JONATAS BATISTA COSTA DAS CHAGAS

ALGORITMOS EXATOS E HEURÍSTICOS PARA O PROBLEMA DE ROTEAMENTO DUPLO DE VEÍCULOS COM MÚLTIPLAS PILHAS E DEMANDA HETEROGÊNEA

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae*

VIÇOSA
MINAS GERAIS − BRASIL
2017

Ficha catalográfica preparada pela Biblioteca Central da Universidade Federal de Viçosa - Câmpus Viçosa

T
C433a
2017

Chagas, Jonatas Batista Costa das, 1991-
Algoritmos exatos e heurísticos para o problema de roteamento duplo de veículos com múltiplas pilhas e demanda heterogênea / Jonatas Batista Costa das Chagas. – Viçosa, MG, 2017.
xi,139f. : il. ; 29 cm.

Orientador: André Gustavo dos Santos.
Dissertação (mestrado) - Universidade Federal de Viçosa.
Referências bibliográficas: f.131-139.

1. Pesquisa operacional. 2. Otimização combinatória. 3. Otimização matemática. 4. Programação heurística. 5. Logística. I. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-graduação em Ciência da Computação. II. Título.

CDD 22 ed. 003

JONATAS BATISTA COSTA DAS CHAGAS

ALGORITMOS EXATOS E HEURÍSTICOS PARA O PROBLEMA DE ROTEAMENTO DUPLO DE VEÍCULOS COM MÚLTIPLAS PILHAS E DEMANDA HETEROGÊNEA

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

APROVADA: 07 de março de 2017.

Geraldo Robson Mateus

Marcone Jamilson Freitas Souza

André Gustavo dos Santos
(Orientador)

Dedico este trabalho aos meus pais,
à minha irmã e também ao meu
cachorro, o famoso Bob.

Agradecimentos

Agradeço a Deus por me manter firme, forte e com saúde para que eu conseguisse chegar até aqui. Nem todos dias são fáceis, mas com a companhia de Deus tudo pode ser superado. Deus toma conta! Deus nos protege! Deus é que olha!

Aos meus queridos e amados pais, João Batista e Adelma, pelo carinho, conselhos e por sempre me apoiarem nas minhas decisões. À minha querida irmã, Jaqueline, por sempre estar disposta a me ajudar. Vocês são os mais nobres exemplos de pessoas. Estou tentando imitá-los. Eu amo vocês!

Agradeço a todos os professores do Departamento de Informática da Universidade Federal de Viçosa, que desde a graduação contribuíram e foram os maiores responsáveis pela minha formação acadêmica. Especialmente, agradeço ao meu professor, orientador e amigo André Gustavo dos Santos pela dedicação, atenção, paciência e pelos ricos ensinamentos e conselhos.

Serei eternamente grato à Universidade Federal de Viçosa e também à cidade de Viçosa por me acolher durante todos esses anos, me proporcionando todo o suporte necessário para que conseguisse concluir a graduação e agora o mestrado.

Também agradeço a todos os amigos que tive a oportunidade de conhecer em Viçosa. Aos meus queridos companheiros do alojamento 2221 e da república Jogatina, que me proporcionaram momentos incrivelmente divertidos e de grande crescimento pessoal. Aos meus colegas de mestrado pela amizade dentro e fora da universidade. Dizem que a amizade é um amor que nunca morre, e eu acredito nisso!

Agradeço à CAPES que financiou este trabalho, tornando possível sua realização.

Enfim, obrigado a todos que me conduziram até aqui.

"A ciência nunca resolve um problema sem criar pelo menos outros dez."
(George Bernard Shaw)

Sumário

Resumo

CHAGAS, Jonatas Batista Costa das, M.Sc., Universidade Federal de Viçosa, março de 2017. **Algoritmos Exatos e Heurísticos para o Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas e Demanda Heterogênea**. Orientador: André Gustavo Dos Santos.

Este trabalho aborda dois problemas de roteamento de veículos de coleta e entrega com restrições de carregamento. Primeiramente foi tratado o Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas (*Double Vehicle Routing Problem with Multiple Stacks* - DVRPMS). Posteriormente foi formulado e proposto o Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas e Demanda Heterogênea (*Double Vehicle Routing Problem with Multiple Stacks and Heterogeneous Demand* - DVRPMSHD), se referindo a um caso mais realista do DVRPMS, quando os clientes têm demandas múltiplas e heterogêneas, sendo que toda a demanda de um mesmo cliente deve ser transportada por um único veículo. Em ambos os problemas, o objetivo é determinar rotas para uma frota de veículos a fim de atender a demanda de um conjunto de clientes de forma que a distância percorrida pelos veículos seja a mínima possível, respeitando algumas restrições de carregamento impostas pelas pilhas de armazenamento dos veículos. Todos os produtos localizados em uma região de coleta devem ser coletados e depois entregues em uma região de entrega pelos veículos. As regiões de coleta e entrega são largamente separadas, portanto todos os produtos devem ser carregados antes de qualquer descarregamento. O DVRPMS foi

abordado principalmente por quatro métodos heurísticos, os quais foram testados em diversas instâncias e comparados aos métodos exatos e heurísticos já existentes na literatura. Os experimentos computacionais mostraram a eficiência dos algoritmos propostos, obtendo soluções de melhor qualidade que as soluções apresentadas na literatura para a maioria dos casos de teste com baixo tempo computacional. Já o DVRPMSHD foi abordado de forma exata e heurística. Inicialmente, foi desenvolvido um método exato *branch-and-price* que apresentou maior eficiência quando comparado à formulação matemática também proposta para o problema. O método heurístico superou os resultados alcançados pelo *branch-and-price* para a maioria das instâncias de teste formuladas.

Abstract

CHAGAS, Jonatas Batista Costa das, M.Sc., Federal University of Viçosa, March, 2017. **Exact and Heuristic Algorithms for the Double Vehicle Routing Problem with Multiple Stacks and Heterogeneous Demand**. Adviser: André Gustavo Dos Santos.

In this work we address two vehicle routing problems with pickup and delivery and loading constraints. Firstly, this work addresses the Double Vehicle Routing Problem with Multiple Stacks (DVRPMS). Posteriorly we formulate and propose the Double Routing Vehicle Problem with Multiple Stacks and Heterogeneous Demand (DVRPMSHD), referring to a more realistic case of the DVRPMS in which customers have multiple and heterogeneous demands and all demand of a same client must be transported by a single vehicle. In both problems, the objective is to determine routes for a fleet of vehicles to meet the demand of a set of customers so that the distance travelled by the vehicles is the minimum possible, respecting some loading constraints imposed by the vehicles' storage stacks. All products located in a pickup region must be collected and then delivered to a delivery region by vehicles. The pickup and delivery regions are largely separated so that all products must be loaded before any unloading. The DVRPMS was approached mainly by four heuristic methods, which were tested in several instances and compared to the exact and heuristic methods already present in literature. The computational experiments showed the efficiency of the proposed algorithms, obtaining solutions

of better quality than those presented in the literature for most of the instances and with low computational time. The DVRPMSHD was approached by an exact method and a heuristic method. Initially, the implemented branch-and-price exact method presented higher efficiency compared to the proposed mathematical formulation for the problem. The heuristic method overcame the results achieved by the branch-and-price for most of the created instances.

Capítulo 1

Introdução

Com a crescente competição entre as empresas que buscam alcançar a consolidação no mercado, avanços científicos/tecnológicos e planejamentos estratégicos de negócios são cruciais para que estas se tornem mais competitivas e consigam conquistar o mercado consumidor. Nesse propósito, as empresas buscam de forma constante aperfeiçoar seus processos logísticos.

Dentre os processos logísticos, o planejamento de transporte se destaca com grande importância, sendo fundamental na prestação de serviços aos clientes [Grabara et al., 2014]. O planejamento de transportes é um dos pontos estratégicos que precisam ser considerados para o êxito das empresas, pois se mal executado pode levá-las à falência.

O objetivo do planejamento de transporte é desenvolver métodos otimizados que minimizem os custos gerados para atender uma certa demanda através de uma frota de veículos, assegurando que as restrições impostas pelo contexto de cada empresa sejam satisfeitas.

O grande problema que comumente se apresenta àquele que pretende realizar o planejamento de transportes é a dificuldade de realizar tal planejamento de forma que os custos associados sejam os menores possíveis, sem desconsiderar as restrições envolvidas, como restrições de tempo, capacidade dos veículos, demanda dos clientes, etc.

Devido à importância econômica e à complexidade de resolução, os problemas de transporte são largamente estudados pela comunidade científica. Uma versão

clássica do problema é conhecida como o Problema de Roteamento de Veículos (*Vehicle Routing Problem* - VRP). O referido problema foi formulado pela primeira vez por Dantzig & Ramser [1959] e teve como objetivo encontrar as rotas ideais (mais curtas) para uma frota de caminhões que transportavam gasolina de um único terminal (depósito) para diversos clientes.

O VRP é um problema de natureza combinatória e pertence à classe $\mathcal{NP}$-Difícil [Lenstra & Kan, 1981]. Desde que foi proposto em 1959, o VRP vem despertando o interesse de pesquisadores que buscam propor novos métodos para resolvê-lo, bem como suas variações.

Um fator que alimenta a importância do VRP é a necessidade de se adaptar os processos logísticos às necessidades atuais, uma vez que os custos associados aos transportes desempenham um alto fator no custo logístico na maioria das empresas [Aggelakakis et al., 2015]. Portanto, variações do VRP se tornam naturais e necessárias para descrever diferentes situações do mundo real.

De acordo com Desrochers et al. [1990], as variantes do VRP diferem em aspectos relacionados ao tipo de frota de veículos, localização e demanda dos clientes, existência de restrições temporais e espaciais, tipo de carregamento, objetivo do problema e várias outras características relacionadas a um determinado cenário.

A fim de suprir as necessidades do mundo moderno, as empresas têm buscado meios cada vez mais práticos para realizar o transporte de seus produtos. Segundo Rondinelli & Berry [2000], com o aumento da globalização e os tratados entre nações, aliado ao aumento da população e de seu consumo, as necessidades por insumos e produtos se tornaram cada vez mais imediatas, demandando uma adaptação da indústria neste sentido, trocando o modelo simples de transporte rodoviário por meios mais complexos.

Entre as importantes adaptações que vêm sendo praticadas, destacam-se os transportes intermodais e os multimodais. Ambos consistem no uso combinado de diferentes modais de transporte (rodoviário, aquaviário, ferroviário e outros) a fim de tornar os serviços de transporte mais eficientes [Rondinelli & Berry, 2000].

De acordo com Crainic & Kim [2007], tem-se visto um aumento anual de cerca de 15% no transporte por contêineres, principalmente por razões de segurança,

redução de custos de manutenção e, consequentemente, no melhor acesso de diferentes métodos de transporte, o que gerou um número considerável de pesquisas nos últimos anos sobre transportes intermodais/multimodais.

SteadieSeifi et al. [2014] fizeram uma avaliação técnica dos estudos de planejamento sobre transportes multimodais. Os principais aspectos que têm ditado os diferentes problemas de transportes nesta área incluem veículos capacitados ou não, o tipo de alocação, problemas de agendamento ou devido a incertezas.

Nesta dissertação são abordados dois problemas de roteamento de veículos, sendo o segundo, uma generalização do primeiro. Em ambos os problemas é preciso atender um conjunto de clientes através de operações de coleta e entrega, sendo que a ordem de atendimento desses clientes deve respeitar algumas restrições. Tais problemas surgem em contextos onde as regiões de coleta e entrega são amplamente separadas e são empregados diferentes modais de transportes para atender os clientes. Porém, como será visto adiante, apenas um modal (rodoviário) faz parte da otimização dos dois problemas abordados neste trabalho, sendo que os demais modais apresentam custos operacionais fixos, podendo ser desconsiderados da fase de otimização.

O primeiro problema aqui abordado é o Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas (*Double Vehicle Routing Problem with Multiple Stacks* - DVRPMS). O DVRPMS foi introduzido na literatura por Iori & Riera-Ledesma [2015], sendo uma generalização do bem conhecido Problema do Caixeiro Viajante Duplo com Múltiplas Pilhas (*Double Traveling Salesman Problem with Multiple Stacks* - DTSPMS) proposto por Petersen & Madsen [2009].

O DTSPMS trata-se de um problema em que um único veículo é responsável por coletar todos os produtos de clientes localizados em uma região de coleta e, em seguida, entregar todos os produtos coletados aos clientes localizados na região de entrega. A cada cliente localizado na região de coleta é associado um único produto que deve ser transportado para um determinado cliente na região de entrega. Em ambas as regiões, o veículo inicia e termina sua rota em um depósito pré-definido em cada região. O transporte entre a região de coleta e a região de entrega é feita por outros modais de transportes, tendo um custo fixo e não faz parte da otimização do problema. O compartimento de carga, isto é, o contêiner do veículo, é dividido em pilhas de alturas fixas. À medida que os produtos são carregados na região de

coleta eles são armazenados pela parte traseira do veículo nas pilhas. Na região de entrega os produtos devem ser entregues respeitando a política *Last-In-First-Out* (LIFO) em cada uma das pilhas, ou seja, apenas os produtos acessíveis a cada momento a partir da traseira do veículo ("topo" das pilhas) podem ser entregues. Portanto a ordem de entrega dos produtos na região de entrega está condicionada à configuração do contêiner carregado na região de coleta. O objetivo do DTSPMS é minimizar a distância percorrida pelo veículo, assegurando que a política LIFO seja respeitada em todas as pilhas do veículo.

O DVRPMS é uma generalização do DTSPMS. Segundo Iori & Riera-Ledesma [2015], essa generalização é motivada pelo fato de que em algumas situações apenas um veículo não é capaz de atender todos os clientes, tornando assim necessário mais de um veículo. Da mesma forma que no DTSPMS, no DVRPMS deve-se coletar os produtos da região de coleta e entregá-los em seus respectivos lugares na região de entrega. A diferença é que agora existe uma frota de veículos para realizar tais operações. A frota de veículos pode ser heterogênea, isto é, os contêineres dos veículos podem ter dimensões diferentes. O objetivo do problema é determinar rotas para a frota de veículos que atendam todos os clientes de forma que a distância total percorrida pelos veículos seja a menor possível, garantindo que a política LIFO seja respeitada em todas as pilhas de todos os veículos.

O segundo problema estudado nesta dissertação foi proposto e introduzido na literatura a partir do trabalho desenvolvido no mestrado que esta dissertação referencia. O problema foi chamado de Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas e Demanda Heterogênea, em inglês *Double Vehicle Routing Problem with Multiple Stacks and Heterogeneous Demand* (DVRPMSHD) Chagas & Santos [2016]. O DVRPMSHD é uma generalização do DVRPMS e foi motivado pelo fato de se tratar de um caso mais realístico do DVRPMS, onde os clientes podem ter demandas heterogêneas, isto é, a cada cliente podem estar associados vários produtos e esse número de produtos ser independente dos demais clientes. Da mesma forma que no DVRPMS, o objetivo do DVRPMSHD é encontrar rotas para a frota de veículos que minimizem o custo total percorrido, respeitando a política LIFO nas pilhas de todos os veículos. O DVRPMSHD tem uma restrição adicional que obriga que todos os produtos de um mesmo cliente sejam transportados por um mesmo veículo.

Nos capítulos posteriores, ambos os problemas são descritos com maior clareza, detalhando suas características e apresentando exemplos que facilitam seus entendimentos, bem como os algoritmos propostos para resolvê-los.

1.1 Motivação

Os problemas de roteamento de veículos surgem em vários contextos de logística, modelando diferentes e importantes problemas do mundo real. O DVRPMS e o DVRPMSHD são problemas complexos de se resolver e tratam-se de problemas que modelam casos mais realísticos do DTSPMS, que é um problema com aplicações reais [Petersen & Madsen, 2009].

O DVRPMS foi introduzido na literatura pelo trabalho [Iori & Riera-Ledesma, 2015], sendo que os autores apresentaram três algoritmos exatos para sua solução. Até o início deste trabalho, tinha-se conhecimento apenas de um artigo que tratava esse problema com abordagens heurísticas [Silveira et al., 2015].

O DVRPMSHD, por sua vez, foi inserido na literatura pelo trabalho de [Chagas & Santos, 2016], sendo o problema tratado apenas por um algoritmo exato.

Portanto, devido ao fato dos problemas terem sido recentemente introduzidos na literatura, o número de trabalhos que os abordam de forma heurística ainda é pequeno. A grande oportunidade de explorar métodos heurísticos para resolvê-los, a complexidade computacional e a importância prática dos problemas, tornaram-se as principais motivações para este trabalho.

1.2 Contribuições

Neste trabalho é proposta uma nova formulação matemática de programação inteira (*Integer Linear Programming* - ILP) para o DVRPMS, bem como quatro algoritmos heurísticos (*Simulated Annealing* (SA)[1], *Variable Neighborhood Search* (VNS) e *Iterated Local Search* (ILS)) para o mesmo. Todas as abordagens foram implementadas e testadas com casos de testes contidos na literatura. Os quatro algoritmos heurísticos propostos foram calibrados a fim de determinar os melhores valores de seus parâmetros.

[1]Dois algoritmos foram baseados na metaheurística SA.

Também foi proposto um novo problema de roteamento de veículos, o DVRPMSHD, que é uma generalização do DVRPMS. Para o DVRPMSHD foi proposta uma formulação matemática de programação inteira (ILP), um algoritmo exato (*branch-and-price*) e uma abordagem heurística (SA). Todas as abordagens propostas foram testadas usando-se um conjunto de instâncias criadas baseado nas instâncias do DVRPMS. O algoritmo heurístico foi calibrado e os melhores valores de seus parâmetros foram escolhidos.

Com relação aos resultados obtidos, os algoritmos heurísticos para o DVRPMS obtiveram a solução ótima para a maioria das instâncias da literatura com baixo tempo computacional. Para o DVRPMSHD o algoritmo exato *branch-and-price* provou a otimalidade para um grande número de instâncias de teste. Enquanto o algoritmo heurístico também apresentou bons resultados, sendo também capaz de encontrar a mesma solução ótima para quase todas as instâncias que foram resolvidas na otimalidade pelo *branch-and-price*. Para a maioria das instâncias que não se conhece a solução ótima, o algoritmo heurístico encontrou soluções melhores que as encontradas pelo algoritmo *branch-and-price*.

1.3 Organização da dissertação

Este trabalho está dividido em 5 capítulos, incluindo essa introdução. No Capítulo 2 são apresentadas as características que compõem os dois problemas (DVRPMS e DVRPMSHD) abordados nesta dissertação. No Capítulo 3 é apresentado o DVRPMS, detalhando suas características. Também no Capítulo 3 o DVRPMS é formalmente definido, os trabalhos anteriores contidos na literatura são discutidos, é apresentada uma nova formulação matemática de programação inteira que modela o problema, são descritos os componentes e estruturas dos algoritmos heurísticos propostos, e, por fim, são apresentados os resultados obtidos pelas abordagens propostas para o DVRPMS. O Capítulo 4 é destinado à apresentação do DVRPMSHD, onde o problema é formalmente definido, é apresentada a formulação matemática de programação inteira, é descrito o algoritmo exato proposto, bem com o algoritmo heurístico proposto, e, por fim, são apresentados os resultados obtidos pelas abordagens propostas para o problema DVRPMSHD. Finalmente, no capítulo 5 são apresentadas as considerações finais.

Capítulo 2

Referencial Teórico

Neste capítulo são apresentados os conceitos básicos e os principais trabalhos necessários para fazer uma contextualização dos problemas a serem tratados nesta dissertação.

2.1 Problemas de roteamento de veículos

Os Problemas de Roteamento de Veículos (*Vehicle Routing Problems* - VRP's) são problemas clássicos na área de pesquisa operacional, sendo profundamente estudados pela comunidade científica da área. De forma geral, o objetivo dos VRP's consiste em determinar rotas para uma frota de veículos que atendam a um conjunto de clientes, de forma que o custo para realizar esse atendimento seja o menor possível, satisfazendo as restrições e características de diferentes contextos.

A versão clássica dos VRP's foi proposta por Dantzig & Ramser [1959] e consistiu em um problema que teve como objetivo encontrar as rotas mais curtas para uma frota de caminhões que transportavam gasolina de um único terminal (depósito) para diversos clientes. Os VRP's são problemas de natureza combinatória e pertencem à classe $\mathcal{NP}$-Difícil [Lenstra & Kan, 1981], e desde que sua versão clássica foi proposta, vêm despertando o interesse de pesquisadores que buscam propor novos métodos mais eficientes para resolvê-los [Laporte et al., 2000].

Entre as variantes do VRP, algumas se destacam pelo grande interesse teórico e prático, tais como: o Problema de Roteamento de Veículos com Coleta e Entrega

(*Vehicle Routing Problem with Pickup and Delivery* - VRPPD) que surge quando produtos precisam ser transportados de pontos de coletas para pontos de entregas [Dumas et al., 1991; Savelsbergh & Sol, 1995; Berbeglia et al., 2007, 2010], o Problema de Roteamento de Veículos Capacidados (*Capacitated Vehicle Routing Problem* - CVRP) onde os veículos devem atender aos clientes respeitando um limite de carga [Laporte et al., 1986; Toth & Vigo, 2002; Lysgaard et al., 2004; Fukasawa et al., 2006], o Problema de Roteamento de Veículos com Janelas de Tempo (*Vehicle Routing Problem with Time Windows* - VRPTW) onde cada cliente pode definir uma janela de tempo para que seja atendido [Solomon, 1987; Dumas et al., 1991; Desrochers et al., 1992], os Problemas de Roteamento com Restrições de Carregamento (*Routing Problems with Loading Constraints*) quando é necessário realizar o roteamento de forma que sejam respeitadas certas restrições de carregamento [Iori et al., 2007; Gendreau et al., 2008; Fuellerer et al., 2010].

Um outro problema na linha de roteamento é o Problema do Caixeiro Viajante (*Traveling Salesman Problem* - TSP). O TSP é um problema de simples descrição, mas que apresenta grande complexidade para ser resolvido. O objetivo do problema é encontrar o ciclo de menor distância que visita todos os clientes exatamente uma vez. O ciclo deve começar e terminar em um único depósito.

O TSP pode ser definido formalmente por meio de um grafo $G = (V, E)$, onde o conjunto de vértices V representa os clientes, o conjunto de arestas E representa os possíveis caminhos entre esses clientes e para cada aresta $(i, j) \in E$ é associado um custo c_{ij} para transitar do cliente i para o cliente j. O objetivo do TSP é encontrar o ciclo Hamiltoniano em G em que a soma dos custos das arestas contidas no ciclo seja o menor possível [Gutin & Punnen, 2002]. O TSP é um problema pertencente à classe de problemas $\mathcal{NP}$-Difícil [Augustine, 2002].

O clássico TSP e suas variações são amplamentes estudados na literatura por terem grandes aplicações no mundo real. As aplicações do TSP e de suas variações vão muito além do problema de planejamento da rota de um caixeiro viajante; na verdade, se estende por várias áreas do conhecimento incluindo matemática, genética, logística, telecomunicações, engenharia e eletrônica [Gutin & Punnen, 2002; Applegate et al., 2011].

A seguir é apresentada a classe de problemas que envolvem coletas e entregas, bem como uma variante do TSP pertencente à essa classe.

2.2 Problemas de coleta e entrega

Os Problemas de Coleta e Entrega (*Pickup and Delivery Problems* - PDP's) são problemas pertencentes a uma classe de problemas de roteamento de veículos em que produtos ou pessoas devem ser transportados entre suas origens e destinos específicos. Tais problemas são largamente estudados, pois modelam diferentes situações presenciadas no cotidiano de prestadoras de serviços de transporte.

Segundo Parragh et al. [2008a,b] a classe dos PDP's pode ser dividida em duas subclasses de problemas. A primeira contém os problemas em que o transporte dos produtos é realizado entre os clientes e os depósitos, ou seja, toda a demanda dos clientes deve ser suprida pelos produtos carregados nos depósitos e toda oferta dos clientes deve ser retornada aos depósitos. Esses problemas são geralmente referenciados por Problemas de Roteamento de Veículos com *Backhauls* (*Vehicle Routing Problems with Backhauls* - VRPB) [Goetschalckx & Jacobs-Blecha, 1989; Gendreau et al., 1996; Toth & Vigo, 1999]. A segunda classe contém todos os problemas em que os produtos são transportados entre os clientes de coleta e os clientes de entrega. Esses problemas são geralmente chamados de Problemas de Roteamento de Veículos com Coleta e Entrega (*Vehicle Routing Problems with Pickups and Deliveries* - VRPPD) [Mosheiov, 1994; Nagy & Salhi, 2005].

Uma segunda classificação desses problemas é realizada por Berbeglia et al. [2010], onde a classe dos PDP's é dividida em três grupos diferentes:

- muitos-para-muitos (*many-to-many*): grupo de problemas em que os produtos são transportados do depósito aos clientes e entre os clientes, onde qualquer produto pode suprir a oferta/demanda de qualquer cliente.
- um-para-muitos-para-um (*one-to-many-to-one*): grupo de problemas em que os produtos destinados à satisfazer as demandas dos clientes estão localizados apenas no depósito e os produtos coletados nos clientes são destinados para o depósito.
- um-para-um (*one-to-one*): grupo de problemas em que cada produto tem uma origem e um destino já determinados.

Um importante problema que envolve coletas e entregas é conhecido por Problema do Caixeiro Viajante com Coleta e Entrega (*Traveling Salesman Problem*

with Pickup and Delivery - TSPPD). O TSPPD é classificado como um problema *one-to-one* de roteamento de um único veículo com coleta e entrega [Dumitrescu et al., 2010], tendo várias aplicações práticas no transporte de carga e de passageiros [Parragh et al., 2008b]. Esse problema é uma variante do clássico Problema do Caixeiro Viajante (*Traveling Salesman Problem* - TSP). Assim como no TSP, no TSPPD um único veículo deve atender um conjunto de solicitações de clientes. A diferença é que no TSPPD, a cada solicitação são associados um local de origem, de onde um produto deve ser coletado, e um local de destino, onde o produto deverá ser entregue. O objetivo do TSPPD consiste em determinar um ciclo Hamiltoniano mais curto que atenda todas as solicitações dos clientes, assegurando que a coleta de uma determinada solicitação seja realizada antes de sua entrega.

Segundo [Dumitrescu et al., 2010], o TSPPD é $\mathcal{NP}$-Difícil, uma vez que qualquer instância TSP pode ser transformada em uma instância TSPPD usando uma transformação polinomial [Renaud et al., 2002].

Algoritmo exatos [Kalantari et al., 1985; Hernández-Pérez & Salazar-González, 2004] e heurísticos [Renaud et al., 2000, 2002] foram introduzidos na literatura para resolver o TSPPD.

2.3 Problemas de roteamento de veículos com restrições de carregamento

Os Problemas de Roteamento de Veículos com Restrições de Carregamento *(Vehicle Routing Problems with Loading Constraints* - VRPLC's) pertencem a uma classe de problemas de roteamento de veículo em que é necessário otimizar o roteamento dos veículos, sendo que esse roteamento deve ser realizado de forma que sejam respeitadas algumas restrições de atendimento/carregamento.

Iori & Martello [2010] fizeram um estudo sobre o atual estado dos VRPLC's, apontando grandes aplicações desses problemas em casos reais da indústria, uma vez que esses problemas combinam os dois principais componentes do processo de transporte: roteamento e carregamento.

Estudos que tratam os VRPLC's são recentes, sendo que alguns desses problemas surgiram a partir da combinação de problemas de carregamento com

problemas de roteamento, como por exemplo, o Problema de Empacotamento de Duas Dimensões (*Two-Dimensional Bin Packing Problem* - 2BPP) [Martello & Vigo, 1998], que foi combinado com o clássico Problema de Roteamento de Veículos Capacitado (*Capacitated Vehicle Routing Problem* - CVRP) [Laporte et al., 1986], gerando o Problema de Roteamento de Veículo Capacitado com Restrições de Carregamento Bidimensional (*Capacitated Vehicle Routing Problem with Two-Dimensional Loading Constraints* - 2L-CVRP) [Gendreau et al., 2008].

A seguir são apresentados alguns problemas de roteamento de veículos que envolvem coletas, entregas e restrições de carregamento. Esses problemas são fundamentais para contextualizar os problemas tratados nesta dissertação.

2.3.1 Problema do caixeiro viajante com coleta e entrega com restrições de carregamento LIFO

O Problema do Caixeiro Viajante com Coleta e Entrega com Restrições de Carregamento LIFO (*Traveling Salesman Problem with Pickup and Delivery and LIFO Loading Constraints* - TSPPDL) é uma variante do TSPPD em que as coletas e entregas dos produtos devem respeitar a ordem *Last-In-First-Out* (LIFO). Portanto, um produto ao ser coletado é armazenado pela parte traseira do veículo, e esse produto só poderá ser entregue quando estiver acessível a partir da parte traseira do veículo.

O TSPPDL surge naturalmente em situações onde o veículo tem apenas um ponto de acesso para carregar e descarregar os produtos, e também quando é necessário evitar realocações de produtos, quando estes forem grandes, pesados ou frágeis [Cordeau et al., 2010].

Segundo Iori & Martello [2010], o primeiro trabalho relacionado ao TSPPDL foi realizado por Ladany & Mehrez [1984], onde foi tratado um problema real de entrega de contêineres de leite. Posteriormente o problema foi tratado por métodos exatos [Carrabs et al., 2007a; Cordeau et al., 2010] e por métodos heurísticos [Pacheco, 1997; Carrabs et al., 2007b].

2.3.2 Problema do caixeiro viajante com coleta e entrega sob múltiplas pilhas

O Problema do Caixeiro Viajante com Coleta e Entrega sob Múltiplas Pilhas (*Pickup and Delivery Traveling Salesman Problem with Multiple Stacks* - PDTSPMS) é uma variação do TSPPDL. No PDTSPMS, um único veículo é designado para atender um conjunto de requisições de transportes. Essas requisições são compostas por uma localização de coleta, onde é preciso coletar um produto, e uma localização de entrega, onde o produto coletado deve ser entregue. Diferentemente do TSPPDL, o compartimento de carga no veículo é dividido em múltiplas pilhas, o que aumenta a flexibilidade do processo de carregamento/descarregamento.

Embora seja utilizado o termo "pilhas", os produtos não são propriamente empilhados (um em cima do outro), todos são armanezados na base do contêiner. As pilhas são formadas horizontalmente no contêiner, onde o fundo e a porta do contêiner representam, respectivamente, a base e o topo das pilhas.

As coletas e entregas dos produtos devem respeitar a ordem *Last-In-First-Out* (LIFO) de cada pilha, ou seja, cada produto coletado deve ser armazenado pela parte traseira do veículo em uma das pilhas e o seu descarregamento só poderá ser feito se este produto também estiver acessível a partir da traseira ("topo" de alguma das pilhas). O objetivo é encontrar uma rota com custo mínimo que atenda todas as requisições dos clientes, assegurando que restrições LIFO sejam respeitadas.

O PDTSPMS foi tratado principalmente por algoritmos exatos [Cordeau et al., 2010; Côté et al., 2012; Sampaio & Urrutia, 2016], os quais propuseram diferentes e eficientes métodos *branch-and-cut* para resolvê-lo.

2.3.3 Problema do caixeiro viajante duplo com múltiplas pilhas

O Problema do Caixeiro Viajante Duplo com Múltiplas Pilhas (*Double Traveling Salesman Problem with Multiple Stacks* - DTSPMS) foi proposto por Petersen & Madsen [2009] e foi motivado por uma necessidade do mundo real que surge no transporte de paletes, onde é preciso transportá-los de uma região para outra.

O DTSPMS é uma variação do PDTSPMS, sendo que o DTSPMS tem suas operações de coleta e de entrega completamente separadas, todas as operações de

coleta devem ser realizadas antes das operações de entrega.

Trata-se de um problema de roteamento de um único veículo (caixeiro) com coleta e entrega, classificado como um problema *one-to-one* e pertence à classe de problemas $\mathcal{NP}$-Difícil [Felipe et al., 2009]. Nesse problema um único veículo, que tem seu compartimento de carga (contêiner) dividido em pilhas de "alturas"[1] fixas, é responsável por atender todos os clientes. O veículo deve coletar todos os produtos espalhados em uma região denominada *região de coleta* e, em seguida, entregar todos esses produtos coletados em uma outra região denominada *região de entrega*. Todos os produtos têm o mesmo tamanho e mesmo formato. Os produtos são armazenados nas pilhas do veículo à medida em que são coletados, respeitando a política *Last-In-First-Out* (LIFO). Os produtos não podem sofrer realocação dentro do veículo, ou seja, uma vez que um determinado produto é armazenado no veículo ele deve permanecer imóvel até o seu descarregamento. Portanto, a ordem de visitação dos pontos da região de entrega é limitada às configurações das pilhas realizadas na coleta dos produtos, pois não é permitido realocação de produtos dentro do veículo e a política LIFO deve ser respeitada.

O DTSPMS surge no contexto em que as regiões de coleta e entrega são largamente separadas, sendo que todos os produtos na região de coleta devem ser coletados antes de qualquer descarregamento na região de entrega. O custo de transporte entre as duas regiões é fixo e não é considerado como parte do problema de otimização. O problema tem aplicações em transporte onde os produtos são carregados e descarregados pela parte traseira do veículo e a realocação dos produtos é proibida devido ao fato dos produtos serem pesados e/ou frágeis, ou que o manuseio dos produtos seja perigoso.

O objetivo do DTSPMS é encontrar dois ciclos Hamiltonianos, um para a região de coleta e outro para a região de entrega, de tal modo que a soma das distâncias percorridas em ambas as regiões sejam as mínimas possíveis, respeitando a restrição de capacidade do veículo e as restrições de precedências impostas pelas pilhas do veículo.

[1] O termo correto seria "profundidade", uma vez que, assim como no PDTSPMS, no DTSPMS as pilhas são formadas na base do contêiner. Porém, essa "profundidade" será referenciada por altura, pois assim é feito nos demais trabalhos que tratam o problema, e também por trazer mais familiaridade ao conceito de pilha. Então, quando se usar a palavra "topo", saiba que estará referindo-se à parte traseira do veículo (porta do contêiner).

A Figura 2.1 representa uma solução viável para o DTSPMS para um caso que envolve 16 produtos. Cada produto é associado a um cliente coleta e um cliente de entrega (o produto 1 está associado com o cliente de coleta 1 e com o cliente de entrega 1, produto 2 está associado com o cliente de coleta 2 e com o cliente de entrega 2, e assim por diante). O contêiner desse veículo é dividido em duas pilhas de altura oito, ou seja, o contêiner tem dimensões (2×8). O veículo sempre inicia sua rota na região de coleta no vértice 0 (depósito) e, neste exemplo, o veículo visita o cliente 15, coleta e armaneza o produto na primeira pilha do veículo, em seguida, visita o cliente 7, armazenando o produto na segunda pilha, depois o cliente 1 é visitado e tem seu produto armazenado na primeira pilha, o atendimento continua conforme é ilustrado na figura até que todos os clientes tenham sido atendidos e seus produtos armazenados no contêiner do veículo. Ao final o veículo retorna ao depósito de onde todo o contêiner é transportado para o depósito da região de coleta. Na região de entrega, o contêiner é carregado em um veículo que também sempre inicia sua rota no vértice 0 (depósito) e, neste exemplo, a partir do depósito, o veículo tem apenas dois possíveis clientes para serem visitados, uma vez que apenas os produtos dos clientes 6 e 16 estão acessíveis a partir do topo das duas pilhas. Como ilustrado na figura, o veículo inicialmente atende o cliente 16, descarregando o seu produto da segunda pilha e fazendo com que o cliente 12 esteja apto para ser atendido. O processo continua conforme é ilustrado, sempre atendendo um cliente que tem seu produto no topo de alguma das pilhas. Ao final de todos os atendimentos, o veículo retorna para o depósito na região de entrega.

Petersen & Madsen [2009] propuseram o DTSPMS, apresentaram uma formulação matemática para o problema e também propuseram quatro abordagens heurísticas para resolvê-lo: *Iterated Local Search* (ILS), *Tabu Search* (TS), *Simulated Annealing* (SA) e *Large Neighborhood Search (LNS)*. As diferentes heurísticas foram testadas para um conjunto de instâncias que se tornaram referências para futuros trabalhos. Dentre as heurísticas propostas a abordagem LNS apresentou melhor desempenho geral.

Desde que foi proposto, o DTSPMS despertou grande interessante na comunidade acadêmica, surgindo diversas abordagens exatas e heurísticas para resolvê-lo.

Felipe et al. [2009] propuseram uma abordagem heurística baseada na

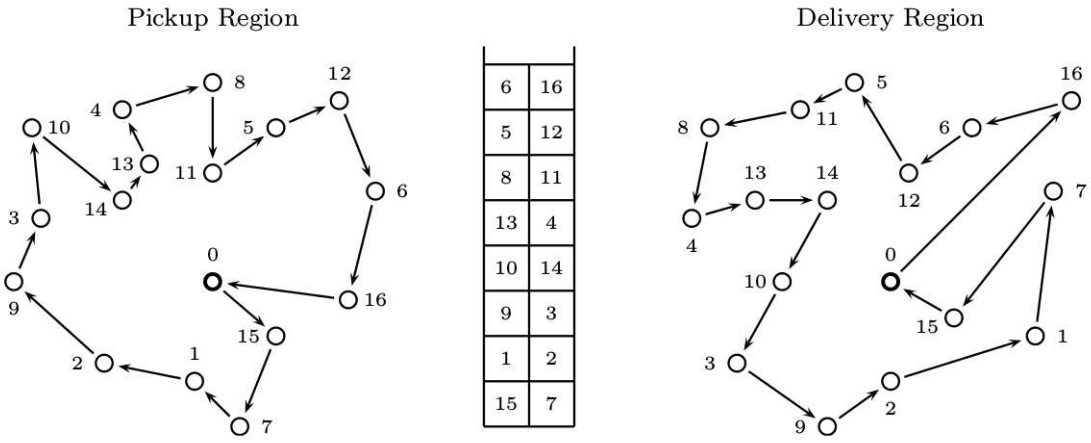


Figura 2.1: Exemplo de uma solução viável para o DTSPMS.
Fonte: Iori & Riera-Ledesma [2015].

metaheurística *Variable Neighborhood Search* (VNS), na qual foram utilizadas seis estruturas de vizinhança. Os resultados computacionais mostraram que a heurística VNS superou os resultados apresentados anteriormente por Petersen & Madsen [2009].

Petersen et al. [2010] propuseram diferentes formulações matemáticas para o DTSPMS e desenvolveram um algoritmo exato (branch-and-cut) para resolvê-lo. Analisando os resultados foi constatado que a dificuldade de uma determinada instância do problema depende não apenas do número de produtos, mas também depende fortemente da altura das pilhas.

Lusby et al. [2010] apresentaram um método de solução exata com base no *matching k-best tours* para cada uma das regiões separadamente. Esse método consiste em encontrar repetidamente soluções para os dois TSP's separados (região de entrega e região de coleta) até que um plano carregamento viável seja encontrado. Os resultados mostraram superioridade significativa dessa abordagem comparado ao método exato anterior de Petersen et al. [2010].

Carrabs et al. [2010] desenvolveram um algoritmo *branch-and-bound* (B&B) para o *Double Traveling Salesman Problem with Two Stacks* (DTSP2S), um caso especial do DTSPMS em que o veículo tem exatamente duas pilhas. Os resultados apresentados mostraram que o algoritmo B&B obteve melhor desempenho que as

outras abordagens exatas da literatura ([Lusby et al., 2010; Petersen et al., 2010]) em termos de tempo computacional e número de soluções ótimas encontradas. Segundo Carrabs et al. [2010], para ambas as abordagens exatas propostas por Lusby et al. [2010] e Petersen et al. [2010], a dificuldade de uma instância depende da capacidade das pilhas. Como a capacidade das pilhas depende do número de produtos e do número de pilhas do contêiner, o desempenho dos algoritmos melhora quando o número de pilhas aumenta. Isto é provavelmente devido ao fato de que a construção das rotas na região de coleta e na região de entrega se tornam menos restritas.

Casazza et al. [2012] estudaram as propriedades teóricas do DTSPMS, analisando a estrutura das soluções do DTSPMS em dois componentes separados: rotas e plano de carregamento. Foi demonstrado que alguns subproblemas do DTSPMS podem ser resolvidos em tempo polinominal, quando são considerados casos específicos do problema.

Alba Martínez et al. [2013] propuseram melhorias para o algoritmo *branch-and-cut* proposto por Petersen et al. [2010], adicionando novas inequações válidas (planos de cortes) que permitiram maior eficiência do algoritmo. Os resultados mostraram que o novo algoritmo superou os métodos exatos que já existiam na literatura.

A partir do DTSPMS, surgiu o DVRPMS, tratado no próximo capítulo.

Capítulo 3

Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas

O Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas (*Double Vehicle Routing Problem with Multiple Stacks* - DVRPMS) foi recentemente proposto por Iori & Riera-Ledesma [2015] com o objetivo de tratar um caso mais realístico do Problema do Caixeiro Viajante Duplo com Múltiplas Pilhas (*Double Traveling Salesman Problem with Multiple Stacks* - DTSPMS).

No DTSPMS um único veículo é responsável por atender um determinado número de clientes através de operações de coleta e entrega de produtos em duas regiões distintas, sendo que a ordem de atendimento dos clientes é condicionada ao plano de carregamento do veículo (veja a Seção 2.3.3). Já no DVRPMS, em vez de um único veículo, uma frota de veículos está disponível para atender os clientes. Essa frota de veículos pode ser heterogênea no que diz respeito às dimensões dos compartimentos de carga, isto é, os contêineres dos veículos podem ter dimensões diferentes.

Portanto, o DVRPMS é considerado uma generalização do DTSPMS. Essa generalização é motivada pelo fato de que nem sempre um veículo é capaz de atender todos os clientes, tornando assim necessário mais de um veículo, uma frota. A utilização de múltiplos veículos também é interessante mesmo que todos os produtos consigam ser transportados por um único veículo, pois a adição de mais veículos ocasiona um aumento na flexibilidade do processo de carregamento e isso pode levar a uma redução nos custos operacionais de transporte [Iori &

Riera-Ledesma, 2015].

O DVRPMS, assim como o DTSPMS, é um problema de roteamento de veículos com coleta e entrega, classificado como um problema *one-to-one*. No DVRPMS todos os produtos na região de coleta também devem ser coletados e carregados antes de qualquer descarregamento na região de entrega. O custo de transporte entre as regiões de coleta e entrega é fixo, não sendo considerado como parte do problema de otimização. O objetivo do problema é determinar rotas para a frota de veículos que atendam todos os clientes de forma que a distância total percorrida pelos veículos seja a menor possível, garantindo que a política LIFO seja respeitada em todas as pilhas de todos os veículos.

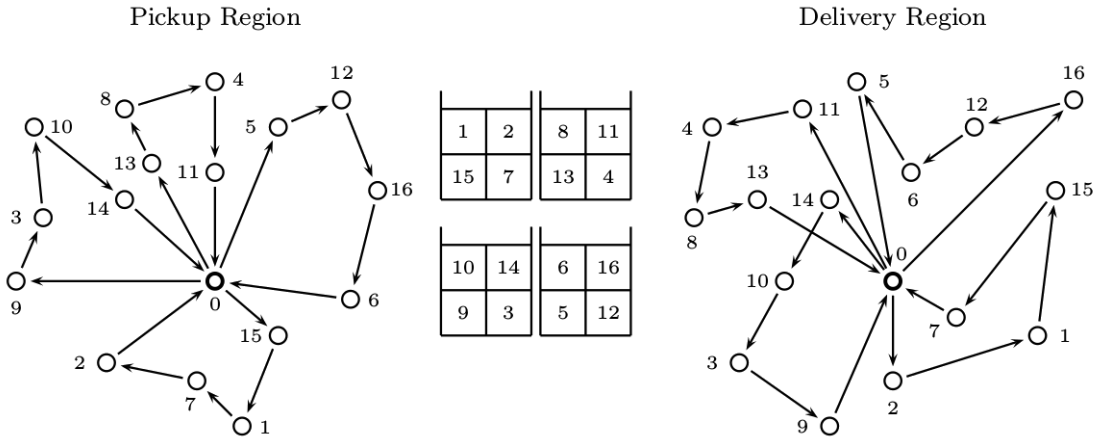


Figura 3.1: Exemplo de uma solução viável para o DVRPMS utilizando uma frota de veículos homogênea e com os contaneirs completamente preenchidos.
Fonte: Adaptado de Iori & Riera-Ledesma [2015].

As Figuras 3.1, 3.2 e 3.3 apresentam três exemplos de soluções para um caso que envolve 16 requisições de transporte, sendo cada requisição é realizada por um cliente. A Figura 3.1 representa uma solução que utiliza quatros veículos homogêneos, sendo que todos os veículos utilizados para realizar o transporte dos produtos têm os contêineres de dimensões (2×2), ou seja, duas pilhas de altura dois. A Figura 3.2 representa uma solução que utiliza três veículos heterogêneos com contêineres de dimensões (2×4), (1×4) e (2×2). A Figura 3.3 representa uma solução que utiliza quatros veículos com contêineres de dimensões (2×4), (2×4),

(2×2) e (2×2), mas diferentemente dos demais exemplos, alguns contêineres não são complementamente ocupados, pois, nesse exemplo, a capacidade dos veículos é maior que o número de clientes.

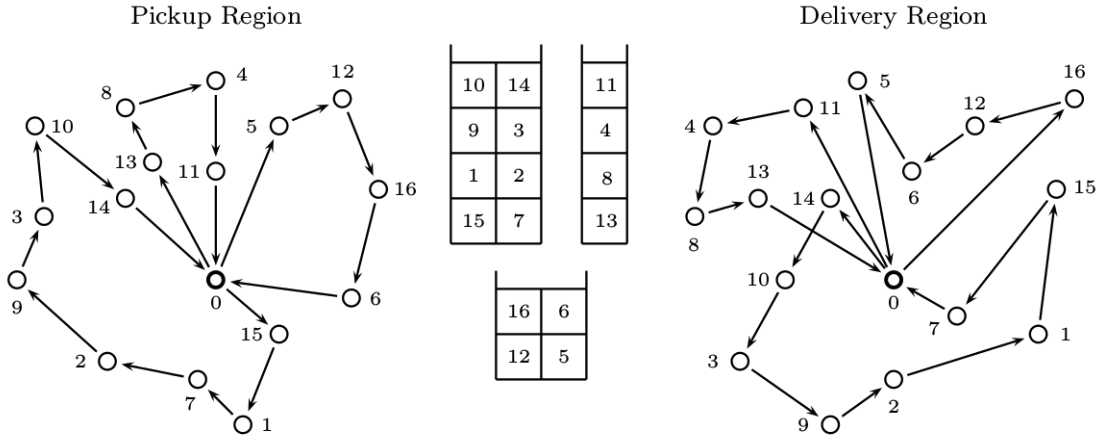


Figura 3.2: Exemplo de uma solução viável para o DVRPMS utilizando uma frota de veículos heterogênea e com os contaneirs completamente preenchidos.

Fonte: Adaptado de Iori & Riera-Ledesma [2015].

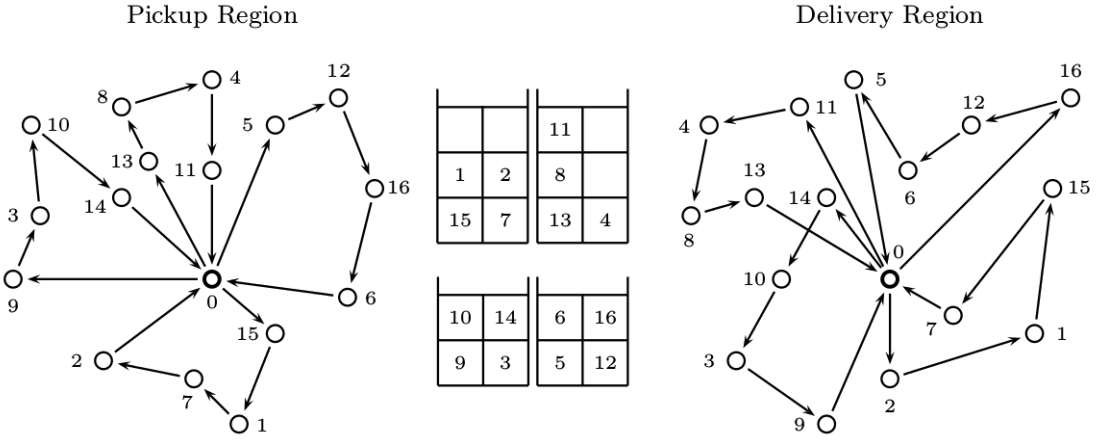


Figura 3.3: Exemplo de uma solução viável para o DVRPMS utilizando uma frota de veículos heterogênea e com alguns contaneirs não preenchidos completamente.

Fonte: Adaptado de Iori & Riera-Ledesma [2015].

Nas três diferentes soluções ilustradas pelas figuras descritas acima, a cada

veículo são associados uma rota na região de coleta, uma rota na região de entrega e um plano de carregamento para seu compartimento de carga (contêiner). As rotas e o plano de carregamento devem ser compatíveis, no sentido que seja possível atender todos os clientes, na ordem estabelecida pelas rotas, sem desrespeitar a política LIFO das pilhas.

Note que, assim como no DTSPMS, no DVRPMS é utilizado os termos "altura" e "topo" referenciando à profundidade do contêiner e à parte traseira do veículo, respectivamente.

O restante desse capítulo traz mais informações, detalhes e propostas de algoritmos para resolver o DVRPMS.

3.1 Definição formal

O DVRPMS pode ser definido por meio de um conjunto de requisições de transporte $I = \{1, 2, ..., n\}$, sendo que cada requisição $i \in I$ é realizada por um cliente. Considere $V_c^P = \{1^P, 2^P, ..., n^P\}$ como o conjunto de vértices que representam as localizações dos clientes na região de coleta e considere também $V_c^D = \{1^D, 2^D, ..., n^D\}$ como o conjunto de vértices que representam as localizações dos clientes na região de entrega. A cada requisição $i \in I$ são associados um vértice i^P na região de coleta, um vértice i^D na região de entrega e um único produto que deve ser carregado no vértice i^P, transportado da região de coleta para a região de entrega e, por fim, descarregado no vértice i^D.

Os grafos $G^P = (V^P,\ E^P)$ e $G^D = (V^D,\ E^D)$ são dois grafos completos e disjuntos entre si, onde $V^P = V_c^P \cup \{0^P\}$ e $V^D = V_c^D \cup \{0^D\}$, representam, respectivamente, os vértices na região de coleta e na região de entrega, sendo 0^P e 0^D os depósitos de cada região. O conjunto de arestas nas regiões de coleta e entrega são definidos por $E^P = \{(i^P, j^P, c_{ij}^P)\ \forall i^P \in V^P, \forall j^P \in V^P \mid j^P \neq i^P\}$ e $E^D = \{(i^D, j^D, c_{ij}^D)\ \forall i^D \in V^D, \forall j^D \in V^D \mid j^D \neq i^D\}$, onde c_{ij}^P e c_{ij}^D são, respectivamente, o custo associado às arestas (i^P, j^P) e (i^D, j^D).

Seja K o conjunto de veículos disponíveis para atender as requisições de transporte, onde o compartimento de carga (contêiner) de cada veículo $k \in K$ é dividido em R_k pilhas, sendo que todas as pilhas do veículo k têm a mesma altura L_k. Todos

os veículos do conjunto K devem iniciar e finalizar suas rotas nos depósitos definidos em cada região. Os veículos devem coletar todos os produtos dos clientes localizados na região de coleta, armazenar esses produtos em seus contêineres e depois entregar os produtos aos clientes localizados na região de entrega. As rotas de cada veículo devem satisfazer a política LIFO em todas as suas pilhas, isto é, se um cliente localizado no vértice i^P (região de coleta) é visitado antes que o cliente localizado no vértice j^P (região de coleta), e o produto da requisição j é armazenado na mesma pilha em que o produto i foi armazenado, então o cliente da requisição j deve ser visitado (vértice j^D) antes do cliente da requisição i (vértice i^D) na região de entrega.

Para simplificar a notação utilizada pela formulação matemática apresentada adiante, utilizou-se a notação descrita por Iori & Riera-Ledesma [2015], a qual utiliza i para denotar ambos os vértices i^P e i^D, e também (i, j) para denotar ambas as arestas (i^P, j^P) e (i^D, j^D).

O objetivo do DVRPMS é atender todas as $|I|$ requisições de transporte, de forma que a distância total percorrida pelos $|K|$ veículos seja a menor possível.

3.2 Trabalhos anteriores

Além de propor o DVRPMS, Iori & Riera-Ledesma [2015] formularam três diferentes abordagens exatas para sua solução: *branch-and-cut*, *branch-and-price* e *branch-and-cut-and-price*.

Iori & Riera-Ledesma [2015] limitaram o tempo de execução de cada uma de suas abordagens em uma hora, tempo suficiente para que a maioria das instâncias (menores instâncias) fossem resolvidas na otimalidade.

Até o início desta pesquisa, tinha-se conhecimento de apenas um outro trabalho que abordou o DVRPMS. Silveira et al. [2015] abordaram o problema através de métodos heurísticos, propondo três diferentes métodos para solucioná-lo: *Iterated Local Search* (ILS), *Simulated Annealing* (SA) e *Variable Neighborhood Descent* (VND). Os resultados apresentaram soluções de boa qualidade, principalmente para as instâncias de menores dimensões.

Posteriormente, como fruto desta pesquisa, foi proposto e introduzido na literatura um algoritmo heurístico baseado na metaheurística SA para o

DVRPMS [Chagas et al., 2016]. O algoritmo foi testado com as instâncias da literatura, sendo que os resultados alcançados superaram as abordagens propostas por Silveira et al. [2015], apresentando a solução ótima para a maioria das instâncias.

3.3 Formulação de programação linear inteira proposta

A formulação de programação linear inteira proposta neste trabalho para o DVRPMS foi baseada na formulação matemática do DTSPMS proposta por Petersen & Madsen [2009] e na formulação matemática para o DVRPMS proposta por Iori & Riera-Ledesma [2015]. As variáveis utilizadas são descritas abaixo:

- $x_{i,j}^{kT}$: variável binária que assume 1 se o veículo k utiliza a aresta (i, j) na região T e 0 caso contrário.
- y_{ij}^{T} : variável binária que assume 1 se o vértice i é visitado antes do vértice j na região T e 0 caso contrário.
- w_{i}^{kT} : variável binária que assume 1 se o veículo k atende a requisição i na região T e 0 caso contrário.
- z_{ir}^{k} : variável binária que assume 1 se o produto referente à requisição i é armazenado na r-ésima pilha do veículo k e 0 caso contrário.

Com essas variáveis é possível descrever a seguinte formulação de programação linear inteira para o DVRPMS:

$$\min \sum_{k \in K} \sum_{T \in \{P,D\}} \sum_{(i,j) \in E^T} c_{ij}^{T} x_{ij}^{kT} \tag{3.1}$$

$$\sum_{j \in V^T} x_{0j}^{kT} = 1 \qquad \forall k \in K, \forall\, T \in \{P, D\} \tag{3.2}$$

$$\sum_{i \in V^T} x_{i0}^{kT} = 1 \qquad \forall k \in K, \forall\, T \in \{P, D\} \tag{3.3}$$

$$\sum_{i \in V^T \setminus \{j\}} x_{ij}^{kT} = w_j^{kT} \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall j \in V_c^T \tag{3.4}$$

$$\sum_{j \in V^T \setminus \{i\}} x_{ij}^{kT} = \sum_{j \in V^T \setminus \{i\}} x_{ji}^{kT} \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall i \in V_c^T \tag{3.5}$$

$$\sum_{k \in K} w_i^{kT} = 1 \qquad \forall\, T \in \{P, D\}, \forall i \in V_c^T \tag{3.6}$$

$$\sum_{i \in I} z_{ir}^k \leq L_k \qquad \forall k \in K, \forall r = 1..R_k \tag{3.7}$$

$$\sum_{r=1}^{R_k} z_{ir}^k = w_i^{kP} \qquad \forall k \in K, \forall i \in I \tag{3.8}$$

$$w_i^{kP} = w_i^{kD} \qquad \forall k \in K, \forall i \in I \tag{3.9}$$

$$y_{ij}^T + y_{ji}^T = 1 \qquad \forall\, T \in \{P, D\}, \forall i \in V_c^T, \forall j \in V_c^T \setminus \{i\} \tag{3.10}$$

$$y_{il}^T + y_{lj}^T \leq y_{ij}^T + 1 \qquad \begin{array}{r} \forall\, T \in \{P, D\}, \forall l \in V_c^T, \\ \forall i \in V_c^T, \forall j \in V_c^T \setminus \{i\} \end{array} \tag{3.11}$$

$$x_{ij}^{kT} \leq y_{ij}^T \qquad \begin{array}{r} \forall k \in K, \forall\, T \in \{P, D\}, \\ \forall i \in V^T, \forall j \in V^T \setminus \{i\} \end{array} \tag{3.12}$$

$$y_{ij}^P + z_{ir}^k + z_{jr}^k \leq 3 - y_{ij}^D \qquad \forall k \in K, \forall i \in I, \forall j \in I \setminus \{i\}, \forall r = 1..R_k \tag{3.13}$$

$$\sum_{j \in I} j x_{0j}^{kP} \leq \sum_{j \in I} j x_{0j}^{k'P} \qquad \begin{array}{r} \forall k \in K, \forall k' \in K \mid k < k', \\ R_k = R_{k'}, L_k = L_{k'} \end{array} \tag{3.14}$$

$$x_{ij}^{kT} \in \{0, 1\} \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall (i, j) \in E^T \tag{3.15}$$

$$y_{ij}^T \in \{0, 1\} \qquad \forall\, T \in \{P, D\}, \forall i \in V^T, \forall j \in V^T \setminus \{i\} \tag{3.16}$$

$$z_{ir}^k \in \{0, 1\} \qquad \forall k \in K, \forall i \in I, \forall r = 1..R_k \tag{3.17}$$

$$w_i^{kT} \in \{0, 1\} \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall i \in V_c^T \tag{3.18}$$

A função objetivo do problema é definida pela equação (3.1), a qual minimiza a distância total percorrida pelos veículos. As restrições (3.2) e (3.3) garantem que cada veículo começa e finaliza sua rota no depósito de cada região. A restrição (3.4) assegura que a requisição j é atendida pelo veículo k somente se o veículo k chega ao vértice j. A restrição (3.5) garante que o mesmo veículo deve chegar e sair de um determinado vértice que representa a localização de um cliente. A restrição (3.6) assegura que cada requisição é atendida por um e apenas um veículo. A restrição

(3.7) assegura que a capacidade das pilhas não seja extrapolada. A restrição (3.8) garante que o produto de uma requisição atendida por um veículo k deve estar armazenado no contêiner do veículo k. A restrição (3.9) assegura que o mesmo veículo deve atender uma requisição i na região de coleta e na região de entrega. As restrições (3.10) e (3.11), respectivamente, estabelecem uma ordem de visita entre todos os pares de vértice em ambas regiões e assegura uma transitividade nesta ordem, ou seja, se i precede l e l precede j, então i precede j. A restrição (3.12) assegura que se uma aresta (i, j) é usada por algum veículo, então i é visitado antes de j. A inequação (3.13) expressa as restrições relativas à política LIFO aplicada em todas as pilhas de todos os veículos, sendo que se os produtos relativos às requisições i e j são armazenados na mesma pilha r do veículo k, sendo i visitada antes que j na região de coleta, então i não pode ser visitado antes que j na região de entrega. A restrição (3.14) quebra a simetria decorrente da formulação, impondo uma ordem lexicográfica nas rotas dos veículos com a mesma configuração de contêiner. E, finalmente, as restrições (3.15) até a (3.18) definem o escopo e domínio das variáveis de decisão.

3.4 Algoritmos heurísticos propostos

Analisando os resultados alcançados pelos métodos exatos propostos por Iori & Riera-Ledesma [2015] é possível notar que se torna impraticável resolver grandes instâncias de dados do DVRPMS através de métodos exatos, devido à alta complexidade do problema.

A fim de encontrar soluções de alta qualidade com baixo tempo computacional, foram propostas heurísticas para o DVRPMS. Tais heurísticas são descritas nesta seção, onde são detalhados os seus diferentes componentes.

3.4.1 Representação da solução

Uma solução do DVRPMS é representada pelos produtos das $|I|$ requisições de transportes, que são distribuídos e alocados nos contêineres dos $|K|$ veículos da frota.

A Figura 3.4 mostra um exemplo de representação para uma solução que envolve 16 requisições de transporte e três veículos com contêineres de dimensões

(2×4), (1×4) e (2×2). Como pode ser visto pela figura, a representação da solução informa apenas a designação das requisições de transporte e o plano de carregamento do contêiner de cada veículo. As rotas na região de coleta e da região de entrega de cada veículo são obtidas pelas funções de avaliação que são descritas na Seção 3.4.2

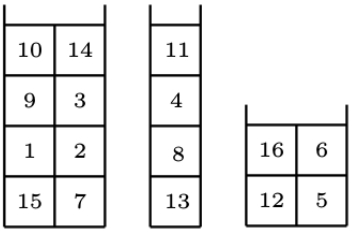


Figura 3.4: Exemplo de representação para uma solução do DVRPMS.

3.4.2 Avaliação da solução

A representação da solução proposta define apenas a designação de requisições de transporte e o plano de carregamento para cada veículo, sendo que a tarefa de determinar a rota na região de coleta e na região de entrega é feita por uma função de avaliação.

O problema de determinar a rota mais curta na região de coleta ou na região de entrega, que obedeça as restrições LIFO impostas por um plano de carregamento conhecido, pode ser visto como o Problema do Caixeiro Viajante com Restrições de Precedência (*Traveling Salesman Problem with Precedence Constraints* - TSPPC) [Savelsbergh & Sol, 1995], onde as restrições de precedências são dadas pelo plano de carregamento.

Segundo Moon et al. [2002], o TSPPC pertence à classe de problemas $\mathcal{NP}$-Difícil, sendo assim a solução ótima para o problema não pode ser obtida dentro de um tempo computacional razoável quando são consideradas grandes instâncias de dados.

Contudo, foram propostas três diferentes estratégias para avaliar as soluções do DVRPMS, sendo duas estratégias exatas, ou seja, as rotas mais curtas que

obedecem as restrições de carregamento LIFO impostas pelo plano de carregamento são encontradas, e uma estratégia heurística que não garante encontrar as melhores rotas, mas tem um custo computacional menor quando comparado aos métodos exatos.

As três estratégias propostas para avaliar uma solução do DVRPMS são descritas a seguir, onde é detalhado apenas como obter as rotas realizadas pelo k-ésimo veículo. A fim de obter as rotas de todos os $|K|$ veículos, é necessário aplicar a mesma estratégia uma vez para cada veículo $k \in K$.

3.4.2.1 Formulação de programação linear inteira mista - MILP

A formulação matemática proposta para encontrar a rota de coleta (entrega) de menor distância percorrida pelo veículo k, dado o seu plano de carregamento, é basicamente a formulação de programação inteira mista (*Mixed Integer Linear Programming* - MILP) para o TSP proposta por Miller et al. [1960], com a adição de uma restrição para garantir que a política LIFO seja respeitada.

A fim de descrever a formulação matemática, consideramos p_k^{rl} como sendo o produto referente à l-ésima requisição (de baixo para cima) armazenada na r-ésima pilha (da esquerda para a direita) do contêiner do veículo k. Seja V_k^P e V_k^D os conjuntos de vértices que representam, respectivamente, as localizações na região de coleta e na região de entrega dos clientes que são atendidos pelo veículo k.

Assim como foi definido por Miller et al. [1960] para o TSP, as seguintes variáveis de decisão são utilizadas:

- x_{ij}: variável binária que assume 1 se a aresta (i, j) é usada e 0 caso contrário.
- u_i: variável contínua que indica a ordem em que a requisição i é atendida pelo veículo.

Utilizando essas variáveis e conhecendo o plano de carregamento do veículo k, é possível descrever uma formulação MILP que obtém a rota mais curta possível feita pelo veículo na região de coleta. Essa formulação é descrita a seguir:

$$\min \sum_{i \in V_k^P \cup \{0^P\}} \sum_{j \in V_k^P \cup \{0^P\}} c_{ij}^P x_{ij} \tag{3.19}$$

$$\sum_{i \in V_k^P \cup \{0^P\}} x_{ij} = 1 \qquad \forall j \in V_k^P \cup \{0^P\} \tag{3.20}$$

$$\sum_{j \in V_k^P \cup \{0^P\}} x_{ij} = 1 \qquad \forall i \in V_k^P \cup \{0^P\} \tag{3.21}$$

$$u_i - u_j + 1 \leq |V_k^P| (1 - x_{ij}) \qquad \forall i \in V_k^P \cup \{0^P\}, \forall j \in V_k^P \tag{3.22}$$

$$u_{p_k^{rl}} \leq u_{p_k^{rl+1}} \qquad \forall r = 1..R_k, \forall l = 1..L_k\text{-}1 \tag{3.23}$$

$$u_i \geq 0 \qquad \forall i \in V_k^P \cup \{0^P\} \tag{3.24}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in V_k^P \cup \{0^P\}, \forall j \in V_k^P \cup \{0^P\} \tag{3.25}$$

A Figura 3.5 ilustra a ideia da restrição (3.23). Para todo par de produtos (a, b) que são empilhados consecutivamente na mesma pilha, sendo que b é empilhado em cima de a, a restrição (3.23) garante que a rota de coleta visitará primeiramente o cliente do produto a antes de visitar o cliente do produto b.

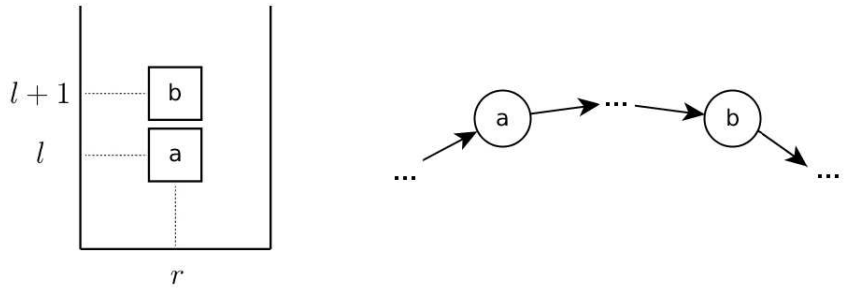


Figura 3.5: Detalhamento da restrição de carregamento LIFO na região de coleta

De maneira análoga, a seguinte formulação MILP obtém a rota mais curta feita pelo veículo k na região de entrega a partir do seu plano de carregamento:

$$\min \sum_{i \in V_k^D \cup \{0^D\}} \sum_{j \in V_k^D \cup \{0^D\}} c_{ij}^D x_{ij} \tag{3.26}$$

$$\sum_{i \in V_k^D \cup \{0^D\}} x_{ij} = 1 \qquad \forall j \in V_k^D \cup \{0^D\} \tag{3.27}$$

$$\sum_{j \in V_k^D \cup \{0^D\}} x_{ij} = 1 \qquad \forall i \in V_k^D \cup \{0^D\} \tag{3.28}$$

$$u_i - u_j + 1 \leq |V_k^D| (1 - x_{ij}) \qquad \forall i \in V_k^D \cup \{0^D\}, \forall j \in V_k^D \tag{3.29}$$

$$u_{p_k^{r,l}} \leq u_{p_k^{r,l-1}} \qquad \forall r = 1..R_k, \forall l = 2..L_k \tag{3.30}$$

$$u_i \geq 0 \qquad \forall i \in V_k^D \cup \{0^D\} \tag{3.31}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in V_k^D \cup \{0^D\}, \forall j \in V_k^D \cup \{0^D\} \tag{3.32}$$

A Figura 3.6 ilustra a ideia da restrição (3.30). Para todo par de produtos (a, b) que são empilhados consecutivamente na mesma pilha, sendo que b é empilhado em cima de a, a restrição (3.23) garante que a rota de coleta visitará primeiramente o cliente do produto b antes de visitar o cliente do produto a.

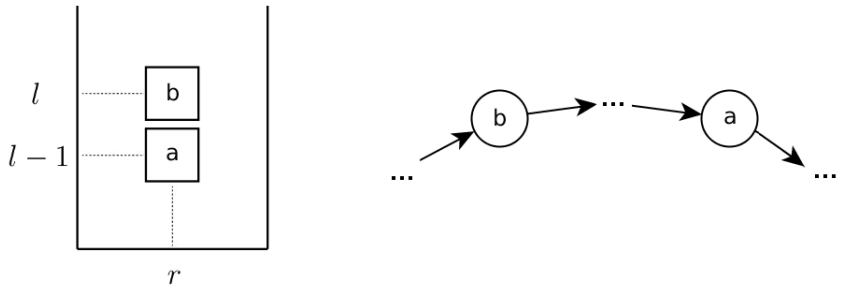


Figura 3.6: Detalhamento da restrição de carregamento LIFO na região de entrega

A função objetivo (3.19) e a função objetivo (3.26) minimizam, respectivamente, a distância percorrida pelo veículo k na região de coleta e na região de entrega. A restrição (3.20) garante que cada cliente na região de coleta será atendido exatamente uma vez, e a restrição (3.28) garante o mesmo comportamento na região de entrega. As restrições (3.22) e (3.29) asseguram a não formação de subciclos nas rotas realizadas na região de coleta e na região de entrega, respectivamente. As restrições (3.23) e (3.30) garantem, respectivamente, que a rota na região de coleta e a rota na região entrega não violam a política LIFO das

pilhas do veículo k. E, finalmente, as restrições (3.24), (3.25), (3.31) e (3.32) definem o escopo e domínio das variáveis de decisão para cada formulação matemática.

3.4.2.2 Branch-and-bound - B&B

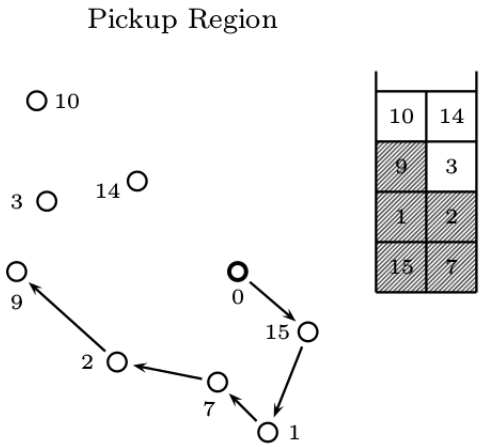
A segunda estratégia exata proposta para encontrar a rota de coleta (entrega) de menor distância percorrida pelo veículo k, dado o seu plano de carregamento, é realizada por um algoritmo *branch-and-bound* (B&B).

O algoritmo proposto trabalha com uma sequência $\mathcal{S}$ que representa a ordem em que os vértices da região de coleta (entrega) são visitados pelo veículo k. Uma sequência $\mathcal{S}$ que representa de forma completa a rota realizada pelo veículo k deve começar com o vértice que representa o depósito 0^P (0^D), seguido por uma permutação dos vértices que representam as localizações dos clientes na região de coleta (entrega), desde que essa permutação respeite as restrições da política LIFO determinadas pelo plano de carregamento do veículo. E, por fim, a sequência $\mathcal{S}$ deve terminar com o vértice 0^P (0^D).

Qualquer nó $\mathcal{N}$ da árvore de pesquisa do algoritmo B&B corresponde a uma rota parcial (ou completa, no caso das folhas da árvore). O *lower-bound* do nó $\mathcal{N}$ ($\text{LB}_{\mathcal{N}}$) é calculado como sendo a distância total da rota já construída (sequência $\mathcal{S}$) mais a distância entre o último vértice adicionado à rota (último vértice adicionado à sequência $\mathcal{S}$) e o vértice que representa o depósito. Se o $\text{LB}_{\mathcal{N}}$ for maior ou igual à melhor solução conhecida (*upper-bound* (UB)), o nó $\mathcal{N}$ é podado. Caso contrário, o algoritmo executa a ramificação neste nó.

Inicialmente, a melhor solução conhecida (UB) é a solução obtida através de uma heurística (descrita na Seção 3.4.2.3). No nó raiz da árvore de pesquisa, a sequência $\mathcal{S}$ começa com apenas o vértice 0^P (0^D). A ramificação (*branching*) realizada a partir de um nó $\mathcal{N}$ da árvore de pesquisa é feita para todos os vértices acessíveis a partir do último vértice adicionado à sequência $\mathcal{S}$ do nó $\mathcal{N}$ ($\mathcal{S}_{\mathcal{N}}$). Note que, não é necessário avaliar novamente as restrições da política LIFO envolvendo as requisições que já pertencem à sequência $\mathcal{S}$, pois essas já foram adicionadas à rota de forma a não desconsiderar as restrições de carregamento impostas pelo contêiner do veículo. A estratégia *depth-first* foi utilizada para construir/percorrer a árvore de pesquisa do algoritmo B&B.

A Figura 3.7 ilustra a operação de ramificação (*branching*) a partir de um nó $\mathcal{N}$, que corresponde a uma rota de coleta parcial formada pela sequência $\mathcal{S}_{\mathcal{N}}$. Os produtos tracejados no contêiner indicam que eles já foram analisados e inseridos na rota pelas iterações precedentes. O algoritmo realiza um *branching* para cada vértice acessível (10^P e 3^P) a partir do último vértice da sequência $\mathcal{S}_{\mathcal{N}}$ (9^P).



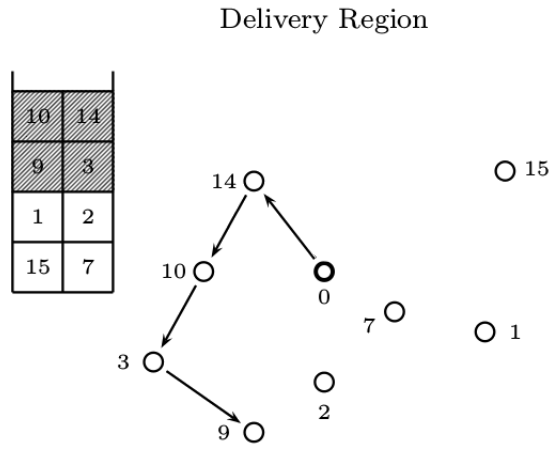
$$\mathcal{S}_{\mathcal{N}} = [0^P, 15^P, 1^P, 7^P, 2^P, 9^P]$$

$$\mathcal{S}_{\mathcal{N}_1} = [0^P, 15^P, 1^P, 7^P, 2^P, 9^P, 10^P] \quad \mathcal{S}_{\mathcal{N}_2} = [0^P, 15^P, 1^P, 7^P, 2^P, 9^P, 3^P]$$

Figura 3.7: Detalhamento do processo de *branching* realizado pelo algoritmo B&B para determinar a melhor rota na região de coleta.

A Figura 3.8 também ilustra a operação de ramificação (*branching*) a partir de um nó $\mathcal{N}$, porém considerando agora a rota de entrega. O algoritmo realiza um *branching* para cada vértice acessível (1^D e 2^D) a partir do último vértice da sequência $\mathcal{S}_{\mathcal{N}}$ (9^D).

A complexidade para se obter a rota na região de coleta e na região de entrega realizada pelo veículo k por meio do algoritmo exato B&B é dada por $O((R_k L_k)^{R_k})$, pois o algoritmo realiza para cada uma das requisições designadas ao veículo k,



$$\mathcal{S}_{\mathcal{N}} = [0^D, 14^D, 10^D, 3^D, 9^D]$$

$$\mathcal{S}_{\mathcal{N}_1} = [0^D, 14^D, 10^D, 3^D, 9^D, 1^D] \quad \mathcal{S}_{\mathcal{N}_2} = [0^D, 14^D, 10^D, 3^D, 9^D, 2^D]$$

Figura 3.8: Detalhamento do processo de *branching* realizado pelo algoritmo B&B para determinar a melhor rota na região de entrega.

isto é, $R_k L_k$ vezes[1], um *branching*, sendo que cada *branching* tem um fator de ramificação igual ao número de pilhas[2], isto é, R_k.

Através dos resultados descritos na Seção 3.5.2, é possível notar que algoritmo B&B se mostrou mais eficiente que a formulação MILP ao encontrar a melhor rota de coleta e de entrega em todos as diferentes configurações de contêineres descritos na literatura para o DVRPMS. Portanto, a partir de agora, quando se pretender obter as melhores rotas de um dado contêiner, será utilizado o algoritmo B&B.

3.4.2.3 Heurística gulosa com busca local - GH+LS

Devido à alta complexidade de se determinar as rotas ótimas a partir de um plano de carregamento, foi proposta uma abordagem heurística com o objetivo de

[1]Considerando que o contêiner k estará completamente preenchido.

[2]Por motivo de simplificação, foi desconsiderado que após alguma pilha ser totalmente analisada, o fator de ramificação do algoritmo diminui.

encontrar rotas de boa qualidade gastando pouco tempo computacional.

A abordagem heurística pode ser dividida em duas fases que são realizadas subsequentemente: uma fase gulosa e uma fase de busca local.

A fase gulosa consiste em construir a rota de coleta (entrega) analisando a cada momento dentre os produtos da base (topo) de cada pilha aquele que causará o menor impacto no custo da rota, ou seja, a cada momento a rota de coleta (entrega) é construída de forma que o produto coletado (entregue) seja o produto mais próximo da rota parcialmente construída.

As Figuras 3.9 e 3.10 ilustram o funcionamento da fase gulosa da heurística, através do detalhamento de uma iteração do procedimento na região de coleta e na região de entrega, respectivamente. Os produtos tracejados nos contêineres de cada figura indicam que eles já foram analisados e inseridos nas respectivas rotas pelas iterações precedentes. Na situação mostrada pela Figura 3.9 o último vértice inserido na rota de coleta é o vértice 9^P referente à requisição 9, e a partir desse vértice apenas os vértices 3^P e 10^P da região de coleta, associados às requisições 3 e 10, são acessíveis, uma vez que as restrições do plano de carregamento devem ser satisfeitas. Entre o vértice 3^P e 10^P será escolhido o vértice 3^P, pois este está mais próximo do vértice 9^P. Na Figura 3.10 o último vértice inserido na rota de entrega é o vértice 9^D referente à requisição 9, e a partir desse vértice apenas os vértices 1^D e 2^D da região de entrega, associados às requisições 1 e 2, são acessíveis, sendo que o vértice 2^D será escolhido, pois este está mais próximo do vértice 9^D.

A complexidade para se obter a rota na região de coleta e na região de entrega realizada pelo veículo k por meio da fase gulosa é dada por $O(R_k^2 L_k)$, pois é preciso decidir em $R_k L_k$ vezes (número de requisições designadas ao veículo k)[3] qual dos R_k produtos[4] da base (topo) das pilhas está mais próximo do último vértice da parcialmente construída[5].

[3]Por motivo de simplificação da notação, foi considerado que o contêiner k estará completamente preenchido. Contudo, caso o contêiner k não esteja totalmente ocupado, a complexidade também será válida, uma vez que $O(R_k^2 L_k)$ é um limite superior para esse procedimento.

[4]Por motivo de simplificação, foi desconsiderado que após alguma pilha ser totalmente analisada, a complexidade de se encontrar o produto mais próximo à rota diminui.

[5]Também pode ser implementado com a complexidade $O(R_k L_k \log_2 R_k)$, se for utilizada uma estrutura eficiente (heap) para descobrir qual produto está localizado mais próximo do último vértice adicionado à rota parcialmente construída.

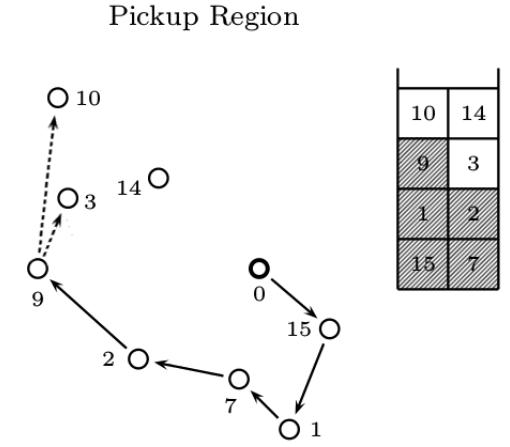


Figura 3.9: Detalhamento da fase gulosa, aplicada na região de coleta, da heurística utilizada na avaliação das soluções do DVRPMS.

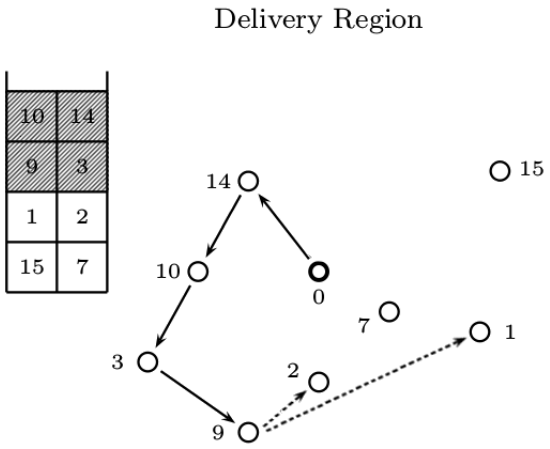
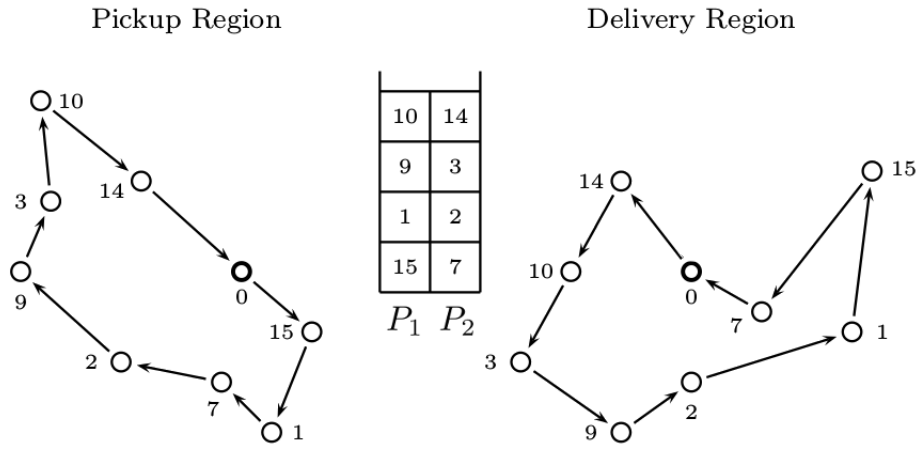


Figura 3.10: Detalhamento da fase gulosa, aplicada na região de entrega, da heurística utilizada na avaliação das soluções do DVRPMS.

Note que a rota realizada por um veículo na região de coleta e na região de entrega pode ser representada através da ordem que as operações são realizadas nas pilhas do contêiner. A Figura 3.11 exemplifica essa afirmação, onde as duas pilhas do contêiner de dimensões (2×4) foram nomeadas por P_1 e P_2 para facilitar o referenciamento de cada pilha. As sequências S^P e S^D representam, respectivamente, a rota realizada na região de coleta e a rota realizada na região de entrega através das operações realizadas nas pilhas do veículo.



Pickup route: $S^P = \{P_1, P_1, P_2, P_2, P_1, P_2, P_1, P_2\}$

Delivery route: $S^D = \{P_2, P_1, P_2, P_1, P_2, P_1, P_1, P_2\}$

Figura 3.11: Detalhamento da fase de busca local da heurística utilizada na avaliação das soluções do DVRPMS.

A fase de busca local consiste em aplicar um algoritmo de refinamento nas rotas determinadas pela fase gulosa descrita acima. Nessa fase é feita uma busca local separadamente em cada uma das sequências S^P e S^D. A estrutura de vizinhança definida para realizar a busca local consiste em trocar duas operações diferentes de ordem. A Figura 3.12 mostra uma representação vizinha $S^{P'}$ a partir da representação S^P (Figura 3.11), bem como a alteração provocada na rota de coleta.

Na busca local, a exploração dos vizinhos é feita de forma casual (vizinho é escolhido aleatoriamente dentre os vizinhos da estrutura de vizinhança) e a vizinhança da sequência atual é explorada até que seja encontrada uma sequência que representa uma rota mais curta que a rota já conhecida ou até que todos os vizinhos sejam explorados (critério de parada).

Definidas todas as três estratégias propostas para avaliar as soluções, a seguir são detalhadas as estruturas de vizinhança aplicadas propriamente às soluções do DVRPMS.

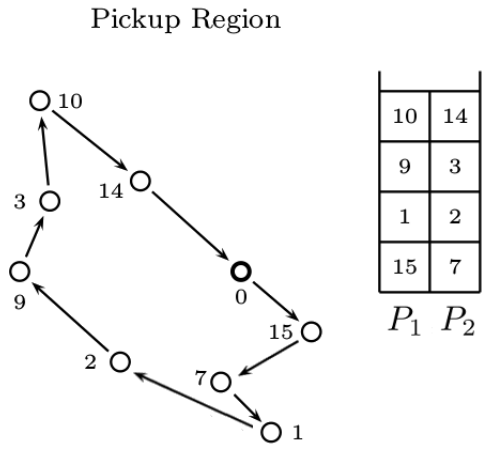


Figura 3.12: Exemplo de vizinho utilizado na fase de busca local da heurística utilizada na avaliação das soluções do DVRPMS.

3.4.3 Estruturas de vizinhança

Foram definidas quatro estruturas de vizinhança para explorar o espaço de soluções do DVRPMS: troca de produtos (TP), cadeia de ejeção sobre os produtos (CP), permutação de pilha (PP) e troca de pilhas (TS). Essas estruturas são utilizadas na fase de busca local e na geração de vizinhos aleatórios, necessários para o funcionamento dos algoritmos heurísticos propostos. Todos os detalhes das quatro estruturas de vizinhança são descritos a seguir.

3.4.3.1 Troca de produtos

A vizinhança *troca de produtos* (TP) de uma solução do DVRPMS contém as soluções que podem ser obtidas trocando dois produtos quaisquer dos contêineres dos veículos. Os produtos a serem realocados podem pertencer ao mesmo contêiner ou a contêineres diferentes, sendo assim o tamanho da estrutura de vizinhança é dado pela combinação do número total de produtos $|I|$ agrupados de 2 a 2, isto é, $O\left(\left(\begin{array}{c} |I| \\ 2 \end{array}\right)\right) = O(|I|^2)$. A Figura 3.13 mostra um exemplo de uma solução s e uma de suas soluções vizinhas s' a partir da estrutura de vizinhança TP.

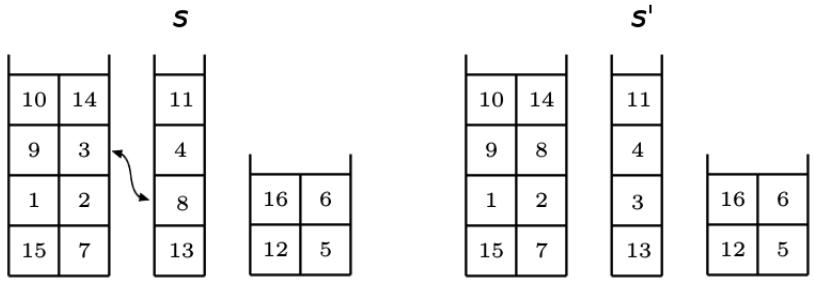


Figura 3.13: Exemplo de um vizinho de uma solução do DVRPMS utilizando a estrutura de vizinhança *troca de produtos* (TP).

3.4.3.2 Cadeia de ejeção sobre os produtos

A vizinhança *cadeia de ejeção sobre os produtos* (CP) de uma solução do DVRPMS contém as soluções que podem ser obtidas trocando três produtos quaisquer dos contêineres por meio de uma cadeia de ejeção, isto é, o primeiro produto é realocado para a posição do segundo, o segundo produto é realocado para a posição do terceiro e o terceiro produto é realocado para a posição do primeiro produto. Como os produtos a serem realocados podem pertencer ao mesmo contêiner ou a contêineres diferentes, e a ordem que os produtos são escolhidos para aplicar a cadeia de ejeção é importante, sendo assim, o tamanho da vizinhança CP é dado por $O\left(\frac{|I|\times(|I|-1)\times(|I|-2)}{3}\right) = O(|I|^3)$. A Figura 3.14 mostra um exemplo de uma solução s e uma de suas soluções vizinhas s' a partir da estrutura de vizinhança CP.

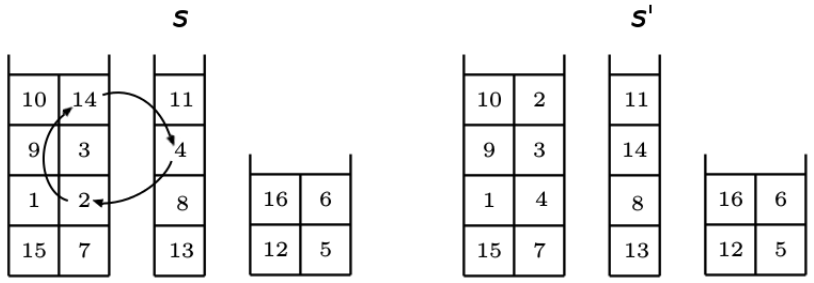


Figura 3.14: Exemplo de um vizinho de uma solução do DVRPMS utilizando a estrutura de vizinhança *cadeia de ejeção sobre os produtos* (CP).

3.4.3.3 Permutação de pilha

A vizinhança *permutação de pilha* (PP) de uma solução do DVRPMS contém as soluções que podem ser obtidas por uma permutação da ordem em que os produtos de uma mesma pilha foram armazenados. Como o número total de pilhas de uma solução é dado por $\sum_{k \in K} R_k$ e cada pilha tem L_k produtos e, consequentemente, $L_k!$ permutações[6], o tamanho da vizinhança PP é dado por $O\left(\sum_{k \in K} L_k!\, R_k\right)$. A Figura 3.15 mostra um exemplo de uma solução s e uma de suas soluções vizinhas s' a partir da estrutura de vizinhança PP.

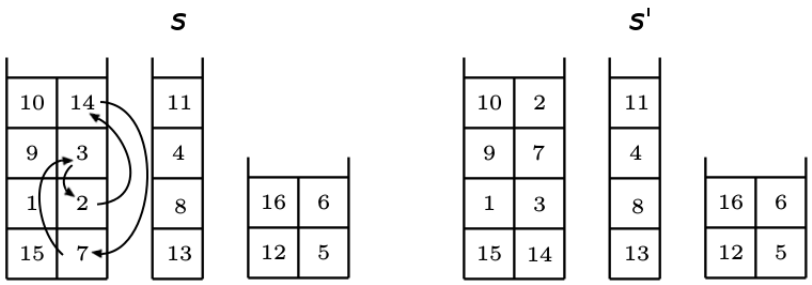


Figura 3.15: Exemplo de um vizinho de uma solução do DVRPMS utilizando a estrutura de vizinhança *permutação de pilha* (PP).

3.4.3.4 Troca de pilhas

A vizinhança *troca de pilhas* (TS) de uma solução do DVRPMS contém as soluções que podem ser obtidas por uma troca de duas pilhas de contêineres diferentes. As pilhas a serem realocadas devem pertencer a diferentes contêineres, então o tamanho da vizinhança (TS) é dado pela combinação do número total de pilhas $\sum_{k \in K} R_k$ agrupados de 2 a 2, desconsiderando os pares de pilhas que pertencem ao mesmo veículo, isto é, $O\left(\left(\begin{array}{c}\sum_{k \in K} R_k \\ 2\end{array}\right) - \sum_{k \in K}\left(\begin{array}{c}R_k \\ 2\end{array}\right)\right)$. A Figura 3.16 mostra um exemplo de uma solução s e uma de suas soluções vizinhas s' a partir da estrutura de vizinhança TS.

Para duas pilhas com tamanhos diferentes, a troca é feita apenas entre os produtos da menor pilha e os produtos carregados nas primeira posições (de baixo

[6]Por motivo de simplicação da notação, foi considerado que as pilhas estarão completamente preenchidas.

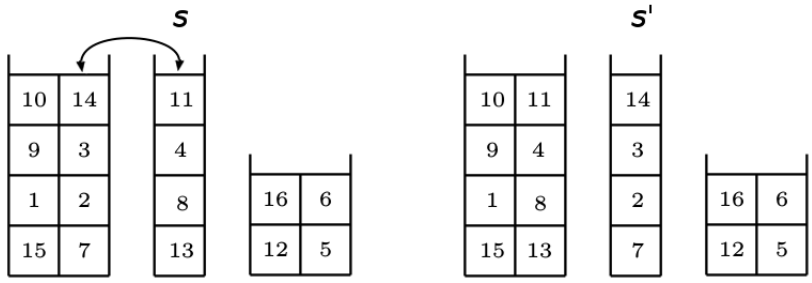


Figura 3.16: Exemplo de um vizinho de uma solução do DVRPMS utilizando a estrutura de vizinhança *troca de pilhas* (TS).

para cima) da maior pilha. A Figura 3.17 mostra um exemplo para esta situação.

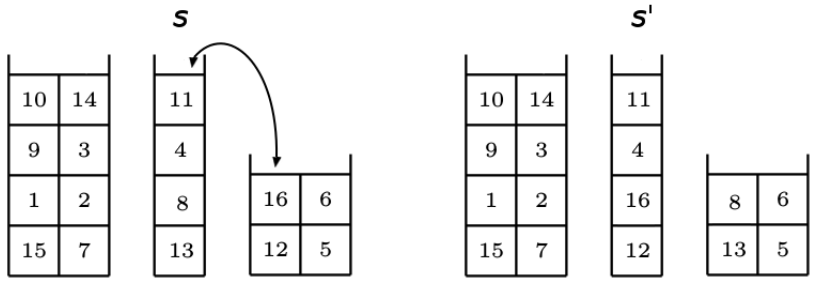


Figura 3.17: Exemplo de um vizinho de uma solução do DVRPMS utilizando a estrutura de vizinhança *troca de pilhas* (TS) para pilhas de tamanho diferentes.

3.4.4 Solução inicial

A solução inicial de todos os algoritmos heurísticos propostos neste trabalho para o DVRPMS é criada de forma aleatória. Para cada veículo é gerado aleatoriamente um plano de carregamento, de modo que cada requisição de transporte seja atendida por um único veículo.

3.4.5 Busca local

A busca local é um procedimento iterativo de melhoria aplicado a uma solução. O procedimento consiste em explorar os vizinhos de uma solução atual em busca de encontrar uma solução vizinha que apresente melhor qualidade. Se uma solução

melhor for encontrada, essa solução se torna a solução atual e o processo é repetido até que não seja possível melhorar essa solução, ou seja, até encontrar um ótimo local.

Segundo Hertz & Widmer [2003], o procedimento de busca local realiza um único caminho em um grafo direcionado $G = (S, A)$, onde o conjunto de vértices S representa o conjunto de soluções do problema em questão e o conjunto de arestas A representa todas as transições entre soluções e seus vizinhos, ou seja, um arco $(s, s') \in A$, se e somente se, s' for uma solução vizinha à solução s.

Portanto, o procedimento de busca local pode ser considerado relativamente simples, não sendo capaz de encontrar boas soluções para problemas de otimização combinatória que apresentam grande número de possíveis soluções, uma vez que apenas um caminho no espaço de soluções é explorado.

Contudo, os procedimentos de busca local apresentam enorme importância, sendo utilizados como subrotinas de diversas metaheurísticas tais como: *Iterated Local Search* (ILS) [Lourenço et al., 2003], *Iterated Greedy* (IG) [Ruiz & Stützle, 2007], *Greedy Randomized Adaptive Search* (GRASP) [Feo & Resende, 1995], *Tabu Search* (TS) [Glover, 1989, 1990], *Variable Neighborhood Search* (VNS) [Mladenović & Hansen, 1997], entre outras.

Neste trabalho foi desenvolvida uma busca local para cada uma das quatro estruturas de vizinhança definidas para o DVRPMS. Tais buscas locais são utilizadas como subrotinas dos algoritmos SA_1, VNS e ILS propostos e descritos, respectivamente, nas Seções 3.4.6, 3.4.8 e 3.4.9. Em todas as quatro buscas locais, a exploração dos vizinhos é feito de forma determinística, sendo que a vizinhança de uma solução atual é explorada até que uma solução melhor seja encontrada (*first improvement*).

Todas as soluções exploradas pelos procedimentos de busca local dos algoritmos VNS e ILS são avaliadas utilizando a função de avaliação heurística GH+LS descrita na Seção 3.4.2.3. Já para o algoritmo SA_1, é utilizada a função de avaliação exata B&B descrita na Seção 3.4.2.2. Tais decisões foram tomadas com o objetivo de tentar equilibrar a qualidade das soluções obtidas e a eficiência dos algoritmos. Nos decorrer deste capítulo essas decisões são justificadas.

3.4.6 Simulated annealing [Chagas et al., 2016]

Os primeiros resultados alcançados neste trabalho para o DVRPMS foram obtidos através de um método heurístico baseado na metaheurística *Simulated Annealing* (SA). Tais resultados foram apresentados na *19th International Conference on Intelligent Transportation Systems* (ITSC 2016) e podem ser consultados em [Chagas et al., 2016].

A *Simulated Annealing* (SA) é uma metaheurística probabilística utilizada para resolver problemas de otimização combinatória que foi proposta simultaneamente por Kirkpatrick et al. [1983] e Černỳ [1985]. Desde que a SA foi proposta, vários trabalhos foram publicados aplicando-a para resolver problemas combinatórios, principalmente, problemas de roteamento de veículos [Osman, 1993; Van Breedam, 1995; Chiang & Russell, 1996; Czech & Czarnas, 2002; Vincent et al., 2010].

A metaheurística SA recebeu esse nome devido à analogia entre uma operação de termodinâmica que simula o resfriamento de um conjunto de átomos aquecidos, operação conhecida como recozimento (*annealing*), e a resolução de problemas combinatórios. A operação *annealing* começa com uma temperatura elevada e é lentamente reduzida até que o sólido chegue ao seu estado de equilíbrio. De forma análoga, uma solução segue o comportamento dos átomos, inicialmente os átomos estão em grande agitação (solução sofre grandes perturbações), mas à medida que a temperatura diminui, os átomos diminuem a agitação (solução sofre pequenas perturbações) até um estado de equilíbrio (solução final).

A metaheurística SA permite que a solução atual seja piorada. A decisão de aceitar uma solução pior do que a solução atual é baseada em uma probabilidade dada em função da qualidade da solução atual e da nova solução, e também em função da temperatura vigente. Quanto maior a temperatura e menor a diferença entre o valor da nova solução e o valor da solução atual, maior a chance de aceitar uma solução pior.

O Algoritmo 1, referido como SA_1, descreve os passos realizados pela metaheurística SA proposta para o DVRPMS. A solução inicial do algoritmo é gerada aleatoriamente, escolhendo casualmente um plano de carregamento para cada veículo. Enquanto os critérios de parada não são satisfeitos, o algoritmo seleciona

Algoritmo 1: Simulated Annealing [Chagas et al., 2016] (SA_1)

```
1  s ← gerar uma solução de forma aleatória
2  melhor_solucao ← s
3  T_atual ← T_i
4  while T_atual ≥ T_f do
5  |  it ← 0
6  |  while it < N do
7  |  |  s' ← gerar aleatoriamente uma solução vizinha s' ∈ N(s)
8  |  |  Δ ← f_MIX(s') − f_MIX(s)
9  |  |  if Δ < 0 or random(0, 1) < e^{−Δ/T_atual} then
10 |  |  |  s ← s'
11 |  |  |  if f_MIX(s') < f_MIX(melhor_solucao) then
12 |  |  |  |  melhor_solucao ← s'
13 |  |  |  end
14 |  |  end
15 |  |  it ← it + 1
16 |  end
17 |  T_atual ← T_atual × (1 − α)
18 end
19 melhor_solucao ← aplicar a busca local na solução melhor_solucao // f_B&B
20 return melhor_solucao
```

aleatoriamente uma solução vizinha a partir da solução atual usando a estrutura de vizinhança *troca de produtos* (TP). O critério para aceitar uma solução como a solução atual é feito em função de uma probabilidade determinada de acordo com a qualidade da solução e com a temperatura atual. Todas as soluções exploradas entre as linhas 1 e 18 do algoritmo são avaliadas pela função $f_{MIX}()$, que obtém a distância total percorrida por todos os veículos, sendo que as rotas para os contêineres de dimensões (2×2) e (2×3) são obtidas utilizando a avaliação exata B&B (descrita na Seção 3.4.2.2) e para os demais contêineres as rotas são obtidas utilizando a primeira fase (gulosa) da avaliação heurística GH+LS (descrita na Seção 3.4.2.3).

A decisão de avaliar apenas os contêineres de dimensões (2×2) e (2×3) de forma exata durante a fase interna do algoritmo SA_1 foi tomada, visto que, para os demais contêineres, a aplicação do algoritmo de avaliação exata B&B tomava

muito tempo.

No final do algoritmo (linha 19), uma busca local é aplicada na melhor solução encontrada, sendo que todos os contêineres (independente das dimensões) das soluções exploradas por essa busca local são avaliadas utilizando a função de avaliação exata B&B. O objetivo dessa busca local é buscar atingir um mínimo local que talvez não fora encontrado anteriormente devido a possibilidade de não obtenção das melhores rotas de cada contêiner.

O fato de ter sido utilizado somente a estrutura de vizinhança TP e de se avaliar alguns tipos de contêineres apenas pela primeira fase da heurística de avaliação GH+LS são justificados porque até a data de submissão do trabalho para a *19th International Conference on Intelligent Transportation Systems*, as outras estruturas de vizinhança e também a segunda fase da heurística de avaliação ainda não tinham sido propostas.

Note que o algoritmo SA_1 apresenta quatro parâmetros: temperatura inicial (T_i), temperatura final (T_f), número de iterações em cada temperatura (N) e fator de resfriamento (α). A fim de obter o melhor desempenho do algoritmo todos os quatros parâmetros foram calibrados (veja a Seção 3.5.3).

3.4.7 Simulated annealing

A segunda metaheurística proposta para o DVRPMS também é baseada na técnica SA, diferindo basicamente na forma que as soluções são avaliadas, quando comparado à metaheurística proposta por Chagas et al. [2016] (Algoritmo SA_1).

O Algoritmo 2 descreve os passos realizados pela segunda metaheurística SA proposta para o DVRPMS. A solução inicial do algoritmo é gerada aleatoriamente, escolhendo casualmente um plano de carregamento para cada veículo. Enquanto os critérios de parada não são satisfeitos, o algoritmo seleciona aleatoriamente uma estrutura de vizinhança a partir do conjunto de estruturas de vizinhança φ (a probabilidade de escolher cada estrutura de vizinhança é igual a $\frac{1}{|\varphi|}$) e então seleciona aleatoriamente uma solução vizinha a partir da solução atual usando a estrutura de vizinhança previamente escolhida. Baseada na qualidade da solução vizinha e da solução corrente, e também da temperatura vigente, o algoritmo decide em aceitar ou não a solução vizinha como sendo a solução corrente. No final do

Algoritmo 2: Simulated Annealing (SA_2)

```
1  s ← gerar uma solução de forma aleatória
2  melhor_solucao ← s
3  T_atual ← T_i
4  while T_atual ≥ T_f do
5      it ← 0
6      while it < N do
7          k ← escolher aleatoriamente uma estrutura de vizinhança em φ
8          s' ← gerar aleatoriamente uma solução vizinha s' ∈ N_k(s)
9          Δ ← f_GH+LS(s') − f_GH+LS(s)
10         if Δ < 0 or random(0, 1) < e^(−Δ/T_atual) then
11             s ← s'
12             if f_GH+LS(s') < f_GH+LS(melhor_solucao) then
13                 melhor_solucao ← s'
14             end
15         end
16         it ← it + 1
17     end
18     T_atual ← T_atual × (1 − α)
19 end
20 f_B&B(melhor_solucao)
21 return melhor_solucao
```

algoritmo, a melhor solução encontrada é retornada.

Note que toda solução explorada na fase interna do Algoritmo 2 (linhas 9 e 12) é avaliada utilizando a função de avaliação heurística GH+LS descrita na Seção 3.4.2.3. Apenas ao fim de todo o processo do algoritmo (linha 20) é aplicada a função de avaliação exata B&B, descrita na Seção 3.4.2.2, na melhor solução encontrada pelo algoritmo. Essas decisões foram tomadas devido a alta complexidade computacional de se avaliar todas as soluções exploradas pelo algoritmo de forma exata.

Além dos parâmetros temperatura inicial (T_i), temperatura final (T_f), número de iterações em cada temperatura (N) e fator de resfriamento (α), o algoritmo SA_2 apresenta um quinto parâmetro (φ), o qual define o conjunto de estruturas de vizinhança utilizadas, sendo que esse conjunto é composto por um subconjunto das quatro estruturas de vizinhança descritas na Seção 3.4.3. A fim de obter o melhor

desempenho do algoritmo todos os parâmetros foram calibrados (veja a Seção 3.5.3).

3.4.8 Variable neighborhood search

Proposta por Mladenović & Hansen [1997], a *Variable Neighborhood Search* (VNS) é uma metaheurística que utiliza um método de buscal local para explorar o espaço de soluções através de trocas sistemáticas nas estruturas de vizinhança. Para isso, o VNS deve ter um conjunto de estruturas de vizinhança pré-determinadas. O objetivo da troca de estruturas de vizinhança é para que o processo não fique preso em ótimos locais, uma vez que o ótimo local alcançado utilizando uma vizinhança pode ser diferente do ótimo local alcançado através de uma outra vizinhança.

A ideia da metaheurística VNS despertou interesse na comunidade científica, como pode ser visto pelos diversos trabalhos que a usam para resolver problemas de otimização combinatória [Polacek et al., 2004; Kytöjoki et al., 2007; Fleszar et al., 2009; Schneider et al., 2015].

Para entender melhor o funcionamento da metaheurística VNS proposta para o DVRPMS, primeiramente, é preciso apresentar seus dois parâmetros: número de iterações sem melhora (N) e a sequência (Φ) de estruturas de vizinhança, na ordem que é analisada pelo algoritmo. A sequência Φ é composta por alguma permutação na ordem das estruturas de vizinhança de algum subconjunto das quatro estruturas definidas na Seção 3.4.3. Com o intuito de obter o melhor desempenho do algoritmo os dois parâmetros foram calibrados (veja a Seção 3.5.3).

O Algoritmo 3 descreve os passos realizados pela metaheurística VNS. Inicialmente uma solução inicial é gerada aleatoriamente. Enquanto o número de iterações sem melhoria (N) não atinge o máximo estabelecido, o algoritmo realiza uma perturbação na solução atual, seguida de uma busca local. Caso seja encontrada uma melhor solução que a melhor solução conhecida, o processo se repete utilizando a primeira estrutura de vizinhança definida, caso contrário, o processo se repete utilizando a estrutura de vizinhança definida subsequentemente, até que todas as estruturas de vizinhanças sejam exploradas.

Da mesma forma e pelas mesmas razões que no Algoritmo 2, no Algoritmo 3 toda solução explorada na fase interna do algoritmo (linha 9) é avaliada utilizando a função de avaliação heurística GH+LS descrita na Seção 3.4.2.3, sendo que apenas

Algoritmo 3: Variable Neighborhood Search (VNS)

```
1  s ← gerar uma solução de forma aleatória
2  k_max ← número de estruturas de vizinhança do conjunto Φ
3  N_sem_melhoria ← 0
4  while N_sem_melhoria ≤ N do
5      k ← 1
6      repeat
7          s' ← gerar aleatoriamente uma solução vizinha s' ∈ N_k(s) utilizando a
           k-ésima estrutura de vizinhança do conjunto Φ
8          s' ← aplicar a busca local na solução s' utilizando a k-ésima estrutura
           de vizinhança do conjunto Φ
9          if f_GH+LS(s') − f_GH+LS(s) < 0 then
10             s ← s'
11             N_sem_melhoria ← 0
12             k ← 1
13         else
14             k ← k + 1
15         end
16     until k > k_max;
17     N_sem_melhoria ← N_sem_melhoria + 1
18 end
19 f_B&B(s)
20 return s
```

ao fim de todo o processo (linha 19) é aplicada a função de avaliação exata B&B, descrita na Seção 3.4.2.2, na melhor solução encontrada pelo algoritmo.

3.4.9 Iterated local search

A metaheurística *Iterated Local Search* (ILS) foi proposta por Lourenço et al. [2003] e consiste de uma técnica que explora o espaço de soluções através de um método de busca local e por um método que aplica perturbações aos ótimos locais encontrados. Basicamente a técnica consiste em aplicar uma busca local a partir de uma solução inicial e, após encontrar uma solução de ótimo local, é aplicada uma perturbação nessa solução para poder explorar outras regiões do espaço de solução. Esse processo se repete até que algum critério de parada seja satisfeito (número de iterações, tempo e outros).

A ILS é largamente aplicada para resolver problemas de otimização combinatória [Stützle, 2006; Vansteenwegen et al., 2009; Penna et al., 2013].

Antes de descrever o funcionamento do algoritmo ILS proposto para o DVRPMS, é importante destacar a existência dos três parâmetros do algoritmo: número de iterações sem melhora (N), a estrutura de vizinhança (ρ) utilizada para fazer a perturbação na solução corrente e a estrutura de vizinhança (μ) utilizada na fase de busca local.

O Algoritmo 4 descreve passo a passo o método heurístico baseado na metaheurística ILS proposta para o DVRPMS. Inicialmente é aplicada uma busca local a partir de uma solução inicial criada aleatoriamente. Enquanto o número de iterações sem melhoria não atinge o máximo estabelecido, o algoritmo realiza uma perturbação na solução atual, seguida de uma busca local. A melhor solução encontrada durante todo o processo é guardada e ao final ela é retornada.

Com o objetivo de obter o melhor desempenho do algoritmo todos os seus parâmetros foram calibrados (veja a Seção 3.5.3).

Algoritmo 4: Iterated Local Search (ILS)

```
1  s ← gerar uma solução de forma aleatória
2  s ← aplicar a busca local na solução s utilizando a estrutura de vizinhança μ
3  N_sem_melhoria ← 0
4  while N_sem_melhoria ≤ N do
5  |   s' ← gerar aleatoriamente uma solução vizinha s' ∈ N_ρ(s) utilizando a
   |       estrutura de vizinhança ρ // perturbação
6  |   s' ← aplicar a busca local na solução s' utilizando a estrutura de vizinhança
   |       μ // busca local
7  |   if f_GH+LS(s') − f_GH+LS(s) < 0 then
8  |   |   s ← s'
9  |   |   N_sem_melhoria ← 0
10 |   else
11 |   |   N_sem_melhoria ← N_sem_melhoria + 1
12 |   end
13 end
14 f_B&B(s)
15 return s
```

Note que, devido à alta complexidade computacional de se avaliar todas as soluções exploradas pelo algoritmo de forma exata, toda solução explorada na fase interna do Algoritmo 3 (linha 7) é avaliada utilizando a função de avaliação heurística GH+LS descrita na Seção 3.4.2.3. Apenas ao fim de todo o processo do algoritmo (linha 14) é aplicada a função de avaliação exata B&B, descrita na Seção 3.4.2.2, na melhor solução encontrada pelo algoritmo.

3.5 Resultados computacionais

Nesta seção são apresentadas as características das instâncias, uma comparação entre as diferentes funções de avaliação propostas, a calibração dos algoritmos heurísticos e os resultados alcançados pelos diferentes métodos propostos para o DVRPMS.

Todos os algoritmos heurísticos propostos para o DVRPMS foram implementados utilizando a linguagem de programação C/C++, os códigos foram compilados com GNU g++ versão 4.8.4 e executados de forma sequencial (sem paralelismo). Já a abordagem exata (formulação ILP) foi implementada via ILOG Concert Technology C++ e foi executada utilizando o software ILOG CPLEX versão acadêmica 12.5, com todas as configurações padrões, exceto para o tempo de execução, que foi limitado a 1 hora.

A fase de calibração dos algoritmos heurísticos foi processada no cluster da Universidade Federal de Viçosa (UFV). Já os demais experimentos foram realizados em um computador Intel Core i5-3570 CPU @ 3.40GHz x 4, com 16GB de memória RAM e sistema operacional Ubuntu 14.04 LTS, 64 bits.

3.5.1 Descrição das instâncias de teste

As instâncias utilizadas para a validação das abordagens propostas foram as mesmas instâncias definidas por Iori & Riera-Ledesma [2015] para o DVRPMS. Iori & Riera-Ledesma [2015] definiram 24 tipos de instâncias diferentes, sendo que cada tipo é definido pelo número de requisições de transporte envolvidas, pelo número de veículos disponíveis e pelas configurações dos compartimentos de carga dos veículos. Esses 24 tipos de instâncias podem ser divididos em dois conjuntos. O primeiro

conjunto, denotado por C, contém 15 tipos de instâncias em que a capacidade total dos veículos é exatamente igual ao número total de requisições de transporte (número de produtos), ou seja, para todos esses tipos de instâncias os contêineres devem ser completamente carregados a fim de atender todos os clientes. O segundo conjunto, denotado por $\neg C$, contém 9 tipos de instâncias em que o número total de requisições de transporte (número de produtos) é menor que a capacidade total dos veículos, assim os contêineres não necessitam ser completamente preenchidos para que os clientes sejam atendidos. As Tabelas 3.1 e 3.2 descrevem, respectivamente, as características dos conjuntos C e $\neg C$.

Com relação às localizações dos clientes, Iori & Riera-Ledesma [2015] utilizaram as informações de 5 instâncias (R05, R06, R07, R08 e R09) descritas por Petersen & Madsen [2009] para o problema DTSPMS. Cada uma dessas instâncias descritas para o DTSPMS informa as localizações dos clientes na região de coleta e na região de entrega. Essas localizações foram escolhidas ao acaso dentro de duas diferentes regiões de dimensões 100×100, sendo que o depósito de cada região foi inserido nas coordenadas $(50, 50)$. A distância entre dois pontos quaisquer de uma mesma região foi definida como sendo a distância euclideana arredondada entre eles.

Tabela 3.1: Características das instâncias do grupo C definidas para o DVRPMS.

Tipo	$\|I\|$	$\|K\|$	$(R \times L)'s$	$\sum_{k \in K} R_k L_k$
(a)	12	2	(2×3) (2×3)	12
(b)	12	2	(2×2) (2×4)	12
(c)	12	3	(2×2) (2×2) (2×2)	12
(d)	16	2	(2×4) (2×4)	16
(e)	16	3	(2×2) (2×3) (2×3)	16
(f)	16	4	(2×2) (2×2) (2×2) (2×2)	16
(g)	18	2	(2×3) (3×4)	18
(h)	18	3	(2×3) (2×3) (2×3)	18
(i)	18	4	(2×2) (2×2) (2×2) (2×3)	18
(j)	20	2	(2×4) (3×4)	20
(k)	20	3	(2×3) (2×3) (2×4)	20
(l)	20	4	(2×3) (2×3) (2×2) (2×2)	20
(m)	24	2	(3×4) (3×4)	24
(n)	24	3	(2×4) (2×4) (2×4)	24
(o)	24	4	(2×3) (2×3) (2×3) (2×3)	24

Tabela 3.2: Características das instâncias do grupo $\neg C$ definidas para o DVRPMS.

Tipo	$\|I\|$	$\|K\|$	$(R \times L)'s$	$\sum_{k \in K} R_k L_k$
(p)	18	2	(4×4) (4×4)	32
(q)	18	3	(3×4) (3×4) (3×4)	36
(r)	18	4	(2×4) (2×4) (2×4) (2×4)	32
(s)	20	2	(4×4) (4×4)	32
(t)	20	3	(3×4) (3×4) (3×4)	36
(u)	20	4	(2×4) (2×4) (2×4) (2×4)	32
(v)	24	2	(4×4) (4×4)	32
(w)	24	3	(3×4) (3×4) (3×4)	36
(x)	24	4	(2×4) (2×4) (2×4) (2×4)	32

3.5.2 Funções de avaliação

Os gráficos das Figuras 3.18 a 3.22 mostram uma comparação entre os dois diferentes métodos exatos utilizados para determinar as rotas mais curtas a partir de um determinado plano de carregamento. Cada uma das figuras mostra o número de contêineres avaliados no decorrer de 60 segundos pela formulação MILP e pelo algoritmo B&B, considerando os diferentes contêineres descritos nas Tabelas 3.1 e 3.2.

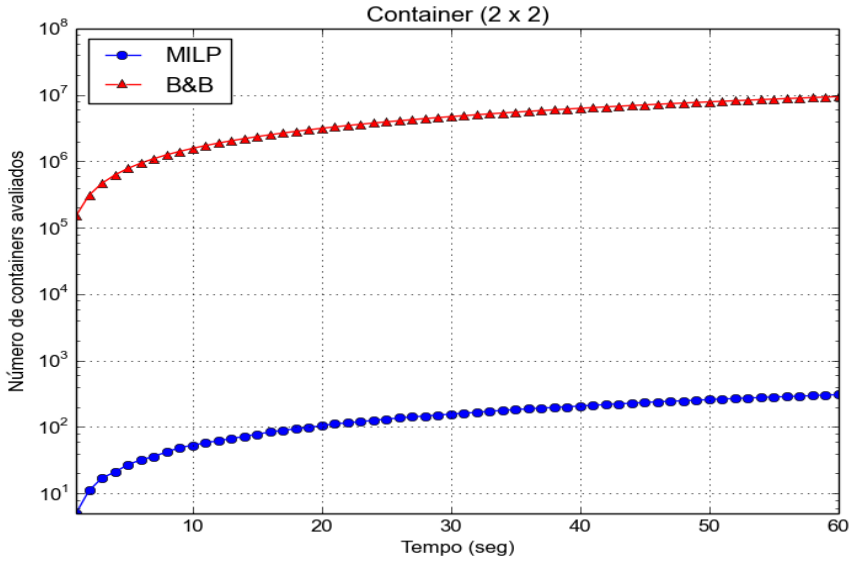


Figura 3.18: Número de contêineres (2×2) avaliados pelas funções de avaliação MILP e B&B.

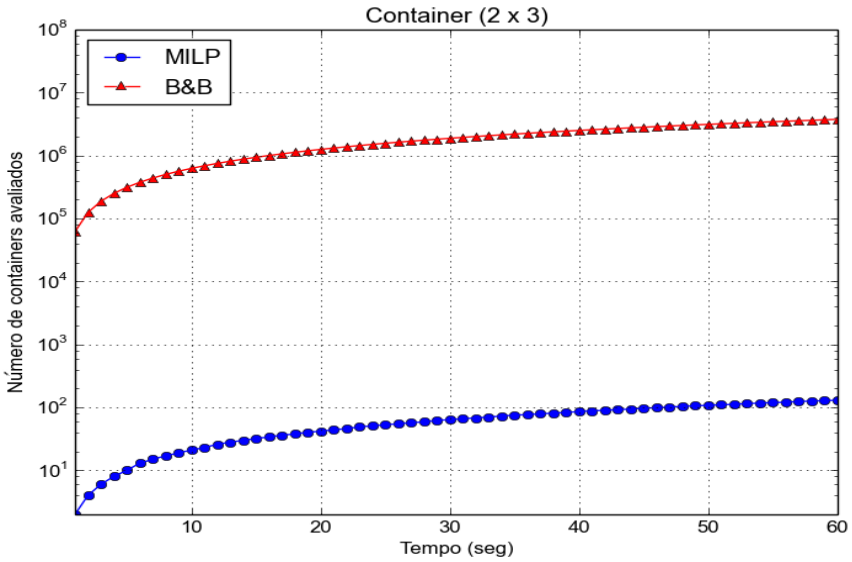


Figura 3.19: Número de contêineres (2×3) avaliados pelas funções de avaliação MILP e B&B.

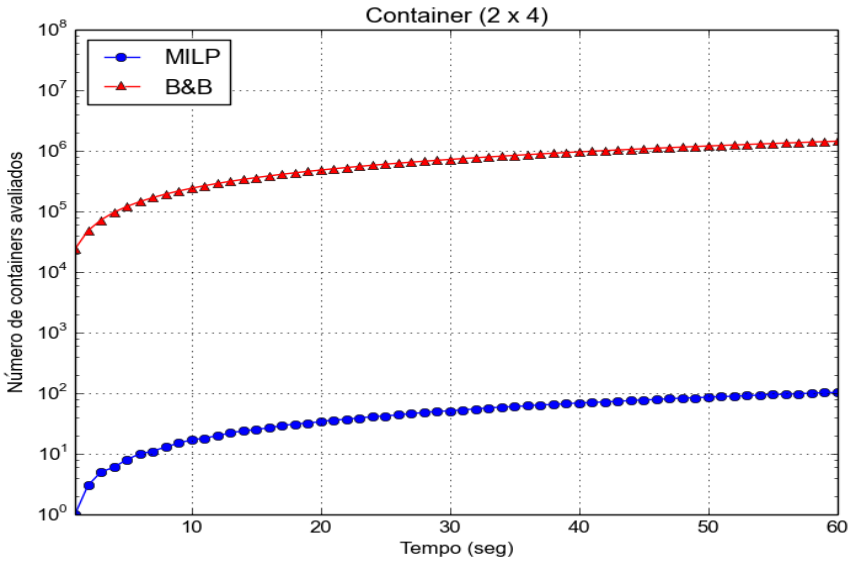


Figura 3.20: Número de contêineres (2×4) avaliados pelas funções de avaliação MILP e B&B.

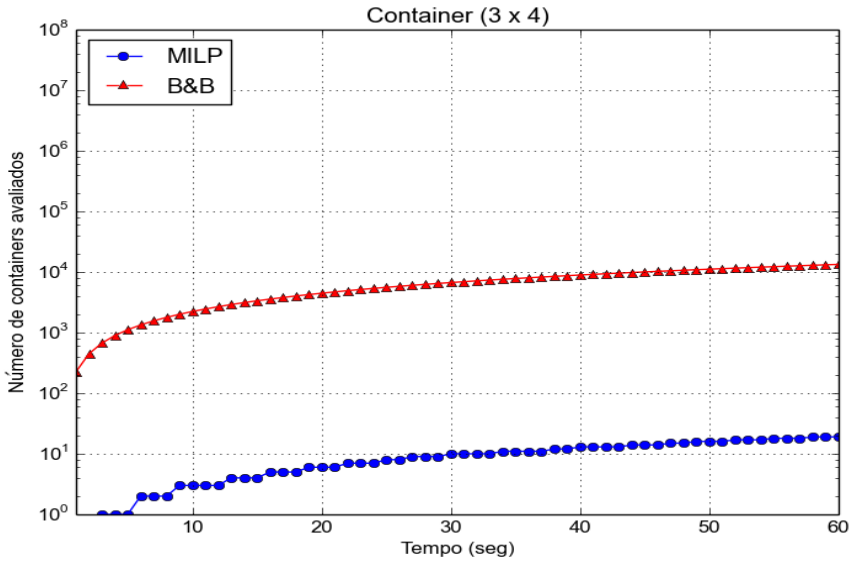


Figura 3.21: Número de contêineres (3×4) avaliados pelas funções de avaliação MILP e B&B.

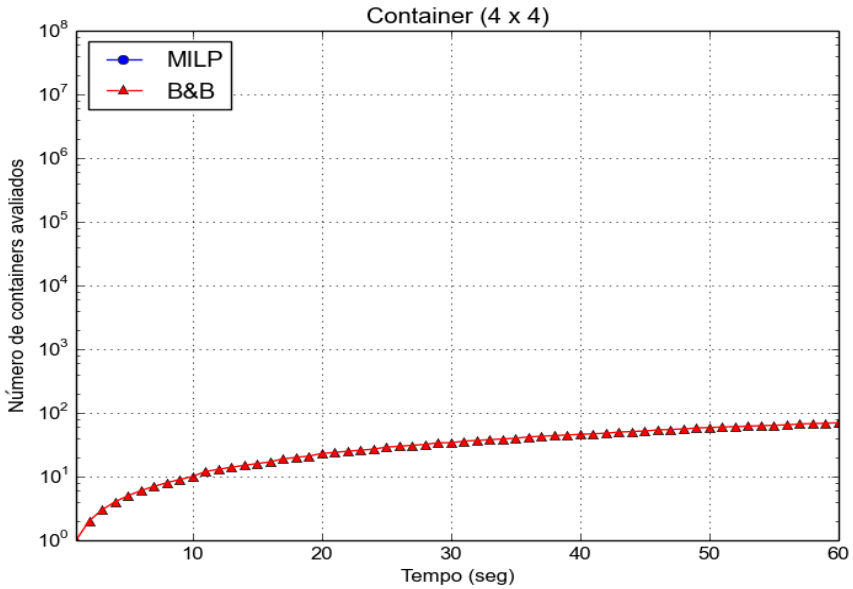


Figura 3.22: Número de contêineres (4×4) avaliados pelas funções de avaliação MILP e B&B.

Através dos gráficos é possível notar que o algoritmo B&B apresenta maior eficiência comparado à formulação MILP, sendo capaz de avaliar um número maior de contêineres em menos tempo. Note que o gráfico da Figura 3.22 mostra que a formulação MILP não foi capaz de avaliar nenhum contêiner de dimensão (4×4) com até 60 segundos de processamento. Também é possível notar que a complexidade do algoritmo B&B aumenta com o aumento das dimensões dos contêineres, isto é, quando maior as dimensões dos contêineres, maior é a complexidade.

Uma comparação entre a função de avaliação exata B&B e função de avaliação heurística GH+LS é feita na Tabela 3.3, a qual resume o comportamento de ambas as funções de avaliação. Foram gerados aleatoriamente 1000 planos de carregamento para cada tipo de contêiner. A coluna t(ms) informa o tempo em milisegundos gasto por cada função para avaliar todos os 1000 planos de carregamento. A coluna Opt informa a porcentagem de vezes que a função heurística GH+LS encontrou a mesma solução (ótima) que o método exato B&B. E, por fim, a coluna RPD informa a diferença relativa média entre as 1000 avaliações realizadas pelos dois métodos.

Tabela 3.3: Comparação entre as funções de avaliação B&B e GH+LS utilizadas para avaliar as soluções do DVRPMS.

$R \times L$	B&B	GH+LS		
	t(ms)	t(ms)	Opt	RPD
(2×2)	6	2	57.0%	1.4%
(2×3)	9	3	44.2%	1.9%
(2×4)	27	6	31.0%	2.4%
(3×4)	3593	17	4.0%	6.5%
(4×4)	721480	36	0.4%	10.1%

Com os dados da Tabela 3.3 é possível constatar que a heurística GH+LS é executada mais rapidamente que o método exato B&B para todos as configurações de contêineres. Analisando a frequência de vezes (coluna Opt) que a heurística GH+LS foi capaz de encontrar as mesmas rotas encontradas pelo método exato B&B e a diferença relativa média (coluna RPD) entre as distâncias das rotas encontradas por ambos os métodos, constata-se um maior desempenho na avaliação dos contêineres de menor tamanho.

A heurística GH+LS pode não parecer eficiente quanto a qualidade das soluções encontradas, principalmente para os contêineres (3×4) e (4×4). Porém, uma característica do DVRPMS e uma decisão ao projetar a fase de busca local (LS) da heurística de avaliação GH+LS pode tornar a função de avaliação heurística mais eficiente do que é reportado na Tabela 3.3.

A característica do DVRPMS supracitada se refere ao fato que um determinado par de rotas (rota de coleta e rota de entrega) pode ser alçancado através de diferentes planos de carregamento, ou seja, diferentes representações de soluções podem determinar as mesmas rotas, aumentando as chances da heurística GH+LS encontrar as rotas ótimas. A Figura 3.23 ilustra essa característica, mostrando quatro possíveis planos de carregamento que satisfazem um mesmo par de rotas.

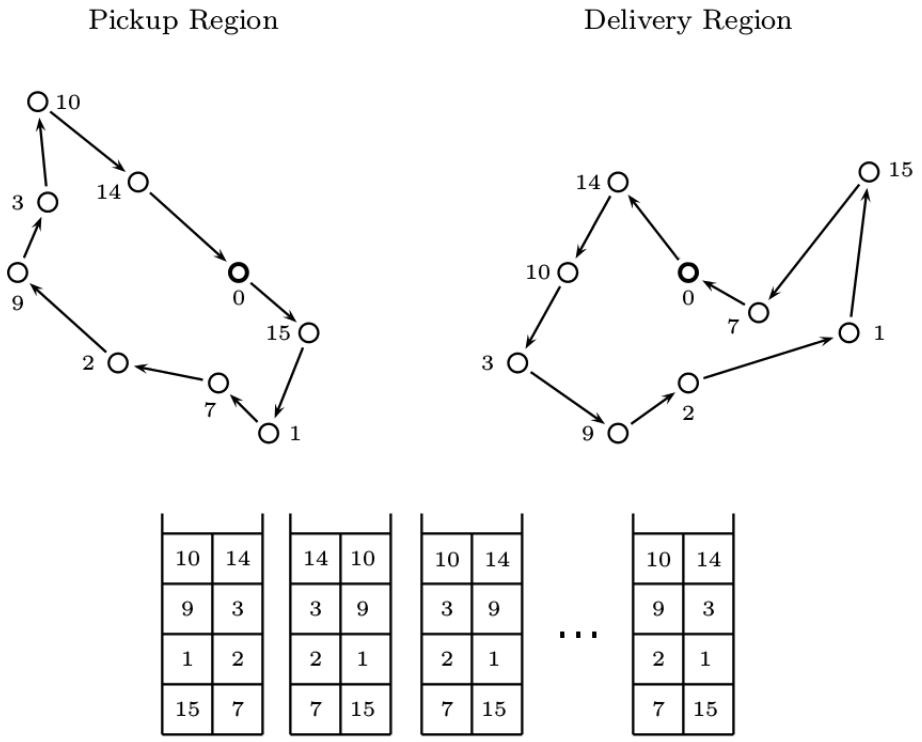


Figura 3.23: Ilustração da propriedade de que um par de rotas pode ser alcançadas a partir de diferentes planos de carregamento.

A decisão, mencionada anteriormente, ao projetar a fase de busca local (LS) da heurística de avaliação GH+LS foi tomada visto que durante o processo de

busca realizado pelos algoritmos heurísticos propostos para o DVRPMS, pode acontecer de um mesmo plano de carregamento ser avaliado mais de uma vez. Com isso, a fase de busca local (LS) da heurística de avaliação GH+LS foi projetada escolhendo-se os vizinhos de forma randômica, pois isso pode aumentar a eficiência da GH+LS, uma vez que a cada execução da busca local, um ótimo local diferente (possivelmente o ótimo global) pode ser encontrado para um mesmo plano de carregamento.

Embora seja impossível saber quantas vezes um certo plano de carregamento será avaliado pela função GH+LS, um experimento foi realizado com o intuito de demonstrar que a qualidade das rotas obtidas pode ser melhorada ao avaliar um determinado plano de carregamento por mais de uma vez pela função GH+LS. Para esse experimento foram gerados aleatoriamente 1000 planos de carregamento para cada tipo de contêiner, sendo que cada um desses contêineres foram avaliados 2, 5 e 10 vezes pela função GH+LS. A Tabela 3.4 descreve os resultados desse experimento, onde as colunas Opt e RPD têm o mesmo significado que os da Tabela 3.3. Pelos resultados, pode-se observar que a eficiência da função de avaliação heurística GH+LS realmente aumenta quando se aumenta o número de vezes de sua aplicação em um mesmo plano de carregamento.

Tabela 3.4: Eficiência da heurística de avaliação das soluções do DVRPMS.

$R \times L$	2x		5x		10x	
	Opt	RPD	Opt	RPD	Opt	RPD
(2×2)	71.5%	0.774%	83.6%	0.407%	85.0%	0.387%
(2×3)	59.0%	1.125%	73.2%	0.770%	78.7%	0.684%
(2×4)	45.2%	1.804%	61.2%	1.308%	67.7%	1.187%
(3×4)	4.8%	5.765%	10.7%	4.909%	13.5%	4.518%
(4×4)	0.6%	9.340%	0.7%	8.540%	1.2%	8.121%

3.5.3 Calibração dos parâmetros dos algoritmos

A qualidade dos resultados obtidos pelos algoritmos heurísticos está diretamente ligada aos seus parâmetros. Portanto, um experimento fatorial completo foi realizado para avaliar o comportamento de cada um dos quatro algoritmos heurísticos

propostos para o DVRPMS utilizando diferentes valores para seus parâmetros, a fim de encontrar a configuração dos valores que melhor representa cada algoritmo. Esse experimento é largamente utilizado e consiste em examinar todas as combinações dos níveis de fatores, sendo que cada fator é um parâmetro controlado.

O algoritmo heurístico SA_1 (Algoritmo 1) apresenta quatro parâmetros controlados: temperatura inicial (T_i), temperatura final (T_f), número de iterações em cada temperatura (N) e fator de resfriamento (α). Tais parâmetros foram calibrados em [Chagas et al., 2016], onde os seguintes valores foram determinados: $T_i = 20$, $T_f = 10$, $N = 2|I|^2$ e $\alpha = 0.002$. Porém, uma nova calibração, mais completa, foi realizada considerando os diferentes valores (níveis) testados para cada um dos parâmetros (fatores) descritos na Tabela 3.5.

Tabela 3.5: Conjunto de valores testados na calibração do SA_1.

Parâmetro	Valores testados	#
T_i	15, 20, 50, 100	4
T_f	1, 5, 10	3
N	$\|K\|\|I\|$, 100, $\|I\|^2$, $2\|I\|^2$, $\|K\|\|I\|^2$	5
α	0.001, 0.002, 0.005, 0.010	4

Já o algoritmo heurístico SA_2 (Algoritmo 2) apresenta cinco parâmetros controlados, pois nesta versão o conjunto de estruturas de vizinhança a ser utilizado também é um parâmetro, logo os cinco parâmetros são: temperatura inicial (T_i), temperatura final (T_f), número de iterações em cada temperatura (N), fator de resfriamento (α) e conjunto de estruturas de vizinhança (φ). Os diferentes valores (níveis) testados para cada um destes parâmetros (fatores) são mostrados na Tabela 3.6.

Note que são testados todos os 15 possíveis[7] subconjuntos provenientes do conjunto das 4 diferentes estruturas de vizinhança definidas para o problema.

O algoritmo heurístico VNS (Algoritmo 3) apresenta dois parâmetros controlados: número de iterações sem melhora (N) e a sequência (Φ) de estruturas de vizinhança, na ordem que é analisada pelo algoritmo. A Tabela 3.7 mostra os

[7] O número total de subconjuntos não vazios formados a partir de um conjunto de 4 elementos é igual a $2^4 - 1 = 15$

Tabela 3.6: Conjunto de valores testados na calibração do SA_2.

Parâmetro	Valores testados	#
T_i	15, 20, 50, 100	4
T_f	1, 5, 10	3
N	$\|K\|\|I\|$, 100, $\|I\|^2$, $2\|I\|^2$, $\|K\|\|I\|^2$	5
α	0.001, 0.002, 0.005, 0.010	4
φ	$\{TP\}$, $\{CP\}$, $\{PP\}$, $\{TS\}$, $\{TP, CP\}$, $\{TP, PP\}$, $\{TP, TS\}$, $\{CP, PP\}$, $\{CP, TS\}$, $\{PP, TS\}$, $\{TP, CP, PP\}$, $\{TP, CP, TS\}$, $\{TP, PP, TS\}$, $\{CP, PP, TS\}$, $\{TP, CP, PP, TS\}$	15

diferentes valores (níveis) testados dos parâmetros (fatores) do algoritmo.

Note que são testados todas as 60 possíveis[8] permutações dos elementos de todos os possíveis subconjuntos de tamanho maior ou igual a 2, sendo que esses subconjuntos são provenientes do conjunto das 4 diferentes estruturas de vizinhança definidas para o problema.

O algoritmo heurístico ILS (Algoritmo 4) apresenta três parâmetros controlados: número de iterações sem melhora (N), a estrutura de vizinhança (ρ) utilizada para fazer a perturbação na solução corrente e a estrutura de vizinhança (μ) utilizada na fase de busca local. Os diferentes valores (níveis) testados para cada um destes parâmetros (fatores) são mostrados na Tabela 3.8.

A fim de determinar a melhor configuração de parâmetros dos algoritmos para diferentes cenários, foi realizado um experimento para as instâncias do grupo C e, separadamente, outro experimento para as instâncias do grupo $\neg C$. Em cada experimento, foi utilizado uma amostra de 20% das instâncias (uma instância de cada tipo). Para cada instância, cada algoritmo foi executado 30 vezes usando cada uma das diferentes configurações de parâmetros descritas nas Tabelas 3.5, 3.6, 3.7 e 3.8. O desvio percentual relativo (*Relative Percentage Deviation* - RPD) foi utilizada como variável resposta na análise estatística. A equação (3.33) descreve como o RPD foi calculado.

$$RPD = \frac{f_i - f_{best}}{f_{best}} \times 100\% \tag{3.33}$$

[8] O número total de permutações dos elementos de todos os possíveis subconjuntos de tamanho maior ou igual a 2 é dado por $\sum_{i=2}^{4} C_{4,i}\ (i\,!) = 6(2!) + 4(3!) + 1(4!) = 60$, onde $C_{4,i} = \frac{4!}{(4-i)!\ i!}$ é o número de maneiras que se pode selecionar i elementos distintos em conjunto com 4 elementos.

Tabela 3.7: Conjunto de valores testados na calibração do VNS.

Parâmetro	Valores testados	#
N	$\|K\|\|I\|$, 100, $\|I\|^2$, 500	4
Φ	$[PP, TS]$, $[TS, PP]$, $[CP, TS]$, $[TS, CP]$, $[CP, PP]$, $[PP, CP]$, $[CP, PP, TS]$, $[CP, TS, PP]$, $[PP, CP, TS]$, $[PP, TS, CP]$, $[TS, CP, PP]$, $[TS, PP, CP]$, $[TP, TS]$, $[TS, TP]$, $[TP, PP]$, $[PP, TP]$, $[TP, PP, TS]$, $[TP, TS, PP]$, $[PP, TP, TS]$, $[PP, TS, TP]$, $[TS, TP, PP]$, $[TS, PP, TP]$, $[TP, CP]$, $[CP, TP]$, $[TP, CP, TS]$, $[TP, TS, CP]$, $[CP, TP, TS]$, $[CP, TS, TP]$, $[TS, TP, CP]$, $[TS, CP, TP]$, $[TP, CP, PP]$, $[TP, PP, CP]$, $[CP, TP, PP]$, $[CP, PP, TP]$, $[PP, TP, CP]$, $[PP, CP, TP]$, $[TP, CP, PP, TS]$, $[TP, CP, TS, PP]$, $[TP, PP, CP, TS]$, $[TP, PP, TS, CP]$, $[TP, TS, CP, PP]$, $[TP, TS, PP, CP]$, $[CP, TP, PP, TS]$, $[CP, TP, TS, PP]$, $[CP, PP, TP, TS]$, $[CP, PP, TS, TP]$, $[CP, TS, TP, PP]$, $[CP, TS, PP, TP]$, $[PP, TP, CP, TS]$, $[PP, TP, TS, CP]$, $[PP, CP, TP, TS]$, $[PP, CP, TS, TP]$, $[PP, TS, TP, CP]$, $[PP, TS, CP, TP]$, $[TS, TP, CP, PP]$, $[TS, TP, PP, CP]$, $[TS, CP, TP, PP]$, $[TS, CP, PP, TP]$, $[TS, PP, TP, CP]$, $[TS, PP, CP, TP]$	60

Tabela 3.8: Conjunto de valores testados na calibração do ILS.

Parâmetro	Valores testados	#
N	$\|K\|\|I\|$, 100, $\|I\|^2$, 500, 1000	5
ρ	TP, CP, PP, TS	4
μ	TP, CP, PP, TS	4

onde f_i é o valor da solução obtida na i-ésima execução para uma determinada instância e f_{best} é o valor da melhor solução obtida para essa instância, considerando todos os parâmetros analisados.

Para analisar os resultados obtidos pelas diferentes configurações de parâmetros, foi realizada uma análise de variância (ANOVA) paramétrica e também foi aplicado o teste não paramétrico de Kruskal-Wallis. Ambos foram utilizados para determinar se as diferenças observadas são estatisticamente significantes com um nível de confiança de 99%.

Os resultados obtidos pela ANOVA e pelo teste de Kruskal-Wallis indicaram,

para todos os quatros algoritmos propostos, nos dois cenários, a existência de pelo menos uma configuração de parâmetros significativamente diferente das demais configurações. Para determinar entre quais configurações de parâmetros existe a diferença apontada pelos testes anteriores, foi realizado um teste de comparações múltiplas pelo método de Scott & Knott [1974].

Os parâmetros foram determinados de forma que a qualidade das soluções obtidas por uma certa configuração de parâmetros e o tempo gasto para executá-la estivesse equilibrada. Portanto, entre o grupo de configurações que foram significativamente melhores que os demais, foi escolhida a configuração de parâmetros que gastou menos tempo para ser executada.

A Tabela 3.9 mostra os parâmetros determinados pela análise estatística para cada um dos algoritmos e para cada grupo de instância.

Tabela 3.9: Parâmetros determinados para os algoritmos heurísticos propostos para o DVRPMS.

Algoritmo	Grupo de instâncias	Parâmetros selecionados
SA_1	C	$T_i = 20, \quad T_f = 10, \quad N = 2\|I\|^2, \quad \alpha = 0.001$
	$\neg C$	$T_i = 15, \quad T_f = 5, \quad N = 2\|I\|^2, \quad \alpha = 0.001$
SA_2	C	$T_i = 15, \quad T_f = 10, \quad N = 2\|I\|^2, \quad \alpha = 0.002, \quad \varphi = \{TP\}$
	$\neg C$	$T_i = 15, \quad T_f = 5, \quad N = \|K\|\|I\|^2, \quad \alpha = 0.002, \quad \varphi = \{TP\}$
VNS	C	$N = 500, \quad \Phi = [TP, PP]$
	$\neg C$	$N = 100, \quad \Phi = [PP, CP, TP]$
ILS	C	$N = 100, \quad \rho = TP, \quad \mu = CP$
	$\neg C$	$N = 100, \quad \rho = CP, \quad \mu = TS$

3.5.4 Resultados experimentais

Os resultados obtidos pelos métodos exatos e heurísticos propostos na literatura e também os que aqui foram propostos são apresentados separadamente a seguir.

3.5.4.1 Métodos exatos

O tempo de processamento da formulação ILP aqui proposta (Seção 3.3) foi limitada a 1 hora, assim como nos métodos exatos propostos por Iori & Riera-Ledesma [2015]. Os resultados obtidos pelos diferentes métodos exatos são apresentados em 3 tabelas. As Tabelas 3.10 e 3.11 reportam os resultados para as instâncias do grupo C, ou seja, aquelas em que capacidade total da frota de veículos é exatamente igual ao número de clientes, sendo assim, os contêineres devem ser completamente carregados para que todos os clientes sejam atendidos. A Tabela 3.10 apresenta os resultados de 30 instâncias (ID 001 até ID 030) que são consideradas por Iori & Riera-Ledesma [2015] como sendo instâncias de pequeno porte e a Tabela 3.11 apresenta os resultados das 45 instâncias (ID 031 até ID 075) de grande porte. Já a Tabela 3.12 reporta os resultados para as 45 instâncias do grupo $\neg C$ (ID 076 até ID 120), isto é, aquelas em que a capacidade total da frota de veículos é maior que o número de clientes e, consequentemente, os contêineres não necessariamente serão completamente carregados.

As três primeiras colunas das Tabelas 3.10, 3.11 e 3.12 descrevem as instâncias e as demais colunas apresentam os resultados obtidos por cada método exato. Os melhores resultados obtidos pelos três algoritmos exatos propostos por Iori & Riera-Ledesma [2015] são apresentados nas colunas LB_0, UB, t(s) e Opt, que informam, respectivamente, o *lower-bound* calculado no nó raiz da árvore de busca do método, o valor da melhor solução obtida, tempo de processamento em segundos, e se o método foi capaz de encontrar e provar a otimalidade da solução (indicado por um asterisco). Os resultados obtidos através da formulação ILP são descritos nas colunas LB_0, UB, g_c (%), g_i (%), t(s) e Opt, onde a coluna LB_0 informa o *lower-bound* computado na relaxação linear da formulação ILP, a coluna UB informa o valor da melhor solução obtida no fim da execução, a coluna g_c (%) informa o *gap* informado pelo CPLEX, a coluna g_i (%) informa o *gap* com relação à melhor solução encontrada por Iori & Riera-Ledesma [2015], a coluna t(s) informa o tempo de processamento em segundos e a coluna Opt informa por um asterisco se a formulação ILP foi capaz de provar ter encontrado a solução ótima.

Com o objetivo de evidenciar a qualidade dos resultados obtidos por cada método, os melhores resultados encontrados para cada instância são destacados em

negrito.

Tabela 3.10: Comparação entre as soluções obtidas por Iori & Riera-Ledesma [2015] e pela formulação ILP, considerando as instâncias de pequeno porte do grupo C.

Instância			Iori & Riera-Ledesma [2015]				Formulação ILP					
ID	R	T	LB_0	UB	t(s)	Opt	LB_0	UB	g_c (%)	g_i (%)	t(s)	Opt
001	R05	(*a*)	692.7	**738**	1	*	344.0	**738**	0.0	0.0	117	*
002	R06		847.4	**895**	0	*	508.0	**895**	0.0	0.0	27	*
003	R07		680.4	**761**	4	*	369.0	**761**	0.0	0.0	633	*
004	R08		802.8	**851**	0	*	595.0	**851**	0.0	0.0	37	*
005	R09		750.0	**776**	0	*	516.0	**776**	0.0	0.0	32	*
006	R05	(*b*)	660.7	**716**	1	*	370.3	**716**	0.0	0.0	46	*
007	R06		829.0	**866**	3	*	525.0	**866**	0.0	0.0	20	*
008	R07		641.5	**733**	2	*	379.6	**733**	0.0	0.0	196	*
009	R08		780.8	**858**	2	*	602.5	**858**	0.0	0.0	28	*
010	R09		722.0	**741**	1	*	525.0	**741**	0.0	0.0	18	*
011	R05	(*c*)	798.0	**855**	0	*	344.0	**855**	0.0	0.0	136	*
012	R06		940.0	**1011**	0	*	508.0	**1011**	0.0	0.0	180	*
013	R07		794.6	**894**	1	*	369.0	**894**	0.0	0.0	294	*
014	R08		917.2	**990**	0	*	595.0	**990**	0.0	0.0	319	*
015	R09		852.0	**852**	0	*	516.0	**852**	0.0	0.0	105	*
016	R05	(*d*)	844.5	**947**	224	*	518.0	**947**	6.7	0.0	1 h	
017	R06		934.2	**1036**	28	*	605.0	**1036**	1.8	0.0	1 h	
018	R07		813.8	**925**	30	*	472.0	**925**	5.1	0.0	1 h	
019	R08		917.9	**1010**	57	*	678.0	**1010**	0.0	0.0	2072	*
020	R09		864.4	**921**	679	*	615.0	924	5.0	0.3	1 h	
021	R05	(*e*)	902.8	**1050**	14	*	518.0	**1050**	7.7	0.0	1 h	
022	R06		996.4	**1114**	6	*	605.0	**1114**	3.0	0.0	1 h	
023	R07		897.8	**1063**	19	*	472.0	1084	14.9	2.0	1 h	
024	R08		1000.7	**1126**	11	*	678.0	**1126**	7.8	0.0	1 h	
025	R09		914.3	**1021**	6	*	615.0	**1021**	8.5	0.0	1 h	
026	R05	(*f*)	1055.6	**1217**	25	*	518.0	**1217**	22.0	0.0	1 h	
027	R06		1190.8	**1290**	7	*	605.0	1291	16.7	0.1	1 h	
028	R07		1071.1	**1230**	30	*	472.0	**1230**	30.2	0.0	1 h	
029	R08		1167.9	**1261**	13	*	678.0	1275	17.6	1.1	1 h	
030	R09		1061.9	**1134**	5	*	615.0	**1134**	17.9	0.0	1 h	
Média			878.1	962.7	39.0	$\frac{30}{30}$	524.3	964.0	5.5	0.1	1822.0	$\frac{16}{30}$

Analisando as Tabelas 3.10 e 3.11 é possível notar a superioridade dos resultados de Iori & Riera-Ledesma [2015] com relação aos resultados alcançados pela formulação ILP proposta, sendo capaz de encontrar melhores soluções com menos tempo para a maioria das instâncias. Porém, para as instâncias de pequeno porte (Tabela 3.10), mesmo que não tenha provado na otimalidade praticamente a metade das instâncias, a formulação ILP encontrou a mesma solução encontrada pelo métodos proposto por Iori & Riera-Ledesma [2015] para 26 das 30 instâncias.

Para as instâncias de grande porte do grupo C (Tabela 3.11) a eficiência dos métodos de Iori & Riera-Ledesma [2015] é mais evidente, pois as soluções encontradas pela formulação ILP estão distantes (coluna $g_i\%$) das soluções encontradas por Iori & Riera-Ledesma [2015], exceto para as instâncias com ID 066, 067, 068 e 070, para as quais nenhuma solução foi encontrada dentro de 1 hora de processamento.

Em geral, embora não seja tão eficiente que os métodos de Iori & Riera-Ledesma [2015], a formulação ILP apresentou grande importância para esta pesquisa, uma vez que foi através de seus resultados que foi possível notar inconsistências nos resultados publicados por Iori & Riera-Ledesma [2015], em que para algumas instâncias as soluções informadas se tratavam de soluções inviáveis (as rotas não respeitavam a política LIFO das pilhas). Após notar essas inconsistências, os autores Iori e Riera-Ledesma foram notificados e se mostraram dispostos a encontrar e resolver os possíveis problemas para depois poderem informar os resultados viáveis das instâncias. Infelizmente, até o momento da escrita desse texto (começo de fevereiro de 2017) apenas os resultados de 70 instâncias (ID 001 até ID 070) foram fornecidos. Portanto, as Tabelas 3.10, 3.11 e 3.12 informam apenas os resultados corretos alcançados por Iori & Riera-Ledesma [2015] para essas instâncias.

Os resultados descritos na Tabela 3.12 mostram a dificuldade da formulação ILP ao resolver as instâncias do grupo $\neg C$. Para todas as 45 instâncias, apenas uma (ID 079) foi resolvida na otimalidade.

A seguir são apresentados os resultados alcançados pelos métodos heurísticos para o DVRPMS.

3.5.4.2 Métodos heurísticos

Os resultados obtidos pelos métodos heurísticos já propostos na literatura e pelos métodos heurísticos aqui propostos são apresentados nas Tabelas 3.13, 3.14 e 3.16, enquanto as Tabelas 3.15 e 3.17 apresentam resultados estatísticos que comparam os diferentes métodos de resolução do DVRPMS.

A Tabela 3.13 reporta os resultados das instâncias de pequeno porte do grupo C (ID 001 até ID 030). Os resultados das demais instâncias do grupo C (ID 031 até ID 075) são descritos na Tabela 3.14. A Tabela 3.16 reporta os resultados das instâncias do grupo $\neg C$ (ID 076 até ID 120). As três primeiras colunas de cada uma

dessas tabelas descrevem as características das instâncias, seguidas das colunas Best e t(s) que informam, respectivamente, o valor da melhor solução obtida por métodos exatos e o tempo de processamento. As colunas seguintes apresentam os melhores resultados obtidos por Silveira et al. [2015] e os resultados obtidos pelos métodos heurísticos propostos são apresentados em três colunas, sendo que as colunas Avg, Best e t(s) informam, respectivamente, o valor da solução média obtida, o valor da melhor solução, e o tempo total de processamento gasto em 10 execuções. A última coluna das Tabelas 3.13, 3.14 e 3.16 informa o valor da melhor solução conhecida para cada instância de dados.

Tabela 3.11: Comparação entre as soluções obtidas por Iori & Riera-Ledesma [2015] e pela formulação ILP, considerando as instâncias de grande porte do grupo C.

Instância			Iori & Riera-Ledesma [2015]				Formulação ILP					
ID	R	T	LB_0	UB	t(s)	Opt	LB_0	UB	g_c (%)	g_i (%)	t(s)	Opt
031	R05	(g)	848.6	**950**	6	*	605.3	**950**	0.0	0.0	2271	*
032	R06		911.9	**1024**	83	*	641.0	**1024**	6.1	0.0	1 h	
033	R07		824.6	**932**	46	*	557.6	**932**	8.7	0.0	1 h	
034	R08		909.5	**1018**	125	*	701.8	1024	5.2	0.6	1 h	
035	R09		831.0	**919**	557	*	616.5	**919**	5.8	0.0	1 h	
036	R05	(h)	989.3	**1147**	41	*	597.0	1206	24.3	5.1	1 h	
037	R06		1067.6	**1177**	11	*	639.0	1187	18.8	0.8	1 h	
038	R07		982.9	**1123**	21	*	536.0	1176	27.9	4.7	1 h	
039	R08		1049.1	**1184**	27	*	680.0	1224	17.2	3.4	1 h	
040	R09		959.8	**1080**	34	*	608.0	1112	21.5	3.0	1 h	
041	R05	(i)	1064.7	**1269**	224	*	597.0	1295	28.7	2.0	1 h	
042	R06		1138.0	**1275**	50	*	639.0	1277	23.0	0.2	1 h	
043	R07		1079.1	**1261**	134	*	536.0	1291	33.2	2.4	1 h	
044	R08		1130.8	**1310**	364	*	680.0	1355	25.4	3.4	1 h	
045	R09		1021.8	**1157**	41	*	608.0	1182	25.8	2.2	1 h	
046	R05	(j)	869.4	**1012**	75	*	612.3	1029	11.9	1.7	1 h	
047	R06		924.0	**1018**	15	*	641.6	**1018**	3.1	0.0	1 h	
048	R07		871.5	**1047**	193	*	578.5	1142	26.0	9.1	1 h	
049	R08		909.9	**1045**	1192	*	693.5	1089	14.1	4.2	1 h	
050	R09		863.5	**973**	767	*	583.2	977	10.0	0.4	1 h	
051	R05	(k)	968.9	**1174**	579	*	607.0	1196	24.8	1.9	1 h	
052	R06		1035.6	**1200**	43	*	632.0	1307	23.4	8.9	1 h	
053	R07		1002.1	**1243**	1049	*	568.0	1359	38.3	9.3	1 h	
054	R08		1013.4	**1206**	1427	*	692.0	1283	24.4	6.4	1 h	
055	R09		939.8	**1144**	900	*	571.0	1198	26.0	4.7	1 h	
056	R05	(l)	1080.4	**1293**	991	*	607.0	1367	33.2	5.7	1 h	
057	R06		1175.0	**1340**	132	*	632.0	1381	28.5	3.1	1 h	
058	R07		1129.2	**1373**	2043	*	568.0	1449	40.4	5.5	1 h	
059	R08		1135.1	**1305**	313	*	692.0	1452	35.4	11.3	1 h	
060	R09		1042.7	**1258**	702	*	571.0	1384	37.9	10.0	1 h	
061	R05	(m)	936.8	**1067**	1 h		616.0	1304	36.3	22.2	1 h	
062	R06		994.8	**1093**	173	*	703.0	1503	39.4	37.5	1 h	
063	R07		950.4	**1098**	206	*	551.0	1265	34.3	15.2	1 h	
064	R08		983.9	**1121**	1184	*	680.0	1372	33.9	22.4	1 h	
065	R09		940.7	**1047**	1 h		628.0	1255	36.8	19.9	1 h	
066	R05	(n)	1038.3	-	1 h		616.0	**1480**	41.7		1 h	
067	R06		1110.4	-	1 h		703.0	**1692**	43.9		1 h	
068	R07		1086.7	-	1 h		551.0	**1659**	48.9		1 h	
069	R08		1099.5	**1297**	1 h		680.0	1600	43.8	23.4	1 h	
070	R09		1033.9	-	1 h		628.0	**1539**	47.6		1 h	
071	R05	(o)					616.0	**1761**	51.9		1 h	
072	R06						703.0	**1661**	42.9		1 h	
073	R07						551.0	**1864**	54.3		1 h	
074	R08						680.0	**1691**	45.6		1 h	
075	R09						628.0	**1770**	53.6		1 h	
Média			998.1	1143.9	973.7	$\frac{33}{45}$	622.8	1315.6	29.0	7.0	1 h	$\frac{1}{45}$

Tabela 3.12: Resultados obtidos pela formulação ILP, considerando as instâncias do grupo $\neg C$.

Instância			Iori & Riera-Ledesma [2015]				Formulação ILP					
ID	R	T	LB_0	UB	t(s)	Opt	LB_0	UB	g_c (%)	g_i (%)	t(s)	Opt
076	R05	(p)					597.0	**916**	4.7		1 h	
077	R06						639.0	**988**	9.6		1 h	
078	R07						536.0	**803**	2.8		1 h	
079	R08						680.0	**902**	0.0		1531.6	*
080	R09						608.0	**874**	3.9		1 h	
081	R05	(q)					597.0	**942**	9.4		1 h	
082	R06						639.0	**1032**	14.5		1 h	
083	R07						536.0	**993**	24.3		1 h	
084	R08						680.0	**1045**	13.2		1 h	
085	R09						608.0	**902**	10.9		1 h	
086	R05	(r)					597.0	**1221**	31.3		1 h	
087	R06						639.0	**1128**	24.1		1 h	
088	R07						536.0	**1169**	37.4		1 h	
089	R08						680.0	**1316**	30.2		1 h	
090	R09						608.0	**1091**	26.0		1 h	
091	R05	(s)					607.0	**950**	7.9		1 h	
092	R06						632.0	**1005**	7.1		1 h	
093	R07						568.0	**952**	14.8		1 h	
094	R08						692.0	**963**	6.0		1 h	
095	R09						571.0	**960**	11.9		1 h	
096	R05	(t)					607.0	**1044**	20.2		1 h	
097	R06						632.0	**1046**	12.5		1 h	
098	R07						568.0	**1091**	26.5		1 h	
099	R08						692.0	**1136**	23.3		1 h	
100	R09						571.0	**1014**	18.6		1 h	
101	R05	(u)					607.0	**1256**	32.6		1 h	
102	R06						632.0	**1348**	32.9		1 h	
103	R07						568.0	**1416**	44.7		1 h	
104	R08						692.0	**1356**	34.0		1 h	
105	R09						571.0	**1210**	32.0		1 h	
106	R05	(v)					616.0	**1090**	21.2		1 h	
107	R06						703.0	**1193**	24.2		1 h	
108	R07						551.0	**1210**	31.5		1 h	
109	R08						680.0	**1204**	25.7		1 h	
110	R09						628.0	**1077**	25.6		1 h	
111	R05	(w)					616.0	**1244**	33.1		1 h	
112	R06						703.0	**1537**	40.8		1 h	
113	R07						551.0	**1376**	41.4		1 h	
114	R08						680.0	**1511**	43.6		1 h	
115	R09						628.0	**1335**	42.1		1 h	
116	R05	(x)					616.0	**1618**	48.3		1 h	
117	R06						703.0	**1959**	54.2		1 h	
118	R07						551.0	**1757**	53.8		1 h	
119	R08						680.0	**1957**	55.7		1 h	
120	R09						628.0	**1705**	55.1		1 h	
Média							620.5	1196.5	25.6		1 h	$\frac{1}{45}$

Tabela 3.13: Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMS, considerando as instâncias de pequeno porte do grupo C.

Instância			Iori & Riera-Ledesma [2015]		Silveira et al. [2015]		SA_1			SA_2			VNS			ILS			Melhor solução conhecida
ID	R	T	Best	t(s)	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	
001	R05	(a)	**738**	1	746	31	**738.0**	**738**	26.8	**738.0**	**738**	3.3	**738.0**	**738**	2.3	**738.0**	**738**	1.7	738
002	R06		**895**	0	897	30	**895.0**	**895**	26.6	**895.0**	**895**	3.3	**895.0**	**895**	2.1	**895.0**	**895**	1.8	895
003	R07		**761**	4	**761**	31	**761.0**	**761**	25.0	765.2	**761**	3.3	765.2	**761**	1.8	766.6	**761**	1.6	761
004	R08		**851**	0	**851**	31	**851.0**	**851**	27.1	**851.0**	**851**	3.3	**851.0**	**851**	1.9	**851.0**	**851**	1.5	851
005	R09		**776**	0	781	31	**776.0**	**776**	27.6	**776.0**	**776**	3.4	776.6	**776**	2.4	**776.0**	**776**	2.0	776
006	R05	(b)	**716**	1	728	31	717.5	**716**	5.7	**716.0**	**716**	3.8	**716.0**	**716**	3.7	**716.0**	**716**	2.4	716
007	R06		**866**	3	873	31	**866.0**	**866**	5.8	**866.0**	**866**	3.7	867.4	**866**	3.0	**866.0**	**866**	2.0	866
008	R07		**733**	2	**733**	31	**733.0**	**733**	5.8	**733.0**	**733**	3.6	**733.0**	**733**	3.2	**733.0**	**733**	2.1	733
009	R08		**858**	2	875	31	**858.0**	**858**	5.8	**858.0**	**858**	3.7	862.1	**858**	2.9	867.9	**858**	1.8	858
010	R09		**741**	1	760	31	741.7	**741**	5.9	**741.0**	**741**	3.8	741.9	**741**	3.3	743.7	**741**	2.4	741
011	R05	(c)	**855**	0	859	35	**855.0**	**855**	11.3	**855.0**	**855**	2.1	**855.0**	**855**	1.0	**855.0**	**855**	1.0	855
012	R06		**1011**	0	**1011**	35	**1011.0**	**1011**	11.3	**1011.0**	**1011**	2.1	**1011.0**	**1011**	1.2	**1011.0**	**1011**	1.0	1011
013	R07		**894**	1	**894**	35	**894.0**	**894**	11.4	**894.0**	**894**	2.0	**894.0**	**894**	0.9	**894.0**	**894**	0.8	894
014	R08		**990**	0	**990**	35	**990.0**	**990**	11.5	**990.0**	**990**	2.1	**990.0**	**990**	1.0	**990.0**	**990**	0.8	990
015	R09		**852**	0	853	35	**852.0**	**852**	11.7	853.0	853	2.1	853.0	853	1.2	853.0	853	1.0	852
016	R05	(d)	**947**	224	976	32	948.0	948	2.9	**947.0**	**947**	9.8	947.1	**947**	10.9	947.1	**947**	6.3	947
017	R06		**1036**	28	1093	32	1038.0	**1036**	2.8	1039.0	**1036**	9.9	1042.2	**1036**	13.1	1044.2	**1036**	6.5	1036
018	R07		**925**	30	971	32	**925.0**	**925**	2.7	**925.0**	**925**	9.5	926.0	**925**	8.9	**925.0**	**925**	5.6	925
019	R08		**1010**	57	1060	33	1021.1	1020	2.8	1013.8	**1010**	9.6	1014.6	**1010**	11.2	1017.6	1012	6.2	1010
020	R09		**921**	679	951	10	925.0	925	2.7	921.3	**921**	9.9	922.8	**921**	11.3	922.2	**921**	8.2	921
021	R05	(e)	**1050**	14	1087	37	**1050.0**	**1050**	42.2	**1050.0**	**1050**	5.7	**1050.0**	**1050**	4.1	1050.4	**1050**	3.3	1050
022	R06		**1114**	6	**1114**	37	**1114.0**	**1114**	42.7	**1114.0**	**1114**	5.8	**1114.0**	**1114**	4.3	**1114.0**	**1114**	3.6	1114
023	R07		**1063**	19	1089	37	**1063.0**	**1063**	41.1	**1063.0**	**1063**	5.7	**1063.0**	**1063**	6.0	1066.3	**1063**	3.9	1063
024	R08		**1126**	11	1142	36	1126.6	**1126**	42.3	1127.2	**1126**	5.7	1138.7	**1126**	4.2	1137.9	**1126**	3.8	1126
025	R09		**1021**	6	1046	36	1021.1	**1021**	43.1	1023.0	**1021**	5.8	1024.4	**1021**	5.2	1026.0	**1021**	4.3	1021
026	R05	(f)	**1217**	25	1223	42	**1217.0**	**1217**	21.1	**1217.0**	**1217**	4.0	**1217.0**	**1217**	2.3	**1217.0**	**1217**	2.2	1217
027	R06		**1290**	7	**1290**	41	**1290.0**	**1290**	21.3	**1290.0**	**1290**	4.0	**1290.0**	**1290**	2.6	**1290.0**	**1290**	2.1	1290
028	R07		**1230**	30	**1230**	41	**1230.0**	**1230**	21.2	**1230.0**	**1230**	3.9	**1230.0**	**1230**	2.1	**1230.0**	**1230**	1.9	1230
029	R08		**1261**	13	1262	44	**1261.0**	**1261**	21.1	**1261.0**	**1261**	4.0	**1261.0**	**1261**	2.3	**1261.0**	**1261**	2.0	1261
030	R09		**1134**	5	**1134**	43	**1134.0**	**1134**	21.8	**1134.0**	**1134**	4.0	**1134.0**	**1134**	2.6	**1134.0**	**1134**	2.5	1134
Média			962.7	39.0	976.0	33.9	963.4	963.2	18.4	963.3	962.8	4.8	964.1	962.8	4.1	964.6	962.8	2.9	962.7

As soluções ótimas de todas as instâncias, encontradas por Iori & Riera-Ledesma [2015] e reportadas na Tabela 3.10, são repetidas na Tabela 3.13 para facilitar a comparação. Os métodos heurísticos de Silveira et al. [2015] foram capazes de determinar a solução ótima para 10 (33.3%) instâncias, já os métodos heurísticos propostos neste trabalho conseguiram resolver com otimalidade 27 (90.0%), 29 (96.6%), 29 (96.6%) e 28 (93.3%) instâncias, considerando, respectivamente, os algoritmos SA_1, SA_2, VNS e ILS. Note que através dos métodos aqui propostos a solução ótima foi obtida para todas as instâncias de pequeno porte, pois para a única instância (ID 015) que os algoritmos SA_2 e VNS não encontraram a solução ótima, tal solução foi encontrada pelo algoritmo SA_1.

Ainda analisando os resultados reportados na Tabela 3.13, é possível notar que os métodos heurísticos propostos superaram todos os resultados obtidos por Silveira et al. [2015], encontrando soluções de mesma ou maior qualidade com menos tempo computacional (exceto para o tempo de execução do SA_1 nas instâncias com ID 021 até ID 025, que gastaram cerca de 6 segundos a mais). Comparando com os resultados de Iori & Riera-Ledesma [2015], constata-se que, para algumas instâncias, os algoritmos propostos se mostraram mais eficientes com relação ao tempo de execução, principalmente para as instâncias do tipo *d* (ID 016 até ID 020).

Comparando a qualidade das soluções encontradas pelos algoritmos propostos, é possível notar que, em geral (última linha da Tabela 3.13), todos os algoritmos se mostraram eficientes, sendo que o algoritmo ILS apresentou menor tempo de execução e o algoritmo SA_2 apresentou melhor média de soluções (coluna Avg).

Tabela 3.14: Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMS, considerando as instâncias de grande porte do grupo C.

Instância			Iori & Riera-Ledesma [2015]		Silveira et al. [2015]		SA_1			SA_2			VNS			ILS			Melhor solução conhecida
ID	R	T	Best	t(s)	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	
031	R05	(g)	**950**	6	1083	31	**950.0**	**950**	34.1	950.2	**950**	20.6	982.7	**950**	21.9	973.9	**950**	13.1	950
032	R06		**1024**	83	1087	31	1033.2	**1024**	35.2	1033.7	**1024**	21.3	1035.8	**1024**	25.0	1035.2	**1024**	16.9	1024
033	R07		**932**	46	1054	32	933.3	**932**	35.5	**932.0**	**932**	20.6	938.4	**932**	20.9	943.1	**932**	14.9	932
034	R08		**1018**	125	1128	31	1033.1	1031	35.6	1031.7	1024	20.6	1043.0	1031	21.7	1035.8	1029	15.0	1018
035	R09		**919**	557	985	31	928.0	**919**	36.0	924.2	**919**	21.6	929.1	**919**	30.4	935.5	**919**	17.8	919
036	R05	(h)	**1147**	41	1198	41	**1147.0**	**1147**	68.1	**1147.0**	**1147**	8.2	1154.7	**1147**	6.2	1151.7	**1147**	6.0	1147
037	R06		**1177**	11	1190	42	**1177.0**	**1177**	66.7	1177.3	**1177**	8.4	**1177.0**	**1177**	9.7	1178.5	**1177**	8.2	1177
038	R07		**1123**	21	1136	41	**1123.0**	**1123**	64.8	**1123.0**	**1123**	8.2	**1123.0**	**1123**	6.3	**1123.0**	**1123**	5.4	1123
039	R08		**1184**	27	1214	41	**1184.0**	**1184**	67.3	**1184.0**	**1184**	8.3	1193.2	**1184**	8.6	1188.6	**1184**	5.7	1184
040	R09		**1080**	34	1111	41	1080.3	**1080**	67.1	1082.1	**1080**	8.4	1081.2	**1080**	8.3	1080.5	**1080**	7.1	1080
041	R05	(i)	**1269**	224	1292	43	1270.0	**1269**	39.1	1274.3	**1269**	6.0	1273.0	**1269**	5.1	1273.0	**1269**	4.0	1269
042	R06		**1275**	50	1282	43	**1275.0**	**1275**	38.0	1278.6	**1275**	6.1	1275.2	**1275**	5.4	**1275.0**	**1275**	5.8	1275
043	R07		**1261**	134	1284	43	**1261.0**	**1261**	38.4	**1261.0**	**1261**	6.0	1268.4	**1261**	4.6	1271.9	**1261**	3.8	1261
044	R08		**1310**	364	1338	44	**1310.0**	**1310**	38.1	1312.6	**1310**	6.0	1318.7	**1310**	4.9	1317.1	**1310**	4.5	1310
045	R09		**1157**	41	1196	44	**1157.0**	**1157**	38.3	**1157.0**	**1157**	6.1	1159.5	**1157**	5.3	1160.4	**1157**	4.5	1157
046	R05	(j)	**1012**	75	1126	34	1012.7	**1012**	8.8	1013.7	**1012**	28.5	1030.8	**1012**	33.2	1021.0	**1012**	21.8	1012
047	R06		**1018**	15	1177	33	1033.8	**1018**	7.5	1042.0	**1018**	28.2	1048.4	**1018**	34.9	1046.3	**1018**	28.0	1018
048	R07		**1047**	193	1162	34	1061.9	1054	7.5	1054.2	**1047**	28.1	1059.8	**1047**	34.4	1062.4	1051	20.3	1047
049	R08		**1045**	1192	1221	10	1067.6	1051	8.7	1053.9	**1045**	28.2	1070.4	1050	34.1	1073.7	1050	23.8	1045
050	R09		**973**	767	1126	34	982.6	974	9.6	979.1	974	29.2	981.1	**973**	46.0	978.5	**973**	25.1	973
051	R05	(k)	**1174**	579	1212	45	1183.1	**1174**	56.7	1180.5	**1174**	12.6	1185.6	**1174**	12.9	1186.2	**1174**	9.9	1174
052	R06		**1200**	43	1253	44	1211.7	**1200**	54.1	1213.4	**1200**	12.4	1229.3	**1200**	12.4	1226.9	**1200**	11.8	1200
053	R07		**1243**	1049	1289	44	1245.7	**1243**	55.3	1245.3	**1243**	12.4	1250.7	**1243**	11.6	1253.9	**1243**	11.0	1243
054	R08		**1206**	1427	1275	44	1211.5	**1206**	55.6	1210.8	**1206**	12.5	1210.7	**1206**	13.9	1209.6	**1206**	12.0	1206
055	R09		**1144**	900	1196	46	1147.3	**1144**	56.4	1147.2	**1144**	12.6	1149.3	**1144**	12.5	1155.3	**1144**	10.1	1144

Continua na próxima página

Tabela 3.14: (continuação) Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMS, considerando as instâncias de grande porte do grupo C.

Instância			Iori & Riera-Ledesma [2015]		Silveira et al. [2015]		SA_1			SA_2			VNS			ILS			Melhor solução conhecida
ID	R	T	Best	t(s)	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	
056	R05	(l)	**1293**	991	1347	49	1293.6	**1293**	59.7	1294.3	**1293**	8.7	1307.5	**1293**	7.2	1299.3	**1293**	6.1	1293
057	R06		**1340**	132	1374	49	1340.2	**1340**	59.1	1342.7	**1340**	8.6	1341.3	**1340**	9.1	1345.8	**1340**	7.9	1340
058	R07		**1373**	2043	1418	49	**1373.0**	**1373**	60.3	1374.6	**1373**	8.7	1382.3	**1373**	7.9	1379.4	**1373**	6.6	1373
059	R08		**1305**	313	1345	49	1310.6	**1305**	59.7	1305.5	**1305**	8.7	1317.8	**1305**	7.0	1319.3	**1305**	7.7	1305
060	R09		**1258**	702	1310	49	**1258.0**	**1258**	60.7	1259.7	**1258**	8.7	**1258.0**	**1258**	7.3	1258.6	**1258**	6.4	1258
061	R05	(m)	**1067**	1 h	1220	35	1087.1	**1067**	15.9	1084.5	**1067**	59.8	1092.5	1069	87.8	1093.5	**1067**	58.9	1067
062	R06		**1093**	173	1324	35	1107.5	**1093**	17.1	1100.0	**1093**	58.7	1103.3	**1093**	92.4	1117.3	**1093**	53.7	1093
063	R07		**1098**	206	1325	35	1105.2	1103	14.8	1103.5	**1098**	57.3	1129.0	**1098**	70.2	1127.3	**1098**	48.3	1098
064	R08		**1121**	1184	1300	34	1152.4	**1121**	15.8	1154.3	1133	58.7	1155.5	1133	88.1	1148.8	**1121**	59.1	1121
065	R09		**1047**	1 h	1266	34	1056.0	**1047**	25.8	1058.3	**1047**	62.1	1077.1	**1047**	103.5	1088.0	**1047**	67.8	1047
066	R05	(n)	1480♣	1 h	1387	47	1271.4	**1262**	7.7	1269.1	**1262**	24.5	1269.6	**1262**	31.5	1273.9	**1262**	26.1	1262
067	R06		1692♣	1 h	1438	48	1314.6	**1304**	7.5	1310.6	**1304**	24.0	1310.1	**1304**	33.3	1312.6	**1304**	22.1	1304
068	R07		1652♣	1 h	1429	46	1342.7	**1333**	7.7	1343.3	1338	23.9	1344.2	1338	33.5	1343.8	1338	25.3	1333
069	R08		**1297**	1 h	1426	47	1315.5	**1297**	7.7	1304.7	**1297**	23.8	1318.9	**1297**	30.4	1324.3	**1297**	26.9	1297
070	R09		1539♣	1 h	1346	47	1254.5	1243	7.8	1251.2	**1240**	24.4	1247.0	**1240**	36.0	1246.8	1243	23.0	1240
071	R05	(o)	1761♣	1 h	1422	53	**1367.0**	**1367**	122.9	1371.0	**1367**	15.7	1369.5	**1367**	18.1	1373.4	**1367**	17.9	1367
072	R06		1661♣	1 h	1516	53	1448.2	**1448**	119.9	1448.2	**1448**	15.4	1448.9	**1448**	15.8	1450.6	**1448**	13.9	1448
073	R07		1864♣	1 h	1509	48	**1465.0**	**1465**	121.1	1470.6	**1465**	15.5	1477.9	**1465**	16.9	1477.6	**1465**	17.8	1465
074	R08		1694♣	1 h	1488	52	1447.5	**1444**	124.3	1449.7	**1444**	15.4	1448.7	**1444**	16.0	1448.2	**1444**	15.3	1444
075	R09		1770♣	1 h	1413	52	1363.4	**1362**	123.7	1364.6	**1362**	15.6	**1362.0**	**1362**	15.7	**1362.0**	**1362**	16.2	1362
Média			1251.0	1265.5	1271.5	41.3	1193.2	1187.3	45.8	1192.6	1187.3	20.4	1198.4	1187.6	25.7	1198.8	1187.4	18.5	1186.8

♣ Soluções obtidas a partir da formulação ILP descrita na Seção 3.3.

Os resultados obtidos para as instâncias de grande porte do grupo C, descritos na Tabela 3.14, também mostram a superioridade dos métodos heurísticos aqui propostos com relação aos métodos heurísticos proposto por Silveira et al. [2015], pois melhores soluções foram encontradas para todas as instâncias, embora o tempo de processamento tenha sido maior para alguns tipos de instâncias. Também nota-se a eficiência dos métodos heurísticos aqui propostos quando comparados aos métodos exatos de Iori & Riera-Ledesma [2015], pois das 33 instâncias de grande porte que se conhece a solução ótima, os métodos heurísticos encontraram a solução ótima para 28 (84.8%), 30 (90.9%), 30 (90.9%) e 30 (90.9%) instâncias, respectivamente, considerando os algoritmos SA_1, SA_2, VNS e ILS. É importante observar que, juntos, os quatros algoritmos heurísticos propostos encontraram a solução ótima para 32 das 33 instâncias; apenas a instância de ID 034 não foi resolvida na otimalidade.

Ainda sobre a Tabela 3.14, além da alta qualidade das soluções alçancadas, os algoritmos aqui propostos se mostraram eficientes comparados aos métodos já introduzidos na literatura. Para todas as instâncias, os algoritmos SA_2, VNS e ILS gastaram menos tempo de processamento comparado ao tempo gasto pelo método heurístico de Silveira et al. [2015], e com relação ao método exato de Iori & Riera-Ledesma [2015] os quatros algoritmos aqui propostos se mostraram mais eficientes para a maioria das instâncias, considerando o tempo de execução e a qualidade das soluções.

A Figura 3.24 mostra o gráfico de caixa (*boxplot*) no qual é possível avaliar a distribuição e variação dos resultados obtidos pelos diferentes métodos de resolução propostos para o DVRPMS. Por este gráfico se nota que os métodos heurísticos propostos neste trabalho apresentam pouca variação nos resultados, diferentemente dos melhores resultados de Silveira et al. [2015] e Iori & Riera-Ledesma [2015].

Para analisar se há diferença estatística significante entre os melhores resultados (colunas Best) obtidos para as instâncias do grupo C (Tabelas 3.13 e 3.14) pelo diferentes métodos de resolução do DVRPMS, foi realizada uma análise de variância (ANOVA), sendo utilizado como variável resposta o desvio percentual relativo (*Relative Percentage Deviation* - RPD) entre o valor objetivo da melhor solução encontrada (f_b) e o valor objetivo da solução de cada método (f_m), calculado por:

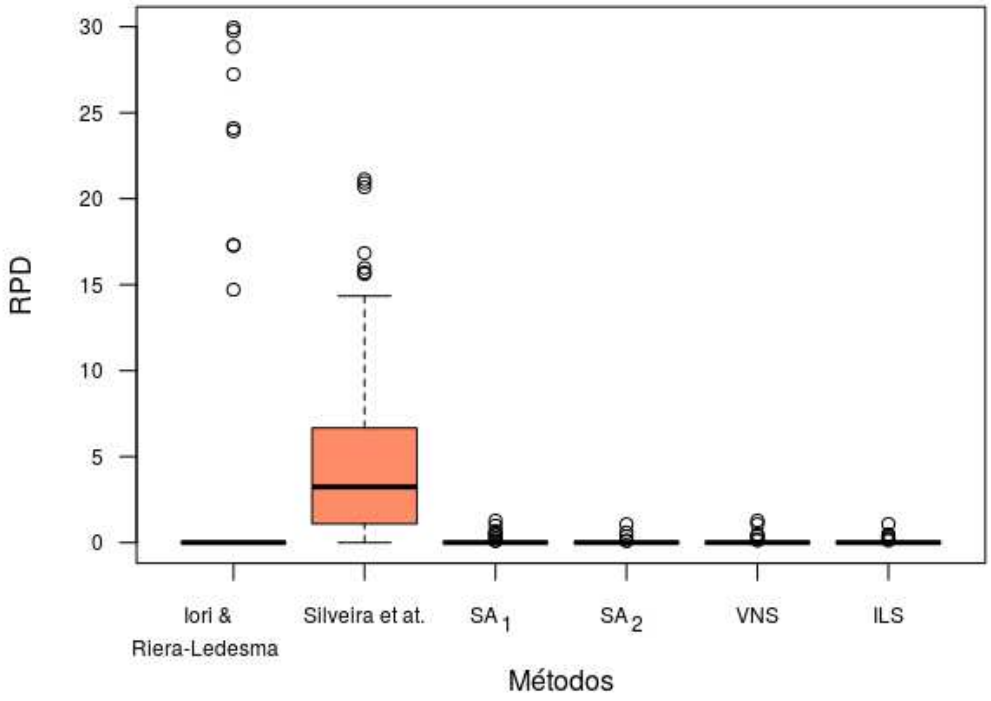


Figura 3.24: Comparação entre as médias do RPD alcançados pelos diferentes métodos para o DVRPMS, considerando as instâncias do grupo C.

$$RPD = \frac{f_m - f_b}{f_m} \ 100\% \tag{3.34}$$

A ANOVA indicou haver uma diferença significativa entre os resultados com nível de confiança de 99%, assim foi realizado o teste de Scott & Knott [1974] para descobrir entre quais métodos ocorre essa diferença. A Tabela 3.15 informa os resultados do teste de Scott & Knott [1974], onde é atribuído um mesmo identificador (coluna Grupos) para os tratamentos (algoritmos) que obtiveram resultados que não apresentam diferença significativa entre si. Assim, não há diferença significativa entre os resultados alcançados pelos métodos propostos neste trabalho (identificados por "c"), mas os resultados de Silveira et al. [2015] e os resultados alcançados pelos métodos exatos apresentam diferença significativa para todos os demais resultados. Como os métodos aqui propostos obtiveram as menores médias de RPD e apresentaram diferença significativa comparados aos

demais métodos, é possível afirmar que os métodos propostos são significativamente melhores que os métodos da literatura.

Tabela 3.15: Resultado do teste de Scott & Knott para o RPD alcançados pelos diferentes métodos de resolução para o DVRPMS, considerando as instâncias do grupo C.

Grupos	Métodos	Médias
a	Silveira et al.	5.0857
b	Iori & Riera-Ledesma	2.8415
c	SA_1	0.0647
c	VNS	0.0467
c	ILS	0.0383
c	SA_2	0.0301

Para as instâncias do grupo $\neg C$, todos os resultados descritos na Tabela 3.16 são referentes apenas aos métodos que foram propostos neste trabalho, uma vez que Silveira et al. [2015] reportam apenas os resultados para as instâncias do grupo C e também pelo fato de que os resultados descritos por Iori & Riera-Ledesma [2015] ainda (fevereiro de 2017) não foram corrigidos pelos autores.

Tabela 3.16: Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMS, considerando as instâncias do grupo $\neg C$.

Instância			Formulação ILP		SA_1			SA_2			VNS			ILS			Melhor solução conhecida
ID	R	T	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	
076	R05	(p)	916	1 h	930.9	921	53.6	923.1	921	61.7	930.7	**914**	455.4	918.3	**914**	1414.3	914
077	R06		988	1 h	1003.5	1003	34.2	997.6	989	63.7	971.6	**948**	526.2	974.9	956	1988.2	948
078	R07		**803**	1 h	820.7	804	390.4	851.6	804	70.3	810.1	804	653.3	807.6	804	2080.9	803
079	R08		**902**	1531.0	962.0	909	517.8	976.9	905	62.2	914.0	**902**	511.9	909.9	**902**	2135.2	902
080	R09		**874**	1 h	889.1	884	340.7	883.4	880	63.4	882.4	**874**	431.6	880.0	880	1537.3	874
081	R05	(q)	**942**	1 h	974.5	973	29.3	969.0	965	63.9	959.8	**942**	408.8	947.0	**942**	1353.2	942
082	R06		1032	1 h	1020.8	**1010**	31.6	1013.6	**1010**	64.3	1022.3	**1010**	363.4	1021.8	**1010**	1262.0	1010
083	R07		993	1 h	**931.0**	**931**	30.2	931.2	**931**	68.5	933.0	**931**	371.8	931.6	**931**	1239.7	931
084	R08		1045	1 h	1002.5	998	30.2	998.0	998	64.9	1011.6	**995**	432.4	996.8	**995**	1317.3	995
085	R09		902	1 h	897.9	**889**	32.5	896.5	891	65.4	892.0	**889**	383.7	889.1	**889**	1510.6	889
086	R05	(r)	1221	1 h	1121.4	1116	31.1	1113.8	**1113**	47.1	1116.7	1115	135.1	1113.6	**1113**	593.4	1113
087	R06		1128	1 h	1134.3	1134	30.8	1121.2	**1121**	47.1	1122.7	**1121**	140.3	**1121.0**	**1121**	634.4	1121
088	R07		1169	1 h	**1042.0**	**1042**	32.0	**1042.0**	**1042**	49.9	1074.7	**1042**	152.6	1048.9	**1042**	729.4	1042
089	R08		1316	1 h	1149.4	**1147**	31.3	1154.1	**1147**	46.7	1174.1	1174	157.8	1171.3	**1147**	600.6	1147
090	R09		1091	1 h	1040.7	1035	31.3	1036.7	**1033**	49.7	1040.3	**1033**	215.0	1038.9	1034	664.4	1033
091	R05	(s)	**950**	1 h	1001.7	989	44.0	988.9	974	90.8	970.9	**950**	487.8	953.3	**950**	2874.2	950
092	R06		**1005**	1 h	1009.5	**1005**	55.2	**1005.0**	**1005**	90.6	1018.9	**1005**	505.4	**1005.0**	**1005**	1903.1	1005
093	R07		**952**	1 h	984.9	963	278.7	992.0	963	91.2	978.4	960	688.0	964.8	956	2254.9	952
094	R08		963	1 h	1008.6	1007	57.8	996.8	967	92.7	991.0	963	554.8	967.7	**958**	3061.3	958
095	R09		960	1 h	946.4	942	42.6	941.3	**940**	95.3	941.1	**940**	511.6	940.4	**940**	2102.0	940
096	R05	(t)	1044	1 h	1008.8	**998**	39.4	1006.3	**998**	89.5	1008.8	**998**	489.1	999.0	**998**	2256.8	998
097	R06		1046	1 h	1037.3	1018	38.5	1054.1	**1017**	79.0	1044.5	1034	502.7	1035.2	**1017**	1962.7	1017
098	R07		1091	1 h	1025.9	**1021**	38.2	**1021.0**	**1021**	95.2	1035.3	**1021**	484.8	1027.1	**1021**	1891.5	1021
099	R08		1136	1 h	1029.6	1025	42.3	1034.5	1025	90.9	1043.0	1025	544.0	1022.9	**1019**	2392.3	1019
100	R09		1014	1 h	964.2	960	37.8	960.3	956	91.6	955.3	**953**	579.6	955.4	**953**	2156.6	953

Continua na próxima página

Tabela 3.16: (continuação) Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMS, considerando as instâncias do grupo $\neg C$.

Instância			Formulação ILP		SA_1			SA_2			VNS			ILS			Melhor solução conhecida
ID	R	T	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	Avg	Best	t(s)	
101	R05	(u)	1256	1 h	1165.5	**1149**	37.7	1149.1	**1149**	63.6	1155.5	**1149**	167.5	**1149.0**	**1149**	685.5	1149
102	R06		1348	1 h	1179.7	**1161**	37.6	**1161.0**	**1161**	64.5	1171.8	**1161**	213.4	**1161.0**	**1161**	931.6	1161
103	R07		1416	1 h	**1232.0**	**1232**	37.3	**1232.0**	**1232**	64.5	1232.8	**1232**	185.0	**1232.0**	**1232**	735.0	1232
104	R08		1356	1 h	1206.8	1200	39.0	1189.6	**1181**	62.7	1202.3	1200	193.5	1196.2	**1181**	745.9	1181
105	R09		1210	1 h	1120.4	**1118**	39.3	1121.3	**1118**	63.9	1139.4	**1118**	192.6	1139.7	1130	768.2	1118
106	R05	(v)	1090	1 h	1053.8	1043	110.8	1049.2	1043	185.4	1051.5	**1033**	895.0	1036.6	**1033**	4191.1	1033
107	R06		1193	1 h	1079.6	**1051**	260.8	1072.2	**1051**	184.0	1072.2	**1051**	903.9	1073.2	**1051**	3283.1	1051
108	R07		1210	1 h	1085.3	1080	145.1	1081.6	1080	180.4	1087.3	1080	839.0	1081.6	**1074**	4322.0	1074
109	R08		1204	1 h	1080.1	**1069**	945.8	1079.4	**1069**	181.9	1088.4	1080	1250.0	1079.8	1076	4480.6	1069
110	R09		1077	1 h	1029.8	1024	567.0	1024.4	**1019**	198.2	1026.9	**1019**	1059.9	1022.3	1022	5086.2	1019
111	R05	(w)	1244	1 h	1143.1	1073	56.8	1168.4	1094	140.0	1091.3	**1067**	875.3	1083.3	**1067**	4478.5	1067
112	R06		1537	1 h	1200.2	1164	53.9	1200.3	1196	135.0	1131.6	**1093**	1072.1	1100.3	**1093**	4890.4	1093
113	R07		1376	1 h	1147.7	1103	60.8	1173.3	1114	142.3	1127.2	1103	785.3	1104.3	**1098**	5045.8	1098
114	R08		1511	1 h	1157.1	1133	65.4	1143.1	1135	148.7	1147.9	**1124**	1104.4	1148.5	1141	4439.0	1124
115	R09		1335	1 h	1061.5	1048	64.8	1070.3	1058	151.7	1091.3	**1047**	1242.6	1054.1	**1047**	8707.8	1047
116	R05	(x)	1618	1 h	1287.7	1268	57.0	1273.7	1264	105.7	1269.7	**1262**	325.7	1266.6	**1262**	1729.1	1262
117	R06		1959	1 h	1315.5	1306	55.3	1308.8	**1304**	107.0	1317.4	**1304**	344.5	1313.0	**1304**	1931.3	1304
118	R07		1757	1 h	1349.6	1342	54.3	1341.0	**1338**	109.5	1343.5	**1338**	340.2	1346.0	**1338**	1561.8	1338
119	R08		1957	1 h	1333.3	**1297**	66.3	1337.9	**1297**	104.5	1342.7	1317	293.3	1311.5	**1297**	1549.7	1297
120	R09		1705	1 h	1253.0	**1240**	56.2	1241.6	**1240**	106.9	1255.4	**1240**	368.4	1250.9	**1240**	1920.4	1240
Média			1196.5	3554.0	1076.0	1062.8	114.1	1073.9	1061 3	95.5	1070.9	1056.9	518.6	1062.0	1055.4	2297.8	1054.1

Pelos resultados descritos na Tabela 3.16, é possível notar a dificuldade de resolver o DVRPMS de forma exata através na formulação ILP proposta. Utilizando essa formulação apenas uma instância (ID 079) foi resolvida na otimalidade. Também é possível notar que apenas em duas instâncias (ID 078 e ID 093) a formulação ILP encontrou unicamente as soluções melhores dentre todos os métodos comparados.

Os quatro métodos heurísticos propostos não foram tão eficientes para as 45 instâncias do grupo $\neg C$, quanto para as instâncias do grupo C. O SA_1 encontrou as melhores soluções para 16 (35.6%) instâncias, enquanto o SA_2, VNS e ILS encontraram, respectivamente, 24 (53.3%), 33 (73.3%), 36 (80.0%) das instâncias. Embora o ILS tenha apresentado o melhor desempenho geral, o tempo de execução gasto para computar as instâncias foi relativamente alto, demorando até 2.4 horas para processar uma certa instância (ID 115).

A Figura 3.25 mostra o gráfico de caixa (*boxplot*) no qual é possível observar que os métodos heurísticos propostos apresentam menor variação nos resultados que a formulação ILP.

Analisando melhor os resultados obtidos para as instâncias descritas na Tabela 3.16 a fim de determinar se existe alguma diferença estatística entre os melhores resultados encontrados por cada método (colunas Best), foi realizada uma análise de variância (ANOVA), sendo utilizado como variável resposta o RPD (equação 3.34) entre o valor objetivo da melhor solução encontrada (f_b) e o valor objetivo da solução de cada método (f_m).

Com um nível de confiança de 99% a ANOVA indicou haver uma diferença significativa entre os resultados. E, a fim de descobrir entre quais métodos ocorre essa diferença foi realizado o teste de Scott & Knott [1974]. A Tabela 3.17 informa os resultados do teste de Scott & Knott [1974], indicando não haver diferença significativa entre os resultados alcançados pelos métodos propostos neste trabalho, mas indicando uma diferença estatística entre os resultados obtidos pela formulação ILP e todos os demais resultados.

A Tabela 3.18 apresenta um resumo dos resultados alcançados por todos os métodos já propostos para resolver o DVRPMS. Cada linha da tabela se refere às cinco instâncias de cada tipo (R05, R06, R07, R08 e R09). A coluna # informa o número de instâncias que cada método foi capaz de encontrar a melhor solução

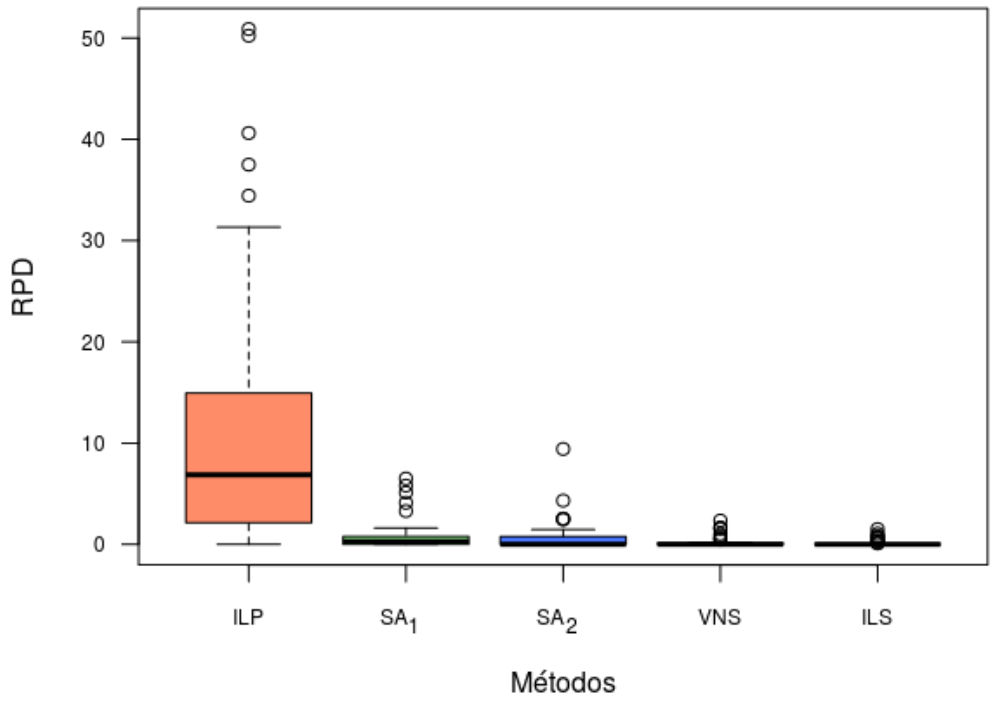


Figura 3.25: Comparação entre as médias do RPD alcançados pelos diferentes métodos para o DVRPMS, considerando as instâncias do grupo $\neg C$.

Tabela 3.17: Resultado do teste de Scott & Knott para o RPD alcançados pelos diferentes métodos de resolução para o DVRPMS, considerando as instâncias do grupo $\neg C$.

Grupos	Métodos	Médias
a	ILP	12.3000
b	SA_1	0.8623
b	SA_2	0.7078
b	VNS	0.2550
b	ILS	0.1268

conhecida e a coluna t(s) informa a tempo médio de execução das cinco instâncias. Por esta tabela, verifica-se a eficiência de todos métodos heurísticos propostos neste trabalho, os quais obtiveram melhor desempenho que o método proposto por Silveira et al. [2015] para todos os tipos de instâncias. Juntos, os quatro algoritmos propostos (duas últimas colunas), foram capazes de encontrar as melhores soluções para 117 das 120 instâncias.

Tabela 3.18: Número de melhores resultados alcançados por algoritmo.

Tipo	Iori & Ledesma		Silveira et al.		ILP		SA_1		SA_2		VNS		ILS		$SA_1 \cup SA_2 \cup$ VNS $\cup$ ILS	
	#	t(s)	#	t(s)	#	t(s)	#	t(s)	#	t(s)	#	t(s)	#	t(s)	#	t(s)
(a)	5	1.0	2	30.8	5	169.2	5	26.6	5	3.3	5	2.1	5	1.7	5	33.7
(b)	5	1.8	1	31.0	5	61.6	5	5.8	5	3.7	5	3.2	5	2.1	5	14.8
(c)	5	0.2	3	35.0	5	206.8	5	11.4	4	2.1	4	1.1	4	0.9	5	15.5
(d)	5	203.6	0	27.8	1	3294.4	2	2.8	5	9.7	5	11.1	4	6.6	5	30.2
(e)	5	11.2	1	36.6	0	1 h	5	42.3	5	5.7	5	4.8	5	3.8	5	56.6
(f)	5	16.0	3	42.2	0	1 h	5	21.3	5	4.0	5	2.4	5	2.1	5	29.8
(g)	5	163.4	0	31.2	1	3334.2	4	35.3	4	20.9	4	24.0	4	15.5	4	95.7
(h)	5	26.8	0	41.2	0	1 h	5	66.8	5	8.3	5	7.8	5	6.5	5	89.4
(i)	5	162.6	0	43.4	0	1 h	5	38.4	5	6.0	5	5.1	5	4.5	5	54.0
(j)	5	448.4	0	29.0	0	1 h	2	8.4	4	28.4	4	36.5	3	23.8	5	97.1
(k)	5	799.6	0	44.6	0	1 h	5	55.6	5	12.5	5	12.7	5	11.0	5	91.8
(l)	5	836.2	0	49.0	0	1 h	5	59.9	5	8.7	5	7.7	5	6.9	5	83.2
(m)	5	1752.6	0	34.6	0	1 h	4	17.9	4	59.3	3	88.4	5	57.6	5	223.2
(n)	1	1 h	0	47.0	0	1 h	4	7.7	4	24.1	4	32.9	3	24.7	5	89.4
(o)			0	51.6	0	1 h	5	122.4	5	15.5	5	16.5	5	16.2	5	170.6
(p)					3	3186.3	0	267.3	0	64.3	4	515.7	2	1831.2	4	2678.5
(q)					1	1 h	3	30.8	2	65.4	5	392.0	5	1336.6	5	1824.8
(r)					0	1 h	2	31.3	5	48.1	3	160.2	4	644.4	5	884.0
(s)					3	1 h	1	95.7	2	92.1	3	549.5	4	2439.1	4	3176.4
(t)					0	1 h	2	39.2	3	89.2	3	520.0	5	2132.0	5	2780.4
(u)					0	1 h	4	38.2	5	63.8	4	190.4	4	773.2	5	1065.6
(v)					0	1 h	2	405.9	3	186.0	3	989.6	3	4272.6	5	5236.3
(w)					0	1 h	0	60.3	0	143.5	4	1015.9	4	5512.3	5	4819.7
(x)					0	1 h	2	57.8	4	106.7	4	334.4	5	1738.5	5	2237.4
Soma	66	-	10	-	24	-	82	-	94	-	102	-	104	-	117	-
Média	-	573.1	-	38.3	-	3127.2	-	64.5	-	44.6	-	205.2	-	869.3	-	1183.7

Capítulo 4

Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas e Demanda Heterogênea

O Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas e Demanda Heterogênea (*Double Vehicle Routing Problem with Multiple Stacks and Heterogeneous Demand* - DVRPMSHD) foi recentemente proposto por Chagas & Santos [2016] como fruto desta pesquisa. Esse problema, assim como o DVRPMS, é um problema de roteamento de veículos com coleta e entrega (*one-to-one*) em que é preciso atender um determinado conjunto de clientes utilizando uma frota de veículos que têm seus compartimentos de carga divididos em pilhas.

O DVRPMSHD pode ser considerado como uma generalização do DVRPMS e sua proposição foi motivada pelo fato de se tratar um caso mais realístico do que o DVRPMS, em que os clientes podem ter demandas heterogêneas, isto é, a cada cliente podem estar associado vários produtos e esse número de produtos ser independente dos demais clientes. O DVRPMSHD herda as características e restrições do DVRPMS, e apresenta uma restrição adicional que obriga que todos os produtos de um mesmo cliente sejam transportados por um mesmo veículo.

As Figuras 4.1 e 4.2 mostram dois exemplos de soluções para um caso que envolve 12 clientes. A cada cliente é associado um valor de demanda que é indicado pelo número ao lado de cada vértice. A Figura 4.1 representa uma solução que

utiliza três veículos com contêineres de dimensões (2×4), (2×3) e (2×3), sendo que nesse exemplo o somatório das demandas dos clientes é igual à capacidade dos veículos. Já na Figura 4.2 a demanda total é menor que a capacidade dos três veículos com contêineres de dimensões (2×3), (2×3) e (3×4), fazendo com que sobrem espaços nesses contêineres.

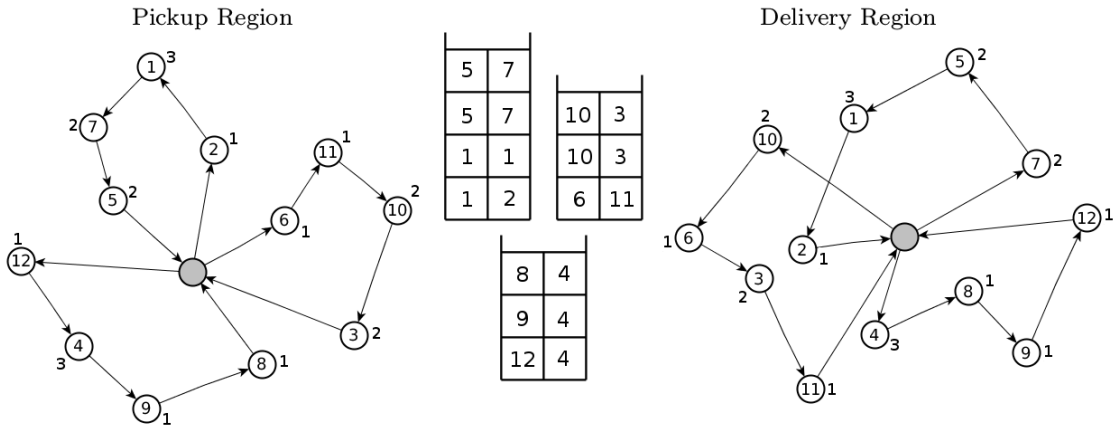


Figura 4.1: Exemplo de uma solução viável para o DVRPMSHD utilizando uma frota de veículos heterogênea e com os contaneirs completamente preenchidos.
Fonte: Chagas & Santos [2016].

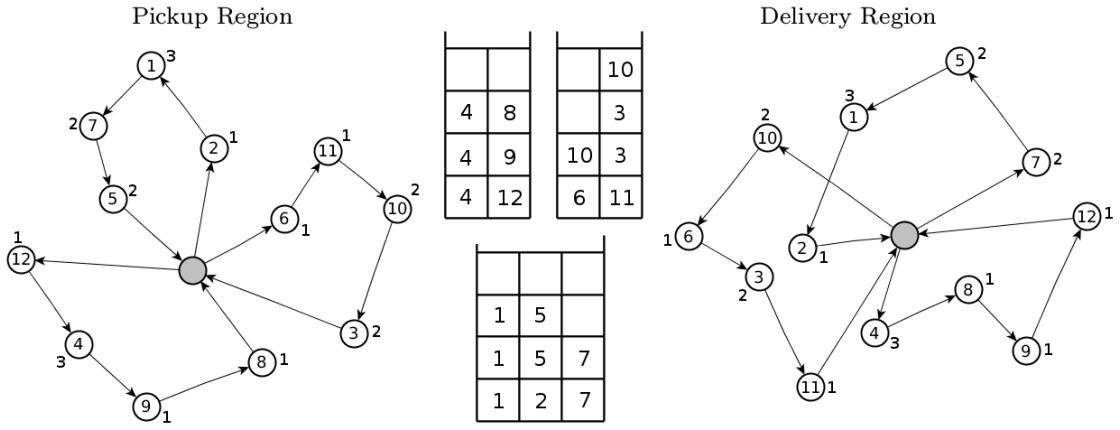


Figura 4.2: Exemplo de uma solução viável para o DVRPMSHD utilizando uma frota de veículos heterogênea e com contaneirs não preenchidos completamente.
Fonte: Adaptado de Chagas & Santos [2016].

Nas duas diferentes soluções ilustradas pelas figuras descritas acima, a cada veículo é associado uma rota na região de coleta, uma rota na região de entrega e um plano de carregamento do contêiner. As rotas de cada veículo devem respeitar a política LIFO estabelecida pelo plano de carregamento em cada contêiner. Note que a demanda de um cliente pode estar dividida em pilhas diferentes, desde que pertençam ao mesmo veículo.

Observe que, assim como no DTSPMS e no DVRPMS, no DVRPMSHD é utilizado os termos "altura" e "topo" referenciando à profundidade do contêiner e à parte traseira do veículo, respectivamente.

A seguir o DVRPMSHD é formalmente definido e posteriormente, no decorrer deste capítulo, mais informações, detalhes e propostas de algoritmos para resolver o problema são apresentadas.

4.1 Definição formal

Semelhantemente ao DVRPMS, o DVRPMSHD pode ser formalmente definido através de um conjunto de requisições de transporte $I = \{1, 2, ..., n\}$, sendo que cada requisição é realizada por um cliente. Considere $V_c^P = \{1^P, 2^P, ..., n^P\}$ o conjunto de vértices que representam as localizações dos clientes na região de coleta e considere também $V_c^D = \{1^D, 2^D, ..., n^D\}$ o conjunto de vértices que representam as localizações dos clientes na região de entrega. A cada requisição $i \in I$ é associado um vértice i^P na região de coleta, um vértice i^D na região de entrega e um valor d_i que representa a demanda da requisição, isto é, o número de produtos que devem ser carregado no vértice i^P, transportado da região de coleta para a região de entrega e, por fim, descarregado no vértice i^D.

Os grafos $G^P = (V^P,\ E^P)$ e $G^D = (V^D,\ E^D)$, onde $V^P = V_c^P \cup \{0^P\}$ e $V^D = V_c^D \cup \{0^D\}$, representam, respectivamente, os vértices na região de coleta e na região de entrega, sendo 0^P e 0^D os depósitos de cada região. O conjunto de arestas nas regiões de coleta e entrega são definidos por $E^P = \{(i^P, j^P, c_{ij}^P)\ \forall i^P \in V^P, \forall j^P \in V^P \mid j^P \neq i^P\}$ e $E^D = \{(i^D, j^D, c_{ij}^D)\ \forall i^D \in V^D, \forall j^D \in V^D \mid j^D \neq i^D\}$, onde c_{ij}^P e c_{ij}^D são, respectivamente, o custo associado às arestas (i^P, j^P) e (i^D, j^D).

Seja K o conjunto de veículos disponíveis para atender as requisições de transporte, onde o compartimento de carga (contêiner) de cada veículo $k \in K$ é dividido em R_k pilhas, sendo que todas as pilhas do veículo k têm a mesma altura L_k. Todos os veículos do conjunto K devem iniciar e finalizar suas rotas nos depósitos definidos em cada região. Os veículos devem coletar todos os produtos dos clientes localizados na região de coleta, armazenar esses produtos nos seus contêineres e depois entregar os produtos aos clientes localizados na região de entrega. As rotas de cada veículo devem satisfazer a política LIFO em todas as pilhas desse veículo, isto é, se um cliente localizado no vértice i^P (região de coleta) é visitado antes que o cliente localizado no vértice j^P (região de coleta), e algum produto da requisição j é armazenado na mesma pilha que algum produto i foi armazenado, então o cliente da requisição j deve ser visitado (vértice j^D) antes do cliente da requisição i (vértice i^D) na região de entrega. Note ainda que para quaisquer clientes i e j, independente da ordem de atendimento, não pode haver uma pilha com os produtos de i sobre os produtos de j e outra com os produtos j sobre os produtos de i.

Com o objetivo de simplificar a notação utilizada pela formulação matemática apresentada adiante, foi utilizado a notação descrita por Iori & Riera-Ledesma [2015], a qual utiliza i para denotar ambos os vértices i^P e i^D, e também (i, j) para denotar ambas as arestas (i^P, j^P) e (i^D, j^D).

O objetivo do DVRPMSHD é atender todas as $|I|$ requisições de transporte, de forma que a distância total percorrida pelos $|K|$ veículos seja a menor possível.

4.2 Formulação de programação linear inteira proposta

A formulação de programação linear inteira proposta para o DVRPMSHD neste trabalho foi baseada na formulação matemática também aqui proposta para o DVRPMS (Seção 3.3), que, por sua vez, foi baseada na formulação matemática do DTSPMS proposta por Petersen & Madsen [2009] e na formulação matemática para o DVRPMS proposta por Iori & Riera-Ledesma [2015]. As variáveis utilizadas são descritas abaixo:

- $x_{i,j}^{kT}$: variável binária que assume 1 se o veículo k utiliza a aresta (i, j) na região T e 0 caso contrário.

- y_{ij}^{T} : variável binária que assume 1 se o vértice i é visitado antes do vértice j na região T e 0 caso contrário.

- w_{i}^{kT} : variável binária que assume 1 se o veículo k atende a requisição i na região T e 0 caso contrário.

- z_{ih}^{rk} : variável binária que assume 1 se o h-ésimo produto referente à requisição i é armazenado na r-ésima pilha do veículo k e 0 caso contrário.

Com as variáveis descritas acima é possível descrever a seguinte formulação de programação linear inteira para o DVRPMSHD:

$$\min \sum_{k \in K} \sum_{T \in \{P,D\}} \sum_{(i,j) \in E^T} c_{ij}^{T} x_{ij}^{kT} \tag{4.1}$$

$$\sum_{j \in V^T} x_{0j}^{kT} = 1 \qquad \forall k \in K, \forall\, T \in \{P, D\} \tag{4.2}$$

$$\sum_{i \in V^T} x_{i0}^{kT} = 1 \qquad \forall k \in K, \forall\, T \in \{P, D\} \tag{4.3}$$

$$\sum_{i \in V^T \setminus \{j\}} x_{ij}^{kT} = w_{j}^{kT} \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall j \in V_c^T \tag{4.4}$$

$$\sum_{j \in V^T \setminus \{i\}} x_{ij}^{kT} = \sum_{j \in V^T \setminus \{i\}} x_{ji}^{kT} \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall i \in V_c^T \tag{4.5}$$

$$\sum_{k \in K} w_{i}^{kT} = 1 \qquad \forall\, T \in \{P, D\}, \forall i \in V_c^T \tag{4.6}$$

$$\sum_{i \in I} \sum_{h=1}^{d_i} z_{ih}^{rk} \leq L_k \qquad \forall k \in K, \forall r = 1..R_k \tag{4.7}$$

$$\sum_{h=1}^{d_i} \sum_{r=1}^{R_k} z_{ih}^{rk} = d_i w_{i}^{kP} \qquad \forall k \in K, \forall i \in I \tag{4.8}$$

$$w_i^{kP} = w_i^{kD} \qquad \forall k \in K, \forall i \in I \tag{4.9}$$

$$y_{ij}^T + y_{ji}^T = 1 \qquad \forall\, T \in \{P, D\}, \forall i \in V_c^T, \forall j \in V_c^T \setminus \{i\} \tag{4.10}$$

$$y_{il}^T + y_{lj}^T \leq y_{ij}^T + 1 \qquad \forall\, T \in \{P, D\}, \forall l \in V_c^T, \forall i \in V_c^T, \forall j \in V_c^T \setminus \{i\} \tag{4.11}$$

$$x_{ij}^{kT} \leq y_{ij}^T \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall i \in V^T, \forall j \in V^T \setminus \{i\} \tag{4.12}$$

$$y_{ij}^P + z_{ih}^{rk} + z_{jh'}^{rk} \leq 3 - y_{ij}^D \qquad \begin{array}{r} \forall k \in K, \forall i \in I, \forall j \in I \setminus \{i\}, \\ \forall h = 1..d_i, \forall h' = 1..d_j, \forall r = 1..R_k \end{array} \tag{4.13}$$

$$\sum_{j \in I} j x_{0j}^{kP} \leq \sum_{j \in I} j x_{0j}^{k'P} \qquad \forall k \in K, \forall k' \in K \mid k < k', R_k = R_{k'}, L_k = L_{k'} \tag{4.14}$$

$$x_{ij}^{kT} \in \{0, 1\} \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall (i, j) \in E^T \tag{4.15}$$

$$y_{ij}^T \in \{0, 1\} \qquad \forall\, T \in \{P, D\}, \forall i \in V^T, \forall j \in V^T \setminus \{i\} \tag{4.16}$$

$$z_{ih}^{rk} \in \{0, 1\} \qquad \forall k \in K, \forall i \in I, \forall h = 1..d_i, \forall r = 1..R_k \tag{4.17}$$

$$w_i^{kT} \in \{0, 1\} \qquad \forall k \in K, \forall\, T \in \{P, D\}, \forall i \in V_c^T \tag{4.18}$$

A função objetivo (4.1) minimiza a distância total percorrida pelos veículos. As restrições (4.2) e (4.3) garantem que cada veículo começa e finaliza sua rota no depósito de cada região. A restrição (4.4) assegura que a aresta (i, j) é utilizada pelo veículo k, somente se a requisição j é atendida pelo veículo k. A restrição (4.5) garante que o mesmo veículo deve chegar e sair de um determinado vértice que representa a localização de um cliente. A restrição (4.6) assegura que cada requisição é atendida por apenas um veículo. A restrição (4.7) assegura que a capacidade das pilhas não seja extrapolada. A restrição (4.8) garante o que todos os produtos de uma requisição atendida por um veículo k devem estar armazenados no contêiner desse veículo. A restrição (4.9) assegura que o mesmo veículo deve atender uma requisição i na região de coleta e na região de entrega. As restrições (4.10) e (4.11), respectivamente, estabelece uma ordem de visita entre todos os pares de vértice em ambas regiões e assegura uma transitividade nesta ordem, ou seja, se i precede l e l precede j, então i precede j. A restrição (4.12) assegura que se uma aresta (i, j) é usada por algum veículo, então i é visitado antes de j. A inequação (4.13) expressa as restrições relativas à política LIFO aplicada em todas as pilhas de todos os veículos, sendo que se os produtos relativos às requisições i e

j são armazenados na mesma pilha r do veículo k, sendo i visitada antes que j na região de coleta, então i não pode ser visitado antes que j na região de entrega. A restrição (4.14) quebra a simetria decorrente da formulação, impondo uma ordem lexicográfica nas rotas dos veículos com a mesma configuração de contêiner. E, finalmente, as restrições (4.15) até a (4.18) definem o escopo e domínio das variáveis de decisão.

Esta formulação descreve completamente o DVRPMSHD e pode ser usada para resolver uma determinada instância do problema através de algum software capaz de resolver formulações de programação inteira. Uma formulação alternativa, mais adequada para ser resolvida por uma técnica de geração de colunas, é descrita na próxima seção.

4.3 Algoritmo exato proposto

Esta seção descreve o algoritmo exato *branch-and-price* (B&P) proposto para o DVRPMSHD, detalhando uma nova formulação matemática para o problema baseada em particionamento de conjuntos. Também são detalhados os componentes da geração de colunas (*pricing*) e os critérios utilizados no processo de ramificação (*branching*).

4.3.1 Formulação baseada em particionamento de conjuntos

Seja $\mathcal{S}_k$ o conjunto de todas as possíveis e viáveis designações de requisições de transporte para o veículo $k \in K$, sendo que para ser viável, uma designação $s \in \mathcal{S}_k$ deve conter um subconjunto das requisições de transportes solicitadas, de forma a garantir que toda a demanda dessas requisições seja alocada no contêiner $(R_k \times L_k)$ do veículo k, assegurando que a capacidade do veículo não seja excedida e que os produtos sejam posicionados no contêiner de maneira que haja uma rota na região de coleta e outra na região de entrega que atenda todas as requisições designadas à s, respeitando a política LIFO nas R_k pilhas do veículo k.

Para uma dada designação $s \in \mathcal{S}_k$, considera-se c_k^s como sendo a distância mínima percorrida pelo veículo k na rota de coleta e na rota de entrega para que todas as requisições de transporte contidas em s sejam atendidas. Para determinar

o valor de c_k^s é necessário resolver um problema de otimização, que consiste em resolver o DTSPMSHD[1] considerando apenas as requisições de s, e utilizando o contêiner $(R_k \times L_k)$ para armazenar os produtos.

Para formular o DVRPMSHD usando a formulação baseada em particionamento de conjuntos, faz-se necessário descrever as designações de requisições através de p_{ki}^s, sendo que p_{ki}^s recebe o valor 1 se a requisição i está contida no conjunto de requisições s designado ao veículo k e 0 caso contrário. Também é necessário definir a variável binária de decisão λ_k^s que assume 1 se as requisições de transporte contidas em s são atribuídas ao veículo k e 0 caso contrário. Com isso, o DVRPMSHD pode ser formulado por:

$$\min \sum_{k \in K} \sum_{s \in \mathcal{S}_k} c_k^s \lambda_k^s \tag{4.19}$$

$$\sum_{k \in K} \sum_{s \in \mathcal{S}_k} p_{ki}^s \lambda_k^s = 1 \qquad \forall i \in I \tag{4.20}$$

$$\sum_{s \in \mathcal{S}_k} \lambda_k^s = 1 \qquad \forall k \in K \tag{4.21}$$

$$\lambda_k^s \in \{0, 1\} \qquad \forall k \in K, \forall s \in \mathcal{S}_k \tag{4.22}$$

A restrição (4.21) garante que exatamente uma designação de requisições de transporte é escolhida para cada veículo. As designações escolhidas devem atender todas as requisições de transporte (restrição (4.20)) de forma que o custo total (distância) para atendê-las seja o mínimo possível (função objetivo (4.19)).

Embora seja de fácil descrição, esta formulação apresenta uma complexidade implícita, que é a necessidade de gerar (e resolver os DTSPMSHD's) o conjunto de todas as designações de requisições para cada veículo, pois o tamanho desses conjuntos pode ser muito grande.

Uma alternativa para contornar esse problema é utilizar uma técnica de geração de colunas, que permite que esses conjuntos sejam gerados à medida que são necessários (*on-the-fly*). No entanto, essa técnica requer que a restrição de

[1] Uma generalização do DTSPMS, onde os clientes podem ter demanda heterogênea.

integridade (4.22) seja relaxada, forçando um processo de *branch-and-bound* (B&B) para encontrar a solução inteira. Em cada nó da árvore de busca do B&B deve-se usar a técnica de geração de colunas, pois novas colunas podem ser necessárias a fim de encontrar a solução ótima e inteira do problema. O método que utiliza a geração de colunas em cada nó de um B&B é conhecido por *branch-and-price* (B&P) [Desaulniers et al., 2006].

4.3.2 Geração de colunas

Como mencionado anteriormente, a formulação matemática (4.19)-(4.22) pode ser difícil de se resolver, devido ao grande número de variáveis (colunas). Nessa formulação, uma variável (coluna) representa uma designação viável s para um veículo $k \in K$.

Em uma abordagem de geração de colunas, a formulação matemática (4.19)-(4.22) é decomposta em um problema mestre e em $|K|$ subproblemas (um para cada veículo $k \in K$). O problema mestre é uma versão linear da formulação matemática original, contendo apenas um número restrito de colunas. Já os subproblemas são responsáveis por gerar iterativamente colunas promissoras para o problema mestre. Para melhorar iterativamente a solução atual, o problema mestre envia os valores dos preços duais de suas restrições aos subproblemas, que, por sua vez, usa esses valores para gerar novas colunas de custo reduzido negativo. O processo continua até que os subproblemas não sejam capazes de gerar novas colunas que melhorem a solução corrente do problema mestre.

Mais detalhes da teoria por trás do método de geração de colunas podem ser encontrados no livro *Column Generation* de Desaulniers et al. [2006].

O problema mestre e os subproblemas propostos para resolver o DVRPMSHD por meio da técnica de geração de colunas são detalhados a seguir.

4.3.2.1 Problema mestre

O problema mestre é a versão linear da formulação baseada em particionamento de conjuntos definida na Seção 4.3.1, onde a restrição (4.22) é relaxada para a restrição (4.26). Além disso, em vez de usar o conjunto completo $\mathcal{S}_k$ de colunas viáveis para o veículo k, o modelo utiliza apenas um subconjunto $\widehat{\mathcal{S}_k} \subseteq \mathcal{S}_k$ das colunas, que depois

é aumentado iterativamente pelos subproblemas. Abaixo é descrita a formulação matemática para o problema mestre.

DVRPMSHD_MASTER

$$\min \sum_{k \in K} \sum_{s \in \widehat{\mathcal{S}_k}} c_k^s \lambda_k^s \tag{4.23}$$

$$\sum_{k \in K} \sum_{s \in \widehat{\mathcal{S}_k}} p_{ki}^s \lambda_k^s = 1 \qquad \forall i \in I \tag{4.24}$$

$$\sum_{s \in \widehat{\mathcal{S}_k}} \lambda_k^s = 1 \qquad \forall k \in K \tag{4.25}$$

$$0 \leq \lambda_k^s \leq 1 \qquad \forall k \in K, \forall s \in \widehat{\mathcal{S}_k} \tag{4.26}$$

As colunas iniciais do problema mestre devem ser capazes de gerar uma solução viável, pois, caso contrário, não serão associados às restrições do problema os valores duais, os quais são necessários para guiar os subproblemas e dar continuação à geração de colunas. Dentre as diversas formas possíveis de gerar tais colunas iniciais, optou-se por determiná-las por meio de um modelo matemático para um simples problema de empacotamento, em que é necessário atribuir as $|I|$ requisições de transporte para os $|K|$ veículos. A variável binária de decisão δ_i^k é utilizada para indicar a designação de requisições de transporte atribuídas aos veículos, onde δ_i^k assume 1 se a requisição i for designada ao veículo k e 0 caso contrário. Com isso, é possível descrever a seguir formulação matemática:

DVRPMSHD_MASTER
initial columns

$$\min \ 1 \tag{4.27}$$

$$\sum_{k \in K} \delta_i^k = 1 \qquad \forall i \in I \tag{4.28}$$

$$\sum_{i \in I} d_i \delta_i^k \leq R_k L_k \qquad \forall k \in K \tag{4.29}$$

$$\delta_i^k \in \{0, 1\} \qquad \forall k \in K, \forall i \in I \tag{4.30}$$

O objetivo do modelo é apenas determinar uma designação viável s de requisições para cada veículo $k \in K$, sem se preocupar com o custo c_k^s para atendê-las[2], por isso a função objetivo (4.27) é insignificante, sendo definida apenas para que o modelo consiga ser resolvido por algum software de resolução de problemas de programação inteira. A restrição (4.28) garante que cada requisição de transporte seja designada para apenas um único veículo. A restrição (4.29) impede que sejam designadas mais requisições que o veículo possa atender. E, por fim, a restrição (4.30) define o escopo e domínio da variável de decisão δ_i^k, garantindo que as restrições de transporte são designadas (ou não) inteiramente a algum veículo.

Toda vez que o problema mestre é resolvido, além de uma solução linear, o método simplex fornece os preços duais associados às suas restrições. Foi denotado por π_i o preço dual associado à requisição i e por α_k o preço dual associado ao veículo k, respectivamente, obtidos pelas restrições (4.24) e (4.25). Esses valores são passados para os subproblemas, a fim de obter novas colunas promissoras.

4.3.2.2 Subproblemas

O fato de se trabalhar com $|K|$ subproblemas, em vez de apenas um, somente é possível porque a única dependência entre os diferentes veículos é que cada requisição deve ser atendida por um e apenas um veículo, e essa restrição já está coberta no problema mestre (restrição (4.24)).

A partir dos preços duais fornecidos pela solução do problema mestre, cada subproblema $k \in K$ busca uma coluna de custo reduzido negativo. Lembrando que uma coluna nesse contexto é uma designação viável s para o veículo k, incluindo o valor c_k^s da menor distância percorrida pelo veículo para atender todas as requisições de transporte contida em s. Portanto, o custo reduzido é o custo de roteamento, menos os preços duais, que dependem das requisições atendidas e do veículo utilizado.

A formulação de ILP para o subproblema k é semelhante ao modelo (4.1)-(4.18), proposto para o DVRPMSHD, com as seguintes diferenças: a função objetivo (4.1) é substituída por (4.31), a qual expressa o custo reduzido do subproblema k; o índice k é descartado em todas as variáveis, uma vez que cada subproblema trata

[2]O custo associado às colunas inicial é infinito, ou seja, $c_k^s = \infty$.

apenas um veículo; as restrições (4.6), (4.9) e (4.14) são removidas, pois somente são necessárias quando são considerados vários veículos; a restrição (4.10) é substituída por (4.38), porque nem todas as requisições devem ser atendidos por um único veículo; e a restrição (4.42) é adicionada para impedir colunas que representam carregamentos excessivamente vazios sejam geradas pelo subproblema e adicionadas ao problema mestre, pois o contêiner do veículo precisa conter pelo menos o número de produtos que sobram quando os demais contêineres estiverem totalmente carregados.

Por motivos de clareza, a formulação ILP dos subproblemas é totalmente descrita abaixo, sendo detalhadas as variáveis utilizadas e as restrições modificadas/inseridas.

- $x_{i,j}^T$: variável binária que assume 1 se o veículo utiliza a aresta (i, j) na região T e 0 caso contrário.
- y_{ij}^T : variável binária que assume 1 se o vértice i é visitado antes do vértice j na região T e 0 caso contrário.
- w_i : variável binária que assume 1 se o veículo atende a requisição i na região T e 0 caso contrário.
- z_{ih}^r : variável binária que assume 1 se o h-ésimo produto referente à requisição i é armazenado na r-ésima pilha do veículo e 0 caso contrário.

DVRPMSHD_SUB$_k$

$$\min \sum_{T \in \{P,D\}} \sum_{(i,j) \in E^T} c_{ij}^T x_{ij}^T - \sum_{i \in I} \pi_i w_i - \alpha_k \tag{4.31}$$

$$\sum_{j \in V_c^T} x_{0j}^T = 1 \qquad \forall\, T \in \{P, D\} \tag{4.32}$$

$$\sum_{i \in V_c^T} x_{i0}^T = 1 \qquad \forall\, T \in \{P, D\} \tag{4.33}$$

$$\sum_{i \in V^T \setminus \{j\}} x_{ij}^T = w_j \qquad \forall\, T \in \{P, D\}, \forall j \in V_c^T \tag{4.34}$$

$$\sum_{j \in V^T \setminus \{i\}} x_{ij}^T = \sum_{j \in V^T \setminus \{i\}} x_{ji}^T \qquad \forall\, T \in \{P, D\}, \forall i \in V_c^T \tag{4.35}$$

$$\sum_{i \in I} \sum_{h=1}^{d_i} z_{ih}^r \leq L_k \qquad \forall r = 1..R_k \tag{4.36}$$

$$\sum_{h=1}^{d_i} \sum_{r=1}^{R_k} z_{ih}^r = d_i w_i \qquad \forall i \in I \tag{4.37}$$

$$y_{ij}^T + y_{ji}^T = w_i \qquad \begin{array}{r} \forall\, T \in \{P, D\}, \forall i \in V_c^T, \\ \forall j \in V_c^T \setminus \{i\} \end{array} \tag{4.38}$$

$$y_{il}^T + y_{lj}^T \leq y_{ij}^T + 1 \qquad \begin{array}{r} \forall\, T \in \{P, D\}, \forall l \in V_c^T, \\ \forall i \in V_c^T, \forall j \in V_c^T \setminus \{i\} \end{array} \tag{4.39}$$

$$x_{ij}^T \leq y_{ij}^T \qquad \begin{array}{r} \forall\, T \in \{P, D\}, \forall i \in V^T, \\ \forall j \in V^T \setminus \{i\} \end{array} \tag{4.40}$$

$$y_{ij}^P + z_{ih}^r + z_{jh'}^r \leq 3 - y_{ij}^D \qquad \begin{array}{r} \forall i \in I, \forall j \in I \setminus \{i\}, \forall h = 1..d_i, \\ \forall h' = 1..d_j, \forall r = 1..R_k \end{array} \tag{4.41}$$

$$\sum_{i \in I} d_i w_i \geq R_k L_k - \left(\sum_{k' \in K} R_{k'} L_{k'} - \sum_{i \in I} d_i \right) \tag{4.42}$$

$$x_{ij}^T \in \{0, 1\} \qquad \forall\, T \in \{P, D\}, \forall (i, j) \in E^T \tag{4.43}$$

$$y_{ij}^T \in \{0, 1\} \qquad \begin{array}{r} \forall\, T \in \{P, D\}, \\ \forall i \in V^T, \forall j \in V^T \setminus \{i\} \end{array} \tag{4.44}$$

$$z_{ih}^r \in \{0, 1\} \qquad \forall i \in I, \forall h = 1..d_i, \forall r = 1..R_k \tag{4.45}$$

A Figura 4.3 ilustra o mecanismo de funcionamento da abordagem de geração de colunas, destacando a interação entre o problema mestre e os subproblemas.

4.3.3 Branching

Se o processo de geração de colunas é executado até que não haja nenhuma coluna com custo reduzido negativo, a solução final do problema mestre é ótima. Porém, essa solução pode não ser inteira (variáveis de decisão assumem valores fracionários), pois o modelo resolvido é uma relaxação linear da formulação baseada em particionamento de conjuntos descrita na Seção 4.3.1. Caso isso ocorra, é necessário

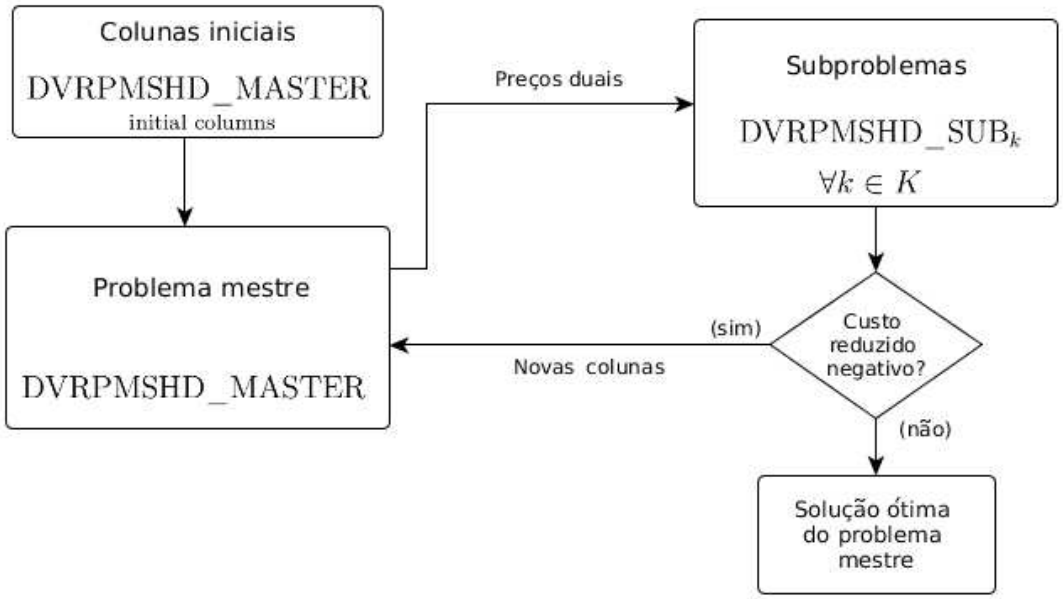


Figura 4.3: Interação entre o problema mestre e os subproblemas no algoritmo de geração de colunas.

utilizar um método de *branch-and-bound* (B&B) para encontrar a solução ótima que apresente valores inteiros nas suas variáveis de decisão. Diferentemente de um B&B padrão, nesse caso talvez seja necessário incluir novas colunas, isto é, novas variáveis, em cada nó da árvore de busca do B&B. Todo esse processo é então chamado de *branch-and-price* (B&P) [Desaulniers et al., 2006].

Em cada nó da árvore B&B, se for encontrada uma solução não inteira, uma requisição de transporte é escolhida para ser obrigatoriamente atribuída (e depois obrigatoriamente não atribuída) em um determinado veículo. Da mesma forma como foi demonstrado por Savelsbergh [1997] para o Problema de Designação Generalizada (*Generalized Assignment Problem* - GAP) [Cattrysse & Van Wassenhove, 1992], se λ_k^s é fracionário para algum $s \in \mathcal{S}_k$, então alguma requisição é apenas parcialmente atendida pelo veículo k, ou seja, existe uma requisição i contida em s, tal que $\sum_{s' \in \widehat{\mathcal{S}}_k} p_{ki}^{s'} \lambda_k^{s'} < 1$.

Pode acontecer que mais de uma requisição seja parcialmente atendida por algum veículo, e é preciso escolher apenas uma para realizar o processo de *branching*. Então, entre todas essas requisições, é escolhida a requisição i que é

servida pela menor fração (valor mais próximo de zero) pelo veículo k.

Em um lado da árvore de busca, o veículo k deve ser proibido de atender a requisição i. Para isso é necessário fixar $w_i = 0$ no subproblema do veículo k, além de proibir que qualquer designação atribuída ao veículo k que inclua a requisição i seja escolhida no problema mestre. Formalmente:

- fixar $w_i = 0$ no subproblema k
- fixar $\lambda_k^s = 0$ no problema mestre $\forall s \in \widehat{\mathcal{S}_k} \mid p_{ki}^s = 1$

No outro lado da árvore de busca, o veículo k deve ser forçado a atender a requisição i. Para isso é preciso fixar $w_i = 1$ no subproblema do veículo k e $w_i = 0$ nos demais subproblemas, além de proibir a escolha de qualquer designação atribuída ao veículo k que não inclua a requisição i e também proibir que os demais veículos escolham alguma designação que contenha essa requisição. Formalmente:

- fixar $w_i = 1$ no subproblema k
- fixar $w_i = 0$ em todo subproblema $k' \in K \mid k' \neq k$
- fixar $\lambda_k^s = 0$ no problema mestre $\forall s \in \widehat{\mathcal{S}_k} \mid p_{ki}^s = 0$
- fixar $\lambda_k^{s'} = 0$ no problema mestre $\forall k' \in K,\ \forall s' \in \widehat{\mathcal{S}_{k'}} \mid k' \neq k,\ p_{k'i}^{s'} = 1$

As soluções inteiras encontradas durante o processo atualizam a melhor solução, a qual é usada como o *upper-bound* (UB) da solução final.

4.3.4 Branch-and-price

O Algoritmo 5 descreve os passos realizados e descritos anteriormente pelo método exato B&P[3] o proposto para o DVRPMSHD.

[3]O algoritmo foi estruturado baseando-se no pseudo-código de um algoritmo *branch-and-price para o Problema de Designação Generalizada (Generalized Assignment Problem - GAP) [Cattrysse & Van Wassenhove, 1992]* apresentado pelo professor André Gustavo dos Santos na disciplina INF 791 - Tópicos Especiais em Otimização, lecionada no segundo semestre do ano de 2015 para alunos de pós-graduação do Departamento de Informática (DPI) da Universidade Federal de Viçosa (UFV).

Note que, na estratégia de geração de colunas proposta, a maior dificuldade foi deslocada para os subproblemas, pois estes correspondem a problemas de alta complexidade computacional ($\mathcal{NP}$-Difícil). Entretanto, as pequenas dimensões dos contêiners permitem que estes subproblemas sejam resolvidos, na maioria das vezes, em um tempo aceitável.

Algoritmo 5: Branch-and-price (B&P)

```
// Solução do nó atual
s ← solução da relaxação linear do modelo (4.19)- (4.22) por geração de colunas
// Bound
if f*(s) ≥ UB ou s for inviável then
    return ∅
end
// Solução fracionária
if s é uma solução fracionária then
    // Branching
    Escolher uma requisição i atendida parcialmente por um veículo k
    // Lado A
    Proibir que i seja atendida por k
    s_a ← B&P
    Retirar a proibição acima
    // Lado B
    Obrigar que i seja atendida por k
    s_b ← B&P
    Retirar a obrigação acima
    s ← melhor solução entre s_a e s_b // (f*(s_a) < f*(s_b) ? s_a : s_b)
end
// Solução inteira
if f*(s) ≤ UB then
    melhor_solucao ← s
    UB ← f*(s)
end
```

O tempo de processamento do algoritmo B&P foi limitado a 1 hora. Caso o algoritmo não seja capaz de determinar a solução ótima dentro deste tempo, o subconjunto de colunas já geradas não necessariamente conterá as colunas da solução

ótima. Porém, com o objetivo de determinar a melhor solução utilizando as colunas já geradas, o problema mestre é executado novamente, impondo a integralidade em todas as variáveis.

4.4 Algoritmo heurístico proposto

O DVRPMSHD apresenta uma alta complexidade e resolvê-lo por métodos exatos requer excessivo tempo computacional. Portanto, nesta seção é descrita uma abordagem heurística a fim de obter boas soluções para o problema gastando pouco tempo. A seguir a heurística proposta e seus componentes são detalhados.

4.4.1 Representação da solução

A representação das soluções do DVRPMSHD foi proposta com o propósito de facilitar a exploração do espaço de busca. A ideia inicial era utilizar a mesma representação utilizada para as soluções do DVRPMS, que informa o plano de carregamento de cada veículo. Porém, o fato de mais de um produto estar associado a uma mesma requisição dificulta e acarreta maior complexidade para percorrer o espaço de soluções viáveis do problema, uma vez que estruturas de vizinhança devem ser projetadas de forma a garantir que o plano de carregamento de um determinado contêiner seja viável, ou seja, garantir que exista uma rota na região de coleta e outra na região de entrega que respeite a política LIFO do plano de carregamento. A Figura 4.4 ilustra um plano de carregamento inviável para um contêiner (2×4). Note que por essa configuração não é possível construir nenhuma rota na região de coleta e nem na região de entrega, pois os produtos das requisições 1 e 8 geram dependências entre si (*deadlock*).

Para contornar essa dificuldade, a ideia foi representar apenas a designação de requisições para cada veículo, deixando que o plano de carregamento que atenda as requisições seja construído a partir de uma heurística de carregamento (detalhada na Seção 4.4.2). Então, uma solução do DVRPMSHD é representada por uma matriz de $|K|$ linhas, sendo que cada linha informa a designação de requisições de transporte associada a um veículo (k-ésima linha refere-se ao k-ésimo veículo), e também a

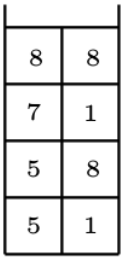


Figura 4.4: Plano de carregamento inviável para um contêiner (2×4).

ordem (esquerda para a direita) que essas requisições são aplicadas à heurística de carregamento.

A Figura 4.5 apresenta um exemplo de representação de uma solução que envolve 9 requisições de transporte que são designadas, conforme mostra a figura, para três veículos. Ao primeiro veículo (primeira linha da matriz) são associadas as requisições 2, 4 e 3, ao segundo veículo (segunda linha da matriz) as requisições 5, 1, 7 e 9 e ao terceiro veículo (terceira linha da matriz) as requisições 6 e 9. A demanda referente à cada requisição é informada pelo número no canto superior direito de cada célula da matriz.

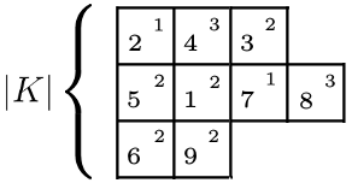


Figura 4.5: Exemplo de representação de uma solução para o DVRPMSHD.

4.4.2 Heurística de carregamento

A heurística de carregamento foi projetada a fim de organizar os produtos referentes às requisições de transporte associadas a cada veículo dentro do compartimento de carga do mesmo. Para isso, as requisições são tratadas da esquerda para a direita, considerando a ordem que elas aparecem na representação da solução.

Para cada requisição de transporte i designada ao veículo k, a heurística de carregamento busca alocar os d_i produtos na pilha r menos ocupada do veículo k,

pois dessa forma o plano de carregamento é construído iterativamente na tentativa de diminuir o número de restrições LIFO. Caso não seja possível alocar todos os produtos de uma requisição em uma mesma pilha, é então alocado o máximo de produtos possíveis (ocupando totalmente a r-ésima pilha do veículo k) e os demais são alocados em outras pilhas, seguindo a mesma estratégia de tentar alocá-los juntos em uma mesma pilha.

A Figura 4.6 mostra um exemplo de funcionamento da heurística de carregamento a partir da representação da solução detalhada na Figura 4.5. Tomando como exemplo o veículo 2 (segunda linha da matriz e segundo contêiner), primeiramente os produtos da requisição 5 são armazenados na primeira pilha, pois ambas as pilhas estão vazias[4], em seguida os produtos da requisição 1 são armazenados na segunda pilha (menos ocupada) do veículo, depois o único produto da requisição 7 é armazenado na primeira pilha (lexicograficamente menor) e, por fim, dois produtos da requisição 8 ocupam o restante da pilha menos ocupada e o terceiro produto é então armazenado em outra pilha (primeira pilha).

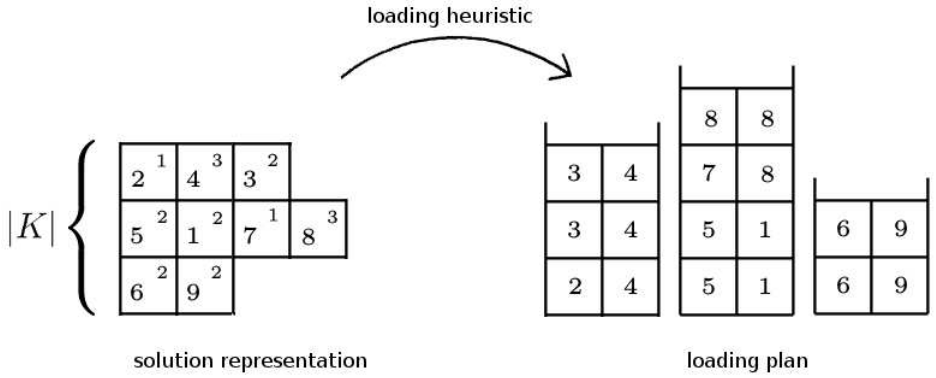


Figura 4.6: Exemplo de aplicação da heurística de carregamento aplicada à uma solução do DVRPMSHD.

Observe que dessa forma os planos de carregamento são construídos de forma a sempre existir, pelo menos, uma rota de coleta e uma rota de entrega possível a partir deles. Porém, é importante ressaltar que a heurística de carregamento pode não ser capaz de construir o plano de carregamento ótimo para um determinado

[4]Quando duas ou mais pilhas estão ocupadas com o mesmo número de produtos e apresentam a melhor ocupação, o desempate é realizado considerando a ordem lexicográfica.

conjunto de requisições s que deve ser atendidas por um veículo, isto é, mesmo que todas as $|s|!$ permutações do conjunto de requisições s sejam aplicadas à heurística de carregamento, é possível que as rotas mais curtas para atender as requisições de s não sejam viáveis a partir de nenhum dos $|s|!$ planos de carregamento construídos. A Figura 4.7 ilustra essa inconveniência, em que a rota de coleta e a rota de entrega, ilustradas na figura, não podem ser realizadas a partir de nenhum contêiner criado pela heurística de carregamento, considerando todas as permutações[5] das requisições $s = \{1, 2, 5, 7\}$.

Embora nenhum dos contêineres construídos pela heurística de carregamento possa atender as rotas ilustradas na Figura 4.7, o plano de carregamento mostrado na Figura 4.8 é compatível com essas rotas.

4.4.3 Avaliação da solução

A representação da solução, juntamente com a heurística de carregamento proposta, define apenas a designação de requisições e o plano de carregamento de cada veículo, sendo que a tarefa de determinar a rota na região de coleta e na região de entrega é feita pela função de avaliação. Para avaliação de uma solução do DVRPMSHD, foram utilizadas as funções de avaliação propostas para avaliar as soluções do DVRPMS, com apenas algumas diferenças oriundas do fato que os clientes podem ter demandas múltiplas. Tais funções e diferenças são detalhados a seguir.

4.4.3.1 Formulação de programação linear inteira mista - MILP

A formulação matemática utilizada para encontrar a rota de coleta (entrega) de menor distância percorrida pelo veículo k, dado o seu plano de carregamento, é a mesma formulação proposta para avaliar uma solução do DVRPMS (Seção 3.4.2.1), exceto pelas restrições (3.23) e (3.30) que devem ser substituídas, respectivamente, pelas restrições (4.46) e (4.47), pois não há restrições LIFO sobre os

[5] Os planos de carregamento aparecem na figura em ordem lexicográfica das permutações de s, isto é, o plano de carregamento (a) foi construído pela sequência [1, 2, 5, 7], (b) pela sequência [1, 2, 7, 5], ..., (x) pela sequência [7, 5, 2, 1].

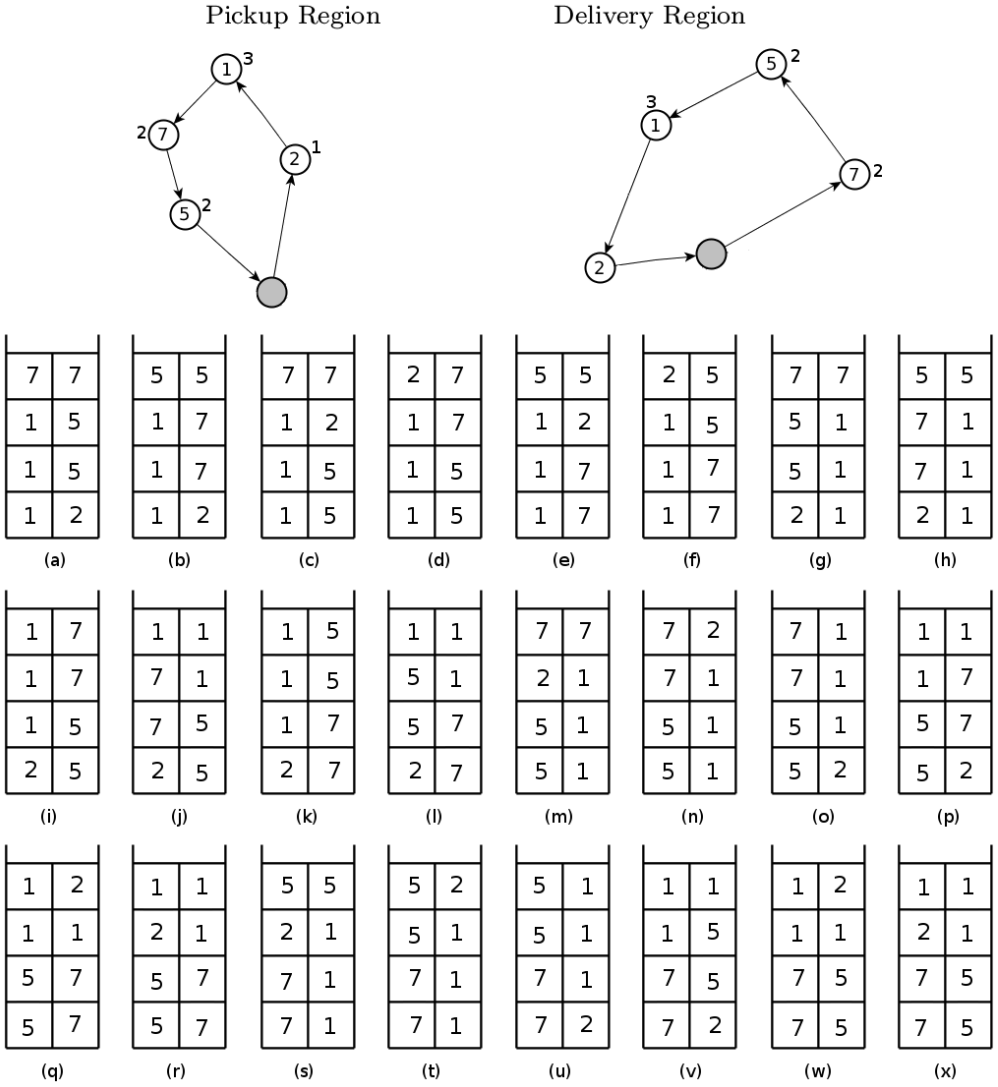


Figura 4.7: Exemplo de falha da heurística de carregamento ao criar o plano de carregamento ótimo para um contêiner (2×4).

produtos de um mesmo cliente, uma vez que todos serão coletados/entregues juntos.

$$u_{p_k^{rl}} \leq u_{p_k^{rl+1}} \qquad \forall\, p_k^{rl} \in C_k \mid p_k^{rl+1} \in C_k,\ p_k^{rl} \neq p_k^{rl+1} \tag{4.46}$$

$$u_{p_k^{r,l}} \leq u_{p_k^{r,l-1}} \qquad \forall\, p_k^{rl} \in C_k \mid p_k^{rl-1} \in C_k,\ p_k^{rl} \neq p_k^{rl-1} \tag{4.47}$$

Note que as restrições (3.23) e (3.30) também podem ser usadas sem a necessidade de serem substituídas pelas respectivas restrições (4.46) e (4.47), uma vez

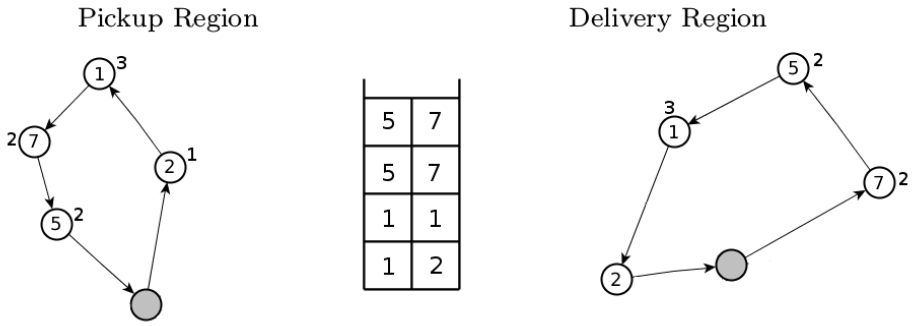


Figura 4.8: Plano de carregamento viável de um contêiner (2×4) com relação às rotas.

que a variável u tem o escopo relacionado aos clientes e não às suas demandas. Assim, essa variável assumirá apenas um valor para cada cliente e isso é tolerável pelas restrições (3.23) e (3.30), mesmo quando se referem a dois produtos de um mesmo cliente (restrições $\leq$). Contudo, as restrições (4.46) e (4.47) são utilizadas para maior clareza, fidelidade à política LIFO e também para diminuir o número de restrições inseridas nos modelos.

4.4.3.2 Branch-and-bound - B&B

A segunda estratégia exata proposta para avaliar as soluções do DVRPMSHD é realizada através do algoritmo *branch-and-bound* (B&B) descrito na Seção 3.4.2.2. A única diferença é que, ao explorar um nó da árvore de busca do algoritmo, todos os produtos de uma mesma requisição são explorados por esse nó, tornando assim o processo mais eficiente, uma vez que dessa forma todo o contêiner é avaliado mais rapidamente.

4.4.3.3 Heurística gulosa - GH

Como já foi apontado anteriormente, determinar as rotas ótimas a partir de um plano de carregamento é um problema $\mathcal{NP}$-Difícil. Visto isso, para avaliar uma solução do DVRPMSHD, após a criação do plano de carregamento de cada veículo pela heurística de carregamento, foi utilizada a fase gulosa da estratégia heurística proposta para o DRVPMS, descrita na Seção 3.4.2.3. A única diferença é que,

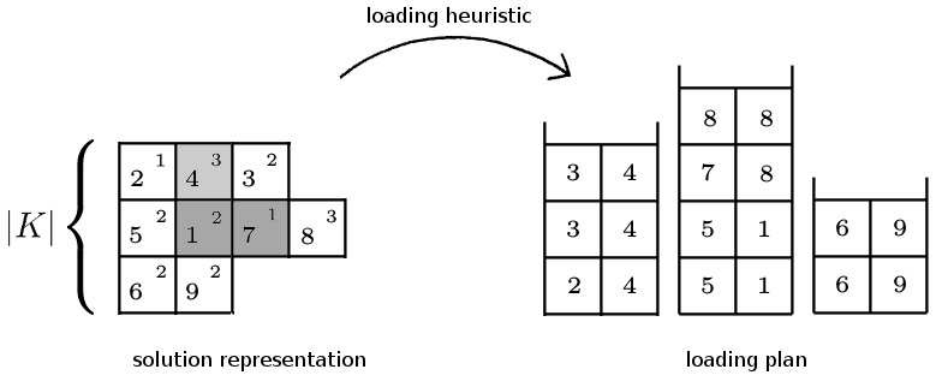
semelhantemente à avaliação realizada pelo algoritmo B&B, todos os produtos de uma mesma requisição são coletados (entregues) em uma mesma iteração do algoritmo guloso, fazendo assim que o processo fique mais eficiente.

4.4.4 Estrutura de vizinhança

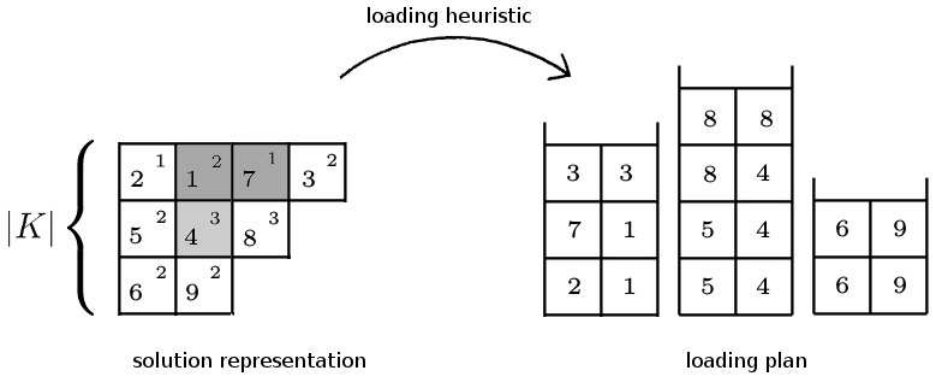
A estrutura de vizinhança definida para explorar o espaço de soluções do DVRPMSHD consiste na troca de posição de duas ou mais requisições de transportes das suas respectivas designações. Uma solução vizinha s' pode ser obtida a partir de uma solução s através dos seguintes passos:

1. Escolher um veículo $k \in K$.
2. Escolher uma requisição i designada ao veículo k.
3. Escolher um veículo $k' \in K$.
4. Escolher um subconjunto de requisições q (possivelmente vazio) designadas ao veículo k', tal que após a troca de posição da requisição i pelas requisições do subconjunto q, a capacidade dos veículos não sejam extrapolada.

A Figura 4.9 (a) mostra um exemplo de uma solução s e a Figura 4.9 (b) uma de suas soluções vizinhas s', a qual foi obtida após a troca de posição entre a requisição de número 4 (veículo 1) e as requisições 1 e 7 (veículo 2). Note que a troca realizada não extrapolou a capacidade dos veículos e que a heurística de carregamento produziu novos planos de carregamento para os contêineres do veículo 1 e 2.



(a) Solução ***s***



(b) Solução ***s'***

Figura 4.9: Exemplo de um vizinho de uma solução do DVRPMSHD utilizando a estrutura de vizinhança definida.

4.4.5 Solução inicial

A solução inicial da abordagem heurística proposta é criada aleatoriamente, sendo designado um subconjunto aleatório de requisições de transporte para cada veículo, de modo que a demanda não exceda a capacidade de cada veículo e que cada requisição de transporte seja atendida por apenas um único veículo.

4.4.6 Busca local

Como foi dito anteriormente na Seção 3.4.5, um procedimento de busca local pode ser considerado relativamente simples, uma vez que apenas um caminho no espaço de soluções é explorado. Porém, os procedimentos de busca locais apresentam

grande importância, sendo utilizados como subrotinas de diversas metaheurísticas.

Neste trabalho foi desenvolvida uma busca local que é aplicada na melhor solução encontrada pelo algoritmo heurístico SA (descrito a seguir na Seção 4.4.7). Nessa busca local, a exploração dos vizinhos é feita de forma aleatória, utilizando os passos descritos na Seção 4.4.4. A vizinhança de uma solução atual é explorada até que uma solução melhor seja encontrada ou até que 100 iterações (soluções) não apresentem melhora (critério de parada).

Todas as soluções exploradas pelo procedimento de busca local definido para o algoritmo SA são avaliadas utilizando a função de avaliação exata B&B descrita na Seção 4.4.3.2.

4.4.7 Simulated annealing

A abordagem heurística proposta para o DVRPMSHD é baseada na metaheurística *Simulated Annealing* (SA). O Algoritmo 6 descreve os passos realizados pela heurística. A solução inicial do algoritmo é gerada aleatoriamente, escolhendo casualmente uma designação de requisições de transporte para cada veículo. Enquanto os critérios de parada não são satisfeitos, o algoritmo seleciona aleatoriamente uma solução vizinha a partir da solução atual usando a estrutura de vizinhança definida (Seção 4.4.4). O critério para aceitar uma solução como a solução atual é feito de acordo com uma probabilidade determinada em função da qualidade da solução atual e da solução vizinha e também em função da temperatura atual. No final do algoritmo, a melhor solução encontrada é retornada.

Note que todas as soluções exploradas na fase interna do Algoritmo 6 (linhas 8 e 11) são avaliadas pela função $f_{MIX}()$, que obtém a distância total percorrida por todos os veículos, sendo que as rotas para os contêineres de dimensões (2×2), (2×3), (2×4), (2×5) e (3×4) são obtidas utilizando a avaliação exata B&B (descrita na Seção 4.4.3.2) e para os demais contêineres ((3×5), (3×6) e (3×8)) as rotas são obtidas utilizando a avaliação heurística GH descrita na Seção 4.4.3.3. Essas decisões foram tomadas após constatar que avaliar os contêineres maiores de forma exata, em todas as soluções exploradas pelo algoritmo, requer excessivo tempo computacional[6].

[6]De acordo com resultados reportados à frente, na Seção 4.5.2.

Algoritmo 6: Simulated Annealing (SA)

1 $s \leftarrow$ gerar uma solução de forma aleatória
2 $melhor\_solucao \leftarrow s$
3 $T_{atual} \leftarrow T_i$
4 **while** $T_{atual} \geq T_f$ **do**
5 $it \leftarrow 0$
6 **while** $it < N$ **do**
7 $s' \leftarrow$ gerar aleatoriamente uma solução vizinha $s' \in N(s)$
8 $\Delta \leftarrow f_{MIX}(s') - f_{MIX}(s)$
9 **if** $\Delta < 0$ *or random(0, 1)* $< e^{-\Delta/T_{atual}}$ **then**
10 $s \leftarrow s'$
11 **if** $f_{MIX}(s') < f_{MIX}(melhor\_solucao)$ **then**
12 $melhor\_solucao \leftarrow s'$
13 **end**
14 **end**
15 $it \leftarrow it + 1$
16 **end**
17 $T_{atual} \leftarrow T_{atual} \times (1 - \alpha)$
18 **end**
19 $melhor\_solucao \leftarrow$ aplicar a busca local na solução $melhor\_solucao$ // $f_{B\&B}$
20 **return** $melhor\_solucao$

No final do algoritmo (linha 19), uma busca local é aplicada na melhor solução encontrada, sendo que todos contêineres (independente das dimensões) das soluções exploradas por essa busca local são avaliadas utilizando a função de avaliação exata B&B descrita na Seção 4.4.3.2. O objetivo dessa busca local é buscar atingir um mínimo local que talvez não tenha sido encontrado anteriormente devido a possibilidade de não obtenção das melhores rotas de cada contêiner.

4.5 Resultados computacionais

Nesta seção são apresentadas as características das instâncias, uma comparação entre as diferentes funções de avaliação propostas, a calibração do algoritmo heurístico e os resultados alcançados para o DVRPMSHD.

A formulação ILP e o algoritmo *branch-and-price* foram implementados via

ILOG Concert Technology C++ e foram executados utilizando o software ILOG CPLEX versão acadêmica 12.5, com todas as configurações padrões, exceto para o tempo de execução que foi limitado a 1 hora.

O algoritmo heurístico SA foi implementado utilizando a linguagem de programação C/C++, o código foi compilado com GNU g++ versão 4.8.4 e executado de forma sequencial (sem paralelismo). A fase de calibração do algoritmo foi processada no cluster da Universidade Federal de Viçosa (UFV).

Os resultados finais descritos para o DVRPMSHD foram processados em um computador Intel Core i5-3570 CPU @ 3.40GHz x 4, com 16GB de memória RAM e sistema operacional Ubuntu 14.04 LTS, 64 bits.

4.5.1 Descrição das instâncias de teste

As instâncias de teste utilizadas para avaliar a eficiência dos algoritmos propostos para o DVRPMSHD foram criadas baseadas nas características das instâncias definidas por Iori & Riera-Ledesma [2015] para o DVRPMS.

Tabela 4.1: Características das instâncias do grupo C definidas para o DVRPMSHD.

Tipo	$\|K\|$	$(R \times L)'s$	$\sum_{k \in K} R_k L_k$	$\|I\|$	$\sum_{i \in I} d_i$
(a)	2	(2×3) (2×3)	12	6	12
(b)	2	(2×2) (2×4)	12	6	12
(c)	3	(2×2) (2×2) (2×2)	12	6	12
(d)	2	(2×3) (3×4)	18	9	18
(e)	3	(2×3) (2×3) (2×3)	18	9	18
(f)	4	(2×2) (2×2) (2×2) (2×3)	18	9	18
(g)	2	(3×4) (3×4)	24	12	24
(h)	3	(2×4) (2×4) (2×4)	24	12	24
(i)	4	(2×3) (2×3) (2×3) (2×3)	24	12	24
(j)	2	(3×5) (3×5)	30	15	30
(k)	3	(2×5) (2×5) (2×5)	30	15	30
(l)	4	(2×3) (2×4) (2×4) (2×4)	30	15	30
(m)	2	(3×6) (3×6)	36	18	36
(n)	3	(3×4) (3×4) (3×4)	36	18	36
(o)	4	(2×4) (2×4) (2×4) (3×4)	36	18	36
(p)	2	(3×8) (3×8)	48	24	48
(q)	3	(3×5) (3×5) (3×6)	48	24	48
(r)	4	(3×4) (3×4) (3×4) (3×4)	48	24	48

Foram criados 33 tipos de instâncias, sendo que cada tipo é definido pelo número de requisições de transporte envolvidas, pelas demandas associadas a cada requisição, pelo número de veículos disponíveis e pelas configurações dos compartimentos de carga dos veículos. A Tabela 4.1 descreve os primeiros 18 tipos de instâncias (denotado como grupo C de instâncias), para as quais a capacidade total dos veículos (coluna $\sum_{k \in K} R_k L_k$) é exatamente igual à demanda total dos clientes (coluna $\sum_{i \in I} d_i$), ou seja, para essas instâncias os contêineres devem estar completamente carregados após atender todos clientes da região de coleta. Já a Tabela 4.2 informa as características dos outros 15 tipos de instâncias (denotado como grupo $\neg C$ de instâncias), em que a capacidade dos veículos é maior que a demanda total dos clientes, assim os contêineres não necessitam ser completamente preenchidos para que os clientes sejam atendidos.

Tabela 4.2: Características das instâncias do grupo $\neg C$ definidas para o DVRPMSHD.

Tipo	$\|K\|$	$(R \times L)'s$	$\sum_{k \in K} R_k L_k$	$\|I\|$	$\sum_{i \in I} d_i$
(s)	2	(2×3) (3×4)	18	6	12
(t)	3	(2×3) (2×3) (2×3)	18	6	12
(u)	4	(2×2) (2×2) (2×2) (2×3)	18	6	12
(v)	2	(3×4) (3×4)	24	9	18
(w)	3	(2×4) (2×4) (2×4)	24	9	18
(x)	4	(2×3) (2×3) (2×3) (2×3)	24	9	18
(y)	2	(3×5) (3×5)	30	12	24
(z)	3	(2×5) (2×5) (2×5)	30	12	24
(α)	4	(2×3) (2×4) (2×4) (2×4)	30	12	24
(β)	2	(3×6) (3×6)	36	15	30
(γ)	3	(3×4) (3×4) (3×4)	36	15	30
(δ)	4	(2×4) (2×4) (2×4) (3×4)	36	15	30
(ζ)	2	(3×8) (3×8)	48	18	36
(η)	3	(3×5) (3×5) (3×6)	48	18	36
(θ)	4	(3×4) (3×4) (3×4) (3×4)	48	18	36

Foi utilizada uma distribuição uniforme de valores inteiros entre 1 e 3 para determinar os valores associados às demandas das requisições de transporte. A Tabela 4.3 mostra os valores gerados considerando 6, 9, 12, 15, 18 e 24 requisições.

Tabela 4.3: Distribuição uniforme dos valores de demanda para os clientes.

$\|I\|$	Demanda dos clientes																								$\sum_{i \in I} d_i$
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
6	3	1	3	1	2	2																			12
9	3	1	3	2	2	2	1	1	3																18
12	3	1	3	2	3	2	1	1	3	2	1	2													24
15	2	1	3	2	3	2	1	1	3	2	1	2	1	3	3										30
18	2	1	3	2	3	2	3	1	3	2	1	1	1	3	3	2	1	2							36
24	2	1	3	3	1	2	3	1	3	2	1	2	1	3	3	2	1	2	2	2	1	3	1	3	48

Com relação às localizações dos clientes, foram também utilizadas as informações de 5 instâncias (R05, R06, R07, R08 e R09) descritas por Petersen & Madsen [2009] para o problema DTSPMS. Lembrando que cada uma dessas instâncias informa as localizações dos clientes na região de coleta e na região de entrega, e que a distância entre dois pontos quaisquer de uma mesma região foi definida como sendo a distância euclideana arredondada entre eles.

4.5.2 Funções de avaliação

Para analisar o desempenho das diferentes funções de avaliação, quando aplicadas às soluções do DVRPMSHD, foi realizado o mesmo experimento usado para avaliar as funções de avaliação do DVRPMS, descrito na Seção 3.5.2.

Os gráficos das Figuras 4.10 a 4.17 mostram uma comparação entre as funções de avaliação exatas MILP e B&B. Cada uma das figuras mostra o número de contêineres avaliados no decorrer de 60 segundos pela formulação MILP e pelo algoritmo B&B, considerando os diferentes contêineres descritos nas Tabelas 4.1 e 4.2.

Note que ambas as funções de avaliação se mostram mais eficientes para avaliar as soluções do DVRPMSHD do que para avaliar as soluções do DVRPMS, comparando os resultados obtidos na avaliação de contêineres com mesmas dimensões (Figuras 3.18 a 3.22, Seção 3.5.2). A explicação para isso é facilmente observada, pois na função de avaliação MILP o número de variáveis e restrições para avaliar uma solução do DVRPMSHD é menor devido ao fato dos clientes terem demandas múltiplas e consequentemente ter menos requisições de transporte

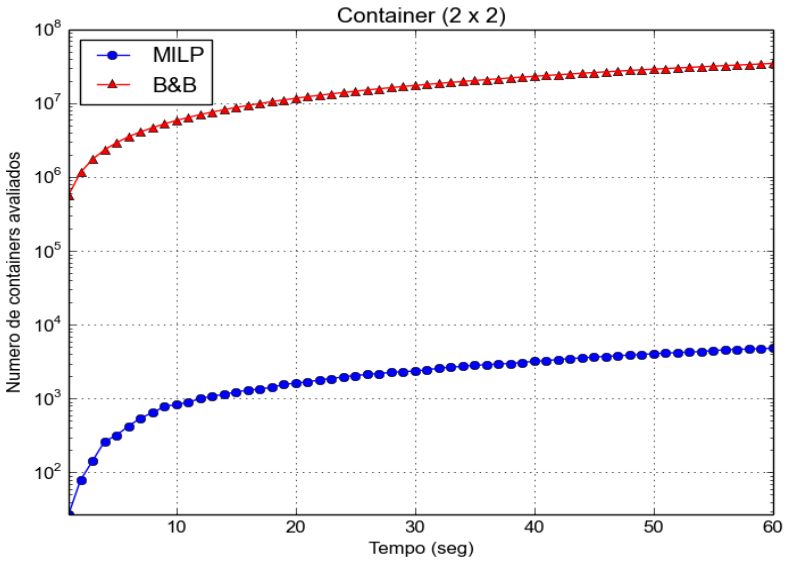


Figura 4.10: Número de contêineres (2×2) avaliados pelas funções de avaliação MILP e B&B

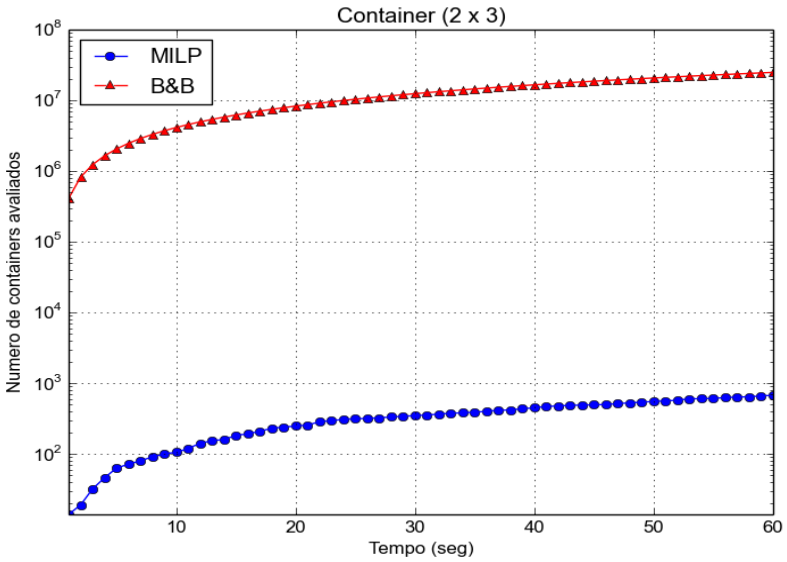


Figura 4.11: Número de contêineres (2×3) avaliados pelas funções de avaliação MILP e B&B

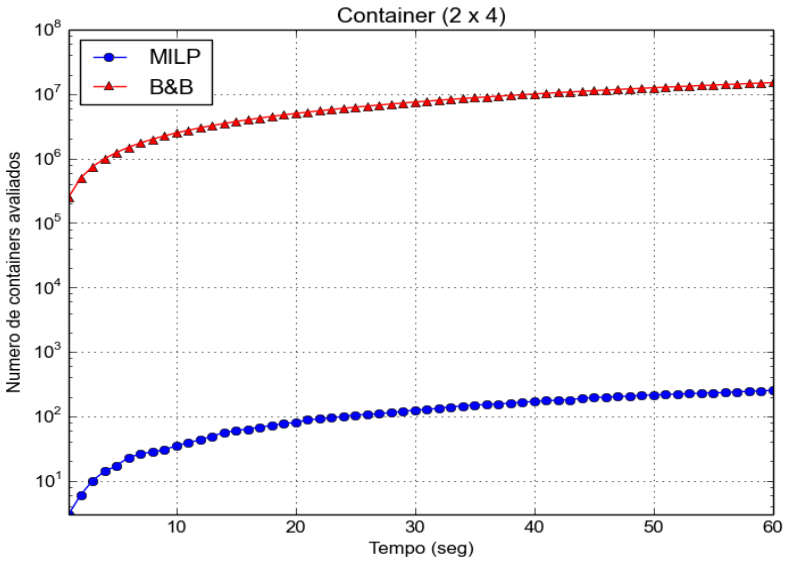


Figura 4.12: Número de contêineres (2×4) avaliados pelas funções de avaliação MILP e B&B

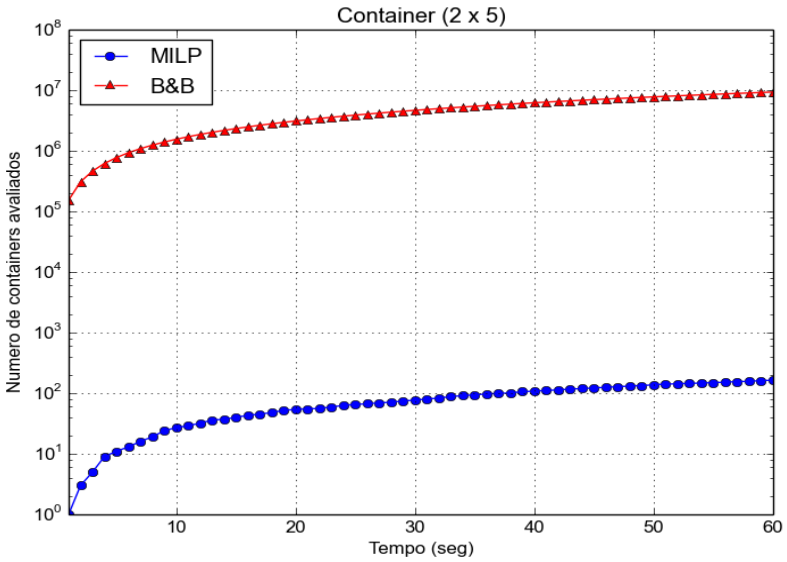


Figura 4.13: Número de contêineres (2×5) avaliados pelas funções de avaliação MILP e B&B

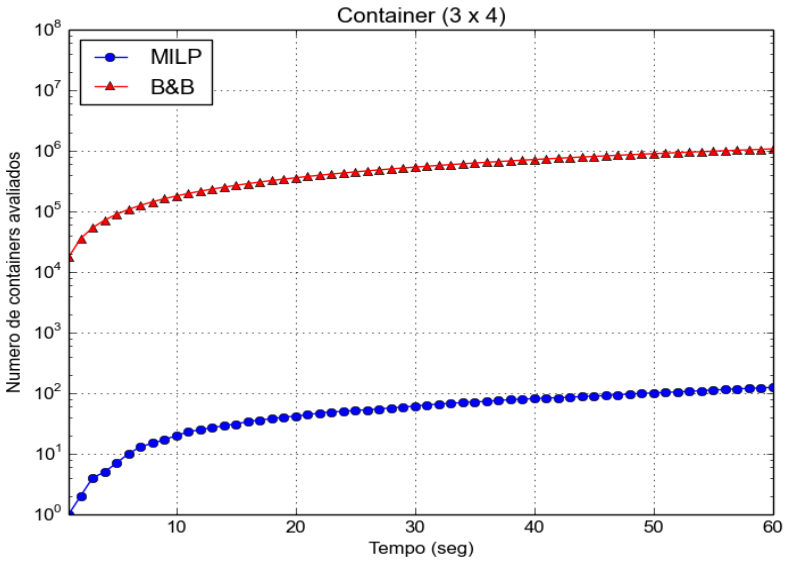


Figura 4.14: Número de contêineres (3×4) avaliados pelas funções de avaliação MILP e B&B

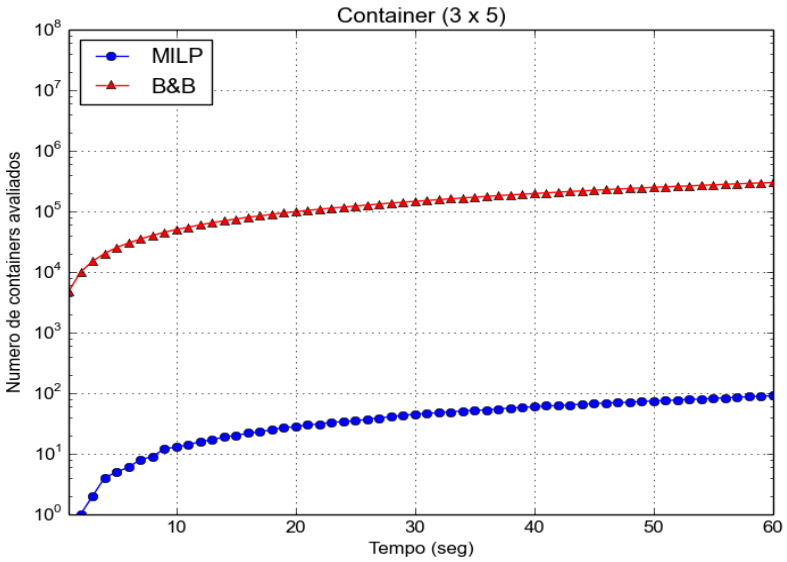


Figura 4.15: Número de contêineres (3×5) avaliados pelas funções de avaliação MILP e B&B

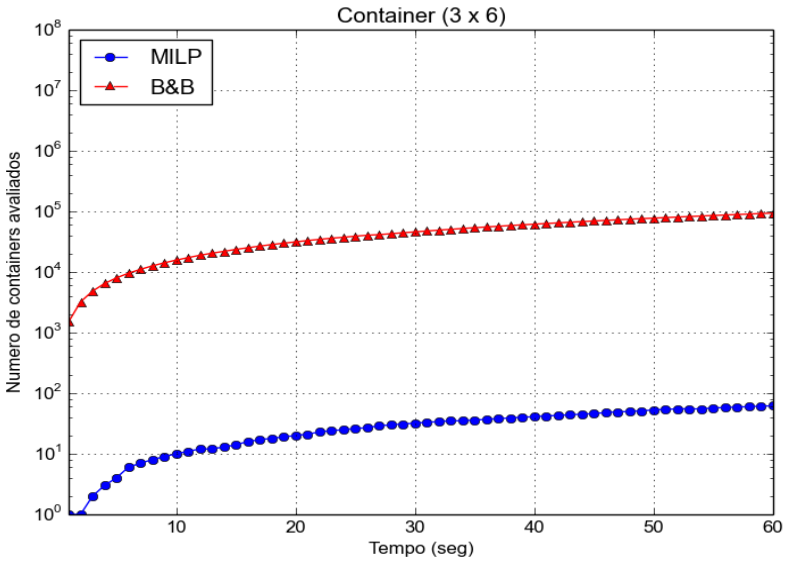


Figura 4.16: Número de contêineres (3×6) avaliados pelas funções de avaliação MILP e B&B

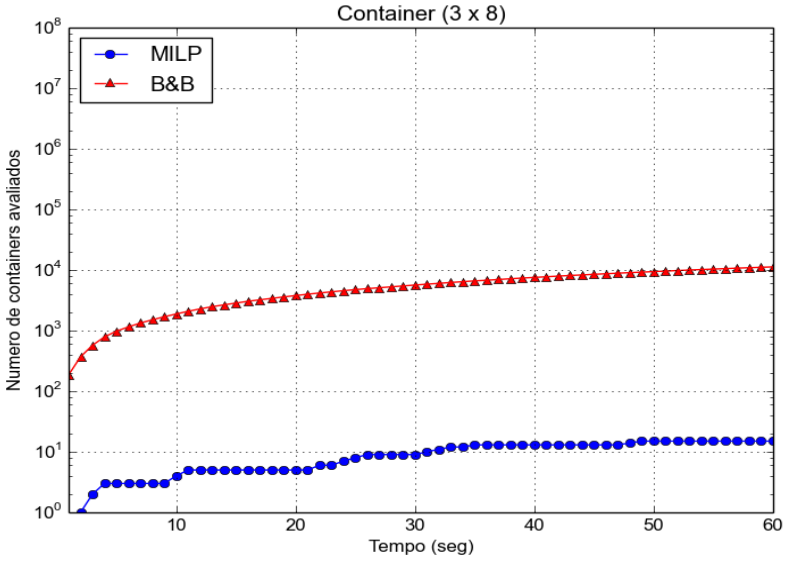


Figura 4.17: Número de contêineres (3×8) avaliados pelas funções de avaliação MILP e B&B

associadas aos contêineres. Já no algoritmo B&B, o fato de todos os produtos de uma mesma requisição serem explorados por um único nó da árvore de busca do algoritmo torna o processo mais eficiente, uma vez que dessa forma todo o contêiner é avaliado mais rapidamente.

É facilmente observado pelos gráficos que o algoritmo B&B apresenta maior eficiência comparado à formulação MILP, sendo capaz de avaliar um número maior de contêineres em menos tempo para todos os tipos de contêineres definidos. Portanto, a partir de agora, quando se pretender obter as melhores de rotas de um dado contêiner, será utilizado o algoritmo B&B.

A Tabela 4.4 mostra uma comparação entre a função de avaliação exata B&B e a função de avaliação heurística GH. Os dados são referentes à avaliação de 1000 planos de carregamento, criados aleatoriamente, para cada tipo de contêiner. A coluna t(ms) informa o tempo, em milisegundos, gasto por cada função para avaliar todos os 1000 planos de carregamento. A coluna Opt informa a porcentagem de vezes que a função heurística GH encontrou a mesma solução (ótima) que o método exato B&B. E, por fim, a coluna RPD informa a diferença relativa média entre as 1000 avaliações realizadas pelos dois métodos.

Tabela 4.4: Comparação entre as funções de avaliação B&B e GH utilizadas para avaliar as soluções do DVRPMSHD.

$R \times L$	B&B	GH		
	t(ms)	t(ms)	Opt	RPD
(2×2)	1.4	0.4	64.5%	4.3%
(2×3)	2.0	0.6	26.8%	6.4%
(2×4)	3.4	0.8	17.4%	7.4%
(2×5)	5.5	0.9	11.2%	8.2%
(3×4)	46.7	1.1	2.2%	12.9%
(3×5)	148.6	1.3	1.7%	12.5%
(3×6)	495.8	1.5	0.3%	13.1%
(3×8)	4084.7	1.6	0.1%	13.4%

Pelos resultados descritos na Tabela 4.4 é possível constatar que, embora a heurística GH seja mais rápida para avaliar todos os tipos de contêineres, apenas para contêineres menores ela mostrou-se eficiente em relação à qualidade das rotas encontradas (colunas Opt e RPD). Percebe-se uma drástica queda na eficiência da

heurística GH quando os contêineres passam a ter 3 pilhas, apresentando baixa porcentagem de ótimas rotas encontradas (coluna Opt) e alta diferença relativa média entre as rotas encontradas e as rotas ótimas (coluna RPD).

Da mesma forma que o DVRPMS, o DVRPMSHD também apresenta a característica que um determinado par de rotas (rota de coleta e rota de entrega) pode ser alçancado através de diferentes planos de carregamento. Referida característica aumenta as chances de que a heurística GH encontre melhores rotas.

4.5.3 Calibração dos parâmetros do algoritmo

A fim de obter o melhor desempenho do algoritmo heurístico SA proposto para o DVRPMSHD (Algoritmo 6), foi realizado um experimento estatístico para determinar a melhor configuração de valores associados aos parâmetros do algoritmo. A metodologia desse experimento segue os mesmos passos descritos na calibração dos algoritmos propostos para o DVRPMS (Seção 3.5.3).

O algoritmo heurístico SA proposto apresenta quatro parâmetros controlados: temperatura inicial (T_i), temperatura final (T_f), número de iterações em cada temperatura (N) e fator de resfriamento (α). A Tabela 4.5 mostra os diferentes valores testados dos parâmetros do algoritmo.

Tabela 4.5: Conjunto de valores testados na calibração do SA.

Parâmetro	Valores testados	#
T_i	15, 20, 50, 100	4
T_f	1, 5, 10	3
N	$\|K\|\|I\|$, 100, $\|I\|^2$, $2\|I\|^2$, $\|K\|\|I\|^2$	5
α	0.001, 0.002, 0.005, 0.010	4

Após os testes estatísticos, os parâmetros determinados para cada grupo de instância são mostrados na Tabela 4.6.

Tabela 4.6: Parâmetros determinados para o algoritmo heurístico SA.

Algoritmo	Grupo de instâncias	Parâmetros selecionados			
SA	C	$T_i = 20$,	$T_f = 10$,	$N = 2\|I\|^2$,	$\alpha = 0.001$
	$\neg C$	$T_i = 50$,	$T_f = 5$,	$N = \|I\|^2$,	$\alpha = 0.005$

4.5.4 Resultados experimentais

Esta seção é destinada à apresentação dos resultados obtidos pelas abordagens exatas e heurísticas propostas neste trabalho para o DVRPMSHD. É importante destacar que, uma vez que não há na literatura nenhum trabalho que proponha outros algoritmos para o problema, a comparação será realizada utilizando apenas os resultados descritos neste trabalho.

As Tabelas 4.7 e 4.9 reportam, respectivamente, os resultados para as instâncias do grupo C e do grupo $\neg C$. Em ambas as tabelas as três primeiras colunas descrevem as instâncias, sendo que cada instância é identificada por um número (coluna ID), um nome indicando as regiões de coleta e entrega (coluna R) e um tipo de instância (coluna T). Os resultados obtidos pela formulação ILP são descritos nas colunas seguintes, onde a coluna UB informa o valor da melhor solução obtida, a coluna LB_0 informa o valor do *lower-bound* computado após a relaxação linear da formulação de ILP, a coluna $\%g_0$ mostra o *gap* entre UB e LB_0, calculado por (UB - LB_0) / UB, a coluna %g informa o *gap* retornado pelo CPLEX, a coluna Opt indica por um asterisco as instâncias que foram resolvidas na otimalidade e a coluna t(s) informa o tempo de processamento total. Os resultados obtidos pelo algoritmo *branch-and-price* são descritos nas colunas UB, LB_0, $\%g_0$, #N, #C, Opt e t(s), que informam respectivamente, o valor da melhor solução obtida, o *lower-bound* calculado no nó raiz da árvore de busca do algoritmo, o número de nós expandidos da árvore de busca, o número de colunas adicionadas pelo algoritmo, se o algoritmo foi capaz de provar a otimalidade da solução (indicado por um asterisco) e o tempo de processamento do algoritmo. Com relação aos resultados alcançados pelo algoritmo heurístico SA, as colunas Avg, Best e t(s) informam, respectivamente, o valor da solução média obtida, o valor da melhor solução, e o tempo total de processamento gasto em 10 execuções. A última coluna de ambas as

tabelas informam o valor da melhor solução conhecida para cada instância de dados.

É importante mencionar que o tempo de processamento da formulação ILP e do algoritmo *branch-and-price* foram limitados à 1 hora, sendo que cada célula vazia nas tabelas significa que a referida informação não foi obtida dentro do tempo de processamento limitado. Também é importante apontar que os melhores resultados foram destacados em negrito com o objetivo de evidenciar a qualidade dos resultados alcançados por cada método.

Vale ressaltar que, para todas as instâncias em que o algoritmo *branch-and-price* não foi capaz de encontrar uma solução inteira, todas as variáveis λ_k^s incluídas no problema mestre (colunas) foram definidas como binárias ($\lambda_k^s \in \{0,1\} \quad \forall k \in K, \forall s \in \widehat{\mathcal{S}_k}$) e o problema mestre resolvido novamente. O objetivo deste processo foi encontrar uma solução inteira usando as colunas já inseridas no problema mestre.

Tabela 4.7: Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMSHD, considerando as instâncias do grupo C.

Instância			Formulação ILP						Branch-and-price							SA			Melhor solução
ID	R	T	UB	LB_0	$\%g_0$	%g	Opt	t(s)	UB	LB_0	$\%g_0$	#N	#C	Opt	t(s)	Avg	Best	t(s)	conhecida
001	R05	(a)	**623**	382.0	38.68	0.00	*	0.2	**623**	623.0	0.00	1	12	*	0.3	**623.0**	**623**	1.7	623
002	R06		**610**	394.0	35.41	0.00	*	0.2	**610**	610.0	0.00	1	16	*	0.5	**610.0**	**610**	1.7	610
003	R07		**608**	326.0	46.38	0.00	*	0.2	**608**	608.0	0.00	1	16	*	0.5	**608.0**	**608**	1.7	608
004	R08		**614**	344.0	43.97	0.00	*	0.2	**614**	614.0	0.00	1	18	*	0.4	**614.0**	**614**	1.7	614
005	R09		**632**	418.0	33.86	0.00	*	0.2	**632**	632.0	0.00	1	12	*	0.3	**632.0**	**632**	1.7	632
006	R05	(b)	**565**	386.6	31.58	0.00	*	0.2	**565**	565.0	0.00	1	15	*	1.0	**565.0**	**565**	1.8	565
007	R06		**650**	434.7	33.12	0.00	*	0.2	**650**	650.0	0.00	1	15	*	1.1	**650.0**	**650**	1.8	650
008	R07		**559**	344.1	38.44	0.00	*	0.2	**559**	559.0	0.00	1	15	*	1.4	**559.0**	**559**	2.0	559
009	R08		**592**	373.6	36.89	0.00	*	0.3	**592**	592.0	0.00	1	14	*	1.2	**592.0**	**592**	2.0	592
010	R09		**583**	429.2	26.38	0.00	*	0.2	**583**	583.0	0.00	1	15	*	1.0	**583.0**	**583**	1.9	583
011	R05	(c)	**814**	382.0	53.07	0.00	*	0.4	**814**	814.0	0.00	1	15	*	0.3	**814.0**	**814**	1.5	814
012	R06		**901**	394.0	56.27	0.00	*	0.3	**901**	901.0	0.00	1	21	*	0.2	**901.0**	**901**	1.5	901
013	R07		**737**	326.0	55.77	0.00	*	0.2	**737**	737.0	0.00	1	15	*	0.2	**737.0**	**737**	1.5	737
014	R08		**760**	344.0	54.74	0.00	*	0.4	**760**	760.0	0.00	1	18	*	0.3	**760.0**	**760**	1.5	760
015	R09		**734**	418.0	43.05	0.00	*	0.2	**734**	734.0	0.00	1	21	*	0.3	**734.0**	**734**	1.5	734
016	R05	(d)	**668**	336.2	49.67	0.00	*	7.7	**668**	668.0	0.00	1	37	*	25.9	**668.0**	**668**	29.6	668
017	R06		**799**	549.0	31.29	0.00	*	1.6	**799**	799.0	0.00	1	24	*	9.7	**799.0**	**799**	31.2	799
018	R07		**623**	385.6	38.11	0.00	*	8.8	**623**	623.0	0.00	1	19	*	12.3	**623.0**	**623**	26.8	623
019	R08		**783**	546.8	30.17	0.00	*	4.8	**783**	783.0	0.00	1	36	*	18.0	**783.0**	**783**	37.4	783
020	R09		**709**	383.0	45.98	0.00	*	2.7	**709**	709.0	0.00	1	27	*	10.3	**709.0**	**709**	27.2	709
021	R05	(e)	**810**	319.0	60.62	0.00	*	14.7	**810**	810.0	0.00	1	54	*	4.5	**810.0**	**810**	4.8	810
022	R06		**923**	530.0	42.58	0.00	*	5.7	**923**	923.0	0.00	1	51	*	2.7	**923.0**	**923**	4.7	923
023	R07		**797**	380.0	52.32	0.00	*	5.5	**797**	797.0	0.00	1	21	*	1.5	**797.0**	**797**	4.7	797
024	R08		**958**	544.0	43.22	0.00	*	20.4	**958**	958.0	0.00	1	60	*	4.6	**958.0**	**958**	4.6	958
025	R09		**845**	366.0	56.69	0.00	*	6.9	**845**	845.0	0.00	1	39	*	2.3	**845.0**	**845**	4.5	845

Continua na próxima página

Tabela 4.7: (continuação) Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMSHD, considerando as instâncias do grupo C.

Instância			Formulação ILP						Branch-and-price							SA			Melhor solução
ID	R	T	UB	LB_0	$\%g_0$	%g	Opt	t(s)	UB	LB_0	$\%g_0$	#N	#C	Opt	t(s)	Avg	Best	t(s)	conhecida
026	R05	(f)	**1004**	319.0	68.23	0.00	*	7.2	**1004**	1004.0	0.00	1	43	*	4.7	1013.7	**1004**	4.1	1004
027	R06		**1152**	530.0	53.99	0.00	*	24.8	**1152**	1152.0	0.00	1	49	*	3.9	**1152.0**	**1152**	4.2	1152
028	R07		**969**	380.0	60.78	0.00	*	6.9	**969**	966.5	0.26	11	56	*	11.0	972.2	**969**	4.2	969
029	R08		**1072**	544.0	49.25	0.00	*	28.3	**1072**	1072.0	0.00	1	45	*	3.5	1079.4	**1072**	4.2	1072
030	R09		**903**	366.0	59.47	0.00	*	4.6	**903**	903.0	0.00	1	44	*	2.4	914.0	**903**	4.2	903
031	R05	(g)	**696**	344.0	50.57	0.00	*	140.1	**696**	696.0	0.00	1	70	*	199.9	**696.0**	**696**	104.4	696
032	R06		**902**	508.0	43.68	0.00	*	59.6	**902**	902.0	0.00	1	78	*	91.7	**902.0**	**902**	87.2	902
033	R07		**738**	369.0	50.00	0.00	*	1612.1	**738**	738.0	0.00	1	66	*	381.6	**738.0**	**738**	86.3	738
034	R08		**848**	595.0	29.83	0.00	*	90.2	**848**	848.0	0.00	1	84	*	123.7	**848.0**	**848**	93.5	848
035	R09		**810**	516.0	36.30	0.00	*	213.3	**810**	810.0	0.00	1	72	*	77.6	**810.0**	**810**	91.6	810
036	R05	(h)	**874**	344.0	60.64	0.00	*	217.6	**874**	874.0	0.00	1	93	*	38.0	**874.0**	**874**	12.8	874
037	R06		**957**	508.0	46.92	0.00	*	78.4	**957**	957.0	0.00	1	102	*	29.7	**957.0**	**957**	12.3	957
038	R07		**894**	369.0	58.72	0.00	*	344.5	**894**	894.0	0.00	1	90	*	36.3	**894.0**	**894**	11.9	894
039	R08		**1012**	595.0	41.21	0.00	*	317.1	**1012**	1012.0	0.00	1	105	*	44.1	**1012.0**	**1012**	12.2	1012
040	R09		**880**	516.0	41.36	0.00	*	192.6	**880**	880.0	0.00	1	96	*	21.7	**880.0**	**880**	12.6	880
041	R05	(i)	**979**	344.0	64.86	0.00	*	446.5	**979**	979.0	0.00	1	104	*	17.0	**979.0**	**979**	9.4	979
042	R06		**1175**	508.0	56.77	0.00	*	484.6	**1175**	1175.0	0.00	1	92	*	14.3	**1175.0**	**1175**	9.1	1175
043	R07		**1015**	369.0	63.65	0.00	*	654.0	**1015**	1015.0	0.00	1	100	*	12.2	**1015.0**	**1015**	9.2	1015
044	R08		**1090**	595.0	45.41	0.00	*	433.5	**1090**	1090.0	0.00	1	100	*	12.3	**1090.0**	**1090**	8.8	1090
045	R09		**1005**	516.0	48.66	0.00	*	413.4	**1005**	1005.0	0.00	1	88	*	7.6	**1005.0**	**1005**	9.2	1005
046	R05	(j)	**894**	468.0	47.65	0.00	*	2580.1	**894**	894.0	0.00	1	102	*	2591.0	906.6	897	9.6	894
047	R06		**976**	622.0	36.27	0.00	*	2509.5	**976**	976.0	0.00	1	122	*	2167.6	983.0	983	9.6	976
048	R07		**883**	447.0	49.38	3.51		1 h	960			1	98		1 h	**883.0**	**883**	9.7	883
049	R08		**969**	688.0	29.00	0.00	*	616.4	**969**	969.0	0.00	1	136	*	1625.2	995.7	992	10.0	969
050	R09		**874**	635.0	27.35	0.00	*	725.2	**874**	874.0	0.00	1	104	*	672.3	898.6	889	9.9	874

Continua na próxima página

Tabela 4.7: (continuação) Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMSHD, considerando as instâncias do grupo C.

Instância			Formulação ILP						Branch-and-price							SA			Melhor solução
ID	R	T	UB	LB_0	$\%g_0$	%g	Opt	t(s)	UB	LB_0	$\%g_0$	#N	#C	Opt	t(s)	Avg	Best	t(s)	conhecida
051	R05	(k)	**1061**	468.0	55.89	5.79		1 h	**1061**	1061.0	0.00	1	165	*	1019.9	**1061.0**	**1061**	27.7	1061
052	R06		**1171**	622.0	46.88	2.97		1 h	**1171**	1155.5	1.32	13	247		1 h	**1171.0**	**1171**	27.6	1171
053	R07		1072	447.0	58.30	12.20		1 h	**1065**	1065.0	0.00	1	171	*	1580.3	**1065.0**	**1065**	26.0	1065
054	R08		**1141**	688.0	39.70	0.00	*	1921.2	**1141**	1141.0	0.00	1	168	*	419.7	**1141.0**	**1141**	26.4	1141
055	R09		**1047**	635.0	39.35	4.62		1 h	**1047**	1045.0	0.19	31	247	*	1316.1	**1047.0**	**1047**	27.5	1047
056	R05	(l)	1211	468.0	61.35	25.28		1 h	**1194**	1187.3	0.56	63	345	*	2306.7	**1194.0**	**1194**	20.6	1194
057	R06		**1268**	622.0	50.95	18.67		1 h	**1268**	1268.0	0.00	1	126	*	128.6	**1268.0**	**1268**	20.3	1268
058	R07		1257	447.0	64.44	28.94		1 h	**1215**	1210.5	0.37	19	240	*	3182.7	**1215.0**	**1215**	19.9	1215
059	R08		1283	688.0	46.38	14.70		1 h	**1277**	1277.0	0.00	1	145	*	241.3	**1277.0**	**1277**	20.5	1277
060	R09		1137	635.0	44.15	14.04		1 h	**1124**	1124.0	0.00	1	143	*	108.0	**1124.0**	**1124**	20.3	1124
061	R05	(m)	1026	597.0	41.81	16.33		1 h				1	84		1 h	1016.4	**1010**	15.5	1010
062	R06		1071	639.0	40.34	14.94		1 h	1386			1	80		1 h	1061.0	**1044**	15.8	1044
063	R07		1005	536.0	46.67	23.39		1 h	1341			1	76		1 h	974.1	**972**	15.7	972
064	R08		1103	680.0	38.35	16.01		1 h	1268			1	78		1 h	1038.9	**1032**	15.5	1032
065	R09		961	608.0	36.73	15.88		1 h	1045			1	84		1 h	941.3	**935**	15.7	935
066	R05	(n)	1146	597.0	47.91	23.78		1 h				1	189		1 h	**1142.0**	**1142**	223.3	1142
067	R06		1175	639.0	45.62	23.74		1 h	1173			1	201		1 h	**1165.0**	**1165**	205.5	1165
068	R07		1200	536.0	55.33	34.16		1 h	**1103**			1	174		1 h	**1103.0**	**1103**	231.2	1103
069	R08		1285	680.0	47.08	26.49		1 h	1751			1	186		1 h	**1186.0**	**1186**	249.5	1186
070	R09		1160	608.0	47.59	29.09		1 h	1187			1	216		1 h	**1073.0**	**1073**	235.1	1073
071	R05	(o)	1344	597.0	55.58	34.67		1 h				1	169		1 h	**1281.0**	**1281**	97.1	1281
072	R06		1276	639.0	49.92	27.58		1 h	**1273**	1273.0	0.00	1	163	*	3078.4	**1273.0**	**1273**	75.3	1273
073	R07		1448	536.0	62.98	46.50		1 h	**1249**			1	148		1 h	**1249.0**	**1249**	98.5	1249
074	R08		1421	680.0	52.15	32.69		1 h	**1273**	1273.0	0.00	1	163	*	3127.6	1326.0	1326	98.9	1273
075	R09		1328	608.0	54.22	40.04		1 h	1193			1	177		1 h	**1185.0**	**1185**	82.6	1185

Continua na próxima página

Tabela 4.7: (continuação) Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMSHD, considerando as instâncias do grupo C.

Instância			Formulação ILP						Branch-and-price							SA			Melhor solução
ID	R	T	UB	LB_0	$\%g_0$	%g	Opt	t(s)	UB	LB_0	$\%g_0$	#N	#C	Opt	t(s)	Avg	Best	t(s)	conhecida
076	R05	(p)	1546	616.0	60.16	46.56		1 h				1	28		1 h	1122.5	**1086**	32.1	1086
077	R06		2706	703.0	74.02	66.78		1 h				1	16		1 h	1152.5	**1114**	33.4	1114
078	R07		2332	551.0	76.37	65.22		1 h	2510			1	58		1 h	1147.4	**1118**	32.9	1118
079	R08		2636	680.0	74.20	66.38		1 h				1	60		1 h	1187.6	**1161**	32.4	1161
080	R09		2001	628.0	68.62	61.24		1 h				1	14		1 h	1110.7	**1085**	33.6	1085
081	R05	(q)	1834	616.0	66.41	56.56		1 h				1	72		1 h	1232.2	**1203**	32.9	1203
082	R06		2079	703.0	66.19	56.44		1 h				1	66		1 h	1295.9	**1270**	33.2	1270
083	R07		2624	551.0	79.00	70.08		1 h				1	72		1 h	1321.6	**1298**	32.6	1298
084	R08		1572	680.0	56.74	46.82		1 h				1	63		1 h	1317.4	**1300**	33.1	1300
085	R09		1768	628.0	64.48	56.40		1 h				1	69		1 h	1240.6	**1220**	32.9	1220
086	R05	(r)	1774	616.0	65.28	57.24		1 h				1	184		1 h	**1354.0**	**1354**	436.1	1354
087	R06		2107	703.0	66.64	58.57		1 h				1	176		1 h	1422.4	**1422**	452.2	1422
088	R07		1938	551.0	71.57	61.13		1 h				1	192		1 h	**1444.0**	**1444**	440.4	1444
089	R08		1779	680.0	61.78	52.89		1 h				1	172		1 h	1442.6	**1442**	433.1	1442
090	R09		1955	628.0	67.88	61.48		1 h				1	196		1 h	1356.3	**1356**	439.1	1356
Média			1130.2	516.3	50.52	15.78	$\frac{50}{90}$	1757.8	-	-	-	2.5	93.0	$\frac{60}{90}$	1475.5	992.2	988.1	58.7	987.0

Os resultados descritos na Tabela 4.7 mostram que, com uma hora de processamento, a formulação de ILP e o algoritmo *branch-and-price* (B&P) foram capazes de resolver na otimalidade, respectivamente, 50 (55.6%) e 60 (66.7%) das 90 instâncias do grupo C. Observa-se também que para 72.2% das instâncias o algoritmo B&P encontrou soluções iguais ou melhores em comparação com a solução obtida pela formulação de ILP e que para 18.9% das instâncias o algoritmo B&P não foi capaz de encontrar nenhuma solução dentro de uma hora de processamento.

Observe que, para todas as instâncias que o nó raiz da árvore de busca do algoritmo B&P foi totalmente processado, o *gap* em relação ao LB_0 (coluna $\%g_0$) é zero ou próximo de zero, enquanto o *gap* relacionado a relaxação linear da formulação ILP (coluna $\%g_0$) é muito elevado para todas as instâncias. Isso indica que o modelo baseado na formulação de particionamento de conjuntos é mais forte, uma vez que tem um valor de relaxação linear melhor (mais próximo do valor da solução ótima).

Também constata-se que, para várias instâncias, o algoritmo B&P consome muito tempo de processamento, não sendo capaz de processar completamente o nó raiz (a solução ótima linear LB_0 não é encontrada). Mesmo quando isso ocorre, as colunas já geradas podem ser suficientes para compor uma boa solução inteira. Isso ocorre para as instâncias com ID 067, 068, 073 e 075 que apresentam valores de UB melhores do que os encontrados utilizando a formulação de ILP.

O algoritmo heurístico SA encontrou a mesma solução para 55 (91.7%) das 60 instâncias do grupo C que se conhece as soluções ótimas. Analisando a eficiência geral do algoritmo SA para as instâncias do grupo C, constata-se que os melhores resultados obtidos pelo SA (coluna Best) são iguais ou melhores que os resultados alcançados pela formulação ILP e pelo algoritmo B&P em, respectivamente, 86 (95.6%) e 85 (94.4%) das 90 instâncias. Note que quatro instâncias em que o algoritmo heurístico não obteve os melhores resultados são do tipo j, composto por dois veículos de contêineres (3×5), que são avaliados de forma exata pelo algoritmo B&B apenas na busca local aplicada ao final do algoritmo SA.

Com relação ao tempo de execução de cada método de resolução, o algoritmo SA, em geral, se mostrou mais eficiente que a formulação ILP e o algoritmo B&P, gastando em média 58.7 segundos (última linha da Tabela 4.7) para as instâncias do grupo C.

Para analisar estatisticamente a qualidade dos resultados obtidos pelos diferentes métodos de resolução propostos para o DVRPMSHD ao resolver as instâncias do grupo C, com o objetivo de determinar se há diferença estatística significante entre resultados obtidos[7], foi realizada uma análise de variância (ANOVA), sendo utilizado como variável resposta o desvio percentual relativo (*Relative Percentage Deviation* - RPD) entre o valor objetivo da melhor solução encontrada (f_b) e o valor objetivo da solução de cada método (f_m), calculado pela equação (4.48).

$$RPD = \frac{f_m - f_b}{f_m} \ 100\% \tag{4.48}$$

A Figura 4.18 mostra o gráfico de caixa (*boxplot*) no qual é possível observar que o algoritmo heurístico SA apresenta menor variação nos resultados que os métodos exatos (ILP e B&P).

A ANOVA indicou haver uma diferença significativa entre os resultados com nível de confiança de 99%, assim foi realizado o teste de Scott & Knott [1974] para descobrir entre quais métodos ocorre essa diferença. A Tabela 4.8 informa os resultados do teste de Scott & Knott [1974], indicando haver uma diferença significativa entre os resultados alcançados pela formulação ILP e os demais resultados. Portanto, conclui-se que os dois métodos de solução propostos, exato (B&P) e heurístico (SA), se sobressaem em relação à resolução da formulação ILP, considerando as instâncias do grupo C, uma vez que a formulação ILP apresentou maior média de RPD's.

Entre os algoritmos B&P e SA não há diferença significante (mesmo identificador na coluna Grupos), porém o SA apresenta a média dos RPD's com uma ordem de grandeza menor, indicando maior eficiência do método heurístico.

[7]Para o SA foram considerados os melhores resultados encontrados, isto é, os resultados descritos na coluna Best.

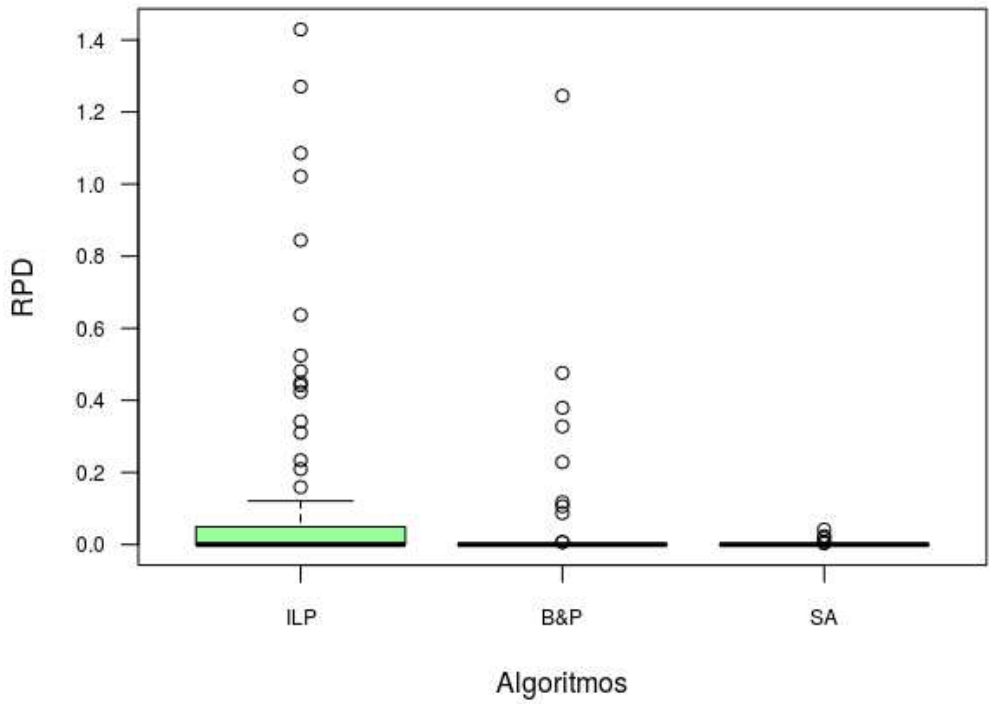


Figura 4.18: Comparação entre as médias do RPD alcançados pelos diferentes métodos para o DVRPMSHD, considerando as instâncias do grupo C.

Tabela 4.8: Resultado do teste de Scott & Knott para o RPD alcançados pelos diferentes métodos de resolução para o DVRPMSHD, considerando as instâncias do grupo C.

Grupos	Métodos	Médias
a	ILP	0.1185
b	B&P	0.0408
b	SA	0.0010

Tabela 4.9: Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMSHD, considerando as instâncias do grupo $\neg C$.

Instância			Formulação ILP						Branch-and-price							SA			Melhor solução
ID	R	T	UB	LB_0	$\%g_0$	%g	Opt	t(s)	UB	LB_0	$\%g_0$	#N	#C	Opt	t(s)	Avg	Best	t(s)	conhecida
091	R05	(s)	**481**	382.0	20.58	0.00	*	0.3	**481**	481.0	0.00	1	26	*	2.6	**481.0**	**481**	2.4	481
092	R06		**537**	394.0	26.63	0.00	*	0.2	**537**	537.0	0.00	1	31	*	2.2	**537.0**	**537**	1.6	537
093	R07		**446**	326.0	26.91	0.00	*	0.2	**446**	446.0	0.00	1	27	*	3.1	**446.0**	**446**	2.4	446
094	R08		**462**	344.0	25.54	0.00	*	0.2	**462**	462.0	0.00	1	25	*	1.7	**462.0**	**462**	2.3	462
095	R09		**501**	418.0	16.57	0.00	*	0.2	**501**	501.0	0.00	1	25	*	1.9	**501.0**	**501**	2.2	501
096	R05	(t)	**623**	382.0	38.68	0.00	*	0.5	**623**	623.0	0.00	1	39	*	0.9	**623.0**	**623**	0.5	623
097	R06		**610**	394.0	35.41	0.00	*	0.3	**610**	610.0	0.00	1	33	*	0.4	**610.0**	**610**	0.5	610
098	R07		**608**	326.0	46.38	0.00	*	0.5	**608**	608.0	0.00	1	45	*	1.1	**608.0**	**608**	0.5	608
099	R08		**614**	344.0	43.97	0.00	*	0.5	**614**	614.0	0.00	1	39	*	0.7	**614.0**	**614**	0.5	614
100	R09		**632**	418.0	33.86	0.00	*	0.5	**632**	632.0	0.00	1	45	*	0.8	**632.0**	**632**	0.5	632
101	R05	(u)	**645**	382.0	40.78	0.00	*	0.6	**645**	645.0	0.00	1	45	*	1.2	**645.0**	**645**	0.5	645
102	R06		**687**	394.0	42.65	0.00	*	0.5	**687**	687.0	0.00	1	51	*	1.2	**687.0**	**687**	0.5	687
103	R07		**688**	326.0	52.62	0.00	*	0.7	**688**	656.5	4.58	21	69	*	10.9	**688.0**	**688**	0.5	688
104	R08		**667**	344.0	48.43	0.00	*	0.8	**667**	667.0	0.00	1	49	*	1.3	**667.0**	**667**	0.5	667
105	R09		**646**	418.0	35.29	0.00	*	0.6	**646**	646.0	0.00	1	44	*	0.8	**646.0**	**646**	0.5	646
106	R05	(v)	**644**	319.0	50.47	0.00	*	10.7	**644**	644.0	0.00	1	54	*	18.0	**644.0**	**644**	5.6	644
107	R06		**799**	530.0	33.67	0.00	*	2.3	**799**	799.0	0.00	1	66	*	12.4	**799.0**	**799**	8.5	799
108	R07		**623**	380.0	39.00	0.00	*	36.1	**623**	623.0	0.00	1	66	*	70.7	**623.0**	**623**	7.6	623
109	R08		**782**	544.0	30.43	0.00	*	33.9	**782**	782.0	0.00	1	56	*	18.0	**782.0**	**782**	7.6	782
110	R09		**692**	366.0	47.11	0.00	*	4.7	**692**	692.0	0.00	1	52	*	10.3	**692.0**	**692**	6.0	692
111	R05	(w)	**753**	319.0	57.64	0.00	*	19.3	**753**	753.0	0.00	1	87	*	8.7	**753.0**	**753**	1.7	753
112	R06		**923**	530.0	42.58	0.00	*	22.5	**923**	923.0	0.00	1	84	*	7.7	**923.0**	**923**	1.7	923
113	R07		**747**	380.0	49.13	0.00	*	155	**747**	732.5	1.94	11	113	*	42.2	**747.0**	**747**	1.8	747
114	R08		**879**	544.0	38.11	0.00	*	76.6	**879**	879.0	0.00	1	84	*	9.6	**879.0**	**879**	1.9	879
115	R09		**761**	366.0	51.91	0.00	*	17.3	**761**	761.0	0.00	1	84	*	5.5	**761.0**	**761**	1.7	761

Continua na próxima página

Tabela 4.9: (continuação) Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMSHD, considerando as instâncias do grupo $\neg C$.

Instância			Formulação ILP						Branch-and-price							SA			Melhor solução conhecida
ID	R	T	UB	LB_0	$\%g_0$	%g	Opt	t(s)	UB	LB_0	$\%g_0$	#N	#C	Opt	t(s)	Avg	Best	t(s)	
116	R05	(x)	**810**	319.0	60.62	0.00	*	459.5	**810**	809.5	0.06	79	173	*	77.6	810.5	**810**	1.5	810
117	R06		**923**	530.0	42.58	0.00	*	31.1	**923**	923.0	0.00	1	84	*	2.3	**923.0**	**923**	1.4	923
118	R07		**797**	380.0	52.32	0.00	*	59.2	**797**	797.0	0.00	1	80	*	3.8	**797.0**	**797**	1.5	797
119	R08		**958**	544.0	43.22	0.00	*	219.3	**958**	943.5	1.51	153	236	*	140.9	**958.0**	**958**	1.5	958
120	R09		**831**	366.0	55.96	0.00	*	41.7	**831**	825.0	0.72	47	146	*	40.5	**831.0**	**831**	1.4	831
121	R05	(y)	**696**	344.0	50.57	0.00	*	131.2	**696**	696.0	0.00	1	98	*	1020.5	715.3	715	2.3	696
122	R06		**837**	508.0	39.31	0.00	*	46.5	**837**	837.0	0.00	1	86	*	181.1	868.7	868	2.2	837
123	R07		**708**	369.0	47.88	0.00	*	974.5	**708**	708.0	0.00	1	98	*	1902.7	725.3	717	2.2	708
124	R08		**836**	595.0	28.83	0.00	*	147.5	**836**	836.0	0.00	1	106	*	452.9	869.4	865	2.3	836
125	R09		**767**	516.0	32.72	0.00	*	71.6	**767**	767.0	0.00	1	112	*	190.6	771.4	**767**	2.2	767
126	R05	(z)	**837**	344.0	58.90	0.00	*	664.5	**837**	825.7	1.35	49	357	*	1426.0	**837.0**	**837**	4.5	837
127	R06		**956**	508.0	46.86	0.00	*	206.2	**956**	956.0	0.00	1	156	*	67.2	**956.0**	**956**	4.5	956
128	R07		**850**	369.0	56.59	0.00	*	915.1	**850**	850.0	0.00	1	132	*	269.6	**850.0**	**850**	4.3	850
129	R08		**984**	595.0	39.53	0.00	*	727.6	**984**	984.0	0.00	25	250	*	684.4	**984.0**	**984**	4.5	984
130	R09		**851**	516.0	39.37	0.00	*	361.4	**851**	851.0	0.00	1	138	*	43.8	**851.0**	**851**	4.4	851
131	R05	(α)	**874**	344.0	60.64	0.00	*	1276.2	**874**	874.0	0.00	1	125	*	57.8	892.0	**874**	3.4	874
132	R06		**957**	508.0	46.92	0.00	*	407.9	**957**	957.0	0.00	1	104	*	33.8	959.5	**957**	3.6	957
133	R07		**894**	369.0	58.72	0.00	*	1906.1	**894**	894.0	0.00	1	124	*	48.2	895.5	**894**	3.3	894
134	R08		**1012**	595.0	41.21	0.00	*	1672.6	**1012**	1012.0	0.00	1	126	*	48.2	1017.2	**1012**	3.4	1012
135	R09		**880**	516.0	41.36	0.00	*	1413.1	**880**	880.0	0.00	1	140	*	39.9	889.1	**880**	3.4	880
136	R05	(β)	891	468.0	47.47	6.98		1 h	1098			1	78		1 h	902.6	**890**	3.9	890
137	R06		**926**	622.0	32.83	0.00	*	1609.9	**926**	926.0	0.00	1	154	*	2917.2	943.2	935	4.0	926
138	R07		860	447.0	48.02	7.50		1 h	1094			1	62		1 h	863.7	**855**	3.9	855
139	R08		**965**	688.0	28.70	0.00	*	1011.6	979			1	128		1 h	1008.2	993	4.1	965
140	R09		**856**	635.0	25.82	0.00	*	784.4	**856**	856.0	0.00	1	154	*	1401.7	879.8	864	4.0	856

Continua na próxima página

Tabela 4.9: (continuação) Comparação entre as soluções obtidas pelos métodos propostos para o DVRPMSHD, considerando as instâncias do grupo $\neg C$.

Instância			Formulação ILP						Branch-and-price							SA			Melhor solução conhecida
ID	R	T	UB	LB_0	$\%g_0$	%g	Opt	t(s)	UB	LB_0	$\%g_0$	#N	#C	Opt	t(s)	Avg	Best	t(s)	
141	R05	(γ)	**994**	468.0	52.92	8.97		1 h	**994**	994.0	0.00	1	198	*	3350.6	**994.0**	**994**	28.8	994
142	R06		**1045**	622.0	40.48	0.00	*	3331.8	**1045**	1045.0	0.00	1	177	*	1354.3	**1045.0**	**1045**	29.0	1045
143	R07		**1032**	447.0	56.69	21.34		1 h	1068			1	177		1 h	1033.8	**1032**	25.9	1032
144	R08		**1082**	688.0	36.41	0.00	*	3251.2	**1082**	1082.0	0.00	1	168	*	417.8	**1082.0**	**1082**	28.5	1082
145	R09		1000	635.0	36.50	11.05		1 h	**999**	993.0	0.60	59	490	*	3038.4	999.8	**999**	27.7	999
146	R05	(δ)	1136	468.0	58.80	24.37		1 h	**1088**	1088.0	0.00	1	156	*	2263.5	1088.6	**1088**	15.5	1088
147	R06		1167	622.0	46.70	14.55		1 h	**1141**	1141.0	0.00	1	141	*	757.5	1143.2	**1141**	15.4	1141
148	R07		1133	447.0	60.55	26.34		1 h	1129	1102.3	2.36	5	152		1 h	1123.8	**1122**	13.9	1122
149	R08		1236	688.0	44.34	17.48		1 h	**1206**	1200.0	0.50	19	249	*	2996.8	1207.1	**1206**	16.0	1206
150	R09		1094	635.0	41.96	19.88		1 h	**1066**	1066.0	0.00	1	136	*	205.3	1068.1	**1066**	14.8	1066
151	R05	(ζ)	980	597.0	39.08	13.38		1 h	1762			1	56		1 h	1000.2	**977**	7.4	977
152	R06		1067	639.0	40.11	18.82		1 h				1	36		1 h	1037.8	**1016**	7.4	1016
153	R07		**943**	536.0	43.16	19.29		1 h	1482			1	34		1 h	979.8	954	7.4	943
154	R08		1090	680.0	37.61	16.99		1 h				1	44		1 h	1062.8	**1021**	7.4	1021
155	R09		**901**	608.0	32.52	8.50		1 h	1161			1	50		1 h	923.1	915	7.6	901
156	R05	(η)	1132	597.0	47.26	26.69		1 h	1544			1	66		1 h	1118.0	**1084**	6.8	1084
157	R06		1128	639.0	43.35	22.84		1 h	1889			1	54		1 h	1136.0	**1127**	6.6	1127
158	R07		1105	536.0	51.49	31.26		1 h				1	59		1 h	1057.3	**1032**	6.8	1032
159	R08		1202	680.0	43.43	25.60		1 h	1446			1	87		1 h	1151.5	**1109**	6.8	1109
160	R09		1090	608.0	44.22	25.96		1 h	1217			1	87		1 h	1035.1	**1014**	6.8	1014
161	R05	(θ)	1331	597.0	55.15	38.94		1 h	1367			1	252		1 h	1146.3	**1142**	41.9	1142
162	R06		1306	639.0	51.07	33.06		1 h	**1165**			1	256		1 h	**1165.0**	**1165**	40.0	1165
163	R07		1408	536.0	61.93	48.11		1 h	1428			1	164		1 h	**1103.0**	**1103**	44.2	1103
164	R08		1299	680.0	47.65	29.92		1 h	1290			1	228		1 h	1188.6	**1186**	47.0	1186
165	R09		1177	608.0	48.34	34.07		1 h	1259			1	236		1 h	1084.3	**1073**	36.8	1073
Média			883.8	490.2	43.51	7.36	$\frac{50}{75}$	1494.8	-	-	-	7.1	113.9	$\frac{55}{75}$	1302.3	872.3	867.0	8.5	864.9

Sobre os resultados obtidos para as instâncias do grupo $\neg C$, descritos na Tabela 4.9, também é possível notar que o algoritmo B&P encontrou mais soluções ótimas comparado à formulação ILP. Para as 75 instâncias desse grupo, o B&P e a formulação ILP resolveram, respectivamente, com otimalidade 55 (73.3%) e 50 (66.7%) instâncias.

Ainda sobre os resultados da Tabela 4.9 (grupo $\neg C$), é possível testemunhar algumas características com respeito à formulação ILP e ao algoritmo B&P que também foram observadas nos resultados descritos da Tabela 4.7 (grupo C), como a superioridade da relaxação linear do modelo baseado na formulação de particionamento de conjuntos em comparação à relaxação linear da formulação ILP (colunas $\%g_0$), e também observa-se que, mesmo o algoritmo B&P não resolvendo com otimalidade boa parte das instâncias, as colunas já geradas podem ser suficientes para compor soluções inteiras melhores que as obtidas pela formulação ILP (instâncias com ID 162 e 164).

Para as 56 instâncias do grupo $\neg C$ que se conhece as soluções ótimas, o algoritmo heurístico SA foi capaz de encontrar a mesma solução para 48 (85.7%) delas. Considerando todas as 75 instâncias do grupo $\neg C$, o algoritmo SA obteve resultados iguais ou melhores que os resultados alcançados pela formulação ILP e pelo algoritmo B&P em, respectivamente, 66 (88.0%) e 68 (90.7%) instâncias. A Figura 4.19 mostra o gráfico de caixa (*boxplot*) no qual é possível observar que o algoritmo heurístico SA apresenta uma menor dispersão dos resultados em comparação aos métodos exatos (ILP e B&P).

Analisando estatisticamente os resultados, a ANOVA indicou haver diferença significativa entre os resultados com nível de confiança de 99%. E segundo os resultados do teste de Scott & Knott [1974] (Tabela 4.10), essa diferença ocorre entre os resultados alcançados pelo algoritmo B&P e os demais resultados obtidos pela formulação ILP e pelo algoritmo SA. Entre a formulação ILP e o algoritmo SA não há diferença significante (mesmo identificador na coluna Grupos), porém o SA apresenta a média dos RPD's com uma ordem de grandeza menor, indicando maior eficiência do método heurístico.

A Tabela 4.11 apresenta um resumo dos resultados alcançados por todos os métodos propostos para resolver o DVRPMSHD. Cada linha da tabela se refere às cinco instâncias de cada tipo (R05, R06, R07, R08 e R09). A coluna # informa

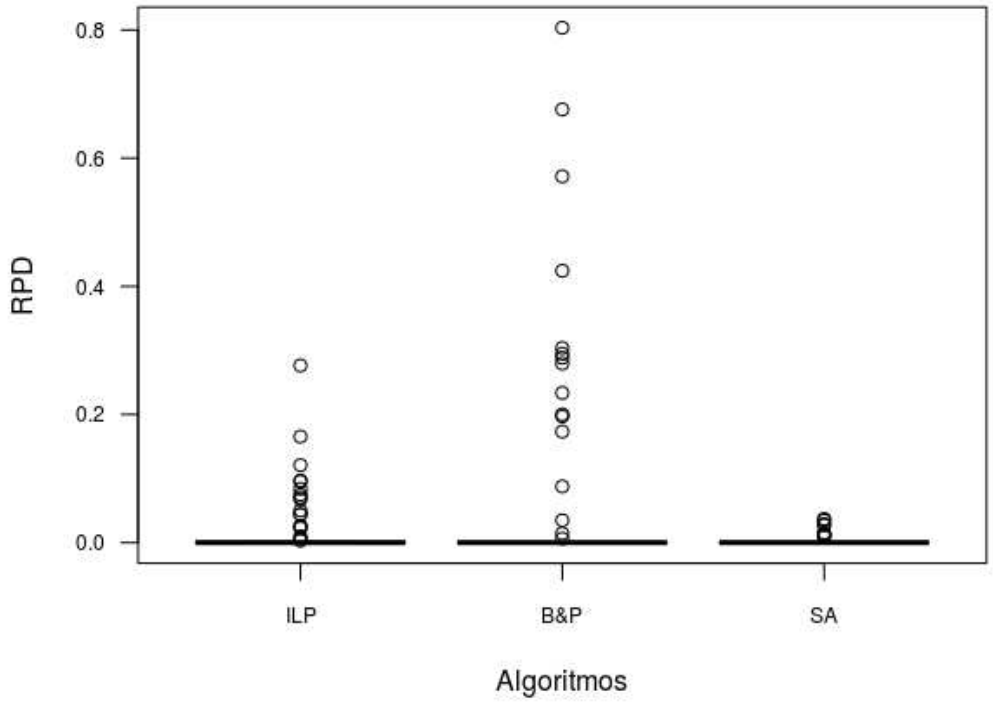


Figura 4.19: Comparação entre as médias do RPD alcançados pelos diferentes métodos para o DVRPMSHD, considerando as instâncias do grupo $\neg C$.

Tabela 4.10: Resultado do teste de Scott & Knott para o RPD alcançados pelos diferentes métodos de resolução para o DVRPMSHD, considerando as instâncias do grupo $\neg C$.

Grupos	Métodos	Médias
a	B&P	0.0637
b	ILP	0.0172
b	SA	0.0025

o número de instâncias que cada método foi capaz de encontrar a melhor solução conhecida e a coluna t(s) informa a tempo médio de execução das cinco instâncias.

Pelos resultados da Tabela 4.11, verifica-se a eficiência dos métodos propostos neste trabalho, principalmente, o algoritmo SA, o qual obteve um maior número de melhores soluções para quase todos os tipos de instâncias.

Tabela 4.11: Número de melhores resultados alcançados por algoritmo.

Tipo	ILP		Branch-and-price		SA	
	#	t(s)	#	t(s)	#	t(s)
(a)	5	0.2	5	0.4	5	1.7
(b)	5	0.2	5	1.1	5	1.9
(c)	5	0.3	5	0.3	5	1.5
(d)	5	5.1	5	15.2	5	30.4
(e)	5	10.6	5	3.1	5	4.7
(f)	5	14.4	5	5.1	5	4.2
(g)	5	423.1	5	174.9	5	92.6
(h)	5	230.0	5	34.0	5	12.4
(i)	5	486.4	5	12.7	5	9.1
(j)	5	2006.2	4	2131.2	1	9.8
(k)	4	3264.2	5	1587.2	5	27.0
(l)	1	1 h	5	1193.5	5	20.3
(m)	0	1 h	0	1 h	5	15.6
(n)	0	1 h	1	1 h	5	228.9
(o)	0	1 h	3	3401.2	4	90.5
(p)	0	1 h	0	1 h	5	32.9
(q)	0	1 h	0	1 h	5	32.9
(r)	0	1 h	0	1 h	5	440.2
(s)	5	0.2	5	2.3	5	2.2
(t)	5	0.5	5	0.8	5	0.5
(u)	5	0.6	5	3.1	5	0.5
(v)	5	17.5	5	25.9	5	7.1
(w)	5	58.1	5	14.7	5	1.8
(x)	5	162.2	5	53.0	5	1.5
(y)	5	274.3	5	749.6	1	2.2
(z)	5	575.0	5	498.2	5	4.4
(α)	5	1335.2	5	45.6	5	3.4
(β)	3	2121.2	2	3023.8	2	4.0
(γ)	4	3476.6	4	2352.2	5	28.0
(δ)	0	1 h	4	1964.6	5	15.1
(ζ)	2	1 h	0	1 h	3	7.4
(η)	0	1 h	0	1 h	5	6.8
(θ)	0	1 h	1	1 h	5	42.0
Soma	109	-	119	-	151	-
Média	-	1689.4	-	1440.4	-	36.9

Capítulo 5

Considerações Finais

Neste trabalho foram abordados dois problemas de roteamento de veículos de coleta e entrega com restrições de carregamento. O primeiro problema tratado foi o Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas (*Double Vehicle Routing Problem with Multiple Stacks* - DVRPMS) e posteriormente foi proposta uma generalização do mesmo, nomeada por Problema de Roteamento Duplo de Veículos com Múltiplas Pilhas e Demanda Heterogênea (*Double Vehicle Routing Problem with Multiple Stacks and Heterogeneous Demand* - DVRPMSHD). Em ambos os problemas, uma frota de veículos deve atender um conjunto de clientes de modo que a distância percorrida pelos veículos seja a mínima possível. Para atender esses clientes, os veículos devem coletar os produtos em uma região de coleta, transportá-los e entregá-los em uma região de entrega. À medida que os produtos são coletados eles são armazenados em pilhas dos compartimentos de carga dos veículos. O carregamento e descarregamento dos produtos devem respeitar a política LIFO das pilhas. No DVRPMS, a cada cliente está associado somente um produto, já no DVRPMSHD, que trata um caso mais realístico do DVRPMS, a cada cliente podem estar associados vários produtos e esse número de produtos pode ser independente dos demais clientes, sendo que todos os produtos de um determinado cliente devem ser transportados por um mesmo veículo.

Com relação ao DVRPMS, foi desenvolvida uma formulação de programação linear inteira (*Integer Linear Programming* - ILP) e também quatro algoritmos heurísticos. Embora a formulação ILP não tenha se mostrado eficiente, com

os resultados obtidos, foi possível constatar inconsistências nos resultados apresentados no primeiro trabalho da literatura que trata o problema. Os quatro algoritmos heurísticos foram desenvolvidos com base nas metaheurísticas *Simulated Annealing* (SA)[1], *Variable Neighborhood Search* (VNS) e *Iterated Local Search* (ILS).

A representação das soluções e as quatro estruturas de vizinhança definidas para explorar o espaço de soluções do DVRPMS foram desenvolvidas de forma que toda solução explorada durante a execução dos algoritmos heurísticos fosse uma solução viável. Para isso, as soluções foram representadas apenas pelo plano de carregamento de cada veículo, e todas as soluções vizinhas obtidas através das estruturas de vizinhança consistem em alterações apenas nos planos de carregamento.

Uma vez que a representação das soluções informa apenas o plano de carregamento, a rota de coleta e a rota de entrega de cada veículo são determinadas pelas funções de avaliação das soluções. Para esse fim, foram desenvolvidas três estratégias para avaliar as soluções do DVRPMS, a primeira (formulação matématica) e segunda (algoritmo B&B) estratégia consistiram em determinar a melhor rota de coleta e de entrega possível a partir de um plano de carregamento, já a terceira estratégia buscava encontrar boas rotas através de um método heurístico. Devido a alta complexidade de utilizar durante todo o processo dos algoritmos as estratégias que determinam as melhores rotas, as estratégias exatas e heurísticas foram utilizadas em diferentes momentos dos algoritmos, buscando-se equilibrar a qualidade das soluções e o tempo de execução.

Todos os algoritmos heurísticos propostos para o DVRPMS foram calibrados e executados com as instâncias definidas na literatura. Os resultados obtidos superaram todos os métodos heurísticos da literatura, encontrando soluções com a mesma ou maior qualidade para todas as instâncias de teste definidas para o problema. Comparando com os métodos exatos da literatura, para o conjunto de instâncias que se conhece as soluções ótimas, os algoritmos heurísticos propostos neste trabalho foram capazes de encontrar a solução ótima em 98.4% das instâncias. Para as instâncias que os métodos exatos não conseguiram resolvê-las na otimalidade, os algoritmos heurísticos encontraram soluções de melhor qualidade para a maioria das instâncias de teste com baixo tempo computacional.

[1]Dois algoritmos heurísticos foram baseados na metaheurística SA.

Com respeito ao DVRPMSHD, foi desenvolvida uma formulação ILP, um algoritmo exato *branch-and-price* (B&P) e também um algoritmo heurístico baseado na metaheurística SA. Dentro do tempo limite de execução estabelecido, a formulação ILP foi capaz de encontrar a solução ótima para 61% das instâncias, já o algoritmo B&P resolveu com otimalidade 70% das instâncias. O algoritmo B&P se mostrou mais eficiente que a formulação ILP com relação ao número de soluções ótimas encontradas, porém em 12% das instâncias o algoritmo B&P não encontrou nenhuma solução para o problema.

No algoritmo heurístico SA proposto para o DVRPMSHD, a representação das soluções e a estrutura de vizinhança definida foram projetadas com o mesmo intuito estabelecido para o DVRPMS, de que toda solução explorada fosse uma solução viável.

O fato de que os clientes podem ter mais de um produto dificulta o processo de exploração dos planos de carregamento viáveis, uma vez que os produtos de um mesmo cliente devem estar "agrupados" em um mesmo compartimento de carga. Em outras palavras, os produtos de um mesmo cliente devem estar dispostos em algum contêiner de forma que não haja nenhum outro produto de uma outra requisição de transporte que impeça o atendimento. Para contornar essa dificuldade, as soluções dos DVRPMSHD foram representadas utilizando apenas a informação de quais requisições serão atendidas por cada veículo, e a partir dessa informação o plano de carregamento é construído por uma simples heurística de carregamento.

Após a heurística de carregamento construir os planos de carregamento dos veículos, as rotas de cada veículo são determinadas pelas funções de avaliação propostas para avaliar as soluções do DVRPMS[2]. A decisão de qual estratégia utilizar depende das dimensões dos contêineres e da etapa do algoritmo SA. Essas decisões foram tomadas para equilibrar a qualidade das soluções e o tempo de execução.

Os resultados obtidos pelo algoritmo heurístico SA demonstraram a eficiência do método heurístico, o qual foi capaz de encontrar a solução ótima para 87.5% das instâncias que foram resolvidas na otimalidade pelo método exatos ILP e B&P.

[2]A função de avaliação heurística aplicada às soluções do DVRPMSHD considera apenas a primeira fase (fase gulosa) da heurística utilizada para avaliar as soluções do DVRPMS.

Para as instâncias que não se conhece as soluções ótimas, o algoritmo SA obteve soluções de melhor qualidade para a maioria dessas instâncias com baixo tempo computacional.

5.1 Publicações

Parte dos resultados alcançados neste trabalho para ambos os problemas já foram publicados. A seguir essas publicações são detalhadas:

1. **Título**: Simulated Annealing Metaheuristic for the Double Vehicle Routing Problem with Multiple Stacks.

   Autores: Jonatas B. C. Chagas, Ulisses E. F. Silveira, Marcelo P. L. Benedito e André G. Santos.

   Publicado em: 19th International Conference on Intelligent Transportation Systems (ITSC 2016).

   Conteúdo: Algoritmo heurístico baseado na metaheurística *Simulated Annealing* (algoritmo SA_1 descrito na Seção 3.4.6); comparação entre os resultados obtidos pelo algoritmo SA_1 e demais algoritmos da literatura.

2. **Título**: A Branch-and-Price Algorithm for the Double Vehicle Routing Problem with Multiple Stacks and Heterogeneous Demand.

   Autores: Jonatas B. C. Chagas e André G. Santos.

   Publicado em: 16th International Conference on Intelligent Systems Design and Applications (ISDA 2016).

   Conteúdo: Definição e formulação matemática do problema; algoritmo exato *branch-and-price* (descrito na Seção 4.3); resultados obtidos pela formulação matemática e pelo algoritmo *branch-and-price*.

Referências Bibliográficas

Aggelakakis, A.; Bernardino, J.; Boile, M.; Christidis, P.; Condeco, A.; Krail, M.; Papanikolaou, A.; Reichenbach, M.; Schippl, J. et al. (2015). The future of the transport industry. Relatório técnico, Institute for Prospective and Technological Studies, Joint Research Centre.

Alba Martínez, M. A.; Cordeau, J.-F.; Dell'Amico, M. & Iori, M. (2013). A branch-and-cut algorithm for the double traveling salesman problem with multiple stacks. *INFORMS Journal on Computing*, 25(1):41--55.

Applegate, D. L.; Bixby, R. E.; Chvatal, V. & Cook, W. J. (2011). *The Traveling Salesman Problem: A Computational Study: A Computational Study*. Princeton University Press.

Augustine, J. E. (2002). Offline and online variants of the traveling salesman problem. Dissertação de mestrado, Louisiana State University, Baton Rouge, Louisiana, United States.

Berbeglia, G.; Cordeau, J.-F.; Gribkovskaia, I. & Laporte, G. (2007). Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1):1--31.

Berbeglia, G.; Cordeau, J.-F. & Laporte, G. (2010). Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1):8--15.

Carrabs, F.; Cerulli, R. & Cordeau, J.-F. (2007a). An additive branch-and-bound algorithm for the pickup and delivery traveling salesman problem with lifo or fifo loading. *INFOR*, 45(4):223.

Carrabs, F.; Cerulli, R. & Speranza, M. G. (2010). A branch-and-bound algorithm for the double tsp with two stacks. Relatório técnico, Dipartimento di Matematica e Informatica, Università di Salerno, Fisciano, Italy.

Carrabs, F.; Cordeau, J.-F. & Laporte, G. (2007b). Variable neighborhood search for the pickup and delivery traveling salesman problem with lifo loading. *INFORMS Journal on Computing*, 19(4):618--632.

Casazza, M.; Ceselli, A. & Nunkesser, M. (2012). Efficient algorithms for the double traveling salesman problem with multiple stacks. *Computers & Operations Research*, 39(5):1044--1053.

Cattrysse, D. G. & Van Wassenhove, L. N. (1992). A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research*, 60(3):260--272.

Černỳ, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41--51.

Chagas, J. B. C. & Santos, A. G. (2016). A branch-and-price algorithm for the double vehicle routing problem with multiple stacks and heterogeneous demand. Em *16th International Conference on Intelligent Systems Design and Applications (ISDA)*, pp. 921--934, Porto, Portugal. Springer.

Chagas, J. B. C.; Silveira, U. E. F.; Benedito, M. P. L. & Santos, A. G. (2016). Simulated annealing metaheuristic for the double vehicle routing problem with multiple stacks. Em *19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1311--1316, Rio de Janeiro, Brasil. IEEE.

Chiang, W.-C. & Russell, R. A. (1996). Simulated annealing metaheuristics for the vehicle routing problem with time windows. *Annals of Operations Research*, 63(1):3--27.

Cordeau, J.-F.; Iori, M.; Laporte, G. & Salazar González, J. J. (2010). A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with lifo loading. *Networks*, 55(1):46--59.

Côté, J.-F.; Archetti, C.; Speranza, M. G.; Gendreau, M. & Potvin, J.-Y. (2012). A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with multiple stacks. *Networks*, 60(4):212--226.

Crainic, T. G. & Kim, K. H. (2007). Intermodal transportation. *Handbooks in Operations Research and Management Science*, 14:467--537.

Czech, Z. J. & Czarnas, P. (2002). Parallel simulated annealing for the vehicle routing problem with time windows. Em *10th Euromicro Workshop on Parallel, Distributed and Network-based*, pp. 376--383. IEEE.

Dantzig, G. B. & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1):80--91.

Desaulniers, G.; Desrosiers, J. & Solomon, M. M. (2006). *Column generation*, volume 5. Springer Science & Business Media.

Desrochers, M.; Desrosiers, J. & Solomon, M. (1992). A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40(2):342--354.

Desrochers, M.; Lenstra, J. K. & Savelsbergh, M. W. (1990). A classification scheme for vehicle routing and scheduling problems. *European Journal of Operational Research*, 46(3):322--332.

Dumas, Y.; Desrosiers, J. & Soumis, F. (1991). The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54(1):7--22.

Dumitrescu, I.; Ropke, S.; Cordeau, J.-F. & Laporte, G. (2010). The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm. *Mathematical Programming*, 121(2):269.

Felipe, Á.; Ortuño, M. T. & Tirado, G. (2009). The double traveling salesman problem with multiple stacks: A variable neighborhood search approach. *Computers & Operations Research*, 36(11):2983--2993.

Feo, T. A. & Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6(2):109--133.

Fleszar, K.; Osman, I. H. & Hindi, K. S. (2009). A variable neighbourhood search algorithm for the open vehicle routing problem. *European Journal of Operational Research*, 195(3):803--809.

Fuellerer, G.; Doerner, K. F.; Hartl, R. F. & Iori, M. (2010). Metaheuristics for vehicle routing problems with three-dimensional loading constraints. *European Journal of Operational Research*, 201(3):751--759.

Fukasawa, R.; Longo, H.; Lysgaard, J.; de Aragão, M. P.; Reis, M.; Uchoa, E. & Werneck, R. F. (2006). Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming*, 106(3):491--511.

Gendreau, M.; Hertz, A. & Laporte, G. (1996). The traveling salesman problem with backhauls. *Computers & Operations Research*, 23(5):501--508.

Gendreau, M.; Iori, M.; Laporte, G. & Martello, S. (2008). A tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. *Networks*, 51(1):4--18.

Glover, F. (1989). Tabu search - part i. *ORSA Journal on Computing*, 1(3):190--206.

Glover, F. (1990). Tabu search - part ii. *ORSA Journal on Computing*, 2(1):4--32.

Goetschalckx, M. & Jacobs-Blecha, C. (1989). The vehicle routing problem with backhauls. *European Journal of Operational Research*, 42(1):39--51.

Grabara, J.; Kolcun, M. & Kot, S. (2014). The role of information systems in transport logistics. *International Journal of Education and Research*, 2(2):1--6.

Gutin, G. & Punnen, A. P. (2002). *The traveling salesman problem and its variations*, volume 12. Springer Science & Business Media.

Hernández-Pérez, H. & Salazar-González, J.-J. (2004). A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, 145(1):126--139.

Hertz, A. & Widmer, M. (2003). Guidelines for the use of meta-heuristics in combinatorial optimization. *European Journal of Operational Research*, 151(2):247--252.

Iori, M. & Martello, S. (2010). Routing problems with loading constraints. *Top*, 18(1):4--27.

Iori, M. & Riera-Ledesma, J. (2015). Exact algorithms for the double vehicle routing problem with multiple stacks. *Computers & Operations Research*, 63:83--101.

Iori, M.; Salazar-González, J.-J. & Vigo, D. (2007). An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, 41(2):253--264.

Kalantari, B.; Hill, A. V. & Arora, S. R. (1985). An algorithm for the traveling salesman problem with pickup and delivery customers. *European Journal of Operational Research*, 22(3):377--386.

Kirkpatrick, S.; Gelatt, C. D.; Vecchi, M. P. et al. (1983). Optimization by simmulated annealing. *Science*, 220(4598):671--680.

Kytöjoki, J.; Nuortio, T.; Bräysy, O. & Gendreau, M. (2007). An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research*, 34(9):2743--2757.

Ladany, S. P. & Mehrez, A. (1984). Optimal routing of a single vehicle with loading and unloading constraints. *Transportation Planning and Technology*, 8(4):301--306.

Laporte, G.; Gendreau, M.; Potvin, J.-Y. & Semet, F. (2000). Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research*, 7(4-5):285--300.

Laporte, G.; Mercure, H. & Nobert, Y. (1986). An exact algorithm for the asymmetrical capacitated vehicle routing problem. *Networks*, 16(1):33--46.

Lenstra, J. K. & Kan, A. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221--227.

Lourenço, H. R.; Martin, O. C. & Stützle, T. (2003). Iterated local search. Em *Handbook of Metaheuristics*, pp. 320--353. Springer.

Lusby, R. M.; Larsen, J.; Ehrgott, M. & Ryan, D. (2010). An exact method for the double tsp with multiple stacks. *International Transactions in Operational Research*, 17(5):637--652.

Lysgaard, J.; Letchford, A. N. & Eglese, R. W. (2004). A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2):423--445.

Martello, S. & Vigo, D. (1998). Exact solution of the two-dimensional finite bin packing problem. *Management Science*, 44(3):388--399.

Miller, C. E.; Tucker, A. W. & Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326--329.

Mladenović, N. & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097--1100.

Moon, C.; Kim, J.; Choi, G. & Seo, Y. (2002). An efficient genetic algorithm for the traveling salesman problem with precedence constraints. *European Journal of Operational Research*, 140(3):606--617.

Mosheiov, G. (1994). The travelling salesman problem with pick-up and delivery. *European Journal of Operational Research*, 79(2):299--310.

Nagy, G. & Salhi, S. (2005). Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, 162(1):126--141.

Osman, I. H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, 41(4):421--451.

Pacheco, J. A. (1997). Heurístico para los problemas de rutas con carga y descarga en sistemas lifo. *SORT, Stat Oper Res Trans*, 21:153–175.

Parragh, S. N.; Doerner, K. F. & Hartl, R. F. (2008a). A survey on pickup and delivery problems - part i: Transportation between customers and depot. *Journal für Betriebswirtschaft*, 58(1):21--51.

Parragh, S. N.; Doerner, K. F. & Hartl, R. F. (2008b). A survey on pickup and delivery problems - part ii: Transportation between pickup and delivery. *Journal für Betriebswirtschaft*, 58(1):81--117.

Penna, P. H. V.; Subramanian, A. & Ochi, L. S. (2013). An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics*, 19(2):201--232.

Petersen, H. L.; Archetti, C. & Speranza, M. G. (2010). Exact solutions to the double travelling salesman problem with multiple stacks. *Networks*, 56(4):229--243.

Petersen, H. L. & Madsen, O. B. (2009). The double travelling salesman problem with multiple stacks–formulation and heuristic solution approaches. *European Journal of Operational Research*, 198(1):139--147.

Polacek, M.; Hartl, R. F.; Doerner, K. & Reimann, M. (2004). A variable neighborhood search for the multi depot vehicle routing problem with time windows. *Journal of Heuristics*, 10(6):613--627.

Renaud, J.; Boctor, F. F. & Laporte, G. (2002). Perturbation heuristics for the pickup and delivery traveling salesman problem. *Computers & Operations Research*, 29(9):1129--1141.

Renaud, J.; Boctor, F. F. & Ouenniche, J. (2000). A heuristic for the pickup and delivery traveling salesman problem. *Computers & Operations Research*, 27(9):905--916.

Rondinelli, D. & Berry, M. (2000). Multimodal transportation, logistics, and the environment: managing interactions in a global economy. *European Management Journal*, 18(4):398--410.

Ruiz, R. & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research*, 177(3):2033--2049.

Sampaio, A. H. & Urrutia, S. (2016). New formulation and branch-and-cut algorithm for the pickup and delivery traveling salesman problem with multiple stacks. *International Transactions in Operational Research*, 24:77--98.

Savelsbergh, M. (1997). A branch-and-price algorithm for the generalized assignment problem. *Operations Research*, 45(6):831--841.

Savelsbergh, M. W. & Sol, M. (1995). The general pickup and delivery problem. *Transportation Science*, 29(1):17--29.

Schneider, M.; Stenger, A. & Hof, J. (2015). An adaptive vns algorithm for vehicle routing problems with intermediate stops. *OR Spectrum*, 37(2):353--387.

Scott, A. J. & Knott, M. (1974). A cluster analysis method for grouping means in the analysis of variance. *Biometrics*, 30(3):507--512.

Silveira, U. E. F.; Benedito, M. P. L. & Santos, A. G. (2015). Heuristic approaches to double vehicle routing problem with multiple stacks. Em *15th International Conference on Intelligent Systems Design and Applications (ISDA)*, pp. 231--236, Marrakesh, Marocco. IEEE.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254--265.

SteadieSeifi, M.; Dellaert, N. P.; Nuijten, W.; Van Woensel, T. & Raoufi, R. (2014). Multimodal freight transportation planning: A literature review. *European Journal of Operational Research*, 233(1):1--15.

Stützle, T. (2006). Iterated local search for the quadratic assignment problem. *European Journal of Operational Research*, 174(3):1519--1539.

Toth, P. & Vigo, D. (1999). A heuristic algorithm for the symmetric and asymmetric vehicle routing problems with backhauls. *European Journal of Operational Research*, 113(3):528--543.

Toth, P. & Vigo, D. (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1):487--512.

Van Breedam, A. (1995). Improvement heuristics for the vehicle routing problem based on simulated annealing. *European Journal of Operational Research*, 86(3):480--490.

Vansteenwegen, P.; Souffriau, W.; Berghe, G. V. & Van Oudheusden, D. (2009). Iterated local search for the team orienteering problem with time windows. *Computers & Operations Research*, 36(12):3281--3290.

Vincent, F. Y.; Lin, S.-W.; Lee, W. & Ting, C.-J. (2010). A simulated annealing heuristic for the capacitated location routing problem. *Computers & Industrial Engineering*, 58(2):288--299.