e no multi depósito os veículos são armazenados em locais diferentes. Em relação à frota de veículos, o DARP pode ser classificado como heterogêneo ou homogêneo, sendo que no heterogêneo os veículos utilizados são diferentes e no homogêneo todos os veículos são iguais, ambos em relação a capacidade. Além disso, quando é levado em consideração diferentes tipos de usuários, novamente, ele pode ser classificado como heterogêneo e homogêneo. As variantes em relação aos usuários são diretamente ligadas às variantes dos veículos, pois trata-se de diferentes tipos de usuários que possuem necessidades distintas quanto ao transporte. Sendo assim, o DARP é considerado heterogêneo quando há diferentes tipos de usuários e homogêneo quando não há diferença entre eles (Ho *et al.* , 2018).

Em relação ao objetivo do problema, é possível encontrar variantes com um único objeto e outras que consideram mais de um objetivo, ou seja, variantes Bi-objetivo e Multiobjetivo. Em vários trabalhos são encontrados combinações das variantes citadas acima. Por fim, uma característica interessante do DARP é que muitos trabalhos utilizam informações fornecidas por empresas, ou seja, utilizam dados reais, embora alguns trabalhos utilizem dados gerados computacionalmente (Ho *et al.* , 2018).

Diversas técnicas de resolução do problema são apresentadas na literatura. Mas como o DARP é um problema da classe NP-difícil (Mauri & Lorena, 2009), ele é normalmente resolvido por meio de métodos heurísticos e meta-heurísticos. Entre esses métodos, podem ser citados os seguintes: Busca Tabu (TS, *Tabu Search*), *Simulated Annealing* – SA, Algoritmos Genéticos (GA, *Genetic Algorithm*), *Variable Neighborhood Search* – VNS, *Iterated Local Search* – ILS, dentre outros (Cordeau, 2006).

1.1 Objetivos

Este trabalho tem como objetivo geral o desenvolvimento de métodos heurísticos para resolver o DARP, baseado na características do problema de transporte identificado no setor de saúde da cidade de Ouro Preto.

São os seguintes os objetivos específicos:

1. Desenvolver métodos heurísticos, baseado nas meta-heurísticas *Multi-Start* – MS, e *Variable Neighborhood Search* –VNS;

2. Tratar um problema real que ocorre na cidade de Ouro Preto, Minas Gerais;

3. Testar os métodos propostos em variantes da literatura com características semelhantes ao caso de Ouro Preto;

4. Criar instâncias de teste baseadas em dados reais fornecidos pelo setor de transporte da PMOP;

5. Realizar testes e apresentar resultados nas instâncias que mostram o cenário da cidade.

1.2 Justificativa

O setor de transporte da PMOP, diariamente, leva pacientes do Sistema Único de Saúde (SUS) para a região metropolitana da cidade de Belo Horizonte para tratamentos relacionados a saúde. Após uma análise dos dados e funcionamento do serviço prestado, ele foi classificado como o DARP. Atualmente, as decisões sobre alocações de pacientes e utilização dos veículos, bem como a definição da ordem de embarque (coleta) e desembarque (entrega) dos pacientes nos pontos específicos, são definidas de acordo com o conhecimento prévio dos funcionários da PMOP.

Neste sentido, a utilização de métodos computacionais pode melhorar as condições dos pacientes, diminuir os custos com as viagens e ter um melhor aproveitamento em relação a quantidade dos veículos utilizados. Portanto, com a aplicação dos métodos propostos, espera-se que haja uma melhoria significativa na qualidade do serviço prestado pela PMOP a população da cidade.

1.3 Metodologia

O desenvolvimento deste trabalho envolverá as seguintes atividades metodológicas:

1. Revisão de literatura: leitura de artigos de revisão sobre técnicas de solução para o problema sob estudo, bem como dos métodos MS e VNS, em suas

versões mais recente, utilizando-se o Portal de Periódicos da CAPES, bem como buscas na internet;

2. Concepção e implementação de estruturas de vizinhança para explorar o espaço de soluções do DARP;

3. Implementação de dois algoritmos, ambos baseados na meta-heurística VNS, porém um guiado pelos métodos MS e VNS e o outro guiado apenas pelo VNS, para resolução do DARP;

4. Calibração dos parâmetros do algoritmo desenvolvido utilizando a ferramenta IRACE.

1.4 Estrutura do trabalho

O restante deste trabalho está organizado como segue. No Capítulo 2 é apresentada a revisão da literatura sobre o DARP. O Capítulo 3 descreve o problema abordado. No Capítulo 4 são detalhadas as meta-heurísticas propostas. No Capítulo 5 são apresentados os experimentos e os resultados alcançados. Finalmente, o Capítulo 6 conclui o trabalho.

Capítulo 2

Revisão da Literatura

Nos últimos anos muitos estudos vêm sendo desenvolvidos para melhoria na área de roteamento de veículos. Problemas de coleta e entrega vêm ganhando uma atenção especial pelo grande número de aplicações reais, assim como os problemas com janela de tempo. O *Dial-a-Ride Problem* (DARP) engloba características desses dois problemas, com algumas outras restrições. O DARP vem sendo estudado e modelado de diversas maneiras de modo a atender as necessidades de cada ambiente de estudo. Diversas técnicas foram estudadas e aplicadas para resolver cada variante do problema. Algoritmos exatos foram utilizados para resolver as versões mais simples do problema, que têm um número reduzido de requisições. Para as abordagens de dimensões mais elevadas, em que o número de requisições é elevado, métodos heurísticos e meta-heurísticos vêm sendo cada vez mais utilizados, assim como os algoritmos que hibridizam essas técnicas.

O DARP pode ser dividido em dois grandes grupos: o estático (SDARP, *Static Dial-a-Ride* Problem) e o dinâmico (DDARP, *Dynamic Dial-a-Ride* Problem) (Cordeau & Laporte, 2007). Nas duas próximas seções são feitas revisões de trabalhos desses dois grupos.

2.1 Problema *Dial-a-ride* Estático (SDARP)

No SDARP todas as informações do problema são conhecidas *a priori*. As rotas e a utilização dos veículos são calculadas e decididas antes do início dos percursos. As aplicações do modelo estático em casos reais são menos recorrentes, na maioria dos casos são utilizados problemas-teste disponíveis na literatura para validar as técnicas construídas para a resolução do problema. Em meio aos trabalhos com aplicações reais podemos citar o de Paquette *et al.* (2013), que estruturou os dados do problema na cidade de Montreal, utilizando uma frota de veículos heterogênea e restrições de tipos de usuários que afetam na capacidade dos veículos. Eles trataram o problema com uma meta-heurística que integra eficientemente um Método

de Referência de Ponto para Otimização Multicritério (RPMMO, *Reference Point Method for Multicriteria Optimization*) dentro de uma Busca Tabu. Os resultados mostraram que o algoritmo chegou a um rico conjunto de soluções e determinou bons resultados em relação ao custo e a qualidade de serviço. Segundo os autores o algoritmo também pode ser usado para outras versões do DARP.

Para a cidade de Vitória – ES, Glaydston *et al.* (2014) apresentaram um método *Cluster-Search* – CS para resolver o DARP. Em sua pesquisa o problema foi caracterizado como multicritério, com uma frota de veículos homogênea e garagem única. Com objetivo minimizar o custo do transporte, total de veículos utilizados e tempo de espera dos usuários, a abordagem resolveu as particularidades encontradas na cidade. Assim, bons resultados foram apresentados nos testes realizados nas instâncias criadas com dados fornecidos pela Secretaria de Transportes, Trânsito e Infraestrutura de Vitória.

Na Bélgica, Molenbruch *et al.* (2017) utilizaram uma heurística de busca local multidirecional integrada com uma Descida de Vizinhança Variável (VND, *Variable Neighborhood Descent*), que, aplicado a uma lista de candidatos inteligentes, reduziu o tempo computacional do método. Eles consideraram o modelo bi-objetivo para o DARP, tendo como objetivos a redução dos custos e maximização da qualidade do serviço. A qualidade do serviço é medida em relação ao tempo de permanência do usuário no veículo. Os autores utilizaram dados de uma organização de transportes que atende demandas para consultas hospitalares, tratamentos médicos, atividades de creche e terapias de reabilitação para fazer seus experimentos. Eles analisaram diferentes tipos de combinações de restrições e concluíram que elas são de grande impacto no custo final, embora isso dependa muito das requisições.

Os trabalhos que não tratam diretamente casos reais normalmente utilizam *benchmarks* disponíveis na literatura para testar suas abordagens. Entre os *bechmarks* disponíveis, um muito utilizado é o definido por Cordeau & Laporte (2003). Em seu trabalho esses autores utilizaram três métodos heurísticos combinados com uma Busca Tabu. Dentre os métodos, o primeiro tem por objetivo apenas minimizar violações nas janelas de tempo, o segundo tem como objetivo minimizar a duração das rotas, além das violações nas janelas e o terceiro reduz o tempo de viagem dos usuários, além de minimizar a duração das rotas e violações nas janelas. Experimentos foram realizados com a execução da Busca Tabu em conjunto com cada um dos métodos. O terceiro método apresentou melhores resultados, porém, demandando maior tempo computacional. Utilizando informações fornecidas pela *Montreal Transit Commission* (MTC), os autores criaram 20 instâncias para realizar seus experimentos. Outros experimentos também foram realizados em instâncias obtidas de problemas reais fornecidos por uma empresa dinamarquesa.

Mais tarde, Cordeau (2006) abordou o DARP utilizando o algoritmo *Branch-*

and-Cut. Para garantir uma resolução ótima para o problema, o autor utilizou instâncias aleatórias geradas com o máximo de 48 usuários. Ropke *et al.* (2007) implementaram dois algoritmos diferentes baseados em *Branch-and-Cut* para trabalhar com o DARP e com o Problema de Coleta e Entrega com Janela de Tempo (PDPTW, *Pickup and Delivery Problem with Time Windows*). Para realizar os testes com o DARP eles utilizaram as instâncias criadas por Cordeau (2006). Eles também adicionaram algumas instâncias com um número maior de requisições para o DARP. Seus testes mostraram que ambas as implementações são competitivas com os resultados já obtidos. No caso do DARP, eles mostraram ser capazes de resolver instâncias maiores.

Já Jorgensen *et al.* (2007) trataram o DARP com uma abordagem evolutiva por meio de um GA. Eles utilizaram a estratégia de "agrupar primeiro" e "rotear depois" para solucionar o problema. A distribuição dos usuários aos veículos é feita pelo GA e o roteamento e determinação dos horários de coleta são realizados por outra heurística. Seus experimentos foram realizados nas instâncias de Cordeau & Laporte (2003). Mauri & Lorena (2009), em uma abordagem multiobjetivo, utilizaram o algoritmo SA para resolver o DARP. O SA foi adaptado para trabalhar com frotas homogêneas ou heterogêneas. Os autores trabalharam com três movimentos de troca selecionados aleatoriamente para gerar novas soluções vizinhas e utilizaram modelos heurísticos separadamente para roteirizar e programar as rotas. Os autores utilizaram as instâncias de Cordeau & Laporte (2003) para testar sua abordagem e obtiveram bons resultados.

Rahmani *et al.* (2017) fizeram uma extensão da formulação de programação linear inteira proposta por Cordeau (2006), adicionando penalizações no tempo de espera do usuário e no tempo total da rota na função objetivo. O algoritmo proposto foi uma heurística baseada em programação dinâmica, com aplicação de algumas agregações de espaço de estado implícitos e uma regra de domínio não exato. Os testes foram feitos nas instâncias de Cordeau (2006). Para instâncias menores o algoritmo desenvolvido conseguiu os mesmos resultados em um tempo consideravelmente menor e para instâncias maiores as ramificações e cortes heurísticos convergiram mais rápidos em 25% das instâncias.

Ainda nos trabalhos que utilizam *benchmark* da literatura para os testes, a variante heterogênea do DARP tem como um dos grupos de instâncias mais utilizados, o criado por Parragh (2011), que adicionou mais alguns tipos de restrições ao DARP clássico e propôs um algoritmo *Branch-and-Cut* para resolver o problema. Em seu trabalho, a autora adicionou características reais do problema nas instâncias de Cordeau (2006), sendo elas, heterogeneidade para os usuários e para os veículos. Ela encontrou soluções ótimas para as instâncias com até 40 requisições. Em seguida, Parragh & Schmid (2013) propuseram um algoritmo híbrido de Geração de Colunas

(CG, Column Generation) com uma Busca em Vizinhança de Larga Escala (LNS, *Large Neighborhood Search*). Eles compararam várias estratégias diferentes de hibridização nas instâncias de Cordeau & Laporte (2003). Das 20 instâncias utilizadas nos testes, em nove delas eles melhoraram a melhor solução conhecida em até 1,1%. Atualmente as melhores soluções para oito dessas instâncias ainda são as encontradas por Parragh & Schmid (2013). Outros testes foram feitos nos dois conjuntos de instâncias de Cordeau (2006) e Ropke *et al.* (2007). Os autores encontraram soluções ótimas em 13 das 21 instâncias em cada conjunto testado, totalizando 26 soluções ótimas das 42 testadas.

Da mesma forma, Braekers *et al.* (2014) trabalharam no problema propondo uma abordagem que utiliza um algoritmo *Branch-and-Cut* combinando com uma meta-heurística *Deterministic Annealing – DA* baseada no SA. Eles realizaram seus experimentos nas instâncias criadas por Cordeau & Laporte (2003), Cordeau (2006) juntamente com as adições feitas por Ropke *et al.* (2007) e Parragh (2011). O primeiro conjunto de Cordeau & Laporte (2003) é uma versão clássica do DARP. Eles igualaram ou superam a maioria dos melhores resultados encontrados para esse conjunto de instâncias. No segundo conjunto de Cordeau (2006) e Ropke *et al.* (2007), eles encontraram soluções melhores e com um tempo computacional menor e no terceiro conjunto de Parragh (2011), que se trata de uma versão heterogênea do problema, eles encontraram a solução ótima para 29 das 36 instâncias. A maioria das soluções encontradas pelos autores ainda é o estado da arte do problema.

Masmoudi *et al.* (2017) resolveram esta mesma versão do DARP utilizando um GA híbrido. Eles utilizaram heurísticas de construção, operadores de *crossover* e técnicas de buscas locais especificamente adaptadas para esta variante do problema. O algoritmo foi testado em 92 instâncias de pequeno e médio porte de Cordeau (2006), Parragh (2011) e Braekers *et al.* (2014) e eles criaram 40 novas instâncias maiores utilizando as mesmas métricas utilizadas por Parragh (2011). Os experimentos apresentados mostraram a eficácia da abordagem em comparação com o estado da arte do problema.

2.2 Problema *Dial-a-ride* Dinâmico (DDARP)

No DDARP, parte das requisições são conhecidas *a priori*, e o restante é revelado no decorrer da rota. No DDARP os trabalhos utilizando dados reais são mais comuns, principalmente na área da saúde, devido a alta semelhança do comportamento da versão dinâmica do problema com a ocorrência de emergências e atendimentos de última hora em centros de saúde. Ainda assim, muitos trabalhos utilizam instâncias da literaruta para seus experimentos.

Attanasio *et al.* (2004) têm como principal objetivo, atender o maior número de requisições. No trabalho, os autores implementaram uma Busca Tabu originalmente desenvolvida por Cordeau & Laporte (2003) para a versão estática do problema. Eles utilizaram uma estratégia de paralelização para se adaptar ao dinamismo. Seus experimentos mostraram que sua abordagem foi capaz de atender uma alta porcentagem das requisições.

Guerriero *et al.* (2014), em uma abordagem bi-objetiva, buscaram otimizar a distância total percorrida pelos veículos e o tempo total de espera dos usuários. Eles utilizaram uma abordagem em duas etapas, sendo que na primeira eles definiram um conjunto de rotas viáveis utilizando dois métodos meta-heurísticos, a Busca Tabu e o SA. Na segunda etapa, eles utilizaram esse conjunto de rotas em um modelo bi-objetivo do *Set Partitioning* em conjunto com o *Epsilon-Constraint framework* para gerar soluções eficientes. Eles realizaram experimentos em instâncias da literatura para avaliar o desempenho da abordagem.

Schilde *et al.* (2014) propuseram dois pares de modelos meta-heurísticos para solução do DDARP, ambos utilizando um método determinístico e sua versão estocástica correspondente. Os modelos utilizadas foram o VNS, Pesquisa em Vizinhança Variável Estocástica (S-VNS, *Stocastic Variable Neighborhood Search*), Abordagem de Planos Múltiplos (MPA, *Multiple Plan Approach*) e Abordagem de Cenário Múltiplo (MSA, *Multiple Scenario Approach*). Eles executaram seus experimentos em um conjunto de instâncias disponíveis na literatura e concluíram que o S-VNS funcionou melhor para instâncias em que o dinamismo varia entre 45% e 90%. Para os casos puramente estáticos ou muito dinâmicos o VNS determinístico se mostrou mais eficiente.

Os trabalhos que utilizam dados reais, em sua maioria, abordam o transporte de pacientes da área de saúde ou transporte de idosos. Vários países do mundo vêm investindo em pesquisas para melhorar esse tipo de serviço. Em Copenhagen, na Dinamarca, Madsen *et al.* (1995) trabalharam em conjunto com o serviço de corpo de bombeiros da cidade (CFFS, *Copenhagen Fire-Fighting Service*). Caracterizado como um DARP heterogêneo e multiobjetivo, eles tinham objetivo de reduzir os custos e o tempo de viagem dos pacientes. Eles modificaram o algoritmo de inserção originalmente criado por Jaw *et al.* (1986) chamado REBUS, e conseguiram de uma maneira flexível equilibrar os objetivos e obter bons resultados.

Na Alemanha, Beaudry *et al.* (2010) propuseram um método heurístico de duas fases para resolver o DARP. Eles fizeram uma proposta de adição de restrição de grau de urgência em algumas requisições e conflito para compartilhamento de mesmo veículo entre pacientes. Utilizando dados fornecidos por hospitais de larga escala da Alemanha para fazer seus experimentos, eles chegaram a soluções de boa qualidade e o algoritmo se mostrou hábil para tratar o dinamismo do problema.

Na Austrália, Schilde *et al.* (2011) modelaram o DARP com expectativa de retorno das requisições. Assim, quando uma requisição era feita, eles já trabalhavam com a possibilidade de atender o mesmo usuário em uma eventual volta ao local de origem. Eles testaram diferentes versões do MPA, MSA e versões dinâmicas do VNS e do S-VNS. Seus experimentos foram realizados em 12 conjuntos de instâncias com base em uma rede rodoviária real e os resultados mostraram que as informações sobre o retorno das requisições elevaram as melhorias médias das soluções em cerca de 15%.

Zhang *et al.* (2015) modelaram o DARP com múltiplas viagens baseando-se em dados fornecidos pelos hospitais da cidade de Hong Kong, que realizavam este transporte utilizando ambulâncias. Eles construíram um modelo matemático e utilizaram um algoritmo memético com operador customizado para resolver o problema. Os experimentos foram realizados utilizando dados reais do ano de 2009 e comparados com os resultados obtidos pela solução do modelo matemático. Eles também fizeram uma adaptação no algoritmo e testaram nas instâncias clássicas do DARP e chegaram a um bom desempenho em instâncias de média escala.

No município de Philippi no norte da Grécia, Lois & Ziliaskopoulos (2017) apresentaram um algoritmo *Regret Based*, que se baseava na utilização de dois subalgoritmos separados para analisar pedidos de viagem recebidos e melhorar continuamente a solução. Seu processo de otimização não é baseado em janelas de tempo constantes e sim contínuas, com intuito de melhorar a solução, explorando até mesmo pequenos fragmentos de tempo ociosos entre pedidos de viagem recebidos. Um algoritmo heurístico foi utilizado para construir a solução inicial a partir da primeira entrada de informações. A cada requisição uma análise é feita para verificar a viabilidade da inserção na rota, se não for viável, um custo de inserção é definido como alto para aquele veículo e é passado para próxima requisição. O algoritmo proposto apresentou resultados interessantes em termos de melhorias no custo das soluções. Nas instâncias maiores, em que os pontos de viagem são muito distintos, o sistema é mais indicado no aspecto rentabilidade em relação ao custo das soluções.

Novamente na Alemanha, Hanne *et al.* (2018) abordaram o problema utilizando restrições específicas do transporte utilizado entre hospitais. Eles projetaram um sistema chamado Opti-TRANS para auxiliar no suporte do transporte de pacientes. Os autores utilizaram o sistema em um dia de trabalho de um grande hospital alemão e o sistema apresentou muitos benefícios, incluindo processos de transporte simplificados e melhoria no fluxo de trabalho.

De modo geral, há vários outros exemplos de aplicações que abordam o DARP. Muitas delas podem ser encontrados em duas revisões sobre o tema, uma feita por Cordeau & Laporte (2007) e outra mais recente por Ho *et al.* (2018).

Capítulo 3

Descrição do Problema

A Secretaria de transportes de Ouro Preto – STOP da Prefeitura Municipal de Ouro Preto – PMOP, leva pacientes diariamente do Sistema Único de Saúde (SUS) para a região metropolitana da cidade de Belo Horizonte para tratamentos, consultas, exames médicos, entre outros serviços relacionados à saúde. Analisando os dados do fornecidos pela STOP, a fim de melhorar as condições dos pacientes e diminuir os gastos com as viagens, nota-se que este problema pode ser classificado como uma versão do DARP, que por sua vez é uma variante do VRP.

O DARP é uma especificação dos VRPs, que consiste no planejamento de rotas que atendam a um conjunto de indivíduos utilizando uma frota de veículos com algumas restrições. No DARP, os usuários fazem requisições para viagens indicando origem e destino. Normalmente são feitas duas requisições, uma para ida até um determinado local e outra para a volta. Todas as requisições são atendidas por uma frota de veículos com certas particularidades dependendo das especificações do problema (Cordeau & Laporte, 2003). No caso da PMOP, em relação aos veículos, o problema pode ser classificado como heterogêneo, pois a frota possui diferentes veículos, no que diz respeito à capacidade. Além disso, todos os veículos da frota são armazenados no mesmo local, ou seja, uma única garagem, sendo que os veículos devem iniciar e finalizar o trajeto nesta garagem.

A resolução do problema envolve encontrar um conjunto de rotas de custo mínimo, que atenda às restrições dos veículos e dos usuários. São consideradas restrições de veículo, o tempo de duração da rota, capacidade dos veículos e quantidade de veículos disponíveis. São considerados restrições de usuário: tempo de permanência no veículo, janelas de tempo e locais de coleta e entrega (Cordeau & Laporte, 2003).

Considerando o problema estudado na STOP, é apresentado na Seção 3.1 o atual funcionamento do serviço prestado, assim como os recursos disponíveis para realizá-lo e na Seção 3.2 a modelagem proposta para o problema.

3.1 Funcionamento atual

Atualmente, na Secretaria Municipal de Ouro Preto, o funcionamento do serviço prestado para realizar as viagens do setor de saúde é dado abaixo.

3.1.1 Veículos

A PMOP disponibiliza cinco tipos diferentes de veículos, no que diz respeito à capacidade dos mesmos. Alguns dos veículos disponíveis são terceirizados, mesmo assim todos os veículos tem ponto de partida na garagem da PMOP. O limite de veículos a serem utilizados está ligado a quantidade disponível para viagem, assim, não há uma cota diária de veículos a serem utilizados. Dos veículos terceirizados uma pequena parte é destinada para a utilização do pessoal da administração da STOP e não diretamente a locomoção de pacientes. Abaixo são descritos os tipos de veículos.

1. Carros Administrativos: são divididos em dois tipos em relação a capacidade, carros que comportam cinco pessoas e carros com capacidade para seis pessoas.

2. Ambulâncias: possuem tamanhos diferentes, mas ambas com mesma capacidade de levar um paciente e até dois acompanhantes, diferem apenas nos tipos de equipamentos que as compõem.

3. Micro-ônibus: possui capacidade para 26 pessoas.

4. Vans: possuem capacidade para 16 pessoas.

Os horários de saída dos veículos são agendados de acordo com os pacientes nele alocados para a viagem. A saída é calculada de acordo com o tempo médio até o destino do usuário que tem que ser entregue primeiro. Os veículos com maior capacidade, como micro-ônibus e algumas vans, têm horário fixo de partida. O micro-ônibus sai diariamente às *04:15 horas* e às *09:00 horas* e as vans saem às *06:00 horas* e às *11:00 horas*.

3.1.2 Agendamentos (Requisições)

Todos os agendamentos são feitos com antecedência, com exceção de emergência médica, como por exemplo acidentes, esses podem ser feitos para o mesmo dia. Em cada agendamento, são cadastrados os dados dos pacientes e possíveis acompanhantes, juntamente com datas, horários e local dos procedimentos a serem realizados no destino. Estes agendamentos podem ser feitos em cinco locais diferentes: na PMOP, na STOP, pelo pessoal do Tratamento Fora Domicílio (TFD), nas sedes do Programa de Saúde na Família (PSF) e pela Atenção Primária, sendo este último

restrito a agendamento de mamografias. Essas requisições são classificados como demanda espontânea, ou seja, agendamentos que acontecem quando o paciente vai até um destes pontos para fazer a requisição.

Entre os agendamentos temos diferentes tipos de pacientes, sendo que para cada tipo de veículo há um tipo de paciente que tem prioridade em sua utilização. Por exemplo, para as ambulâncias têm prioridade os pacientes com maior dificuldade de locomoção, que necessitam ficar deitados durante o transporte ou precisam da aplicação de medicamentos durante o trajeto, como por exemplo medicamentos intravenosos. Para os carros administrativos, tem prioridade pacientes com dificuldade de locomoção e acesso, pacientes em estado grave, esses são pegos em casa pelos veículos. E para os micro-ônibus e vans são agendados pacientes com consultas rotineiras e procedimentos simples.

3.1.3 Viagens

As viagens são feitas para as cidades de Belo Horizonte-MG, Itabirito-MG, Ouro Branco-MG, Ponte Nova-MG, Mariana-MG, Contagem-MG, Betim-MG, e cidades próximas à região de Ouro Preto-MG. Estas viagens são restritas a assuntos relacionados a saúde, como consultas médicas, exames, retornos médicos, cirurgias, etc.

Os trajetos feitos pelos veículos são definidos no dia anterior à viagem, analisando todas as requisições para o dia da viagem. Emergências são encaixadas em viagens com lugares em veículos ou com a utilização de outro veículo disponível. Os locais de coleta de cada paciente variam de acordo com o veículo no qual ela irá ser transportado. Os pacientes que estiverem agendados para ir na ambulância e nos carros administrativos serão pegos em sua residência e entregues na mesma após retorno da viagem feita. Os pacientes transportados no micro-ônibus e nas vans, serão pegos em pontos específicos, sendo três deles localizados na cidade de Ouro Preto e quatro em distritos pertencentes à cidade e entregues nestes mesmos pontos após o retorno da viagem.

3.1.4 Problemas identificados

Alguns problemas identificados na PMOP, como por exemplo, casos em que o paciente não irá comparecer à consulta e não avisa ao setor responsável pela alocação dos veículos, devido a natureza do problema, os algoritmos não conseguem resolver. Porém, problemas que são muito recorrentes, como os casos que pacientes com horário que possuem intervalos muito grandes estão no mesmo veículo, acarretando assim uma grande espera por parte dos motoristas e dos outros pacientes para o retorno a cidade.

3.2 Modelagem proposta para o problema

Neste trabalho são consideradas apenas as requisições agendadas para serem servidas, ou seja, o problema será tratado como DARP estático. A frota de veículos é caracterizada como heterogênea, dada a disponibilidade de veículos com diferentes capacidades. A garagem é única, todos os veículos têm ponto de partida e de chegada no mesmo local (Garagem de Veículos da PMOP). As restrições de janela de tempo foram definidas por meio dos horários das consultas. Os pontos de coleta individual são as casas dos pacientes e os pontos de coleta coletiva são definidos pela PMOP. Esses locais são bem conhecidos pela população e de fácil acesso. Para as entregas, são considerados clínicas médicas, hospitais públicos e particulares e consultórios de tratamento de saúde.

3.2.1 Modelo Matemático

O modelo é representado a partir de um grafo completo $G = (V, E)$, sendo V o conjunto de vértices e E o conjunto de arcos. O conjunto de vértices V é dividido em $\{\{0, 2n+1\}, P, D\}$, em que 0 e $2n+1$ representam o mesmo depósito e $P = \{1, \cdots, n\}$ e $D = \{n+1, \cdots, 2n\}$ são, respectivamente, os subconjuntos de coleta e entrega. O conjunto de arcos que ligam os vértices é denotado por $E = \{1, \cdots, m\}$. Para cada arco (i, j) é associado um custo de roteamento c_{ij} e um tempo de viagem t_{ij}.

Cada ponto de coleta i possui um par equivalente à entrega $n + i$, em que $i \in P$ e $n + i \in D$. O tempo máximo de viagem estabelecido para cada usuário não pode exceder L. Para cada vértice de coleta $i \in V$ é atribuída uma carga q_i, sendo $q_0 = q_{2n+1} = 0$, $q_i \geq 0 \ \forall i \in P$ e $q_i = -q_{i+n} \ \forall i \in D$ e um tempo de serviço τ_i, sendo $\tau_i \geq 0$. É também associada uma janela de tempo $[e_i, l_i] \ \forall i \in V$, em que e_i e l_i são inteiros não negativos.

A frota de veículos é representada por K, para cada $k \in K$ uma capacidade máxima Q_k e um tempo de viagem máximo da rota T_k são atribuídos.

Para a representação do modelo temos as seguintes variáveis de decisão:

- x_{ij}^k: uma variável binária indicando se o veículo k viajou do vértice i para o vértice j, 1 se viajou e 0 caso contrário.
- B_i^k: o horário em que o veículo k inicia o serviço no vértice i, 0 caso o veículo não sirva o vértice.
- Q_i^k: a quantidade de pessoas no veículo k depois de visitar o vértice i, 0 caso o veículo não sirva o vértice.
- L_i^k: Tempo de viagem do paciente i no veículo k, 0 caso o veículo não sirva este paciente.

Função Objetivo

$$\min \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} x_{ij}^k c_{ij} \tag{3.1}$$

Sujeito a:

$$\sum_{k \in K} \sum_{j \in V} x_{ij}^k = 1, \qquad \forall i \in P \tag{3.2}$$

$$\sum_{j \in (P \cup D)} x_{0j}^k = 1, \qquad \forall k \in K \tag{3.3}$$

$$\sum_{i \in (P \cup D)} x_{i,2n+1}^k = 1, \qquad \forall k \in K \tag{3.4}$$

$$\sum_{j \in V} x_{ij}^k - \sum_{j \in V} x_{j,n+i}^k = 0, \qquad \forall i \in P, \forall k \in K \tag{3.5}$$

$$\sum_{i \in V} x_{il}^k - \sum_{j \in V} x_{lj}^k = 0, \qquad \forall l \in V, \forall k \in K \tag{3.6}$$

$$B_j^k \geq B_i^k + \tau_i + t_{ij} - M1 \left(1 - x_{ij}^k\right), \qquad \forall i \in V, \forall j \in V, \forall k \in K \tag{3.7}$$

$$Q_j^k \geq Q_i^k + q_i - M2 \left(1 - x_{ij}^k\right), \qquad \forall i \in V, \forall j \in V, \forall k \in K \tag{3.8}$$

$$L_i^k = B_{n+i}^k - \left(B_i^k + \tau_i\right), \qquad \forall i \in V, \forall k \in K \tag{3.9}$$

$$B_{2n+1}^k - B_0^k \leq T_k, \qquad \forall k \in K \tag{3.10}$$

$$e_i \leq B_i^k \leq l_i, \qquad \forall i \in V, k \in K \tag{3.11}$$

$$t_{i,n+i} \leq L_i^k \leq L, \qquad \forall i \in P, k \in K \tag{3.12}$$

$$\max\{0, q_i\} \leq Q_i^k \leq \min\{Q_k, Q_k + q_i\}, \qquad \forall i \in V, k \in K \tag{3.13}$$

$$x_{ij}^k \in \{0, 1\}, \qquad \forall i \in V, j \in V, k \in K \tag{3.14}$$

Neste modelo a função objetivo (3.1) busca minimizar os custos das viagens feitas; as restrições (3.2), (3.3), (3.4) garantem que todos os pacientes sejam atendidos apenas uma vez e que todas as rotas se iniciem e terminem na garagem. As restrições (3.5) e (3.6) garantem que o mesmo veículo atenda o vértice de coleta e entrega do paciente e o fluxo de coleta e entrega do mesmo. A restrição (3.7) define o início do serviço em cada vértice e a restrição (3.8) calcula a quantidade de pessoas no veículo no momento em que passa pelo vértice i, sendo $M1 \geq \max\{0, l_i + \tau_i + t_{ij} - e_j\}$ e $M2 \geq \min\{Q_k, Q_k + q_i\}$. A restrição (3.9) calcula a tempo de permanência de cada paciente i no veículo k. A restrição (3.10) garante que cada veículo k respeite o limite de tempo T_k definido para o veículo k. As restrições (3.11) e (3.12) garantem que o início do serviço respeitará as janelas de tempo do paciente i e o tempo máximo de permanência do paciente i no veículo k não ultrapasse o limite L. A restrição (3.13) garante que a capacidade Q_k do veículo não será violada e por fim a restrição (3.14) garante que o domínio da variável x_{ij}^k seja binário.

Capítulo 4

Algoritmos Propostos

Este capítulo está organizado como segue. Na Seção 4.1.1 é descrito o algoritmo MS-VNS1 proposto. Na Seção 4.1.2, por sua vez, é apresentado o algoritmo VNS2 proposto. A Seção 4.2 descreve o modelo de Cobertura de conjuntos utilizado como método de busca local desses algoritmos. Na Seção 4.3 é mostrado como uma solução é representada e na Seção 4.4 como ela é avaliada. Na Seção 4.5 é mostrado o procedimento de construção da solução inicial. A descrição das vizinhanças utilizadas é apresentado na Seção 4.6. Os procedimentos de *shaking* denominados *shaking1* e *shaking2* são descritos na Seção 4.7. Por fim, na Seção 4.8 é apresentado o método de busca local.

4.1 Algoritmos MS-VNS1 e VNS2

Para resolver o DARP, foram propostos dois algoritmos meta-heurísticos, MS-VNS1 e VNS2, ambos baseados na meta-heurística *Variable Neighborhood Search* (Hansen & Mladenović, 2001). O primeiro, MS-VNS1, é guiado pela meta-heurística *Multi-Start* tendo como busca local o VNS. O segundo, por sua vez, é guiado apenas pelo VNS. Nos dois algoritmos o método de busca local do VNS é o procedimento heurístico *Randomized Variable Neighborhood Descent* (RVND), o qual usa os movimentos de realocação, troca e cruzamento para explorar o espaço de soluções do problema.

Para entender o funcionamento da meta-heurística VNS, três componentes devem ser especificados:

- Procedimento de geração de uma solução inicial, apresentado na Seção 4.5.
- Procedimento de Busca Local, descrito na Seção 4.8. Para o procedimento de busca local do algoritmo MS-VNS1 foram utilizadas as vizinhanças Realocação(1), Troca(1), Troca(2) e Cruzamento e para o algoritmo VNS2 foram

utilizadas apenas as vizinhanças Troca(1), Cruzamento e Troca(2). Todas essas vizinhanças estão descritas na Seção 4.6.

- Procedimento *Shaking*. Esse procedimento é apresentado na Seção 4.7.

4.1.1 Algoritmo MS-VNS1

O Algoritmo 1 ilustra o pseudocódigo do método VNS1, que é apresentado como busca local do algoritmo MS-VNS1. O algoritmo recebe por parâmetro uma solução s, um conjunto de rotas R e o numero máximo de iterações $iterMax$. Nas linhas 4 e 7 é aplicado na solução corrente o procedimento RVND, que realiza a busca local da solução corrente. Na linha 6 é executado o *shaking1* sobre essa solução. Na linha 16 é aplicado um procedimento nomeado *coberturaConjuntos*. Nesse procedimento, descrito em detalhes na Seção 4.2, o conjunto de rotas viáveis R é passado por parâmetro para o método, e ele funciona como um método exato para esse conjunto de rotas. Todos os módulos do algoritmo proposto são apresentados em detalhes nas seções seguintes.

Algoritmo 1: VNS1(s, R, iterMax)

```
1  iter ← 0;
2  iterSC ← 0;
3  V = {Realocacao(1), Troca(1), Troca(2), Cruzamento}
4  s ← RVND(s_0, V);
5  enquanto iter ≤ iterMax faça
6      s' ← shaking1(s);
7      s" ← RVND(s', V);
8      se f(s") < f(s) então
9          s ← s";
10         iter ← 0;
11     senão
12         iter ← iter + 1;
13     fim
14     iterSC ← iterSC + 1;
15     se iterSC ≥ iterMax/3 então
16         s" ← coberturaConjuntos(R);
17         se f(s") < f(s) então
18             s ← s";
19             iter ← 0;
20         fim
21         iterSC ← 0;
22     fim
23 fim
24 retorna s;
```

Para melhorar as soluções encontradas durante a execução do método VNS1 proposto, rotas viáveis são armazenadas, formando um conjunto de rotas. Um

algoritmo de Cobertura de Conjuntos é aplicado utilizando esse conjunto de rotas. O modelo matemático do algoritmo de Cobertura de Conjuntos é apresentado na Seção 4.2. Esse algoritmo é executado cada vez que as execuções chegam a 1/3 do número máximo de iterações sem melhora.

Para um diversificação de rotas a serem armazenadas para a cobertura de conjuntos, o algoritmo VNS1 é guiado pelo meta-heurística MS.

O método MS tem duas fases a cada iteração. Na primeira, uma solução inicial é gerada e na segunda, um processo de melhoria é aplicado a essa solução. Assim, a cada iteração, é gerada uma solução ótima local. Essas duas fases são aplicadas durante um número pré-definido de vezes e a melhor solução encontrada é retornada pelo método (Martí, 2003). O Algoritmo 2 mostra o funcionamento do algoritmo MS-VNS1 proposto. O algoritmo recebe por parâmetro o número máximo de iterações do MS e do procedimento VNS1. Na linha 5 é gerada a solução inicial s, essa que é passada por parâmetro para o método VNS1, aplicado na linha 6. Para este trabalho o procedimento de gerar uma solução inicial é apresentado na Seção 4.5 e como método de busca local é utilizado o algoritmo VNS1.

Algoritmo 2: Multi-Start(iterMax, IterMaxVNS)

```
1  iter ← 0;
2  R ← ∅;
3  s ← ∅;
4  enquanto iter ≤ iterMax faça
5      s ← GeraSolucaoInicial();
6      s' ← VNS1(s, R, iterMaxVNS);
7      se vazio(s) ou f(s) < f(s') então
8          s ← s';
9          iter ← 0;
10     senão
11         iter ← iter + 1;
12     fim
13 fim
14 retorna s;
```

4.1.2 Algoritmo VNS2

O Algoritmo 3 exibe o pseudocódigo do VNS2 desenvolvido para solucionar o DARP. O algoritmo recebe por parâmetro uma solução s, um conjunto de rotas R e o numero máximo de iterações $iterMax$. Na linha 5 é executado o *shaking2* na solução corrente. Na linha 6 o RVND é aplicado na solução. Tal como no algoritmo VNS1, na busca local são armazenadas rotas viáveis para execução da cobertura de conjuntos. A cobertura de conjuntos é executada na linha 15.

Assim como no algoritmo VNS1, para melhorar as soluções encontradas durante a execução, o algoritmo de Cobertura de Conjuntos também é aplicado ao VNS2. No VNS2 ele é executado cada vez que as execuções atingem a metade do número máximo de iterações sem melhora.

Algoritmo 3: VNS2(s, R, iterMax)

```
1  iter ← 0;
2  iterSC ← 0;
3  V = {Troca(2), Troca(1), Cruzamento};
4  enquanto iter ≤ iterMax faça
5      s' ← shaking(s);
6      s" ← buscaLocal(s', V);
7      se f(s") < f(s) então
8          s ← s";
9          iter ← 0;
10     senão
11         iter ← iter + 1;
12     fim
13     iterSC ← iterSC + 1;
14     se iterSC ≥ iterMax/2 então
15         s" ← coberturaConjuntos(R);
16         se f(s") < f(s) então
17             s ← s";
18             iter ← 0;
19         fim
20         iterSC ← 0;
21     fim
22 fim
23 retorna s;
```

4.2 Cobertura de Conjuntos

O algoritmo de Cobertura de Conjuntos funciona como segue. Seja R um conjunto composto por diferentes rotas viáveis encontradas pelo método heurístico, V o conjunto de usuários e $|K|$ o número máximo de veículos disponíveis. Uma cobertura de V é uma combinação J de rotas $j \in R$. Seja c_j o custo de cada rota j e uma constante ρ_{ij} que informa se o usuário i é atendido pela rota j. O Problema de Cobertura de Conjunto consiste em encontrar uma combinação $J \subseteq R$ de forma que o custo total seja minimizado. Após toda aplicação da cobertura de conjuntos nos algoritmos propostos, uma avaliação da solução retornada é realizada para garantir a viabilidade.

Para apresentar o modelo matemático para a Cobertura de Conjuntos, seja y_j uma variável binária associada a uma rota $j \in J$. A seguir o modelo usado em nosso

algoritmo.

$$\min \sum_{j \in R} c_j y_j$$

Sujeito a:

$$\sum_{j \in R} \rho_{ij} y_j \geq 1, \qquad \forall i \in V \tag{4.1}$$

$$\sum_{j \in R} y_j \leq |K|, \tag{4.2}$$

$$y_j \in \{0, 1\} \tag{4.3}$$

4.3 Representação da Solução

A representação da solução é dada por uma matriz de $|k|$ linhas, em que cada linha representa a rota de um veículo da frota. Cada rota se inicia no depósito representado por 0, seguido de uma lista de coletas e entregas, finalizando no mesmo depósito. Os pontos de coleta e entrega são inseridos satisfazendo as restrições do problema. A Tabela 4.1 representa uma solução correspondente a 5 veículos $(A, B, \cdots, E)$ e 13 requisições, sendo elas $(1+ \ \cdots \ 13+)$ representando as coletas e $(1- \ \cdots \ 13-)$ representando as entregas.

Tabela 4.1: Representação da Solução

A	0	9+	4+	9-	2+	2-	4-	6+	6-	0
B	0	3+	5+	3-	1+	1-	5-	0		
C	0	7+	8+	7-	8-	0				
D	0	10+	12+	13+	10-	12-	13-	0		
E	0	11+	11-	0						

4.4 Avaliação da Solução

As soluções são avaliadas pelo método descrito em Cordeau & Laporte (2003) e Parragh *et al.* (2010). Seja uma rota $k = (v_0, \cdots, v_i, \cdots, v_q)$, na qual v_0 e v_q representam a garagem. Para um melhor aproveitamento do tempo de viagem de cada usuário é calculado o tempo de folga F_i em cada vértice. D_i representa o momento em que o veículo sai do vértice i; A_i o momento em que o veículo chega ao vértice i; B_i o início do serviço no vértice i e W_i o tempo de espera no vértice $i \ \forall i \in V$. Cada um desses componentes é calculado da seguinte maneira:

- $A_i = D_{i-1} + t_{i-1,i}$: o tempo de chegada em cada vértice é calculado somando-se o tempo de saída do vértice anterior ao tempo de deslocamento compreendido entre os vértices.

- $B_i = \max\{e_i, A_i\}$: o início do serviço em cada vértice é dado pelo valor máximo entre a janela de tempo inicial do vértice e o tempo de chegada nele.

- $D_i = B_i + \tau_i$: o tempo de saída é dado pela soma do tempo de início do serviço no vértice com o tempo de serviço do usuário.

- $W_i = B_i - A_i$: o tempo de espera é calculado pela subtração do tempo de início do serviço pelo tempo de chegada no vértice.

- $F_i = \min\limits_{i \leq j \leq q} \left\{ \sum\limits_{i < p \leq j} W_p + (\min\{l_j - Bj, L - P_j\})^+ \right\}$: O tempo de folga é dado pelo menor valor das folgas entre os vértices i e q, sendo q o ultimo vértice da rota, desconsiderando a garagem. Assume-se que $x^+ = \max\{0, x\}$. Essa folga é calculada pela soma de todas as esperas dos vértices entre i e o j corrente somado ao menor valor entre a subtração do fim da janela de tempo de j, isto é, l_j, pelo tempo do inicio do serviço em j, isto é, e_j, e a subtração do tempo máximo de viagem do usuário por P_j, em que P_j representa o tempo de viagem do usuário em que a entrega é j e 0 para os outros casos.

A avaliação ocorre na seguinte maneira:

1. É feita a atribuição: $D_0 = e_0$, sendo e_0 o início da janela de tempo.

2. São calculados os valores de A_i, W_i, B_i, e D_i para todos os vértices da rota.

3. Calcula-se F_0.

4. É feita a atribuição: $D_0 = e_0 + \min \left\{ F_0, \sum\limits_{0 < i < q} W_i \right\}$.

5. São atualizados os valores de A_i, W_i, B_i, e D_i para todos os vértices da rota.

6. É calculado o valor L_i para cada requisição da rota.
   Se nenhum usuário violar a restrição de permanência no veículo L, o passo 7 não é executado.

7. Para todo vértice j que é um vértice de coleta são feitas as seguintes operações:

   - Calcule F_j.

- Faça as seguintes atribuições:
  $W_j = W_j + \min\left\{F_j, \sum_{j<i<q} W_i\right\}$, $B_j = A_j + W_j$ e $D_j = B_j + \tau_j$.
- Atualize os valores de A_i, W_i, B_i, e D_i para todos os vértices que estão depois do vértice j.
- Atualize L_i para todos os vértices de entrega depois do vértice j.

8. Calcule as violações de janela de tempo, tempo de viagem do usuário, tempo de viagem do veículo e capacidade dos veículos.

As violações são calculadas e adicionadas ao valor da função objetivo. Para uma solução s, o valor de função de avaliação é calculado da seguinte maneira: $f(s) = c(s) + \alpha w(s) + \beta t(s) + \gamma d(s) + \delta q(s)$, de maneira que $c(s)$ representa o custo da solução e α, β, γ e δ representam as penalizações para janela de tempo, tempo de permanência do usuário no veículo, duração da rota e capacidade dos veículos, respectivamente. Cada violação é calculada da seguinte forma. Para a violação de janelas de tempo: $w(s) = \sum_{i \in V} (B_i - l_i)^+$, para o tempo de permanência de usuário no veículo: $t(s) = \sum_{i \in P} (L_i - L)^+$, para duração da rota: $d(s) = \sum_{k \in K} (B^k_{2n+1} - B^k_0 - T_k)^+$ e para capacidade dos veículos: $q(s) = \sum_{i \in V} \sum_{k \in K} (y^k_i - Q_k)^+$, sendo que y_i representa a quantidade de usuários no veículo k no vértice i.

4.5 Geração da Solução Inicial

A solução inicial é gerada por um procedimento, nomeado SolucaoInicial, que a cada passo seleciona de forma gulosa as requisições. Considerando um conjunto R com todas as requisições do problema, o procedimento de geração de uma solução inicial funciona como segue. Primeiramente é escolhido um veículo i, selecionado de forma aleatória no conjunto $\{1, nveic\}$ de veículos disponíveis. Em seguida, a requisição $req1$ que possui menor valor e_{req1} com relação à janela de tempo é selecionada, juntamente com seu par equivalente $req2$. Ambas são inseridas na rota do veículo i e removidas do conjunto R. Essa ação é repetida até que o conjunto R esteja vazio. Atendidas todas as requisições, o procedimento passa, então, a incluir todas as rotas dos veículos no conjunto solução s. O Algoritmo 4 apresenta o procedimento de construção da solução.

Algoritmo 4: SolucaoInicial(nveic, R)

enquanto $R \neq \emptyset$ **faça**
    $i \leftarrow$ veículo aleatório $\in \{1, nveic\}$;
    $req1 \leftarrow$ requisição com menor valor e_{req1} com relação à janela de tempo;
    $req2 \leftarrow$ par de **req1**;
    $RotaVeiculo_i \leftarrow$ **req1** e **req2**;
    remova **req1, req2** de **R**;
fim
$s \leftarrow \emptyset$;
para $i = 1$ **até** $nveic$ **faça**
    $s \leftarrow s \cup \{RotaVeiculo_i\}$;
fim
retorna s;

4.6 Estruturas de Vizinhança

Nove estruturas de vizinhanças diferentes foram criadas para explorar o espaço de soluções do problema, sendo seis delas inter-rota e três intrarrota. O algoritmo MS-VNS1, em sua busca local, utiliza quatro vizinhanças inter-rota diferentes para compor a lista de vizinhanças. O Algoritmo VNS2, por sua vez, utiliza apenas três. Para ambos os algoritmos, a cada melhoria alcançada por um movimento inter-rota, movimentos intrarrota são aplicados às rotas modificadas.

4.6.1 Vizinhanças Inter-rota

Nas estruturas de vizinhança inter-rota, são aplicados movimentos entre rotas distintas da solução na tentativa de melhora. Para cada uma das figuras exemplificadas a seguir, duas rotas r_1 e r_2 são selecionados randomicamente e depois aplicados os movimentos nas vizinhanças descritas. Nas figuras, a rota r_1 está destacada em azul e a rota r_2 na cor preta preta. O algoritmo MS-VNS1 em sua busca local utiliza as vizinhanças de Realocação(1), Troca(1), Cruzamento e Troca(2). O método VNS2, por sua vez, utiliza as vizinhanças Troca(1), Cruzamento e Troca(2).

- Realocação(1): Uma requisição (coleta e entrega) é removida da rota r_1 e inserida na rota r_2. Na Figura 4.1, que ilustra esse movimento, a requisição (1+, 1−) é realocado de r_1 para r_2.

- Troca(1): Uma requisição é selecionado em cada rota e são trocados da rota r_1 para a rota r_2. A Figura 4.2 ilustra esse movimento. Nela, uma requisição (6+, 6−) é selecionado na rota r_1 e outra (4+, 4−) na rota r_2 e eles são trocados, coleta com coleta e entrega com entrega.

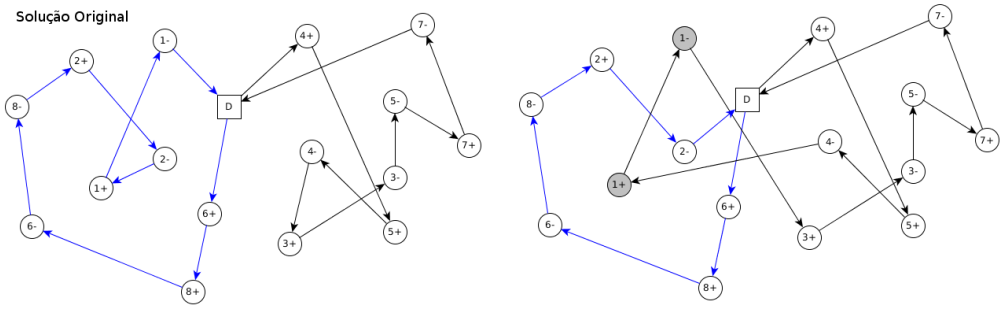

Figura 4.1: Realocação(1)

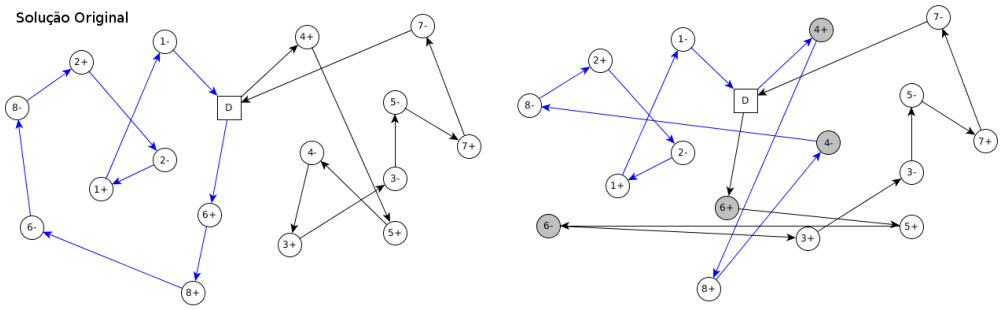

Figura 4.2: Troca(1)

- Cruzamento: Randomicamente dois pontos são selecionados, um em r_1 e outro em r_2. Todos os usuários de coleta (e suas respectivas entregas) que estão antes do ponto selecionado em R_1 são colocados na mesma rota que todos os usuários de coleta (e suas respectivas entregas) que estão depois do ponto selecionado em R_2. Da mesma maneira, são colocados em outra rota os usuários que estão depois do ponto selecionado em r_1 e antes do ponto em r_2. Na Figura 4.3, que ilustra essa operação, os pontos selecionados estão no vértice 2+ na rota r_1 e no vértice 3+ na rota r_2. A linha vermelha representa os pontos e o cruzamento das rotas.

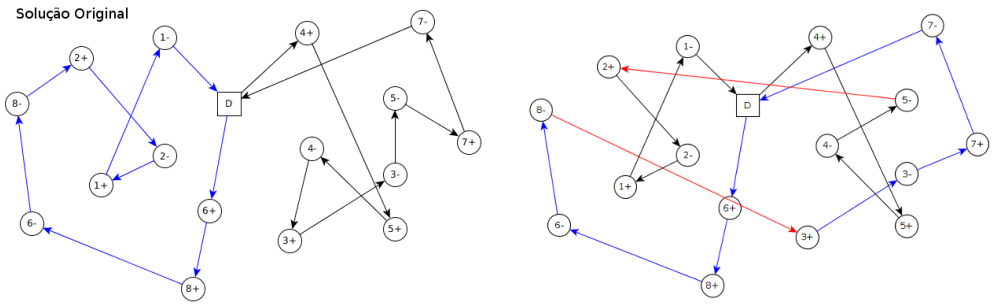

Figura 4.3: Cruzamento

- Troca(2): Duas requisições são selecionados, uma em r_1 e outra em r_2. Todas as possibilidades na ordem de troca são examinadas. A Figura 4.4 ilustra essa

operação. Nela, duas requisições (6+, 6−) e (8+, 8−) são selecionados nesta ordem na rota r_1 e outras duas (4+, 4−) e (3+, 3−) também nesta ordem na rota r_2 e elas são trocadas, coleta com coleta e entrega com entrega.

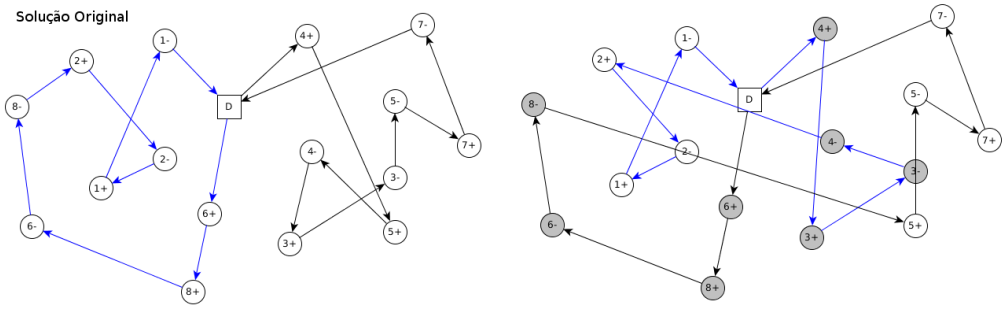

Figura 4.4: Troca(2)

- Troca(2,1): Três requisições são selecionadas, duas na rota r_1 e uma na rota r_2. Na Figura 4.5, que ilustra essa operação, duas requisições (6+, 6−) e (8+, 8−) são selecionadas na rota r_1 e outra (4+, 4−) na rota r_2 e elas são trocadas. Para essa vizinhança, também são avaliadas todas as possibilidades na ordem da troca.

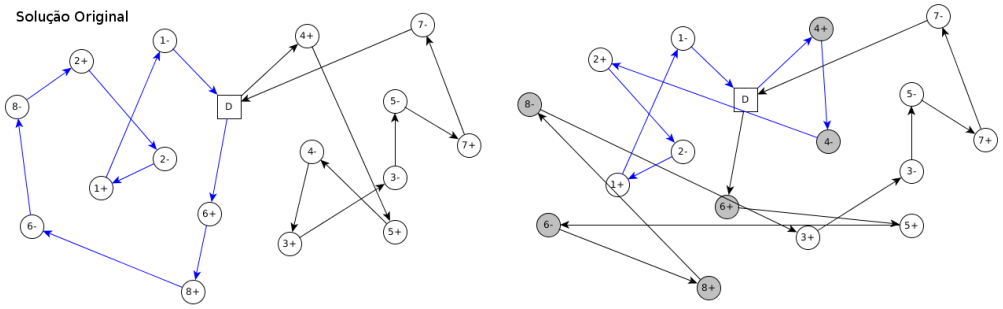

Figura 4.5: Troca(2,1)

- Realocação(2): Duas requisições são realocadas da rota r_1 para a rota r_2. Na Figura 4.6 as requisições (6+, 6−) e (8+, 8−) são realocadas de r_1 para r_2.

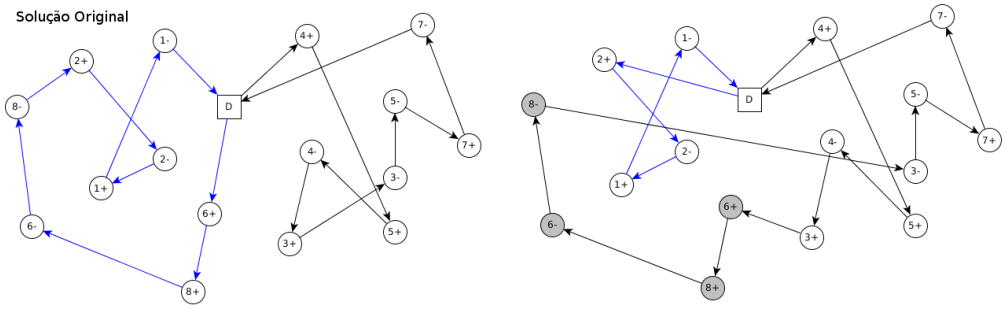

Figura 4.6: Realocação(2)

4.6.2 Vizinhanças Intrarrota

As estruturas de vizinhança intrarrota são executadas em uma única rota e estão descritas a seguir.

- Or-opt1: Um vértice é retirado de sua posição e realocado em outra posição na rota. A Figura 4.7 b) exemplifica o movimento.
- Or-opt2: Dois vértices em sequência são retirados de suas posições e realocados em outro ponto da rota. A Figura 4.7 c) mostra o movimento.
- *Swap*: Um vértice é removido de sua posição e permutado com outra vértice da rota. A Figura 4.7 d) exibe a troca.

Essas vizinhanças somente são aplicadas em rotas que sofreram alterações durante a execução da vizinhança inter-rota. É importante destacar que em todas essas vizinhanças somente são avaliadas trocas e realocações se a requisição de coleta não ultrapassar a de entrega, pois do contrário seria gerada uma solução inviável.

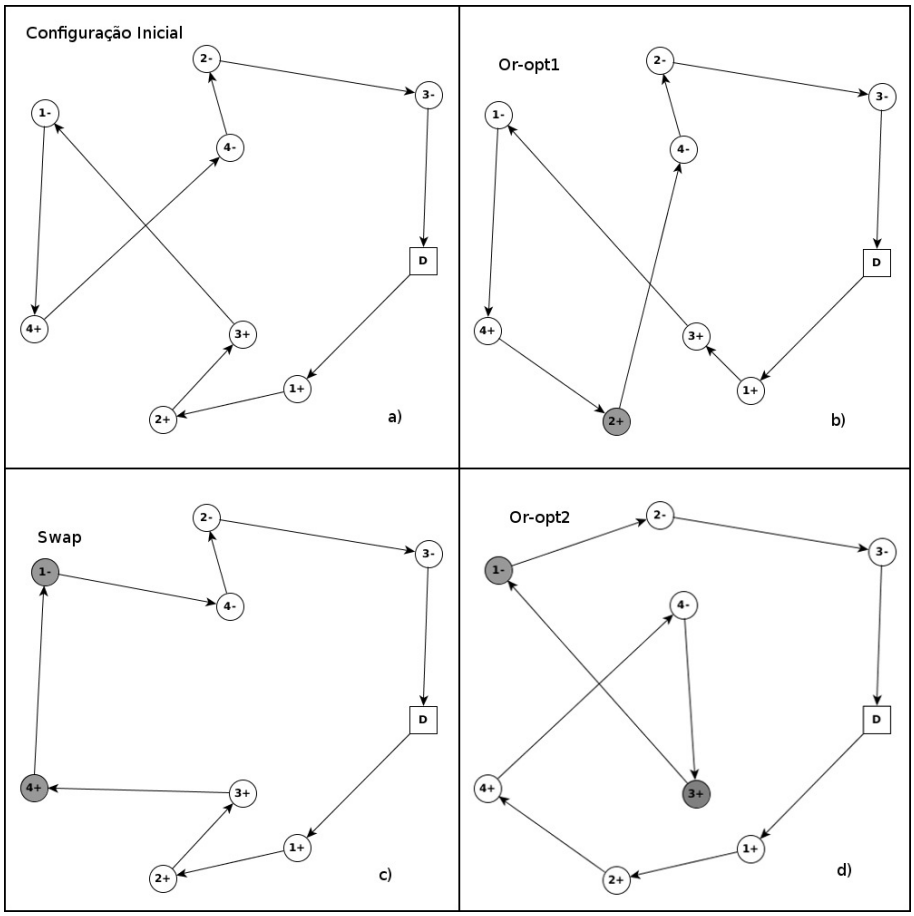

Figura 4.7: Estruturas de vizinhança intrarrota

4.7 Shaking

O procedimento de *shaking* é utilizado para que o algoritmo não fique preso em um ótimo local, dessa maneira, esse procedimento é aplicado de maneira forte o suficiente para que o algoritmo não retorne a mesma solução e ao mesmo tempo fraca o suficiente para evitar um reinício aleatório. Neste trabalho dois procedimentos *shaking1* e *shaking2* foram criados e estão descritos a seguir.

4.7.1 Shaking1

O procedimento *shaking1* é diretamente relacionado com o número de iterações sem melhora do algoritmo MS-VNS1. A cada iteração, um laço de tamanho k é executado, sendo k limitado ao valor máximo 5 de acordo com o número de iterações sem melhora, ou seja, quando o número de iterações sem melhora ultrapassa 5 unidades, o valor de k é mantido, para evitar uma aleatorização da nova solução gerada. O valor 5 foi selecionado após testes empíricos, nos quais valores superiores a 5 praticamente equivaliam a uma reinicialização da solução.

O *shaking1* é aplicado à melhor solução corrente. Três diferentes movimentos (Cruzamento, Troca e Realocação) formam o conjunto VP de vizinhanças e podem ser executados a soluções que possuam mais de três veículos. Para uma solução com apenas um veículo ou dois, apenas os movimentos de troca e realocação são aplicados. Isso se deve em função do modo de execução da vizinhança de cruzamento. Em ambos os casos todas as vizinhanças têm mesma probabilidade de serem selecionadas.

A seguir são descritas cada uma das vizinhanças utilizadas em VP. Nessa descrição são consideradas duas rotas r_1 e r_2 aleatoriamente selecionadas.

- Realocação(1): Um par aleatório de coleta e entrega é transferido de uma rota r_1 para uma outra rota r_2.
- Troca(1): É selecionado aleatoriamente um par de coleta e entrega em cada rota r_1 e r_2 e é feita uma permutação entre esses pares.
- Cruzamento: São selecionados randomicamente três rotas. A rota com maior quantidade de requisições é definida como r_M, e as outras duas são definidas como r_1 e r_2. Em seguida,, a rota r_M é dividida em duas partes, sendo cada parte armazenada em r_1 e r_2. As rotas r_1 e r_2 são unificadas e armazenadas em r_M.

O Algoritmo 5 mostra o o funcionamento do procedimento shaking1.

Algoritmo 5: Shaking1(s, k)

1 $VP = \{Realocacao(1), Troca(1), Cruzamento\}$
2 **para** $i = 1$ **até** k **faça**
3 $p \leftarrow$ vizinhança aleatória do conjunto VP;
4 $s' \leftarrow$ resultado da aplicação a s de um movimento aleatório na vizinhança $VP_p(s)$;
5 **fim**
6 **retorna** s;

4.7.2 Shaking2

O *shaking2* é aplicado ao algoritmo VNS2 e tem o seguinte funcionamento. Dada uma lista com as seis vizinhanças inter-rotas descritas anteriormente, a saber: Realocação(1), Realocação(2), Troca(1), Troca(2), Troca(2,1) e Cruzamento, uma delas é selecionada aleatoriamente. Em seguida, um movimento aleatório da vizinhança selecionada é aplicado à solução corrente, independentemente se há ou não melhora na solução. Essa nova solução é retornada pela função.

4.8 Busca Local

O método utilizado como busca local foi o *Ramdomized Variable Neigborhood Descent* (RVND).

Esse método funciona como segue. Uma vizinhança é selecionada aleatoriamente a partir da lista V de vizinhanças. Em seguida, é realizada uma descida na solução corrente usando-se a estratégia *First Improvement*, onde, ao encontrar uma solução de melhora, essa solução é retornada. Caso não haja melhora na solução, essa vizinhança escolhida é removida da lista e outra vizinhança é selecionada, também de forma aleatória, até que a lista esteja vazia. Caso haja alguma melhora depois da aplicação de alguma vizinhança, todas as vizinhanças são recolocadas na lista. Para essa lista são consideradas as vizinhanças inter-rota definidas para cada algoritmo.

O Algoritmo 6 apresenta o funcionamento do método RVND de busca local.

Algoritmo 6: RVND(s, V)

```
1  enquanto (V ≠ ∅) faça
2      k ← Vizinhança aleatória do conjunto V;
3      s' ← FirstImprovement(V_k(s));
4      se f(s') < f(s) então
5          s ← s';
6          reinicialize V
7      senão
8          remova V_k de V;
9      fim
10 fim
11 retorna s;
```

Capítulo 5

Experimentos Computacionais

Neste capitulo são apresentados o ambiente de desenvolvimento do algoritmo proposto e a configuração do equipamento usado para testá-lo (Seção 5.1), as instâncias utilizadas para testar os métodos e as instâncias criadas a partir do cenário da cidade de Ouro Preto (Seção 5.2), os experimentos para seleção das vizinhanças a serem utilizadas na exploração do espaço de busca do DARP (Seção 5.3) e, finalmente, na Seção 5.4, os resultados obtidos pela aplicação dos algoritmos MS-VNS1 e VNS2.

5.1 Ambiente de Desenvolvimento e Testes

Os algoritmos propostos foram implementados na linguagem C++ usando o compilador gcc 9.2.1 e testados em um computador com processador intel Core I7-7700 CPU @ 3.60GHz com 16GB de memória RAM e sistema operacional Linux Ubuntu 16.04 LTS.

5.2 Instâncias

Para testar os algoritmos desenvolvidos, foram utilizados diferentes tamanhos e grupos de instâncias. Para as instâncias pequenas foram usadas as instâncias criadas por Parragh (2011), estas contêm três grupos diferentes de instâncias (E, I e U), definidas com base nas instâncias geradas por Cordeau (2006) para a versão clássica do problema. A autora adicionou características de heterogeneidade para veículos e usuários nessas instâncias. Dessa maneira, neste trabalho foram criadas instâncias baseadas no contexto da cidade de Ouro Preto, através de uma análise feita em dados fornecidos pelo setor de transporte da PMOP. Estas instâncias foram criadas analisando dias operacionais da cidade. Todas são detalhadas a seguir.

5.2.1 Instâncias da Literatura

Os conjuntos de instâncias definidas por Parragh (2011) foram baseadas nas instâncias de Cordeau (2006). Estas por sua vez, possuem dois grupos de geradas aleatoriamente e com o número de requisições, variando de 16 a 48. Para ambos os grupos todos os pontos de coleta e entrega são gerados em um plano euclidiano $[-10, 10]$ x $[-10, 10]$ utilizando distribuição uniforme. A garagem é localizada no centro desse plano. Para cada aresta $(v_i, v_j) \in A$, o custo c_{ij} e o tempo t_{ij} são calculados como distância euclidiana entre os vértices. Todos os vértices possuem uma janela de tempo $[e_i, l_i]$ e essa janela é restrita em metade deles. As janelas são definidas da seguinte forma: para os vértices de coleta, e_i foi escolhido no intervalo $[0, T - 60]$, sendo T o tamanho do horizonte de planejamento, e l_i recebe o valor $e_i + 15$. Para os vértices de entrega, l_i é escolhido no intervalo $[60, T]$ e e_i recebe o valor $l_i - 15$.

Para o primeiro grupo de instâncias foi fixada a capacidade máxima $Q = 3$ para cada veículo, em cada ponto é coletado apenas um usuário $q_i = 1$, o tempo de serviço foi fixado em $\tau_i = 3$ e o tempo máximo de viagem $L = 30$ é estabelecido para cada usuário. Para o segundo grupo de instâncias, a capacidade máxima foi definida em $Q = 6$ e o número de usuários coletados q_i em cada ponto foi randomicamente gerado utilizando-se uma distribuição uniforme no conjunto $\{1, \cdots, Q\}$. O tempo máximo de permanência no veículo foi definido como $L = 45$ e para esse grupo foi estabelecido que o tempo de serviço é proporcional à quantidade de usuários coletados em cada ponto, assim, $\tau_i = q_i$.

Parragh (2011) adicionou heterogeneidade nos veículos e usuários, assim propôs três grupos de instâncias (E, I e U). Essa heterogeneidade foi baseada em dados obtidos da da Cruz Vermelha Austríaca (*Austrian Red Cross* - ARC). Cada grupo possui 12 instâncias, sendo criadas com quantidade de veículos variando de 2 a 4 e o número de requisições variando de 16 a 48. A heterogeneidade dos pacientes foi estabelecida utilizando quatro tipos diferentes, sendo eles, pacientes de maca, pacientes de cadeira de rodas (cadeirantes), pacientes simples e acompanhantes de paciente. Os lugares que cada tipo de paciente pode ocupar no veículo estão descritos na Tabela 5.1.

A heterogeneidade dos veículos é baseada na variação da capacidade dos veículos quanto a cada tipo de paciente. Nessas instâncias, todas as requisições possuem mais de uma pessoa a ser coletada por ponto. No grupo de instâncias "E", 50% dos usuários são pacientes simples, 25% são cadeirantes e 25% são pacientes de maca. Assume-se que 10% dos pacientes possuem acompanhante. Também foi estabelecido uma frota de veículos homogênea, sendo estes veículos classificados como T1, cada um possuindo dois lugares para acompanhantes, um lugar para cadeirante, um lugar

para paciente de maca e um para paciente simples. No grupo de instâncias "I", 83% das requisições são de pacientes simples, 11% são de cadeirantes e 6% são pacientes de maca. Assume-se que 50% dos pacientes possuem acompanhante. A frota de veículos é considerada como heterogênea, em que além dos veículos T1 possui também veículos do tipo T2, estes que possuem seis lugares para pacientes simples, um lugar para cadeirante, um para paciente de maca e um para acompanhante. No grupo de instâncias "U", apenas pacientes simples são considerados, sem ocorrência de acompanhantes. A frota foi considerada como homogênea, sendo estes veículos classificados como T3, cada um possuindo três lugares para pacientes simples apenas. A Tabela 5.2 ilustra a heterogeneização dos veículos e pacientes e os grupos criados.

Tabela 5.1: Ocupações de pacientes

Tipo de paciente	Tipo de Assento:			
	Simples	Cadeirante	Maca	Acompanhante
Simples	x	x		
Cadeirante		x		
Maca			x	
Acompanhante	x	x		x

Tabela 5.2: Informações adicionadas nas instâncias

Grupo Instâncias	Probabilidade do paciente ser:				Frota de
	Simples	Cadeirante	Maca	Acomp	Veículos
E	0.50	0.25	0.25	0.10	hom(T1)
I	0.83	0.11	0.06	0.50	het(T1, T2)
U	1.00	0.00	0.00	0.00	hom(T3)

Detalhes dos veículos na Tabela 5.3

Tabela 5.3: Informações sobre veículos

Tipo	Lugar para pacientes:			
	Simples	Cadeirante	Maca	Acomp
V1	1	1	1	2
V2	6	1	1	1
V3	3	0	0	0

5.2.2 Instâncias do contexto de Ouro Preto

Tendo em vista a aplicação da abordagem proposta na cidade de Ouro Preto, uma base de dados fornecida pelo setor de Transporte da PMOP foi utilizada para construção de instâncias que representam a realidade do serviço na cidade. Diferente das outras instâncias disponíveis na literatura, na cidade de Ouro Preto os pontos

de coleta dos pacientes podem ser coletivos, ou seja, mais de uma requisição pode ser atendida no mesmo local. Assim, entre os dados fornecidos pelo setor de transporte foram identificados seis pontos de coleta coletiva. Os outros pontos de coleta considerados nas instâncias são as residências dos pacientes que estão utilizando o serviço. Dessa maneira, dois tipos de coleta foram definidos, os pontos de coleta coletiva e os pontos residenciais. Estes locais são distribuídos na cidade de Ouro Preto, nos distritos da cidade e na cidade de Itabirito, que é uma cidade vizinha. Para os pontos de entrega são considerados os hospitais, clínicas médicas e centros de tratamento, todos situados na cidade de Belo Horizonte e em sua região metropolitana. A cidade de Itabirito também possui alguns pontos de entrega, pois alguns atendimentos são feitos nessa cidade. Entre os dados fornecidos para análise, as cidades de Ouro Branco-MG, Ponte Nova-MG, Mariana-MG não aparecem, devido ao baixo número de viagens feitas para essas cidades. Sendo assim, a Figura 5.1 representa em azul a região que possui apenas coletas, em vermelho a região que ocorrem apenas entregas e em amarelo a região que podem ocorrer ambos, coletas e entregas.

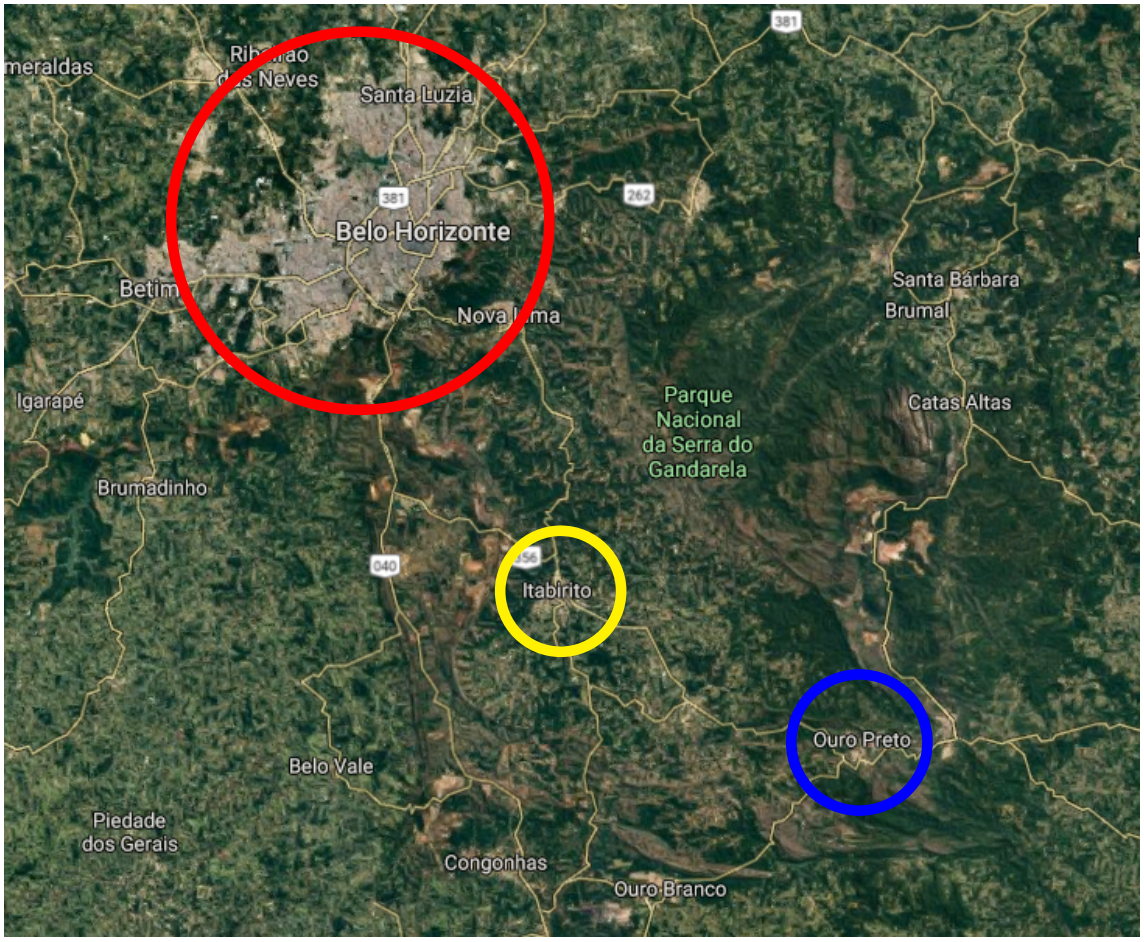

Figura 5.1: Regiões de coleta e entrega

Nas instâncias definidas neste trabalho considerou-se como janelas de tempo, o

horário de atendimento dos pacientes na região de entrega, assim, todas as janelas de tempo restritas são situadas nos pontos de entrega. Para localização dos pontos foram utilizadas coordenadas fornecidas pela aplicação *Google Maps*, assim a distância entre os pontos foi calculada como distância euclidiana. Para a frota de veículos, todos os veículos listados pelo setor de transporte estão como disponíveis, exceto o micro-ônibus, que possui horário e rota fixa. Um custo fixo é atribuído para utilização de cada veículo e está relacionado na Tabela 5.4. Essa frota de veículos é uma frota heterogênea (detalhes em 3.1.1). Para cada ponto foi definido um tempo de usuário $\tau_i = 10$, para cada pessoa neste ponto, um tempo máximo de viagem $L = 110$. O número de pessoas a serem adicionadas ao veículo em cada ponto pode ser um ou dois em pontos residenciais, dependendo da necessidade ou não de um acompanhante. Para pontos coletivos a adição de pessoas aos veículos é restrita à capacidade de cada veículo.

Tabela 5.4: Custo dos Veículos

Veículo	Representação	Capacidade	Custo
Administrativo 1	A	5	100
Administrativo 2	B	6	120
Ambulância 1	C	2	100
Ambulância 2	D	4	150
Van 1	E	16	200

Quatro instâncias foram criadas com base nos dados fornecidos. Estas instâncias possuem 21, 24, 34 e 48 requisições. A primeira instância é a9-21, com 21 requisições e foi criada com base em um único dia de trabalho do setor de transporte. Todas as requisições utilizadas nesta instância têm pontos exatamente iguais aos transportes realizados no dia analisado. Entre os pontos de coleta 14,29% deles são pontos residenciais, sendo o restante deles pontos coletivos. Da mesma forma a instância a9-24, esta com 24 requisições, representa exatamente a jornada de um dia de trabalho do setor de transporte, com 20,83% dos pontos de coleta sendo residenciais e o restante deles coletivos. Já a instância a9-34, foi definida com base na união de vários dias em que o número de requisições foi inferior a oito. Esta instância possui um número maior de pontos residenciais, são 41,18% dos pontos. Por fim, a instância a9-48 foi totalmente gerada, utilizando-se pontos de todos os dias informados na base de dados fornecida, os locais selecionados como pontos de coleta e entrega foram aleatoriamente definidos em meio a todos pontos presentes na base de dados. Metade dos pontos de coleta são pontos residenciais e a outra metade são pontos coletivos. Estas instâncias estão disponíveis em Souza (2019). A Tabela 5.5 mostra as configurações básicas das instâncias.

Tabela 5.5: Configuração das instâncias

Instância	Requisições	Pontos Coletivos	Pontos Residenciais
a9-21	21	18	3
a9-24	24	19	5
a9-34	34	20	14
a9-48	48	24	24

5.3 Experimentos

Para uma maior diversidade de buscas, seis vizinhanças inter-rota e três intrarrota foram definidas para resolução do DARP. Todas as três vizinhanças intrarrota foram utilizadas no algoritmo proposto e dentre as seis vizinhanças inter-rota (Realocação(1), Realocação(2), Troca(1), Troca(2,1), Troca(2) e Cruzamento), utilizou-se testes empíricos e a ferramenta IRACE (López-Ibáñez *et al.* , 2016) para seleção da melhor combinação de vizinhanças.

Nos testes empíricos, foram testadas todas as combinações possíveis das seis vizinhanças. Quando testadas separadamente, as vizinhanças Realocação(1) e Cruzamento se mostraram mais eficientes do que as demais. Nos testes feitos em duplas, as duplas que continham as vizinhanças Realocação(1), Troca(1) e Cruzamento obtiveram resultados melhores. Nos testes em trio, os trios em que as vizinhanças Realocação(1), Troca(1), Troca(2) e Cruzamento apareciam, os resultados obtidos pela aplicação dessas soluções eram superiores aos outros trios, destacando-se o trio que possuía Realocação(1), Troca(1) e Cruzamento.

Para as combinações com mais de três vizinhanças, a Tabela 5.7 mostra a média do GAP obtido de cada combinação de quatro vizinhanças delas, em relação aos melhores resultados obtidos até então para cada instância. Para a realização do teste empírico, foram selecionadas aleatoriamente nove instâncias dos três grupos da literatura relacionados na Seção 5.2.1, sendo um terço da seleção de cada grupo.

As combinações de vizinhanças adotadas nos teste estão apresentadas na Tabela 5.6. Nesta tabela, as combinações são definidas nas linhas, onde o "x" na coluna da vizinhança indica que ela foi utilizada na respectiva combinação.

Tabela 5.6: Combinações de vizinhanças

Nome	Realocação(1)	Troca(1)	Cruzamento	Troca(2,1)	Troca(2)	Realocação(2)
C1	x	x	x			
C2	x	x	x	x		
C3	x	x	x		x	
C4	x	x	x			x
C5	x	x	x	x	x	
C6	x	x	x	x		x
C7	x	x	x		x	x
C8	x	x	x	x	x	x

Tabela 5.7: Teste empírico para escolha de vizinhanças

Instância	Grupo	**BKS**	C1	C2	C3	C4	C5	C6	C7	C8
a2-16	E	**331,16**	0,003	0,000	0,004	0,008	0,000	0,003	0,000	0,036
a2-20	U	**344,83**	0,000	0,000	0,000	0,000	0,000	0,000	0,003	0,000
a3-18	U	**300,48**	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
a3-30	U	**494,85**	0,030	0,033	0,032	0,049	0,062	0,012	0,058	0,041
a4-32	E	**500,24**	0,049	0,020	0,038	0,024	0,036	0,027	0,035	0,016
a5-60	I	**816,15**	0,122	0,122	0,110	0,123	0,119	0,120	0,127	0,116
a6-72	I	**936,32**	0,141	0,102	0,123	0,128	0,109	0,109	0,154	0,129
a7-56	E	**740,63**	0,054	0,054	0,069	0,058	0,068	0,060	0,039	0,052
a8-64	I	**748,04**	0,026	0,024	0,031	0,029	0,021	0,026	0,038	0,029
Média Geral			0,047	**0,040**	0,045	0,046	0,045	0,041	0,051	0,047

C1, C2, C3, C4, C5, C6, C7, C8 = Tabela 5.6.

O método VNS2, utilizando as mesmas vizinhanças do VNS1, mostrou uma lentidão na execução e pequena diferença nos resultados comparado com execuções com apenas três vizinhanças. Dessa maneira, utiliza apenas três vizinhanças em sua busca local. Para seleção dos três dos parâmetros do VNS2 utilizou-se a ferramenta IRACE. Os parâmetros calibrados com a ferramenta foram: a combinação de vizinhanças NB, a penalização inicial PI e o número de iterações sem melhora $iterMax$. A Tabela 5.8 mostra os parâmetros calibrados, assim como os valores testados para cada parâmetro e destacado em negrito o valor escolhido para cada um deles. Para as vizinhanças foram considerados os valores 1 para Realocação(1), 2 para Troca(1), 3 para Cruzamento, 4 para Troca(2), 5 para Realocação(2) e 6 para Troca(2,1).

Tabela 5.8: Calibração de Parâmetros Utilizando o IRACE

Sigla	Parâmetro	Valor
NB	Vizinhanças	123, 124, 125, 126, 134, 135, 136 145, 146, 156, **234**, 235, 236, 245 246, 256, 345, 346, 356, 456
PI	Penalidade Inicial	1, 3, 5, 7, **10**
$iterMax$	Máximo de iterações	50, **100**, 150, 200

5.4 Resultados

Utilizando as instâncias descritas no Seção 5.2, a seguir os resultados obtidos da aplicação dos algoritmos propostos para cada grupo de instâncias. Em cada tabela a seguir a coluna **Instância** indica para qual instância o teste foi realizado, sendo o nome da instância definido como $ak - n$, em que k representa a quantidade de veículos e n a quantidade de requisições, de maneira que, a instância a3-30 possui

três veículos e 30 requisições. A coluna **OPT** apresenta o valor ótimo para cada instância, esses valores foram provados ótimos por Braekers *et al.* (2014). A coluna Parragh (2011) exibe os resultados obtidos pela autora em seus experimentos. Parragh (2011) propôs um algoritmo heurístico baseado em VNS. A coluna **MS-VNS1** apresenta os resultados encontrados pelo algoritmo MS-VNS1 e a coluna **VNS2** exibe os resultados obtidos pelo algoritmo VNS2. As subcolunas **med sol** e **melhor sol** apresentam a média das soluções encontradas em 10 execuções e a melhor solução encontrada. As subcolunas **gap** e **med gap** exibem a diferença entre o valor da melhor solução e da média das soluções, respectivamente, da solução ótima da instância e a subcoluna **tempo** a média do tempo computacional gasto em segundos nas 10 execuções.

Na Tabela 5.9 são apresentados os resultados de ambos os algoritmos propostos no grupo de instâncias E. As instâncias possuem usuários heterogêneos e frota de veículos homogênea. O algoritmo MS-VNS1 chegou a bons resultados chagando à um gap máximo de 3,8% com exceção das instâncias a3-36 e a4-40. O algoritmo MS-VNS1 encontrou apenas duas das soluções ótimas nesse grupo de instâncias. O algoritmo VNS2 mostrou-se mais eficiente, entre as 12 instâncias encontrou a solução ótima para quatro delas, e o pior gap chega à apenas 2,71%.

Em relação ao tempo computacional, o algoritmo MS-VNS1 foi mais lento que o VNS2 para todas as 12 instâncias, e comparado à Parragh (2011) é mais rápido em 5 das 12 instâncias. O algoritmo VNS2 é mais rápido em 9 das 12 instâncias comparado à Parragh (2011).

Tabela 5.9: Grupo de instâncias E

Instância	OPT	**Parragh (2011)**			**MS-VNS1**					**VNS2**				
		sol	gap	tempo	melhor sol	gap	med sol	med gap	tempo	melhor sol	gap	med sol	med gap	tempo
a2-16	**331,16**	**331,16**	0,00	65,60	**331,16**	0,00	336,48	1,58	18,75	**331,16**	0,00	**331,16**	0,00	4,89
a2-20	**347,03**	**347,03**	0,00	120,00	347,89	0,25	350,55	1,00	51,38	**347,03**	0,00	**347,03**	0,00	16,32
a2-24	**450,25**	**450,25**	0,00	160,40	450,35	0,02	459,05	1,92	128,66	450,38	0,02	452,92	0,65	76,08
a3-18	**300,63**	**300,63**	0,00	47,60	**300,63**	0,00	**300,63**	0,00	21,87	**300,63**	0,00	302,01	0,46	2,05
a3-24	**344,91**	**344,91**	0,00	76,20	345,31	0,11	356,45	3,24	100,63	**344,91**	0,00	347,63	0,86	17,73
a3-30	**500,58**	**500,58**	0,00	107,60	520,40	3,81	536,01	6,61	379,80	505,64	1,00	507,72	1,54	108,75
a3-36	**583,19**	**583,19**	0,00	161,60	636,68	8,40	658,91	11,49	1180,84	599,43	2,71	610,46	4,47	316,76
a4-16	**285,99**	**285,99**	0,00	25,00	291,55	1,91	291,67	1,95	9,61	291,55	1,91	291,76	1,98	1,25
a4-24	**383,84**	**383,84**	0,00	52,60	389,46	1,44	394,81	2,78	77,92	386,06	0,58	289,09	1,35	8,60
a4-32	**500,24**	502,52	0,45	83,00	510,67	2,04	519,73	3,75	326,27	501,85	0,32	507,11	1,35	49,16
a4-40	**580,42**	585,64	0,90	121,00	624,79	7,10	649,89	10,69	1113,30	589,29	1,51	597,89	2,92	204,60
a4-48	**670,52**	675,37	0,72	252,20	697,28	3,84	738,54	9,21	3166,54	676,28	0,85	688,57	2,62	786,57

Na Tabela 5.10 são apresentados os resultados da execução de ambos os algoritmos propostos no grupo de instâncias I. Esse grupo possui usuários e frota de veículos heterogêneos. O algoritmo MS-VNS1 chegou à solução ótima em apenas 2 instâncias e o restante obteve soluções com gap máximo de 1,90% com exceção das instâncias a3-36 e a4-48, que possuem respectivamente um gap de 4,63% e 7,14%. Assim como o algoritmo MS-VNS1, o VNS2 também encontrou apenas duas das soluções ótimas nesse grupo de instâncias. Entre as 10 instâncias restantes obteve soluções com uma diferença de no máximo 2,95% em relação às soluções ótimas.

Em relação ao tempo computacional, assim como no grupo E de instâncias, o algoritmo MS-VNS1 foi mais lento que o VNS2 para todas as 12 instâncias, e comparado à Parragh (2011) é mais rápido em 5 dos 12 casos. O algoritmo VNS2 é mais rápido em 9 das 12 instâncias comparado à Parragh (2011).

Tabela 5.10: Grupo de instâncias I

Instância	OPT	**Parragh (2011)**			**MS-VNS1**					**VNS2**				
		sol	gap	tempo	melhor sol	gap	med sol	med gap	tempo	melhor sol	gap	med sol	med gap	tempo
a2-16	**294,25**	**294,25**	0,00	68,40	**294,25**	0,00	295,84	0,55	17,46	**294,25**	0,00	**294,25**	0,00	5,33
a2-20	**355,74**	**355,74**	0,00	141,80	360,23	1,25	361,33	1,55	63,74	360,23	1,25	362,69	1,92	14,16
a2-24	**431,12**	**431,12**	0,00	211,00	438,96	1,79	448,24	3,82	136,81	434,53	0,78	441,32	2,31	59,08
a3-18	**302,17**	**302,17**	0,00	47,20	**302,17**	0,00	303,54	0,45	27,99	**302,17**	0,00	303,27	0,36	3,53
a3-24	**344,83**	**344,83**	0,00	83,60	346,88	0,59	359,93	4,19	118,13	345,31	0,14	350,79	1,70	19,23
a3-30	**494,85**	**494,85**	0,00	106,80	518,86	4,63	536,27	7,72	413,31	502,77	1,57	511,75	3,30	93,29
a3-36	**618,15**	618,58	0,07	170,60	665,31	7,09	678,54	8,90	1193,58	636,97	2,95	641,98	3,71	296,38
a4-16	**299,05**	**299,05**	0,00	27,00	302,65	1,19	303,28	1,40	11,09	302,87	1,26	307,23	2,66	1,09
a4-24	**375,02**	375,07	0,01	51,60	379,36	1,14	381,83	1,78	82,12	379,36	1,14	381,91	1,80	9,83
a4-32	**486,93**	**486,93**	0,00	88,00	492,73	1,18	512,18	4,93	317,93	488,74	0,37	496,25	1,88	59,96
a4-40	**557,69**	561,35	0,66	132,20	568,47	1,90	603,77	7,63	1050,67	566,11	1,49	573,02	2,68	279,11
a4-48	**670,72**	680,43	1,45	262,40	722,25	7,14	745,02	9,97	3288,66	684,37	1,99	695,95	3,63	749.48

Na Tabela 5.11 são apresentados os resultados de ambos os algoritmos propostos no grupo de instâncias U. Os usuários e frota de veículos são homogêneos nesse grupo de instâncias. O algoritmo MS-VNS1 encontrou a solução ótima para três instâncias desse grupo e obteve soluções com gap máximo de 4,09%. O algoritmo VNS2 chegou à solução ótima em 4 dos 12 casos e o pior gap chega à apenas 2,71%.

Em relação ao tempo computacional, novamente o algoritmo MS-VNS1 obteve um desempenho inferior ao VNS2 para todas as 12 instâncias, e mais rápido que Parragh (2011) em 7 delas. O algoritmo VNS2 possui um desempenho computacional melhor em 10 das 12 instâncias comparado ao método de Parragh (2011).

Tabela 5.11: Grupo de instâncias U

Instância	OPT	**Parragh (2011)**			**MS-VNS1**					**VNS2**				
		sol	gap	tempo	melhor sol	gap	med sol	med gap	tempo	melhor sol	gap	med sol	med gap	tempo
a2-16	**294,25**	**294,25**	0,00	68,20	**294,25**	0,00	295,84	0,54	15,16	**294,25**	0,00	**294,25**	0,00	3,83
a2-20	**344,83**	**344,83**	0,00	133,80	**344,83**	0,00	**344,83**	0,00	43,86	**344,83**	0,00	**344,83**	0,00	15,69
a2-24	**431,12**	**431,12**	0,00	187,80	434,53	0,78	441,75	2,41	123,85	434,53	0,78	436,48	1,23	42,26
a3-18	**300,48**	**300,48**	0,00	45,40	**300,48**	0,00	**300,48**	0,00	20,42	**300,48**	0,00	302,53	0,68	2,60
a3-24	**344,83**	**344,83**	0,00	86,80	350,41	1,59	357,20	3,46	84,35	344,91	0,02	347,50	0,77	14,58
a3-30	**494,85**	**494,85**	0,00	105,60	511,83	3,32	529,46	6,54	327,14	500,51	1,13	502,52	1,53	74,02
a3-36	**583,19**	583,30	0,02	162,60	608,08	4,09	642,19	9,19	960,29	599,43	2,71	607,24	3,96	247,54
a4-16	**282,68**	**282,68**	0,00	26,00	283,10	0,15	283,10	0,15	9,41	283,10	0,15	283,10	0,15	1,04
a4-24	**375,02**	**375,02**	0,00	50,80	380,90	1,54	381,48	1,69	62,79	379,36	1,14	380,81	1,52	6,36
a4-32	**485,50**	486,88	0,28	86,00	488,74	0,66	499,57	2,82	307,48	487,31	0,37	491,70	1,26	54,24
a4-40	**557,69**	561,80	0,74	130,60	572,58	2,60	592,69	5,90	892,05	**557,69**	0,00	569,36	2,05	243,66
a4-48	**668,82**	673,64	0,72	253,80	693,47	3,56	716,45	6,65	2568,28	678,98	1,50	683,13	2,09	580,92

Nas Tabelas 5.12 e 5.13 são apresentados o valor médio e o melhor valor de solução encontrado para a execução dos algoritmos **MS-VNS1** e **VNS2** nas instâncias criadas baseadas no contexto da cidade de Ouro Preto. A Tabela 5.13 apresenta os valores médios em 10 execuções dos algoritmos. As colunas **sol**, **CR**, **CV**, **qV** e **tempo**, representam o valor da função objetivo, o custo de roteamento, o custo dos veículos, a quantidade de veículos utilizados e o tempo computacional em segundos, nessa ordem.

A Tabela 5.12 apresenta a melhor solução encontrada por cada um dos algoritmos propostos para cada uma das instâncias. As colunas **sol**, **CR**, **CV**, **V1**, **V2**, **V3**, **V4**, **V5** e **tempo**, representam, respectivamente o valor da função objetivo, o custo de roteamento, o custo dos veículos, quantos veículos foram usados dos tipos de V1 a V5 (detalhes na Tabela 5.4) e o tempo computacional em segundos.

Na instância a9-21 o algoritmo VNS2 encontrou a melhor solução para a instância, mas o método MS-VNS1 encontrou em média soluções melhores levanto em conta os valores de função objetivo. Porém, o MS-VNS1 demorou cerca de 10 vezes mais tempo em média para encontrar as soluções. Os custos de roteamento encontrados pelo MS-VNS1 são melhores e o algoritmo utiliza veículos de custo mais alto para realizar o transporte. Mesmo assim, em média o MS-VNS1 chega a soluções melhores que o algoritmo VNS2.

Em relação à instância a9-24, o algoritmo VNS2 obteve desempenho superior ao MS-VNS1, encontrando a melhor solução e mantendo uma melhor média entre as soluções encontradas. Novamente os custos de roteamento encontrados pelas soluções do MS-VNS1 são menores, mas devido ao fato do custo de utilização dos veículos ser maior que o VNS2, as soluções do VNS2 obtiveram custo de função objetivo menor. Quanto ao tempo computacional novamente, o algoritmo VNS2 mostrou um melhor desempenho, chegando a ser cinco vezes mais rápido em média.

Na Instância a9-34, novamente o algoritmo VNS2 obteve desempenho superior ao MS-VNS1, encontrando a melhor solução e mantendo uma melhor média entre as soluções encontradas. Os custos de roteamento encontrados pelo VNS2 são maiores que os encontrados pelo MS-VNS1, mas devido ao fato do custo de utilização dos veículos ser menor, as soluções do VNS2 obtiveram custo de função objetivo menor. Quanto ao tempo computacional, algoritmo VNS2 é em média cerca de cinco vezes mais rápido. Por fim, na instância a9-48 tanto em relação à melhores valores de função objetivo, quanto a tempo computacional, o VNS2 obteve resultados melhores.

O MS-VNS1, para todos os casos, em média, encontrou menores valores de roteamento, porém, necessitou de mais veículos para atender as requisições. Portanto, nestes testes, o VNS2 mostrou ser superior ao MS-VNS1.

Tabela 5.12: Melhor resultado encontrado para as instâncias da Cidade de Ouro Preto

Instância	**MS-VNS1**					**VNS2**				
	sol	CR	CV	Veic	tempo	sol	CR	CV	Veic	tempo
a9-21	979,87	489,87	490	A_1, B_2, D_1	213,20	**941,82**	501,82	440	A_1, B_2, C_1	19,40
a9-24	1064,14	474,14	590	B_2, D_1, E_1	447,73	**980,90**	540,90	440	A_1, B_2, C_1	104,17
a9-34	1268,78	578,78	690	A_1, B_2, D_1, E_1	2366,18	**1136,68**	566,68	570	A_1, B_1, D_1, E_1	510,36
a9-48	1779,20	889,20	890	A_3, B_2, D_1E_1	7733,75	**1764,18**	974,18	790	A_4, B_2, D_1	3682,11

Tabela 5.13: Médias de 10 execuções dos algoritmos nas instâncias da Cidade de Ouro Preto

Instância	**MS-VNS1**					**VNS2**				
	sol	CR	CV	qV	tempo	sol	CR	CV	qV	tempo
a9-21	1033,43	469,43	564	4	221,04	1041,78	526,78	515	4	22,16
a9-24	1140,67	474,67	666	5	511,23	1108,42	513,42	595	5	109,86
a9-34	1348,39	588,39	760	6	2687,90	1325,03	635,03	690	5	529,67
a9-48	1932,38	932,38	1000	8	8308,17	1858,78	970,78	888	7	2432,41

Capítulo 6

Conclusões e Trabalhos Futuros

Este trabalho teve como objetivo resolver o problema *Dial-a-Ride*. Dois métodos meta-heurísticos, MS-VNS1 e VNS2, foram criados a fim de solucionar o problema e durante a implementação diversas vizinhanças foram testadas. Ambos os métodos foram baseados na meta-heurística *Variable Neighborhood Search*. Eles foram testados em instâncias disponíveis na literatura e comparados com autores que obtiveram os melhores resultados para a variante do problema. Boas soluções foram encontradas por ambos os métodos criados, sendo que o VNS2 chegou a melhores soluções em menor tempo. Para as instâncias consideradas pequenas, que mais se assemelham ao contexto estudado na cidade, a maioria das soluções ótimas foram encontradas. Para os casos em que os métodos não chegaram à solução ótima, os resultados dos algoritmos VNS2 e MS-VNS1 chegaram à um GAP de no máximo 2,95% e 8,4% respectivamente, em relação às soluções ótimas. Dessa maneira os algoritmos mostraram-se suficientes para atender as situações semelhantes as da cidade de Ouro Preto - MG, visto que a demanda de requisições diárias na cidade não é muito alta, variando de 20 a 30 requisições por dia.

Analisando o contexto da cidade de Ouro Preto, concluiu-se que o problema abordado pode ser caracterizado como o Problema *Dial-a-Ride* Heterogêneo, com características de heterogeneidade em relação aos pacientes e aos veículos. Por meio de dados fornecidos pela Prefeitura Municipal de Ouro Preto, quatro instâncias foram criadas para simular jornadas diárias do setor de transporte da Secretária de Saúde. Em duas dessas instâncias, a construção é feita de maneira semelhante a dois dias diferentes de trabalho do setor de transporte. Outras duas instâncias foram criadas contendo um número maior de requisições, simulando um provável aumento da demanda da cidade.

Testes foram realizados com as quatro instâncias que retratam a situação da cidade. O algoritmo VNS2 encontrou as melhores soluções para as quatro, porém na instância a9-21, mostrou um desempenho pior que o MS-VNS1 na média das 10 execuções. Em relação aos custos de roteamento, o algoritmo MS-VNS1 obteve

um desempenho melhor para todos os casos, porém necessitando de mais veículos. Em relação ao custo computacional, o algoritmo VNS2 mostrou ser superior ao MS-VNS1, na maioria dos casos ele foi cerca de cinco vezes mais rápido que o MS-VNS1.

Ambos os algoritmos se mostraram capazes de atender as demandas da cidade de Ouro Preto de forma automatizada, proporcionando ao setor de transporte da prefeitura uma ferramenta que possibilita reduzir os custos com o transporte de pacientes e diminuir a alocação de funcionários para cumprir essa atividade.

Como trabalhos futuros, propõe-se um estudo aprofundado sobre as variantes do DARP mais estudadas e quais os principais métodos utilizados para resolver cada uma delas. Dessa maneira, construir um único algoritmo que seja facilmente adaptável, e que resolva de maneira eficiente as principais variantes do DARP que combinem outras restrições relacionadas ao problema, desde que não sejam mutuamente excludentes.

Referências Bibliográficas

Attanasio, A., Cordeau, J. F., Ghiani, G., & Laporte, G. 2004. Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. *Parallel Computing*, **30**, 377–387.

Beaudry, A., Laporte, G., Melo, T., & Nickel, S. 2010. *Dynamic transportation of patients in hospitals*. Vol. 32.

Braekers, K., Caris, A., & Janssens, G. K. 2014. Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots. *Transportation Research Part B: Methodological*, **67**, 166–186.

Cordeau, J. F. 2006. A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research*, **54**, 573–586.

Cordeau, J. F., & Laporte, G. 2003. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, **37**, 579–594.

Cordeau, J. F., & Laporte, G. 2007. The dial-a-ride problem: Models and algorithms. *Annals of Operations Research*, **153**, 29–46.

Glaydston, M. R., Rodrigues, P. P., Resendo, L. C., & Mauri, G. R. 2014. Resolução de um caso real do problema dial-a-ride multicritério via clustering search. *Production*, **24**, 572–582.

Guerriero, F., Pezzella, F., Pisacane, O., & Trollini, L. 2014. Multi-objective optimization in dial-a-ride public transportation. *Transportation Research Procedia*, **3**, 299–308.

Hanne, T., Melo, T., & Nickel, S. 2018. Bringing Robustness to Patient Flow Management Through Optimized Patient Transportation in Hospitals. *INFORMS Journal on Computing*, **39**(3), 241–255.

Hansen, P., & Mladenović, N. 2001. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, **130**(3), 449–467.

Ho, S. C., Szeto, W. Y., Kuo, Y. H., Leung, J. M. Y., Petering, M., & Tou, T. W. H. 2018. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, **111**, 395–421.

Jaw, J. J., Odoni, A. R., Psaraftis, H. N., & Wilson, N. H. M. 1986. A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B*, **20**(3), 243–257.

Jorgensen, R. M., Larsen, J., & Bergvinsdottir, K. B. 2007. Solving the Dial-a-Ride problem using genetic algorithms. *Journal of the Operational Research Society*, **58**, 1321–1331.

Lois, A., & Ziliaskopoulos, A. 2017. Online algorithm for dynamic dial a ride problem and its metrics. *Transportation Research Procedia*, **24**, 377–384.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., & Stützle, T. 2016. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, **3**, 43–58.

Lutz, W., Sanderson, W., & Scherbov, S. 2008. The coming acceleration of global population ageing. *Nature*, **451**, 716–719.

Madsen, O. B. G., Ravn, H. F., & Rygaard, J. M. 1995. A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives. *Annals of Operations Research*, **60**, 193–208.

Martí, R. 2003. Chapter 12 MULTI-START METHODS. *Handbook of metaheuristics*, 355–368.

Masmoudi, M. A., Braekers, K., Masmoudi, M., & Dammak, A. 2017. A hybrid Genetic Algorithm for the Heterogeneous Dial-A-Ride Problem. *Computers & Operations Research*, **81**, 1–13.

Mauri, G. R., & Lorena, L. A. N. 2009. Uma nova abordagem para o problema dial-a-ride. *Produção*, **19**, 41–54.

Molenbruch, Y., Braekers, K., Caris, A., & Berghe, G. V. 2017. Multi-directional local search for a bi-objective dial-a-ride problem in patient transportation. *Computers and Operations Research*, **77**, 58–78.

Paquette, J., Cordeau, J. F., Laporte, G., & Pascoal, M. M. B. 2013. Combining multicriteria analysis and tabu search for dial-a-ride problems. *Transportation Research Part B: Methodological*, **52**, 1–16.

Parragh, S. N. 2011. Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. *Transportation Research Part C: Emerging Technologies*, **19**, 912–930.

Parragh, S. N., & Schmid, V. 2013. Hybrid column generation and large neighborhood search for the dial-a-ride problem. *Computers and Operations Research*, **40**, 490–497.

Parragh, S. N., Doerner, K. F., & Hartl, R. F. 2010. Variable neighborhood search for the dial-a-ride problem. *Computers and Operations Research*, **37**, 1129–1138.

Rahmani, N., Detienne, B., Sadykov, R., & F., Vanderbeck. 2017. A Column Generation Based Heuristic for the Dial-A-Ride Problem.

Ropke, S., Cordeau, J. F., & Laporte, G. 2007. Models and Branch-and-Cut Algorithms for Pickup and Delivery Problems with Time Windows. *NETWORKS*, **49(4)**, 258–272.

Schilde, M., Doerner, K. F., & Hartl, R. F. 2011. Metaheuristics for the dynamic stochastic dial-a-ride problem with expected return transports. *Computers and Operations Research*, **38**, 1719–1730.

Schilde, M., Doerner, K. F., & Hartl, R. F. 2014. Integrating stochastic time-dependent travel speed in solution methods for the dynamic dial-A-ride problem. *European Journal of Operational Research*, **238**, 18–30.

Souza, A. L. S. 2019. Dial-a-Ride Problem (DARP) data set from Ouro Preto-MG, Brazil. Mendeley dataSet, Vehicle Routing Problem, DARP.

Toth, P., & Vigo, D. 2002. *The Vehicle Routing Problem*.

Zhang, Z., Liu, M., & Lim, A. 2015. A memetic algorithm for the patient transportation problem. *Omega (United Kingdom)*, **54**, 60–71.