JORGE VON ATZINGEN DOS REIS

META-HEURÍSTICAS BASEADAS EM BUSCA EM VIZINHANÇA VARIÁVEL APLICADAS A PROBLEMAS DE OPERAÇÃO DE TRANSPORTES

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Doutor em Engenharia de Transportes.

São Paulo
2013

JORGE VON ATZINGEN DOS REIS

META-HEURÍSTICAS BASEADAS EM BUSCA EM VIZINHANÇA VARIÁVEL APLICADAS A PROBLEMAS DE OPERAÇÃO DE TRANSPORTES

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Doutor em Engenharia de Transportes.

Área de Concentração:
Engenharia de Transportes.

Orientador:
Prof. Dr. Claudio Barbieri da Cunha.

São Paulo
2013

Este exemplar foi revisado e alterado em relação à versão original sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 28 de novembro de 2013.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Reis, Jorge Von Atzingen dos
Meta-heurísticas baseadas em busca em vizinhança variável aplicadas a problemas de operação de transportes / J.V.A. Reis. -- ed. Rev. -- São Paulo, 2013.
207 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Transportes.

1.Meta-heuristica 2.Busca em vizinhança variável 3.Problema de Bin-packing com frota heterogênea 4.Problema de programação integrada da tabela de horários, veículos e tripulações
I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Transportes II.t.

Dedico este trabalho a Deus, à minha família, principalmente aos meus pais Vitor e Yara, às minhas irmãs Marina e Laís, à minha esposa Déborah e à minha filha Juliana.

AGRADECIMENTOS

Agradeço à Escola Politécnica da Universidade de São Paulo, ao Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq – Brasil, à Universidade Federal de Uberlândia pelo apoio para a realização da pesquisa que resultou no presente trabalho, ao meu orientador Prof. Dr. Claudio Barbieri da Cunha pela orientação realizada durante o desenvolvimento deste trabalho e aos professores Dr. Gustavo Peixoto Silva e Dr. Nicolau Dionísio Faraes Gualda pelas suas contribuições ao trabalho, bem como a todos que, direta ou indiretamente, colaboraram na execução do mesmo.

RESUMO

Esta pesquisa trata da aplicação de meta-heurísticas baseadas em busca em vizinhança variável em problemas de operação de transportes. Desta forma, buscou-se encontrar problemas complexos durante a operação de sistemas de transportes, nas grandes cidades, que possam ser resolvidos com a aplicação de meta-heurística baseada em busca em vizinhança variável.

Este trabalho aborda dois diferentes problemas de planejamento e operação de transportes. O primeiro problema abordado neste trabalho é o Problema de Programação da Tabela de Horários, de Veículos e de Tripulantes de Ônibus, no qual as viagens que comporão a tabela de horários, os veículos que executarão as viagens e as tripulações que operarão os veículos são alocadas simultaneamente e de maneira integrada.

O segundo problema a ser abordado é o problema de distribuição física, o qual envolve o agrupamento e a alocação de entregas a uma frota de veículos visando minimizar o frete total. Uma abordagem para a modelagem matemática deste problema é modelar como um problema de *bin-packing,* com *bins* de tamanho variável unidimensional (do inglês *Variable Sized Bin-Packing Problem* - VSBPP), ou seja, uma generalização do tradicional problema de *bin-packing* no qual *bins* (veículos) de diferentes capacidades e custos estão disponíveis para a alocação de um conjunto de objetos (cargas), de modo que o custo total dos *bins* (veículos) utilizados seja mínimo.

A outra abordagem proposta para o problema de distribuição física é modelar o problema como um problema de *bin-packing,* com *bins* de tamanho variável bidimensional (do inglês *Bidimensional Variable Sized Bin-Packing Problem* – BiD-VSBPP). Assim sendo, trata-se de uma expansão do problema de *bin-packing* com *bins* de tamanho variável unidimensional (VSBPP), no qual *bins* (veículos) de diferentes capacidades (capacidade volumétrica e capacidade de carga) e custos estão disponíveis para a alocação de um conjunto de objetos (cargas), os quais possuem as dimensões peso e volume, de modo que o custo total dos *bins* (veículos) utilizados seja mínimo.

Durante a realização deste trabalho, foi desenvolvido um programa computacional em C++, o qual implementa a meta-heurística Busca em Vizinhança Variável (VNS) e duas meta-heurísticas baseadas em VNS. São apresentados resultados de experimentos computacionais com dados reais e dados *benchmarking*. Os resultados obtidos comprovam a eficácia das meta-heurísticas propostas.

ABSTRACT

This work approaches variable neighborhood search meta-heuristic applicate on transport operation problems. This way, we sought find complex transport operation problems in large cities that can be solved with the variable neighborhood search meta-heuristic application.
This work approaches two different transport planning and operation problems. The first problem approached in this paper is the Bus Timetable Vehicle Crew Scheduling Problem, in which timetabling, bus and crew schedules are simultaneously determined in an integrated approach.
The second problem to be approached is the physical distribution problem which comprises grouping and assigning deliveries to a heterogeneous fleet of vehicles aiming to minimize the total freight cost. The problem can be mathematical modeled as one-dimensional Variable Sized Bin-Packing Problem (VSBPP), a generalization of the traditional bin-packing problem, in which bins (vehicles) with different sizes and costs are available for the assignment of the objects (deliveries) such that the total cost of the used bins (vehicles) is minimized.
Another proposed approach to the problem of physical distribution is model as two dimensional Variable Sized Bin-Packing Problem (BiD-VSBPP). Therefore, it is an expansion of the bin-packing problem with bins variable-length-dimensional (VSBPP), in which bins (vehicle) of different capacity (capacity and load carrying capacity) and costs are available for allocation a set of objects (loads), which have the dimensions weight and volume, so that minimized the total cost of bins (vehicle).
In this work, was developed a C++ software implemented, which was implemented a meta-heuristic Variable Neighborhood Search (VNS) and two others meta-heuristics based on VNS. Computational results for real-world problems and benchmarking problems are presented, showing the effectiveness of these proposed meta-heuristics.

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

BiD-VSBPP	*Bidimensional Variable Size Bin-Packing*
BPP	*Bin-Packing Problem*
CSP	*Crew Scheduling Problem*
EEC	Enumeração Explícita de Colunas
FA	Função de Avaliação
FO	Função Objetivo
GC	Geração de Colunas
GRASP	*Greedy Randomized Adaptive Search Procedure*
ILS	*Iterated Local Search*
MS	Melhor Solução neste trabalho
ND	Não Disponível
OSO	Ordem de Serviço de Operação
OT	Oportunidade de Troca
PPH	Problema de Programação da Tabela de Horários
PPHVT	Problema de Programação Integrada da Tabela de Horários, Veículos e Tripulações
PPT	Problema de Programação de Tripulações
PPV	Problema de Programação de Veículos
PPVT	Problema de Programação Integrada de Veículos e Tripulações
SPP	*Set Partitioning Problem*
STC	Sistema de Transporte Coletivo por Ônibus Urbano
TS	*Tabu Search*
TTP	*Time Tabling Problem*
TTVCSP	*Time Tabling Vehicle Crew Scheduling Problem*
VCSP	*Vehicle Crew Scheduling Problem*
VND	*Variable Neighborhood Descended*
VNS	*Variable Neighborhood Search*
VNS-LR	*Variable Neighborhood Search* com Lista Restrita
VSBPP	*Variable Size Bin-Packing*
VSP	*Vehicle Scheduling Problem*

1. INTRODUÇÃO

Esta pesquisa trata da aplicação de meta-heurísticas baseadas em busca em vizinhança variável em problemas de operação de transportes. Desta forma, foram selecionados problemas complexos que ocorrem na prática relacionados ao planejamento e operação de transportes e que possam ser resolvidos com a aplicação de alguma meta-heurística baseada em busca em vizinhança variável. Nesta pesquisa são estudadas três meta-heurísticas: a meta-heurística de busca em vizinhança variável e duas outras meta-heurísticas baseadas em busca em vizinhança variável (busca em vizinhança variável com lista restrita e busca decomposta em vizinhança variável).

Parte desta pesquisa de doutorado corresponde à continuidade da pesquisa de mestrado e dos projetos de iniciação científica desenvolvidos pelo autor durante a sua graduação. Em Reis (2006) foi proposta uma metodologia para resolver o PPVT (Problema Integrado de Programação de Veículos e Tripulações), na qual a programação dos veículos é realizada considerando características das tripulações.

No trabalho de Reis (2008) foi desenvolvido um modelo computacional que possui uma estrutura de vizinhança com capacidade de exploração das características das interações entre o PPV (Problema de Programação de Veículos) e o PPT (Problema de Programação da Tripulação), ou seja, resolver a programação dos veículos simultaneamente à programação das tripulações.

1.1. RELEVÂNCIA DO TEMA

Um dos problemas estudados nesta pesquisa, que envolve a programação de veículos e tripulantes no transporte coletivo por ônibus urbano, é muito recorrente no contexto brasileiro.

Segundo os dados do IBGE (2010), existiam 190.775.799 pessoas no Brasil no ano de 2010, das quais 84,36% viviam nas cidades. Pesquisas feitas pela CNI-Ibope (2011) revelam que o transporte coletivo é utilizado por 61% dos brasileiros, mas que apenas 42% utilizam como principal meio de locomoção, e dentre esses tipos de transportes coletivos, o

mais utilizado é o ônibus; cerca de 34% da população o utiliza como principal forma de se locomover.

Uma pesquisa feita pela CNI-Ibope (2011) revela que as pessoas que usufruem o transporte público brasileiro perdem muito tempo no trajeto, gerando algum tipo de atraso em seus compromissos. Segundo os dados em cidades com mais de 100 mil habitantes, aproximadamente 32% dos usuários do transporte coletivo por ônibus urbano dispendem mais de uma hora para se locomover de casa para o trabalho ou de casa para a escola (CNI-Ibope, 2011).

A pesquisa feita pela CNI-Ibope (2011) revela ainda que quase metade da população que não utiliza o transporte público, não o utiliza devido à falta de disponibilidade do mesmo ou por incompatibilidade de horários, porém esse problema é mais comum nas cidades pequenas (consideradas assim cidades com menos de 100 mil habitantes), ou nas cidades do interior, ao passo que nas capitais, o problema principal é o alto custo, o elevado tempo de viagem ou a falta de conforto.

Na maioria das cidades brasileiras, o ônibus é o principal, senão o único meio de transporte público de passageiros, atendendo aqueles que não possuem carro, mas também contribuindo para reduzir os congestionamentos e, dessa forma, melhorando a qualidade de vida. A fim de incentivar a sua utilização em muitas cidades, diversas medidas devem ser tomadas, incluindo um esforço para reduzir custos e, em decorrência, as tarifas pagas pelos usuários, definindo uma tarifa justa que privilegie a população de baixa renda, na qual estão inseridos uma grande parcela dos seus usuários. Nesse contexto, a programação eficiente de veículos e tripulações são essenciais para essa redução de custos, uma vez que representa uma parcela significativa dos mesmos.

No caso particular do transporte coletivo urbano por ônibus no Brasil, as instâncias reais presentes neste trabalho possuem até 26.779 viagens a serem realizadas em um único dia. Desta forma, a operação deste sistema de transporte coletivo por ônibus urbano torna-se um problema complexo e a utilização de ferramentas de pesquisa operacional são necessárias para obter soluções eficazes e eficientes para o PPHVT (Problema de Programação Integrada da Tabela de Horários, Veículos e Tripulações).

Desde 2012, os caminhões estão proibidos de trafegar nas principais vias da cidade de São Paulo nos horários considerados de pico (entre 05h00 e 09h00 e entre 17h00 e 22h00) pela nova legislação (Lei 14.751/08 regulamentada pelas portarias da secretaria

municipal de transportes 123-12, 124-12, 125-12 e pelo Decreto 53.149/12). Esta mudança na legislação resulta em um aumento do número de caminhões, devido a serem veículos menores do tipo VUC (veículo urbano de carga), que necessitam circular pela cidade de São Paulo.

A resolução do Problema de *Bin-packing* com *Bins* Heterogêneos Unidimensional (VSBPP) ou Bidimensional (BiD-VSBPP), pode contribuir para a redução do número de veículos necessários para o transporte urbano de cargas.

Este problema surge em diferentes situações práticas, nas quais os pontos a serem atendidos estão suficientemente próximos entre si, de tal modo que as distâncias percorridas entre paradas consecutivas podem ser consideradas irrelevantes para o custo total da frota e, portanto, podem ser desconsideradas para o cálculo dos fretes pagos a terceiros que realizam esse serviço.

Problemas reais de transporte de cargas, frequentemente encontrados nas cidades brasileiras, envolvem um número elevado de entregas a serem realizadas em um único dia e, aliando isto à sua natureza combinatória, estes problemas são considerados como complexos e são candidatos a serem abordados por ferramentas de pesquisa operacional.

Este trabalho contribui no sentido de apresentar técnicas de otimização, acumular conhecimentos e desenvolver ferramentas que permitam reduzir os custos com o planejamento e a operação de sistemas de transporte. Mais especificamente, estão sendo propostas três meta-heurísticas (a meta-heurística VNS e duas meta-heurísticas baseadas em Busca em Vizinhança Variável) para resolver o PPHVT (Problema de Programação Integrada da Tabela de Horários, Veículos e Tripulações), o VSBPP (Problema de *Bin-packing* com *Bins* Heterogêneos) e o BiD-VSBPP (Problema Bidimensional de *Bin-packing* com *Bins* Heterogêneos), as quais são testadas e avaliadas considerando dados reais e/ou dados de instâncias *benchmarking* da literatura.

1.2. OBJETIVO

O objetivo deste trabalho é resolver dois problemas complexos de operação de transporte através da utilização de meta-heurísticas baseadas em busca em vizinhança variável. Estas meta-heurísticas baseadas em VNS serão propostas visando explorar as possibilidades de resolução de importantes problemas de transportes presentes no dia a dia das

grandes cidades brasileiras e em suas regiões metropolitanas. A meta-heurística VNS foi escolhida devido ao seu potencial de adaptação a problemas distintos, o baixo número de parâmetros a serem ajustados quando comparado à outras meta-heurísticas e a experiência do autor do presente trabalho com a utilização da meta-heurística VNS em trabalhos anteriores.

O primeiro problema abordado é o Problema de Programação Integrada da Tabela de Horários, Veículos e Tripulações (PPHVT) de um sistema de transporte coletivo por ônibus urbano. O PPHVT consiste em resolver simultaneamente a tabela de horários, a programação dos veículos (PPV) e das tripulações (PPT), isto é, determinar os horários de início das viagens a serem realizadas, atribuir as viagens a cada um dos veículos e as tarefas (conjunto de viagens) a cada um dos tripulantes, de tal modo que o custo operacional total, englobando as parcelas referentes aos veículos e às tripulações, seja mínimo e a demanda de passageiros seja atendida.

A revisão da literatura encontrou trabalhos que abordam a integração entre a programação dos veículos e a programação das tripulações, também foram encontrados trabalhos que modificam a tabela de horários das viagens para minimizar o número de veículos utilizados. Entretanto, a revisão da literatura não encontrou nenhum trabalho abordando a integração entre a tabela de horários, veículos e tripulações.

O segundo problema abordado é o problema de *bin-packing* com *bins* heterogêneos (VSBPP). O problema de *bin-packing* (BPP) pode ser descrito da seguinte forma: dados $j = 1,\ldots, n$ objetos ou itens com seus respectivos pesos w_j e $i = 1,\ldots, m$ *bins* idênticos de capacidade finita c determinar a alocação (ou designação) dos n itens aos m *bins*, de tal modo que o número de *bins* utilizados seja mínimo, respeitando-se as restrições de capacidade em cada um dos *bins*.

O VSBPP é o caso mais geral do BPP, no qual a frota de distribuição pode ser composta por veículos de diferentes tamanhos e capacidades (em geral, de dois a cinco tipos), aos quais estão associados custos diferentes. Nesse caso, busca-se não mais minimizar o número total de veículos, mas sim o custo total dos veículos utilizados.

Normalmente o frete dos veículos maiores, embora maior, costuma ser menor, em termos unitários (por unidade de capacidade), em comparação ao dos veículos menores, uma vez que o custo diário do veículo não é diretamente proporcional à sua capacidade de carga. Por exemplo, o consumo unitário de combustível não varia diretamente com a capacidade de

carga; tampouco o salário do motorista costuma variar diretamente com o tamanho do veículo.

O problema bidimensional de *bin-packing* com *bins* heterogêneos (BiD-VSBPP) é uma variação do VSBPP, na qual é considerado que os objetos possuem duas dimensões (peso e volume), aumentando a complexidade do problema abordado.

1.3. Artigos Publicados

O presente trabalho resultou, até o presente momento, na publicação dos seguintes artigos e de outros artigos que ainda não foram publicados:

- REIS, J. A.; CUNHA, C. B. Uma Heurística Baseada em Busca em Vizinhança Variável para o Problema de Agrupamento de Entregas em Veículos de uma Frota Heterogênea. In: Confederação Nacional do Transporte; Associação Nacional de Pesquisa e Ensino em Transportes. (Org.). Transporte em Transformação XIV: trabalhos vencedores do Prêmio CNT Produção Acadêmica 2009. 1ed.Brasília: Positiva, 2009, v. 1, p. 81-99.
- REIS, J. A.; CUNHA, C. B. Uma Heurística Baseada em Busca em Vizinhança Variável para o Problema de Agrupamento de Entregas em Veículos de uma Frota Heterogênea. In: XXIII ANPET - Congresso de Pesquisa e Ensino em Transportes, 2009, Vitória. Anais do XXIII ANPET - Congresso de Pesquisa e Ensino em Transportes. Rio de Janeiro: ANPET, 2009. p. 1-12.
- REIS, J. A.; CUNHA, C. B. Uma heurística baseada em VNS para o problema bidimensional de *bin-packing* com frota heterogênea. In: XVI Congresso Pan-americano de Engenharia de Tráfego e Transportes e Logística, 2010, Lisboa. Anais do XVI Congresso Pan-americano de Engenharia de Tráfego e Transportes e Logística. Lisboa: XVI PANAM, 2010. v. 1. p. 1-19.
- REIS, J. A.; CUNHA, C. B. Uma nova Heurística Baseada na Meta-heurística VNS para o Problema de Agrupamento de Entregas em Veículos de uma Frota Heterogênea. Transportes (Rio de Janeiro), v. 1, p. 1-18, 2011.

1.4. Estrutura do Trabalho

Este trabalho possui a seguinte organização:

O presente capítulo corresponde a uma introdução ao tema, relatando suas origens e importância, assim como objetivos do trabalho e suas limitações. O segundo capítulo apresenta uma revisão sobre meta-heurísticas utilizadas para otimização, com maior detalhamento para a meta-heurística de Busca em Vizinhança Variável e as meta-heurísticas baseadas em Busca em Vizinhança Variável.

No terceiro capítulo são apresentados o Problema de Programação de Horários, Veículos e Tripulações de Ônibus Urbano (PPHVT) e os problemas correlatos: programação de veículos, programação de tripulações, bem como a descrição do problema e uma breve introdução ao sistema de transporte coletivo por ônibus urbano (STC). No quarto capítulo são detalhados os métodos exatos e as meta-heurísticas utilizadas para abordar o PPV, o PPT, o PPHVT e os experimentos computacionais realizados.

O quinto capítulo aborda o Problema de Agrupamento de Entregas em Veículos com Frota Heterogênea (VSBPP), apresentando a metodologia utilizada e a aplicação da meta-heurística proposta para o VSBPP, juntamente com os experimentos computacionais.

No sexto capítulo é detalhado o Problema Bidimensional de Agrupamento de Entregas em Veículos com Frota Heterogênea (BiD-VSBPP), o qual é um caso geral do VSBPP com duas dimensões. Neste capítulo também são apresentados o estado da arte do BiD-VSBPP, as meta-heurísticas implementadas e os experimentos computacionais. No sétimo capítulo são destacadas as conclusões e considerações finais.

2. META-HEURÍSTICAS PARA OTIMIZAÇÃO

Tendo em vista que tanto o PPHVT quanto o VSBPP e o BiD-VSBPP são problemas de natureza combinatória, cujo esforço computacional é não polinomial, ou seja, cresce exponencialmente com o tamanho do problema, torna-se necessária uma abordagem heurística para a resolução destes problemas, uma vez que os algoritmos exatos, apesar de todos os avanços observados, são incapazes de obter soluções ótimas em um tempo de processamento viável do ponto de vista prático (Garey & Johnson, 1979; Kwan & Rahin, 1997; Wren, 1998).

Estratégias de solução heurística apoiam-se, em geral, em alguma abordagem intuitiva, na qual a estrutura particular do problema pode ser considerada e explorada de forma a maximizar a chance de se obter uma solução satisfatória, considerando-se a relação entre a qualidade da solução e o esforço computacional necessário para obtê-la (Glover & Kochenberg, 2003).

Segundo Cunha (2006), o termo *meta-heurística*, introduzido pela primeira vez por Glover (1986), é a união de duas palavras gregas: *heurística*, derivada do verbo *heuriskein* ($\varepsilon\upsilon\rho\iota\sigma\kappa\varepsilon\iota\upsilon$), que significa "*encontrar*", e o prefixo *meta*, que significa "*além, acima*" (no sentido de superior). Assim, as meta-heurísticas podem ser definidas como heurísticas estruturadas, ou seja, possuem alguma estratégia de manipulação das heurísticas da busca local visando a melhorar a exploração do espaço de vizinhança, escapando de ótimos locais.

As meta-heurísticas, de forma geral, são técnicas utilizadas para resolver problemas de maximização ou minimização, sujeitos a um determinado conjunto de restrições. Da mesma forma que as heurísticas, as meta-heurísticas são apoiadas em alguma abordagem intuitiva, utilizando combinações de escolhas aleatórias e conhecimento histórico, dos resultados anteriores, para guiarem as buscas pelo espaço de soluções (viáveis ou inviáveis), objetivando determinar as melhores soluções vizinhas à solução corrente e escapar de ótimos locais. As meta-heurísticas controlam as heurísticas de busca local e são apoiadas em um padrão de comportamento seguindo uma estrutura lógica.

A solução corrente é utilizada para realizar o procedimento definido na meta-heurística. Uma solução vizinha é gerada a partir da solução corrente, utilizando um

movimento da estrutura de vizinhança. A estrutura de vizinhança contém os movimentos (as alterações) possíveis de serem realizadas na solução corrente para gerar uma solução vizinha. A vizinhança é o conjunto de soluções geradas a partir da solução corrente, utilizando um movimento definido pela estrutura de vizinhança.

Entre as meta-heurísticas mais conhecidas e amplamente utilizadas com sucesso, podem-se citar: algoritmo genético, busca tabu, *simulated annealing*, GRASP ("*Greedy Randomized Adaptive Search Procedure*"), ILS (*Iterated Local Search*), VND (*Variable Neighborhood Descended*), busca em vizinhança variável (do inglês "*variable neighborhood search"* – VNS), colônia de formigas, entre outras.

Neste capítulo são apresentados as meta-heurísticas que influenciaram o desenvolvimento do presente trabalho. A junção das meta-heurísticas ILS (*Iterated Local Search*) e VND (*Variable Neighborhood Descend*) resultou no desenvolvimento da meta-heurística VNS (*Variable Neighborhood Search*) e a junção das meta-heurísticas TS (*Tabu Search* ou Busca Tabu) e VNS resultou no desenvolvimento da meta-heurística VNS-LR (VNS com Lista Restrita). O método de Geração de Colunas foi utilizado para obtenção de soluções ótimas para instâncias de pequeno porte do PPHVT e para comparar com as soluções obtidas através das meta-heurísticas desenvolvidas.

2.1. META-HEURÍSTICA *TABU SEARCH*

O TS (Busca Tabu, do inglês *Tabu Search*) foi proposto inicialmente por Glover (1986). A proposta inicial da Busca Tabu é incluir uma estrutura de memória que permita o método de busca local "*escapar*" de soluções ótimas locais.

Segundo Glover (1986), a Busca Tabu inclui elementos de memória de curto prazo, que evitam a reversão de movimentos recentes, e uma memória de prazo mais longo, para reforçar a atração por soluções mais distantes. O princípio da Busca Tabu é realizar buscas locais sempre visando a encontrar um ótimo local, permitindo movimentos que não sejam de melhoria, para evitar retornar em soluções previamente avaliadas (Glover, 1986).

Os movimentos previamente realizados são armazenados através da utilização de memórias, chamadas listas tabu, que formam um registro dos movimentos realizados recentemente. Estes movimentos ficam armazenados na lista tabu segundo um certo critério

de tempo (por exemplo número de iterações) e, enquanto um movimento estiver registrado na lista tabu, este movimento é considerado tabu, não podendo ser realizado, e impedindo a ciclagem da busca local.

```
Procedimento Busca Tabu
    1.s* ← s;
    2.Iter ← 0;
    3.MelhorIter ← 0;
    4.Esvazie a lista tabu;
    5.Inicialize a função de aspiração;
    6.enquanto (FO(s) > FO_min e Iter − MelhorIter ≤ max_Iter) faça
    7.          Iter ← Iter + 1;
    8.          Encontre o melhor elemento da vizinhança cujo movimento
    não esteja na lista tabu ou atenda o critério de aspiração;
    9.          Atualize a lista tabu;
    10.         s ← s';
    11.         se (FO(s) < FO(s*)) então
    12.                 s* ← s;
    13.                 MelhorIter ← Iter;
    14.         fim-se;
    15.         Atualize a função de aspiração;
    16.fim-enquanto;
    17.s ← s*;
    18.Retorne s;
Fim
```

Figura 2.1 – Pseudocódigo da meta-heurística TS para Problema de Minimização (Glover & Kochenberg, 2003).

A função de aspiração é utilizada para considerar uma solução *s* que melhore a melhor solução encontrada até o momento, mesmo se o movimento realizado para obter a solução *s* seja um movimento proibido (movimento tabu). Por exemplo, suponha que realizar um determinado movimento tabu resulte em uma solução *s* que possua um custo inferior a qualquer outra solução já visitada. Neste caso, o movimento tabu que resulta na solução *s* é realizado através da utilização da função de aspiração.

2.2. META-HEURÍSTICA *ITERATED LOCAL SEARCH*

O ILS (Busca através de Iterações Locais, do inglês *Iterated Local Search*) é baseado na ideia de que um procedimento de busca local pode ser melhorado através de sucessivas perturbações na solução local, gerando novas soluções iniciais para o método de busca (Glover & Kochenberg, 2003).

Segundo Glover & Kochenberg (2003), uma meta-heurística ILS é baseada em quatro componentes (procedimentos) principais:

- Procedimento gerador da solução inicial: gera uma solução inicial s_0 para o problema;
- Procedimento de busca local: realiza uma busca na vizinhança da solução corrente até encontrar uma solução possivelmente melhorada s'';
- Procedimento perturbatório: modifica a solução corrente s para uma solução intermediária s';
- Procedimento do critério de aceitação: decide em qual solução a próxima perturbação será aplicada.

Procedimento *Iterated Local Search*
1. $s_0 \leftarrow$ gere uma solução inicial( );
2. $s \leftarrow$ busca local(s_0);
3. repita
4. $s' \leftarrow$ perturbação(s, histórico);
5. $s'' \leftarrow$ busca local(s');
6. $s \leftarrow$ critério de aceitação(s, s'', histórico);
7. até critério de parada ser atendido;

Fim

Figura 2.2 – Pseudocódigo da meta-heurística ILS para Problema de Minimização (Glover & Kochenberg, 2003).

O desempenho do ILS está diretamente relacionado com o conjunto das soluções locais, com a escolha do método de busca local, das perturbações e dos critérios de aceitação. A qualidade da solução obtida pelo ILS e a velocidade da convergência para uma solução

depende fortemente do método de busca local utilizado (Katayama & Narihisa, 1999; Johnson & McGeoch, 1997). Um método de descida poderia ser utilizado, mas também é possível aplicar métodos mais sofisticados como, por exemplo, a Busca Tabu.

A intensidade da perturbação deve ser calibrada para possuir força suficiente para escapar de soluções locais, mas fraca o suficiente para manter as melhores características da solução local corrente. O critério de aceitação deve ser definido para decidir a partir de qual solução será realizada a busca local e qual a perturbação que será aplicada. O critério de aceitação e perturbação devem seguir dois princípios básicos: o princípio da diversificação e o princípio da intensificação.

Segundo Glover & Kochenberg (2003), o princípio da diversificação deve ser aplicado através de grandes perturbações na solução ótima local, visando explorar soluções de vizinhanças distantes da solução corrente. O princípio da intensificação é aplicado através de leves perturbações na solução corrente, visando intensificar a busca local no entorno da melhor solução encontrada. Um critério de aceitação facilmente aplicável é mover-se para a solução local s'' somente se s'' for melhor que a solução local corrente s.

2.3. META-HEURÍSTICA *VARIABLE NEIGHBORHOOD SEARCH*

Segundo Glover & Kochenberg (2003), o VNS (*Variable Neighborhood Search*) é uma evolução do VND (*Variable Neighborhood Descent*) e possui similaridades com o ILS (*Iterated Local Search*). O VND, proposto em Mladenovic & Hansen (1997), é um método de busca local que explora o espaço de soluções por meio de trocas de estruturas de vizinhança, aceitando somente soluções de melhora da solução corrente e retornando à primeira estrutura quando uma solução melhor é encontrada. Caso não seja encontrada uma solução de melhora, a busca continuará na estrutura de vizinhança seguinte. Caso não exista uma estrutura de vizinhança seguinte, a execução é interrompida e a solução corrente é retornada como a solução ótima local.

A partir de uma solução corrente, o VNS realiza uma busca local na sua vizinhança, similar ao VND. No caso de não encontrar uma solução de melhora, escolhe-se, aleatoriamente, uma vizinhança cada vez mais distante da solução corrente; caso se encontre uma solução melhor, inicia-se uma nova busca local, similar ao VND (Mladenovic & Hansen,

1997).

Procedimento *Variable Neighborhood Descent*

1. Seja s_0 uma solução inicial e r o número de estruturas de vizinhança;
2. $s \leftarrow s_0$; {Solução corrente}
3. $k \leftarrow 1$; {Tipo de estrutura de vizinhança}
4. enquanto ($k \leq r$) faça
5. Encontre o melhor vizinho $s' \in N^{(k)}(s)$;
6. Se f(s') < f(s)
7. então $s \leftarrow s'$;
8. $k \leftarrow 1$;
9. senão $k \leftarrow k + 1$;
10. Fim-se;
11. Fim-enquanto;
12. Retorne s;

Fim

Figura 2.3 – Pseudocódigo da meta-heurística VND para Problema de Minimização.

O VNS é um método de busca local, o qual consiste em explorar o espaço de soluções, por meio de trocas sistemáticas de estruturas de vizinhança. Contrariamente a outras meta-heurísticas, baseadas em métodos de busca local, o método VNS não segue uma trajetória, mas sim explora vizinhanças gradativamente mais "*distantes*" da solução corrente e concentra a busca em torno de uma nova solução se, e somente se, um movimento de melhora é realizado (Glover & Kochenberg, 2003).

O método inclui, também, um procedimento de busca local a ser aplicado sobre a solução corrente. Esta rotina de busca local também pode usar diferentes estruturas de vizinhança. O pseudocódigo da meta-heurística VNS é apresentado na Figura 2.4. Detalhes adicionais desta meta-heurística podem ser obtidos em Mladenovic & Hansen (1997).

No VNS, parte-se de uma solução inicial qualquer e, a cada iteração, seleciona-se aleatoriamente um vizinho s' dentro da vizinhança $N^k(s)$, obtida por meio da estrutura de vizinhança de ordem k ($1 \leqslant k \leqslant r$), a partir da solução s corrente. Esse vizinho s' é então submetido a um procedimento de busca local.

Se a nova solução obtida s'', resultante dessa busca local no entorno de s', for

melhor que a solução corrente s (isto é $f(s'') < f(s)$), a busca prossegue a partir da nova solução encontrada s'', reiniciando a partir da primeira estrutura de vizinhança $N^1(s)$, que corresponde a $k = 1$. Caso contrário, a busca prossegue a partir da próxima estrutura de vizinhança $N^{k+1}(s)$.

```
Procedimento Variable Neighborhood Search
    1. Seja s0 uma solução inicial e r o número de estruturas de vizinhança;
    2. s ← s0;                                          {Solução corrente}
    3. Enquanto (Critério de parada não satisfeito) faça
    4.     k ← 1;                                {Tipo de estrutura de vizinhança}
    5.     Enquanto (k ≤ r) faça
    6.         Gere um vizinho qualquer s' ∈ N(k)(s);
    7.         s'' ← Busca_Local(s');
    8.         Se f(s'') < f(s)
    9.             Então s ← s'';
    10.                k ← 1;
    11.            Senão k ← k + 1;
    12.        Fim-se;
    13.    Fim-enquanto;
    14. Fim-enquanto;
    15. Retorne s;
Fim
```

Figura 2.4 – Pseudocódigo da meta-heurística VNS para Problema de Minimização.

Este procedimento encerra-se quando uma condição de parada for atingida (por exemplo, tempo de processamento, número máximo de iterações, ou número máximo de iterações consecutivas sem melhorias). A solução s' é obtida de maneira aleatória, no passo seis do pseudocódigo da Figura 2.4, a fim de evitar ciclos, situação que pode ocorrer caso alguma regra determinística seja utilizada (Mladenovic & Hansen, 1997).

Na Figura 2.5, pode-se observar como a meta-heurística VNS explora o espaço de soluções, em um problema de minimização. A partir de uma solução corrente s, é realizado um movimento de exploração da vizinhança e uma solução vizinha s' é selecionada dentro do raio de alcance da estrutura de vizinhança $N^1(s)$. Este movimento de exploração consiste em fazer uma modificação na solução corrente, utilizando um critério de escolha aleatório, para possibilitar que a meta-heurística VNS escape das melhores soluções locais.

A partir desta solução vizinha s' é realizada uma busca local na vizinhança da

solução s'. Caso a melhor solução vizinha de s' não resulte em melhoria, retorna-se à solução s. Uma solução s' será considerada de melhoria se $f(s') < f(s)$ (em um problema de minimização). Após retornar à solução corrente s, é realizado um novo movimento de exploração da vizinhança, escolhendo-se uma solução vizinha s'' pertencente ao raio de alcance da estrutura de vizinhança $N^2(s)$. Este movimento de exploração consiste em fazer uma modificação na solução corrente, utilizando um critério de escolha pré-definido, para possibilitar que a meta-heurística VNS escape das melhores soluções locais.

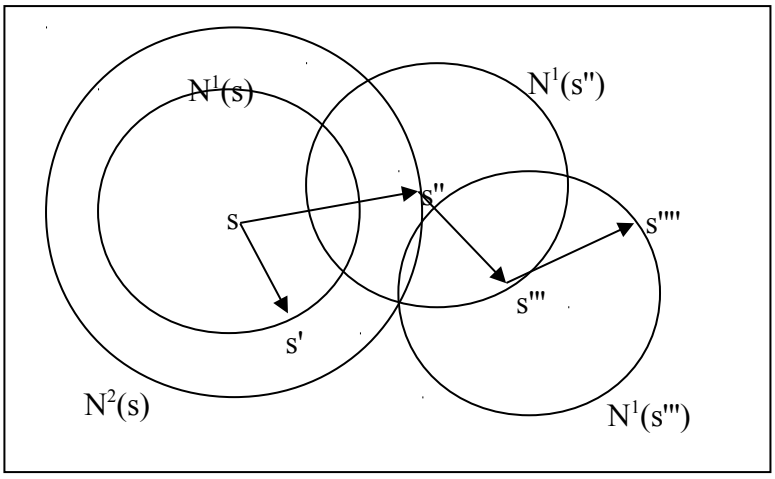

Figura 2.5 – Funcionamento do VNS (Reis, 2008).

Neste momento, é realizada uma busca local na vizinhança da solução s''. Caso a melhor solução vizinha de s'' seja aceita como solução de melhora, a partir dessa solução é escolhida uma solução s''', dentro do raio de alcance da estrutura $N^1(s'')$, após a realização de um movimento de exploração da vizinhança. Se a solução s''' for aceita, uma nova solução s'''', pertencente à vizinhança de s''', é escolhida.

Sempre que uma solução de melhora for encontrada pela meta-heurística VNS, a solução corrente é movida para a nova solução e a busca é reiniciada, considerando a primeira estrutura de vizinhança (N^1). E na meta-heurística VNS, a busca local sempre é realizada na vizinhança de uma solução vizinha (escolhida aleatoriamente) da solução corrente.

As estruturas de vizinhança, as heurísticas construtivas e as demais adaptações necessárias da meta-heurística VNS para abordar cada um dos problemas (PPHVT, VSBPP e BiD-VSBPP) são apresentadas, respectivamente, nas seções 4.2, 5.2 e 6.2.

2.4. META-HEURÍSTICA *VARIABLE NEIGHBORHOOD SEARCH* COM LISTA RESTRITA

O VNS-LR (Método de Busca em Vizinhança Variável com Lista Restrita) é uma junção da meta-heurística VNS, apresentada na seção 2.3, com a meta-heurística Busca Tabu, apresentada na seção 2.1. O VNS-LR basicamente é uma meta-heurística VNS que possui uma lista restrita similar à lista tabu presente na meta-heurística Busca Tabu. O princípio de funcionamento do VNS-LR é manter uma lista com as soluções vizinhas anteriormente visitadas e impedir que o VNS-LR visite estas soluções novamente, exceto quando atenda aos critérios de aspiração (Villa *et al.*, 2006; Turajlic & Dragovic, 2012).

Procedimento VNS-LR
1. Seja s_0 uma solução inicial e r o número de estruturas de vizinhança;
2. $s \leftarrow s_0$; {Solução corrente}
3. Enquanto (Critério de parada não satisfeito) faça
4. $k \leftarrow 1$; {Tipo de estrutura de vizinhança}
5. Enquanto ($k \leq r$) faça
6. Gere um vizinho qualquer $s' \in N^{(k)}(s)$;
7. Repita
8. $s'' \leftarrow$ Busca_Local(s');
9. Até $s'' \notin$ lista restrita() ou atenda aos critérios de aspiração;
10. Se f(s'') < f(s)
11. Então
12. adicione s'' na lista restrita();
13. $s \leftarrow s''$;
14. $k \leftarrow 1$;
15. Senão $k \leftarrow k + 1$;
16. Fim-se;
17. Fim-enquanto;
18. Fim-enquanto;
19. Retorne s;

Fim

Figura 2.6 – Pseudocódigo do VNS com Lista Restrita.

Neste trabalho, o critério de aspiração utilizado foi o número máximo de iterações sem melhora, ou se esta solução vizinha s' melhore a melhor solução encontrada (s*) até o momento ($f(s') < f(s^*)$, para um problema de minimização).

As estruturas de vizinhança, as heurísticas construtivas e as demais adaptações necessárias da meta-heurística VNS-LR para abordar cada um dos problemas (PPHVT,

VSBPP e BiD-VSBPP) são apresentadas, respectivamente, nas seções 4.2, 5.2 e 6.2.

2.5. META-HEURÍSTICA *VARIABLE NEIGHBORHOOD DECOMPOSITION SEARCH*

Hansen *et al.* (2001) propuseram a meta-heurística VNDS (Método de Busca Decomposta em Vizinhança Variável) como uma variação da meta-heurística VNS proposta por Mladenovic & Hansen (1997). A principal mudança em relação ao VNS é a busca local, que é decomposta em duas etapas.

Procedimento *Variable Neighborhood Decomposition Search*
1. Seja s_0 uma solução inicial e r o número de estruturas de vizinhança;
2. $s \leftarrow s_0$; {Solução corrente}
3. Enquanto (Critério de parada não satisfeito) faça
4. $k \leftarrow 1$; {Tipo de estrutura de vizinhança}
5. Enquanto ($k \leq r$) faça
6. Gere um vizinho qualquer $s' \in N^{(k)}(s)$;
7. $s'' \leftarrow$ Busca_Local_Decomposta(s');
8. Se f(s'') < f(s)
9. Então $s \leftarrow s''$;
10. $k \leftarrow 1$;
11. Senão $k \leftarrow k + 1$;
12. Fim-se;
13. Fim-enquanto;
14. Fim-enquanto;
15. Retorne s;

Fim VNDS

Figura 2.7 – Pseudocódigo do VNDS.

O pseudocódigo que descreve genericamente o funcionamento da meta-heurística VNDS é apresentado na Figura 2.7. Após ser gerada aleatoriamente uma solução s' dentro da estrutura de vizinhança k, a primeira etapa da busca local é uma busca realizada na vizinhança de s', utilizando a estrutura de vizinhança k.

A segunda etapa da busca local é uma busca realizada dentro da vizinhança de s' além da estrutura de vizinhança k que está sendo analisada. Por exemplo, considere que a estrutura de vizinhança k é a realocação de uma tarefa do PPT (Problema de Programação da Tripulação). A primeira etapa envolve encontrar outra tripulação para receber esta tarefa

que possua um custo menor do que a solução s'. A segunda etapa é encontrar outra tripulação, para trocar uma tarefa com esta primeira tripulação que possua um custo menor do que a solução s'.

As estruturas de vizinhança, as heurísticas construtivas e as demais adaptações necessárias da meta-heurística VNDS para abordar cada um dos problemas (PPHVT, VSBPP e BiD-VSBPP) são apresentadas, respectivamente, nas seções 4.2, 5.2 e 6.2.

2.6. GERAÇÃO DE COLUNAS

A geração de colunas é empregada em problemas lineares com instâncias de grande porte, além de ser utilizada juntamente com a decomposição de *Dantzig-Wolfe*. (Lorena *et al.*, 2003).

O princípio do método de geração de colunas é decompor o problema original, utilizando a enumeração implícita (Baldo, 2009). Segundo Santos (2008), a divisão desse problema original é:

- Problema mestre (PM): semelhante ao problema original, entretanto o número de colunas explícitas é menor;
- Subproblema (SP): interage com o problema mestre, gerando colunas novas para que seja incorporada no problema mestre e melhore o resultado atual;

O problema mestre utiliza uma pequena quantidade de colunas; assim, é possível encontrar a solução rapidamente. A adição das colunas é feita gradativamente com as colunas geradas no subproblema conduzindo a solução otimizada do problema mestre (Santos, 2008).

Uma abordagem bastante explorada é formular o PPT (problema de programação da tripulação) como um problema de recobrimento ou de particionamento (*set covering* ou *set partitioning model*) e utilizar a técnica de geração de colunas para resolvê-lo (Smith & Wren, 1988; Desrochers & Soumis, 1989; Desrochers *et al*., 1992; Fores *et al*., 1999; Barnhart *et al*., 1998; Friberg & Haase, 1999). A variedade de trabalhos deriva das diferentes maneiras de gerar as colunas e diferentes metodologias para resolver o problema, tais como: *branch-and-bound*, *branch-and-price* e relaxação lagrangeana.

Desrochers & Soumis (1989) propõem uma abordagem de geração de colunas para o PPT que decompõe o problema em duas partes: um problema principal, que é o problema de recobrimento, e um subproblema de caminho mínimo. O problema de recobrimento escolhe uma programação dentre as jornadas factíveis previamente conhecidas. O subproblema é usado para encontrar uma nova jornada factível que melhore a solução corrente do problema de recobrimento. Nesse subproblema, cada caminho factível existente na rede que liga a origem até o destino representa uma jornada viável.

No modelo de recobrimento são consideradas as restrições globais do problema, tais como o número máximo e mínimo de pegadas corridas e o número máximo de tripulações. As jornadas de trabalho são divididas em dois tipos: pegada corrida e dupla pegada. Pegada corrida é uma jornada de trabalho com o tempo contínuo, isto é, com intervalos entre as viagens inferiores a duas horas e, a jornada de trabalho do tipo dupla pegada é caracterizada pela existência de um intervalo igual ou superior a duas horas durante o qual a tripulação é dispensada do serviço. Esse problema é resolvido pelo método Simplex e fornece uma relaxação linear do problema de recobrimento. Uma vez resolvido o problema principal, as novas colunas são geradas pelo algoritmo de caminho mínimo com restrições adicionais.

Desrochers & Soumis (1989) constituem a base do método HASTUS (Rousseau & Blais, 1985), que consiste no sistema comercial para a programação de tripulações desenvolvido no centro de pesquisa GERARD da Universidade de Montreal.

Um outro sistema comercial de referência internacional é o TRACS II (Wren *et al*., 1994) desenvolvido pelo Grupo de Programação de Restrições e Pesquisa Operacional da Universidade de Leeds, no Reino Unido. O sistema TRACS II é uma versão atual do IMPACS e vem sendo aprimorado e adaptado para resolver também o problema de programação de tripulações de trens (Wren *et al*., 1994; Fores *et al*., 1999).

O sistema IMPACS tem como base o trabalho de Smith & Wren (1988), que emprega o modelo de recobrimento associado a um método de busca *branch-and-bound* para a obtenção de uma solução inteira para o PPT. Nesse caso, para diminuir o número de colunas, são consideradas as características do problema, gerando apenas as colunas que se referem às jornadas de trabalho similares àquelas praticadas pelas empresas operadoras do sistema. Dessa forma, o espaço de busca do Simplex é reduzido consideravelmente, permitindo a resolução do problema em um tempo razoável de processamento computacional.

O método de geração de colunas utilizado neste trabalho será detalhado na seção 4.1.

3. O PROBLEMA DE PROGRAMAÇÃO DE HORÁRIOS, VEÍCULOS E TRIPULAÇÕES

Este capítulo e o próximo abordam o problema de programação integrada da tabela de horários, veículos e tripulações de um sistema de transporte coletivo por ônibus urbano (PPHVT), iniciando com uma breve introdução ao Sistema de Transporte Coletivo por Ônibus Urbano, de forma a enfatizar a importância do mesmo; em seguida, é apresentada uma caracterização detalhada do problema em termos das restrições consideradas, e é feita uma revisão da literatura sobre o PPHVT e os problemas correlatos (problema de programação de veículos – PPV; problema de programação de tripulações – PPT; e o problema de programação integrada de veículos e tripulações PPVT).

A estratégia de solução utilizada para o PPHVT e os problemas correlatos (PPV, PPT e PPVT), incluindo o detalhamento das meta-heurísticas utilizadas e os experimentos computacionais são apresentados no próximo capítulo.

Conforme mencionado anteriormente, este capítulo e o próximo correspondem à continuidade do trabalho desenvolvido em Reis (2008), ao integrar a elaboração da tabela de horários ao PPVT (Problema de Programação Integrada de Veículos e Tripulações).

3.1. Sistema de Transporte Coletivo por Ônibus Urbano

O Sistema de Transporte Coletivo (STC) é composto por todo e qualquer modo de transporte que permita o deslocamento coletivo de indivíduos, a fim de permitir a realização das atividades diárias de cada indivíduo. Como exemplos de sistemas de transporte coletivo de passageiros, pode-se citar o ferroviário, o metroviário, o aéreo, por ônibus intermunicipal, por ônibus interestadual e por ônibus urbano. O escopo de pesquisa deste trabalho é o transporte coletivo por ônibus urbano.

A estrutura do sistema de transporte coletivo por ônibus urbano compreende os tipos de serviços prestados e as linhas oferecidas, a política de cálculo e cobrança de tarifas, as integrações entre os diversos modos, os equipamentos necessários e o sistema de informações aos passageiros.

De uma forma geral, o poder público, por meio da administração direta ou de empresas estatais, assume a responsabilidade de definir as linhas e os itinerários do transporte coletivo urbano de passageiros, de modo a cobrir toda a região de abrangência do município. O poder público também determina a frequência das linhas, de forma a atender à demanda diária de passageiros. Além disso, o poder público realiza a concessão da operação das linhas para as empresas operadoras, fiscaliza e gerencia os serviços prestados e administra os mecanismos de gestão de receita, caso esses existam.

As empresas operadoras são os agentes do sistema de transporte coletivo responsáveis pela prestação dos serviços, incluindo a alocação dos veículos e das tripulações (motoristas e cobradores). Cabe às empresas operadoras a determinação do número de veículos e de tripulantes necessários para operar cada linha de ônibus e a execução das viagens diárias, conforme estabelecido pelo poder concedente.

As desigualdades econômicas e sociais, presentes em várias cidades brasileiras, se expressam na ocupação e uso do solo, segregando na periferia e nas áreas mais distantes e carentes de infraestrutura a parcela mais pobre da população. Ao se concentrar a maior parte das oportunidades de trabalho na região central das cidades, criou-se a necessidade de transportar diariamente muitas pessoas para o local de realização de suas atividades diárias, o que gera uma grande demanda pelo uso do sistema de transporte coletivo.

Segundo SPTRANS (2006), a priorização do transporte individual, nas últimas décadas, caracterizada pelos investimentos realizados no sistema viário visando a reduzir os congestionamentos, levou as grandes metrópoles a enfrentar uma crise de mobilidade. A consequência desta preferência pelo transporte individual é o crescimento da parcela das viagens motorizadas, realizadas por meio de transporte individual, a cada ano (SPTRANS, 2006).

A metrópole de São Paulo é uma das poucas do mundo na qual o principal meio de transporte é o ônibus e seus similares, como micro-ônibus e vans (SPTRANS, 2006). Segundo dados da SPTRANS (2006), 77% das viagens coletivas diárias, dentro da cidade de São Paulo, são realizadas por ônibus e seus correlatos micro-ônibus e vans, o que equivale a, aproximadamente, 6,8 milhões de viagens diárias. O transporte coletivo sobre trilhos atende a apenas 10% das viagens coletivas metropolitanas, apesar de seus 270 km de extensão, sendo 134 km dentro do município de São Paulo (SPTRANS, 2006).

O aumento do uso do automóvel teve como consequência o aumento do tráfego

urbano e um dos principais problemas de transporte: os congestionamentos, os quais desgastam os motoristas, reduzem a produtividade e acarretam enormes prejuízos para muitos setores da economia (FREITAS *et al.*, 2001).

A cidade de São Paulo é um exemplo de como os congestionamentos podem ser prejudiciais à qualidade de vida. São Paulo se caracteriza por um crescimento não planejado, o qual ocorreu sem a implantação simultânea de uma estrutura de transporte coletivo de maior capacidade, como trens urbanos e metrô. Outro grande problema de um sistema de transporte público é a definição de sua política tarifária, a qual visa a determinar o preço dos serviços prestados pelos agentes operacionais, isto é, determinar a contrapartida financeira do usuário. Esta contrapartida depende da forma como o Estado e os empregadores participam do financiamento dos serviços e da forma como certos grupos de usuários (idosos, estudantes, deficientes físicos) tenham direito a concessões de descontos e gratuidades (AGUIAR, 2001).

As políticas tarifárias não se limitam apenas à definição do valor da tarifa, mas também sobre a forma como a tarifa é cobrada e as eventuais formas de subsídios e contribuições sociais. Conforme Aguiar (2001), existem três conjuntos de fatores ou instrumentos sobre os quais as decisões e ações da política tarifária são realizadas:

- Nível da tarifa: os valores monetários pelos quais as tarifas são cobradas;
- Estrutura tarifária: os elementos espaciais e funcionais que servem de base para a cobrança da tarifa;
- Concessões especiais: os descontos e gratuidades atribuídos a alguns grupos de usuários.

O cálculo do valor da tarifa a ser cobrada do usuário considera que, caso não exista uma política de subsídios, o valor estabelecido deve cobrir os custos do sistema de transporte coletivo por ônibus. Isto é, no valor da tarifa devem estar incluídos, além do custo de transporte daquele usuário que está usufruindo o serviço, os custos das gratuidades (idosos e deficientes) e dos descontos concedidos a certas classes de usuários (por exemplo, estudantes).

No Brasil, é comum utilizar-se da política de tarifa única, na qual existe uma tarifa única cobrada, independentemente, da distância percorrida. Este tipo de política tarifária cria um grande problema devido à existência de linhas cujo percurso é longo e o usuário entra no

ponto inicial e desce somente no ponto final. Este tipo de linha é bastante encontrada no trajeto da periferia para o centro das cidades. Essas linhas longas, muitas vezes não cobrem os seus custos, visto que o seu índice de passageiros por quilômetro é muito baixo. Em um outro extremo, existem linhas cujo índice de passageiros por quilômetros é elevado, criando trajetos altamente lucrativos.

O custo operacional de um sistema de transporte coletivo por ônibus urbano pode ser reduzido por meio de uma operação mais eficiente da frota, como proposto neste trabalho, ou por meio da melhoria do sistema de transporte como um todo. Uma redução nos congestionamentos urbanos possibilitaria um aumento da velocidade média dos ônibus, reduzindo o tempo necessário para a execução das viagens e aumentaria a utilização dos veículos devido ao aumento da produtividade. Desta forma, será necessário um número menor de veículos e de tripulantes para operar a mesma quantidade de viagens diárias, reduzindo o custo operacional do sistema de transporte coletivo por ônibus urbano.

O poder público é o órgão responsável pela definição do tipo e do tamanho dos veículos a serem utilizados pelas empresas operadoras em cada linha e serviço. Entretanto, não é atribuição do poder público definir a alocação dos veículos para a realização das viagens diárias, na maioria dos casos limitando-se a indicar a frota necessária para cada linha. No entanto, é interessante para o poder público reduzir a frota de veículos necessários, como uma forma de minimizar os custos operacionais e os repasses de subsídios às empresas operadoras.

3.2. JUSTIFICATIVA E IMPORTÂNCIA DO PPHVT.

No Brasil, o transporte coletivo urbano por ônibus é o principal meio de transporte público utilizado pelas pessoas que residem em áreas urbanas, nos seus deslocamentos para realização das suas atividades diárias cotidianas, incluindo trabalho, escola, compras, lazer, etc.

Um sistema de transporte coletivo eficiente corresponde a uma importante alternativa para a melhoria da qualidade de vida nos centros urbanos, pois permite reduzir as viagens realizadas por automóvel, acarreta diminuição dos congestionamentos, redução da poluição ambiental, diminuição dos acidentes e da queima de combustíveis fósseis, os quais

contribuem para o aquecimento global.

Por outro lado, existe uma percepção mais ou menos generalizada, entre os diversos agentes, como o poder público, operadoras e usuários, de que o transporte coletivo urbano por ônibus vem enfrentando dificuldades consubstanciadas por uma perda de demanda e de produtividade, entre outros fatores. Segundo dados da NTU – Associação Nacional das Empresas de Transportes Urbanos (2008), na média nacional, os sistemas de transporte público transportaram em 2008 cerca de 35% menos passageiros do que transportavam nos anos 90. Diversos fatores contribuem para essa situação: os valores elevados das tarifas, os crescentes congestionamentos urbanos e a falta de prioridade para o transporte coletivo, que induzem as pessoas a trocarem o transporte coletivo pelo individual.

Dentre os itens que compõem o valor da tarifa, pode-se dizer que os custos da frota de veículos (englobando depreciação e remuneração de capital) e o salário das tripulações (motoristas e cobradores), em conjunto com o custo do combustível, representam uma parcela significativa do valor total da tarifa. Assim, uma alocação eficaz dos veículos às linhas e das tripulações aos veículos ajudam na obtenção de uma tarifa justa, privilegiando os usuários do transporte público, os quais, na sua maioria, são pessoas de baixa renda, que não dispõem de outra alternativa para seus deslocamentos diários.

3.3. CARACTERIZAÇÃO DO PROBLEMA

Em um sistema de transporte coletivo urbano por ônibus (STC), o planejamento operacional e a programação dos serviços, incluindo veículos e tripulantes (motoristas e cobradores), são, devido à sua grande complexidade, geralmente decompostos em quatro etapas. As decisões sobre a estrutura dos serviços, isto é, as linhas que compõem cada serviço, seus respectivos itinerários e frequências, assim como o tipo de veículo utilizado, são de atribuição do poder público. Estas decisões devem ser tomadas com base nas demandas de passageiros, nos serviços requeridos, na infraestrutura viária disponível e nas condições de tráfego e de circulação dos veículos.

Como resultado, é definida uma Ordem de Serviço de Operação (OSO) para cada linha, contendo todas as informações operacionais, incluindo itinerários de ida e volta, pontos terminais, frota a ser alocada e tabela de viagens, que indica as viagens correspondentes a

cada horário e respectivos locais de partida e de chegada. Com base nas programações das viagens a serem cumpridas diariamente, cabe à empresa operadora a programação de veículos e de tripulantes.

O problema de programação de horários (PPH) consiste em definir o horário de início de cada viagem a ser realizada com base na demanda de passageiros. Uma alteração no horário de início de uma viagem pode permitir que um mesmo veículo execute duas viagens que não seriam possíveis de serem executadas pelo mesmo veículo. Por exemplo, suponha duas viagens da mesma linha: a viagem i tem início às 06h00 e término às 06h50 e a viagem j tem início às 06h48 e término às 07h30; são necessários dois veículos para executar estas duas viagens, devido à sobreposição de horários; entretanto, se o horário de início da viagem j for atrasado em 3 minutos, é possível que um único veículo execute as viagens i e j.

Um problema é definir quanto tempo seria possível atrasar ou adiantar o horário de início de uma viagem sem alterar a percepção do usuário sobre o nível de serviço oferecido. Uma alteração nos horários de início de uma viagem modifica o *headway* (intervalo entre partidas dos ônibus da mesma linha), dependendo do horário (horário de "*pico"* ou não) pode ocorrer um excesso de passageiros (usuários do STC) no ponto de ônibus. Por exemplo, entre 07h00 e 08h00 pode haver uma demanda de 400 passageiros em uma determinada linha, mas esta demanda pode estar concentrada em um horário entre 07h00 e 07h30; caso o horário de início de uma viagem entre 07h00 e 07h30 seja alterado, ocorrerá uma redução no nível de serviço percebido pelos usuários devido ao acúmulo de passageiros no ponto de ônibus.

O problema de programação dos veículos (PPV) consiste em definir quantos e quais veículos serão utilizados nas linhas a serem operadas, bem como a alocação de cada veículo às viagens diárias programadas. Como resultado, obtém-se, para cada veículo, o chamado "*bloco*", ou seja, o conjunto de viagens designadas a cada veículo ao longo do dia, começando e terminando na garagem, sua sequência, e todos os aspectos espaciais e temporais relacionados, incluindo locais e horários de partidas e chegadas das viagens a serem realizadas, assim como os eventuais percursos ociosos nos deslocamentos entre viagens consecutivas. Cada bloco mostra também as Oportunidades de Troca (OT). Uma OT é um intervalo de tempo suficiente, em um ponto apropriado, para haver a troca das tripulações.

O problema de programação dos tripulantes (PPT) consiste em formar tarefas a partir dos blocos dos veículos e atribuir essas tarefas aos tripulantes (motoristas e cobradores),

definindo assim as jornadas de trabalho dos mesmos. Cada tarefa corresponde a um conjunto de viagens compreendidas entre duas OT's: uma no início e outra no final da tarefa. Assim, durante a realização da tarefa, não é possível que haja troca de tripulação. A programação de uma tripulação é formada por um conjunto de tarefas, denominado jornada de trabalho ou somente jornada. Uma jornada de trabalho corresponde à chamada escala de um tripulante (ou de uma dupla, motorista/cobrador).

É importante destacar a diferença entre a escala de tripulantes e o rodízio das tripulações: o primeiro termo refere-se à programação de um dado dia, que pode ser dia útil, sábado ou domingo, e o segundo termo, refere-se à programação de um mês ou de um período de dias, considerando folgas, férias e demais descansos obrigatórios, ao qual cada tripulante tem direito.

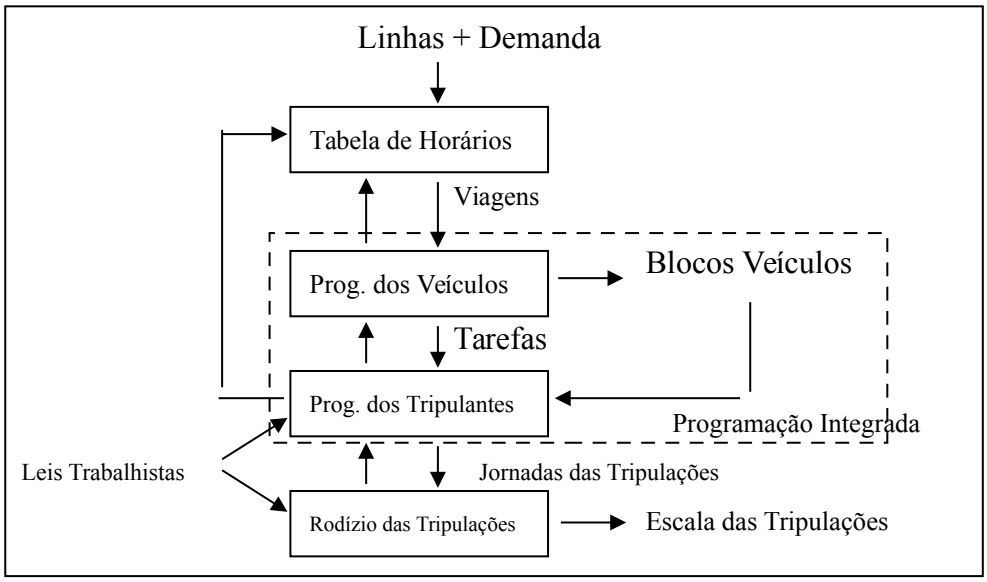

Figura 3.1 – Relação entre as operações no sistema de transporte público (Reis, 2008).

A Figura 3.1 apresenta as relações entre os problemas envolvidos na operação de um sistema de transporte público por ônibus urbano. As modificações na tabela de horários afetam os horários das viagens e consequentemente modificam a programação dos veículos. Uma alteração na programação dos veículos modifica o agrupamento de viagem para a formação das tarefas modificando a programação dos tripulantes. Caso a programação dos tripulantes seja modificada, isto acarretará um novo conjunto de jornadas de trabalho e um

novo rodízio das tripulações. Desta forma, pode-se concluir que a resolução integrada destes problemas operacionais pode aproveitar as interações existentes e reduzir os custos.

Com base na situação encontrada na empresa que forneceu os dados para este trabalho, as jornadas podem ser divididas em dois tipos: pegada simples ou dupla pegada, conforme o tempo ocioso existente entre as tarefas. No primeiro tipo, as tarefas são realizadas de uma única vez e os intervalos de tempo entre as tarefas consecutivas são inferiores a duas horas. Caso ocorra um intervalo entre uma tarefa e a seguinte igual ou superior a duas horas, a jornada é classificada como do tipo dupla pegada. Este intervalo não é contabilizado na remuneração da tripulação, que fica liberada durante este tempo. Após o término do intervalo, a tripulação deve retornar ao trabalho para realizar as tarefas remanescentes.

3.3.1. Restrições Consideradas

As restrições de definição da tabela de horários, alocação dos veículos e de escala das tripulações são tais que permitem definir o problema, por meio das regras operacionais adotadas pelas empresas e cumprimento das leis trabalhistas. Com base no trabalho de Reis (2008), na experiência prática e na interação com as empresas operadoras, foram consideradas, no processo de resolução do PPHVT, as seguintes restrições operacionais ditas essenciais, isto é, que não podem ser violadas:

Para a tabela de horários:

- Uma linha não pode ter duas ou mais viagens com o mesmo horário inicial;
- A demanda de passageiros deve ser atendida;
- O tempo entre duas viagens da mesma linha deve ser inferior a uma hora;
- O horário de início de uma viagem pode ser adiantado em até 15 minutos;
- O horário de início de uma viagem pode ser atrasado em até 15 minutos.

Para os veículos:

- Um veículo não pode realizar duas ou mais viagens ao mesmo tempo, ou seja, não pode haver sobreposição de viagens alocadas ao mesmo veículo;
- Um veículo deve iniciar e terminar a sua jornada diária de trabalho na

garagem;

- Um veículo deve permanecer na garagem por no mínimo 1 hora por dia, para manutenção;
- Para um veículo ser classificado como fazendo uma jornada denominada *dupla pegada* (jornadas compostas por dois períodos distintos de trabalho, por exemplo, manhã e tarde), deve haver um intervalo entre duas viagens superior a 2 horas, somado com o tempo de deslocamento do veículo até a garagem;
- O número de veículos com programação de tripulação em jornada de trabalho do tipo *dupla pegada* estará limitado a um determinado limite superior máximo admissível.

Para as tripulações:

- Uma tripulação não pode realizar mais de uma tarefa ao mesmo tempo, ou seja, não pode haver sobreposição de tarefas;
- As trocas das tripulações só podem ocorrer nos locais correspondentes às oportunidades de trocas, ou seja, em determinados pontos e entre viagens com intervalo de tempo suficiente para ocorrer a troca dos tripulantes;
- As trocas das tripulações somente podem ocorrer entre tripulantes, que operem grupos de linhas com as mesmas características (por exemplo, linhas da mesma região geográfica);
- Existem dois tipos de jornadas, ditas *pegada simples* e *dupla pegada*. A jornada tipo *pegada simples* tem duração de 7 horas e 10 minutos, com direito a uma pausa para descanso e/ou alimentação. A *dupla pegada* tem duração total de 6 horas e 40 minutos, dividida em duas etapas, separadas por pelo menos duas horas de interrupção. Neste caso, a tripulação não tem direito a pausa para descanso/alimentação e o intervalo de interrupção não é remunerado;
- A pausa na jornada do tipo *pegada simples* deve ser no mínimo de 20 minutos, podendo este intervalo ser fracionado, desde que pelo menos um deles seja maior ou igual a 10 minutos;
- O número de tripulações com *dupla pegada* deve estar limitado a um certo valor;
- Podem ser acrescidas até duas horas extras às jornadas diárias de trabalho.

As restrições ditas *não-essenciais*, são aquelas que correspondem a características desejáveis para a programação resultante da resolução do PPHVT, pois melhoram a eficiência do serviço. As restrições *não-essenciais* consideradas são:

Para a tabela de horários:

- O intervalo entre partidas deve ser minimizado;
- O número de viagens em cada linha deve ser minimizado;

Para os veículos:

- O tempo ocioso de um veículo deve ser minimizado;
- O número de veículos utilizados deve ser o menor possível;
- O número de vezes que um veículo realiza uma troca de linha deve ser limitado;
- O tempo total de viagem vazia, ou seja, com o veículo fora de operação, deve ser o menor possível.

Para as tripulações:

- O tempo ocioso de uma tripulação deve ser o menor possível;
- O número de horas extras deve ser minimizado;
- O número de tripulações deve ser mínimo;
- O número de vezes que uma tripulação troca de veículo deve ser limitado;
- O número de vezes que uma tripulação com *dupla pegada* finaliza a primeira pegada em um ponto e inicia a segunda pegada em um outro ponto deve ser minimizado.

As restrições *não-essenciais* não constituem exatamente restrições do PPHVT, pois definem metas ou alvos desejáveis e o não atendimento dessas não torna a solução obtida inviável do ponto de vista prático. O PPHVT pode ser formulado como um problema de múltiplos objetivos, como minimizar tempo de viagem morta, minimizar frota, minimizar o número de horas extras, minimizar o número de tripulações, entre outros. Cabe ressaltar que estes objetivos podem ser conflitantes, por exemplo, reduzir o número de tripulações, pode

implicar no aumento do número de horas extras.

3.4. Revisão da Literatura

Nesta seção é realizada uma revisão da literatura sobre o PPHVT e os problemas correlatos (PPV, PPT e PPVT), visando a levantar os trabalhos e as abordagens mais utilizadas. Esta revisão está dividida em três subseções: na primeira é realizada uma revisão sobre o PPV e a sua integração com o PPH; na subseção seguinte é abordado o PPV; e na terceira subseção é abordada a integração entre o PPV e o PPT (PPVT).

Os trabalhos científicos mais antigos resolviam o PPV e o PPT de forma separada e sequencial; os trabalhos mais recentes abordam a integração entre o PPV e o PPT. O motivo disto ocorrer é a evolução dos computadores e o fato que o custo operacional da programação das tripulações costuma superar os custos operacionais da programação dos veículos (Freling *et al*., 2001).

3.4.1. O Problema de Programação dos Veículos

Vários métodos de resolução do PPV são apresentados na literatura por diversos autores e os resultados comprovam que existe uma redução significativa no custo operacional quando métodos heurísticos ou de otimização são aplicados a casos reais (Smith & Wren, 1981; Wren & Gualda, 1999; Silva & Gualda, 2000; Silva, 2001; Silva & Gualda, 2001; Freling *et al.,* 2001).

Silva (2001) resolve o PPV utilizando uma abordagem de geração de arcos para resolver uma representação de rede do problema. O princípio da geração de arcos é derivado da abordagem de geração de colunas por programação linear. Com esta técnica é possível reduzir o número de arcos na rede que representa o problema, reduzindo o tempo de processamento necessário para resolver o problema de fluxo de rede subjacente. Também é possível introduzir algumas restrições laterais que descrevem características práticas do problema, fazendo a rede mostrar uma representação mais compatível com casos reais.

Silva & Gualda (2000) abordam problemas testes baseados em dados reais da

cidade de *Reading* no Reino Unido e da cidade de Sorocaba no Brasil, que foram resolvidos usando a abordagem de geração de arcos e os resultados foram comparados com os obtidos pelo sistema heurístico BOOST, o qual foi desenvolvido e comercializado no Reino Unido. Na maioria dos casos o método heurístico e o algoritmo de fluxo em redes encontraram a mesma solução e, em algumas instâncias de grande porte, o algoritmo de fluxo em redes encontra o ótimo global que o método heurístico não consegue alcançar. Silva & Gualda (2001) utilizam a abordagem de geração de arcos para resolver o problema de programação dos veículos usando dados reais da cidade de Belo Horizonte.

Soares *et al.* (2006) implementam uma meta-heurística VLNS na qual o PPV é resolvido, aceitando-se algumas variações (de até 3 minutos) nos horários de partida e chegada das viagens diárias, visando a otimizar a solução do PPV ao resolvê-lo de forma integrada com a programação de horários.

Silva & Gualda (2009) apresentam uma evolução do método *ArcGen*, denominada *ArcGenX*, que permite identificar viagens da tabela de horários cuja eliminação leva à redução da frota de veículos e dos custos operacionais envolvidos, realizando uma integração entre o problema de programação de horários e o problema de programação de veículos. O método *ArcGenX* foi aplicado com sucesso para resolver o problema de programação de veículos utilizando dados reais da cidade de Santos. A integração entre a programação de veículos e a programação de horários possibilitaram reduzir algumas viagens previstas na tabela de horários, reduzindo a frota necessária para a operação e respeitando a taxa de ocupação dos veículos.

3.4.2. O Problema de Programação das Tripulações

A programação de tripulações, tipicamente aplicada em sistemas de transporte público, envolve a seleção do melhor conjunto de jornadas de trabalho. O PPT está relacionado com a geração de jornadas de trabalho viáveis. A construção de jornadas de trabalho é bastante complexa, devido às diversas regras trabalhistas e operacionais que devem ser atendidas. A programação de tripulações é uma das áreas dominantes na programação e roteamento de pessoal. Muitos artigos têm sido publicados abordando diversas técnicas para a programação de tripulações em aviões, trens e ônibus (Reis, 2008).

Esse tema tem sido largamente estudado e seus resultados, geralmente, são utilizados nos países europeus. A abordagem mais explorada é aquela que formula o PPT como um problema de recobrimento ou de particionamento (*set covering* ou *set partitioning model*) e utiliza a técnica de geração de colunas para resolvê-lo (Smith & Wren, 1988; Desrochers & Soumis, 1989; Desrochers *et al*., 1992; Barnhart *et al*., 1998; Fores *et al*., 1999). A variedade de trabalhos deriva das diferentes maneiras de gerar as colunas e diferentes metodologias para resolver o problema, tais como: *branch-and-bound*, *branch-and-price* e relaxação lagrangeana.

O método proposto por Ball *et al*. (1981) consiste em uma solução iterativa que resolve o problema combinando pequenos pedaços de trabalho em pedaços maiores, e estes pedaços maiores são transformados em jornadas de trabalho. Em várias partes do *software*, o usuário possui a oportunidade de eliminar combinações inviáveis ou fazer outros ajustes manuais.

Um modelo matemático para resolver o PPT, incluindo a duração e a colocação de pontos de troca é apresentado por Ball *et al*. (1983). O problema, o qual é formulado usando uma representação de redes, é modelado como um problema de particionamento em um grafo acíclico. É usada uma decomposição do problema para a sua resolução. Um emparelhamento de um subproblema é acompanhado de uma fase de refinamento e combinação de tarefas de trabalho de forma a determinar uma jornada.

Blais *et al*. (1990) descrevem uma aplicação orientada de sofisticados métodos de programação utilizando o HASTUS, um sistema de apoio à decisão para a programação de veículos e de tripulações. O artigo trata da implementação em uma empresa de transporte coletivo de Montreal. Devido à implementação de meta-heurísticas para o desenvolvimento da escala de tripulação e de veículos, os modelos implementados utilizando o sistema HASTUS, segundo os autores, economizaram aproximadamente US$4 milhões para o operador de transporte coletivo de Montreal.

Bodin *et al*. (1983) apresentam uma revisão de modelos e métodos utilizados para resolver problemas de programação de linhas aéreas, tripulação de ônibus e roteamento de veículos. Uma classificação por categorias é apresentada segundo uma revisão de modelos, algoritmos e métodos. Ceder *et al*. (1988) descrevem um sistema computadorizado que integra o desenvolvimento da escala de horários de ônibus, escala de veículos e escala de tripulação. Depois de determinar a tabela de horários, escala de veículos e oportunidades de

trocas, o sistema determina a escala das tripulações utilizando uma heurística. A dimensão dos problemas que podem ser resolvidos é pequena; para se resolver um problema com 10 viagens, foi necessário um minuto; para um problema com 20 viagens, foi necessária uma hora; e problemas com 30 viagens não puderam ser resolvidos.

Freling *et al*. (2001) formulam o PPT como um problema de recobrimento e o resolvem combinando a técnica de relaxação lagrangeana com a de geração de colunas. Haase *et al*. (2001) usam uma aproximação exata para resolver o problema de programação da tripulação integrado com o problema de programação de veículos. O PPVT é resolvido usando o método *branch-and-bound* e o método de planos de cortes, juntamente com a geração de colunas, foi possível resolver problemas com até 300 viagens em aproximadamente 90 minutos.

Pedrosa & Constantino (2001) apresentam um modelo de recobrimento, no qual a mão de obra é dividida em grupos usando uma lista circular e o PPT é resolvido usando uma heurística construtiva com geração de colunas. Valouxis & Housos (2002) dividem a tripulação em níveis, de forma a reduzir as dimensões do PPT. Estes níveis são baseados em um algoritmo de emparelhamento das trocas de tripulação. Moura *et al*. (2000) utilizam programação inteira e geração de colunas para resolver o PPT, formulado como um problema de particionamento.

Um caso especial do problema de programação integrada da escala de veículos e tripulantes é considerado por Fischetti *et al*. (2001). O problema é modelado com programação inteira, ao invés de um problema de recobrimento. A modelagem inteira resultante tem uma relaxação linear muito livre, mas pode ser apertada por meio de novas famílias de desigualdades válidas (cortes). Estes cortes são baseados em um algoritmo *branch-and-cut*. Testes numéricos com um conjunto de problemas de testes gerados aleatoriamente e um conjunto de características reais mostram como o método é competitivo em alguns tipos de problemas testes.

Segundo Freling *et al*. (1999), os tipos de abordagens para a resolução do problema de programação de veículos e de tripulantes podem ser classificados em: independente, sequencial e integrada. A abordagem sequencial pode ser subdividida em tradicional e inversa, enquanto que a abordagem integrada pode ser subdividida em dois tipos: resolução do PPV com características dos tripulantes e resolução do PPV simultaneamente ao PPT.

Tradicionalmente, devido à complexidade dos modelos matemáticos, os problemas de planejamento operacional de um sistema de transporte público por ônibus são resolvidos de forma sequencial (Freling *et al*., 1999; Friberg & Haase, 1999). Entretanto, na maioria das situações práticas, a programação dos veículos afeta a programação das tripulações e vice-versa (Reis *et al*., 2006; Silva *et al*., 2006a; Silva *et al*., 2006b). Portanto, pode-se presumir que uma abordagem integrada do problema pode proporcionar melhores resultados que a abordagem sequencial.

A abordagem sequencial tradicional foi a primeira a ser utilizada e consiste em resolver o PPV e, em seguida, o PPT. A abordagem sequencial inversa é uma variação desta abordagem, a qual consiste em inverter a ordem de resolução dos problemas, isto é, resolver primeiramente o PPT e em seguida o PPV (Marinho *et al*., 2004; Silva *et al*., 2004; Souza *et al*., 2004; Reis, 2006).

As principais vantagens das abordagens sequenciais (tradicional e inversa) são a facilidade de implementação e a diminuição da complexidade do problema. A abordagem sequencial tradicional é recomendada para os casos em que a programação dos veículos possui um custo operacional superior ao custo operacional da programação dos tripulantes e a abordagem sequencial inversa é indicada para o caso oposto, ou seja, quando o custo operacional do PPT é superior ao do PPV. A desvantagem das abordagens sequenciais (tradicional e inversa) é o não aproveitamento das interações existentes entre o PPT e o PPV, o que torna o custo operacional destas abordagens superiores ao custo operacional das abordagens integradas.

Na abordagem independente, a programação dos veículos é resolvida ignorando-se a programação dos tripulantes; na sequência, a programação dos tripulantes é determinada independentemente dos resultados obtidos pela programação dos veículos. Cabe ressaltar que o grande problema dessa abordagem, é o fato de as soluções obtidas resultarem, na maioria dos casos, inviáveis do ponto de vista prático, pois apesar de se obter a solução ótima para o PPV e para o PPT, dificilmente, essas duas soluções são compatíveis entre si, o que impede a sua aplicação (Freling *et al.*, 1999).

3.4.3. O Problema de Programação Integrada de Veículos e Tripulações

A abordagem integrada começou a ser estudada mais recentemente, tendo em vista a evolução dos computadores, uma vez que, devido à sua grande complexidade, o custo computacional desta abordagem era proibitivo. Existem dois níveis de abordagem integrada: o primeiro, consiste em resolver simultaneamente a programação dos veículos e dos tripulantes, o qual é um problema complexo, devido às suas grandes dimensões. Existem alguns modelos computacionais para a sua resolução e os artigos científicos encontrados na literatura mostram diversos casos de aplicação deste tipo de abordagem a problemas com dados reais (Freling *et al*., 2001; Freling *et al*., 2003; Huisman *et al*., 2001; Huisman *et al*., 2003; Huisman & Wagelmans, 2004; Reis, 2008; Reis *et al*., 2006; Huisman & Wagelmans, 2006).

O segundo tipo de abordagem integrada é baseado na resolução do PPV, considerando características dos tripulantes de tal forma que a programação dos tripulantes é facilitada pela programação dos veículos. Os artigos científicos encontrados na literatura, mostram diversos casos de aplicação deste tipo de abordagem a problemas com dados reais (Freling *et al*., 2001; Freling *et al*., 2003; Huisman *et al*., 2001; Huisman *et al*., 2003; Huisman & Wagelmans, 2004; Reis, 2008; Reis *et al*., 2006).

Huisman & Wagelmans (2006) apresentam resultados de um modelo matemático exato, que considera o segundo nível de abordagem integrada para problemas com, aproximadamente, 300 viagens diárias. Entretanto, esse número é bastante reduzido, considerando-se o porte típico de uma empresa de ônibus no Brasil, que normalmente é responsável por mais de 1.000 viagens diárias (Reis, 2008). Como o PPVT corresponde a um problema de natureza combinatória, no qual o número de combinações, em termos de blocos de veículos e escalas de tripulantes, cresce exponencialmente com o tamanho do problema, torna-se muito complicado resolver instâncias reais por meio de modelos exatos de programação linear inteira utilizando os pacotes de otimização (*solvers*) comerciais (Garey & Johnson, 1979; Kwan & Rahin, 1997; Wren, 1998).

Freling *et al*. (2001) aplicam uma abordagem integrada para os problemas de programação de veículos e de programação de tripulações de ônibus urbano. Esta abordagem integrada foi originalmente proposta em Freling *et al*. (1999) e adaptada de forma a considerar restrições complexas presentes em aplicações reais. Tradicionalmente, estes problemas são

abordados sequencialmente.

Com o objetivo de avaliar a utilidade desta abordagem integrada, Freling *et al*. (2003) comparam diversas soluções obtidas com a abordagem integrada e com a abordagem sequencial tradicional. São propostos modelos matemáticos para o problema integrado e para o problema sequencial. Relaxação lagrangeana e heurísticas lagrangeanas foram desenvolvidas para o problema integrado. O modelo associado de recobrimento foi resolvido usando geração de colunas. Os resultados obtidos demonstram que caso o custo fixo com as tripulações seja muito maior que o custo fixo com os veículos, a abordagem mais adequada para o PPVT é a integrada, caso contrário é a sequencial.

Gaffi & Nonato (1999) apresentam uma heurística lagrangeana para o problema integrado. O procedimento heurístico para o subproblema de recobrimento é proposto utilizando restrições simplificadas e abordagem baseada em tarefas.

Uma modelagem matemática para o PPVT com uma única garagem e restrições de recursos no grafo de escalonamento é apresentada por Haase & Friberg (1999). A modelagem proposta é suficientemente genérica para ser estendida para múltiplos depósitos e receber restrições adicionais. O modelo é resolvido usando geração de colunas, formulando um problema de particionamento (*set partitioning*). A geração de colunas é usada para desenvolver baixos valores em um algoritmo *branch-and-bound*.

Uma abordagem exata é apresentada por Haase *et al*. (2001) para resolver o PPVT. O problema é formulado como um problema de particionamento para a escala de motoristas com restrições laterais para o itinerário dos veículos. Isto é resolvido usando o método *branch-and-bound* em conjunto com geração de colunas e o método de planos de cortes. Definida a escala de tripulantes, a resolução de um problema de fluxo em rede determina o itinerário dos veículos.

De acordo com Freling *et al*. (2003), a abordagem integrada do PPT e do PPV sempre permitirá obter resultados superiores ou iguais aos resultados da abordagem sequencial, e resultados inferiores ou iguais aos resultados da abordagem independente, tendo em vista que essa última considera cada problema de forma independente, e retorna soluções inviáveis. Entretanto, a abordagem independente é útil, pois permite determinar os limitantes inferiores para a solução ótima do PPVT.

Em Reis (2006) foi proposta uma metodologia para resolver o PPVT na qual é utilizado o primeiro tipo de abordagem integrada, isto é, considera-se na programação dos

veículos características pertinentes aos tripulantes, como tempo de folga e hora-extra. Desta forma, é obtida uma programação dos veículos capaz de facilitar a resolução da programação dos tripulantes, reduzindo o custo operacional total.

Rodrigues *et al.* (2006) propõem um algoritmo hierárquico para resolver o PPVT. O algoritmo possui quatro fases: *(i)* um gerador de escala preliminar, *(ii)* um gerador de blocos do veículo, *(iii)* um gerador de programação final, e *(iv)* uma heurística que ajusta os horários de início das viagens. Este algoritmo foi testado com dados reais das cidades de São Paulo e São Bernardo do Campo; os problemas reais utilizados possuem entre 15 e 26 veículos.

Mesquita & Paias (2008) apresentam uma formulação matemática linear inteira combinada com um modelo de fluxo em redes e um modelo de particionamento/recobrimento (*set partitioning/covering*) para o PPVT. Esta formulação matemática é resolvida utilizando um método de geração de colunas.

Em Reis (2008) foi proposta uma metodologia para resolver o PPVT, na qual é utilizado o segundo tipo de abordagem integrada, ou seja, a programação dos veículos e das tripulações são resolvidas simultaneamente, utilizando um modelo computacional, que possui uma estrutura de vizinhança com capacidade de exploração das características das interações entre o PPV e o PPT.

Outro tipo de problema que envolve a programação de veículos e tripulações é o *Airline Crew Scheduling Problem* (Problema de Programação de Tripulações Aéreas). Este problema envolve restrições muito distintas do PPVT, não sendo possível fazer analogias e aproveitar os mesmos métodos e abordagens desenvolvidos entre os dois problemas. Essa incompatibilidade entre a programação aérea e a de veículos e tripulações ônibus urbano decorre das restrições presentes no transporte aéreo, como tempo de duração da jornada, tipo de voo, autonomia, tipo de aeronave e configurações do voo (Gopalakrishnan & Johnson, 2005).

Segundo Gopalakrishnan & Johnson (2005), o problema de programação de tripulações aéreas (PPTA) pode ser dividido em cinco estágios de planejamentos: *(i)* programação dos voos, com a definição de todos os voos que serão realizados; *(ii)* alocação da frota de aviões, com a alocação das viagens aos aviões satisfazendo as restrições referentes ao tipo de aeronave; *(iii)* roteamento dos aviões, com a alocação das viagens aos aviões satisfazendo as restrições referentes à manutenção; *(iv)* emparelhamento dos tripulantes, com

a alocação dos voos aos tripulantes, satisfazendo as restrições referentes ao tipo de tripulação necessária para operar cada aeronave; e *(v)* roteamento dos tripulantes, com a definição da escala mensal ou semanal de trabalho da tripulação satisfazendo as restrições referentes a intervalos de descansos e folgas. Boas referências e conceitos sobre este problema do transporte aéreo podem ser encontradas em Arabeyre (1969), Ball & Roberts (1985), Etschmaier & Mathaisel (1985), Klabjan *et al*. (2002), Vance *et al*. (1995), Caetano & Gualda (2011) e Gomes & Gualda (2011).

3.5. CONSIDERAÇÕES

Em um levantamento prévio da literatura não foram encontrados trabalhos acerca do PPHVT. O levantamento bibliográfico realizado demonstrou que os trabalhos científicos mais antigos (décadas de 1980 e 1990) resolviam a programação de veículos e a programação das tripulações de forma separada e sequencial. Trabalhos mais recentes (final da década de 1990 e década de 2000) começaram a explorar a integração entre os problemas de programação de veículos e tripulações. Existem diversos modelos matemáticos formulados para a resolução do PPVT, entretanto o PPVT é um problema matemático complexo e geralmente são resolvidas instâncias de teste com poucas viagens diárias.

O próximo capítulo irá apresentar a estratégia de solução para o PPHVT e os problemas correlatos (PPV, PPT e PPVT), bem como os experimentos computacionais realizados.

4. ESTRATÉGIAS DE SOLUÇÃO PARA O PPHVT E EXPERIMENTOS COMPUTACIONAIS

Neste capítulo são apresentadas as estratégias de solução e os experimentos computacionais para o problema de programação integrada da tabela de horários, veículos e tripulações de um sistema de transporte coletivo por ônibus urbano (PPHVT), bem como os problemas correlatos (problema de programação de veículos, problema de programação de tripulações e o problema de programação integrada de veículos e tripulações).

Mais especificamente, na seção 4.1, são apresentados métodos exatos para a resolução sequencial da programação de veículos e da programação de tripulações (enumeração explícitas de colunas e geração de colunas). Três novas meta-heurísticas são propostas para o PPVT que podem serem estendidas para a resolução do PPHVT. Os resultados obtidos pelos métodos exatos servirão como parâmetro para aferir a qualidade das soluções obtidas pelas meta-heurísticas propostas na seção 4.2. Os experimentos computacionais realizados são apresentados na seção 4.3.

4.1. MÉTODOS EXATOS

Um grande problema na resolução da programação dos veículos ou na programação das tripulações é o elevado número de possibilidades (combinações de tarefas e de viagens possíveis). Mesmo existindo várias regras que reduzem o conjunto de soluções viáveis, o conjunto completo de viagens é bastante grande.

Nesta seção são expostos dois métodos exatos implementados e testados para a resolução do PPV e do PPT. Apesar destes métodos exatos não possuírem a capacidade de resolver instâncias grandes do PPV e do PPT, estes métodos serão utilizados como parâmetros para averiguar a qualidade das soluções obtidas pelas meta-heurísticas VNS, VNS-LR e VNDS.

4.1.1. Enumeração Explícita de Colunas

O objetivo da enumeração explícita de colunas é gerar todas as combinações possíveis para obter uma solução para o PPV e depois gerar todas as combinações possíveis para obter uma solução para o PPT compatível com a solução obtida anteriormente para o PPV. A enumeração explícita de colunas (EEC para simplificação) proposta neste trabalho não será utilizada para resolver o PPHVT. A EEC será utilizada para resolver a programação dos veículos e depois para resolver a programação das tripulações (PPV e PPT resolvidos sequencialmente). O escopo da EEC neste trabalho será servir como parâmetro para comparação das soluções obtidas pelas meta-heurísticas VNS, VNS-LR e VNDS para a abordagem sequencial tradicional (veículo primeiro, tripulação depois) e para a abordagem sequencial inversa (tripulação primeiro, veículo depois) da programação dos veículos e das tripulações de ônibus urbano.

No caso da programação dos veículos cada coluna irá representar um bloco de veículo. Um bloco de veículo é um conjunto de viagens que podem ser executadas por um único veículo. Cada linha da coluna receberá o valor 1 se o veículo alocado ao bloco puder executar a viagem correspondente à linha da coluna. Por exemplo, suponha a seguinte coluna $[1,0,0,1,0,1,1,1512]$, isto representa que o veículo que for alocado a este bloco de veículo deverá executar a primeira viagem, a quarta viagem, a sexta viagem e a sétima viagem da instância de teste e o custo deste bloco de veículo será de 1.512 unidades.

Para garantir a otimalidade da solução é necessário garantir que todas as combinações de viagens para o PPV sejam analisadas. Isto é feito através de um programa desenvolvido em C++ que utilizando estruturas de repetição gera todas as combinações possíveis e elimina as combinações que não geram um bloco de veículo viável, ou seja, um bloco de veículo que não possa ser executado por um único veículo.

Os dados de entrada utilizados para a resolução do PPV são apresentados na Figura 4.1. Estes dados foram gerados a partir das instâncias reais utilizadas por Reis (2008). Reis (2008) utilizou dados reais de uma empresa de transporte coletivo de uma cidade brasileira, estas instâncias reais estão divididas em 8 garagens, as quais contêm as programações de viagens a serem realizadas em três tipos de dias: úteis, sábados e domingos, remetendo a um total de 24 instâncias de testes.

A primeira coluna da Figura 4.1 apresenta o número da viagem; a segunda coluna apresenta o horário de início da viagem em minutos, por exemplo, o valor 366 indica 06h06 (366 minutos); a terceira coluna apresenta o ponto de início da viagem; a quarta coluna indica o horário de término da viagem, por exemplo, o valor 401 indica 06h41 (401 minutos); e a quinta coluna indica o ponto de término da viagem.

```
[TRIP]
Trip0   366    1       401     1
Trip1   369    1       414     1
Trip2   460    2       492     2
Trip3   469    1       515     1
Trip4   494    1       526     2
Trip5   625    1       660     1
Trip6   631    1       676     1
Trip7   646    1       692     1
Trip8   705    2       754     2
Trip9   806    2       838     2
Trip10  856    2       905     2
Trip11  870    1       909     2
Trip12  881    1       926     1
Trip13  883    1       923     1
Trip14  928    1       959     1
Trip15  936    2       973     2
Trip16  980    1       1029    1
Trip17  1022   1       1066    1
Trip18  1033   1       1072    1
Trip19  1089   2       1139    1
---------------------------------
Total de Viagens: 20
```

Figura 4.1 – Dados de entrada para o PPV.

Para o exemplo apresentado na Figura 4.1 foram selecionadas 20 viagens de uma instância com dados referentes a um dia útil. Como os dados das instâncias reais do trabalho de Reis (2008) já contém a tabela de horários definida, a EEC não será utilizada para resolver o problema da programação de horários. Estas 20 viagens geraram um total de 974 colunas, isto é, 974 blocos de veículos. Onde cada bloco de veículo é um conjunto de viagens que podem serem executadas por um único veículo.

Após a geração das 974 colunas para o PPV foi proposto neste trabalho o modelo de particionamento, apresentado em (4.1) a (4.4), cada linha da matriz está relacionada a uma única viagem, enquanto as colunas se relacionam com os possíveis blocos de veículos. Assim, o elemento $a_{ij}=1$ se a viagem i faz parte do bloco do veículo j e $a_{ij}=0$ caso contrário, para $i=1,\ldots,n$, $j=1,\ldots,m$. A variável $x_j=1$ se o bloco do veículo j faz parte da

solução, ou seja, da escala diária da empresa e $x_j = 0$ caso contrário (4.4). O vetor b tem todos os elementos iguais a 1, garantindo através da restrição (4.2) que cada viagem seja executada uma e uma única vez.

$$Min \sum_{i=1}^{n} c_i x_i \tag{4.1}$$

$$\sum_{i=1}^{n} x_i a_{ij} = b \quad \forall\, j = 1, \ldots, m \tag{4.2}$$

$$\sum_{j=1}^{n} a_{m+1,j} x_j \leqslant \text{max\_dupla\_pegada} \tag{4.3}$$

$$x_i \in 0,1 \quad \forall\, i = 1, \ldots, n \tag{4.4}$$

Ao final das m tarefas é acrescida a linha $m+1$ que se refere aos blocos de veículos do tipo *dupla pegada*, ou seja, aqueles blocos de veículos que possuem um retorno à garagem por um tempo superior a 2 horas. Se o bloco do veículo j for desse tipo, então $a_{m+1,j} = 1$, e $a_{m+1,j} = 0$ caso contrário. Nessa linha a componente b_{m+1} do vetor b tem o número máximo permitido de jornadas do tipo dupla pegada (30% do número total de veículos). A restrição (4.3) garante que esse número nunca seja ultrapassado.

A solução do PPV obtida pelo pacote de otimização (*solver*) GUROBI 5 após resolver o problema de particionamento (4.1)-(4.4) utilizando os dados de entrada da Figura 4.1 resulta em um total de 5 veículos e 19 tarefas apresentadas na Figura 4.2.

A primeira coluna da Figura 4.2 apresenta o número da viagem; a segunda coluna apresenta o horário de início da viagem em minutos, por exemplo, o valor 460 indica 07h40 (460 minutos); a terceira coluna apresenta o ponto de início da viagem; a quarta coluna indica o horário de término da viagem, por exemplo, o valor 492 indica 08h12 (492 minutos); e a quinta coluna indica o ponto de término da viagem. As viagens foram agrupadas em colunas onde cada coluna representa uma jornada de trabalho viável.

As viagens *Trip16* e *Trip18* serão unidas em uma única tarefa, as demais viagens serão iguais a uma tarefa, totalizando 19 tarefas a serem atribuídas às tripulações. Considerando todas as combinações viáveis possíveis resulta em 4.845 jornadas de trabalho viáveis.

Coluna 2:				
Trip2	460	2	492	2
Coluna 296:				
Trip7	646	1	692	1
Trip12	881	1	926	1
Trip17	1022	1	1066	1
Coluna 602:				
Trip6	631	1	676	1
Trip11	870	1	909	2
Trip15	936	2	973	2
Trip19	1089	2	1139	1
Coluna 775:				
Trip1	369	1	414	1
Trip4	494	1	526	2
Trip8	705	2	754	2
Trip9	806	2	838	2
Trip10	856	2	905	2
Coluna 961:				
Trip0	366	1	401	1
Trip3	469	1	515	1
Trip5	625	1	660	1
Trip13	883	1	923	1
Trip14	928	1	959	1
Trip16	980	1	1029	1
Trip18	1033	1	1072	1

Figura 4.2 – Solução do PPV e dados de entrada do PPT.

No modelo de particionamento proposto neste trabalho para o PPT é análogo ao modelo de particionamento proposto para o PPV (apresentado em (4.1)–(4.4)). O modelo de particionamento para o PPT é apresentado em (4.5)–(4.8) cada linha da matriz está relacionada a uma única tarefa, enquanto as colunas se relacionam com as possíveis jornadas de trabalho. Assim, o elemento $a_{ij}=1$ se a tarefa i faz parte da jornada j e $a_{ij}=0$ caso contrário, para $i=1,\dots,m$, $j=1,\dots,n$.

A variável $x_j=1$ se a jornada j faz parte da solução, ou seja, da escala diária da empresa e $x_j=0$ caso contrário (4.8). O vetor b tem todos os elementos iguais a 1, garantindo através da restrição (4.6) que cada tarefa seja executada uma e uma única vez. Ao final das m tarefas é acrescida a linha $m+1$ que se refere às jornadas do tipo *dupla pegada*, ou seja, aquelas jornadas compostas por dois períodos de trabalho, geralmente nos horários de pico, com um intervalo entre eles maior do que duas horas.

Caso a jornada j for do tipo *dupla pegada*, então $a_{m+1,\,j}=1$, e $a_{m+1,\,j}=0$ caso contrário. Nessa linha a componente b_{m+1} do vetor b tem o número máximo permitido de

jornadas do tipo dupla pegada, que em geral fica em torno de 15% do número total de tripulantes. A restrição (4.7) garante que esse número nunca seja ultrapassado.

$$Min \sum_{i=1}^{n} c_i x_i \tag{4.5}$$

$$\sum_{i=1}^{n} x_i a_{ij} = b \quad \forall\ j = 1, \ldots, m \tag{4.6}$$

$$\sum_{j=1}^{n} a_{m+1,j} x_j \leqslant \text{max\_dupla\_pegada} \tag{4.7}$$

$$x_i \in 0,1 \quad \forall\ i = 1, \ldots, n \tag{4.8}$$

4.1.2. Geração de Colunas

A grande dificuldade na resolução do PPV ou do PPT é o grande número de jornadas de trabalho viáveis. Devido ao elevado número de jornadas de trabalho viáveis o tempo computacional para enumerar todas as colunas é muito elevado e após gerar todas as colunas o problema de particionamento resultante de difícil resolução (Desrochers & Soumis, 1989; Desrochers *et al*., 1992; Lorena *et al*., 2003; Baldo, 2009; Santos, 2008).

Devido à grande complexidade dos modelos matemáticos para a resolução integrada da programação de veículos e tripulações disponíveis em trabalhos como Huisman & Wagelmans (2006), Friberg & Haase (1999) e Desrochers *et al*. (1992), o escopo deste trabalho restringirá a geração de colunas para a resolução sequencial do PPV e do PPT. O objetivo da Geração de Colunas (GC) neste trabalho é servir como parâmetro de comparação para a qualidade das soluções obtidas pelas meta-heurísticas VNS, VNS-LR e VNDS.

A Tabela 4.1 mostra o resultado de algumas das instâncias de teste deste trabalho relacionadas ao problema de programação dos veículos. As quatro primeiras instâncias são instâncias com dados gerados aleatoriamente para o PPV e as oito instâncias seguintes são dados reais. Observa-se que o número de colunas geradas parece crescer exponencialmente com o aumento do número de viagens.

Tabela 4.1 – Número de colunas geradas versus número de viagens.

Tipo	Nº de viagens	Nº de colunas geradas
Aleatória	10	120
Aleatória	20	974
Aleatória	30	4.060
Aleatória	40	9.880
Real	53	23.426
Real	69	52.394
Real	90	117.480
Real	98	152.096
Real	108	204.156
Real	121	287.980
Real	161	682.640

A Tabela 4.2 mostra o resultado para as mesmas instâncias de teste da Tabela 4.1 mas considerando o número de tarefas relacionadas ao problema de programação da tripulação.

Tabela 4.2 – Número de colunas geradas versus número de tarefas.

Tipo	Nº de tarefas	Nº de colunas geradas
Aleatória	10	330
Aleatória	19	4.845
Aleatória	28	23.751
Aleatória	37	73.815
Real	46	178.365
Real	61	557.845
Real	77	1.426.425
Real	86	2.225.895
Real	93	3.049.501
Real	100	4.082.925
Real	148	19.720.001

Observe que existe uma redução entre o número de tarefas de uma determinada

instância e o número de viagens desta mesma instância. Isto decorre do agrupamento de viagens atribuídas a um mesmo veículo que não podem ser executadas por tripulações diferentes.

Uma observação importante é que apesar do grande número de blocos de veículos viáveis ou jornadas viáveis grande parte não são blocos de veículos interessantes ou jornadas interessantes para a resolução do problema: atendem todas as restrições mas possuem custo alto. Por exemplo, a solução ótima da abordagem sequencial destas instâncias possui poucos blocos de veículos e poucas jornadas de trabalho quando comparadas ao total de colunas geradas.

Analisando-se as colunas geradas é possível concluir que certos blocos de veículos ou certas jornadas de trabalho não fazem parte da solução ótima ou que têm pequenas chances de participar desta solução (por causa do alto custo de algumas colunas). Provavelmente é possível trabalhar com um número reduzido de blocos de veículos e de jornadas e encontrar a solução ótima para a abordagem sequencial. A dificuldade é como retirar estas colunas e garantir matematicamente que estas colunas não fazem parte da solução ótima. O método de geração de colunas que é um método exato e pode encontrar a solução ótima da abordagem sequencial (Desrochers & Soumis, 1989; Desrochers *et al*., 1992; Lorena *et al*., 2003; Baldo, 2009; Santos, 2008).

A geração de colunas é um método empregado em problemas lineares com grande quantidade de dados, podendo também ser utilizado juntamente com a decomposição de Dantzig-Wolfe (Lorena *et al.*, 2003).

A ideia principal do método de geração de coluna é decompor o problema linear original em dois subproblemas de resolução mais fácil do que o problema original utilizando a enumeração explícita (Baldo, 2009). Segundo Santos (2008) a divisão desse problema original é:

- Problema mestre (PM): semelhante ao problema original, entretanto o número de colunas explícitas é menor;
- Subproblema (SP): interage com o problema mestre, gerando colunas novas para que sejam incorporadas ao problema mestre e otimizando o resultado atual;

O problema mestre utiliza uma quantidade de dados relativamente pequena

(quando comparado com o problema original) possibilitando encontrar a solução do problema mestre em um tempo computacional viável. A adição das colunas é feita gradativamente conforme for obtida a solução anterior, sendo que as colunas geradas no subproblema conduzirão a solução otimizada do problema mestre (Santos, 2008).

O subproblema designa as novas colunas e é utilizado como critério de parada. Segundo Santos (2008) a principal dificuldade encontrada durante a utilização do método de geração de colunas é resolver o subproblema de forma eficiente.

A Figura 4.3 mostra a iteração entre o problema mestre e o subproblema no método de geração de colunas. No caso da resolução do PPV o conjunto de colunas com custo mínimo são determinados como sendo o conjunto de blocos de veículos que possuem um custo mínimo e conseguem executar todas as viagens; e gerar novas colunas para o PPV representa gerar novos blocos de veículos que possuem custo reduzido negativo e portanto são candidatos a fazerem parte da solução ótima.

No caso da resolução do PPT, o conjunto de colunas com custo mínimo são determinados como sendo o conjunto de jornadas de trabalho que possuem um custo mínimo e conseguem executar todas as tarefas; e gerar novas colunas para o PPT representa gerar novas jornadas de trabalho que possuem custo reduzido negativo e, portanto, são candidatas a fazerem parte da solução ótima.

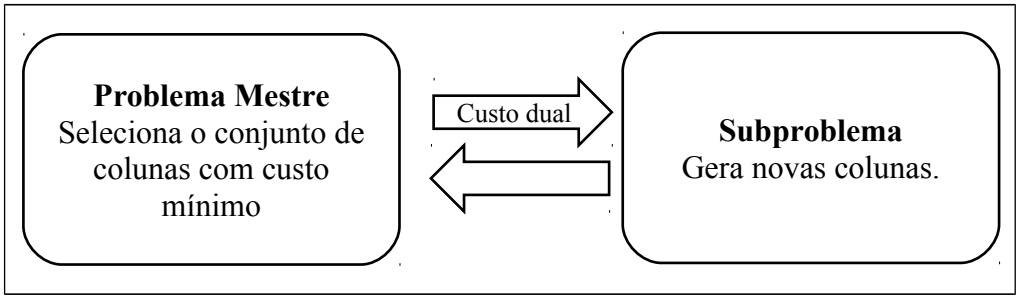

Figura 4.3 – Iteração dos problemas mestre e subproblema (Santos, 2008).

4.1.3. Problema Mestre

O problema mestre aplicado ao PPV é um problema de particionamento de conjuntos (SPP – *Set Partitioning Problem*). Cada variável do problema (cada coluna da

matriz de cobertura) é um bloco de veículo viável que cobre um conjunto de viagens. Cada linha da matriz de cobertura, isto é, cada restrição é uma viagem que necessita ser coberta. Devido ao grande número de blocos de veículos viáveis, o PPV possui muitas colunas e o método de geração de colunas é indicado para a resolução do PPV.

Para o caso do PPT o problema mestre é análogo ao problema mestre do PPV, ou seja, o problema mestre aplicado ao PPT é um problema de particionamento de conjuntos. Cada variável do problema (cada coluna da matriz de cobertura) é uma jornada de trabalho viável que cobre um conjunto de tarefas. Cada linha da matriz de cobertura, isto é, cada restrição é uma tarefa que necessita ser coberta. Devido ao grande número de jornadas de trabalho viáveis, o PPT possui muitas colunas e o método de geração de colunas é indicado para a resolução do PPT.

O SPP é um problema de programação linear inteira, logo não possui a propriedade de integralidade da solução e, consequentemente, não possui um problema dual associado equivalente ao problema primal. Esta característica do SPP dificulta o cálculo dos custos duais das viagens ou das tarefas.

A forma utilizada neste trabalho para resolver este problema é resolver um problema de relaxação linear do SPP. Nesse caso as variáveis correspondentes a cada coluna são não-negativas ao invés de serem binárias. As restrições de não-negatividade e as restrições de cobertura impõem que as variáveis correspondentes a cada coluna tenham o seu valor limitado a 1 (um) mesmo que esta limitação não esteja explícita.

O método de *branch-and-bound* é utilizado no caso da solução ótima do problema relaxado não seja uma solução inteira. O modelo matemático (4.9)-(4.12) apresenta o problema relaxado para o SPP.

$$Min \sum_{i=1}^{n} c_i x_i \tag{4.9}$$

$$\sum_{i=1}^{n} x_i a_{ij} = b, \forall\ j = 1, \ldots, m \tag{4.10}$$

$$\sum_{j=1}^{n} a_{m+1,j} x_j \leq \text{max\_dupla\_pegada} \tag{4.11}$$

$$x_i \geqslant 0, \forall\, i = 1, \ldots, n \tag{4.12}$$

O método denominado de *branch-and-bound* baseia-se na ideia de desenvolver uma enumeração inteligente das soluções candidatas à solução ótima inteira de um problema. O termo *branch* refere-se ao fato de que o método efetua partições no espaço das soluções. O termo *bound* ressalta que a prova da otimalidade da solução utiliza-se de limites calculados ao longo da enumeração. A Figura 4.4 apresenta as etapas do método *branch-and-bound.*

Passo 0 - Inicialização: Faça $\bar{z}=\infty$, $z^{star}=-\infty$, $x^{star}=\emptyset$, $L=P$

Passo 1 – seleção de nó: Selecione o nó ativo i, associado ao problema P^i, da lista de nós ativos. Se a lista estiver vazia, vá para o passo 6.

Passo 2 – teste de eliminação 1: Se a região factível de PL^i for vazia, vá para o passo 1.

Passo 3 – teste de eliminação 2: Se o valor z^i da solução ótima de PL^i é tal que $z^i \leqslant z^{star}$, vá para o passo 1.

Passo 4 – teste de eliminacão 3: Se a solucão ótima x^i de PL^i é inteira com valor z^i, e se $z^i > z^{star}$. atualize x^{star} e z^{star}. Elimine os nós ativos i da lista L, tais que $z^i \leqslant z^{star}$, e volte para o passo 1.

Passo 5 – ramificação: Selecione uma variável da solução ótima x^i de PL^i com valor não inteiro e divida P^i em dois problemas. Adicione estes problemas à lista L e vá para o passo 1.

Passo 6 – fim: Se $z^{star}=-\infty$, não existe solução factível; caso contrário, a solução x^{star} é uma solução ótima.

Onde: z^i é o valor ótimo do problema P^i (limitante inferior de z); z^i é o valor ótimo do problema PL^i (limitante superior de z): x^{star} é a melhor solução encontrada até o momento; z^{star} é o valor de x^{star}

Figura 4.4 – Etapas do método *branch-and-bound* (Arenales *et al.,* 2006*)*.

4.1.4. Subproblema

O objetivo do subproblema para o PPV é gerar um ou mais blocos de veículos novos (novas colunas) viáveis e com custo reduzido negativo, quando considerado os custos duais das viagens do problema mestre resolvido anteriormente. Estas novas colunas são adicionadas ao problema mestre e melhoram o valor da solução corrente ou atualizam os custos duais para permitir que o subproblema encontre novas colunas em sua próxima

execução.

A geração de um novo bloco de veículo é feita através da modelagem em grafos. Utiliza-se um grafo direcionado $G=(V,A)$ onde: o conjunto dos vértices V possui um vértice v_i para cada viagem i ($\forall i=1,\ldots,n$) a ser coberta no problema mestre e dois vértices adicionais que representam a garagem v_0 (nó *source* ou fonte) e v_{n+1} (nó *sink* ou sorvedouro); o conjunto dos arcos A que contém os arcos ($v_{0,}v_i$) e ($v_i, v_{(n+1)}$), $\forall i=i,\ldots,n$ e os arcos (v_i, v_j) para todo par de viagens i e j tal que a viagem j possa ser executada após a viagem i por um único veículo no mesmo bloco de veículo. Na definição destes últimos arcos são considerados os horários de execução das viagens. Desta forma, a existência de um arco (v_i, v_j) indica que o horário de execução da viagem i é anterior ao horário de execução da viagem j , que não ocorre sobreposição de horários, e que essa sequência atende todos os requisitos do problema abordado.

O grafo $G=(V,A)$ assim construído é um grafo direcionado acíclico, e qualquer bloco de veículo viável é um caminho do vértice v_0 ao vértice v_{n+1} no grafo. Os vértices v_i presentes neste caminho representam as viagens i do bloco de veículo construído. A cada vértice v_i é associado um custo c_i referente ao custo dual do bloco de veículo equivalente ao bloco de veículo correspondente no problema mestre.

O objetivo do subproblema para o PPT é gerar uma ou mais jornadas de trabalho novas (novas colunas) viáveis e com custo reduzido negativo, quando considerado os custos duais das tarefas do problema mestre resolvido anteriormente. Estas novas jornadas de trabalho são novas colunas, que após serem inseridas no problema mestre, melhoram o valor da solução corrente ou atualizam os custos duais para permitir que o subproblema encontre novas jornadas de trabalho (novas colunas) em sua próxima execução.

A geração de uma nova jornada de trabalho é realizada utilizando modelagem em grafos. Utiliza-se um grafo direcionado $G=(V,A)$ onde: o conjunto dos vértices V possui um vértice v_i para cada tarefa i ($\forall i=1,\ldots,n$) a ser coberta no problema mestre e dois vértices adicionais que representam a garagem v_0 (nó *source* ou fonte) e v_{n+1} (nó *sink* ou sorvedouro); o conjunto dos arcos A que contém os arcos ($v_{0,}v_i$) e ($v_i, v_{(n+1)}$), $\forall i=i,\ldots,n$ e os arcos (v_i, v_j) para todo par de tarefas i e j tal que a tarefa j possa ser

executada após a tarefa i por uma mesma tripulação (cobrador e motorista) em uma única jornada de trabalho. Na definição destes últimos arcos são considerados os horários de execução de cada tarefa. Desta forma, a existência de um arco (v_i, v_j) indica que o horário de execução da tarefa i é anterior ao horário de execução da tarefa j, que não ocorre sobreposição de horários, e que essa sequência atende a todos os requisitos do problema abordado.

O grafo $G=(V,A)$ assim construído é um grafo direcionado acíclico, e qualquer jornada de trabalho viável é um caminho entre o vértice v_0 e o vértice v_{n+1} no grafo. Os vértices v_i presentes neste caminho representam as tarefas i da jornada de trabalho percorrida. A cada vértice v_i é associado um custo c_i referente ao custo dual da jornada equivalente à jornada de trabalho correspondente no problema mestre.

O caminho mais curto entre o nó fonte e o nó sorvedouro representará a solução do subproblema que será a coluna que está sendo procurada para ser adicionada ao problema mestre (Salomão, 2005; Baldo, 2009; Santos, 2008). Se este caminho possuir custo negativo, a coluna (bloco do veículo) será adicionada ao problema mestre e então o problema mestre é resolvido novamente, obtendo-se novos custos duais para realizar a busca de um outro caminho mínimo no grafo com os custos dos vértices atualizados. Quando o caminho mínimo possuir um custo positivo ou nulo, a solução corrente do problema mestre é a solução ótima do problema relaxado (Salomão, 2005; Baldo, 2009; Santos, 2008).

Segundo Santos (2008), a otimalidade da solução do problema mestre é garantida se o subproblema for resolvido de forma exata. Todas as colunas com custo reduzido negativo devem ser encontradas, isto é, deve se resolver o problema de caminho mínimo por um método exato (Desrochers & Soumis, 1989; Salomão, 2005).

Freling *et al.* (1999) e Freling *et al.* (2003) apresentam um modelo inteiro que combina uma formulação de quase-alocação (*quasi-assignment*) para programação de veículos e uma formulação de *set partitioning* para a programação da tripulação. Eles propuseram uma abordagem baseada em geração de colunas onde o problema principal é resolvido por uma relaxação lagrangeana.

Righini & Salani (2006) utilizam o método de geração de colunas para o problema de roteamento de veículos e utilizam programação dinâmica bidirecional delimitada para resolver o problema de caminho mínimo do subproblema.

Mesquita & Paias (2008) utilizam um algoritmo adaptativo baseado em relaxação lagrangeana para resolver o subproblema da abordagem baseada em geração de colunas aplicada ao PPVT. Santos (2008) utiliza o algoritmo de *Floyd-Warshall* para encontrar o caminho mínimo na rede do subproblema para selecionar as colunas a serem adicionadas ao problema mestre.

Após o grafo ser montado foi utilizado um método exato para resolver o subproblema de forma a garantir a otimalidade da solução do problema mestre. Neste trabalho foi escolhido o algoritmo *Radix Heap* para resolver um problema de caminho mínimo com a diferença que o objetivo não é encontrar o menor caminho mas sim encontrar todos os caminhos que tenham um custo reduzido negativo. O *Radix Heap* foi escolhido devido a ser um algoritmo exato de fácil implementação computacional, eficaz e eficiente para resolver o problema de caminho mínimo (Reis *et al.*, 2007).

4.1.5. *Radix Heap*

Considerando que o PPV (ou o PPT) pode ser representado por um grafo $G=(N,A)$, onde N é o conjunto de nós e A é o conjunto de arcos que interligam estes nós. Neste trabalho será considerado o número de nós $|N|=n$ e o número de arcos $|A|=m$; para cada arco $(i,j)\in A$ está associado um custo unitário c_{ij}. O caminho entre um nó origem (s) e um nó destino (t) é definido por uma sequência de arcos: $(s,i),...(k,l),...(j,t)$. O Problema de Caminho Mínimo (PCM) consiste em determinar um caminho entre s e t tal que a somatória dos custos unitários dos arcos (ou alguma outra medida de impedância) que compõem este caminho seja o mínimo.

O algoritmo *Radix Heap* foi proposto inicialmente por Ahuja *et al.* (1990) e ainda é considerado no meio científico como um dos algoritmos mais eficientes para resolver o problema do caminho mínimo. Segundo Ahuja *et al.* (1990), a complexidade do algoritmo *Radix Heap*, o qual é uma evolução do Algoritmo *Dial*, é $O(m+n\log(nC))$. A Figura 4.5 apresenta o pseudocódigo do algoritmo *Radix Heap*.

Cada nó da rede possui um rótulo com a distância entre este nó e o nó origem. O algoritmo inicia todos os rótulos como temporários e com valor infinito, cria K *buckets*

vazios com faixas de valores que variam entre 2^{k-1} e $2^k - 1$ com $k = 1, 2, \ldots, K$. Em seguida, insere cada nó i no *bucket* cuja faixa de valores corresponda ao valor atual do rótulo do nó i.

Em seguida, o algoritmo seleciona o nó com o menor rótulo temporário e o rotula permanentemente. A partir deste último nó rotulado permanentemente, o algoritmo varre todos os arcos que saem deste nó e atualiza as etiquetas temporárias dos nós que foram alcançados a partir deste último nó rotulado permanentemente.

```
Procedimento Radix Heap
    1.  inicializa os buckets;
    2.  Para i = 0 até i = Número de buckets Faça
    3.      Ini Bucket = 2^(k-1) ; Fim Bucket = 2^k - 1 ;
    4.  Fim-para;
    5.  Para i = 1 até Número de nós Faça
    6.      s[i]= 0, nó i não rotulado;
    7.      d[i] = MAXINT, distancia da origem ao nó i igual a infinito;
    8.  Fim-para;
    9.  Enquanto todos os nós não forem rotulados permanentemente Faça
    10.     escolha o nó com o menor rótulo temporário;
    11.     remove do heap o nó que será rotulado permanentemente;
    12.     Se o bucket estiver muito cheio Faça
    13.         Recalcula as faixas de valores;
    14.         Redistribui os nós;
    15.     Fim-se;
    16.     rotula o nó i permanentemente;
    17.     k = o primeiro arco que sai do nó i para o nó j;
    18.     Enquanto k diferente de 0 Faça
    19.         d[j]= d[i]+dist[k], atualiza a distância do nó origem ao nó j;
    20.         insere o nó j no bucket que contenha a sua faixa de valores;
    21.         k= linkout[k], k recebe o próximo arco ou zero caso tenha
    22.                         acabado de percorrer todos os arcos que saem do nó i;
    23.     Fim-enquanto;
    24. Fim-enquanto;
Fim Radix Heap
```

Figura 4.5 – Pseudocódigo do Radix Heap.

O rótulo temporário de um nó somente é atualizado quando o novo valor do rótulo for inferior ao atual, isto é, o novo caminho encontrado possui o custo inferior ao do caminho atual. Quando um rótulo temporário é atualizado o algoritmo verifica se o nó referente a este rótulo deve ser realocado para um novo *bucket* ou se deve continuar no mesmo *bucket*.

Quando um *bucket* contiver muitos nós dentro dele, as faixas de valores de todos os *buckets* serão recalculadas e o conteúdo deste *bucket* será dividido entre todos os demais *buckets*. O critério de parada do *Radix Heap* ocorre quando o nó destino for rotulado ou quando todos os nós receberem rótulos permanentes. Um fator importante a ser observado do algoritmo *Radix Heap* é que as dimensões dos *buckets* crescem exponencialmente e a sua faixa de valores muda dinamicamente enquanto o algoritmo é executado.

4.1.6. Aplicação do método de Geração de Colunas ao PPV

A Figura 4.6 apresenta as etapas do método de geração de colunas aplicado ao PPV. O passo 0 envolve a inicialização do método de geração de colunas, neste passo é gerado um conjunto inicial de colunas (cada coluna representa um bloco de veículo). Nesta geração inicial de colunas é importante garantir que o conjunto de blocos de veículos possua ao menos uma solução factível.

De forma a garantir a viabilidade deste conjunto inicial de colunas cada bloco de veículo recebe uma única viagem, ou seja, a matriz inicial é uma matriz identidade. Nesta geração de colunas iniciais também são adicionadas as colunas nas quais os blocos de veículos possuem duas viagens, isto permite que exista mais de uma solução viável no conjunto inicial de colunas.

O passo 1 do método de geração de colunas aplicado ao PPV envolve adicionar os novos blocos de veículos selecionados pela resolução do subproblema ou pela geração inicial de blocos de veículos. O passo 2 do método de geração de colunas é resolver o problema relaxado do PPV (o qual é um problema de particionamento de conjuntos). O GUROBI foi o pacote de otimização utilizado e as expressões (4.9) a (4.12) apresentam o modelo matemático utilizado e envia os custos duais calculados para o subproblema. No passo 3, caso a solução encontrada não seja inteira, o método de *branch-and-bound* (Figura 4.4) é utilizado para encontrar uma solução inteira para o PPV.

O passo 4 do método de geração de colunas aplicado ao PPV envolve a montagem do grafo com os blocos de veículos. O nó fonte e o nó sorvedouro do grafo correspondem à garagem do veículo e, cada nó intermediário é uma viagem diária do PPV. Caso exista um arco (i, j) interligando o nó i ao nó j isto representa que a viagem j pode ser executada

após a viagem i pelo mesmo veículo que executou a viagem i. Cada caminho (sequência de nós e arcos) entre o nó fonte e o nó sorvedouro representa um bloco de veículo viável, isto é, uma coluna viável. Caso este caminho tenha um custo reduzido negativo, ele será selecionado pelo algoritmo *Radix Heap* (Figura 4.5) durante a execução do passo 5 do método de geração de colunas.

Passo 0 → Inicialização: gerar um conjunto inicial de blocos de veículos;

Passo 1 → Problema Mestre: adicionar os novos blocos de veículos selecionados pela resolução do subproblema ou pela geração inicial;

Passo 2 → Problema Mestre: resolver o problema relaxado para o SPP (modelo matemático (4.9)-(4.12)) utilizando o pacote de otimização GUROBI e enviar os custos duais para o subproblema;

Passo 3 → Problema Mestre: caso a solução encontrada não seja inteira, utilize o método de *branch-and-bound* para encontrar uma solução inteira;

Passo 4 → Subproblema: montar o grafo representando o PPV;

Passo 5 → Subproblema: encontrar todos os caminhos do grafo que tenham custo reduzido negativo;

Passo 6 → Subproblema: caso exista algum bloco de veículo com custo reduzido negativo, enviar o bloco de veículo com custo reduzido negativo para o problema mestre e retorna ao Passo 1; Caso contrário retornar a solução ótima do problema mestre.

Figura 4.6 – Método de geração de colunas aplicado ao PPV.

O algoritmo *Radix Heap* geralmente retorna o caminho mínimo em cada execução do algoritmo. No caso do PPV o objetivo não é encontrar o caminho mínimo mas retornar todos os caminhos que possuam custo reduzido negativo. Em cada etiqueta do nó é adicionado o nó percorrido anteriormente criando uma lista ligada com custo reduzido negativo.

No passo 6 do método de geração de colunas aplicado ao PPV é enviado o caminho (bloco de veículo) com custo reduzido negativo para o problema mestre e retorna-se ao passo 1. Caso não exista nenhum caminho (bloco de veículo) com custo reduzido negativo, isto significa que não existe nenhuma coluna que possa ser adicionada ao problema mestre que reduza o valor da função de avaliação e a solução atual do problema mestre é a solução

ótima do PPV.

4.1.7. Aplicação do método de Geração de Colunas ao PPT

O método de geração de colunas aplicado ao PPT pode ser executado após a obtenção da solução ótima através do método de geração de colunas ao PPV caracterizando uma abordagem sequencial tradicional do PPV e do PPT; ou pode ser executado antes da resolução do PPV caracterizando uma abordagem sequencial inversa do PPT e do PPV.

As etapas do método de geração de colunas aplicado ao PPT são apresentadas na Figura 4.7. O passo 0 envolve a inicialização do método de geração de colunas para isto as viagens são agrupadas em tarefas. Uma tarefa é um conjunto de viagens que necessitam serem executadas por uma única tripulação devido à ausência de oportunidades de trocas entre as viagens. Uma oportunidade de troca é um intervalo de tempo durante o qual é possível efetuar a rendição de uma tripulação (cobrador e motorista).

No passo 0 também é gerado um conjunto inicial de colunas (cada coluna representa uma jornada de trabalho). Nesta geração inicial de colunas é importante garantir que o conjunto de jornadas de trabalho possua ao menos uma solução factível. Para garantir a viabilidade deste conjunto inicial de colunas cada jornada de trabalho recebe uma única tarefa, ou seja, a matriz inicial é uma matriz identidade. Nesta geração de colunas iniciais também são adicionadas as colunas nas quais as jornadas de trabalho possuem duas tarefas, isto permite que exista mais de uma solução viável no conjunto inicial de colunas.

O passo 1 do método de geração de colunas aplicado ao PPT envolve adicionar as novas jornadas de trabalho selecionadas pela resolução do subproblema ou pela geração inicial das jornadas de trabalho. O passo 2 do método de geração de colunas é resolver o problema relaxado do PPT (o qual é um problema de particionamento de conjuntos). O GUROBI foi o pacote de otimização utilizado e as expressões (4.9) a (4.12) apresentam o modelo matemático utilizado e envia os custos duais calculados para o subproblema. No passo 3, caso a solução encontrada não seja inteira, o método de *branch-and-bound* (Figura 4.4) é utilizado para encontrar uma solução inteira para o PPT.

O passo 4 do método de geração de colunas aplicado ao PPT envolve a montagem do grafo com as jornadas de trabalho. O nó fonte e o nó sorvedouro do grafo correspondem,

respectivamente, ao ponto de início da jornada de trabalho e ao ponto de término da jornada de trabalho e, cada nó intermediário é uma tarefa do PPT. Caso exista um arco (i, j) interligando o nó i ao nó j isto representa que a tarefa j pode ser executada após a tarefa i pela mesma tripulação que executou a tarefa i. Cada caminho (sequência de nós e arcos) entre o nó fonte e o nó sorvedouro representa uma jornada de trabalho viável, isto é, uma coluna viável. Caso este caminho tenha um custo reduzido negativo, ele será selecionado pelo algoritmo *Radix Heap* (Figura 4.5) durante a execução do passo 5 do método de geração de colunas aplicado ao PPT.

Passo 0 → Inicialização: agrupar as viagens em tarefas e gerar um conjunto inicial de jornadas de trabalho;

Passo 1 → Problema Mestre: adicionar as novas jornadas de trabalho selecionados pela resolução do subproblema ou pela geração inicial;

Passo 2 → Problema Mestre: resolver o problema relaxado para o SPP (modelo matemático (4.9)-(4.12)) utilizando o pacote de otimização GUROBI e enviar os custos duais para o subproblema;

Passo 3 → Problema Mestre: caso a solução encontrada não seja inteira, utilize o método de *branch-and-bound* para encontrar uma solução inteira;

Passo 4 → Subproblema: montar o grafo representando o PPT;

Passo 5 → Subproblema: encontrar todos os caminhos do grafo que tenham custo reduzido negativo;

Passo 6 → Subproblema: caso exista alguma jornada de trabalho com custo reduzido negativo, enviar a jornada de trabalho com custo reduzido negativo para o problema mestre e retorna ao Passo 1; Caso contrário retornar a solução ótima do problema mestre.

Figura 4.7 – Método de geração de colunas aplicado ao PPT.

O algoritmo *Radix Heap* geralmente retorna o caminho mínimo em cada execução do algoritmo. No caso do PPT o objetivo não é encontrar o caminho mínimo mas retornar todos os caminhos que possuam custo reduzido negativo. Em cada etiqueta do nó é adicionado o nó percorrido anteriormente criando uma lista ligada com custo reduzido negativo.

No passo 6 do método de geração de colunas aplicado ao PPT é o caminho

(jornada de trabalho) com custo reduzido negativo para o problema mestre e retorna-se ao passo 1. Caso não exista nenhum caminho (jornada de trabalho) com custo reduzido negativo, isto significa que não existe nenhuma coluna que possa ser adicionada ao problema mestre que reduza o valor da função de avaliação e a solução atual do problema mestre é a solução ótima do PPT.

4.2. META-HEURÍSTICAS

Nesta seção são apresentadas três meta-heurísticas para resolver o PPHVT. A primeira meta-heurística é a Busca em Vizinhança Variável (VNS), a segunda meta-heurística é baseada em Busca em Vizinhança Variável com Lista Restrita (VNS-LR) e a terceira meta-heurística é baseada em Busca Decomposta em Vizinhança Variável (VNDS) Também serão apresentadas a heurística construtiva para a solução inicial, a função de avaliação utilizada, as estruturas de vizinhança desenvolvidas e os procedimentos de busca local utilizados.

4.2.1. Heurística Construtiva para a Solução Inicial

Para a aplicação do VNS, do VNS-LR ou do VNDS, é necessária a geração de uma solução inicial, a qual consiste de uma solução *s* com todas as viagens diárias alocadas aos veículos da frota e as tripulações designadas aos veículos programados para realizar as viagens.

A solução inicial para o problema de programação integrada da tabela de horários, veículos e tripulações (PPHVT), é obtida por meio de uma heurística construtiva, a qual será detalhada nesta seção. O pseudocódigo da heurística construtiva da solução inicial é apresentado na Figura 4.8.

Uma empresa de transporte coletivo urbano forneceu os dados da sua operação diária contendo o número da linha, o número do veículo, o horário de início, o horário de término de cada viagem e o número de passageiros transportados em cada viagem. A Figura 4.9 demonstra um exemplo destes dados informados.

```
Procedimento Heurística construtiva para a Solução Inicial
    1.  Separar os dados de entrada por linha;
    2.  Para cada linha i=1 até total de linhas faça
    3.    Somar o número de passageiros por hora;
    4.    Calcular o tempo médio de viagem por hora;
    5.    Definir intervalo entre partidas;
    6.    Definir a hora inicial de cada viagem;
    7.    Para cada veículo j=1 até total de veículos da linha i faça
    8.      k=0;
    9.      Faça
    10.       veiculo[j].viagem_ini[k] = hora inicial;
    11.       veiculo[j].viagem_fim[k] = hora inicial+tempo viagem;
    12.       k++;
    13.     Enquanto (veiculo[j].viagem_fim[k-1]<horario final da linha i);
    14.   Fim-para
    15.   Para cada veículo j=1 até total de veículos faça
    16.       Para cada viagem k até total de viagens do veículo j faça
    17.         Se (tempo_troca_trip < tempo_min)
    18.           Então forme a tarefa k agrupando a viagem k à viagem k-1;
    19.                 Senão
    20.                       tarefa k ← viagem i; k++;
    21.         Fim-se;
    22.       Fim-para;
    23.   Fim-para;
    24.   Para cada tarefa k até total de tarefas do veículo j faça
    25.     tripulação livre mais próxima ← tarefa k;
    26.   Fim-para;
    27. Fim-Para
Fim
```

Figura 4.8 – Pseudocódigo da heurística construtiva da solução inicial.

Linha	N. Veículo	Início	Final	N. Passageiros
0001	1000	05h30	06h25	37
0001	2000	05h40	06h26	30
0001	3000	05h44	06h40	60
0001	4000	05h52	06h57	77
0001	5000	06h00	06h59	69
0001	6000	06h05	07h05	41

Figura 4.9 – Exemplo dos dados de entrada.

A partir destes dados informados, o programa desenvolvido calcula a demanda de cada linha (número de passageiros por hora) e calcula o tempo médio de viagem em cada hora. Desta forma, os dados utilizados conseguem representar o aumento da demanda de

passageiros, nos horários de pico e o aumento dos congestionamentos, nos horários de pico, devido à variação do tempo de viagem. Como dado de entrada, foi considerada a capacidade dos veículos de 110 passageiros. Com estas informações, foram calculados: a demanda de passageiros e o tempo de viagem para cada hora de operação de cada linha.

Tabela 4.3 – Demanda de passageiros e tempo de viagem para a linha 01.

Horário	Demanda (nº de passageiros)	Tempo viagem (min)
05h00 - 06h00	217	55
06h00 – 07h00	837	68
07h00 – 08h00	670	65
08h00 – 09h00	137	68
09h00 – 10h00	249	66
10h00 – 11h00	304	67
11h00 – 12h00	317	70
12h00 – 13h00	513	75
13h00 – 14h00	370	71
14h00 – 15h00	340	72
15h00 – 16h00	369	69
16h00 – 17h00	461	72
17h00 – 18h00	366	74
18h00 – 19h00	271	77
19h00 – 20h00	286	64
20h00 – 21h00	189	56
21h00 – 22h00	86	65

A Tabela 4.3 apresenta a demanda de passageiros e o tempo de viagem para cada hora de operação da linha 01. A demanda de passageiros apresentada na Tabela 4.3 foi calculada a partir de todos os dados de entrada fornecidos para cada linha.

Para determinar o número de veículos necessários em cada hora, faz-se a divisão da demanda (número de passageiros) pela capacidade do veículo (110 passageiros). Ao passo que, o intervalo entre partidas das viagens é calculado dividindo o intervalo da demanda pelo número de veículos. Por exemplo, entre as 05h00 e 06h00 são necessários $217/110=1{,}97$ (arredondando o valor para cima resulta em 2 veículos) na linha 01 e o intervalo entre partidas é de $(06\text{h}00-05\text{h}00)/2=30$ minutos. Desta forma, serão realizadas duas viagens na linha 01 entre as 05h00 e 06h00. A primeira viagem terá início às 05h00 e a segunda viagem terá início

às 05h30. O tempo de duração de cada viagem será de 55 minutos conforme a Tabela 4.3.

Outro exemplo, entre as 06h00 e 07h00 existe uma demanda de 837 passageiros, logo como $837/110=7,61$ arredondando para cima, resulta em 8 veículos. Como no horário anterior, havia dois veículos operando nesta linha, será necessário adicionar mais 6 veículos na linha 01. O intervalo entre as partidas será $(07\text{h}00-06\text{h}00)/8=7,5$ minutos, arredondando para cima resulta em 8 minutos. Desta forma, os horários de partida serão 06h00, 06h08, 06h16, 06h24, 06h32, 06h40, 06h48 e 06h56 e a duração de cada viagem será de 68 minutos.

Considerando-se que os dois veículos, que já estavam operando na linha 01, executaram as viagens entre 05h00 e 05h55 e entre 05h30 e 06h25, estes veículos poderão, respectivamente, executar as viagens com início às 06h00 e às 06h32. A forma de alocação dos veículos é alocar um veículo j disponível a uma viagem k, com início no mesmo ponto de término da viagem anterior do veículo j.

Caso não exista nenhum veículo disponível no ponto de início da viagem k, é alocado um veículo que será deslocado da garagem da empresa. Desta forma, o objetivo é reduzir o tempo de cada veículo ocioso e, consequentemente, o número de veículos necessários para operar cada linha.

Após todas as viagens terem sido alocadas aos veículos, são identificadas as oportunidades de troca (OT), isto é, oportunidades nas quais existe um tempo mínimo para a rendição de uma tripulação por outra e existe um fiscal da empresa naquele terminal onde a troca é realizada. As viagens consecutivas, de um mesmo veículo, entre duas OT são agrupadas em tarefas, e essas tarefas são alocadas às tripulações.

Após todas as viagens terem sido atribuídas a algum veículo, a heurística da solução inicial divide estas viagens em tarefas e atribui cada tarefa à tripulação livre mais próxima do local de realização desta tarefa.

A tripulação livre mais próxima pode ser definida como a tripulação que não esteja executando nenhuma tarefa entre o horário de início e o horário de término da tarefa a ser alocada e se encontra no mesmo ponto inicial da tarefa a ser alocada ou no ponto mais próximo. A distância é definida considerando-se o tempo de deslocamento da tripulação, juntamente com um veículo, até o ponto inicial da tarefa.

O número de veículos inicialmente disponíveis é um dado de entrada do problema. Desta forma, pode ser definido conforme o tamanho da frota disponível, para operar as linhas que serão programadas. No início da heurística, todos os veículos estão

disponíveis na garagem da empresa e todas tripulações são consideradas disponíveis na garagem da empresa.

Caso a frota disponível não seja suficientemente grande, para atender todas as viagens programadas, são criados veículos extras para atender esta demanda. Uma solução que utilize veículos extras para executar as viagens programadas, é considerada viável, apesar destes veículos extras não existirem a princípio. Caso a empresa deseje executar esta solução, será necessário o deslocamento de veículos de outra garagem ou adquirir novos veículos.

No caso de o número de tripulações disponíveis não ser suficientemente grande, para atender todas as tarefas programadas, são criadas jornadas de trabalho extras para atender esta demanda. Uma solução que utiliza jornadas de trabalho extras, para executar as tarefas programadas, é considerada viável. Caso a empresa deseje executar esta solução, será necessário o deslocamento de tripulantes de outras linhas ou contratar novos funcionários.

Após todas as viagens e tarefas serem atribuídas, respectivamente, aos veículos e tripulações, é verificada a existência dos intervalos de descansos regulamentados, pela legislação trabalhista para cada tripulação. No caso de existir alguma tripulação sem o intervalo de descanso necessário, o mesmo é adicionado ao final da jornada de trabalho da tripulação, como um tempo adicional que a tripulação permanecerá parada no ponto final da viagem (garagem ou terminal). Desta forma, a heurística construtiva não gera soluções iniciais inviáveis, ou seja, as soluções geradas inicialmente podem ser executadas em situações práticas, embora não assegurem os menores custos operacionais.

4.2.2. Função de Avaliação

A função de avaliação tem o objetivo de quantificar a qualidade de uma solução. A função de avaliação proposta considera os atributos clássicos da programação medidos pelo atendimento da demanda de passageiros, pelo número de veículos utilizados e pelo número de tripulantes alocados, e também visa a avaliar a qualidade de uma solução para o PPHVT. A solução para o PPHVT considera a penalização de cada uma das restrições essenciais e não essenciais que forem infringidas. Para cada restrição infringida é atribuído um peso numérico, conforme a importância daquela restrição para a empresa de transporte coletivo.

Desta forma, a função de avaliação não representa um valor monetário para o

custo operacional de uma determinada programação, mas determina um valor numérico representativo da qualidade de cada solução para o PPHVT.

A qualidade de uma solução não é determinada somente pelo custo operacional, embora também depende de fatores inerentes a sua aplicabilidade como, por exemplo, a quantidade de vezes que um veículo ou um tripulante pode alternar entre duas linhas de ônibus distintas. Assim, a função de avaliação proposta é dada pela somatória dos produtos dos pesos pelas restrições essenciais e não essenciais.

A função de avaliação utilizada, é baseada na função de avaliação proposta por Reis (2008) e visa a avaliar matematicamente uma solução *s* para o PPHVT por meio da penalização de cada uma das restrições essenciais e não-essenciais que forem infringidas. A mesma é dada pela expressão (4.13), onde s é uma solução para o PPHVT.

$$FA(s) = FA_{PPH}(s) + FA_{PPV}(s) + FA_{PPT}(s) \quad (4.13)$$

Conforme a expressão (4.13), a função de avaliação é dada pela soma das parcelas referentes ao PPH, ao PPV e ao PPT. A expressão (4.14) apresenta a função de avaliação do PPH, a qual é definida pela soma de duas parcelas definidas pelas restrições essenciais e pelas restrições não essenciais referentes à tabela de horários.

A expressão (4.15) apresenta a função de avaliação do PPV, a qual é determinada pela soma de duas parcelas, definidas pelas restrições essenciais e pelas restrições não essenciais referentes aos veículos. A função de avaliação do PPT, apresentada na expressão (4.16), é análoga à referente ao PPV e ao PPH, com a diferença de considerar apenas as restrições referentes aos tripulantes.

$$FA_{PPH}(s) = FRE_{PPH}(s) + FRNE_{PPH}(s) \quad (4.14)$$

$$FA_{PPV}(s) = FRE_{PPV}(s) + FRNE_{PPV}(s) \quad (4.15)$$

$$FA_{PPT}(s) = FRE_{PPT}(s) + FRNE_{PPT}(s) \quad (4.16)$$

A parcela FRE(s), dada pela expressão (4.17), é composta pelas restrições essenciais descritas na seção 3.3.1. Já a expressão (4.18), computa as penalidades referentes ao não atendimento das restrições não essenciais consideradas, as quais também se encontram na seção 3.3.1.

$$FRE(s) = A_1 \times FRE_1(s) + A_2 \times FRE_2(s) + \ldots + A_i \times FRE_i(s) \quad (4.17)$$

$$FRNE(s) = B_1 \times FRNE_1(s) + B_2 \times FRNE_2(s) + \ldots + B_j \times FRNE_j(s) \quad (4.18)$$

Os valores dos pesos são dados de entrada para o PPHVT, definidos a partir das características desejáveis para a solução e ajustados, empiricamente, de acordo com a realidade local. Esse tipo de abordagem é fundamental para permitir gerar programações de veículos e tripulantes, que sejam consideradas viáveis pelos operadores do transporte público.

Este ajuste empírico dos pesos é realizado, observando-se as soluções obtidas e avaliando se essas refletem uma solução considerada como de boa qualidade pela empresa. Para tanto, é necessário definir o que é uma solução de boa qualidade para a empresa. Para o caso estudado, uma solução com um número reduzido de veículos, mas com poucas viagens mortas (viagens para reposicionamento de ônibus sem passageiros) e, um número reduzido de tripulações, mas com poucas horas extras, foi considerada como sendo de boa qualidade.

Entretanto, reduzir o número de veículos resulta em aumentar a quantidade de viagens mortas e a consequência de reduzir o número de tripulações é aumentar o tempo total de horas extras.

Basicamente, o ajuste empírico dos pesos consiste em, por exemplo, aumentar o peso por minuto de viagem morta e observar como isto influencia na redução do número do tempo de viagem morta, sem que ocorra um aumento significativo no número de veículos ou de tripulações da solução obtida.

O ajuste de um grande número de parâmetros é muito difícil, devido à existência de objetivos conflitantes no PPHVT, por exemplo, minimizar o número de tripulações pode resultar no aumento do número de horas extras. Parâmetros mal ajustados podem conduzir o método de busca em vizinhança variável (VNS) a uma direção errada, durante a exploração do espaço de soluções viáveis. Desse modo, o mesmo poderá retornar uma solução de baixa qualidade, como se fosse a melhor possível.

Quando é utilizado o ajuste empírico dos pesos deve-se realizar alguns questionamentos como: quanto custa uma tripulação ou um veículo adicional; se é melhor reduzir o número de horas extras ou o número de tripulações; a partir de quanto tempo parado no terminal é preferível enviar o veículo para a garagem, entre outros. Essas perguntas são difíceis de serem respondidas (inclusive em alguns casos as empresas operadoras não sabem a

resposta). Entretanto o VNS, o VNS-LR e o VNDS necessitam de uma função matemática que indique por meio de um valor numérico o quanto que a solução s é superior ou inferior à solução s'. Não é possível garantir, que os pesos utilizados são os melhores possíveis, mas os pesos utilizados são formam o melhor ajuste encontrado para o PPHVT.

Tabela 4.4 – Valores limites de restrições e critérios determinados empiricamente.

Parâmetro	Critério
Porcentagem Máxima de Veículos com Dupla Pegada	30%
Porcentagem Máxima de Tripulações com Dupla Pegada	15%
Tempo Máximo de Atraso de uma Viagem	00h05
Tempo Máximo de Antecipação de uma Viagem	00h05
Tempo Mínimo de Permanência do Veículo na Garagem	01h00
Tempo Normal de Trabalho da Tripulação	07h10
Tempo Máximo de Horas Extras	01h00
Tempo Máximo de Horas Super Extras	01h00
Tempo Mínimo para Troca de Tripulações	00h05
Tempo de Folga Corrida (tripulação)	00h10
Tempo de Folga Acumulada (tripulação)	00h20
Tempo Mínimo para uma Tripulação ser Dupla Pegada	02h00
Tempo Mínimo para um Veículo ser Dupla Pegada	02h00
Peso por Duas Viagens com mesmo Horário Inicial	1.000
Peso por Ausência do Tempo de Folga (tripulação)	80
Peso por Minuto de Hora Extra (tripulação)	2
Peso por Minuto de Hora Super Extra (tripulação)	10
Peso por Minuto de Hora Ociosa (tripulação)	1
Peso por Minuto de Sobreposição (veículo ou tripulação)	80
Peso por Utilizar um Veículo	1.000
Peso por Utilizar uma Tripulação	1.000
Peso por Excesso de Veículos com Dupla Pegada	800
Peso por Excesso de Tripulações com Dupla Pegada	800
Peso por Minuto de Viagem Morta (veículo)	2
Peso por Minuto de Tempo de Terminal (veículo)	1
Troca de linha por veículo	2
Troca de tripulação por veículo	1

Os pesos têm como objetivo estimar estes valores e, neste trabalho, o modo de ajuste escolhido foi o empírico. A partir do qual, tentou-se calibrar os pesos para que a função

de avaliação tenha a capacidade de avaliar as soluções obtidas e indicar para as heurísticas utilizadas, qual a melhor solução dentro do espaço de soluções que foi explorado. O ajuste empírico dos pesos resultou nos valores da Tabela 4.4.

Uma tripulação ou um veículo serão considerados como *dupla pegada,* sempre que a jornada diária for divisível em dois blocos de tarefas ou viagens (respectivamente, tripulação e veículo) e existe um tempo mínimo ocioso entre estes dois blocos, para que seja caracterizada a *dupla pegada*. Normalmente, este tipo de jornada é utilizado para aumentar a oferta de transporte nos horários de pico (manhã e fim de tarde), nos quais ocorrem o aumento da demanda de passageiros.

Devido à jornada do tipo *dupla pegada,* gerar insatisfação por parte dos tripulantes, os quais possuem preferência por jornadas corridas (executar a jornada de trabalho em um único bloco), este tipo de jornada é limitado a uma porcentagem máxima diária calculada, em relação ao total de tripulações ou de veículos utilizados. Sempre que o número de tripulações ou de veículos com *dupla pegada* exceder o valor máximo diário, o número excedente será penalizado, utilizando o valor do peso por excesso de tripulações ou veículos com *dupla pegada*. Esta porcentagem máxima de veículos com dupla pegada e de tripulações com dupla pegada, foi determinada com base nas características das soluções, atualmente em uso por uma empresa de transporte coletivo.

O tempo máximo de atraso ou antecipação de uma viagem, é quanto tempo que o horário inicial de uma viagem pode ser adiantado ou atrasado. Isto permite que, por exemplo, se um veículo j da linha i termine a viagem k às 06h27 e a viagem $k+1$ da linha i inicie às 06h25, esta viagem $k+1$ deverá ser executada por um outro veículo. Entretanto, se atrasarmos o início da viagem $k+1$ para 06h28, o mesmo veículo j que executou a viagem k, poderá executar a viagem $k+1$.

O tempo mínimo de permanência do veículo na garagem é o número de horas que cada um dos veículos deve permanecer na garagem, para manutenção preventiva e/ou limpeza/higienização do veículo (este tempo pode ocorrer durante a jornada, caso o veículo seja recolhido). Caso este tempo não seja respeitado pela solução s, as viagens executadas, durante este tempo mínimo de permanência, serão consideradas como sobreposição. O tempo mínimo para troca da tripulação é o tempo necessário para que uma tripulação, seja rendida por outra tripulação, a qual continuará a operar o mesmo veículo. Para cada veículo utilizado na solução, haverá um peso por utilizar o veículo e para cada tripulação utilizada, também

existirá um peso por utilizar a tripulação.

O tempo de folga corrida é o tempo que uma tripulação deve ter de intervalo de descanso ininterrupto. O tempo de folga acumulada é o tempo total que uma tripulação deve ter para descanso, durante um dia de trabalho, objetivando tornar esta jornada viável. Por exemplo, uma jornada de trabalho será considerada viável, se existir um intervalo de descanso de 10 minutos e mais dois intervalos de 5 minutos, totalizando os 20 minutos de folga acumulada, sendo pelo menos 10 minutos corridos.

A ausência de folga é definida como o tempo (em minutos), que faltou para uma tripulação completar o tempo de folga corrida e/ou o tempo de folga acumulada. A hora *super extra* é definida como o tempo, no qual uma tripulação excede o tempo máximo total de trabalho, considerando o tempo máximo de hora extra.

Será considerado sobreposição, sempre que houver a ocorrência de duas ou mais viagens ou tarefas sendo executadas, respectivamente, por um mesmo veículo ou por uma mesma tripulação. A sobreposição torna a solução inviável, mas uma solução inviável, vizinha a uma solução viável, pode gerar uma solução de boa qualidade no movimento seguinte.

Uma viagem será considerada como viagem morta sempre que ocorrer um percurso ocioso ou morto de um veículo, ou seja, percurso no qual não são transportados passageiros (por exemplo, na viagem entre a garagem e um terminal de ônibus). O tempo de terminal é definido como o tempo que um veículo permanece ocioso, em um terminal de ônibus.

Existe um peso para duas viagens da mesma linha iniciadas no mesmo horário, para evitar que, por exemplo, se em uma determinada linha, forem necessários 4 viagens entre às 06h00 e às 07h00, estas viagens não possuam início no mesmo horário inicial, resultando em um melhor atendimento da demanda de passageiros.

Observe que, alguns pesos são multiplicados por minutos (tempo de viagem morta) e outros por unidades (número de veículos utilizados). Quando a restrição associada a um peso for referente ao tempo, o tempo sempre será medido em minutos. Por exemplo, considere uma solução s com as seguintes características:

- 80 viagens;
- 10 veículos;
- 18 tripulações;

- duas viagens da mesma linha com o mesmo horário inicial;
- uma tripulação com um intervalo de descanso de apenas 5 minutos (faltam 15 minutos);
- 8 horas extras;
- 30 minutos de hora super extra;
- 5 minutos de sobreposição de um veículo;
- 6 veículos com dupla pegada;
- 5 tripulações com dupla pegada;
- 2 horas e 30 minutos de viagem morta;
- 3 horas de tempo de terminal (veículo ocioso no terminal).

A função de avaliação seria calculada da seguinte forma:

$$FA(s) = FA_{PPH}(s) + FA_{PPV}(s) + FA_{PPT}(s) \quad (4.13)$$

$$FA_{PPH}(s) = FRE_{PPH}(s) + FRNE_{PPH}(s) \quad (4.14)$$

$$FA_{PPV}(s) = FRE_{PPV}(s) + FRNE_{PPV}(s) \quad (4.15)$$

$$FA_{PPT}(s) = FRE_{PPT}(s) + FRNE_{PPT}(s) \quad (4.16)$$

A parcela $FRNE_{PPH}(s)$, expressão (4.14), seria composta pelo número de viagens realizadas, 80 viagens. Já a parcela $FRNE_{PPV}(s)$, expressão (4.15), seria composta pelo número de veículos utilizados (10 veículos), o tempo de viagem morta (150 minutos) e o tempo de terminal (180 minutos). A parcela $FRNE_{PPT}(s)$, expressão (4.16), consideraria o número de tripulações utilizadas (18 tripulações), o tempo de hora extra (480 minutos) e o tempo de hora super extra (30 minutos). Estas parcelas correspondem às restrições não essenciais e, portanto, representam atributos de qualidade da solução.

A parcela $FRE_{PPH}(s)$, expressão (4.14), seria composta, somente, pelo número de viagens da mesma linha com o mesmo horário inicial. A parcela $FRE_{PPV}(s)$, expressão (4.15), seria composta, somente, pelo tempo de sobreposição (5 minutos) e a $FRE_{PPT}(s)$, expressão (4.16), consideraria o tempo ausente de folga (15 minutos) e o excesso de duplas pegadas (2 duplas pegadas além do permitido que é de 15% do total de 18 tripulações, ou seja, 2 *duplas pegada*). Estas parcelas representam as restrições essenciais e inviabilidades, ou seja, caso

ambas as parcelas não tenham os seus valores iguais a zero, a solução s será considerada inviável. Substituindo os valores referentes à solução *s* e os pesos da Tabela 4.4, na função de avaliação, obtemos:

$$FRE_{PPH}(s) = 1.000 * 1 = 1.000 \quad (4.19)$$

$$FRE_{PPV}(s) = 80 * 5 = 400 \quad (4.20)$$

$$FRE_{PPT}(s) = 80 * 15 + 80 * 2 = 1.360 \quad (4.21)$$

$$FRNE_{PPH}(s) = 0 \quad (4.22)$$

$$FRNE_{PPV}(s) = 1.000 * 10 + 2 * 150 + 1 * 180 = 10.480 \quad (4.23)$$

$$FRNE_{PPT}(s) = 1.000 * 18 + 2 * 480 + 10 * 30 = 19.260 \quad (4.24)$$

$$FA_{PPH}(s) = 1.000 + 0 = 1.000 \quad (4.25)$$

$$FA_{PPV}(s) = 400 + 10.480 = 10.880 \quad (4.26)$$

$$FA_{PPT}(s) = 1.360 + 19.260 = 20.620 \quad (4.27)$$

$$FA(s) = 1.000 + 10.880 + 20.620 = 32.500 \quad (4.28)$$

As parcelas (4.19) à (4.21), respectivamente, $FRE_{PPH}(s)$, $FRE_{PPV}(s)$ e $FRE_{PPT}(s)$, representam as penalizações referentes às restrições essenciais, não atendidas pela solução s. Dessa forma, a solução s é uma solução inviável, pois as parcelas (4.19), (4.20) e (4.21) possuem o valor diferente de zero e um custo de 32.500 unidades (expressão (4.28)), ou seja, as restrições essenciais não são atendidas pela solução s.

4.2.3. Estruturas de Vizinhança

Conforme observado no pseudocódigo da meta-heurística VNS (Figura 2.4) ou no pseudocódigo da meta-heurística VNS-LR (Figura 2.6) ou no pseudocódigo da meta-heurística VNDS (Figura 2.7), dada uma solução, é necessário gerar uma solução vizinha, cuja qualidade será avaliada pela função de avaliação. As três meta-heurísticas propostas neste trabalho para o PPHVT utilizam a mesma estrutura de vizinhança. A estrutura

de vizinhança proposta para o PPHVT, neste trabalho, é um aperfeiçoamento da estrutura de vizinhança proposta em Reis (2008). A estrutura de vizinhança utilizada no presente trabalho é:

- Movimento de realocação de uma tarefa para o PPT após um movimento inicial aleatório envolvendo o PPT;
- Movimento de troca de duas tarefas para o PPT após um movimento inicial aleatório envolvendo o PPT;
- Movimento de realocação de duas tarefas para o PPT após um movimento inicial aleatório envolvendo o PPT;
- Movimento de realocação de uma viagem para o PPV após um movimento inicial aleatório envolvendo o PPV;
- Movimento de troca de duas viagens para o PPV após um movimento inicial aleatório envolvendo o PPV;
- Movimento de realocação de duas viagens para o PPV após um movimento inicial aleatório envolvendo o PPV;
- Movimento de atraso ou de adiantamento de até 5 minutos no horário de saída de uma viagem e correção do PPV e do PPT;
- Movimento de atraso ou de adiantamento de até 5 minutos no horário de saída de duas viagens e correção do PPV e do PPT.

Neste trabalho, um movimento é definido como uma modificação realizada na solução corrente. Por exemplo: retirar uma tarefa de uma tripulação e atribuir esta tarefa a uma tripulação diferente ou trocar uma viagem de um veículo com uma viagem de outro veículo ou atrasar o início de uma viagem em 3 minutos.

Para melhor entendimento do funcionamento da estrutura de vizinhança proposta, considere uma solução s. Note que uma solução vizinha s', pode ser obtida por meio de oito tipos de movimentos diferentes: movimento de realocação de uma tarefa para o PPT, movimento de troca de duas tarefas para o PPT, movimento de realocação de duas tarefas para o PPT, movimento de realocação de uma viagem para o PPV, movimento de troca de duas viagens para o PPV, movimento de realocação de duas viagens para o PPV, movimento de atraso no horário de início da viagem e movimento de adiantamento no horário de início da

viagem.

Qualquer um destes movimentos, pode gerar uma solução inviável. Cabe ressaltar, que esta característica é mantida para que o VNS implementado, possua uma liberdade maior para explorar o espaço de soluções do PPHVT do que se fosse aceito somente soluções viáveis. Uma solução vizinha s' é uma solução que está na vizinhança de uma solução corrente s e pode ser obtida após realizar um dos movimentos existentes na estrutura de vizinhança.

O movimento de realocação de uma tarefa para o PPT, consiste em realocar uma tarefa de uma tripulação a outra tripulação. Este movimento é apresentado na Figura 4.10. O procedimento para a realização de um movimento de realocação de uma tarefa para o PPT após um movimento inicial aleatório envolvendo o PPT consiste em aleatoriamente escolher uma tripulação que cederá uma tarefa. Em seguida aleatoriamente escolhe-se uma tripulação que receberá a tarefa da primeira tripulação. O passo seguinte é aleatoriamente escolher uma tarefa da primeira tripulação e atribuí-la à segunda tripulação. Após este movimento inicial aleatório é realizada uma busca local visando encontrar a melhor solução vizinha que pode ser obtida a partir da solução corrente realocando somente uma tarefa entre duas tripulações.

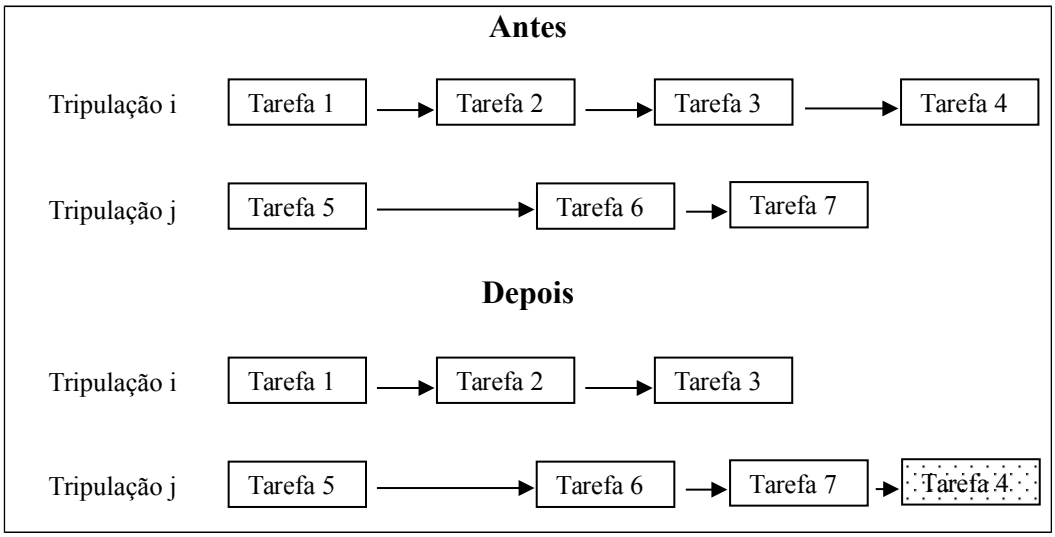

Figura 4.10 – Movimento de Realocação de uma tarefa para o PPT.

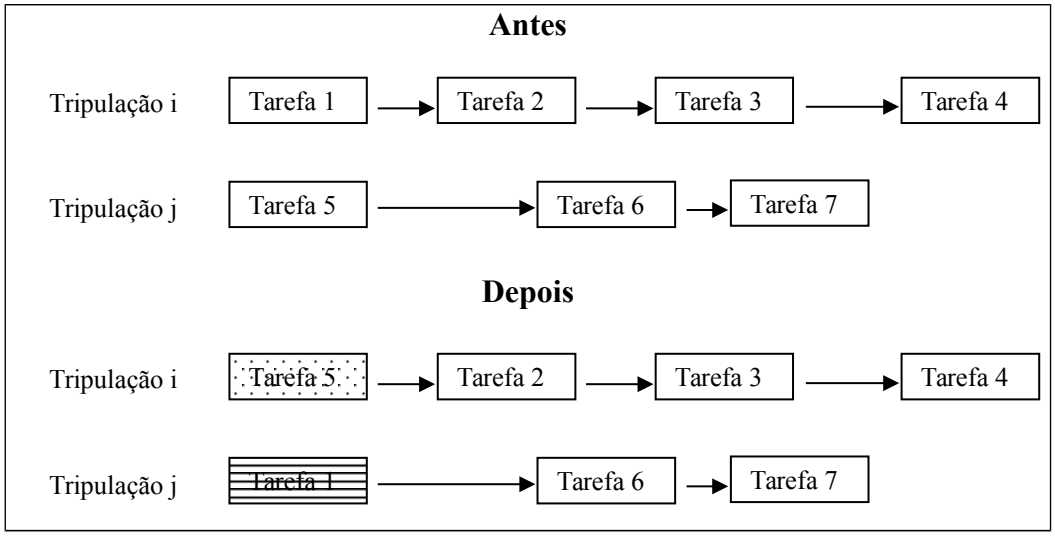

Figura 4.11 – Movimento de Troca para o PPT.

O movimento de troca para o PPT é realizado, trocando uma tarefa de uma tripulação com outra tarefa de outra tripulação. Esta estrutura de vizinhança é ilustrada na Figura 4.11. A sequência de operações necessárias, para se realizar um movimento de troca para o PPT é iniciada por meio da escolha aleatória de uma tripulação e, aleatoriamente, escolhe-se uma tarefa desta tripulação. Em seguida, escolhe-se, aleatoriamente, uma segunda tripulação e seleciona-se uma tarefa da segunda tripulação, cujo horário de início ou de término esteja compreendido no intervalo de tempo da tarefa selecionada da primeira tripulação. O último passo do movimento de troca é a troca da tarefa selecionada, da primeira tripulação pela tarefa da segunda tripulação.

O movimento de realocação de duas tarefas para o PPT, consiste em realocar duas tarefas de uma tripulação a outra tripulação. Este movimento é apresentado na Figura 4.12.

O procedimento para a realização de um movimento de realocação de duas tarefas para o PPT após um movimento inicial aleatório envolvendo o PPT consiste em aleatoriamente escolher uma tripulação que cederá duas tarefas. Em seguida aleatoriamente escolhe-se uma tripulação que receberá as tarefas da primeira tripulação. O passo seguinte é aleatoriamente escolher duas tarefas da primeira tripulação e atribuí-la à segunda tripulação. Após este movimento inicial aleatório é realizada uma busca local visando encontrar a melhor solução vizinha que pode ser obtida a partir da solução corrente realocando duas tarefas entre duas tripulações.

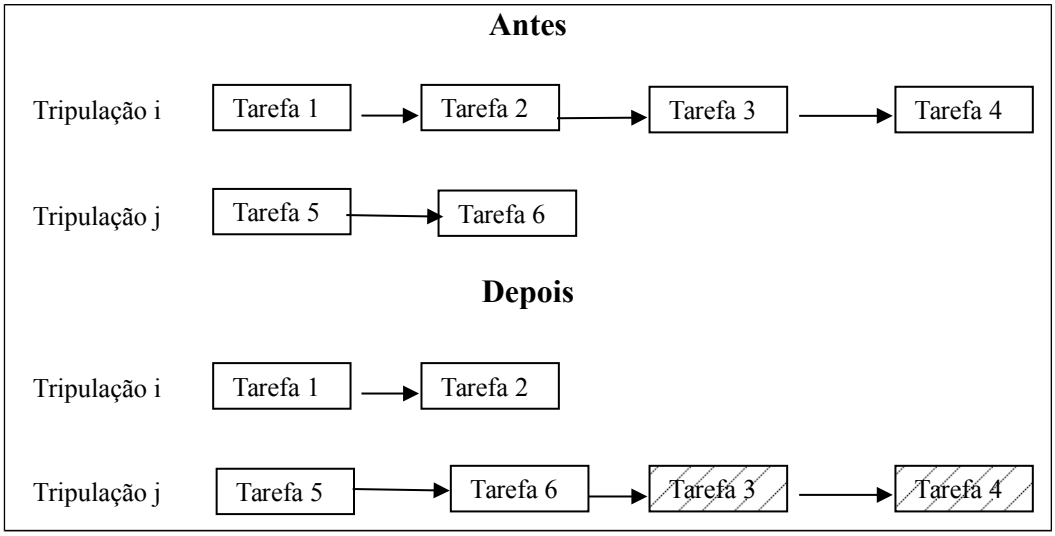

Figura 4.12 – Movimento de Realocação de 2 tarefas para o PPT.

O movimento de realocação de uma viagem para o PPV, funciona de forma análoga ao movimento de realocação de uma tarefa para o PPT. A diferença é que ao invés de uma tripulação é selecionado um veículo e, no lugar de realocar-se uma tarefa é realocada uma viagem. A Figura 4.13 mostra este movimento.

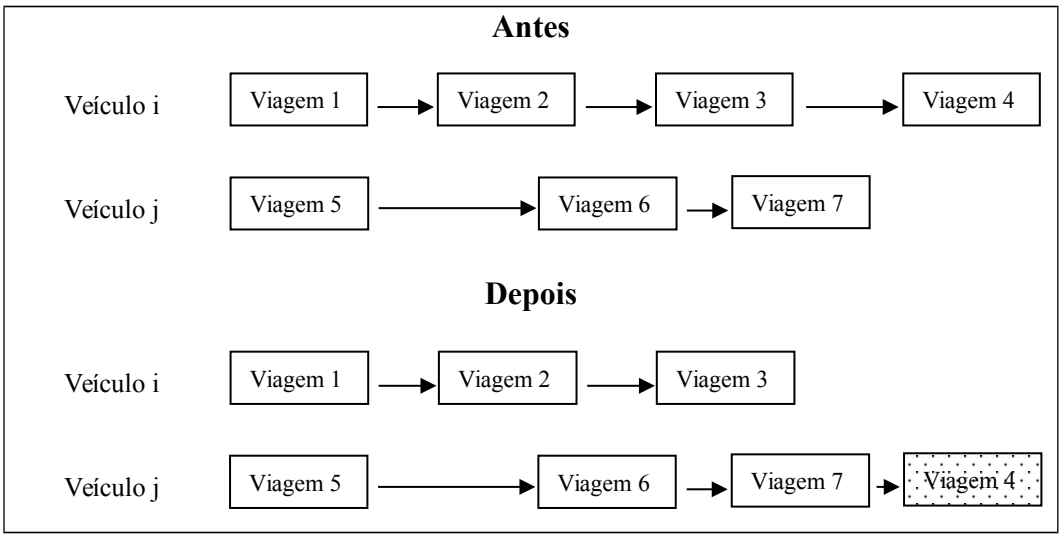

Figura 4.13 – Movimento de Realocação de uma viagem para o PPV.

O movimento de realocação de uma viagem para o PPV é mais complexo do que o movimento de realocação de uma tarefa para o PPT, pois uma alteração na programação dos veículos pode afetar o agrupamento das viagens em tarefas e consequentemente modificar a

solução do PPT. Desta forma, é necessário refazer a programação das tripulações que tiverem as suas tarefas modificadas após uma mudança no PPV.

O movimento de troca para o PPV, consiste em trocar uma viagem de um veículo pela viagem de outro veículo, de forma análoga, ao movimento de troca para o PPT. Este movimento é exposto na Figura 4.14.

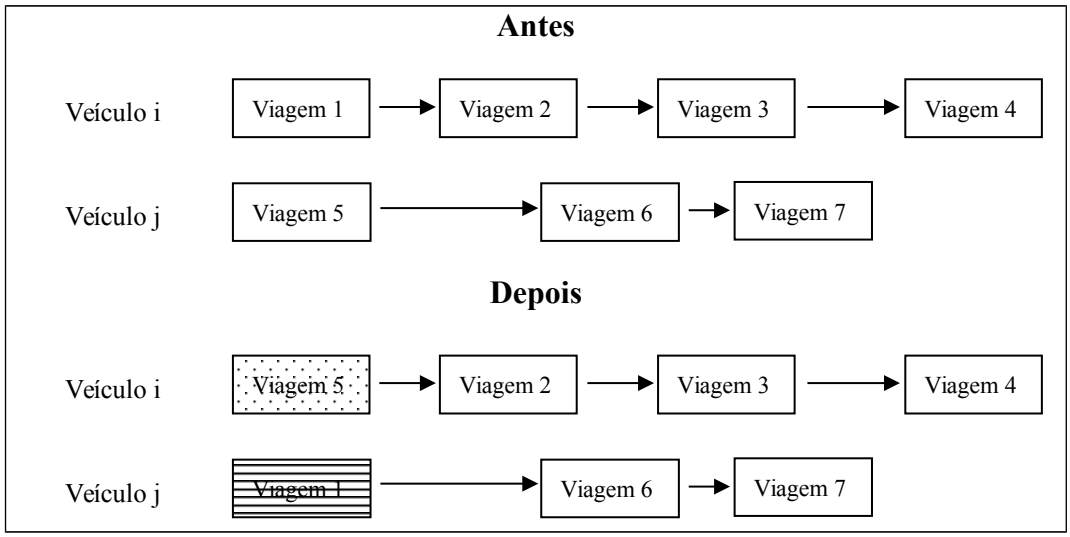

Figura 4.14 – Movimento de Troca para o PPV.

Os movimentos relacionados aos veículos são mais complexos, em comparação aos movimentos relacionados aos tripulantes. Isto ocorre devido ao fato de que, após uma mudança em uma ou mais viagens, podem ocorrer mudanças nas tarefas e consequentemente, nos tripulantes que irão executá-las.

O movimento de realocação de duas viagens para o PPV, funciona de forma análoga ao movimento de realocação de duas tarefas para o PPT. A diferença é que ao invés de uma tripulação é selecionado um veículo e, no lugar de realocar-se duas tarefas são realocadas duas viagens. A Figura 4.15 mostra este movimento.

O movimento de atraso no horário de início da viagem consiste em selecionar uma viagem de uma linha e atrasar o horário de início da viagem em um intervalo de até 5 minutos. Este movimento é apresentado na Figura 4.16.

Observe que o atraso de dois minutos, no horário de início da viagem destacada na Figura 4.16, permite que o veículo número 2 execute esta viagem, tornando desnecessária a utilização do veículo número 5.

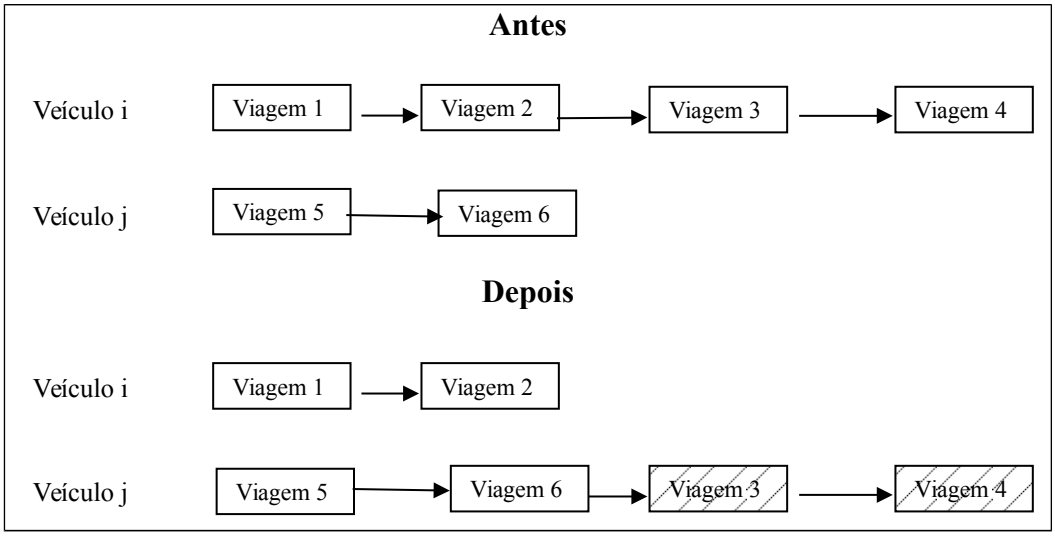

Figura 4.15 – Movimento de Realocação de 2 viagens para o PPV.

Antes				**Depois**			
Linha	N. Veículo	Início	Final	Linha	N. Veículo	Início	Final
0001	0001	05h00	05h55	0001	0001	05h00	05h55
0001	0002	05h30	06h25	0001	0002	05h30	06h25
0001	0001	06h00	07h08	0001	0001	06h00	07h08
0001	0003	06h08	07h16	0001	0003	06h08	07h16
0001	0004	06h16	07h24	0001	0004	06h16	07h24
0001	0005	06h24	07h32	0001	**0002**	**06h26**	**07h34**

Figura 4.16 – Movimento de Atraso no Horário de Início de uma viagem.

O movimento de adiantamento, no horário de início da viagem, consiste em selecionar, aleatoriamente, uma viagem de uma linha e adiantar o horário de início da viagem em até 5 minutos. Este movimento é apresentado na Figura 4.17.

Antes				**Depois**			
Linha	N. Veículo	Início	Final	Linha	N. Veículo	Início	Final
0001	0001	05h00	05h55	0001	0001	05h00	05h55
0001	0002	05h30	06h25	0001	0002	05h30	06h25
0001	0001	06h00	07h08	0001	0001	06h00	07h08
0001	0003	06h10	07h18	0001	0003	06h10	07h18
0001	0004	06h20	07h28	0001	0004	06h20	07h28
0001	0002	06h30	07h38	0001	0002	**06h27**	**07h35**

Figura 4.17 – Movimento de Adiantamento no Horário de Início de uma viagem.

Observe que o adiantamento de três minutos, no horário de início da viagem, destacada na Figura 4.17, permite que o veículo número 2 reduza o seu tempo ocioso no terminal, de 5 minutos para 2 minutos.

Os movimentos de adiantamento ou de atraso, no horário de início de uma viagem, são os movimentos mais complexos, propostos neste trabalho, para a exploração da vizinhança do PPHVT. Cabe ressaltar que, após uma modificação no horário de início de uma viagem, todos os veículos e tripulações afetados por esta modificação, devem ter a sua programação corrigida, pois uma viagem pode mudar de veículo, tornando inclusive um dos veículos desnecessários ou pode reduzir o tempo ocioso de um veículo ou eliminar um intervalo de descanso de uma tripulação, tornando a solução inviável. A correção da programação dos veículos e dos tripulantes é realizada, somente, na linha modificada, fazendo uso da heurística construtiva proposta na solução inicial (Figura 4.8).

O movimento inicial aleatório para o PPT, consiste em sortear um tipo de movimento (realocação ou troca), uma tripulação e uma tarefa desta tripulação e realizar o movimento (realocação ou troca).

Já o movimento inicial aleatório para o PPV, consiste em sortear um tipo de movimento (realocação ou troca), um veículo e uma viagem deste veículo e realizar o movimento (realocação ou troca). Cabe ressaltar, que as meta-heurísticas propostas para o PPHVT (VNS, Figura 2.4; VNS-LR, Figura 2.6; VNDS, Figura 2.7), somente, permitem o avanço para o movimento da estrutura de vizinhança seguinte, quando não for possível realizar um movimento anterior que tenha obtido uma solução de melhora.

4.2.4. Busca Local

Uma vez encontrada uma nova solução viável s' vizinha da solução corrente, é realizada uma busca local com a finalidade de tentar melhorar s'. O mesmo procedimento de busca local é utilizado pelas meta-heurísticas VNS e VNS-LR; já a meta-heurística VNDS utiliza um outro procedimento de busca local que será apresentado mais adiante na seção 4.2.5.

O procedimento de busca local é realizado após um movimento inicial aleatório e o objetivo é encontrar a melhor solução pertencente à vizinhança da solução vizinha à solução

corrente. O procedimento de busca local varia conforme o movimento da estrutura de vizinhança que esteja sendo explorado.

Mais especificamente, o procedimento de busca local é realizado da seguinte maneira: suponha que a solução corrente seja s e o movimento atualmente selecionado pela estrutura de vizinhança seja $N^1(s)$, ou seja, o movimento de realocação de uma tarefa para o PPT após um movimento inicial aleatório envolvendo o PPT. Inicialmente seleciona-se aleatoriamente duas tripulações e uma tarefa da primeira tripulação, em seguida realoca-se esta tarefa da primeira tripulação para a segunda tripulação. Esta nova solução é chamada de s' e é uma solução vizinha da solução corrente s.

Em seguida é escolhida aleatoriamente uma tripulação da solução s' e uma tarefa desta tripulação. O procedimento de busca local tenta inserir esta tarefa em cada uma das outras tripulações (todas as soluções vizinhas de s' dentro da estrutura de vizinhança $N^1(s)$), a inserção que resultar em uma nova solução s'' com custo inferior ao custo da solução s ($f(s'')<f(s)$) será realizada e a solução corrente passará a ser s''. Caso não seja encontrada nenhuma solução s'' que atenda a $f(s'')<f(s)$, a busca local retornará a solução s e a meta-heurística irá avançar para a estrutura de vizinhança $N^2(s)$.

O procedimento de busca local, no caso de ser selecionadas as estruturas de vizinhança $N^2(s)$ (Movimento de troca de duas tarefas para o PPT após um movimento inicial aleatório envolvendo o PPT) ou $N^3(s)$ (Movimento de realocação de duas tarefas para o PPT após um movimento inicial aleatório envolvendo o PPT), é análogo ao procedimento de busca local quando a estrutura de vizinhança utilizada é a $N^1(s)$. A diferença é a realização de uma troca de duas tarefas ao invés de realocação de uma tarefa (no caso da estrutura $N^2(s)$) ou a realização de uma realocação de duas tarefas no lugar de realocar uma tarefa (no caso da estrutura $N^3(s)$).

Suponha que a solução corrente seja s e o movimento atualmente selecionado pela estrutura de vizinhança seja $N^4(s)$, ou seja, o movimento de realocação de uma viagem para o PPV após um movimento inicial aleatório envolvendo o PPV. Inicialmente seleciona-se aleatoriamente dois veículos e uma viagem do primeiro veículo, em seguida realoca-se esta viagem do primeiro veículo para o segundo veículo.

Esta modificação na programação dos veículos pode modificar o agrupamento de

viagens que formará o conjunto de tarefas. Neste caso o conjunto de tarefas será ajustado para esta nova solução e serão feitas as modificações necessárias na programação da tripulação. Esta nova solução é chamada de s' e é uma solução vizinha da solução corrente s.

Em seguida é escolhida aleatoriamente um veículo da solução s' e uma viagem deste veículo. O procedimento de busca local tenta inserir esta viagem em cada um dos demais veículos (todas as soluções vizinhas de s' dentro da estrutura de vizinhança $N^4(s)$). Esta modificação na programação dos veículos pode modificar o agrupamento de viagens que formará o conjunto de tarefas.

Neste caso o conjunto de tarefas será ajustado para esta nova solução e serão feitas as modificações necessárias na programação da tripulação. A inserção que resultar em uma nova solução s'' com custo inferior ao custo da solução s ($f(s'')<f(s)$) será realizada e a solução corrente passará a ser s''. Caso não seja encontrada nenhuma solução s'' que atenda a $f(s'')<f(s)$, a busca local retornará a solução s e a meta-heurística irá avançar para a estrutura de vizinhança $N^5(s)$.

O procedimento de busca local quando são selecionadas as estruturas de vizinhança $N^5(s)$ (Movimento de troca de duas viagens para o PPV após um movimento inicial aleatório envolvendo o PPV) ou $N^6(s)$ (Movimento de realocação de duas viagens para o PPV após um movimento inicial aleatório envolvendo o PPV) é análogo ao procedimento de busca local quando a estrutura de vizinhança utilizada é a $N^4(s)$. A diferença é a realização de uma troca de duas viagens ao invés de realocação de uma viagem (no caso da estrutura $N^5(s)$) ou a realização de uma realocação de duas viagens no lugar de realocar uma viagem (no caso da estrutura $N^6(s)$).

Para exemplificar o funcionamento do procedimento de busca local quando a estrutura de vizinhança seja $N^7(s)$, ou seja, movimento de atraso ou de adiantamento de até 5 minutos no horário de saída de uma viagem e correção do PPV e do PPT, suponha que a solução corrente seja s. Neste caso seleciona-se aleatoriamente fazer um atraso ou um adiantamento e escolha aleatoriamente um tempo entre 1 e 5 minutos para o atraso ou adiantamento.

Suponha que foi escolhida a opção de atrasar o horário de início de uma viagem em 3 minutos. Para cada um das viagens do problema, o horário de início de uma viagem será

atrasado em 3 minutos e a heurística construtiva da solução inicial (Figura 4.8) será utilizada para reconstruir a solução do PPV e do PPT somente corrigindo os veículos e as tripulações afetadas por esta mudança.

Esta solução com uma viagem com horário de início atrasado em 3 minutos será denominada s'. A modificação no horário de uma viagem que resultar em uma nova solução s' com custo inferior ao custo da solução s ($f(s') < f(s)$) será realizada e a solução corrente passará a ser s'. Caso não seja encontrada nenhuma solução s' que atenda a $f(s') < f(s)$, a busca local retornará a solução s e a meta-heurística irá avançar para a estrutura de vizinhança $N^8(s)$.

O procedimento de busca local quando é selecionada a estrutura de vizinhança $N^8(s)$ (Movimento de atraso ou de adiantamento de até 5 minutos no horário de saída de duas viagens e correção do PPV e do PPT) é análogo ao procedimento de busca local quando a estrutura de vizinhança utilizada é a $N^7(s)$. A diferença é a alteração (atraso ou adiantamento) no horário de início de duas viagens ao invés de uma viagem.

4.2.5. Busca Local Decomposta

Nesta seção é apresentada o procedimento de busca local decomposta utilizado pela meta-heurística VNDS. A principal diferença entre o procedimento de busca local utilizado pelas meta-heurísticas VNS e VNS-LR e o procedimento de busca local decomposta utilizado pela meta-heurística VNDS é a decomposição do procedimento de busca local em duas etapas.

Após ser gerada uma solução s' aleatória dentro da estrutura de vizinhança $N^k(s)$, a primeira etapa da busca local decomposta é uma busca realizada na vizinhança de s' e utilizando a estrutura de vizinhança $N^k(s)$. A segunda etapa da busca local é uma busca realizada dentro da vizinhança de s' e além da estrutura de vizinhança $N^k(s)$ que está sendo analisada.

O objetivo do procedimento de busca local decomposta é utilizar-se um movimento da estrutura de vizinhança $N^k(s)$ utilizado pelo procedimento de busca local (VNS e VNS-LR, seção 4.2.4) mas considerando a possibilidade de explorar outras tarefas ou

outras viagens para serem movimentadas.

Mais especificamente, o procedimento de busca local decomposta é realizado da seguinte maneira: suponha que a solução corrente seja s e o movimento atualmente selecionado pela estrutura de vizinhança seja $N^1(s)$, ou seja, o movimento de realocação de uma tarefa para o PPT após um movimento inicial aleatório envolvendo o PPT. Inicialmente seleciona-se aleatoriamente duas tripulações e uma tarefa da primeira tripulação, em seguida realoca-se esta tarefa da primeira tripulação para a segunda tripulação. Esta nova solução é chamada de s' e é uma solução vizinha da solução corrente s.

A primeira etapa do procedimento de busca local decomposta é idêntico ao procedimento de busca local do VNS e do VNS-LR. Na segunda etapa do procedimento de busca local decomposta é escolhida aleatoriamente uma tripulação da solução s' e o procedimento tenta inserir cada uma das tarefas da tripulação escolhida em cada uma das outras tripulações (todas as soluções vizinhas de s' dentro da estrutura de vizinhança $N^1(s)$), a inserção que resultar em uma nova solução s'' com custo inferior ao custo da solução s ($f(s'')<f(s)$) será realizada e a solução corrente passará a ser s''. Caso não seja encontrada nenhuma solução s'' que atenda a $f(s'')<f(s)$, a busca local decomposta retornará a solução s e a meta-heurística irá avançar para a estrutura de vizinhança $N^2(s)$.

A diferença entre o procedimento de busca local (seção 4.2.4) e este procedimento de busca local decomposta é a exploração das soluções vizinhas à solução s' que podem ser alcançadas através da realocação de qualquer uma das tarefas da tripulação escolhida aleatoriamente da solução s'. Esta liberdade adicional para o procedimento de busca local decomposta explorar dentro e fora da estrutura de vizinhança possibilita uma exploração mais ampla do espaço de soluções viáveis.

Em resumo, o procedimento de busca local decomposta difere do procedimento de busca local da seguinte forma em cada uma das estruturas de vizinhança:

- $N^1(s)$ → Movimento de realocação de uma tarefa para o PPT após um movimento inicial aleatório envolvendo o PPT: durante a exploração da vizinhança da solução s' é possível analisar a realocação de qualquer uma das tarefas da tripulação escolhida;
- $N^2(s)$ → Movimento de troca de duas tarefas para o PPT após um movimento

inicial aleatório envolvendo o PPT: durante a exploração da vizinhança da solução s' é possível analisar a troca de qualquer par de tarefas das tripulações escolhidas;

- $N^3(s) \rightarrow$ Movimento de realocação de duas tarefas para o PPT após um movimento inicial aleatório envolvendo o PPT: durante a exploração da vizinhança da solução s' é possível analisar a realocação de duas tarefas quaisquer da tripulação escolhida;
- $N^4(s) \rightarrow$ Movimento de realocação de uma viagem para o PPV após um movimento inicial aleatório envolvendo o PPV: durante a exploração da vizinhança da solução s' é possível analisar a realocação de qualquer uma das viagens do bloco de veículo escolhido;
- $N^5(s) \rightarrow$ Movimento de troca de duas viagens para o PPV após um movimento inicial aleatório envolvendo o PPV: durante a exploração da vizinhança da solução s' é possível analisar a troca de qualquer par de viagens dos blocos de veículos escolhidos;
- $N^6(s) \rightarrow$ Movimento de realocação de duas viagens para o PPV após um movimento inicial aleatório envolvendo o PPV: durante a exploração da vizinhança da solução s' é possível analisar a realocação de duas viagens quaisquer do bloco de veículo escolhido;
- $N^7(s) \rightarrow$ Movimento de atraso ou de adiantamento de até 5 minutos no horário de saída de uma viagem e correção do PPV e do PPT: não há diferença entre os dois procedimentos de busca local;
- $N^8(s) \rightarrow$ Movimento de atraso ou de adiantamento de até 5 minutos no horário de saída de duas viagens e correção do PPV e do PPT: não há diferença entre os dois procedimentos de busca local.

4.3. EXPERIMENTOS COMPUTACIONAIS

As meta-heurísticas baseadas em busca em vizinhança variável (VNS, VNS-LR e VNDS) aqui propostas a fim de resolver o PPHVT foram testadas com 3 (três) diferentes

conjuntos de dados de entrada. O primeiro conjunto de dados de entrada são correspondentes às instâncias geradas aleatoriamente, com base nas características dos dados utilizados por Reis (2008); este conjunto possui poucas viagens e foi utilizado para efetuar uma comparação entre os métodos exatos (seção 4.1) e as meta-heurísticas (seção 4.2). O segundo conjunto de dados são os dados reais utilizados por Reis (2008).

Os dados utilizados por Reis (2008) são dados reais fornecidos por uma empresa de transporte coletivo por ônibus urbano da cidade de Belo Horizonte; esses dados são as viagens planejadas para serem executadas em 3 (três) tipos diferentes de dia: dias úteis, domingo ou sábado. Os ônibus podem iniciar a jornada diária a partir de 8 (oito) garagens diferentes conforme a linha, isto totaliza 24 instâncias de testes com o número de viagens variando entre 53 e 1.038 viagens diárias. Visando a obter instâncias menores para teste, escolheu-se a maior instância com 1.038 viagens diárias referentes a um dia útil e foram selecionados aleatoriamente entre 10 e 40 viagens de uma mesma linha para gerar 4 (quatro) instâncias de testes pequenas para validar os métodos exatos implementados.

O terceiro conjunto de dados são dados reais de uma empresa de transporte coletivo por ônibus urbano da cidade de Belo Horizonte. Os dados fornecidos pela empresa são: o número de passageiros transportados e o horário de cada viagem, correspondem a operação real durante um dia útil (uma segunda-feira). Os testes foram realizados em uma única garagem, ou seja, todos os veículos iniciam a sua jornada diária e encerram a sua jornada no mesmo ponto.

A empresa realizou neste dia de operação 26.779 viagens e para realizar estas viagens utilizou 2.705 veículos e transportou 1.420.965 passageiros. Com base nos dados fornecidos pela empresa, é possível estimar que será necessário 3.923 tripulações para operar os 2.705 veículos. O número de tripulações necessários para executar a solução da empresa foi estimado dividindo o tempo efetivo de viagem dos veículos pelo tempo normal da jornada de trabalho (07h10 - sete horas e dez minutos), o que resultou em um número médio de 1,45 tripulações por veículo.

Em Reis (2008) foram considerados somente a programação dos veículos e das tripulações e eram considerados 8 problemas diferentes (cada um referente a uma garagem) para cada tipo de dia. O conjunto de problemas relativos aos dias úteis compreende um total de 3.654 viagens diárias, enquanto que os conjuntos relativos ao sábado e domingo correspondem a 2.613 e 1.914 viagens diárias, respectivamente.

Em cada teste realizado, o programa desenvolvido em C++ foi executado cinco vezes e as melhores soluções obtidas estão apresentadas nesta seção. O computador utilizado foi um Core2Quad 2.83GHz com 8GB de RAM e 500GB de HD rodando GNU/Linux Ubuntu 12.04 LTS 64 bits.

4.3.1. ABORDAGEM SEQUENCIAL

A Tabela 4.5 apresenta os resultados obtidos para o primeiro conjunto de dados, o qual foi gerado para criar instâncias de teste de pequeno porte que pudessem ser resolvidos pela enumeração explícita de todas as colunas (EEC). Desta forma, é possível testar o desempenho das meta-heurísticas propostas e verificar se é possível obter a solução ótima para estas instâncias de pequeno porte.

A coluna *Nº de viagens* apresenta a quantidade de viagens em cada instância de testes, a coluna *EEC* apresenta a solução obtida pelo método de enumeração explícita de colunas, a coluna *GC* apresenta a solução obtida pelo método de geração de colunas, as colunas *VNS*, *VNS-LR* e *VNDS* apresentam, respectivamente, as soluções obtidas pelas meta-heurísticas *Variable Neighborhood Search*, *Variable Neighborhood Search* com Lista Restrita e *Variable Neighborhood Decomposition Search.* A linha *FA* apresenta o valor da função de avaliação para cada instância e a linha *Tempo (s)* apresenta o tempo de processamento expresso em segundos.

As linhas *Nº veículos*, *Nº tripulações*, *Viagem Morta* e *Horas Extras* apresentam, respectivamente, o número de veículos, o número de tripulações, o tempo total de viagem morta, isto é, o tempo total de viagem no qual o veículo é deslocado sem passageiros e o tempo total de horas extras. A abordagem sequencial tradicional, isto é, resolver o problema de programação de veículos e depois o problema de programação das tripulações foi utilizada para a obtenção dos resultados presentes na Tabela 4.5.

Os resultados da Tabela 4.5 mostram que as meta-heurísticas propostas (VNS, VNS-LR e VNDS) conseguiram obter a mesma solução ótima encontrada pelos métodos exatos (EEC e GC). Em todas as instâncias de testes presentes na Tabela 4.5 o tempo computacional que as meta-heurísticas propostas utilizaram para encontrar a solução ótima foi igual ou menor do que o tempo computacional utilizado pelos métodos EEC ou GC. Isto

demonstra a qualidade das meta-heurísticas propostas para encontrar soluções de boa qualidade em baixo tempo computacional neste conjunto de instâncias de testes.

Tabela 4.5 – Resultados obtidos para o primeiro conjunto de dados utilizando a abordagem sequencial tradicional.

Nº de viagens	Métodos	EEC	GC	VNS	VNS-LR	VNDS
10	**FA**	5.447	5.447	5.447	5.447	5.447
	Tempo (s)	0,18	0,11	0,10	0,10	0,10
	Nº veículos	2	2	2	2	2
	Nº tripulações	3	3	3	3	3
	Viagem Morta	01:27	01:27	01:27	01:27	01:27
	Horas Extras	00:32	00:32	00:32	00:32	00:32
20	**FA**	15.220	15.220	15.220	15.220	15.220
	Tempo (s)	0,44	0,42	0,28	0,27	0,27
	Nº veículos	5	5	5	5	5
	Nº tripulações	9	9	9	9	9
	Viagem Morta	02:49	02:49	02:49	02:49	02:49
	Horas Extras	02:33	02:33	02:33	02:33	02:33
30	**FA**	20.719	20.719	20.719	20.719	20.719
	Tempo (s)	8,45	0,04	0,02	0,02	0,02
	Nº veículos	7	7	7	7	7
	Nº tripulações	12	12	12	12	12
	Viagem Morta	03:19	03:19	03:19	03:19	03:19
	Horas Extras	02:17	02:17	02:17	02:17	02:17
40	**FA**	28.027	28.027	28.027	28.027	28.027
	Tempo (s)	63,09	4,37	0,49	0,47	0,47
	Nº veículos	9	9	9	9	9
	Nº tripulações	17	17	17	17	17
	Viagem Morta	03:37	03:37	03:37	03:37	03:37
	Horas Extras	01:52	01:52	01:52	01:52	01:52

A Tabela 4.6 apresenta os resultados obtidos para o mesmo conjunto de dados da Tabela 4.5 mas resolvidos utilizando a abordagem sequencial inversa, isto é, resolver a programação das tripulações primeiro e depois resolver a programação dos veículos.

Os resultados da Tabela 4.6 mostram que as três meta-heurísticas propostas conseguiram obter a mesma solução ótima encontrada pelos métodos exatos em um tempo

computacional menor do que o requerido pelos métodos exatos para obter a solução ótima.

Tabela 4.6 – Resultados obtidos para o primeiro conjunto de dados utilizando a abordagem sequencial inversa.

Nº de viagens	Métodos	EEC	GC	VNS	VNS-LR	VNDS
10	**FA**	5.435	5.435	5.435	5.435	5.435
	Tempo (s)	0,41	0,34	0,12	0,12	0,12
	Nº veículos	2	2	2	2	2
	Nº tripulações	3	3	3	3	3
	Viagem Morta	01:52	01:52	01:52	01:52	01:52
	Horas Extras	00:15	00:15	00:15	00:15	00:15
20	**FA**	15.217	15.217	15.217	15.217	15.217
	Tempo (s)	0,94	0,40	0,07	0,07	0,07
	Nº veículos	5	5	5	5	5
	Nº tripulações	9	9	9	9	9
	Viagem Morta	04:17	04:17	04:17	04:17	04:17
	Horas Extras	01:50	01:50	01:50	01:50	01:50
30	**FA**	20.975	20.975	20.975	20.975	20.975
	Tempo (s)	9,17	0,89	0,20	0,19	0,19
	Nº veículos	7	7	7	7	7
	Nº tripulações	12	12	12	12	12
	Viagem Morta	05:48	05:48	05:48	05:48	05:48
	Horas Extras	01:31	01:31	01:31	01:31	01:31
40	**FA**	28.315	28.315	28.315	28.315	28.315
	Tempo (s)	62,10	5,87	0,69	0,67	0,66
	Nº veículos	9	9	9	9	9
	Nº tripulações	17	17	17	17	17
	Viagem Morta	05:15	05:15	05:15	05:15	05:15
	Horas Extras	00:54	00:54	00:54	00:54	00:54

O primeiro conjunto de dados somente contém instâncias de teste pequenas com no máximo 9 veículos e 17 tripulações, quanto menor for a dimensão do problema mais fácil será para as meta-heurísticas encontrarem a solução ótima. As meta-heurísticas encontraram a solução ótima para estes problemas resolvidos através da abordagem sequencial tradicional ou abordagem sequencial inversa. No entanto, isto não garante que as meta-heurísticas encontrarão a solução ótima para o problema integrado da programação de veículos e

tripulações.

Tabela 4.7 – Resultados obtidos para o segundo conjunto de dados utilizando a abordagem sequencial tradicional.

Nº de viagens	Métodos	EEC	GC	VNS	VNS-LR	VNDS
53	**FA**	17.039	17.039	17.039	17.039	17.039
	Tempo (s)	20.527,06	9,40	3,10	2,98	2,94
	Nº veículos	5	5	5	5	5
	Nº tripulações	10	10	10	10	10
	Viagem Morta	03:50	03:50	03:50	03:50	03:50
	Horas Extras	11:11	11:11	11:11	11:11	11:11
69	**FA**	24.174	24.174	24.174	24.174	24.174
	Tempo (s)	252.813,58	213,90	29,59	28,94	28,05
	Nº veículos	7	7	7	7	7
	Nº tripulações	14	14	14	14	14
	Viagem Morta	05:22	05:22	05:22	05:22	05:22
	Horas Extras	08:20	08:20	08:20	08:20	08:20
90	**FA**	ND	33.175	33.175	33.175	33.175
	Tempo (s)	ND	185,72	44,76	43,82	42,56
	Nº veículos	ND	10	10	10	10
	Nº tripulações	ND	20	20	20	20
	Viagem Morta	ND	06:14	06:14	06:14	06:14
	Horas Extras	ND	05:19	05:19	05:19	05:19
98	**FA**	ND	32.526	32.526	32.526	32.526
	Tempo (s)	ND	226,41	62,37	60,90	59,08
	Nº veículos	ND	11	11	11	11
	Nº tripulações	ND	19	19	19	19
	Viagem Morta	ND	08:26	08:26	08:26	08:26
	Horas Extras	ND	02:47	02:47	02:47	02:47

A Tabela 4.7 apresenta um comparativo entre os resultados obtidos pelos métodos exatos (EEC e GC) e pelas meta-heurísticas (VNS, VNS-LR e VNDS) referentes ao segundo conjunto de dados. Este comparativo apresenta as menores instâncias reais de testes fornecidas por uma empresa de transporte coletivo urbano por ônibus que opera na cidade de Belo Horizonte. As instâncias com 53 ou 90 viagens diárias se referem a um dia de domingo, a instância com 69 viagens se refere a um dia de sábado e a instância com 98 viagens se refere

a um dia útil. Devido ao número de viagens diárias do problema ser o principal fator que afeta a capacidade de resolução do PPV e do PPT (quanto maior o número de viagens diárias, maior o tempo necessário para se obter uma solução), as instâncias de teste foram separadas conforme o número de viagens diárias. A abordagem sequencial tradicional (PPV primeiro e PPT depois) foi utilizada para a obtenção destes resultados.

Os resultados obtidos pelos métodos exatos para as instâncias com 53 ou com 69 viagens foram iguais aos resultados obtidos pelas meta-heurísticas. A grande vantagem dos métodos meta-heurísticos sobre os métodos exatos foi o menor tempo computacional para a resolução do PPV+PPT. O método EEC (Enumeração Explícitas das Colunas necessitou de 20.527,06 segundos (aproximadamente 5 horas e 42 minutos) para encontrar a solução ótima para a instância de testes com 53 viagens e 252.813,58 segundos (aproximadamente 2 dias, 22 horas e 13 minutos) para obter solução ótima para a instância de testes com 69 viagens. Durante a resolução das instâncias de testes com 90 ou 98 viagens o método EEC não conseguiu encontrar nenhuma solução devido ao elevado número de colunas geradas e a consequentemente elevada necessidade de memória RAM no computador.

O tempo computacional requerido pelas meta-heurísticas foi menor do que o tempo requerido pelo método de geração e colunas em todas as instâncias de teste. Para as instâncias de teste da Tabela 4.7, a meta-heurística VNDS foi entre 4,92% e 5,27% mais rápida do que a VNS e entre 1,28% e 3,07% mais rápida do que a meta-heurística VNS-LR; a meta-heurística VNS-LR foi entre 2,10% e 3,91% mais rápida do que a meta-heurística VNS.

As mesmas instâncias de teste da Tabela 4.7 foram resolvidas utilizando a abordagem sequencial inversa e nesta abordagem todas as 3 meta-heurísticas propostas obtiveram a mesma solução ótima obtida pela geração de colunas. A tabela com estes resultados encontra-se no Apêndice A (Tabela A.4).

A Tabela 4.8 apresenta os resultados obtidos referentes às instâncias de testes com dados com um número de viagens diárias entre 108 e 172 viagens diárias. As instâncias com 108 ou 161 viagens diárias se referem a um dia de domingo e as instâncias com 121 ou 172 viagens se referem a um dia de sábado. A linha *% acima MS* apresenta o percentual que a solução está acima da melhor solução obtida para aquela instância de teste e a sigla *MS* significa que aquela é a melhor solução encontrada.

Tabela 4.8 – Resultados obtidos para o segundo conjunto de dados utilizando a abordagem sequencial tradicional.

Nº de viagens	Métodos	GC	VNS	VNS-LR	VNDS
108	**FA**	19.315	19.410	19.315	19.315
	Tempo (s)	368,83	68,46	65,55	65,30
	% acima MS	MS	0,49%	MS	MS
	Nº veículos	5	5	5	5
	Nº tripulantes	11	12	11	11
	Viagem Morta	08:20	08:20	08:20	08:20
	Horas Extras	06:52	01:57	06:52	06:52
121	**FA**	19.797	19.797	19.797	19.797
	Tempo (s)	375,65	78,05	76,36	76,10
	% acima MS	MS	MS	MS	MS
	Nº veículos	5	5	5	5
	Nº tripulantes	12	12	12	12
	Viagem Morta	08:20	08:20	08:20	08:20
	Horas Extras	06:11	06:11	06:11	06:11
161	**FA**	30.468	30.855	30.783	30.468
	Tempo (s)	887,49	104,54	100,30	99,54
	% acima MS	MS	1,27%	1,03%	MS
	Nº veículos	8	8	8	8
	Nº tripulantes	18	19	19	18
	Viagem Morta	02:16	02:16	02:16	02:16
	Horas Extras	09:03	04:56	04:56	09:03
172	**FA**	66.438	66.878	66.878	66.438
	Tempo (s)	1.410,86	111,89	107,20	105,55
	% acima MS	MS	0,66%	0,66%	MS
	Nº veículos	25	25	25	25
	Nº tripulantes	35	34	34	35
	Viagem Morta	15:00	15:00	15:00	15:00
	Horas Extras	20:37	28:03	28:03	20:37

O método EEC não conseguiu obter soluções para estes dados devido ao elevado número de viagens diárias e consequentemente um elevado número de colunas geradas (vide Tabela 4.1) e foi retirado do comparativo. Para a instância de teste com 108 viagens diárias foram geradas 204.156 colunas para o PPV e 3.049.501 colunas para o PPT e para instância de teste com 161 viagens diárias foram geradas 682.640 colunas para o PPV e 19.720.001

colunas para o PPT. Este número de colunas geradas está acima da capacidade de processamento do computador utilizado nos testes.

A meta-heurística VNDS conseguiu obter as mesmas soluções ótimas obtida pela geração de colunas em todas as instâncias da Tabela 4.8. O VNS-LR obteve a solução ótima em 2 instâncias e o VNS em somente 1 instância. Nas instâncias em que não conseguiu obter a solução ótima o VNS esteve entre 0,49% e 1,27% acima do valor da solução ótima e o VNS-LR esteve 0,66% e 1,03% acima da solução ótima quando não conseguiu encontrar a solução ótima.

Os resultados apresentados na Tabela 4.8 comprovam que o tempo computacional das meta-heurísticas propostas são menores ao tempo computacional do método de geração de colunas e o VNDS é 0,37% mais rápida do que a meta-heurística VNS-LR e 4,61% mais rápida do que o VNS e a meta-heurística VNS-LR é 4,25% mais rápida do que o VNS.

O método GC encontrou a solução ótima para a instância com 172 viagens após 1.410 segundos (aproximadamente 23 minutos), a meta-heurística VNDS precisou de 105 segundos (menos de 2 minutos) para obter a solução ótima e as meta-heurísticas VNS e VNS-LR não encontraram a solução ótima mas encontraram uma solução 0,66% acima da ótima em menos de 2 minutos.

A Tabela 4.9 apresenta os resultados obtidos para instâncias reais entre 207 e 287 viagens diárias. Estas são as maiores instâncias nas quais foi possível obter a solução ótima através de um método exato (GC). A meta-heurística VNDS não conseguiu obter a solução ótima para as instâncias com 253 viagens ou com 269 viagens, a solução obtida pelo VNDS foi 0,03% e 0,09%, respectivamente, acima do valor da solução ótima.

No entanto, o tempo computacional necessário para o VNDS obter estas soluções, ligeiramente acima das soluções ótimas, foi de 293 segundos (4 minutos e 53 segundos) e 414 segundos (6 minutos e 54 segundos), respectivamente, contra 4.951 segundos (1 hora, 12 minutos e 31 segundos) e 8.479 segundos (2 horas, 21 minutos e 19 segundos) que o método de geração de colunas utilizou para obter a solução ótima. Observe que quanto maior o número de viagens diárias envolvidas na instância de teste maior a diferença de tempo requerido entre a Geração de Colunas e as meta-heurísticas.

Tabela 4.9 – Resultados obtidos para o segundo conjunto de dados utilizando a abordagem sequencial tradicional.

Nº de viagens	Métodos	GC	VNS	VNS-LR	VNDS
207	**FA**	38.289	39.011	39.011	38.289
	Tempo (s)	2.168,93	189,43	183,18	179,59
	% acima MS	MS	1,89%	1,89%	MS
	Nº veículos	14	15	15	14
	Nº tripulantes	18	19	19	18
	Viagem Morta	1400	1480	1480	1400
	Horas Extras	1373	850	850	1373
253	**FA**	59.235	59.251	59.251	59.251
	Tempo (s)	4.951,43	305,85	296,86	293,52
	% acima MS	MS	0,03%	0,03%	0,03%
	Nº veículos	18	18	18	18
	Nº tripulantes	35	34	34	34
	Viagem Morta	421	421	421	421
	Horas Extras	965	1715	1715	1715
269	**FA**	113.376	113.564	113.564	113.480
	Tempo (s)	8.479,49	436,08	422,08	414,19
	% acima MS	MS	0,17%	0,17%	0,09%
	Nº veículos	44	43	43	44
	Nº tripulantes	58	59	59	59
	Viagem Morta	1598	1680	1680	1598
	Horas Extras	1890	1416	1416	1640
287	**FA**	61.552	61.723	61.723	61.552
	Tempo (s)	15.531,24	655,96	635,36	622,54
	% acima MS	MS	0,28%	0,28%	MS
	Nº veículos	18	18	18	18
	Nº tripulantes	38	37	37	38
	Viagem Morta	374	415	415	374
	Horas Extras	1330	1876	1876	1330

Em resumo, considerando as 16 instâncias de testes nas quais foi possível encontrar a solução ótima para o problema sequencial de programação dos veículos e tripulações (veículo primeiro, tripulação depois), em 9 instâncias a meta-heurística VNS obteve a solução ótima, em 10 instâncias a meta-heurística VNS-LR conseguiu encontrar a solução ótima e em 14 instâncias a meta-heurística VNDS encontrou a solução ótima. O

Apêndice A contém as tabelas com os resultados detalhados para todas as instâncias de teste.

4.3.2. ABORDAGEM INTEGRADA

Os resultados apresentados na Tabela 4.10 apresentam os resultados obtidos com a variação do tempo de atraso ou adiantamento, isto é, uma análise da variação dos resultados obtidos quanto ao último movimento da estrutura de vizinhança apresentada na seção 4.2.3:

- Movimento de atraso ou de adiantamento de até 5 minutos no horário de saída de uma viagem e correção do PPV e do PPT;
- Movimento de atraso ou de adiantamento de até 5 minutos no horário de saída de duas viagens e correção do PPV e do PPT.

A meta-heurística VNDS foi utilizada para obter os resultados da Tabela 4.10 e a instância de testes utilizada foi a maior instância com 26.779 viagens diárias. Foi utilizado a meta-heurística VNDS por ser a meta-heurística proposta com o melhor desempenho. Foi utilizada a maior instância, devido a sua elevada complexidade de resolução e a ser a única instância de teste na qual está disponível a demanda de passageiros.

Na primeira coluna da Tabela 4.10, o valor do tempo de atraso ou adiantamento foi 1 minuto e nas demais colunas foi 3, 5, 10 e 15 minutos respectivamente para a segunda, terceira, quarta e quinta colunas. A segunda linha apresenta o valor da função de avaliação calculada através da expressão (4.13) utilizando os parâmetros da Tabela 4.4. A linha *% acima MS* representa o percentual que a solução obtida está acima da melhor solução obtida. As linhas *Nº veículos* e *Nº tripulações* representam, respectivamente, o número de veículos e o número de tripulações. A linha *Viagem Morta* apresenta o tempo total de viagem vazia, isto é, o tempo em que o veículo foi movimentado sem passageiros e, a linha *Horas Extras* apresenta o tempo total de horas extras pagas aos tripulantes.

O melhor resultado apresentado na Tabela 4.10 foi obtido com um tempo de adiantamento ou atraso de 10 minutos nos dois últimos movimentos da estrutura de vizinhança apresentada na seção 4.2.3. O total de veículos utilizados, nesta solução, foi de 2.531 operados por 3.980 tripulações.

Tabela 4.10 – Resultados obtidos com variação do tempo de atraso ou adiantamento - VNDS.

Minutos	1	3	5	10	15
Função de Avaliação	7.240.555	7.321.157	7.232.746	7.137.930	7.264.620
% acima MS	1,44%	2,57%	1,33%	MS	1,77%
Nº veículos	2.657	2.690	2.664	2.531	2.584
Nº tripulações	3.932	3.944	3.943	3.980	3.903
Viagem Morta	2159:10	2124:56	2117:26	2246:38	2264:41
Horas Extras	2106:12	2020:12	2027:22	1475:32	2314:02
Tempo de Terminal	621:37	900:15	610:56	902:43	1139:45
Tempo Ocioso	1706:54	2262:06	1528:34	2101:47	2663:09

A segunda melhor solução foi obtida com um tempo de adiantamento ou atraso de 5 minutos no horário de saída de uma ou duas viagens e utilizou 2.664 veículos e 3.943 tripulações do que a melhor solução obtida.

Como os parâmetros definidos na Tabela 4.4 penalizam de forma mais rígida o número de veículos e o número de tripulações utilizado do que o tempo total de viagem morta e de horas extras, a função de avaliação obtida com um tempo de adiantamento ou atraso de 5 minutos nos dois últimos movimentos da estrutura de vizinhança apresentada na seção 4.2.3 possui um valor 1,33% superior ao valor da solução obtida com um tempo de adiantamento ou atraso de 10 minutos.

No caso da empresa considerar a redução do número de horas extras ou do tempo de viagem morta mais importante do que o número de veículos ou tripulações utilizadas, os pesos da Tabela 4.4 poderiam ser modificados para refletir esta situação. No entanto, o custo de adquirir um novo veículo ou contratar uma nova tripulação costumam ser superiores aos custos de pagamento de horas extras ou ao custo operacional de deslocar-se com um veículo vazio.

Outro aspecto constatado é que um atraso ou adiantamento no horário de início de uma viagem de 10 minutos pode não ser adequado para o passageiro do ônibus. Por exemplo, se o intervalo entre viagens for superior a 2 horas, provavelmente o passageiro não irá se incomodar com um atraso de 10 minutos no horário da viagem. Por outro lado, se o intervalo entre viagens for de 20 minutos provavelmente este atraso de 10 minutos não será adequado.

O ideal seria usar um valor de adiantamento ou atraso variável conforme o

intervalo entre partidas para cada linha; no entanto, isto criaria uma dificuldade de implementação computacional, devido à existência da possibilidade de um veículo trocar de linha durante a execução de sua jornada diária. Visando a reduzir a complexidade computacional da implementação da meta-heurística, optou-se neste trabalho por se manter um atraso ou adiantamento em um valor único para todas as linhas. Como a linha com o menor intervalo entre partidas é de 20 minutos, optou-se por utilizar um atraso ou adiantamento máximo de 5 minutos.

A Tabela 4.11 apresenta os resultados obtidos pela meta-heurística VNDS para a instância de maior porte (26.779 viagens) quanto ao tempo máximo para a execução do VNDS é fixado em 15, 30, 60 ou 120 minutos. Observe que o valor da função de avaliação possui pequenas reduções de valor conforme aumenta o tempo computacional. Dobrar o tempo de execução de 15 minutos para 30 minutos melhora a solução encontrada em somente 0,07%. Aumentar o tempo de execução do VNDS de uma hora para duas horas melhora a solução obtida pelo VNDS em 0,01%. Desta forma, optou-se neste trabalho a limitar o tempo de execução do VNDS em 15 minutos.

Tabela 4.11 – Resultados obtidos pelo VNDS com variação no tempo computacional.

Tempo Computacional	**15**	**30**	**60**	**120**
Função de Avaliação	7.232.746	7.227.186	7.227.141	7.226.157
% acima MS	0,09%	0,01%	0,01%	MS
Nº veículos	2.664	2.664	2.664	2.663
Nº tripulações	3.943	3.944	3.944	3.944
Viagem Morta	2117:26	2117:26	2117:26	2118:06
Horas Extras	2027:22	1972:00	1971:30	1971:16
Tempo de Terminal	610:56	610:56	610:56	610:31
Tempo Ocioso	1528:34	1529:58	1530:13	1530:02

Os resultados da Tabela 4.12 foram obtidos utilizando as meta-heurísticas propostas para resolver o PPHVT com os dados de entrada de instância de maior porte (26.779 viagens). O tempo máximo de processamento foi fixado em 15 minutos, ou seja, esta à melhor solução obtida em 15 minutos para o PPHVT por cada meta-heurística proposta. O tempo de 15 minutos foi escolhido devido a não haver melhoras significativas em um tempo maior de processamento do VNDS.

A meta-heurística VNDS obteve a melhor solução para esta instância de teste com 2.664 veículos e 3.943 tripulações. A solução obtida pelo VNS-LR foi 1,58% pior do que a solução obtida pelo VNDS com 2.705 veículos e 3.923 tripulações. A solução obtida pelo VNS foi 2,60% pior do que a melhor solução pois, apesar de utilizar o mesmo número de veículos e tripulações da solução obtida pelo VNS-LR, utiliza um número maior de viagem morta, hora extra, tempo de terminal e tempo ocioso.

A solução obtida pelo VNDS possui um número maior de tripulações do que as demais meta-heurísticas (20 tripulações a mais); entretanto, este aumento no número de tripulações é compensado pela redução no número de veículos utilizados, viagem morta, hora extra, tempo de terminal e tempo ocioso.

Tabela 4.12 – Comparativo entre resultados das meta-heurísticas.

Meta-heurística	**VNS**	**VNS-LR**	**VNDS**
Função de Avaliação	7.421.153	7.346.793	7.232.746
% acima MS	2,60%	1,58%	MS
Nº veículos	2.705	2.705	2.664
Nº tripulações	3.923	3.923	3.943
Viagem Morta	2469:13	2468:19	2117:26
Horas Extras	2754:36	2249:50	2027:22
Tempo de Terminal	1084:42	918:48	610:56
Tempo Ocioso	1686:53	1624:47	1528:34

A Tabela 4.13 apresenta um comparativo entre os tipos de abordagens: integrada entre tabela de horários, veículos e tripulações (coluna *PPHVT*); integrada entre veículos e tripulações (coluna *PPVT*); sequencial tradicional (coluna *PPV+PPT*); e sequencial inversa (coluna *PPT+PPV*). A meta-heurística utilizada para a obtenção destes resultados foi a VNDS e o tempo de processamento computacional foi limitado a 15 minutos. Os dados de entrada utilizados foram os dados referentes à maior instância de teste (26.779 viagens realizadas em uma segunda-feira).

Na abordagem sequencial tradicional foi resolvida somente a programação dos veículos e após o PPV ter sido resolvido foi resolvido o PPT considerando a solução do PPV como um dado de entrada para o PPT. Nenhuma alteração nos horários de início ou término

das viagens foi permitida.

Durante a resolução do problema utilizando a abordagem sequencial inversa, primeiramente foi resolvida somente a programação das tripulações utilizando a tabela de horários que foi executada nos dados de entrada. Não foi realizada nenhuma alteração na tabela de horários e após a resolução do PPT, foi iniciada a resolução do PPV.

A abordagem integrada entre a programação dos veículos e tripulações (PPVT) foi resolvida sem permitir alterações nos horários das viagens. Nesta abordagem integrada foi permitido a alteração simultânea da programação dos veículos e da programação das tripulações.

Na abordagem integrada entre tabela de horários, veículos e tripulações (PPHVT) foi a única abordagem na qual foi permitido fazer modificações na tabela de horários. Estas modificações no horário de início de uma viagem podem permitir que um mesmo veículo ou uma mesma tripulação executem viagens ou tarefas que possuiriam sobreposição de horários caso o horário de início não tivesse sido alterado.

Tabela 4.13 – Comparativo entre as abordagens sequenciais e a integrada.

Abordagem	**PPHVT**	**PPVT**	**PPV+PPT**	**PPT+PPV**
Função de Avaliação	7.232.746	7.324.887	7.401.153	7.408.579
% acima MS	MS	1,27%	2,33%	2,43%
Nº veículos	2.664	2.673	2.673	2.697
Nº tripulações	3.943	3.923	3.951	3.884
Viagem Morta	2.117:26	2.634:50	2.944:32	2.385:27
Horas Extras	2.027:22	2.365:56	1.831:56	3.312:06
Tempo de Terminal	610:56	755:02	1.518:07	833:21
Tempo Ocioso	1.528:34	1.391:33	1.881:30	1.564:32

Os resultados presentes na Tabela 4.13 revelam que a melhor solução obtida pela meta-heurística VNDS foi utilizando a abordagem integrada entre tabela de horários, veículos e tripulações. O menor número de veículos foi obtidos utilizando a abordagem integrada com a tabela de horários (PPHVT), para obter este resultado foi necessário aumentar o número de tripulações e o tempo ocioso em relação à abordagem integrada (PPVT). A utilização da abordagem sequencial inversa possibilitou encontrar o menor número de tripulações

necessárias mas como consequência foi necessário aumentar o número de horas extras.

A solução obtida para a abordagem sequencial inversa (PPT+PPV) foi 2,43% pior do que a solução para o PPHVT, a abordagem sequencial inversa utilizou 33 veículos a mais (1,24%) e 59 tripulações a menos (1,59%) do que a abordagem integrada com a tabela de horários (PPHVT). A abordagem sequencial inversa também apresentou, em relação à bordagem integrada com a tabela de horários (PPHVT): 12,66% de aumento no tempo de viagem morta; 63,37% de aumento no número de horas extras; 36,41% de aumento no tempo total de terminal dos veículos e 2,35% no tempo ocioso total das tripulações.

A solução obtida para a abordagem sequencial tradicional (PPV+PPT) foi 2,33% pior do que a solução para o PPHVT, a abordagem sequencial tradicional utilizou 9 veículos a mais (0,34%) e 8 tripulações a mais (0,20%) do que a abordagem integrada com a tabela de horários (PPHVT). A abordagem sequencial tradicional também apresentou, em relação à bordagem integrada com a tabela de horários (PPHVT): 39,06% de aumento no tempo de viagem morta; 9,64% de redução no número de horas extras; 148,49% de aumento no tempo total de terminal dos veículos e 23,09% de aumento no tempo ocioso total das tripulações.

A solução obtida para a abordagem integrada (PPVT) foi 1,27% pior do que a solução para o PPHVT, a abordagem integrada utilizou 9 veículos a mais (0,34%) e 20 tripulações a menos (0,51%) do que a abordagem integrada com a tabela de horários (PPHVT). A abordagem integrada também apresentou, em relação à bordagem integrada com a tabela de horários (PPHVT): 24,43% de aumento no tempo de viagem morta; 16,70% de redução no número de horas extras; 23,59% de aumento no tempo total de terminal dos veículos e 8,96% de redução no tempo ocioso total das tripulações.

4.4. CONCLUSÕES

Este capítulo abordou o Problema de Programação Integrada da Tabela de Horários, Veículos e Tripulações (PPHVT) no contexto do transporte coletivo urbano por ônibus. Foram propostas três meta-heurísticas: a meta-heurística de Busca em Vizinhança Variável (VNS) e duas meta-heurísticas baseadas em VNS (VNS-LR e VNDS). Estas meta-heurísticas propostas foram aplicadas a problemas reais e a dados gerados aleatoriamente. Os resultados obtidos pelas meta-heurísticas foram comparados às soluções

ótimas obtidas por métodos exatos (EEC e GC).

Analisando os resultados obtidos para a abordagem sequencial, a Enumeração Explícita das Colunas (EEC) conseguiu encontrar a solução ótima em 6 instâncias de um total de 28 instâncias de teste. Nas instâncias menores que 40 viagens diárias, o tempo computacional foi abaixo de 62 segundos, para uma instância com 69 viagens diárias a EEC necessitou de mais de 70 horas (252.813,58 segundos). Para encontrar a mesma solução, a meta-heurística VNDS necessitou de 28,05 segundos. Os resultados indicam que as três meta-heurísticas propostas conseguiram obter as mesmas soluções ótimas obtidas pela EEC e sempre em um tempo computacional menor. Lembrando que o tempo computacional requerido pela EEC para obter a solução ótima aumentou exponencialmente em relação ao aumento do número de viagens.

Os tempos de processamento das meta-heurísticas cresceram de forma mais rápida do que os tempos de processamento da geração de colunas, isto decorre das meta-heurísticas exigirem maior poder de processamento do computador apesar de consumirem pouca memória RAM. O método de geração de colunas exige uma quantidade menor de processamento do computador mas possui um elevado consumo de memória RAM devido à grande quantidade de colunas geradas.

Essa diferença de tempo de processamento computacional pode não ser significativa do ponto de vista prático devido ao fato da empresa somente necessitar resolver o problema uma única vez por dia.

O método de geração de colunas (GC) conseguiu obter a solução ótima em 16 instâncias de um total de 28 instâncias. Foi possível encontrar a solução ótima somente nas instâncias com no máximo 287 viagens. O tempo computacional requerido pela GC para encontrar a solução ótima aumentou exponencialmente em relação ao aumento do número de viagens, no entanto o aumento foi menor que o aumento do tempo requerido pela EEC.

Entre o método EEC e o GC, a geração de colunas se mostrou mais adequada para encontrar a solução ótima tanto utilizando a abordagem sequencial tradicional quanto a sequencial inversa, para instâncias com até 287 viagens.

Considerando as 16 instâncias de testes nas quais foi possível encontrar a solução ótima para o problema sequencial tradicional de programação dos veículos e tripulações (veículo primeiro, tripulação depois), em 9 instâncias a meta-heurística VNS obteve a solução ótima, em 10 instâncias a meta-heurística VNS-LR conseguiu encontrar a solução ótima e em

14 instâncias a meta-heurística VNDS encontrou a solução ótima.

Para estas mesmas 16 instâncias de testes nas quais foi encontrada a solução ótima utilizando a abordagem sequencial tradicional, também foi possível encontrar a solução ótima utilizando a abordagem sequencial inversa (tripulação primeiro, veículo depois), Considerando as 16 instâncias de testes nas quais foi possível encontrar a solução ótima para o problema sequencial tradicional de programação dos veículos e tripulações (veículo primeiro, tripulação depois), em 10 instâncias a meta-heurística VNS obteve a solução ótima, em 10 instâncias a meta-heurística VNS-LR conseguiu encontrar a solução ótima e em 14 instâncias a meta-heurística VNDS encontrou a solução ótima.

Em todas as instâncias testadas a meta-heurística VNDS obteve um resultado igual ou melhor do que os resultados obtidos pelas meta-heurísticas VNS ou VNS-LR. A meta-heurística VNDS foi entre 0,34% e 2,49% mais rápida do que a meta-heurística VNS-LR e entre 2,50% e 5,66% mais rápida do que a meta-heurística VNS para encontrar a solução de cada instância. A meta-heurística VNS-LR foi entre 2,10% e 4,25% mais rápida do que a meta-heurística VNS.

Contudo a eficácia das três meta-heurísticas propostas para encontrar as soluções ótimas utilizando a abordagem sequencial tradicional ou sequencial inversa não garante a qualidade das soluções encontradas para o problema integrado (PPVT ou PPHVT) mas pode ser utilizado como um indicativo da capacidade destas meta-heurísticas para resolver o problema integrado. Esta qualidade das soluções obtidas pelas meta-heurísticas pode ser expressa tanto em relação ao tempo requerido quanto em relação ao número de veículos e tripulações necessárias.

Os resultados obtidos também mostram que elevar o tempo computacional de processamento resulta em uma melhora nas soluções obtidas, entretanto dobrar o tempo computacional de 15 minutos para 30 minutos resulta em uma redução de 0,08% no valor da função de avaliação da solução obtida e, dobrar o tempo computacional de 30 minutos para 60 minutos resulta em uma redução de 0,01% no valor da função de avaliação da solução obtida.

Esta estabilização da solução por volta de 15 minutos pode ocorrer devido às meta-heurísticas já terem explorado todo o espaço de soluções ao alcance da estrutura de vizinhança proposta ou por ter atingido uma solução ótima do problema.

Ao fixar o tempo computacional em 15 minutos, a meta-heurística VNDS obteve a melhor solução para a maior instância de teste superando o resultado obtido pelas

meta-heurísticas VNS-LR ou VNS. Desta forma, podemos concluir que a meta-heurística VNDS é a meta-heurística mais eficaz e eficiente para resolver o PPHVT ou abordar sequencialmente o PPV e o PPT (na ordem tradicional ou inversa).

A comparação entre as abordagens sequenciais (PPV+PPT ou PPT+PPV) e as integradas (PPVT ou PPHVT) comprovou que as abordagens integradas conseguem obter soluções melhores do que as abordagens sequenciais devido a aproveitarem melhor as interações existentes entre a programação da tabela de horários, dos veículos e das tripulações.

A abordagem sequencial inversa (PPT+PPV) conseguiu obter o menor número de tripulações necessárias, isto se deve ao fato dessa abordagem priorizar a programação das tripulações. Desta forma, sempre que o número total de tripulações for o fator mais importante a abordagem sequencial inversa poderá ter resultados melhores do que as demais abordagens.

Em todas as soluções obtidas, sempre que ocorria uma redução no número de tripulações acontece um aumento no número de horas extras; isto ocorre devido ao tempo total de viagem ter que ser sempre o mesmo e quando o número de tripulações é reduzido, torna-se necessário aumentar o total de horas extras para poder manter o mesmo tempo total de viagens.

Outra observação interessante é que um aumento no número de tripulações resulta no aumento do tempo ocioso, isto ocorre porque, como o tempo total de viagem é constante, quanto maior o número de tripulações, maior será a possibilidade de alguma tripulação ficar ociosa entre duas viagens.

A abordagem integrada PPHVT possibilitou a maior redução no número de veículos, no tempo de viagem morta e no tempo de terminal. Isto ocorreu devido ao adiantamento ou atraso no horário de início das viagens na tabela de horários possibilitando que um mesmo veículo executa-se duas viagens que seriam executadas por veículos diferentes. Esta mudança nos horários de início das viagens também reduziu o tempo de terminal e o deslocamento dos veículos para iniciarem viagens em outros pontos.

5. O PROBLEMA DE AGRUPAMENTO DE ENTREGAS EM VEÍCULOS COM FROTA HETEROGÊNEA

No contexto logístico, a distribuição física de produtos, geralmente, envolve o roteamento de veículos, tendo em vista que o tamanho médio das cargas a serem entregues aos clientes não é suficiente para lotar um veículo, o que acarreta que cada veículo atenda a diversos clientes; consequentemente, busca-se determinar quais entregas devem ser alocadas a cada veículo de uma frota disponível, e qual a sequência de paradas (ou roteiro) de cada veículo, de forma a minimizar o custo total do serviço, geralmente composto da soma ponderada dos custos proporcionais às distâncias percorridas e dos custos fixos dos veículos.

Entretanto, em diversas situações práticas um problema diferente pode surgir, o qual envolve o transporte de várias cargas para um único destino, ou para vários destinos próximos entre si. Desta forma torna-se necessário otimizar o agrupamento de entregas nos veículos, de modo a reduzir a frota necessária e o custo com o frete pago aos transportadores. Este tipo de problema é conhecido como *districting problem* ou problema de zoneamento e é explorado em diversos trabalhos científicos (Novaes *et al*., 2000; da Silva, 2004; Galvão *et al*., 2006, Novaes *et al*., 2009; Novaes, 2007).

Tal situação ocorre em sistemas de distribuição física, tais como a entrega de cimento ou de aço para a construção civil, os quais abrangem inúmeros clientes, localizados em municípios distintos e atendidos a partir de uma única base de origem dos veículos. Em muitos casos, a distância de deslocamento entre a base de origem, onde o veículo é carregado, até um município ou região onde se localizam as entregas a serem realizadas, é muito superior às distâncias de percurso em rota (isto é, entre entregas consecutivas). Características das cargas e dos tempos envolvidos (de entregas e de deslocamento do veículo) podem acarretar um número de entregas não elevado para cada veículo, (em geral não mais do que 10 entregas); consequentemente, de forma geral, é necessário despachar vários veículos a um município, ou a uma região, que compreende municípios vizinhos, próximos entre si, de forma a realizar as entregas para todos os clientes ali localizados (Miura, 2008).

Deve-se ressaltar que, não é aceitável, de um ponto de vista prático, que uma entrega possa ser dividida (ou fracionada) em mais de um veículo (estratégia conhecida como *split deliveries*), buscando facilitar o melhor aproveitamento das capacidades dos veículos.

Isso decorre do fato de que para os clientes, é indesejável receber a mercadoria fracionada em mais de uma entrega, uma vez que isso dificulta e até impede a conferência do que está sendo entregue de acordo com o pedido, em termos dos itens e respectivas quantidades; já para os embarcadores, o fracionamento da carga correspondente a um pedido implica na multiplicação de tarefas administrativas e operacionais, tais como a emissão de mais documentos (notas fiscais, conhecimentos de transporte) e em uma maior dificuldade de conferência da carga sendo embarcada e, consequentemente, do correto atendimento dos pedidos.

Em particular, no Brasil a forma peculiar de contratação do transporte por caminhão, e de remuneração dos transportadores, favorece esse agrupamento por região ou município (Novaes, 2007). Em muitos problemas, pode-se assumir produto único ou homogêneo, como é o caso, por exemplo, do cimento ensacado (Miura, 2008), ou de produtos em caixas, ou paletes, ou até mesmo em gaiolas, permitindo assim desconsiderar a questão do arranjo espacial ou acomodação das cargas nos veículos.

Como se trata de um problema de entregas de carga fracionada (ou parcelada), pode-se assumir, sem perda de generalidade, que nenhuma entrega supera a capacidade do menor veículo disponível. Consequentemente, esse problema de distribuição pode ser modelado como um problema conhecido na literatura com o nome de *bin-packing problem* (BPP), definido da seguinte forma: dado um conjunto $j = 1,\ldots, n$ de objetos (ou itens), com seus respectivos pesos capacitados w_j e um conjunto $i = 1,\ldots, m$ de *bins* (ou mochilas) idênticos e de capacidade finita c determinar a alocação (ou designação) dos n itens aos *bins* (sendo $w_j \leq c, \forall j$), de tal modo que o número de *bins* utilizados seja mínimo, e a capacidade de cada *bin* não seja violada.

No caso do problema de distribuição em questão, os veículos de uma frota homogênea correspondem aos *bins*, e os objetos (ou itens) às cargas a serem entregues, cujos respectivos clientes estão localizados em uma mesma localidade, ou em localidades próximas, de tal forma que o frete de entrega não seja afetado pelas diferentes paradas, para a entrega ao longo do trajeto.

Problemas de *bin-packing* constituem um grupo de problemas desafiadores de otimização combinatória, os quais pertencem a uma classe mais geral, denominada "*problemas da mochila*" (do inglês *knapsack problems*), e que vêm sendo intensamente

estudados há várias décadas, atraindo teóricos e práticos (Martello & Toth, 1990). O interesse dos teóricos deve-se, principalmente, à estrutura simples do problema, o que permite, além da exploração de diversas propriedades do mesmo, a sua resolução como parte de problemas mais complexos. Já do ponto de vista prático, estes problemas aparecem em diversas aplicações reais, incluindo, entre outros, carregamento de veículos (Eilon & Christofides, 1971) e corte e empacotamento (Dyckhoff, 1990; Marques & Arenales, 2007; Hoto *et al.*, 2007).

Em um caso generalizado, a frota pode ser composta por veículos de diferentes tamanhos e capacidades, contendo dois ou mais tipos de veículos, aos quais estão associados custos diferentes. Neste caso, busca-se não mais minimizar o número total de veículos, mas sim o custo (ou frete) total dos veículos utilizados. Comumente, os veículos maiores possuem fretes unitários (por unidade de capacidade) menores, uma vez que tanto o custo fixo quanto o custo variável não são diretamente proporcionais à capacidade de carga do veículo; por exemplo, o consumo unitário de combustível não varia diretamente com a capacidade de carga; tampouco o salário do motorista costuma variar na mesma proporção da variação do tamanho do veículo, o que faz com que, por exemplo, um veículo com o dobro da capacidade de carga do outro não tenha o seu custo fixo dobrado.

Esse problema mais geral com frota heterogênea é conhecido na literatura como o problema de *bin-packing* com *bins* heterogêneos, ou de tamanho variável (do inglês *variable sized bin-packing problem*, ou simplesmente VSBPP), que é objeto do presente trabalho, tendo em vista sua aplicação prática no contexto logístico e de transporte. Ao contrário do problema de *bin-packing* tradicional, em que os *bins* são idênticos, este é um problema mais complexo.

Desta forma, são propostas três meta-heurísticas para o problema de *bin-packing* com *bins* heterogêneos. A primeira meta-heurística é baseada em Busca em Vizinhança Variável (VNS), a segunda meta-heurística é baseada em Busca em Vizinhança Variável com Lista Restrita (VNS-LR) e a terceira meta-heurística é baseada em Busca Decomposta em Vizinhança Variável (VNDS). As meta-heurísticas propostas são avaliadas considerando instâncias derivadas de instâncias *benchmarking* da literatura.

Neste trabalho busca-se encontrar um método de solução eficiente que produza boas soluções em tempos de processamento reduzidos, a fim de permitir a sua aplicação na programação diária da distribuição de produtos aos pontos de entrega. Tal programação

compreende a definição, para cada município ou região de destino, de quantos veículos de cada tipo serão necessários, e a alocação dos pontos de entrega aos mesmos, de modo que o custo total da frota utilizada seja mínimo. Tendo em vista a sua aplicação à programação diária de entregas, e ao prazo reduzido para essa atividade, o tempo de processamento, assim como a qualidade das soluções, são aspectos importantes para a avaliação da meta-heurística proposta.

O Problema de *Bin-Packing* com frota heterogênea (VSBPP) é *NP-hard*, uma vez que ele tem o BPP como caso particular, em que todos os *bins* são idênticos, que também é *NP-hard* (Garey & Johnson, 1979). Para o BPP, diversas heurísticas têm sido propostas para a sua resolução. As mais populares são aquelas em que os objetos são ordenados em ordem não-crescente de peso (w_j) e aplicadas regras para a alocação dos objetos aos *bins*, tais como *first-fit*, *best-fit* ou *next-fit* (Coffman *et al.*, 1997).

Uma boa revisão dos principais trabalhos na literatura para o BPP é apresentada por Scholl (1997), que também propõe um método exato para a sua resolução. Carvalho (1999) propôs um método exato baseado em geração de colunas e *branch-and-bound*. Gupta & Ho (1999) propõem uma heurística construtiva baseada na melhor folga do *bin*, denominada *minimum bin slack* (MBS), que possibilitou obter resultados superiores. Fleszar & Hindi (2002) propõem alterações na heurística MBS proposta por Gupta & Ho (1999), possibilitando encontrar resultados melhores com tempos computacionais menores; adicionalmente é também proposta uma meta-heurística baseada em VNS (*Variable Neighborhood Search*).

Motivados pela sua aplicação em problemas de distribuição física, Cunha *et al*. (2008) propuseram três meta-heurísticas para a solução desse problema: duas baseadas em GRASP e uma baseada em Busca em Vizinhança Variável (VNS). Experimentos computacionais realizados usando instâncias *benchmarking* evidenciaram que as três meta-heurísticas, que são simples e facilmente implementáveis, fornecem resultados de boa qualidade em tempos de processamento reduzidos, possibilitando a sua aplicação a problemas práticos em logística.

Já no caso do VSBPP, a literatura é bem mais restrita, e só mais recentemente esta generalização do problema clássico passou a atrair a atenção dos pesquisadores. Segundo Zhang (1997) as heurísticas clássicas para o BPP não fornecem bons resultados. O trabalho de

Friesen & Langston (1986) é um dos pioneiros sobre o VSBPP, definindo *bounds* e apontando a direção de pesquisas futuras com a utilização de heurísticas e meta-heurísticas. Burkard & Zhang (1997) e Kang & Park (2003) propõem heurísticas aplicadas ao problema de VSBPP e determinam o desempenho no pior caso de cada heurística para casos específicos.

Correia *et al.* (2008) utilizam uma formulação de programação linear inteira e uma reformulação do modelo de otimização discreta para a solução do VSBPP. Novas desigualdades sugeridas pelas variáveis do modelo discreto são adicionadas a fim de melhorar a relaxação dos limites do modelo linear original. Os resultados obtidos indicam que o modelo proposto obteve bons limites para a programação linear tornado-o adequado para resolver instâncias consideradas pequenas ou com até 1.000 itens. Para instâncias maiores o modelo ainda pode ser utilizado para a obtenção dos limites inferiores da solução.

Alves e Carvalho (2007) apresentam diferentes estratégias para aperfeiçoar o método de geração de colunas quando aplicado ao VSBPP e propõem dois algoritmos, um baseado em relaxação Lagrangiana e outro que resolve iterativamente sequencias de modelos de agregação. Os resultados comprovam uma redução significativa no número de colunas geradas e no tempo computacional necessário.

Mais recentemente, Haouari e Serairi (2009) analisaram o desempenho empírico de seis heurísticas, sendo quatro baseadas em métodos construtivos a partir da solução exata de subconjuntos dos dados; uma heurística baseada em Algoritmo Genético e uma heurística baseada em cobertura de conjuntos (*set covering*). As heurísticas propostas foram testadas e comparadas em conjuntos de dados, com até 2.000 objetos e sete tamanhos de *bins*, randomicamente gerados. A heurística baseada em cobertura de conjunto obteve sucesso conseguindo soluções de alta qualidade em baixo tempo computacional.

5.1. FORMULAÇÃO MATEMÁTICA

O problema de *bin-packing* com *bins* heterogêneos (VSBPP) pode ser descrito da seguinte forma: dados $j = 1,\ldots, n$ objetos ou itens com seus respectivos pesos w_j (representando as entregas a serem realizadas) e $i = 1,\ldots, m$ *bins* (ou mochilas, denotando os veículos a serem utilizados), para os quais são conhecidos seus respectivos custos c_i e suas

capacidades b_i, determinar a alocação (ou designação) dos n itens (entregas) aos m *bins* (veículos), de tal modo a minimizar o custo total dos *bins* utilizados, e respeitado-se as restrições de capacidade nos *bins*.

Assume-se, sem perda de generalidade, que os objetos j estejam ordenados em ordem decrescente de peso ($w_1 \geqslant w_2 \geqslant ... \geqslant w_n$) e que os *bins* i estejam ordenados em ordem crescente de tamanho ($b_1 \leqslant b_2 \leqslant ... \leqslant b_m$); adicionalmente, assume-se ainda que $w_j \leqslant b_i \forall i, j$ e que o número de *bins* m seja suficientemente elevado a fim de assegurar a viabilidade do problema. Assim, a formulação matemática do VSBPP pode ser escrita da seguinte forma:

$$Min \sum_{i=1}^{m} c_i y_i \tag{5.1}$$

$$\sum_{j=1}^{n} w_j x_{ij} \leq b_i y_i \quad \forall i = 1,\ldots,m \tag{5.2}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall j = 1,\ldots,n \tag{5.3}$$

$$x_{ij} \in [0,1] \quad \forall i = 1,\ldots,m \quad \forall j = 1,\ldots,n \tag{5.4}$$

$$y_i \in [0,1] \quad \forall i = 1,\ldots,m \tag{5.5}$$

Onde: x_{ij} é uma variável binária que assume o valor 1 quando o objeto j é alocado ao *bin* i e recebe o valor zero caso contrário; y_i é uma variável binária que recebe o valor 1 quando o *bin* i é utilizado na solução e assume o valor zero caso contrário.

A função de avaliação (5.1) busca minimizar o custo total dos veículos (ou *bins*) utilizados. A restrição (5.2) assegura que a capacidade de cada veículo utilizado não seja violada. A restrição (5.3) garante que cada entrega (ou objeto) j seja alocado a exatamente um veículo. As restrições (5.4) e (5.5) asseguram a integralidade das variáveis de decisão.

5.2. ESTRATÉGIA DE SOLUÇÃO

Nesta seção são apresentadas três meta-heurísticas para resolver o VSBPP. A

primeira meta-heurística é a Busca em Vizinhança Variável (VNS), a segunda meta-heurística é a Busca em Vizinhança Variável com Lista Restrita (VNS-LR) e a terceira meta-heurística é a Busca Decomposta em Vizinhança Variável (VNDS). Também serão apresentadas a função de avaliação utilizada e as estruturas de vizinhança desenvolvidas, bem como as definições necessárias para a aplicação de cada meta-heurística.

5.2.1. Heurística Construtiva para a Solução Inicial

Para a aplicação do VNS, do VNS-LR ou do VNDS, é necessária a geração de uma solução inicial. A solução inicial consiste de uma solução s na qual todas as entregas são alocadas aos veículos da frota. A solução inicial para a heurística aplicada ao VSBPP é obtida através de uma heurística construtiva, cujo pseudocódigo é apresentado na Figura 5.1.

```
Início SoluçãoInicial
    1. Ordenar cargas/ entregas em ordem decrescente de peso w_j;
    2. Ordenar veículos em ordem crescente de capacidade b_i;
    3. Para cada veículo i=1 até m faça  b̄_i = b_i
    4.     Inicializar o custo total inicial da solução C ← 0; y_i ← 0;
    5. Para cada carga j = 1 até n faça
    6.     i ← 1
    7.     Enquanto i ≤ m e w_j > b̄_i faça i ← i+1
    8.     Se i ≤ m então
    9.         x_ij ← 1
    10.        b̄_i = b̄_i − w_j
    11.        Se y_i = 0 então
    12.            y_i ← 1
    13.            C ← C + C_i
    14.        Fim-se
    15.    Fim-se
    16. Fim-Para
Fim SoluçãoInicial
```

Figura 5.1 – Pseudocódigo da heurística construtiva para obtenção da solução inicial.

A mesma heurística construtiva é utilizada para a geração solução inicial nas três meta-heurísticas desenvolvidas. A heurística construtiva para a geração da solução inicial, primeiramente, ordena as cargas (objetos) em ordem decrescente de peso e os veículos (*bins*) em ordem decrescente de folga (capacidade de carga ociosa), utilizando o método de

ordenação do algoritmo *QuickSort*. Maiores detalhes sobre o algoritmo *QuickSort* podem ser obtidos em Hoare (1962).

Após a ordenação dos objetos e dos *bins* cada uma das cargas é inserida, uma a uma e em ordem, no primeiro veículo que possuir espaço suficiente para recebê-la. Esta heurística construtiva gera somente soluções viáveis, isto é, garante que todos os objetos serão alocados a exatamente um único veículo e que nenhum veículo possuirá a sua capacidade violada.

A frota de veículos é considerada suficientemente grande para alocar todos os objetos a um único tipo de veículo caso seja necessário. O número de veículos de cada tamanho é grande o suficiente para transportar todos os objetos do problema.

5.2.2. Estruturas de Vizinhança

Para a aplicação das meta-heurísticas VNS, VNS-LR ou VNDS é necessário obter uma solução vizinha da solução corrente. Esta solução vizinha é gerada a partir de estruturas de vizinhança que visam a encontrar soluções cada vez mais "*distantes*" da solução corrente. A mesma sequência de movimentos foi utilizada para as três meta-heurísticas e é a seguinte:

- Troca 1-1;
- Troca 2-0A;
- Troca 2-0B;
- Troca 2-1;
- Reconstrução.

Na estrutura de vizinhança Troca 1-1 escolhe-se aleatoriamente um veículo que não esteja vazio, e se sorteia aleatoriamente uma entrega deste veículo. Esta entrega é candidata a ser removida do veículo ao qual está alocada e trocada com outra entrega alocada a outro veículo da frota. São testadas todas as possibilidades de troca da entrega selecionada aleatoriamente com as entregas alocadas aos demais veículos, até que uma troca viável seja encontrada. Por exemplo, suponha que foi sorteada a entrega j do *bin* i, inicialmente será feita uma tentativa de alocar esta entrega j (onde $j = 1,\ldots,$ total de entregas alocadas ao *bin*

i) do *bin* i no *bin* $i+1$ e o *bin* $i+1$ irá ceder a entrega k (onde $k=1,\ldots,$ total de entregas alocadas ao *bin* $i+1$) do *bin* $i+1$ para a troca. Caso seja viável, ou seja, existe espaço no *bin* i para receber a entrega k e espaço no *bin* $i+1$ para receber a entrega j esta troca é realizada. Caso contrário, o *bin* $i+1$ tentará ceder a entrega $k+1$ ao invés de ceder a entrega k na troca com o *bin* i pela entrega j .

Caso não seja encontrada nenhuma troca viável em termos das capacidades dos veículos cujas respectivas entregas são candidatas a serem trocadas, é realizada uma nova tentativa de troca, ou seja, repete-se o procedimento de escolher outro veículo da frota e selecionar aleatoriamente uma entrega alocada ao mesmo.

Na estrutura de vizinhança Troca 2-0A é escolhido aleatoriamente um veículo que não esteja vazio, e sorteadas duas entregas desse veículo. Estas duas entregas são removidas do veículo e o movimento de vizinhança tenta inserir ambas as entregas em outro veículo da frota, até ser gerado um movimento de vizinhança viável. Caso a troca não resulte em um movimento viável, é realizada uma nova tentativa de troca selecionando outro veículo da frota.

Na estrutura de vizinhança Troca 2-0B escolhe-se aleatoriamente um veículo que não esteja vazio, e sorteiam-se duas entregas desse veículo. Estas duas entregas são removidas do veículo e o movimento de vizinhança tenta inseri-las em outro veículo da frota, sendo que as entregas podem ser inseridas em um mesmo veículo ou em dois veículos distintos, de maneira mais flexível que o movimento de troca 2-0A, em que ambas as entregas devem ser alocadas a um mesmo veículo.

Já na estrutura de vizinhança denominada Troca 2-1 escolhe-se aleatoriamente um veículo que não esteja vazio, e sorteiam-se duas entregas desse veículo. Estas duas entregas são removidas do veículo, e o movimento de vizinhança tenta inserir ambas as entregas em outro veículo da frota, sendo que este veículo que recebeu as duas entregas cede a entrega com maior peso para o veículo de onde foram originariamente removidas as duas entregas.

Por fim, propõe-se a estrutura de vizinhança denominada Reconstrução, que consiste em retirar todas as entregas do veículo i^* com maior folga de capacidade $\overline{b_i}$ dentre os veículos utilizados (isto é, que possuam pelo menos uma entrega alocadas a eles). Esta estrutura de vizinhança também remove 20% das entregas alocadas a cada um dos demais veículos; em seguida, tenta-se sistematicamente inserir as entregas removidas do veículo i^*

nos demais outros veículos até encontrar um movimento viável.

As entregas a serem removidas dos demais veículos são escolhidas aleatoriamente e utiliza-se o método da solução inicial descrito na Figura 5.1 para realocar as entregas nos demais veículos. Esta estrutura de vizinhança permite uma maior perturbação na solução, e consequentemente uma melhor exploração do espaço de soluções viáveis devido a sua capacidade de escapar de soluções locais.

Deve-se notar que as estruturas de vizinhança propostas são progressivamente mais complexas, em termos dos tamanhos das vizinhanças a serem exploradas em cada movimento, o que está de acordo com os princípios que norteiam a busca em vizinhança variável.

5.2.3. Busca Local

Uma vez encontrada uma nova solução viável s' vizinha da solução corrente, é realizada uma busca local com a finalidade de tentar melhorar s'. O mesmo procedimento de busca local é utilizado pelas meta-heurísticas VNS e VNS-LR; já a meta-heurística VNDS utiliza um outro procedimento de busca local que será apresentado mais adiante na seção 5.2.4.

O procedimento de busca local é realizado utilizando-se um movimento de realocação de entregas, uma a uma, de modo a buscar diminuir o número de veículos utilizados. Como o custo da solução depende do número de veículos de cada tipo utilizado, e não das entregas alocadas a cada veículo, quanto menor for o número de veículos utilizados menor será o custo. As entregas somente podem ser alocadas a cada veículo se for assegurada a viabilidade das restrições de capacidade dos veículos.

Esse procedimento de melhoria da solução corrente corresponde a uma busca na vizinhança. A vizinhança é definida como o espaço de soluções possíveis de serem atingidas a partir da transferência de uma entrega a outro veículo. Neste contexto, a busca local considera como um ótimo local uma solução que é encontrada na qual não há transferências adicionais de uma entrega de um veículo a outro visando a reduzir o custo da solução corrente.

Mais especificamente, o procedimento de busca local é realizado da seguinte maneira: os veículos efetivamente utilizados (isto é, tais que $y_i = 1$) são ordenados em ordem

decrescente de folga $\overline{b_i}$. Seleciona-se o veículo i^* com maior capacidade ociosa $\overline{b_i}$ e ordenam-se as entregas alocadas ao mesmo (i.e. tais que $x_{ij}=1$) por ordem decrescente de peso w_j .

Em seguida, o procedimento de busca local tenta realocar cada uma das entregas desse veículo i^*, em ordem, a outro veículo; ou seja, a realocação é sempre iniciada pela entrega j^* de maior peso alocada ao veículo i^*, e sempre buscando o outro veículo i que corresponda à menor folga de capacidade $\overline{b_i}$ tal que $w_j \leq \overline{b_i}$ e $\overline{b_i}>0$, isto é, que o veículo i tenha capacidade para receber a carga j^* candidata a ser realocada. O procedimento de busca local não altera a carga dos veículos com capacidade ociosa nula para não prejudicar a busca pelas soluções de melhoria.

A realocação de cargas do veículo i^* é repetida até que todas as entregas originalmente alocadas ao mesmo tenham sido reinseridas em outros veículos, ou não seja mais possível encontrar alguma reinserção viável devido à falta de capacidade nos demais veículos. Neste caso, é determinado o menor tamanho de veículo que permite acomodar as cargas remanescentes no veículo i^*, que não puderam ser reacomodadas em outros veículos e, em seguida, recalculado o custo da solução de acordo.

Deve-se notar que, a cada nova reinserção de uma carga do veículo i^* em outro veículo i , há necessidade de reatualizar a ordenação dos demais veículos segundo ordem decrescente de folga $\overline{b_i}$, uma vez que a reinserção acarreta a alteração da capacidade ociosa do veículo i que recebe a entrega j^*. No entanto, isso é feito de maneira simplificada e rápida sem necessidade de uma reordenação completa dos veículos. Esta reordenação é rápida pois basta comparar a nova folga $\overline{b_i}$ do veículo i com as folgas dos demais veículos j que possuíam folgas menores que a do veículo i de modo a encontrar a nova posição do veículo i na lista ordenada de veículos na ordem decrescente de folga.

Em outras palavras, apenas o novo veículo que receber o objeto tem sua posição relativa alterada, sendo que os demais veículos permanecem na ordem correta. Embora a folga do veículo i^* seja alterada a cada vez que uma entrega é removida, não há necessidade de rever a sua ordem dentre os veículos, uma vez que a sua ociosidade só aumenta conforme entregas vão sendo realocadas a outros veículos. Como i^* corresponde ao veículo com maior ociosidade de capacidade no início da realocação, sua posição não muda.

Em síntese, através da busca local tenta-se reduzir o número de veículos utilizados

por meio da realocação das entregas dos veículos com maior ociosidade $\overline{b_i}$ para veículos que possuem uma maior quantidade de carga, mas ainda não estão transportando carga suficiente para atingir o limite de capacidade. A Figura 5.2 apresenta o procedimento de busca local.

O procedimento de busca local apresentado na Figura 5.2 visa a reduzir o número de veículos utilizados e a capacidade ociosa da frota, realocando os objetos dos veículos que possuem menor quantidade de carga atribuída para veículos que possuem uma maior quantidade de carga, mas ainda têm capacidade ociosa.

```
Início BuscaLocal
    1.  Ordene os veículos por ordem decrescente de folga;
    2.  Ordene as cargas por ordem decrescente de peso;
    3.  Para i= 1 até tot_veículos faça
    4.      Para j= i até tot_veículos faça
    5.          Se folgaVeic[j] > peso_objeto[k][i] então
    6.              Veic[j] ← objeto[k][i];
    7.              Atualiza custo[j];
    8.              Atualiza custo[i];
    9.          Fim-Se;
    10.     Fim-Para;
    11. Fim-Para;
Fim BuscaLocal
```

Figura 5.2 – Pseudocódigo do procedimento de busca local – VNS e VNS-LR.

5.2.4. Busca Local Decomposta

Nesta seção é apresentado o procedimento de busca local decomposta utilizado pela meta-heurística VNDS. A principal diferença entre o procedimento de busca local utilizado pelas meta-heurísticas VNS e VNS-LR e este procedimento de busca local decomposta utilizado pela meta-heurística VNDS é a decomposição do procedimento de busca local em duas etapas.

Mais especificamente, após ser gerada uma solução s' aleatória dentro da estrutura de vizinhança k, a primeira etapa da busca local decomposta é uma busca realizada na vizinhança de s' e utilizando a estrutura de vizinhança k. A segunda etapa da busca local é uma busca realizada dentro da vizinhança de s' e além da estrutura de vizinhança k que está sendo analisada.

O objetivo do procedimento de busca local decomposta é utilizar-se um movimento de realocação de entregas, uma a uma, de modo a buscar diminuir o número de veículos utilizados da mesma forma que o procedimento de busca local (VNS e VNS-LR, seção 5.2.3).

Mais especificamente, o procedimento de busca local decomposta é realizado da seguinte maneira: os veículos efetivamente utilizados (isto é, tais que $y_i=1$) são ordenados em ordem decrescente de folga $\overline{b_i}$. Seleciona-se o veículo i^* com maior capacidade ociosa $\overline{b_i}$ e ordenam-se as entregas alocadas ao mesmo (isto é, tais que $x_{ij}=1$) por ordem decrescente de peso w_j .

Em seguida, o procedimento de busca local decomposta tenta realocar cada uma das entregas desse veículo i^*, em ordem, a outro veículo; ou seja, a realocação é sempre iniciada pela entrega j^* de maior peso alocada ao veículo i^*, e sempre buscando o outro veículo i que corresponda à menor folga de capacidade $\overline{b_i}$ tal que $w_j \leq \overline{b_i}$, isto é, que o veículo i tenha capacidade para receber a carga j^* candidata a ser realocada.

A realocação de cargas do veículo i^* é repetida até que todas as entregas originalmente alocadas ao mesmo tenham sido reinseridas em outros veículos, ou não seja mais possível encontrar alguma reinserção viável devido à falta de capacidade nos demais veículos. Ou seja, a primeira etapa da busca local decomposta é idêntica ao procedimento de busca local utilizado pelas meta-heurísticas VNS e VNS-LR.

A segunda etapa do procedimento de busca local decomposta ocorre quando não existe mais a possibilidade de se encontrar reinserções viáveis das entregas j^* ao veículo i^* para outro veículo i que possua capacidade ociosa para receber esta entrega. Neste caso, é testada a troca de entregas entre o veículo i^* e outros veículos i . Desta forma, esta segunda etapa do procedimento de busca local decomposta tenta redistribuir as entregas visando obter uma melhor alocação e a liberação de espaço sem aumentar o número de veículos necessários.

Nesta segunda etapa do procedimento de busca local é permitido que sejam realizados movimentos que envolvam 3 (três) veículos. Por exemplo, uma entrega j' pode ser retirada do veículo i' e ser alocada ao veículo i'' ao mesmo tempo que uma entrega j'' do veículo i'' é alocada ao veículo i^* e uma entrega j^* do veículo i^* é alocada ao veículo i'. A Figura 5.3 ilustra esta segunda etapa.

Em síntese, o procedimento de busca local decomposta consegue realizar os

movimentos previstos no procedimento de busca local utilizado pelas meta-heurísticas VNS e VNS-LR e consegue analisar outros movimentos que não estão previstos no procedimento de busca local da seção anterior.

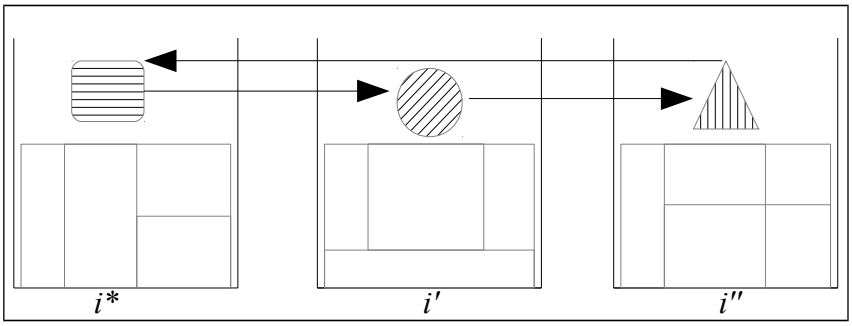

Figura 5.3 – Segunda etapa do procedimento de busca local decomposta - VNDS.

Desta forma o procedimento de busca local decomposta possui a capacidade de convergir mais rapidamente para uma solução de melhor qualidade devido a sua melhor exploração do espaço de soluções viáveis.

5.2.5. Busca Local com Função de Custo Modificada

Nesta seção são apresentadas as modificações feitas no procedimento de busca local utilizado pelas meta-heurísticas VNS e VNS-LR para a realização dos experimentos computacionais, utilizando as instâncias do trabalho de Haouari & Seriari (2009).

O procedimento de busca local com função de custo modificada visa reduzir o número de veículos utilizados e a capacidade ociosa da frota através da realocação dos objetos dos veículos que possuem menor quantidade de carga atribuída para veículos que possuem uma maior quantidade de carga, mas ainda possuem capacidade ociosa.

A busca local proposta corresponde a uma busca no espaço de soluções possíveis de serem atingidas a partir da transferência de uma entrega a outro veículo (vizinhança). Neste contexto, a busca local considera como um ótimo local uma solução na qual transferências adicionais de uma entrega de um veículo a outro não conseguem reduzir o custo da solução corrente.

Para os experimentos computacionais, utilizando as instâncias do trabalho de

Haouari & Seriari (2009), foi definido um procedimento de busca local para cada uma das três funções de custo utilizadas: (*i*) função de custo linear ($B1$): $c_i = w_i, i = 1,\ldots,m$; (*ii*) função de custo côncava ($B2$): $c_i = 10 * \sqrt{w_i}, i = 1,\ldots,m$; e (*iii*) função de custo convexa ($B3$): $c_i = 0{,}1 * w_i^{3/2}, i = 1,\ldots,m$. A função de custo côncava simula uma situação na qual quanto maior a capacidade de carga do caminhão, menor será o custo por unidade de carga transportada. A função de custo convexa simula uma situação na qual quanto maior a capacidade de carga do caminhão, maior será o custo por unidade de carga transportada.

O procedimento de busca local, no caso da função de custo linear, consiste em ordenar todos os veículos efetivamente utilizados em ordem decrescente de folga (capacidade ociosa). Em seguida, seleciona-se o veículo i com maior capacidade ociosa e ordena-se as entregas alocadas ao veículo i por ordem decrescente de peso. Após a ordenação das entregas alocadas ao veículo i , o algoritmo de busca local tenta realocar cada uma destas entregas a outro veículo, iniciando pela entrega j com maior peso e pelo veículo com a menor capacidade de folga disponível mas que possua capacidade para receber a entrega j .

A realocação das entregas do veículo i é repetida até que todas as entregas originalmente alocadas ao mesmo tenham sido reinseridas em outros veículos, ou não seja mais possível encontrar algum veículo para realocar alguma entrega devido à falta de capacidade. Deve-se observar que, a cada nova reinserção de uma carga do veículo i em outro veículo, existe a necessidade de atualizar a ordenação dos demais veículos.

O procedimento de busca local é encerrado quando todas as entregas originalmente alocadas ao veículo i tenham sido realocadas em outros veículos, ou não seja possível encontrar algum outro veículo que possua capacidade ociosa suficiente para receber as entregas remanescentes no veículo i .

O procedimento de busca local, no caso da função de custo côncava, consiste em ordenar todos os veículos de um mesmo tipo (mesma capacidade de carga) efetivamente utilizados em ordem decrescente de folga (capacidade ociosa). A busca local seleciona o veículo i com maior capacidade ociosa entre os veículos de cada tipo com menor capacidade de carga efetivamente utilizado, devido a serem os veículos com o maior custo relativo (custo/capacidade de carga).

Após a ordenação por ordem decrescente de peso das entregas alocadas ao veículo i o procedimento de busca local tenta realocar cada uma destas entregas a outro veículo que

pertença a um tipo com maior capacidade de carga do que o veículo atual, devido a serem os veículos com o menor custo relativo (custo/capacidade de carga). Por exemplo, se o veículo i for do tipo pequeno, o procedimento de busca local tentará retirar as entregas deste veículo e alocar a um veículo do tipo médio ou do tipo grande. O restante do procedimento de busca local é idêntico ao procedimento de busca local no caso da função de custo linear.

No caso da função de custo convexa, o procedimento de busca local é análogo ao caso da função de custo côncava. Entretanto, devido aos veículos com menor custo relativo serem os de menor capacidade e aos veículos com maior custo relativo serem os veículos de maior capacidade de carga, a busca local seleciona o veículo i com maior capacidade ociosa entre os veículos do tipo com maior capacidade de carga efetivamente utilizado e, após a ordenação por ordem decrescente de peso das entregas alocadas ao veículo i, o algoritmo de busca local tenta realocar cada uma destas entregas a outro veículo que pertença ao tipo com menor capacidade de carga. Por exemplo, se o veículo i for do tipo grande, o procedimento de busca local tentará retirar as entregas deste veículo e alocar a um veículo do tipo médio ou do tipo pequeno.

5.2.6. Busca Local Decomposta com Função de Custo Modificada

O procedimento de busca local decomposta com função de custo modificada utilizado pela meta-heurística VNDS é parecido com o procedimento de busca local apresentado na seção 5.2.5.

A primeira etapa do procedimento de busca local decomposta é igual ao procedimento de busca local apresentado na seção 5.2.5. A segunda etapa do procedimento de busca local decomposta é análoga ao procedimento de busca local decomposta apresentado na seção 5.2.4. Isto é, quando não é possível realocar a entrega j^* do veículo i^* a outro veículo i', é realizado um movimento de troca envolvendo 3 (três) veículos: uma entrega j' é retirada do veículo i' e é alocada ao veículo i'' ao mesmo tempo que uma entrega j'' do veículo i'' é alocada ao veículo i^* e uma entrega j^* do veículo i^* é alocada ao veículo i'.

5.3. EXPERIMENTOS COMPUTACIONAIS

Nesta seção são apresentados os experimentos computacionais realizados com cada uma das meta-heurísticas propostas e a comparação destes resultados com o valor da solução ótima obtida por um pacote de otimização (*solver*), utilizando o modelo matemático (5.1)-(5.5).

Foram utilizados três tamanhos de *bins*: pequeno, médio e grande. O *bin* do tipo pequeno possui capacidade de 80 unidades de carga (25% inferior ao *bin* do tipo médio) e custo de 112 unidades monetárias (20% menor do que o *bin* do tipo médio). O *bin* do tipo médio possui capacidade de 150 unidades de carga e custo de 100 unidades monetárias. O *bin* do tipo grande possui capacidade de 187 unidades de carga (25% acima do *bin* do tipo médio) e custo de 120 unidades monetárias (20% acima do *bin* do tipo médio).

Dessa forma, os *bins* que representam os veículos possuem custos unitários decrescentes com o aumento da capacidade, conforme ocorre na prática do transporte. A Tabela 5.1 apresenta a capacidade de cada *bin* em unidades de carga e o custo de cada *bin* em unidades monetárias.

Tabela 5.1 – Custo e Capacidade dos Bins.

	Pequeno	Médio	Grande
Custo (u.m.)	80	100	120
Capacidade (u.c.)	112	150	187

A função de avaliação tem o objetivo de quantificar a qualidade de uma solução vizinha, obtida através de um movimento realizado em uma das estruturas de vizinhança utilizadas pelas meta-heurísticas propostas (VNS, VNS-LR e VNDS). A função de avaliação proposta visa a quantificar a qualidade de uma solução para o VSBPP, considerando o custo da frota alocada e pode ser calculada através da expressão (5.6).

$$FA = \sum_{i=1}^{3} Custo_i \, NumBinTipo_i \tag{5.6}$$

Onde: $Custo_i$ é o custo por utilizar um veículo do tipo i e $NumBinTipo_i$ é a quantidade de veículos (*bins*) do tipo i utilizados.

As três meta-heurísticas baseadas em busca em vizinhança variável (VNS, VNS-LR e VNDS) para o VSBPP propostas neste trabalho foram testadas com dados de instâncias da literatura.

As instâncias de testes são instâncias *benchmarking* do VSBPP, disponíveis na *OR Library* (http://people.brunel.ac.uk/~mastjjb/jeb/orlib/binpackinfo.html), originalmente propostas por Falkenauer (1996) para o BPP, nos quais os *bins* são idênticos, e que haviam sido utilizadas por Cunha *et al*. (2008). A principal diferença das novas instâncias geradas foi a consideração de mais dois tamanhos de *bins* (veículos) além daquele de capacidade igual a 150.Tomando por base o tamanho de *bin* correspondente à capacidade igual a 150, conforme proposto por Falkenauer (1996), e atribuindo um custo de 100, foram criados mais dois tipos de *bins* da seguinte forma: um tipo de *bin* com capacidade 25% inferior e custo 20% inferior e um tipo de *bin* com capacidade 25% superior e custo 20% superior, como apresentado na Tabela 5.1. Dessa forma, o *bin* de maior capacidade é o que apresenta o menor custo por unidade de capacidade, tal como ocorre na prática onde veículos maiores são os de menor custo unitário.

Os testes foram realizados em um total de três conjuntos de dados, com instâncias cujo o número de objetos varia entre 10 e 1.000. As meta-heurísticas propostas VNS, VNS-LR e VNDS foram codificadas em C++ utilizando o compilador g++. Tendo em vista a aleatoriedade dos métodos propostos, cada instância foi resolvida cinco vezes e os valores das melhores soluções obtidas para cada instância estão apresentadas nas Tabelas 5.2, 5.3 e 5.4. O computador utilizado foi um Core2Quad 2.83GHz com 8GB de RAM com GNU/Linux Ubuntu 12.04 LTS 64bits. O pacote de otimização GUROBI foi utilizado para a resolução do modelo matemático (5.1)-(5.5). O apêndice B apresenta os resultados que justificam a escolha do pacote de otimização linear GUROBI.

A primeira coluna da Tabela 5.2 indica o número de objetos (cargas) a serem alocadas. A segunda coluna apresenta o limite inferior (*lower bound*) para cada uma das instâncias de teste. O limite inferior representa o custo mínimo para cada instância e é calculado considerando que todas as cargas sejam alocadas ao veículo com a melhor relação custo/benefício e este limite pode ser obtido pela expressão 5.7.

$$LowerBound = \frac{Custo_{GDE} \sum_{i=1}^{n} Peso_i}{Capacidade_{GDE}} \quad (5.7)$$

Onde: $Custo_{GDE}$ é o custo por utilizar um *bin* do tipo grande; $Capacidade_{GDE}$ é a capacidade do *bins* do tipo grande; $Peso_i$ é o peso do objeto i ; n é o total de objetos da instância do problema.

A coluna *Custo Sol* da Tabela 5.2 apresenta o custo da solução obtida para cada instância pelo GUROBI (o qual obteve as soluções ótimas somente para as instâncias da Tabela 5.2) e pelas meta-heurísticas VNS, VNS-LR e VNDS. A coluna *Tempo* apresenta o tempo computacional em segundos para obter a solução de cada instância de teste. Os resultados detalhados contendo o número de veículos do tipo grande, o número de veículos do tipo médio e o número de veículos do tipo pequeno utilizados na solução encontrada para cada instância são apresentados no apêndice C.

Tabela 5.2 - Resultados obtidos para instâncias de pequeno porte.

Número de Objetos	***Lower Bound***	**Solução Ótima**		**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
		Custo Sol	**Tempo (seg)**	**Custo Sol**	**Tempo (seg)**	**Custo Sol**	**Tempo**	**Custo Sol**	**Tempo (seg)**
10	371	400	1,7	**400**	0,22	**400**	0,21	**400**	0,21
12	443	460	2,1	**460**	0,31	**460**	0,30	**460**	0,29
12	344	360	3,9	**360**	0,39	**360**	0,38	**360**	0,37
15	589	600	0,0	**600**	0,33	**600**	0,32	**600**	0,31
20	681	700	0,8	**700**	0,34	**700**	0,33	**700**	0,32
25	1.037	1.060	7,6	**1.060**	0,53	**1.060**	0,51	**1.060**	0,50
40	1.634	1.660	7,6	**1.660**	1,02	**1.660**	0,99	**1.660**	0,97
40	1.542	1.560	0,8	1.580	1,09	**1.560**	1,06	**1.560**	1,04
40	1.635	1.660	40,7	**1.660**	1,03	**1.660**	1,00	**1.660**	0,98
40	1.542	1.560	2,5	1.580	0,88	**1.560**	0,85	**1.560**	0,83
40	1.634	1.660	8,5	1.680	0,92	**1.660**	0,89	**1.660**	0,87
40	1.592	1.620	5,1	1.640	0,99	**1.620**	0,96	**1.620**	0,94
40	1.530	1.540	1.784,0	1.560	0,93	1.560	0,90	**1.540**	0,88
40	1.496	1.520	1.238,0	**1.520**	0,92	**1.520**	0,89	**1.520**	0,87
40	1.633	1.660	1.417,0	**1.660**	0,93	**1.660**	0,90	**1.660**	0,88
40	1.551	1.560	2.393,0	1.600	0,94	1.600	0,92	**1.560**	0,89
40	1.503	1.520	65,3	1.540	0,78	1.540	0,76	**1.520**	0,74
Total	**20.757**	**21.100**	**6.979**	**21.260**	**12,55**	**21.180**	**12,18**	**21.100**	**11,92**

Os resultados apresentados na Tabela 5.2 são relativos às instâncias de pequeno porte, ou seja, com o número de entregas variando entre 10 e 40. Este conjunto de instâncias de teste foi definido como pequeno porte, devido a ser possível encontrar as soluções ótimas para cada um deles utilizando um pacote de otimização linear. Os valores das soluções ótimas apresentadas na Tabela 5.2 foram obtidos resolvendo o modelo matemático (5.1)-(5.5). Os resultados destacados mostram as instâncias nas quais as meta-heurísticas propostas obtiveram as soluções ótimas.

A meta-heurística VNS obteve as soluções ótimas em 10 das 17 instâncias testadas e nas instâncias em que não encontrou as soluções ótimas os resultados estiverem, em média, 1,45% acima (entre 1,20% e 2,56%). E o tempo computacional utilizado pela meta-heurística VNS é inferior ao tempo utilizado para a obtenção da solução ótima.

A meta-heurística VNS-LR obteve as soluções ótimas em 14 das 17 instâncias testadas e nas instâncias em que não obteve as soluções ótimas os resultados estiverem em média 1,73% acima (entre 1,30% e 2,56%). E o tempo computacional utilizado pela meta-heurística VNS-LR é, em média, 2,97% inferior ao tempo utilizado pela meta-heurística VNS para a resolução das instâncias de pequeno porte.

A meta-heurística VNDS obteve as soluções ótimas em todas as 17 instâncias testadas e nas instâncias e o tempo computacional requerido pela meta-heurística VNDS é, em média, 5,05% inferior ao tempo utilizado pela meta-heurística VNS para a resolução do problema e 2,14% inferior ao tempo utilizado pela meta-heurística VNS-LR para a resolução do problema.

Outra observação interessante é que os valores das soluções ótimas (obtida pela resolução do modelo matemático (5.1)-(5.5) e pela meta-heurística VNDS), estiveram em média, 2,21% acima do valor do limite inferior (*lower bound*).

Os problemas apresentados na Tabela 5.3 possuem entre 30 e 500 objetos a serem alocados e foram considerados como instâncias de médio porte devido ao GUROBI não conseguir encontrar uma solução ótima em um tempo inferior a uma hora (3.600 segundos). O valor da solução GUROBI apresentada na Tabela 5.3 mostra o valor da melhor solução encontrada dentro de uma hora e a coluna *GAP* apresenta a porcentagem que este valor de solução está acima do limite inferior calculado pelo GUROBI, utilizando relaxação linear.

Entre as 23 instâncias de médio porte resolvidas, em 5 deles a meta-heurística VNS encontrou uma solução pior que a obtida pelo GUROBI (em média, 0,92% acima). Em

11 das instâncias a meta-heurística VNS encontrou uma solução melhor que a encontrada pelo GUROBI (resultados destacados na Tabela 5.3). Os valores das soluções encontradas pela meta-heurística VNS ficaram, em média, 2,59% acima dos valores calculado dos limites inferiores.

Tabela 5.3 - Resultados obtidos para instâncias de médio porte.

Nº. de Objetos	***Lower Bound***	**Solução GUROBI**		**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
		Custo Sol	**GAP (%)**	**Custo Sol**	**Tempo (seg)**	**Custo Sol**	**Tempo (seg)**	**Custo Sol**	**Tempo (seg)**
30	1.215	1.240	1,32	1.260	0,70	1.240	0,68	1.240	0,66
35	1.362	1.400	2,24	1.400	0,87	1.400	0,85	1.400	0,82
40	**1.444**	**1.500**	**3,26**	**1.480**	**1,03**	**1.480**	**1,00**	**1.480**	**0,98**
45	1.966	2.000	1,28	2.000	1,36	2.000	1,32	2.000	1,29
50	2.002	2.040	1,60	2.060	1,28	2.040	1,25	2.040	1,22
55	1.998	2.040	1,82	2.060	1,33	2.040	1,29	2.040	1,26
60	2.356	2.400	1,58	2.420	1,55	2.400	1,50	2.400	1,47
70	**2.783**	**2.860**	**2,33**	**2.840**	**1,91**	**2.840**	**1,86**	**2.840**	**1,81**
120	4.921	5.000	1,26	5.000	5,14	5.000	4,97	5.000	4,87
120	4.930	5.040	1,91	5.040	6,88	5.040	6,70	5.040	6,55
120	4.644	4.720	1,36	4.720	4,22	4.720	4,10	4.720	3,99
120	4.687	4.760	1,29	4.760	5,10	4.760	4,94	4.760	4,86
120	4.687	4.760	1,29	4.760	4,25	4.760	4,10	4.760	4,02
250	**9.507**	**9.680**	**1,53**	**9.620**	**14,47**	**9.620**	**14,09**	**9.620**	**13,72**
250	**9.737**	**9.880**	**1,38**	**9.860**	**15,37**	**9.860**	**14,95**	**9.860**	**14,55**
250	**9.659**	**9.820**	**1,17**	**9.780**	**14,43**	**9.780**	**14,05**	**9.780**	**13,69**
250	9.572	9.680	0,86	9.700	15,33	9.680	14,90	9.680	14,55
250	**9.731**	**9.880**	**1,25**	**9.860**	**15,30**	**9.860**	**14,85**	**9.860**	**14,56**
500	**18.968**	**19.780**	**4,43**	**19.160**	**61,96**	**19.160**	**60,15**	**19.160**	**59,13**
500	**19.282**	**20.060**	**3,62**	**19.460**	**81,48**	**19.460**	**79,06**	**19.460**	**77,76**
500	**19.339**	**20.260**	**3,72**	**19.540**	**67,02**	**19.540**	**65,29**	**19.540**	**63,66**
500	**16.567**	**20.400**	**2,69**	**19.720**	**72,13**	**19.720**	**69,81**	**19.720**	**68,63**
500	**19.691**	**20.540**	**3,88**	**19.900**	**80,60**	**19.900**	**77,99**	**19.900**	**76,41**
Total	**181.048**	**189.740**	**2,05**	**186.400**	**473,71**	**186.300**	**459,7**	**186.300**	**450,5**

Nas 23 instâncias de médio porte resolvidas, em 12 deles a meta-heurística VNS-LR encontrou soluções com os mesmos valores obtidos pelo GUROBI. Em 11 das instâncias a meta-heurística VNS-LR encontrou soluções melhores que as obtidas pelo GUROBI (em média 1,78% abaixo). Os valores das soluções encontradas pela meta-heurística

VNS-LR ficaram, em média, 2,38% acima dos valores dos limites inferiores, ou seja, muito próximo do valor que as soluções ótimas estiveram acima dos valores calculados dos limites inferiores (2,21%) nas instâncias de pequeno porte.

Tabela 5.4 - Resultados obtidos para instâncias de grande porte (1.000 objetos).

Lower Bound	**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
	FA	**Tempo (seg)**	**FA**	**Tempo (seg)**	**FA**	**Tempo (seg)**
38.249	38.560	315	38.540	307	38.540	299
38.905	39.260	278	39.200	270	39.200	265
39.380	39.720	271	39.700	264	39.700	258
39.444	39.760	271	39.760	264	39.760	258
38.087	38.420	538	38.420	519	38.420	509
38.256	38.600	342	38.540	333	38.540	323
37.844	38.140	346	38.140	334	38.140	329
38.704	39.000	271	38.960	264	38.960	258
38.250	38.540	347	38.500	336	38.500	331
38.105	38.400	295	38.400	287	38.400	279
38.337	38.640	297	38.620	289	38.620	284
38.450	38.740	259	38.740	250	38.740	245
37.655	38.000	458	38.000	444	38.000	434
37.947	38.260	491	38.220	477	38.220	464
37.814	38.160	261	38.160	253	38.160	248
38.574	38.900	492	38.860	478	38.860	468
38.691	39.000	259	38.820	251	38.820	246
38.765	39.140	274	39.100	266	39.100	261
38.227	38.520	427	38.480	412	38.480	404
38.336	38.660	268	38.660	260	38.660	254
768.020	**774.420**	**6.760**	**773.820**	**6.558**	**773.820**	**6.416**

Em todas as 23 instâncias de médio porte resolvidas, os valores das soluções encontradas pela meta-heurística VNDS foram iguais aos valores das soluções obtidas pela meta-heurística VNS-LR. O tempo computacional utilizado pela meta-heurística VNDS é, em média, 5,06% inferior ao tempo utilizado pela meta-heurística VNS para a resolução do problema e 2,22% inferior ao tempo utilizado pela meta-heurística VNS-LR para a resolução do problema. Desta forma, a meta-heurística VNDS comprovou ser mais eficiente (menor

tempo computacional) e eficaz (melhor solução) do que as meta-heurísticas VNS ou VNS-LR para as instâncias de médio porte utilizadas neste trabalho.

Os resultados apresentados na Tabela 5.4 são relativos às instâncias de grande porte e possuem 1.000 objetos a serem alocados aos *bins*. Para os problemas cujos resultados estão apresentados na Tabela 5.4, o computador utilizado nos testes não conseguiu carregar o modelo matemático utilizado no GUROBI devido à falta de memória disponível, mesmo configurando o GUROBI para utilizar o espaço disponível no disco rígido para diminuir a necessidade de utilização da memória RAM.

A meta-heurística VNS conseguiu resolver os problemas relativos às instâncias de grande porte com um tempo computacional aproximadamente entre 4,32 e 8,97 minutos. Os valores das soluções obtidas pela meta-heurística VNS estão 0,83% acima dos valores dos limites inferiores calculados através da expressão 5.7 (*lower bound*).

A meta-heurística VNS-LR conseguiu resolver os problemas relativos às instâncias de grande porte com um tempo computacional aproximadamente entre 4,17 e 8,66 minutos.

O tempo computacional utilizado pela meta-heurística VNDS é, em média, 5,06% inferior ao tempo utilizado pela meta-heurística VNS para a resolução das instâncias de grande porte e 2,16% inferior ao tempo utilizado pela meta-heurística VNS-LR para a resolução das instâncias de grande porte. Os resultados obtidos pelas meta-heurísticas VNS-LR e VNDS estão 0,76% acima dos limites inferiores, o que pode indicar que a metodologia proposta pode ter conseguido se aproximar da solução ótima. Os valores dos limites inferiores são muito bons, isto sugere que as instâncias de teste de grande porte são fáceis de serem resolvidas. Isto facilitaria a convergência da meta-heurísticas.

5.3.1. Variação nos Resultados Obtidos

Tendo em vista a característica aleatória das meta-heurísticas VNS, VNS-LR e VNDS, em que um vizinho da solução corrente é obtido aleatoriamente (Figuras 2.4, 2.6 e 2.7), cada instância de teste foi submetida a cinco processamentos distintos, a fim de avaliar a sua robustez. No escopo deste trabalho, uma meta-heurística será considera robusta caso o desvio padrão entre os valores das soluções obtidas nas diferentes execuções seja próximo de

zero.

Os resultados apresentados anteriormente nas Tabelas 5.2 a 5.4 correspondem ao melhores resultados obtidos dentre os cinco processamentos para cada instância. Já a Tabela 5.5 apresenta um detalhamento das soluções obtidas; mais especificamente, são apresentados o desvio padrão do valor da função de avaliação entre os cinco processamentos para cada uma das meta-heurísticas implementadas em cada instância de grande porte.

Tabela 5.5 – Variação nos resultados obtidos para instâncias de grande porte (1.000 objetos).

Instância	**VNS**	**VNS-LR**	**VNDS**
	Desvio Padrão	**Desvio Padrão**	**Desvio Padrão**
1	0,02%	0,03%	0,03%
2	0,07%	0,08%	0,00%
3	0,07%	0,08%	0,00%
4	0,09%	0,07%	0,07%
5	0,05%	0,01%	0,06%
6	0,05%	0,08%	0,00%
7	0,06%	0,10%	0,00%
8	0,05%	0,05%	0,04%
9	0,08%	0,02%	0,03%
10	0,03%	0,02%	0,06%
11	0,06%	0,08%	0,06%
12	0,02%	0,01%	0,01%
13	0,04%	0,03%	0,04%
14	0,04%	0,06%	0,03%
15	0,05%	0,02%	0,02%
16	0,08%	0,09%	0,02%
17	0,05%	0,08%	0,02%
18	0,04%	0,09%	0,05%
19	0,08%	0,09%	0,03%
20	0,09%	0,09%	0,04%

Observe que os valores dos desvios padrões encontrados para cada uma das instâncias testadas são próximos de zero, mostrando que as meta-heurísticas desenvolvidas apresentam uma boa robustez.

O desvio padrão da meta-heurística VNS variou entre 0,02% (instâncias 1 e 12) e

0,09% (instâncias 4 e 20). O desvio padrão da meta-heurística VNS-LR variou entre 0,01% (instâncias 5 e 12) e 0,10% (instância 7). O desvio padrão da meta-heurística VNDS variou entre 0,00% (instâncias 2, 3, 6 e 7) e 0,07% (instância 4), isto significa que nas 5 vezes em que a meta-heurística VNDS foi executada para resolver as instâncias 2, 3, 6 e 7 foram obtidas as mesmas soluções. Desta forma, pode-se concluir que a meta-heurística VNDS comprovou ser a mais robusta (menor variação entre os resultados obtidos) meta-heurística implementada neste trabalho para as instâncias testadas.

5.4. Experimentos Computacionais com Função de Custo Modificada

As meta-heurísticas desenvolvidas para a resolução do VSBPP com a função de custo modificada, apresentada nas seções 5.2.5 e 5.2.6, foram testadas utilizando-se dois conjuntos de instâncias. O primeiro conjunto de instâncias de testes contempla instâncias com custo linear. O peso de cada objeto varia entre 1 e 100. Foram utilizadas 10 instâncias com n objetos $n \in \{25, 50, 100, 200, 500\}$. Estas instâncias possuem 3 tipos de *bins* com capacidades distintas 100, 120 e 150. A Tabela 5.6 apresenta o custo e a capacidade de cada *bin.*

Tabela 5.6 – Custo e Capacidade dos Bins.

Capacidade (u.c.)	**100**	**120**	**150**
Custo linear (u.m.)	100	120	150

Os dados da Tabela 5.7 apresentam os resultados obtidos por duas heurísticas propostas no trabalho de Haouari & Seriari (2009), uma heurística baseada em *set covering* (SC) e um algoritmo genético; e os resultados obtidos pelas meta-heurísticas VNS, VNS-LR e VNDS propostas neste trabalho.

A Tabela 5.7 apresenta o desvio padrão médio para 10 instâncias em relação ao valor da solução ótima; a coluna *desvio mínimo* apresenta o menor desvio padrão em relação ao valor da solução ótima; a coluna *desvio máximo* apresenta o maior desvio padrão em relação ao valor da solução ótima; a coluna *ótimo* apresenta o número de vezes que uma meta-heurística obteve a solução ótima para cada instância de teste; a coluna *tempo* mostra o

tempo de processamento computacional médio, em segundos, para cada meta-heurística encontrar a solução por instância de teste.

Tabela 5.7 – Performance das meta-heurísticas nas instâncias com custo linear.

linear		**Desvio Padrão médio**	**Desvio Padrão Máximo**	**Ótimo**	**Tempo (s)**
SC	**25**	0,54%	1,46%	4	0,08
	50	0,25%	0,52%	4	0,1
	100	0,10%	0,22%	5	0,39
	200	0,06%	0,10%	4	5,88
	500	0,10%	0,25%	2	18,43
	Total	**- - -**	**- - -**	**19**	**4,98**
GA	**25**	0,00%	0,00%	10	0,1
	50	0,07%	0,37%	8	0,11
	100	0,02%	0,18%	9	0,11
	200	0,01%	0,09%	9	0,22
	500	0,00%	0,00%	10	0,39
	Total	**- - -**	**- - -**	**46**	**0,19**
VNS	**25**	0,00%	0,00%	10	0,59
	50	0,00%	0,00%	10	1,69
	100	0,00%	0,00%	10	5,51
	200	0,01%	0,15%	9	12,19
	500	0,00%	0,00%	10	123,53
	Total	**- - -**	**- - -**	**49**	**143,51**
VNS-LR	**25**	0,00%	0,00%	10	0,57
	50	0,00%	0,00%	10	1,64
	100	0,00%	0,00%	10	5,32
	200	0,01%	0,15%	9	11,86
	500	0,00%	0,00%	10	119,87
	Total	**- - -**	**- - -**	**49**	**139,26**
VNDS	**25**	0,00%	0,00%	10	0,56
	50	0,00%	0,00%	10	1,61
	100	0,00%	0,00%	10	5,26
	200	0,01%	0,15%	9	11,53
	500	0,00%	0,00%	10	117,62
	Total	**- - -**	**- - -**	**49**	**136,58**

Os resultados apresentados na Tabela 5.7 mostram que as meta-heurísticas VNS,

VNS-LR e VNDS propostas neste trabalho obtiveram as soluções ótimas em 49 das 50 instâncias contra 46 vezes da meta-heurística GA e 19 vezes da heurística SC. A meta-heurística VNDS foi 1,92% mais rápida do que a meta-heurística VNS-LR e 4,83% mais rápida do que a meta-heurística VNS. Entretanto, o tempo computacional utilizado pela meta-heurística VNDS para obter as soluções ótimas foi muito maior que o tempo utilizado pela meta-heurística GA (136,58 segundos contra 0,19 segundos).

O segundo conjunto de testes contempla instâncias com custo linear, custo côncavo e custo convexo. Neste segundo conjunto de instâncias de teste foram utilizados sete tipos diferentes de *bins* (70, 100, 130, 160, 190, 220, 250). A Tabela 5.8 apresenta a capacidade de cada *bin* em unidades de carga e o custo de cada *bin* em unidades monetárias.

A Tabela 5.8 apresenta três funções diferentes de custo: (i) função de custo linear (B1): $c_i = w_i, i = 1,\ldots,m$; (ii) função de custo côncava (B2): $c_i = 10 * \sqrt{w_i}, i = 1,\ldots,m$; e (iii) função de custo convexa (B3): $c_i = 0{,}1 * w_i^{3/2}, i = 1,\ldots,m$.

Tabela 5.8 – Custo e Capacidade dos Bins.

Capacidade (u.c.)	**70**	**100**	**130**	**160**	**190**	**220**	**250**
Função de custo linear (u.m.)	70	100	130	160	190	220	250
Função de custo côncava (u.m.)	84	100	114	126	138	148	158
Função de custo convexa (u.m.)	59	100	148	202	262	326	395

As meta-heurísticas VNS, VNS-LR e VNDS propostas para o VSBPP com a função de custo modificada apresentada nas seções 5.2.5 a 5.2.6 foram testadas com os dados utilizados por Haouari & Seriari (2009). Os testes foram realizados em dois conjuntos de dados com problemas com o número de objetos variando entre 25 e 2.000. Para cada instância do problema, o programa desenvolvido em C++ foi executado cinco vezes e as melhores soluções obtidas para cada instância estão apresentadas nas Tabelas 5.9, 5.10 e 5.11.

Em todas as instâncias de teste apresentadas nas Tabelas 5.9, 5.10 e 5.11 foram encontradas as soluções ótimas pelas meta-heurísticas implementadas, as quais foram obtidas resolvendo o modelo matemático (5.1)-(5.5), utilizando o pacote de otimização GUROBI.

Os dados da Tabela 5.9 apresentam os resultados obtidos pelas meta-heurísticas baseadas em busca em vizinhança variável (VNS, VNS-LR e VNDS), propostas neste

trabalho, considerando a função de custo linear ($B1$). Os resultados apresentados na Tabela 5.9 mostram que a meta-heurística VNS-LR, proposta neste trabalho, obteve soluções com qualidade ligeiramente superior, desvio padrão médio de 0,58% e desvio padrão máximo de 1,64%, às soluções obtidas pelas meta-heurísticas VNS e VNDS.

Tabela 5.9 – Performance das meta-heurísticas resolvendo instâncias com função de custo linear (B1).

B1		**Desvio Padrão Médio**	**Desvio Padrão Máximo**	**Tempo (seg)**
VNS	**100**	0,95%	1,41%	19,99
	200	0,91%	1,63%	86,08
	500	0,45%	0,80%	771,2
	1000	0,36%	0,73%	1365,1
	2000	0,29%	0,42%	1217,74
	Média	**0,59%**	**1,63%**	**692,022**
VNS-LR	**100**	0,94%	1,40%	19,45
	200	0,89%	1,64%	83,88
	500	0,44%	0,79%	747,06
	1000	0,36%	0,74%	1320,87
	2000	0,28%	0,41%	1175,12
	Média	**0,58%**	**1,64%**	**669,276**
VNDS	**100**	0,94%	1,41%	19,07
	200	0,92%	1,63%	81,35
	500	0,44%	0,79%	735,84
	1000	0,35%	0,74%	1300,94
	2000	0,28%	0,43%	1159,66
	Média	**0,59%**	**1,63%**	**659,372**

Entretanto, o tempo computacional utilizado pela meta-heurística VNS-LR para obter essas soluções foram superiores aos tempos computacionais requeridos pela meta-heurística VNDS. As meta-heurísticas VNS e VNDS obtiveram soluções com desvio padrão médio de 0,59% e desvio padrão máximo de 1,63%. O tempo computacional utilizado pela meta-heurística VNS para obter essas soluções foi 3,40% superior ao tempo computacional requerido pela meta-heurística VNS-LR. A meta-heurística VNDS foi 1,48% mais rápida do que a meta-heurística VNS-LR.

Em resumo, para o conjunto de dados com 7 tamanhos de *bins* diferentes e com

função de custo linear, a meta-heurística VNS-LR obteve os melhores resultados e a meta-heurística VNDS foi a mais rápida.

A Tabela 5.10 apresenta os resultados obtidos considerando a função de custo côncava ($B2$), isto é, quanto maior o tamanho do *bin-packing* (caminhão) menor o custo por unidade de carga transportada. Este é o caso mais comum em problemas reais de transporte.

Tabela 5.10 – Performance das meta-heurísticas resolvendo instâncias com função de custo côncava (B2).

B2		**Desvio Padrão Médio**	**Desvio Padrão Máximo**	**Tempo (seg)**
VNS	**100**	0,97%	1,58%	22,85
	200	1,03%	1,93%	91,64
	500	0,68%	0,89%	373,89
	1000	0,55%	0,76%	1.493,07
	2000	0,42%	0,69%	1.130,46
	Média	**0,73%**	**1,93%**	**622,38**
VNS-LR	**100**	0,96%	1,57%	22,19
	200	1,02%	1,92%	89,32
	500	0,67%	0,89%	361,18
	1000	0,54%	0,76%	1.448,28
	2000	0,42%	0,68%	1.092,02
	Média	**0,72%**	**1,92%**	**602,60**
VNDS	**100**	0,95%	1,56%	21,76
	200	1,01%	1,91%	86,95
	500	0,66%	0,88%	356,35
	1000	0,54%	0,75%	1.423,79
	2000	0,41%	0,68%	1.077,22
	Média	**0,71%**	**1,91%**	**593,21**

Os resultados apresentados na Tabela 5.10 mostram que a meta-heurística VNDS proposta obteve soluções com qualidade superior, desvio padrão médio de 0,71% e desvio padrão máximo de 1,91%, às soluções obtidas pelas meta-heurísticas VNS e VNS-LR.

A meta-heurística VNS obteve soluções com desvio padrão médio de 0,73% e desvio padrão máximo de 1,93% e um tempo 4,92% superior à meta-heurística VNDS. A meta-heurística VNS-LR obteve soluções com desvio padrão médio de 0,72% e desvio padrão máximo de 1,92% e um tempo 1,58% superior à meta-heurística VNDS.

Em resumo, para o conjunto de dados com 7 tamanhos de *bins* diferentes e com

função de custo côncava, a meta-heurística VNDS obteve os melhores resultados e também foi a meta-heurística mais rápida.

A Tabela 5.11 apresenta os resultados obtidos considerando a função de custo convexa (B3), isto é, quanto maior o tamanho do *bin-packing* (caminhão) maior o custo por unidade de carga transportada.

Tabela 5.11 – Performance das meta-heurísticas resolvendo instâncias com função de custo convexa (B3).

B3		**Desvio Padrão Médio**	**Desvio Padrão Máximo**	**Tempo (seg)**
VNS	**100**	2,85%	7,34%	27,94
	200	3,61%	7,69%	111,88
	500	3,98%	8,89%	1.110,37
	1000	4,60%	8,58%	1.212,05
	2000	4,61%	9,37%	898,13
	Média	**3,93%**	**9,37%**	**672,07**
VNS-LR	**100**	2,85%	7,28%	27,09
	200	3,56%	7,69%	108,85
	500	3,97%	8,78%	1.074,39
	1000	4,58%	8,48%	1.170,23
	2000	4,56%	9,25%	873,43
	Média	**3,90%**	**9,25%**	**650,80**
VNDS	**100**	2,82%	7,20%	26,51
	200	3,52%	7,63%	106,34
	500	3,94%	8,65%	1.058,85
	1000	4,54%	8,41%	1.157,27
	2000	4,53%	9,20%	856,46
	Média	**3,87%**	**9,20%**	**641,08**

Os resultados apresentados na Tabela 5.11 mostram que a meta-heurística VNDS obteve soluções com qualidade superior, desvio médio de 3,87% e desvio máximo de 9,20%, às soluções obtidas pelas meta-heurísticas VNS e VNS-LR.

A meta-heurística VNS obteve soluções com desvio padrão médio de 3,93% e desvio padrão máximo de 9,37% e um tempo 4,83% superior à meta-heurística VNDS. A meta-heurística VNS-LR obteve soluções com desvio padrão médio de 3,90% e desvio padrão máximo de 9,25% e um tempo 1,52% superior à meta-heurística VNDS.

5.5. CONCLUSÕES

O presente capítulo abordou um problema de distribuição física em que se busca a otimização do agrupamento de entregas e carregamento nos veículos de modo a reduzir a frota necessária e o custo com o frete pago aos transportadores, que é modelado como um problema de *bin-packing* com *bins* heterogêneos (VSBPP).

A metodologia proposta baseia-se na meta-heurística VNS e em meta-heurística baseadas em busca em vizinhança variável (VNS-LR e VNDS), para a avaliação foram consideradas instâncias de teste *benchmarking* da literatura para o BPP, adaptadas para o VSBPP. Também foi feita uma comparação com os valores das soluções ótimas obtidas através da utilização do pacote de otimização GUROBI.

Os resultados indicam que a meta-heurística VNDS proposta permitiu obter as soluções ótimas em todas as instâncias nas quais foi possível a obtenção de soluções ótimas das mesmas. E nos demais testes as soluções obtidas sempre tiveram valores próximos dos limitantes inferiores (menos de 2,38% acima), sendo que as soluções ótimas obtidas encontram-se sempre acima dos respectivos limitantes (mais do que 2,21%), o que indica que a estratégia de solução proposta atinge resultados de excelente qualidade com tempos de processamento reduzidos. A meta-heurística VNDS implementada também se mostrou mais rápida que a solução ótima do modelo matemático e do que as meta-heurísticas VNS e VNS-LR.

O tempo computacional utilizado pela meta-heurística VNDS para resolver as instâncias de teste foi entre 4,83% e 5,06% inferior ao tempo utilizado pela meta-heurística VNS e entre 1,48% e 2,22% inferior ao tempo utilizado pela meta-heurística VNS-LR para a resolução do problema.

A análise de robustez das meta-heurísticas VNS, VNS-LR e VNDS comprovou que a meta-heurística VNDS, é a mais robusta dentre as meta-heurísticas implementadas, devido a possuir a menor variação entre os resultados obtidos. A meta-heurística VNS demonstrou ser uma meta-heurística mais robusta do que a VNS-LR, por apresentar uma variação menor nos resultados do que a variação nos resultados da VNS-LR.

Nas instâncias de teste de grande porte os valores das soluções obtidas por todas as meta-heurísticas se aproximaram bastante (0,76% acima) dos valores calculados para os

limites inferiores. Isto pode indicar que o limitante inferior é muito bom ou que as instâncias de grande porte são muito fáceis de serem resolvidas justificando o fato das meta-heurísticas terem convergido. Estes resultados não foram comparados com as heurísticas da literatura devido à indisponibilidade das soluções detalhadas obtidas em outros trabalhos.

As meta-heurísticas propostas foram testadas com uma função de custo modificada para permitir a comparação com outras heurísticas propostas na literatura, no caso foi utilizado o trabalho de Haouari & Seriari (2009) como base para comparação.

Os valores das soluções obtidas pelas meta-heurísticas VNS, VNS-LR e VNDS, propostas neste trabalho, obtiveram as soluções ótimas em 49 das 50 instâncias contra 46 vezes da meta-heurística GA e 19 vezes da heurística SC, propostas em Haouari & Seriari (2009). O desvio padrão máximo das meta-heurísticas, propostas neste trabalho, foram menores do que os desvios padrões máximos apresentados por Haouari & Seriairi (2009).

Entretanto, o tempo computacional utilizado pela meta-heurística VNDS para obter as soluções ótimas foi muito maior que o tempo utilizado pela meta-heurística GA (136,58 segundos contra 0,19 segundos). Desta forma, pode-se concluir que as meta-heurísticas propostas neste trabalho são mais robustas e eficazes para encontrar os valores das soluções ótimas neste conjunto de instâncias. As heurísticas propostas por Haouari & Seriari (2009) utilizam um tempo computacional muito inferior ao requerido pelas meta-heurísticas propostas neste trabalho, ou seja, as heurísticas propostas por Haouari & Seriari (2009) são mais eficientes.

A meta-heurística VNS-LR obteve os melhores resultados quando foi utilizado o conjunto de dados com 7 tamanhos de *bins* diferentes, com uma função de avaliação com custo linear. No caso em que a função de custo é convexa (quanto maior for a capacidade de carga do caminhão, maior é o custo por unidade de carga transportada), ou quando a função de custo é côncava (quanto maior for a capacidade de carga do caminhão, menor é o custo por unidade de carga transportada), a meta-heurística VNDS obteve as soluções de melhor qualidade.

Inúmeras são as possíveis extensões para a continuidade desta pesquisa, incluindo a investigação de novos métodos de solução, a aplicação e validação dos métodos propostos a problemas reais que envolvem o agrupamento de cargas no contexto da distribuição física de produtos. Outra proposta para a continuidade deste trabalho é o estudo do problema de *bin-packing* com duas dimensões (volume e peso dos objetos) e frota heterogênea.

6. O PROBLEMA BIDIMENSIONAL DE AGRUPAMENTO DE ENTREGAS EM VEÍCULOS COM FROTA HETEROGÊNEA

Neste capítulo é apresentado o problema de *bin-packing* bidimensional com *bins* de tamanho variável (do inglês *Bi-dimensional Variable Sized Bin-Packing Problem* – BiD-VSBPP). A principal diferença em relação ao VSBPP apresentado no capítulo 5 é a existência de uma segunda dimensão no problema; no BiD-VSBPP é considerado que cada objeto possui um peso e um volume, assim como cada *bin* (caminhão) possui uma capacidade de carga e uma capacidade volumétrica.

Neste capítulo são propostas três meta-heurísticas: uma meta-heurística de Busca em Vizinhança Variável (VNS) e duas meta-heurísticas baseadas em Busca em Vizinhança Variável (VNS-LR e VNDS) para o problema de *bin-packing* bidimensional ou tridimensional com *bins* heterogêneos. O problema bidimensional considera que as entregas a serem alocadas aos veículos possuem duas dimensões, no caso deste trabalho são consideradas as dimensões peso e volume. Mais especificamente, esta modelagem incorpora uma segunda dimensão de restrição de capacidade (volume), além do peso. Devido ao fato do volume de uma carga ser tridimensional, as meta-heurísticas propostas poderiam ser consideradas para abordarem problemas de *bin-packing* bidimensional ou tridimensional com *bins* heterogêneos.

As meta-heurísticas propostas são avaliadas considerando instâncias de *benchmarking* da literatura. Busca-se encontrar um método de solução eficaz e eficiente, que produza boas soluções em tempos de processamento reduzidos, a fim de permitir a sua aplicação na programação diária da distribuição de produtos aos pontos de entrega.

Esta programação compreende a definição, para cada município ou região de destino, de quantos veículos de qual tipo serão necessários, e a alocação dos pontos de entrega aos mesmos, de modo que o custo total da frota utilizada seja mínimo. Tendo em vista a sua aplicação à programação diária de entregas, e ao prazo reduzido para essa atividade, o tempo de processamento, assim como a qualidade das soluções, são aspectos importantes para a avaliação das meta-heurísticas propostas.

6.1. FORMULAÇÃO MATEMÁTICA

O problema de *bin-packing* bidimensional com *bins* heterogêneos (BiD-VSBPP) pode ser descrito da seguinte forma: dados $j = 1,\ldots, n$ objetos ou itens com seus respectivos pesos w_j^1 e volumes w_j^2, e $i = 1,\ldots, m$ *bins* (caminhões) para os quais são conhecidos seus respectivos custos c_i e suas capacidades em peso b_i^1 e volumétrica b_i^2, determinar a alocação (ou designação) dos n itens aos m *bins*, de tal modo a minimizar o custo total dos *bins* utilizados, e respeitado-se as restrições de capacidade nos *bins*. Assume-se, sem perda de generalidade, que todos os objetos possuam um tamanho inferior à capacidade do menor *bin* e que o número de *bins* m seja suficientemente elevado a fim de assegurar a viabilidade do problema. Assim, a formulação matemática do BiD-VSBPP pode ser escrita da seguinte forma:

$$Min \sum_{i=1}^{m} c_i y_i \tag{6.1}$$

$$\sum_{j=1}^{n} w_j^1 x_{ij} \leq b_i^1 y_i \quad \forall i = 1,\ldots,m \tag{6.2}$$

$$\sum_{j=1}^{n} w_j^2 x_{ij} \leq b_i^2 y_i \quad \forall i = 1,\ldots,m \tag{6.3}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad \forall j = 1,\ldots,n \tag{6.4}$$

$$x_{ij} \in \{0,1\} \quad \forall i = 1,\ldots,m \quad \forall j = 1,\ldots,n \tag{6.5}$$

$$y_i \in \{0,1\} \quad \forall i = 1,\ldots,m \tag{6.6}$$

Onde: x_{ij} é uma variável binária que assume o valor 1 quando o objeto i é alocado ao *bin* i e recebe o valor zero caso contrário; y_i é uma variável binária que recebe o valor 1 quando o *bin* i é utilizado na solução e assume o valor zero caso contrário.

A função de avaliação (6.1) busca minimizar o custo total dos veículos (ou *bins*) utilizados. As restrições (6.2) asseguram que a capacidade (peso) de cada veículo utilizado não seja violada; analogamente, as restrições (6.3) garantem que a capacidade volumétrica de cada veículo utilizado não seja violada. As restrições (6.4) impõem que cada entrega (ou

objeto) j seja alocada a exatamente um veículo. Por fim, as restrições (6.5) e (6.6) asseguram a integralidade das variáveis de decisão.

6.2. ESTRATÉGIA DE SOLUÇÃO

A estratégia de solução utilizada para a resolução do problema de *bin-packing* bidimensional com *bins* heterogêneos (BiD-VSBPP) é similar à estratégia de solução apresentada na seção 5.2. Entretanto devido ao fato de o BiD-VSBPP possuir uma dimensão a mais que o VSBPP, são necessárias algumas adaptações na estratégia de solução que serão apresentadas nesta seção.

Nesta seção são apresentadas três meta-heurísticas para resolver o BiD-VSBPP. A primeira meta-heurística é baseada em Busca em Vizinhança Variável (VNS), a segunda meta-heurística é a Busca em Vizinhança Variável com Lista Restrita (VNS-LR) e a terceira meta-heurística é a Busca Decomposta em Vizinhança Variável (VNDS) Também serão apresentadas a função de avaliação utilizada e as estruturas de vizinhança desenvolvidas, bem como as definições necessárias para a aplicação de cada meta-heurística.

6.2.1. Heurística Construtiva para a Solução Inicial

Para a aplicação das meta-heurísticas VNS, VNS-LR ou VNDS, é necessária a geração de uma solução inicial, a qual consiste de uma solução *s* na qual todas as entregas são alocadas aos veículos da frota. A solução inicial para a heurística aplicada ao BiD-VSBPP é obtida através de uma heurística construtiva, cujo pseudocódigo é apresentado na Figura 6.1.

A heurística construtiva para a geração da solução inicial, primeiramente, ordena as cargas (objetos) em ordem decrescente de peso cubado e os veículos (*bins*) em ordem decrescente de folga (capacidade de carga ociosa), utilizando o método de ordenação do algoritmo *QuickSort*. Maiores detalhes sobre o algoritmo *QuickSort* podem ser obtidos em Hoare (1962). O peso cubado é calculado pela expressão (6.7).

$$Max[\, w_j^1 ; 300\, w_j^2 \,] \tag{6.7}$$

Onde: w_j^1 é o peso em Quilogramas; w_j^2 é o volume em metros cúbicos; *300 kg/m³* é o valor de uma constante de conversão utilizada para o transporte rodoviário de carga.

```
Início SoluçãoInicial
1. Ordene as cargas/entregas em ordem decrescente de peso cubado;
2. Ordene os veículos em ordem crescente de capacidade b_i;
3. Para i= 1 até tot_entregas faça
4.      Se folgaPeso[j] > peso_objeto[k][i] E
5.          folgaVolume[j] > volume_objeto[k][i] Então
6.          Veic[j] ← entrega[i];
7.          Atualiza custo[j];
8.      Senão
9.          j ← j + 1;
10. Fim-Para;
Fim SoluçãoInicial
```

Figura 6.1 – Pseudocódigo da heurística construtiva para obtenção da solução inicial.

Em seguida, cada uma das cargas é inserida, uma a uma e em ordem, no primeiro veículo que possuir espaço suficiente (tanto em relação ao peso quanto ao volume) para recebê-la. Esta heurística construtiva gera somente soluções viáveis, isto é, garante que todos os objetos serão alocados a exatamente um único veículo e que nenhum veículo possuirá a sua capacidade violada.

Assume-se, sem perda de generalidade, que a frota de veículos seja considerada suficientemente grande para possibilitar a alocação de todos os objetos a apenas um único tipo de veículo caso necessário; em outras palavras, o número de veículos de cada tamanho é grande o suficiente para permitir transportar todos os objetos do problema. A fim de determinar o número de veículos, de cada tamanho, suficiente para transportar todos os objetos do problema, a heurística da solução inicial é aplicada isoladamente para cada tamanho de veículo de forma a alocar todas as entregas para um único tipo de veículo. Deve-se notar que essa heurística aplicada desta forma fornece um limitante superior da frota total necessária, uma vez que não há limitação no número de veículos de cada tipo disponível.

6.2.2. Estruturas de Vizinhança

Para a aplicação das meta-heurísticas VNS, VNS-LR ou VNDS é necessário obter uma solução vizinha da solução corrente. Essa solução vizinha é gerada a partir de estruturas de vizinhança que visam encontrar soluções cada vez mais "*distantes*" da solução corrente. A mesma sequência de movimentos foi utilizada para as três meta-heurísticas é a seguinte:

- Troca 1-1;
- Troca 2-0A;
- Troca 2-0B;
- Troca 2-1;
- Perturbação;
- Reconstrução.

Na estrutura de vizinhança Troca 1-1 escolhe-se aleatoriamente um veículo i ($i = 1,\ldots,$ total de veículos), que não esteja vazio, e sorteia-se uma entrega j ($j = 1,\ldots,$ total de entregas do veículo i) desse veículo. Esta entrega é removida do veículo e o movimento de vizinhança tenta trocar com uma entrega de outro veículo da frota até ser gerado um movimento de vizinhança viável. Caso a troca não resulte em um movimento viável, é realizada uma nova tentativa de troca, selecionando-se outro veículo da frota. Após a realização de um movimento viável é feita uma busca local no entorno da solução atual. O movimento somente é considerado viável se as restrições de peso máximo e de volume máximo do veículo sejam respeitadas.

O procedimento utilizado durante a busca local é escolher o veículo que possuir a maior folga em relação à sua capacidade, considerando o peso cubado, e tentar realocar cada um dos objetos deste veículo a outro veículo. A realocação é sempre iniciada pelo maior objeto (em relação ao peso cubado) e sempre buscando o veículo com a menor folga de capacidade, considerando o peso cubado para receber o objeto realocado. A realocação de objetos é repetida até que todos os objetos do veículo tenham sido realocados, ou não seja possível realocar nenhum objeto devido à falta de espaço nos demais veículos.

Já na estrutura de vizinhança Troca 2-0A é escolhido aleatoriamente um veículo

que não esteja vazio, e sorteadas duas entregas desse veículo. Estas duas entregas são removidas do veículo e o movimento de vizinhança tenta inserir ambas as entregas em outro veículo da frota, até ser gerado um movimento de vizinhança viável. Caso a troca não resulte em um movimento viável, é realizada uma nova tentativa de troca selecionando outro veículo da frota. Após a realização de um movimento viável é feita uma busca local no entorno da solução atual.

Na estrutura de vizinhança Troca 2-0B escolhe-se aleatoriamente um veículo que não esteja vazio, e sorteiam-se duas entregas desse veículo. Estas duas entregas são removidas do veículo e o movimento de vizinhança tenta inseri-las em outro veículo da frota, sendo que ambas as entregas podem ser inseridas em um mesmo veículo ou em dois veículos distintos.

Por fim, na estrutura de vizinhança Troca 2-1 escolhe-se aleatoriamente um veículo, que não esteja vazio e sorteiam-se duas entregas desse veículo. Estas duas entregas são removidas do veículo, e o movimento de vizinhança tenta inserir ambas as entregas em outro veículo da frota, sendo que este veículo que recebeu as duas entregas cede uma entrega (escolhida aleatoriamente) para o veículo de onde foram originariamente removidas as duas entregas.

A estrutura de vizinhança Perturbação escolhe aleatoriamente uma entrega de um veículo v_1 e insere em um veículo v_2 ao mesmo tempo em que retira uma entrega de v_2 e insere no veículo v_3 ; este movimento é repetido até encontrar um movimento viável.

A estrutura de vizinhança Reconstrução retira todas as entregas do veículo com maior folga de capacidade (que possua pelo menos uma entrega) e retira 20% das entregas alocadas aos demais veículos, em seguida, tenta sistematicamente inserir as entregas em outros veículos até encontrar um movimento viável. Após encontrar um movimento viável é realizada uma busca local no entorno da solução encontrada. Esta estrutura de vizinhança permite uma maior perturbação na solução e consequentemente uma melhor exploração do espaço de soluções viáveis devido a sua capacidade de escapar de ótimos locais.

Em seguida à geração de uma solução vizinha à solução corrente é realizada uma busca local. Os veículos são ordenados em ordem decrescente de folga (capacidade ociosa considerando o peso cubado). A etapa seguinte é tentar realocar todas as entregas deste veículo para outros veículos, sempre buscando o veículo com a menor folga de capacidade para receber a carga sendo redistribuída. Este procedimento é repetido até que todas as cargas

tenham sido realocadas ou não seja possível realocar mais nenhum objeto.

6.2.3. Busca Local

Uma vez encontrada uma nova solução viável s' vizinha da solução corrente, é realizada uma busca local com a finalidade de tentar melhorar s'. O mesmo procedimento de busca local é utilizado pelas meta-heurísticas VNS e VNS-LR; já a meta-heurística VNDS utiliza um outro procedimento de busca local que será apresentado mais adiante na seção 6.2.4.

O procedimento de busca local é realizado utilizando-se um movimento de realocação de entregas, uma a uma, de modo a buscar diminuir o número de veículos utilizados. Como o custo da solução depende do número de veículos de cada tipo utilizado, e não das entregas alocadas a cada veículo, quanto menor for o número de veículos utilizados menor será o custo. As entregas somente podem ser alocadas a cada veículo se for assegurada a viabilidade das restrições de capacidade dos veículos.

Esse procedimento de melhoria da solução corrente corresponde a uma busca na vizinhança. A vizinhança é definida como o espaço de soluções possíveis de serem atingidas a partir da transferência de um objeto a outro veículo. Neste contexto, a busca local considera como um ótimo local uma solução na qual transferências adicionais de um objeto de um veículo a outro não são possíveis de reduzir o custo da solução corrente.

Mais especificamente, o procedimento de busca local é realizado da seguinte maneira: os veículos efetivamente utilizados (isto é, tais que y_{ij}) são ordenados em ordem decrescente de folga $\overline{b_i}$. Seleciona-se o veículo i^* com maior capacidade ociosa $\overline{b_i}$ e ordenam-se as entregas alocadas ao mesmo (isto é, tais que $x_{ij}=1$) por ordem decrescente de peso cubado.

Em seguida, o procedimento de busca local tenta realocar cada uma das entregas desse veículo i^*, em ordem, a outro veículo; ou seja, a realocação é sempre iniciada pela entrega j^* de maior peso cubado alocada ao veículo i^*, e sempre buscando o outro veículo i que corresponda à menor folga de capacidade considerando o peso cubado $\overline{b_i}$ tal que o veículo i tenha capacidade para receber a carga j^* candidata a ser realocada. O procedimento

de busca local não altera a carga dos veículos com capacidade volumétrica ociosa nula ou capacidade de peso ociosa nula para não prejudicar a busca pelas soluções de melhoria.

A realocação de cargas do veículo i^* é repetida até que todas as entregas originalmente alocadas ao mesmo tenham sido reinseridas em outros veículos, ou não seja mais possível encontrar alguma reinserção viável devido à falta de capacidade nos demais veículos. Neste caso, é verificado o menor tipo de veículo que permite acomodar as cargas remanescentes no veículo i^*, que não puderam ser reacomodadas em outros veículos, e recalculado o custo da solução de acordo.

Deve-se notar que, a cada nova reinserção de uma carga do veículo i^*em outro veículo i , há necessidade de reatualizar a ordenação dos demais veículos segundo ordem decrescente de folga considerando o peso cubado, uma vez que a reinserção acarreta a alteração da capacidade ociosa do veículo i que recebe a entrega j^*. No entanto, isso é feito de maneira simplificada e rápida sem necessidade de uma reordenação completa dos veículos. Esta reordenação é rápida pois basta comparar a nova folga do veículo i com as folgas dos demais veículos j que possuíam folga menor que a do veículo i de modo a encontrar a nova posição do veículo i na lista ordenada de veículos ordem decrescente de folga.

Em outras palavras, apenas o novo veículo que receber o objeto tem sua posição relativa alterada, sendo que os demais veículos permanecem na ordem correta. Embora a folga do veículo i^* seja alterada a cada vez que uma entrega é removida, não há necessidade de rever a sua ordem dentre os veículos, uma vez que a sua ociosidade só aumenta conforme entregas vão sendo realocadas a outros veículos. Como i^* corresponde ao veículo com maior ociosidade de capacidade ao início da realocação, sua posição não muda.

Em síntese, através da busca local tenta-se reduzir o número de veículos utilizados por meio da realocação das entregas dos veículos com maior ociosidade para veículos que possuem uma maior quantidade de carga, mas ainda não estão transportando carga suficiente para atingir o limite de capacidade.

6.2.4. Busca Local Decomposta

Nesta seção é apresentado o procedimento de busca local decomposta utilizado pela meta-heurística VNDS. A principal diferença entre o procedimento de busca local

utilizado pelas meta-heurísticas VNS e VNS-LR e o procedimento de busca local decomposta utilizado pela meta-heurística VNDS é a decomposição do procedimento de busca local em duas etapas.

Após ser gerada uma solução s' aleatória dentro da estrutura de vizinhança k a primeira etapa da busca local decomposta é uma busca realizada na vizinhança de s' e utilizando a estrutura de vizinhança k . A segunda etapa da busca local é uma busca realizada dentro da vizinhança de s' e além da estrutura de vizinhança k que está sendo analisada.

O objetivo do procedimento de busca local decomposta é utilizar-se um movimento de realocação de entregas, uma a uma, de modo a buscar diminuir o número de veículos utilizados da mesma forma que o procedimento de busca local (VNS e VNS-LR, seção 6.2.3).

Mais especificamente, o procedimento de busca local decomposta é realizado da seguinte maneira: os veículos efetivamente utilizados (isto é, tais que $y_{ij}=1$) são ordenados em ordem decrescente de folga considerando o peso cubado. Seleciona-se o veículo i^* com maior capacidade ociosa e ordenam-se as entregas alocadas ao mesmo por ordem decrescente de peso cubado.

Em seguida, o procedimento de busca local decomposta tenta realocar cada uma das entregas desse veículo i^*, em ordem, a outro veículo; ou seja, a realocação é sempre iniciada pela entrega j^* de maior peso cubado alocada ao veículo i^*, e sempre buscando o outro veículo i que corresponda à menor folga de capacidade considerando o peso cubado desde que o veículo i tenha capacidade (volume e peso) para receber a carga j^* candidata a ser realocada.

A realocação de cargas do veículo i^* é repetida até que todas as entregas originalmente alocadas ao mesmo tenham sido reinseridas em outros veículos, ou não seja mais possível encontrar alguma reinserção viável devido à falta de capacidade nos demais veículos. Ou seja, a primeira etapa da busca local decomposta é idêntica ao procedimento de busca local utilizado pelas meta-heurísticas VNS e VNS-LR.

A segunda etapa do procedimento de busca local decomposta ocorre quando não existe mais a possibilidade de se encontrar reinserções viáveis das entregas j^* ao veículo i^* para outro veículo i que possua capacidade ociosa para receber esta entrega. Neste caso, é testado a troca de entregas entre o veículo i^* e outros veículos i . Desta forma, esta segunda

etapa do procedimento de busca local decomposta tenta redistribuir as entregas visando obter uma melhor alocação e a liberação de espaço sem aumentar o número de veículos necessários.

Nesta segunda etapa do procedimento de busca local é permitido que sejam realizados movimentos que envolvam 3 veículos. Por exemplo, uma entrega j' pode ser retirada do veículo i' e ser alocada ao veículo i'' ao mesmo tempo que uma entrega j'' do veículo i'' é alocada ao veículo i^* e uma entrega j^* do veículo i^* é alocada ao veículo i'. A Figura 6.2 apresenta ilustra esta segunda etapa.

Em síntese, o procedimento de busca local decomposta consegue realizar os movimentos previstos no procedimento de busca local utilizado pelas meta-heurísticas VNS e VNS-LR e consegue analisar outros movimentos que não estão previsto no procedimento de busca local da seção anterior.

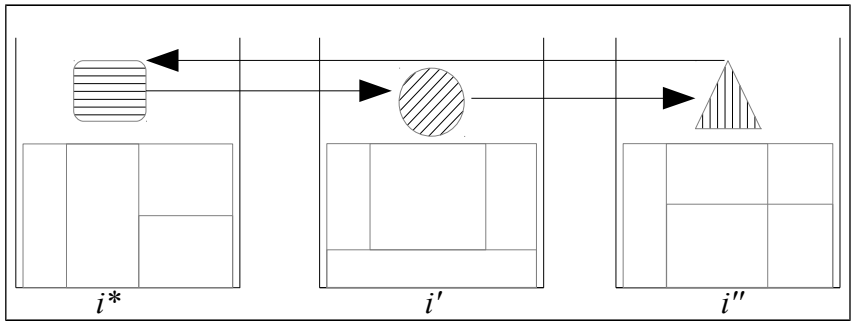

Figura 6.2 – Segunda etapa do procedimento de busca local decomposta - VNDS.

Desta forma o procedimento de busca local decomposta possui a capacidade de convergir mais rapidamente para uma solução de melhor qualidade, devido à sua melhor exploração do espaço de soluções viáveis.

6.3. EXPERIMENTOS COMPUTACIONAIS

As meta-heurísticas VNS, VNS-LR e VNDS implementadas para resolver o BiD-VSBPP foram testadas com dados de instâncias da literatura utilizados na seção 5.3. A diferença básica das novas instâncias foi a geração de uma capacidade volumétrica idêntica à capacidade em quilogramas do veículo e a adição de um volume para cada objeto a ser alocado aos *bins*. Cada instância foi resolvida 5 vezes pelas meta-heurísticas desenvolvidas,

as variâncias dos resultados são apresentadas na seção 6.3.1, avaliando a robustez das meta-heurísticas propostas.

Por simplificação, considerou-se que as capacidades volumétricas dos *bins* são iguais às capacidades em peso. Entretanto, o peso w_j^1 e o volume w_j^2 de cada entrega j diferem com o objetivo de configurar produtos com diferentes densidades de carga (ou, similarmente, fatores de estiva) de modo que os veículos possam lotar ou por peso ou por volume, dependendo das entregas alocadas a cada um deles. Para determinar o volume w_j^2 de cada entrega j foi escolhido, aleatoriamente, um valor de volume correspondente ao intervalo entre 50% e 150% do valor do peso w_j^1 de cada entrega.

Tabela 6.1 – Custo e Capacidade dos veículos.

	Grande	**Médio**	**Pequeno**
Custo ($)	120	100	80
Capacidade (kg)	187	150	112
Capacidade (m³)	187	150	112

A função de avaliação tem o objetivo de avaliar a qualidade de uma solução vizinha obtida através de um movimento realizado por uma das estruturas de vizinhança utilizada pelas meta-heurísticas VNS, VNS-LR ou VNDS. A função de avaliação proposta visa a avaliar a qualidade de uma solução para o BiD-VSBPP considerando o custo da frota alocada e pode ser calculada através da expressão (6.8).

$$FA = \sum_{i=1}^{3} Custo_i \, NumBinTipo_i \tag{6.8}$$

Onde: $Custo_i$ é o custo por utilizar um veículo do tipo i ; $NumBinTipo_i$ é a quantidade de veículos (*bins*) do tipo i utilizados.

Os testes foram realizados em um total de quatro conjuntos de dados com problemas com o número de objetos variando entre 120 e 1.000. As meta-heurísticas propostas VNS, VNS-LR e VNDS foram codificadas em C++ utilizando o compilador g++. Tendo em vista a aleatoriedade dos métodos propostos, cada instância foi resolvida cinco

vezes e as melhores soluções obtidas para cada instância estão apresentadas nas Tabelas 6.2, 6.3, 6.4 e 6.5. O computador utilizado foi um Core2Quad 2.83GHz com 8GB de RAM e 500GB de HD rodando GNU/Linux Ubuntu 12.04 LTS 64 bits.

A primeira coluna da Tabela 6.2 apresenta o número de objetos (cargas) a serem alocadas. A segunda coluna apresenta o limite inferior (*lower bound*) para cada uma das instâncias de teste. O limite inferior representa o custo mínimo para cada instância e é calculado considerando que todas as cargas sejam alocadas ao veículo com a melhor relação custo/benefício e este limite pode ser obtido pela expressão (6.9).

$$LowerBound = \frac{Custo_{GDE} \sum_{i=1}^{n} Peso_i}{Capacidade_{GDE}} \tag{6.9}$$

Onde: $Custo_{GDE}$ é o custo por utilizar um veículo do tipo grande; $Capacidade_{GDE}$ é a capacidade dos veículos do tipo grande; $Peso_i$ é o peso da carga i ; n é o total de cargas da instância do problema.

A coluna *FO* da Tabela 6.2 apresenta o custo da solução obtida para cada instância pelo pacote de otimização GUROBI (o qual não possibilitou determinar a solução ótima em nenhuma das instâncias testadas) e pelas 3 meta-heurísticas. A coluna *Tempo* apresenta o tempo computacional, medido em segundos, para a obtenção da solução apresentada por cada uma das meta-heurísticas implementadas.

Os resultados apresentados na Tabela 6.2 são relativos aos problemas de pequeno porte, com 120 entregas a serem alocadas aos veículos. Apesar deste conjunto de problemas ser definido como de pequeno porte, não foi possível encontrar a solução ótima para cada um deles utilizando o GUROBI. Os valores de função objetivo das soluções apresentadas na coluna *Solução GUROBI* na Tabela 6.2 é a melhor solução obtida pelo pacote de otimização GUROBI após uma hora de processamento computacional.

Os resultados da Tabela 6.2 mostram que a *solução GUROBI* esteve, em média, 4,25% acima do valor do limite inferior (*lower bound*), a solução obtida pela meta-heurística VNS esteve, em média, 5,24% acima, e as soluções obtidas pelas meta-heurísticas VNS-LR e VNDS estiveram, em média, 4,37% acima do limite inferior.

A meta-heurística VNDS foi 5,09% mais rápida do que a meta-heurística VNS e

2,07% mais rápida do que a meta-heurística VNS-LR. A meta-heurística VNS-LR, por sua vez, foi 3,09% mais rápida do que a meta-heurística VNS.

Tabela 6.2 – Resultados obtidos para instâncias com 120 objetos.

Lower Bound	**Solução GUROBI**		**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
	FO	**GAP (%)**	**FO**	**Tempo (s)**	**FO**	**Tempo (s)**	**FO**	**Tempo (s)**
4.529	4.740	4,43	4.720	17,52	4.720	16,90	4.720	16,57
4.611	4.760	2,04	4.820	14,34	4.760	13,94	4.760	13,63
4.348	4.520	2,62	4.480	8,22	4.480	7,94	4.480	7,81
4.662	4.740	1,27	4.840	6,24	4.740	6,02	4.740	5,87
4.706	4.900	3,62	4.960	8,47	4.900	8,17	4.900	7,97
4.558	4.640	1,17	4.720	8,16	4.640	7,95	4.640	7,78
4.567	4.680	2,33	4.720	6,98	4.680	6,74	4.680	6,59
4.668	4.860	2,82	4.880	8,32	4.860	8,03	4.860	7,93
4.785	4.920	2,17	4.980	17,77	4.920	17,09	4.920	17,04
4.396	4.600	4,07	4.700	7,58	4.600	7,35	4.600	7,19
4.915	5.140	1,69	5.220	16,60	5.140	16,23	5.140	15,90
4.638	4.880	2,88	4.900	8,02	4.900	7,83	4.900	7,65
4.595	4.860	3,18	4.820	8,46	4.820	8,17	4.820	8,07
4.609	4.900	2,13	5.000	10,59	5.000	10,18	5.000	9,99
4.718	4.900	1,97	4.980	7,61	4.900	7,42	4.900	7,27
4.545	4.760	2,45	4.760	9,59	4.760	9,28	4.760	9,05
4.920	5.340	1,94	5.340	20,50	5.340	19,68	5.340	19,29
4.929	5.280	1,50	5.260	8,75	5.260	8,55	5.260	8,33
4.643	4.800	2,78	4.920	16,56	4.920	16,21	4.920	15,74
4.686	4.780	1,53	4.900	8,48	4.780	8,26	4.780	7,99
93.028	**97.000**	**2,43**	**97.920**	**218,76**	**97.120**	**211,95**	**97.120**	**207,66**

Os dados apresentados na Tabela 6.3 consideram instâncias com 250 objetos a serem alocados e foram considerados como instâncias de médio porte. A coluna *solução GUROBI* apresentada na Tabela 6.3 apresenta os valores das melhores soluções encontradas dentro de uma hora de processamento e a coluna *GAP* apresenta a porcentagem que esta solução está acima do limite inferior calculado pelo GUROBI.

Os resultados da Tabela 6.3 mostram que os valores das soluções obtidas pelo *GUROBI* estiveram, em média, 5,35% acima do valor do limite inferior (*lower bound*); os valores das soluções obtidas pela meta-heurística VNS estiveram, em média, 5,63% acima; e

os valores das soluções obtidas pela meta-heurística VNS-LR estiveram, em média, 5,26% acima; e os valores das soluções obtidas pela meta-heurística VNDS estiveram, em média, 5,16% acima do limite inferior.

A meta-heurística VNDS foi 5,02% mais rápida do que a meta-heurística VNS e 2,08% mais rápida do que a meta-heurística VNS-LR. A meta-heurística VNS-LR, por sua vez, foi 3,00% mais rápida do que a meta-heurística VNS.

Tabela 6.3 – Resultados obtidos para instâncias com 250 objetos.

Lower Bound	**Solução GUROBI**		**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
	FO	**GAP (%)**	**FO**	**Tempo (s)**	**FO**	**Tempo (s)**	**FO**	**Tempo (s)**
9.461	9.920	4,37	9.960	42,60	9.920	41,33	9.920	40,32
9.506	9.800	2,73	10.000	26,50	9.800	25,95	9.800	25,08
9.736	10.320	4,20	10.400	42,23	10.400	40,76	10.320	39,78
9.544	9.960	3,91	9.980	39,90	9.980	38,70	9.980	38,15
9.658	10.080	3,92	10.140	57,40	10.080	55,90	10.080	54,21
9.679	10.280	4,54	10.240	52,22	10.240	50,26	10.240	50,08
9.698	10.060	3,34	10.140	42,16	10.140	41,00	10.140	39,99
9.867	10.500	5,75	10.460	63,30	10.460	61,22	10.460	60,17
10.071	10.660	5,27	10.620	52,46	10.620	50,84	10.620	49,74
9.619	9.980	2,98	10.140	23,54	9.980	23,05	9.980	22,51
10.019	10.520	4,99	10.580	47,18	10.520	45,73	10.520	44,71
9.668	10.320	3,41	10.440	42,27	10.320	40,80	10.320	40,31
10.073	10.680	3,32	10.620	68,82	10.620	66,31	10.620	65,56
9.788	10.520	4,54	10.320	44,64	10.320	43,22	10.320	42,56
9.520	9.900	5,37	10.040	51,26	10.040	49,92	10.040	48,42
10.062	10.800	4,75	10.640	44,02	10.640	42,88	10.640	41,54
9.265	9.660	4,57	9.700	43,13	9.700	42,11	9.700	40,67
9.520	9.920	3,03	9.980	62,03	9.920	60,18	9.920	59,48
9.571	10.160	4,38	10.280	64,74	10.160	62,26	10.160	61,74
9.730	10.420	5,18	10.300	47,25	10.420	45,83	10.300	44,72
194.055	**204.460**	**4,23**	**204.980**	**957,65**	**204.280**	**928,21**	**204.080**	**909,75**

Os dados apresentados na Tabela 6.4 consideram as instâncias com 500 objetos a serem alocados e foram considerados como instâncias de médio porte. A coluna *solução GUROBI* apresentada na Tabela 6.4 mostra os valores das melhores soluções encontradas dentro de uma hora e a coluna *GAP* apresenta a porcentagem que esta solução está acima do

limite inferior calculado pelo GUROBI.

Tabela 6.4 – Resultados obtidos para instâncias com 500 objetos.

Lower Bound	**Solução GUROBI**		**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
	FO	**GAP (%)**	**FO**	**Tempo (s)**	**FO**	**Tempo (s)**	**FO**	**Tempo (s)**
18.967	20.640	6,29	19.840	516,07	19.840	502,24	19.840	491,56
19.281	21.820	11,29	20.400	390,17	20.400	375,07	20.400	371,44
19.338	22.080	8,06	20.900	338,10	20.900	327,08	20.900	320,72
19.566	21.680	10,46	20.620	299,83	20.620	290,15	20.620	287,18
19.690	22.520	12,58	20.700	279,73	20.700	270,72	20.700	263,45
19.688	21.720	5,82	21.360	197,99	21.360	191,14	21.360	186,59
19.861	22.540	11,42	21.080	460,15	21.080	442,16	21.080	432,63
19.582	21.780	8,40	20.940	639,94	20.940	619,46	20.940	610,69
18.785	21.360	8,36	20.280	332,35	20.280	322,35	20.280	315,60
19.301	21.600	5,40	20.240	406,67	20.240	391,58	20.240	386,05
19.110	21.880	5,51	20.200	518,19	20.200	506,53	20.200	493,94
19.145	21.860	8,98	20.320	240,10	20.320	231,07	20.320	230,21
19.067	21.500	8,79	20.120	403,21	20.120	393,09	20.120	384,90
18.776	20.460	10,34	19.460	194,05	19.460	187,47	19.460	185,65
19.490	22.300	8,41	20.760	274,35	20.760	263,40	20.760	259,78
19.212	21.060	5,87	20.000	263,98	20.000	257,83	20.000	250,36
19.296	21.140	10,26	20.300	920,82	20.300	891,91	20.300	875,42
18.953	20.400	6,46	19.800	391,00	19.800	375,48	19.800	368,79
19.324	21.980	8,65	20.500	338,97	20.500	329,11	20.500	324,94
18.780	21.460	9,56	19.840	407,75	19.840	398,86	19.840	388,34
385.212	**431.780**	**8,55**	**407.660**	**7.813,42**	**407.660**	**7.566,69**	**407.660**	**7.428,24**

Os resultados da Tabela 6.4 mostram que os valores das soluções encontradas pelo *GUROBI* estiveram, em média, 12,08% acima do valor do limite inferior (l*ower bound*); as meta-heurísticas VNS, VNS-LR e VNDS encontraram soluções com os mesmos valores que estiveram, em média, 5,82% acima do limite inferior. Uma possível razão para as três meta-heurísticas terem encontrado soluções com valores iguais aos obtidos pelo GUROBI pode ser uma solução ótima local da qual nenhuma das meta-heurísticas conseguiu escapar ou pode ser a solução ótima global, no entanto não foi possível comprovar o motivo.

A meta-heurística VNDS foi 4,94% mais rápida do que a meta-heurística VNS e 1,80% mais rápida do que a meta-heurística VNS-LR. A meta-heurística VNS-LR, por sua

vez, foi 3,20% mais rápida do que a meta-heurística VNS.

Os resultados da Tabela 6.5 são relativos às instâncias de grande porte, com 1.000 cargas a serem alocados aos veículos. Para esses, foi impossível carregar o modelo matemático no GUROBI, tendo em vista a falta de memória, mesmo configurando o GUROBI para utilizar o espaço disponível no disco rígido (utilizar a partição *swap*) para diminuir a necessidade de utilização da memória RAM do computador utilizado nos testes. Isso evidencia a complexidade computacional do problema tratado, em particular o elevado número de variáveis de decisões e de restrições.

Tabela 6.5 – Resultados obtidos para instâncias com 1.000 objetos.

Lower Bound	**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
	FO	**Tempo (s)**	**FO**	**Tempo (s)**	**FO**	**Tempo (s)**
38.249	40.600	3.081,60	40.600	2.987,61	40.600	2.943,24
38.904	41.100	2.355,87	41.100	2.282,60	41.100	2.241,37
39.379	41.880	1.219,76	41.880	1.186,83	41.880	1.157,06
39.443	42.060	2.296,82	42.060	2.232,05	42.060	2.190,94
38.087	40.080	1.301,93	40.080	1.255,71	40.080	1.225,64
38.255	40.600	1.606,11	40.600	1.549,25	40.600	1.516,33
37.843	40.100	2.711,80	40.100	2.644,55	40.100	2.582,99
38.703	41.800	1.486,76	41.800	1.439,33	41.800	1.425,51
38.249	40.680	2.879,11	40.680	2.800,22	40.680	2.757,04
38.105	40.920	1.087,78	40.920	1.060,37	40.920	1.032,52
38.336	40.580	3.000,51	40.580	2.886,19	40.580	2.857,99
38.449	40.820	1.412,08	40.820	1.381,72	40.820	1.351,22
37.655	39.720	1.277,21	39.720	1.239,02	39.720	1.204,79
37.946	40.700	955,74	40.700	927,45	40.700	912,73
37.813	40.420	1.540,18	40.420	1.494,13	40.420	1.464,56
38.573	40.960	1.191,37	40.960	1.148,12	40.960	1.143,24
38.690	41.320	2.945,28	41.320	2.844,85	41.320	2.805,67
38.764	41.300	1.017,08	41.300	987,58	41.300	975,28
38.226	40.600	2.873,54	40.600	2.762,05	40.600	2.708,89
38.336	41.220	1.402,14	41.220	1.364,00	41.220	1.326,00
768.005	**817.460**	**37.642,67**	**817.460**	**36.473,64**	**817.460**	**35.823,00**

As meta-heurísticas VNS, VNS-LR e VNDS, propostas neste trabalho, conseguiram obter soluções para os problemas relativos às instâncias de grande porte com um

tempo computacional, aproximadamente entre 15 e 52 minutos (o tempo indicado na Tabela 6.5 está em segundos).

Os resultados da Tabela 6.5 mostram que as meta-heurísticas VNS, VNS-LR e VNDS encontraram a mesma solução que esteve, em média, 6,44% acima do limite inferior.

A meta-heurística VNDS foi 4,85% mais rápida do que a meta-heurística VNS e 1,84% mais rápida do que a meta-heurística VNS-LR. A meta-heurística VNS-LR, por sua vez, foi 3,06% mais rápida do que a meta-heurística VNS. Em todos os resultados apresentados nas Tabelas 6.2, 6.3, 6.4 e 6.5 a meta-heurística VNDS foi a mais rápida (entre 4,85% e 5,09% do que a meta-heurística VNS e entre 1,84% e 2,08% do que a meta-heurística VNS-LR) e sempre obteve as soluções mais próximas do limite inferior. Desta forma, podemos concluir que a meta-heurística VNDS foi mais eficiente do que as meta-heurísticas VNS-LR e VNS para o presente conjunto de dados.

6.3.1. Variação nos Resultados Obtidos

Tendo em vista a característica aleatória das meta-heurísticas VNS, VNS-LR e VNDS, em que um vizinho da solução corrente é obtido randomicamente (Figuras 2.4, 2.6 e 2.7), cada instância de teste foi submetida a cinco execuções, a fim de avaliar a sua robustez. Os resultados apresentados anteriormente nas Tabelas 6.2 a 6.5 correspondem ao melhor resultado obtido dentre os cinco processamentos.

A Tabela 6.6 apresenta o desvio padrão do valor da função objetivo entre as cinco execuções para cada uma das instâncias de teste com 120 objetos. O desvio padrão entre as soluções obtidas pela meta-heurística VNS varia entre 0,36% e 1,18%; o desvio padrão entre as soluções obtidas pela meta-heurística VNS-LR varia entre 0,22% e 1,00%; e o desvio padrão entre as soluções obtidas pela meta-heurística VNDS varia entre 0,16% e 0,73%. Como a meta-heurística VNDS possui os menores valores para o desvio padrão, ela é considerada uma meta-heurística mais robusta do que a VNS ou a VNS-LR para a resolução do BiD-VSBPP nestas instâncias de teste com 120 objetos.

Tabela 6.6 – Variação nos resultados obtidos para instâncias com 120 objetos.

Instância	VNS	VNS-LR	VNDS
	Desvio Padrão	Desvio Padrão	Desvio Padrão
1	1,18%	1,00%	0,49%
2	0,45%	0,98%	0,70%
3	1,05%	0,22%	0,18%
4	0,96%	0,68%	0,16%
5	0,64%	0,89%	0,72%
6	0,68%	0,93%	0,73%
7	0,88%	0,65%	0,66%
8	0,94%	0,24%	0,49%
9	0,90%	0,95%	0,37%
10	0,36%	0,83%	0,53%
11	0,89%	0,73%	0,39%
12	0,90%	0,70%	0,49%
13	0,93%	0,60%	0,60%
14	0,84%	0,22%	0,65%
15	0,94%	0,88%	0,43%
16	1,18%	0,81%	0,36%
17	0,58%	0,72%	0,54%
18	0,82%	0,56%	0,71%
19	0,92%	0,84%	0,30%
20	0,37%	0,87%	0,27%

A Tabela 6.7 apresenta o desvio padrão do valor da função objetivo entre as cinco execuções para cada uma das instâncias de teste com 250 objetos. O desvio padrão entre as soluções obtidas pela meta-heurística VNS varia entre 0,17% e 0,57%; o desvio padrão entre as soluções obtidas pela meta-heurística VNS-LR varia entre 0,16% e 0,50%; e o desvio padrão entre as soluções obtidas pela meta-heurística VNDS varia entre 0,24% e 0,47%. Como a meta-heurística VNS-LR possui os menores valores para o desvio padrão, ela é considerada uma meta-heurística mais robusta do que a VNS ou a VNDS para a resolução do BiD-VSBPP nestas instâncias de teste com 250 objetos.

Tabela 6.7 – Variação nos resultados obtidos para instâncias com 250 objetos.

Instância	VNS	VNS-LR	VNDS
	Desvio Padrão	Desvio Padrão	Desvio Padrão
1	0,40%	0,31%	0,45%
2	0,42%	0,46%	0,31%
3	0,17%	0,36%	0,47%
4	0,46%	0,49%	0,36%
5	0,47%	0,33%	0,45%
6	0,46%	0,16%	0,25%
7	0,51%	0,16%	0,36%
8	0,50%	0,44%	0,32%
9	0,29%	0,22%	0,25%
10	0,33%	0,33%	0,46%
11	0,45%	0,30%	0,28%
12	0,45%	0,50%	0,44%
13	0,42%	0,44%	0,27%
14	0,44%	0,25%	0,24%
15	0,34%	0,49%	0,30%
16	0,31%	0,27%	0,26%
17	0,42%	0,46%	0,42%
18	0,57%	0,41%	0,31%
19	0,40%	0,26%	0,27%
20	0,42%	0,35%	0,38%

A Tabela 6.8 apresenta o desvio padrão do valor da função objetivo entre as cinco execuções para cada uma das instâncias de teste com 500 objetos. O desvio padrão entre as soluções obtidas pela meta-heurística VNS varia entre 0,18% e 0,38%; o desvio padrão entre as soluções obtidas pela meta-heurística VNS-LR varia entre 0,17% e 0,36%; e o desvio padrão entre as soluções obtidas pela meta-heurística VNDS varia entre 0,15% e 0,30%. Como a meta-heurística VNDS possui os menores valores para o desvio padrão, ela é considerada uma meta-heurística mais robusta do que a VNS ou a VNS-LR para a resolução do BiD-VSBPP nestas instâncias de teste com 500 objetos.

Tabela 6.8 – Variação nos resultados obtidos para instâncias com 500 objetos.

Instância	VNS	VNS-LR	VNDS
	Desvio Padrão	Desvio Padrão	Desvio Padrão
1	0,18%	0,21%	0,25%
2	0,28%	0,31%	0,23%
3	0,36%	0,36%	0,19%
4	0,35%	0,19%	0,21%
5	0,38%	0,26%	0,30%
6	0,29%	0,20%	0,28%
7	0,33%	0,24%	0,22%
8	0,21%	0,19%	0,22%
9	0,22%	0,34%	0,18%
10	0,28%	0,20%	0,26%
11	0,35%	0,19%	0,18%
12	0,25%	0,29%	0,17%
13	0,22%	0,32%	0,28%
14	0,30%	0,22%	0,25%
15	0,25%	0,26%	0,20%
16	0,26%	0,28%	0,19%
17	0,32%	0,29%	0,28%
18	0,31%	0,17%	0,19%
19	0,36%	0,30%	0,22%
20	0,36%	0,22%	0,15%

A Tabela 6.9 apresenta o desvio padrão do valor da função objetivo entre as cinco execuções para cada uma das instâncias de teste com 1.000 objetos. O desvio padrão entre as soluções obtidas pela meta-heurística VNS varia entre 0,07% e 0,23%; o desvio padrão entre as soluções obtidas pela meta-heurística VNS-LR varia entre 0,05% e 0,22%; e o desvio padrão entre as soluções obtidas pela meta-heurística VNDS varia entre 0,05% e 0,21%. Como as meta-heurísticas VNS-LR e VNDS possuem os menores valores para o desvio padrão, elas são consideradas meta-heurísticas mais robusta do que a VNS para a resolução do BiD-VSBPP nestas instâncias de teste com 1.000 objetos.

Tabela 6.9 – Variação nos resultados obtidos para instâncias com 1.000 objetos.

Instância	VNS	VNS-LR	VNDS
	Desvio Padrão	Desvio Padrão	Desvio Padrão
1	0,13%	0,05%	0,17%
2	0,23%	0,07%	0,18%
3	0,13%	0,12%	0,14%
4	0,18%	0,18%	0,09%
5	0,18%	0,13%	0,05%
6	0,20%	0,19%	0,21%
7	0,18%	0,12%	0,17%
8	0,20%	0,16%	0,09%
9	0,17%	0,16%	0,19%
10	0,16%	0,15%	0,18%
11	0,20%	0,05%	0,17%
12	0,13%	0,09%	0,19%
13	0,15%	0,21%	0,20%
14	0,15%	0,14%	0,07%
15	0,17%	0,18%	0,15%
16	0,11%	0,22%	0,10%
17	0,15%	0,12%	0,21%
18	0,07%	0,14%	0,05%
19	0,21%	0,20%	0,21%
20	0,18%	0,14%	0,15%

Os resultados apresentados anteriormente nas Tabelas 6.6 a 6.9 apresentaram o desvio padrão para cada uma das instâncias com 120, 250, 500 ou 1.000 objetos. Observe que o maior valor do desvio padrão foi 1,18% e o menor foi 0,05% isto demonstra que as três meta-heurísticas implementadas obtiveram poucas variações entre as soluções obtidas mostrando a robustez das meta-heurísticas VNS, VNS-LR e VNDS. A robustez da meta-heurística VNDS se destacou nas instâncias com 120, 500 ou 1.000 objetos e a meta-heurística VNS-LR mostrou maior robustez nas instâncias com 250 ou 1.000 objetos.

6.4. EXPERIMENTOS COMPUTACIONAIS COM *BINS* MODIFICADOS

Os experimentos computacionais apresentados nesta seção utilizam dados gerados aleatoriamente que visam simular instâncias reais nas quais cada objeto possui baixa densidade, isto é, o volume é superior ao peso. Para cada objeto foi sorteado um peso entre 600 e 1.500kg e um volume entre 2 e 6 vezes o valor do peso sorteado. A capacidade dos *bins* também foi alterada visando refletir problemas reais. Desta forma, é possível explorar o comportamento das meta-heurísticas VNS, VNS-LR e VNDS na obtenção de soluções em problemas com dados de entrada com características distintas. A Tabela 6.10 apresenta os custos e as capacidades de cada veículo.

Tabela 6.10 – Custo e Capacidade dos veículos.

	Grande	**Médio**	**Pequeno**
Custo ($)	360	200	120
Capacidade (kg)	25.000	12.000	6.000
Capacidade (m³)	100	48	24

O *bin* do tipo médio possui capacidade de 12.000 quilogramas e 48 metros cúbicos e custo de 200 unidades monetárias. O *bin* do tipo grande possui um custo 80% maior e a capacidade 108% superior enquanto o *bin* do tipo pequeno possui um custo 40% inferior e a capacidade 50% menor do que o *bin* do tipo médio. Dessa forma, os veículos possuem custos unitários decrescentes com o aumento da capacidade, conforme ocorre na prática do transporte.

A função de avaliação é a mesma função da seção 6.3 e pode ser calculada através da expressão (6.8). Os testes foram realizados em um total de quatro conjuntos de dados com problemas com o número de objetos variando entre 120 e 1.000. Para cada instância do problema, o programa desenvolvido em C++ foi executado cinco vezes e as melhores soluções obtidas para cada instância estão apresentadas nas Tabelas 6.11, 6.12, 6.13 e 6.14. O Apêndice E possui os resultados detalhados com os valores das soluções ótimas obtidas para cada instância deste conjunto de teste.

Os resultados apresentados na Tabela 6.11 mostram os valores das soluções

obtidas para as menores instâncias testadas (120 objetos). A coluna *Solução GUROBI* apresenta o valor da solução obtida pelo pacote de otimização GUROBI ao resolver o modelo matemático (6.1)-(6.6). A coluna *Solução VNS* apresenta os valores das soluções obtidas pela meta-heurística VNS, a coluna *Solução VNS-LR* apresenta os valores das soluções obtidas pela meta-heurística VNS-LR e a coluna *Solução VNDS* apresenta os valores das soluções obtidas pela meta-heurística VNDS. A coluna *FA* apresenta o valor da função de avaliação e a coluna *Tempo* mostra o tempo computacional, em segundos, utilizado para obter a solução apresentada.

Tabela 6.11 – Resultados obtidos para instâncias com 120 objetos.

Instância	**Solução ótima**		**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
	FA	**Tempo (s)**	**FA**	**Tempo (s)**	**FA**	**Tempo (s)**	**FA**	**Tempo (s)**
1	2.720	1,30	2.720	3,89	2.720	3,77	2.720	3,58
2	2.840	1,63	2.840	4,74	2.840	4,60	2.840	4,37
3	2.840	1,06	2.840	4,65	2.840	4,51	2.840	4,28
4	3.080	1,94	3.080	4,86	3.080	4,69	3.080	4,46
5	2.840	1,88	2.840	4,92	2.840	4,77	2.840	4,53
6	2.880	0,83	2.880	4,07	2.880	3,93	2.880	3,75
7	2.880	1,89	2.880	3,46	2.880	3,37	2.880	3,20
8	2.720	1,21	2.720	6,17	2.720	6,01	2.720	5,74
9	2.880	0,10	2.880	3,43	2.880	3,33	2.880	3,15
10	2.720	2,81	2.720	3,94	2.720	3,84	2.720	3,64
11	2.880	0,29	2.880	3,24	2.880	3,14	2.880	2,97
12	2.720	1,73	2.720	5,68	2.720	5,50	2.720	5,24
13	2.840	1,44	2.840	3,64	2.840	3,55	2.840	3,38
14	2.640	1,59	2.640	3,41	2.640	3,30	2.640	3,14
15	2.880	1,11	2.880	4,74	2.880	4,59	2.880	4,37
16	2.720	1,95	2.720	4,63	2.720	4,50	2.720	4,27
17	2.720	1,69	2.720	4,31	2.720	4,20	2.720	4,00
18	2.720	1,18	2.720	3,25	2.720	3,14	2.720	2,99
19	2.720	0,38	2.720	4,20	2.720	4,09	2.720	3,87
20	2.880	1,17	2.880	4,32	2.880	4,17	2.880	3,96
Total	**56.120**	**27,18**	**56.120**	**85,54**	**56.120**	**83,00**	**56.120**	**78,89**

Em todas as instâncias com 120 objetos os valores das soluções obtidas pelas meta-heurísticas VNS, VNS-LR e VNDS foram iguais aos valores das soluções ótimas

obtidas pelo pacote de otimização GUROBI, o que comprova que as meta-heurísticas propostas foram eficazes para encontrar as soluções ótimas para estas instâncias com 120 objetos.

A meta-heurística VNDS foi 7,78% mais rápida do que a meta-heurística VNS e 4,95% mais rápida do que a meta-heurística VNS-LR. A meta-heurística VNS-LR, por sua vez, foi 2,98% mais rápida do que a meta-heurística VNS. Entretanto a meta-heurística VNDS foi quase 3 vezes (290,21%) mais lenta do que o pacote de otimização GUROBI para obter a solução ótima. Desta forma, pode-se concluir que apesar de eficazes para obterem as soluções ótimas, para as instâncias com 120 objetos, as meta-heurísticas VNS, VNS-LR e VNDS propostas não foram eficientes em comparação com o GUROBI.

Tabela 6.12 – Resultados obtidos para instâncias com 250 objetos.

Instância	**Solução ótima**		**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
	FA	**Tempo (s)**	**FA**	**Tempo (s)**	**FA**	**Tempo (s)**	**FA**	**Tempo (s)**
1	5.720	64,76	5.720	10,54	5.720	10,24	5.720	9,97
2	5.760	0,58	5.760	12,31	5.760	11,89	5.760	11,66
3	5.760	5,24	5.760	11,16	5.760	10,86	5.760	10,64
4	5.720	15,56	5.720	10,27	5.720	9,96	5.720	9,72
5	5.520	8,26	5.520	10,43	5.520	10,16	5.520	9,92
6	5.600	31,73	5.600	10,05	5.600	9,79	5.600	9,59
7	5.760	5,11	5.760	11,24	5.760	10,89	5.760	10,66
8	5.880	32,89	5.880	14,95	5.880	14,51	5.880	14,19
9	5.520	10,97	5.520	11,55	5.520	11,15	5.520	11,00
10	5.520	32,52	5.520	11,62	5.520	11,24	5.520	11,08
11	5.760	1,50	5.760	10,05	5.760	9,75	5.760	9,58
12	5.600	19,38	5.600	11,67	5.600	11,30	5.600	11,06
13	5.720	45,10	5.720	10,41	5.720	10,07	5.720	9,84
14	5.720	48,12	5.720	10,54	5.720	10,27	5.720	10,03
15	5.760	4,79	5.760	11,41	5.760	11,12	5.760	10,88
16	5.760	31,49	5.760	10,88	5.760	10,58	5.760	10,29
17	5.880	9,73	5.880	12,97	5.880	12,62	5.880	12,39
18	5.600	8,48	5.600	13,44	5.600	13,00	5.600	12,76
19	5.720	43,44	5.720	10,63	5.720	10,36	5.720	10,05
20	5.520	32,03	5.520	9,38	5.520	9,08	5.520	8,90
Total	**113.800**	**451,67**	**113.800**	**225,49**	**113.800**	**218,82**	**113.800**	**214,23**

Na Tabela 6.12 são apresentados os resultados obtidos para as instâncias com 250 objetos. Em todas as instâncias com 250 objetos as soluções obtidas pelas meta-heurísticas VNS, VNS-LR e VNDS foram iguais às soluções ótimas obtidas pelo pacote de otimização GUROBI, o que comprova que as meta-heurísticas propostas foram eficazes para encontrarem as soluções ótimas para estas instâncias com 250 objetos.

A meta-heurística VNDS foi 4,99% mais rápida do que a meta-heurística VNS e 2,10% mais rápida do que a meta-heurística VNS-LR. A meta-heurística VNS-LR, por sua vez, foi 2,96% mais rápida do que a meta-heurística VNS. A meta-heurística VNDS foi aproximadamente 2 vezes (210,84%) mais rápida do que o pacote de otimização GUROBI para obter a solução ótima.

Tabela 6.13 – Resultados obtidos para instâncias com 500 objetos.

Instância	Solução ótima		Solução VNS		Solução VNS-LR		Solução VNDS	
	FA	Tempo (s)	FA	Tempo (s)	FA	Tempo (s)	FA	Tempo (s)
1	11.280	260,81	11.280	68,77	11.280	66,87	11.280	65,44
2	11.360	144,57	11.360	53,20	11.360	51,65	11.360	50,49
3	10.760	310,14	10.760	42,25	10.760	41,11	10.760	40,15
4	11.280	145,15	11.280	54,69	11.280	52,80	11.280	52,15
5	11.360	293,14	11.360	44,64	11.360	43,52	11.360	42,41
6	11.160	44,15	11.160	44,93	11.160	43,54	11.160	42,66
7	11.480	125,28	11.480	54,76	11.480	53,04	11.480	51,83
8	11.520	61,96	11.520	79,60	11.520	76,96	11.520	76,00
9	11.520	9,37	11.520	55,17	11.520	53,54	11.520	52,18
10	11.640	237,30	11.640	47,76	11.640	46,29	11.640	45,16
11	11.160	299,22	11.160	45,02	11.160	43,88	11.160	42,60
12	11.360	276,41	11.360	39,98	11.360	38,74	11.360	37,89
13	11.520	8,41	11.520	49,33	11.520	47,65	11.520	47,06
14	11.280	117,14	11.280	67,16	11.280	65,26	11.280	63,65
15	11.280	124,37	11.280	48,14	11.280	46,75	11.280	45,93
16	11.000	163,29	11.000	41,47	11.000	40,24	11.000	39,53
17	11.360	319,31	11.360	41,31	11.360	40,21	11.360	39,31
18	11.360	218,20	11.360	57,41	11.360	55,77	11.360	54,45
19	10.920	199,27	10.920	47,43	10.920	46,07	10.920	45,08
20	11.480	175,46	11.480	44,68	11.480	43,34	11.480	42,45
Total	**226.080**	**3.532,93**	**226.080**	**1.027,70**	**226.080**	**997,24**	**226.080**	**976,41**

Na Tabela 6.13 são apresentados os resultados obtidos para as instâncias com 500 objetos. Em todas as instâncias com 500 objetos os valores das soluções obtidas pelas meta-heurísticas VNS, VNS-LR e VNDS foram iguais às soluções ótimas obtidas pelo pacote de otimização GUROBI, o que comprova que as meta-heurísticas propostas foram eficazes para encontrar as soluções ótimas para estas instâncias com 500 objetos.

A meta-heurística VNDS foi 4,99% mais rápida do que a meta-heurística VNS e 2,09% mais rápida do que a meta-heurística VNS-LR. A meta-heurística VNS-LR, por sua vez, foi 2,96% mais rápida do que a meta-heurística VNS. A meta-heurística VNDS foi aproximadamente 3,6 vezes (361,83%) mais rápida do que o pacote de otimização GUROBI para obter a solução ótima.

Tabela 6.14 – Resultados obtidos para instâncias com 1.000 objetos.

Instância	**Solução ótima**		**Solução VNS**		**Solução VNS-LR**		**Solução VNDS**	
	FA	**Tempo (s)**	**FA**	**Tempo (s)**	**FA**	**Tempo (s)**	**FA**	**Tempo (s)**
1	22.520	3.598,71	22.520	140,64	22.520	136,25	22.520	133,21
2	22.320	296,64	22.320	127,14	22.320	123,85	22.320	121,19
3	22.640	3.707,09	22.640	189,89	22.640	183,72	22.640	180,62
4	22.320	66,12	22.320	140,97	22.320	137,02	22.320	134,45
5	22.320	308,78	22.320	182,30	22.320	177,17	22.320	173,51
6	22.440	2.002,46	22.440	143,03	22.440	139,11	22.440	136,16
7	22.680	2.475,42	22.680	122,43	22.680	118,19	22.680	116,78
8	22.800	1.626,09	22.800	154,61	22.800	150,06	22.800	146,35
9	22.320	55,65	22.320	121,44	22.320	118,41	22.320	115,67
10	22.320	2.401,69	22.320	155,86	22.320	150,64	22.320	147,71
11	22.280	2.016,55	22.280	249,29	22.280	242,08	22.280	236,62
12	22.160	2.942,43	22.160	214,16	22.160	207,90	22.160	203,25
13	22.320	66,88	22.320	136,50	22.320	132,56	22.320	129,91
14	22.320	3.094,80	22.320	170,38	22.320	165,01	22.320	162,64
15	22.440	32.334,11	22.440	149,28	22.440	144,92	22.440	141,46
16	22.320	61,52	22.320	145,29	22.320	140,99	22.320	137,56
17	22.320	69,73	22.320	200,83	22.320	195,77	22.320	190,57
18	22.680	62,87	22.680	143,69	22.680	138,78	22.680	136,39
19	22.440	4.736,12	22.440	143,38	22.440	139,48	22.440	135,71
20	22.520	2.269,91	22.520	131,49	22.520	127,53	22.520	124,88
Total	**448.480**	**64.193,59**	**448.480**	**3.162,58**	**448.480**	**3.069,43**	**448.480**	**3.004,65**

A Tabela 6.14 apresenta os resultados obtidos para instâncias com 1.000 objetos. Em 19 das 20 instâncias testadas as meta-heurísticas VNS, VNS-LR e VNDS propostas neste trabalho obtiveram as soluções ótimas. Em uma instância, o GUROBI não conseguiu obter uma solução ótima devido à falta de capacidade computacional do computador utilizado, isto é, o consumo de memória RAM para a resolução desta instância foi superior à 8 GB somados com 8 GB de memória *swap* (espaço reservado no disco rígido como área de troca). Apesar das meta-heurísticas VNS, VNS-LR e VNDS terem obtido soluções para esta instância, não é possível afirmar que a solução obtida seja ótima.

A meta-heurística VNDS foi 4,99% mais rápida do que a meta-heurística VNS e 2,11% mais rápida do que a meta-heurística VNS-LR. A meta-heurística VNS-LR, por sua vez, foi 2,95% mais rápida do que a meta-heurística VNS.

6.4.1. Variação nos Resultados Obtidos

A Tabela 6.15 apresenta a variação nos resultados obtidos (Tabelas 6.11 a 6.14) pelas meta-heurísticas VNS, VNS-LR e VNDS propostas. Para cada instância de teste o programa desenvolvido em C++ com a implementação computacional das meta-heurísticas propostas foi executado 5 vezes e em todas as execuções sempre foi obtido o mesmo resultado; desta forma, o desvio padrão de todas foi nulo, o que comprova a elevada robustez das meta-heurísticas propostas.

Tabela 6.15 – Variação nos resultados obtidos.

Número de objetos	**VNS**	**VNS-LR**	**VNDS**
	Desvio Padrão	**Desvio Padrão**	**Desvio Padrão**
120	0,00%	0,00%	0,00%
250	0,00%	0,00%	0,00%
500	0,00%	0,00%	0,00%
1.000	0,00%	0,00%	0,00%

6.5. CONCLUSÕES

Os resultados da seção 6.3 indicam que as meta-heurísticas propostas permitiram obter os melhores resultados para as instâncias acima de 500 objetos. E, nos demais testes (instâncias inferiores a 250 objetos), as soluções obtidas pelas três meta-heurísticas propostas sempre tiveram valores próximos aos das soluções obtidas pelo GUROBI (menos de 1% acima para instâncias com 120 objetos e menos de 0,3% acima para instâncias com 250 objetos), o que indica que as meta-heurísticas propostas atingiram resultados de excelente qualidade com tempos de processamento muito reduzidos. As meta-heurísticas implementadas também se mostraram mais rápida que a implementação do GUROBI para resolver o modelo matemático e nos problemas de grande porte (1.000 objetos) o GUROBI não conseguiu resolver o modelo matemático.

Os valores das soluções obtidas pelo GUROBI estiveram, em média, entre 4,25% e 12,08% acima do limitante inferior (*lower bound*), a meta-heurística VNS obteve soluções com valores entre 5,24% e 6,44% acima do limitante inferior e as meta-heurísticas VNS-LR e VNDS obtiveram soluções com valores entre 4,38% e 6,44% acima do limitante inferior.

A meta-heurística VNDS foi entre 1,80% e 2,08% mais rápida do que a meta-heurística VNS-LR e entre 4,85% e 5,10% mais rápida do que a meta-heurística VNS. O VNS-LR foi entre 3,00% e 3,20% mais rápida do que o VNS.

Os resultados da seção 6.4 indicam que as meta-heurísticas propostas permitiram obter soluções ótimas para 79 das 80 instâncias testadas e em uma instância o resultado obtido não pôde ser comparado devido ao pacote de otimização utilizado não ter conseguido obter uma solução viável.

Nas instâncias de menor porte (120 objetos) a meta-heurística VNDS foi 2,9 vezes mais lenta que o GUROBI. Nas instâncias de maior porte (250, 500 ou 1.000 objetos) a meta-heurística VNDS teve um desempenho superior ao GUROBI, conseguindo encontrar soluções ótimas em tempos computacionais, geralmente, inferiores ao do GUROBI.

A meta-heurística VNDS foi entre 1,80% e 4,97% mais rápida do que a meta-heurística VNS-LR e entre 4,91% e 7,80% mais rápida do que a meta-heurística VNS. O VNS-LR foi entre 2,95% e 3,16% mais rápida do que o VNS.

Os resultados referentes às instâncias de testes com 500 ou 1.000 objetos a serem

alocados (Tabelas 6.4 e 6.5) mostram que as três meta-heurísticas propostas obtiveram os mesmos valores para as soluções. Isto pode indicar que as soluções obtidas são soluções locais e que nenhuma meta-heurística conseguiu "escapar" de sua vizinhança ou existe a possibilidade da solução ser ótima.

Considerando que em 79 das 80 instâncias de teste com os *bins* modificados (seção 6.4), as meta-heurísticas propostas obtiveram as soluções ótimas e que os dados da Tabela 6.15 comprovam que o desvio padrão entre as soluções obtidas pelas meta-heurísticas é nulo, pode-se concluir que as meta-heurísticas propostas foram eficazes para obter as soluções ótimas para as instâncias de teste do BiD-VSBPP.

Para uma melhor análise da eficácia das meta-heurísticas propostas neste trabalho seria interessante a comparação com dados da literatura para poder determinar com exatidão se as meta-heurísticas propostas neste trabalho são eficazes para obter as soluções ou se as instâncias de testes utilizadas eram fáceis de serem resolvidas.

Inúmeras são as possíveis extensões para a continuidade desta pesquisa, tendo em vista o número reduzido de trabalhos na literatura, incluindo a investigação de novos métodos de solução e a aplicação e validação dos métodos propostos a problemas reais que envolvem o agrupamento de cargas no contexto da distribuição física de produtos.

Outra proposta para a continuidade deste trabalho é o estudo de novas estruturas de vizinhança para o problema de *bin-packing* bidimensional (volume e peso dos objetos) com frota heterogênea. Inclusive com novas instâncias de testes com variações nas capacidades (tanto peso, quanto volume) dos veículos e limitações nos tamanhos de frota disponível.

7. CONCLUSÕES

Esta pesquisa mostrou que as meta-heurísticas baseadas em busca em vizinhança variável propostas para: o PPHVT (Problema de Programação Integrada da Tabela de Horários, Veículos e Tripulações), o VSBPP (Problema de *Bin-packing* com *Bins* Heterogêneos) e o BiD-VSBPP (Problema Bidimensional de *Bin-packing* com *Bins* Heterogêneos) foram eficazes para, em várias instâncias testadas, obter soluções ótimas para o problema e para as demais instâncias há indicativos de que as soluções obtidas são de boa qualidade.

A meta-heurística de busca em vizinha variável com lista restrita (VNS-LR) possui uma estrutura de memória que permite escapar de ótimos locais. Esta estrutura de memória provou ser eficaz e permitiu que a implementação do VNS-LR conseguisse obter resultados melhores do que a implementação de uma meta-heurística de busca em vizinhança variável pura (VNS).

A implementação da meta-heurística de busca decomposta em vizinhança variável (VNDS) foi a meta-heurística que obteve os valores dos melhores resultados. A implementação do VNDS foi a mais robusta (menor variação entre os resultados), mais rápida (superou as meta-heurísticas VNS e VNS-LR) e foi eficaz para obter soluções de boa qualidade (inclusive obtendo soluções ótimas em várias instâncias).

A meta-heurística VNDS comprovou ser uma meta-heurística de boa adaptação para resolver problemas complexos de operação de transportes como o PPHVT (Problema de Programação Integrada da Tabela de Horários, Veículos e Tripulações), o VSBPP (Problema de *Bin-Packing* com frota Heterogênea) ou o BiD-VSBPP (Problema Bi-dimensional de *Bin-Packing* com frota Heterogênea).

A revisão da literatura mostrou não haver trabalhos que abordem a programação dos veículos e tripulações de ônibus coletivo urbano integrada com a tabela de horários, isto é, permitindo a modificação nos horários de início das viagens para um melhor aproveitamento dos ônibus ociosos. A abordagem sequencial inversa (tripulação primeiro, veículo depois) demonstrou ser a abordagem ideal para minimizar o número de tripulações necessárias para operar uma frota de ônibus urbano. A abordagem integrada entre veículos, tripulações e tabela de horários mostrou ser a melhor abordagem para reduzir o número de veículos, o tempo de

terminal e o número de horas ociosas. Os resultados obtidos pelas meta-heurísticas foram próximos aos obtidos pelos métodos exatos implementados e testado neste trabalho.

Os resultados obtidos pelas meta-heurísticas propostas para a resolução do VSBPP e do BiD-VSBPP comprovam a eficácia do método proposto. No caso do VSBPP os resultados foram próximos ou iguais às soluções ótimas obtidas pelo GUROBI e foram melhores do que os resultados obtidos na literatura para as mesmas instâncias de teste. Para o BiD-VSBPP, as meta-heurísticas propostas obtiveram soluções ótimas em todas as instâncias nas quais foi possível obtê-las utilizando o pacote de otimização GUROBI e nas demais instâncias testadas há indícios que indiquem que as soluções obtidas sejam de boa qualidade. No entanto, ainda é necessário fazer comparações com dados da literatura para comprovar a eficácia das meta-heurísticas propostas para a resolução do BiD-VSBPP.

A proposta de continuidade desta pesquisa abordará a busca por outros complexos problemas de operação de transportes, que possam ser resolvidos utilizando uma meta-heurística baseada em busca em vizinhança variável como o VNDS.

Outra proposta de continuidade é a implementação e validação de um método de geração de colunas, que visa resolver de forma integrada a programação de veículos e tripulações (PPVT) e posteriormente resolver o PPHVT.

REFERÊNCIAS

AGUIAR, M. R. Avaliação dos objetivos de política tarifária sob o enfoque da elasticidade-tarifa e elasticidade-renda da demanda: o caso da RMSP. **Trabalhos Técnicos em Eventos da ANTP**. 2001. Disponível na internet via www. URL: http://portal.antp.org.br/Eventos/Congressos/13_ANTP/Trabalhos/012/html/012.html. Acessado em 01/03/2010.

AHUJA, R. K.; MEHLHORN, K.; ORLIN, J.; TARJAN, R. E. Faster algorithms for the shortest path problem. **Journal of the Association for Computing Machinery**, v. 37, n. 2, p. 213-223. 1990.

ALVES, C.; Carvalho, J. M. V. Accelerating Column Generation for Variable Sized Bin-Packing Problems, **European Journal of Operational Research**, v. 183, n. 3, p. 1333–1352. 2007.

ARABEYRE, J. P.; FEARNLEY, J.; STEIGER, F. C.; TEATHER, W. The airline crew scheduling problem: a survey. **Transportation Science**, n. 3, p. 140–163. 1969.

ARENALES, M.; ARMENTANO, V.; MORABITO, R.; YANASSE, H. **Pesquisa operacional para cursos de engenharia**. Campus Elsevier, Rio de Janeiro, 524p. 2006.

BALDO, T. A. **Geração de colunas para o problema de dimensionamento de lotes de produção com limitações de capacidade**. 2009. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemática e de Computação, ICMC – USP. São Carlos, São Paulo. 2009.

BALL, M.; BODIN, L.; DIAL, R. Experimentation with a computerized system for scheduling mass transit vehicles and crews. **Computer Scheduling of Public Transport, Urban Passenger Vehicle and Crew Scheduling**, p. 313–334. 1981.

BALL, M.; BODIN, L.; DIAL, R. A matching based heuristic for scheduling mass transit crews and vehicles. **Transportation Science**, n. 17, v. 1, p. 4–31. 1983.

BALL, M.; ROBERTS, A. A graph partitioning approach to airline crew scheduling. **Transportation Science**, n. 19, v. 2, p. 107–126. 1985

BARNHART, C.; JOHNSON, E. L.; NEMHAUSER, G. L.; SAVELSBERGH, M. P.; VANCE, P. H. Branch-and-price: column generation for solving huge integer programs. **Operations Research**, n. 46, p. 316-329. 1998

BLAIS, J.; LAMONT, J.; ROUSSEAU, J. The HASTUS vehicle and manpower scheduling systems at society de transport de la communate urbane de montreal. **Interfaces**, n. 20, v. 1, p. 26–42. 1990.

BODIN, L.; GOLDEN, B.; ASSAD, A.; BALL, M. Routing and scheduling of vehicles and crews – the state of the art. **Computers and Operations Research,** v. 10, n. 2, p. 63–211. 1983.

BURKARD, R. E.; ZHANG, G. Bounded space on-line variable-sized bin packing. **Acta Cybernetica** v. 13, p. 63-76. 1997.

CAETANO, D. J.; GUALDA, N. D. F. Um modelo integrado para a programação de voos e alocação de frotas. **Transportes** (Rio de Janeiro), v. 19, n. 2, p. 16-24. 2011.

CARVALHO, J. M. V. Exact solutions of bin-packing problems using column generation and branch-and-bound, **Annals of Operations Research**, v. 86, p. 629-659. 1999.

CEDER, A.; FJORNES, B.; STERN, H. OPTIBUS: a scheduling package. **Computer-aided Transit Scheduling**, Lecture Notes in Economics and Mathematical Systems, v. 308, p. 212–225. 1988.

CNI-IBOPE (2011). Disponível na internet via www. URL: http://www.correio24horas.com.br/noticias/detalhes/detalhes-2/artigo/transporte-rouba-tempo-do-brasileiro-segundo-pesquisa/. Acessado em 16/11/2013.

COFFMAN, E. G.; GAREY, M. R.; JOHNSON, D. S. Approximation algorithm for bin-packing: a survey. In: **Hochbaum D (eds) Approximation algorithm for NP-hard problems**. PWS Publishing, Boston, p. 46–93. 1997.

CORREIA, I.; GOUVEIA, L.; SALDANHA-DA-GAMA, F. Solving the variable size bin packing problem with discretized formulations. **Computers & Operations Research**, v. 35, p. 2103-2113. 2008.

CUNHA, C. B. **Contribuição à Modelagem de Problemas em Logística e Transportes**. Tese (livre docência), Escola Politécnica 1. ed. São Paulo: Universidade de São Paulo, 2006. 315 p.

CUNHA, C. B; MANIERI, G.; YOSHIZAKI, H. T. Y.; MALUTA, L.; HENRIQUES, L. R. S. Heurísticas para o Problema de *Bin-Packing* no Contexto da Distribuição Física de Produtos. In: **XL SBPO - 40º Simpósio Brasileiro de Pesquisa Operacional**, 2008, João Pessoa. Anais p. 712-723. 2008.

DA SILVA, A. C. L. **Estratégia para Divisão da Áreas de Estudo em Problemas Logísticos – Uso de Diagrama de Voronoi com Obstáculos**, Tese de Doutorado, Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, 2004.

DESROCHERS, M.; SOUMIS, F. A Column Generation approach to the urban transit crew scheduling problem. **Transportation Science**, v. 23, p. 1-13. 1989.

DESROCHERS, M; GILBERT, J.; SAUVE, M.; SOUMIS, F. CREW-OPT: Subproblem modeling in a column generation approach to urban crew scheduling. **Computer-Aided Transit Scheduling**, Desrochers, M. & Rousseau, J. M. (eds.), Spring, Berlin, p. 395-406. 1992.

DYCKHOFF, H. A typology of cutting and packing problems. **European Journal of Operational Research**, v. 44, p. 145-159. 1990.

EILON, S.; CHRISFOFIDES, N., The loading problem, **Management Science**, v. 17, p. 259-268. 1971.

ETSCHMAIER, M. M.; MATHAISEL, D. F. X. Airline scheduling: an overview. **Transportation Science**, n. 19, v. 2, p. 127–138. 1985.

FALKENAUER, E. A Hybrid Grouping Genetic Algorithm for Bin Packing. **Journal of Heuristics**, v. 2, n. 1, p. 5–30. 1996.

FISCHETTI, M.; LODI, A.; MARTELLO, S.; TOTH, P. A polyhedral approach to simplified crew scheduling and vehicle scheduling problems. **Management Science**, n. 47, v. 6, p. 833–850. 2001.

FLESZAR, K.; Hindi, K. S. New Heuristics for one-dimensional bin-packing, **Computers and Operations Research**, v. 29, p. 821-839. 2002.

FORES, S.; PROLL, L.; WREN A. An Improved ILP System For Driver Scheduling. **Computer-Aided Transit Scheduling**, Wilson, N. H. M. (ed.), Springer, Berlin, p. 43-61. 1999.

FREITAS, S. L. O.; CARDOSO, G.; FERNANDES Jr., J. L. Análise das características geométricas de corredores de ônibus: estudo na cidade de Porto Alegre, RS. **Trabalhos Técnicos em Eventos da ANTP**. 2001. Disponível na internet via www. URL: http://portal.antp.org.br/Eventos/Congressos/13_ANTP/Trabalhos/186/html/168.html. Acessado em 01/03/2010.

FRELING, R.; WAGELMANS, P.; PAIXÃO, M. An overview of models and techniques for integrating vehicle and crew scheduling. **Computer-Aided Transit Scheduling**, Lecture Notes in Economics and Mathematical Systems, Wilson, N. H. M. (ed.), Springer-Verlag, Berlim, v. 471, p. 441–460. 1999.

FRELING, R.; PAIXÃO, J. M. P.; WAGELMANS, A. P. M. Models and Algorithms for Vehicle Scheduling. **Transportation Science,** v. 35, p. 165-180. 2001.

FRELING, R.; HUISMAN, D.; WAGELMANS, A. Models and algorithms for integration of vehicle and crew scheduling. **Journal of Scheduling**, v. 6, n. 1, p. 63–85. 2003.

FRIBERG, C.; HAASE, K. An exact branch and cut algorithm for the vehicle and crew scheduling problem. **Computer-Aided Transit Scheduling**, Wilson, N. H. M. (ed.), Springer-Verlag, Berlim, p. 63-80. 1999.

FRIESEN, D. K.; LANGSTON, M. A. Variable-sized bin packing. **SIAM J. Comput**. v. 15, p. 222–230. 1986.

GAFFI, A.; NONATO, M. An integrated approach to ex-urban crew and vehicle scheduling. **Computer-aided Transit Scheduling**, Lecture Notes in Economics and Mathematical Systems, Wilson, N. H. M. (ed.), Springer, v. 471, p. 103–128. 1999.

GALVÃO, L. C.; NOVAES, A. G.; DE CURSI, J. E. S.; SOUZA, J. C. A Multiplicatively-weighted Voronoi Diagram Approach to Logistics Districting. **Computers & Operations Research**, v.33, p. 93-114. 2006.

GAREY, M. R.; JOHNSON, D. S. **Computers and intractability: a guide to the theory of NP-completeness**. W. H. Freeman, San Francisco. 1979.

GLOVER, F. Future paths for integer programming and links to artificial intelligence. **Computers and Operations Research**, 13, 533–549. 1986.

GLOVER, F.; KOCHENBERG, G. A. **Handbook of Metaheuristics**. International Series in Operations Research & Management Series, Kluwer's International Series, Stanford University, 556p. 2003.

GOMES, W. P.; GUALDA, N. D. F. Modelagem integrada do problema de programação de tripulantes de aeronaves. **Transportes** (Rio de Janeiro), v. 19, n. 1, p. 23-32. 2011.

GOPALAKRISHNAN, B.; JOHNSON, E. L. Airline crew scheduling: state-of-the-art. **Annals of**

Operations Research, n. 140, v. 1, p. 305-337. 2005.

GUPTA, J. N. D.; HO, J. C. A new heuristic algorithm for the one-dimensional bin-packing problem. **Production Planning & Control**, v. 10, p. 598–603. 1999.

HAASE, K.; FRIBERG, C. An exact algorithm for the vehicle and crew scheduling problem. **Computer-aided Transit Scheduling**, Lecture Notes in Economics and Mathematical Systems, Wilson, N. H. M. (ed.), Springer, v. 471, p. 63–80. 1999.

HAASE, K.; DESAULNIERS, G.; DESROSRIERS, J. Simultaneous vehicle and crew scheduling in urban mass transit systems. **Transportation Science**, v. 35, n. 3, p. 286–303. 2001.

HANSEN, P.; MLADENOVIC, N; PEREZ-BRITO, D. Variable neighborhood decomposition search. **Journal of Heuristics**, v. 7, n. 4, p. 335–350. 2001.

HAOUARI, M.; SERIARI, M. Heuristics, for the variable sized bin-packing problem, **Computers and Operations Research**, v. 36, p. 2877-2884. 2009.

HOARE, C. A. R. Quicksort. **Computer Journal**, vol. 5, n. 1, p. 10–15. 1962.

HOTO, R.; ARENALES, M.; MACULAN, N. The one dimensional compartimentalized knapsack problem: a case study. **European Journal of Operational Research** v. 183, p. 1183-1195. 2007.

HUISMAN, D.; FRELING, R.; WAGELMANS, A. P. M. A dynamic approach to vehicle scheduling. **Econometric Institute Report**, Econometric Institute, Erasmus University Rotterdam, n. 225, v. 2. 2001.

HUISMAN, D.; FRELING, R.; WAGELMANS, A. P. M. Multiple-depot integrated vehicle and crew scheduling. **Econometric Institute Report***,* Econometric Institute, Erasmus University Rotterdam, n. 2001-17. 2003.

HUISMAN, D.; WAGELMANS, A. P. M. A solution approach for dynamic vehicle and crew scheduling. **Econometric Institute Report**, v. 2. 2004.

HUISMAN, D.; WAGELMANS, A. P. M. A solution approach for dynamic vehicle and crew scheduling. **European Journal of Operational Research**, v. 172, ed. 2, p. 453-471. 2006.

IBGE Disponível na internet via www. URL: http://www.ibge.gov.br/paisesat/main.php. Acessado em 16/11/2013. 2010.

JOHNSON, D. S.; MCGEOCH, L. A. The travelling salesman problem: A case study in local optimization. In: E.H.L. Aarts and J.K. Lenstra (eds.), **Local Search in Combinatorial Optimization**. John Wiley & Sons, Chichester, England. p. 215–310. 1997.

KANG, J.; PARK, S. Algorithms for the Variable Sized Bin-packing Problem, **European Journal of Operation Research**, v. 147, p. 365-372. 2003.

KATAYAMA, K.; NARIHISA, H. Iterated local search approach using genetic transformation to the traveling salesman problem. In: Proceedings of GECCO'99, **Morgan Kaufmann.** p. 321–328. 1999.

KLABJAN, D.; JOHNSON, E. L.; NEMHAUSER, G. L.; GELMAN, E.; RAMASWAMY, S. Airline crew scheduling with time windows and plane count constraints. **Transportation Science,** n. 3, **v**. 36,

p. 337–348. 2002.

KWAN, R. S. K.; RAHIN, M. A. Object oriented bus vehicle scheduling—the boost system. In: Proceedings of the 7th International Workshop on Computer-Aided Scheduling of Public Transport. **Annals of the 7th International Workshop on Computer-Aided Scheduling of Public Transport**. p. 36-52. 1997.

LORENA, L. A. N; PEREIRA, M. A.; SALOMÃO, S. N. A. A relaxação lagrangeana/surrogate e o método de geração de colunas: novos limitantes e novas colunas. **Pesquisa Operacional**, v. 23, n. 1, p. 29-47. 2003.

MARINHO, E. H.; OCHI, L. S.; DRUMMOND, L. M. A.; SOUZA, M. J. F.; SILVA, G. P. Busca tabu aplicada ao problema de programação de tripulações de ônibus urbano. In: SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, XXXVI, 2004, São João Del Rei. **Anais do XXXVI SBPO**. São João Del Rei, SOBRAPO, 2004. p. 1471-1482. 2004.

MARQUES, F. P.; ARENALES, M. N. The constrained compartimentalized knapsack problem. **Computers and Operations Research**, v. 34, p. 2109-2129. 2007.

MARTELLO, S.; TOTH, P. **Knapsack Problems: Algorithms and Computer Implementations**, John Wiley & Sons, West Sussex, 1990.

MESQUITA, M.; PAIAS, A. Set partitioning/covering-based approaches for the integrated vehicle and crew scheduling problem. **Computers & Operations Research**, v. 35, p. 1562–1575. 2008.

MLADENOVIC, N.; HANSEN, P. Variable neighborhood Search. **Computers and Operations Research**, v. 24, n. 11, p. 1097–1100. 1997.

MIURA, M. **Modelagem heurística no problema de distribuição de cargas fracionadas de cimento**. Dissertação (Mestrado em Engenharia de Transporte). ed. São Paulo: Universidade de São Paulo, 2008.

MOURA, A. V.; YUNES, T. H.; SOUZA, C. C. A hybrid approach for solving large scale crew scheduling problems with constraint programming and integer programming. **Second Workshop on Practical Aspects of Declarative Languages**, Lecture Notes in Computer Science. Heidelebrg: Springer. 2000.

NOVAES, A. G.; SOUZA DE CURSI, J. E.; GRACIOLLI, O. D. A Continuous Approach to the Design of Physical Distribution Systems, **Computers & Operations Research**, v. 27, n. 9, p. 877-893. 2000.

NOVAES, A. G. N. . Resolução de Problemas de Transportes com Diagramas de Voronoi. In: XXI ANPET, 2007, Rio de Janeiro, RJ. **Panorama Nacional da Pesquisa em Transportes 2007**. Rio de Janeiro, RJ: Asociação Nacional de Pesquisa e Ensino em Transportes, 2007.

NOVAES, A. G.; SOUZA DE CURSI, J. E.; DA SILVA, A. L.; SOUZA, J. C. Solving Continuous Location-districting Problems with Voronoi Diagrams, **Computers & Operations Research**, v. 36, p. 40-59. 2009.

NTU. Disponível na internet via www. URL: http://www.ntu.org.br/novosite/arquivos/anuario2006_2007.pdf. Acessado em 13/05/2008.

PEDROSA, D.; CONSTANTINO, M. Days-off scheduling in public transport companies. **Computer-aided Scheduling of Public Transport**, Lecture Notes in Economics and Mathematical Systems, Springer, v. 505, p. 215–232. 2001.

REIS, J. A. **Técnicas de otimização aplicadas à programação diária de veículos e de tripulações de ônibus**. Monografia (curso de Engenharia de Produção). ed. Ouro Preto: Universidade Federal de Ouro Preto, 2006. 70 p.

REIS, J. A.; SILVA, G. P.; SOUZA, M. J. F. Otimização integrada no sistema de transporte público. In: CONGRESSO DE PESQUISA E ENSINO EM TRANSPORTES, XX, 2006, Brasília. **Anais do XX ANPET**. Brasília, ANPET. v. II, p. 717-728. 2006.

REIS, J. A.; CUNHA, C. B.; NAKAMOTO, F. Y.; RIBEIRO, F. R.; SCHARDONG, A. Análise Comparativa de Algoritmos Eficientes para o Problema de Caminho Mínimo. In: XXI ANPET - Congresso de Pesquisa e Ensino em Transportes, 2007, Rio de Janeiro. **Anais do XXI ANPET - Congresso de Pesquisa e Ensino em Transportes**. Rio de Janeiro: ANPET. 2007.

REIS, J. A. **Heurísticas baseadas em busca em vizinhança variável para o problema de programação integrada de veículos e tripulações no transporte coletivo urbano por ônibus**. Dissertação (Mestrado em Engenharia de Transporte). ed. São Paulo: Universidade de São Paulo, 2008. 97 p.

RIGHINI, G.; SALANI, M. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. **Discrete Optimization**, n. 3, p. 255-273. 2006.

RODRIGUES, M. M.; SOUZA, C. C.; MOURA, A. V. Vehicle and crew scheduling for urban bus lines. **European Journal of Operational Research**, n. 170, p. 844–862. 2006.

ROUSSEAU, J. M.; BLAIS, J. Y. HASTUS: an interactive system for buses and crew scheduling. **Computer scheduling of public transport**, J. M. Rousseau (ed.), North-Holland, v. 2, p. 45-60. 1985

SALOMÃO, S. N. A. **Métodos de Geração de Colunas para Problemas de Atribuição**. Tese (Doutorado em Computação Aplicada). ed. São José dos Campos: Instituto Nacional de Pesquisas Espaciais, 2005. 146 p.

SANTOS, A. G. **Método de Geração de Colunas e Meta-heurísticas para Alocação de Tripulação**. 2001. 80f. Tese (Pós-graduação em Ciência da Computação) – Ciências da Computação – Universidade Federal de Minas Gerais, Belo Horizonte. 2008.

SCHOLL, A.; KLEIN, R.; JÜRGENS, C. BISON: a fast hybrid procedure for exactly solving the one-dimensional bin-packing problem. **Computers & Operations Research**, v. 24, n. 7, p. 627-645. 1997.

SILVA, G. P.; GUALDA, N. D. F. Um Algoritmo de Geração de Arcos para o Problema de Programação de Veículos. **Transportes**, Rio de Janeiro, v. 8, n. 1, p. 35-55. 2000.

SILVA, G. P.; GUALDA, N. D. F. O Método Arcgen para Programação de Veículos: Um Estudo de Caso da Cidade de Belo Horizonte. In: Congresso de Pesquisa e Ensino em Transportes, 15., Campinas, 2001. **Anais: Panorama Nacional da Pesquisa em Transportes 2001** – v.1, Campinas, ANPET, p. 129-137. 2001.

SILVA, G. P. **Uma metodologia baseada na técnica da geração de arcos para o problema de programação de veículos**, Tese (doutorado em Engenharia de Transportes), 1. ed. Escola Politécnica, Universidade de São Paulo, 2001. 146 p.

SILVA, G. P.; SOUZA, M. J. F.; REIS, J. A. Um método híbrido de geração de colunas para otimizar a mão de obra do sistema de transporte público. SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, XXXVI, 2004, São João Del Rei. **Anais do XXXVI SBPO**. São João Del Rei, SOBRAPO, p. 1989-1995. 2004.

SILVA, G. P.; SOUZA, M. J. F.; REIS, J. A. Metaheurísticas Aplicadas ao Sistema de Transporte Público. In: CONGRESO LATINO-IBEROAMERICANO DE INVESTIGACIÓN OPERATIVA, XIII, 2006, Montevidéu, Uruguai. **Anais do XIII CLAIO**. Montevideo, GECA Multimedios, v. 1, p. 147-154. 2006a.

SILVA, G. P.; SOUZA, M. J. F.; REIS, J. A. Resolução Integrada do Problema de Programação de Veículos e Tripulações no Sistema de Transporte Público. In: CONGRESSO LUSO BRASILEIRO PARA O PLANEJAMENTO URBANO, REGIONAL, INTEGRADO E SUSTENTÁVEL, II, 2006, Braga, Portugal. **Anais do II PLURIS**. Braga, v. 1, p. 121-132. 2006b.

SILVA, G. P.; GUALDA, N. D. F. O método ArcGenX para programação de ônibus urbano e interação com a tabela de horários. **Transportes** (Rio de Janeiro), v. 17, p. 53-61. 2009.

SMITH, B. M.; WREN, A. A Bus Crew Scheduling System Using a Set Covering Formulation, **Transportation Research**, v. 22A, p. 97-108. 1988.

SOARES, G. F.; SILVA, G. P.; MARINHO, E. H.; SIMÕES, E. M. L.; SOUZA, M. J. F. Otimização no Sistema de Transporte Público. In: SIMPÓSIO DE PESQUISA OPERACIONAL E LOGÍSTICA DA MARINHA, 2006, Rio de Janeiro. **Anais do SPOLM 2006**. Rio de Janeiro, 2006, Infoweb Sistemas e Serviços. v. 1. p. 206-219. 2006.

SOUZA, M. J. F.; CARDOSO, L. X. T.; SILVA, G. P.; RODRIGUES, M. M. S.; MAPA, S. M. S. Metaheurísticas aplicadas ao problema de programação de tripulações no sistema de transporte público, **Tendências em Matemática Aplicada e Computacional**, v. 5, n. 12, p. 357-368. 2004.

SPTRANS **Expresso Tiradentes:** transformar e urbanizar o meio ambiente. São Paulo, 2006, p. 23-54. 2006.

TURAJLIC, N.; DRAGOVIC, I. A hybrid metaheuristic based on variable neighborhood search and Tabu Search for the web service selection problem. **Electronic Notes in Discrete Mathematics**, v. 39, p. 145-152. 2012.

VALOUXIS, C.; HOUSOS, E. Combined bus and driver scheduling. **Computers and Operations Research**, Elsevier, v. 29, n. 3, p. 243–259. 2002.

VANCE, P. H.; BARNHART, C.; JOHNSON, E. L.; NEMHAUSER, G. L. Airline crew scheduling: a new formulation and decomposition algorithm. **Operations Research,** n. 45, v. 2, p. 188–200. 1995.

VILLA, G.; LOZANO, S.; RACERO, J.; CANCA, D. A Hybrid VNS/Tabu Search algorithm for apportioning the european parliament. Evolutionary Computation in Combinatorial Optimization. **Lecture Notes in Computer Science**, v. 3906, p. 284-292. 2006.

WREN, A.; KWAN, R. S. K.; PARKER, M. E. Scheduling of rail driver duties. **Computers in**

Railways IV, Murty, T. K. S.; Mellitt, B.; Brebbia, C. A.; Sciutto, G. & Sone, S. (eds.), Southampton, Boston, v. 2. 1994.

WREN, A. Heuristics ancient and modern: Transport scheduling through the ages, **Journal of Heuristics**, n. 4, p. 87-100. 1998.

WREN, A.; GUALDA, N. D. F. Integrated Scheduling of Buses and Drivers. **Computer-aided Scheduling of Public Transport,** Lecture Notes in Economics and Mathematical Systems, Wilson, N.H.M. (ed.), Cambridge MA. Springer-Verlag, v. 471, p.155-176. 1999.

ZHANG, G. A New Version of On-line Variable-sized Bin-packing, **Discrete Applied Mathematics**, v. 72, p. 193-197. 1997.

APÊNDICE A

A.1. Resultados Obtidos para a Abordagem Sequencial PPV e PPT

Tabela A.1 – Primeiro conjunto de dados utilizando abordagem sequencial tradicional.

Nº de viagens	Métodos	EEC	GC	VNS	VNS-LR	VNDS
10	**Função Avaliação**	5.447	5.447	5.447	5.447	5.447
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	2	2	2	2	2
	Nº tripulantes	3	3	3	3	3
	Viagem Morta	87	87	87	87	87
	Horas Extras	32	32	32	32	32
	Tempo de Terminal	64	64	64	64	64
	Tempo Ocioso	145	145	145	145	145
20	**Função Avaliação**	15.220	15.220	15.220	15.220	15.220
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	5	5	5	5	5
	Nº tripulantes	9	9	9	9	9
	Viagem Morta	169	169	169	169	169
	Horas Extras	153	153	153	153	153
	Tempo de Terminal	149	149	149	149	149
	Tempo Ocioso	427	427	427	427	427
30	**Função Avaliação**	20.719	20.719	20.719	20.719	20.719
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	7	7	7	7	7
	Nº tripulantes	12	12	12	12	12
	Viagem Morta	199	199	199	199	199
	Horas Extras	137	137	137	137	137
	Tempo de Terminal	599	599	599	599	599
	Tempo Ocioso	448	448	448	448	448
40	**Função Avaliação**	28.027	28.027	28.027	28.027	28.027
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	9	9	9	9	9
	Nº tripulantes	17	17	17	17	17
	Viagem Morta	217	217	217	217	217
	Horas Extras	112	112	112	112	112
	Tempo de Terminal	551	551	551	551	551
	Tempo Ocioso	818	818	818	818	818

Tabela A.2 – Primeiro conjunto de dados utilizando abordagem sequencial inversa

Nº de viagens	Métodos	EEC	GC	VNS	VNS-LR	VNDS
10	**Função Avaliação**	5.435	5.435	5.435	5.435	5.435
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	2	2	2	2	2
	Nº tripulantes	3	3	3	3	3
	Viagem Morta	112	112	112	112	112
	Horas Extras	15	15	15	15	15
	Tempo de Terminal	85	85	85	85	85
	Tempo Ocioso	96	96	96	96	96
20	**Função Avaliação**	15.217	15.217	15.217	15.217	15.217
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	5	5	5	5	5
	Nº tripulantes	9	9	9	9	9
	Viagem Morta	257	257	257	257	257
	Horas Extras	110	110	110	110	110
	Tempo de Terminal	168	168	168	168	168
	Tempo Ocioso	315	315	315	315	315
30	**Função Avaliação**	20.975	20.975	20.975	20.975	20.975
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	7	7	7	7	7
	Nº tripulantes	12	12	12	12	12
	Viagem Morta	348	348	348	348	348
	Horas Extras	91	91	91	91	91
	Tempo de Terminal	705	705	705	705	705
	Tempo Ocioso	392	392	392	392	392
40	**Função Avaliação**	28.315	28.315	28.315	28.315	28.315
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	9	9	9	9	9
	Nº tripulantes	17	17	17	17	17
	Viagem Morta	315	315	315	315	315
	Horas Extras	54	54	54	54	54
	Tempo de Terminal	902	902	902	902	902
	Tempo Ocioso	675	675	675	675	675

Tabela A.3 – Instâncias de teste entre 53 e 98 viagens diárias conjunto de dados utilizando abordagem sequencial tradicional.

Nº de viagens	Métodos	EEC	GC	VNS	VNS-LR	VNDS
53	**Função Avaliação**	17.039	17.039	17.039	17.039	17.039
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	5	5	5	5	5
	Nº tripulantes	10	10	10	10	10
	Viagem Morta	230	230	230	230	230
	Horas Extras	151	151	151	151	151
	Tempo de Terminal	606	606	606	606	606
	Tempo Ocioso	671	671	671	671	671
69	**Função Avaliação**	24.174	24.174	24.174	24.174	24.174
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	7	7	7	7	7
	Nº tripulantes	14	14	14	14	14
	Viagem Morta	322	322	322	322	322
	Horas Extras	508	508	508	508	508
	Tempo de Terminal	546	546	546	546	546
	Tempo Ocioso	968	968	968	968	968
90	**Função Avaliação**	ND	33.175	33.175	33.175	33.175
	% acima MS	ND	MS	MS	MS	MS
	Nº veículos	ND	10	10	10	10
	Nº tripulantes	ND	20	20	20	20
	Viagem Morta	ND	374	374	374	374
	Horas Extras	ND	319	319	319	319
	Tempo de Terminal	ND	854	854	854	854
	Tempo Ocioso	ND	935	935	935	935
98	**Função Avaliação**	ND	32.526	32.526	32.526	32.526
	% acima MS	ND	MS	MS	MS	MS
	Nº veículos	ND	11	11	11	11
	Nº tripulantes	ND	19	19	19	19
	Viagem Morta	ND	506	506	506	506
	Horas Extras	ND	167	167	167	167
	Tempo de Terminal	ND	573	573	573	573
	Tempo Ocioso	ND	607	607	607	607

Tabela A.4 – Instâncias de teste entre 53 e 98 viagens diárias conjunto de dados utilizando abordagem sequencial inversa.

Nº de viagens	Métodos	EEC	GC	VNS	VNS-LR	VNDS
53	**Função Avaliação**	17.158	17.158	17.158	17.158	17.158
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	5	5	5	5	5
	Nº tripulantes	10	10	10	10	10
	Viagem Morta	317	317	317	317	317
	Horas Extras	119	119	119	119	119
	Tempo de Terminal	745	745	745	745	745
	Tempo Ocioso	541	541	541	541	541
69	**Função Avaliação**	24.349	24.349	24.349	24.349	24.349
	% acima MS	MS	MS	MS	MS	MS
	Nº veículos	7	7	7	7	7
	Nº tripulantes	14	14	14	14	14
	Viagem Morta	459	459	459	459	459
	Horas Extras	452	452	452	452	452
	Tempo de Terminal	703	703	703	703	703
	Tempo Ocioso	824	824	824	824	824
90	**Função Avaliação**	ND	33.116	33.116	33.116	33.116
	% acima MS	ND	MS	MS	MS	MS
	Nº veículos	ND	10	10	10	10
	Nº tripulantes	ND	20	20	20	20
	Viagem Morta	ND	451	451	451	451
	Horas Extras	ND	197	197	197	197
	Tempo de Terminal	ND	1012	1012	1012	1012
	Tempo Ocioso	ND	808	808	808	808
98	**Função Avaliação**	ND	32.759	32.759	32.759	32.759
	% acima MS	ND	MS	MS	MS	MS
	Nº veículos	ND	11	11	11	11
	Nº tripulantes	ND	19	19	19	19
	Viagem Morta	ND	684	684	684	684
	Horas Extras	ND	110	110	110	110
	Tempo de Terminal	ND	619	619	619	619
	Tempo Ocioso	ND	552	552	552	552

Tabela A.5 – Instâncias de teste entre 108 e 172 viagens diárias conjunto de dados utilizando abordagem sequencial tradicional.

Nº de viagens	Métodos	GC	VNS	VNS-LR	VNDS
108	**Função Avaliação**	19.315	19.410	19.315	19.315
	% acima MS	MS	0,49%	MS	MS
	Nº veículos	5	5	5	5
	Nº tripulantes	11	12	11	11
	Viagem Morta	500	500	500	500
	Horas Extras	412	117	412	412
	Tempo de Terminal	524	524	524	524
	Tempo Ocioso	967	652	967	967
121	**Função Avaliação**	19.797	19.797	19.797	19.797
	% acima MS	MS	MS	MS	MS
	Nº veículos	5	5	5	5
	Nº tripulantes	12	12	12	12
	Viagem Morta	500	500	500	500
	Horas Extras	371	371	371	371
	Tempo de Terminal	577	577	577	577
	Tempo Ocioso	478	478	478	478
161	**Função Avaliação**	30.468	30.855	30.783	30.468
	% acima MS	MS	1,27%	1,03%	MS
	Nº veículos	8	8	8	8
	Nº tripulantes	18	19	19	18
	Viagem Morta	136	136	136	136
	Horas Extras	543	296	296	543
	Tempo de Terminal	1804	1804	1804	1804
	Tempo Ocioso	1306	1187	1115	1306
172	**Função Avaliação**	66.438	66.878	66.878	66.438
	% acima MS	MS	0,66%	0,66%	MS
	Nº veículos	25	25	25	25
	Nº tripulantes	35	34	34	35
	Viagem Morta	900	900	900	900
	Horas Extras	1237	1683	1683	1237
	Tempo de Terminal	1000	1000	1000	1000
	Tempo Ocioso	1164	1712	1712	1164

Tabela A.6 – Instâncias de teste entre 108 e 172 viagens diárias conjunto de dados utilizando abordagem sequencial inversa.

Nº de viagens	Métodos	GC	VNS	VNS-LR	VNDS
108	**Função Avaliação**	19.296	19.296	19.296	19.296
	% acima MS	MS	MS	MS	MS
	Nº veículos	5	5	5	5
	Nº tripulantes	11	11	11	11
	Viagem Morta	482	482	482	482
	Horas Extras	415	415	415	415
	Tempo de Terminal	613	613	613	613
	Tempo Ocioso	889	889	889	889
121	**Função Avaliação**	20.009	20.009	20.009	20.009
	% acima MS	MS	MS	MS	MS
	Nº veículos	5	5	5	5
	Nº tripulantes	12	12	12	12
	Viagem Morta	500	500	500	500
	Horas Extras	405	405	405	405
	Tempo de Terminal	693	693	693	693
	Tempo Ocioso	506	506	506	506
161	**Função Avaliação**	30.342	30.518	30.518	30.342
	% acima MS	MS	0,58%	0,58%	MS
	Nº veículos	9	9	9	9
	Nº tripulantes	17	18	18	17
	Viagem Morta	118	142	142	118
	Horas Extras	690	214	214	690
	Tempo de Terminal	1752	1781	1781	1752
	Tempo Ocioso	974	1025	1025	974
172	**Função Avaliação**	64.652	65.707	65.707	64.652
	% acima MS	MS	1,63%	1,63%	MS
	Nº veículos	25	25	25	25
	Nº tripulantes	33	34	34	33
	Viagem Morta	812	868	868	812
	Horas Extras	1354	810	810	1354
	Tempo de Terminal	1263	1371	1371	1263
	Tempo Ocioso	1057	1980	1980	1057

Tabela A.7 – Instâncias de teste entre 207 e 287 viagens diárias conjunto de dados utilizando abordagem sequencial tradicional.

Nº de viagens	Métodos	GC	VNS	VNS-LR	VNDS
207	**Função Avaliação**	38.289	39.011	39.011	38.289
	% acima MS	MS	1,89%	1,89%	MS
	Nº veículos	14	15	15	14
	Nº tripulantes	18	19	19	18
	Viagem Morta	1400	1480	1480	1400
	Horas Extras	1373	850	850	1373
	Tempo de Terminal	389	238	238	389
	Tempo Ocioso	354	113	113	354
253	**Função Avaliação**	59.235	59.251	59.251	59.251
	% acima MS	MS	0,03%	0,03%	0,03%
	Nº veículos	18	18	18	18
	Nº tripulantes	35	34	34	34
	Viagem Morta	421	421	421	421
	Horas Extras	965	1715	1715	1715
	Tempo de Terminal	1829	1829	1829	1829
	Tempo Ocioso	1634	1150	1150	1150
269	**Função Avaliação**	113.376	113.564	113.564	113.480
	% acima MS	MS	0,17%	0,17%	0,09%
	Nº veículos	44	43	43	44
	Nº tripulantes	58	59	59	59
	Viagem Morta	1598	1680	1680	1598
	Horas Extras	1890	1416	1416	1640
	Tempo de Terminal	909	812	812	909
	Tempo Ocioso	3491	4560	4560	3095
287	**Função Avaliação**	61.552	61.723	61.723	61.552
	% acima MS	MS	0,28%	0,28%	MS
	Nº veículos	18	18	18	18
	Nº tripulantes	38	37	37	38
	Viagem Morta	374	415	415	374
	Horas Extras	1330	1876	1876	1330
	Tempo de Terminal	854	931	931	854
	Tempo Ocioso	1290	1210	1210	1290

Tabela A.8 – Instâncias de teste entre 207 e 287 viagens diárias conjunto de dados utilizando abordagem sequencial inversa.

Nº de viagens	Métodos	GC	VNS	VNS-LR	VNDS
207	**Função Avaliação**	39.509	39.918	39.918	39.509
	% acima MS	MS	1,04%	1,04%	MS
	Nº veículos	15	15	15	15
	Nº tripulantes	17	17	17	17
	Viagem Morta	1614	1702	1702	1614
	Horas Extras	1805	1956	1956	1805
	Tempo de Terminal	420	398	398	420
	Tempo Ocioso	251	204	204	251
253	**Função Avaliação**	59.023	59.981	59.981	59.981
	% acima MS	MS	1,62%	1,62%	1,62%
	Nº veículos	18	18	18	18
	Nº tripulantes	34	34	34	34
	Viagem Morta	421	513	513	513
	Horas Extras	1040	1498	1498	1498
	Tempo de Terminal	2145	2089	2089	2089
	Tempo Ocioso	1956	1870	1870	1870
269	**Função Avaliação**	113.006	113.535	113.535	113.205
	% acima MS	MS	0,47%	0,47%	0,18%
	Nº veículos	45	45	45	45
	Nº tripulantes	57	57	57	57
	Viagem Morta	1430	1732	1732	1676
	Horas Extras	1916	2390	2390	2120
	Tempo de Terminal	1264	812	812	974
	Tempo Ocioso	3050	2479	2479	2639
287	**Função Avaliação**	61.667	62.528	62.528	61.667
	% acima MS	MS	1,40%	1,40%	MS
	Nº veículos	19	19	19	19
	Nº tripulantes	36	37	37	36
	Viagem Morta	418	478	478	418
	Horas Extras	2131	1754	1754	2131
	Tempo de Terminal	602	542	542	602
	Tempo Ocioso	967	1522	1522	967

Tabela A.9 – Instâncias de teste entre 299 e 372 viagens diárias conjunto de dados utilizando abordagem sequencial tradicional.

Nº de viagens	Métodos	VNS	VNS-LR	VNDS
299	**Função Avaliação**	60.206	59.742	59.608
	% acima MS	1,00%	0,22%	MS
	Nº veículos	16	15	15
	Nº tripulantes	38	37	38
	Viagem Morta	595	455	455
	Horas Extras	480	925	480
	Tempo de Terminal	2187	2828	2828
	Tempo Ocioso	1869	2154	1910
359	**Função Avaliação**	73.045	72.716	71.904
	% acima MS	1,59%	1,13%	MS
	Nº veículos	20	20	19
	Nº tripulantes	46	44	45
	Viagem Morta	512	490	605
	Horas Extras	528	1408	990
	Tempo de Terminal	2615	2810	2810
	Tempo Ocioso	2350	2110	1904
361	**Função Avaliação**	94.773	94.353	93.213
	% acima MS	1,67%	1,22%	MS
	Nº veículos	32	32	31
	Nº tripulantes	53	54	54
	Viagem Morta	839	839	748
	Horas Extras	1696	1320	1290
	Tempo de Terminal	1749	1837	1913
	Tempo Ocioso	2954	2198	2224
372	**Função Avaliação**	87.332	87.016	85.382
	% acima MS	2,28%	1,91%	MS
	Nº veículos	24	26	25
	Nº tripulantes	53	52	52
	Viagem Morta	1296	750	503
	Horas Extras	640	961	937
	Tempo de Terminal	3120	2754	2548
	Tempo Ocioso	3340	2840	2954

Tabela A.10 – Instâncias de teste entre 378 e 507 viagens diárias conjunto de dados utilizando abordagem sequencial tradicional.

Nº de viagens	Métodos	VNS	VNS-LR	VNDS
378	**Função Avaliação**	97.495	96.360	95.238
	% acima MS	2,37%	1,18%	MS
	Nº veículos	27	26	27
	Nº tripulantes	56	57	57
	Viagem Morta	1622	947	1083
	Horas Extras	2368	2298	1380
	Tempo de Terminal	3247	3100	2784
	Tempo Ocioso	3268	3770	3528
468	**Função Avaliação**	104.541	104.541	103.908
	% acima MS	0,61%	0,61%	MS
	Nº veículos	31	31	31
	Nº tripulantes	62	62	61
	Viagem Morta	829	829	829
	Horas Extras	2514	2514	2806
	Tempo de Terminal	2953	2953	2953
	Tempo Ocioso	1902	1902	1685
503	**Função Avaliação**	120.952	120.952	119.012
	% acima MS	1,63%	1,63%	MS
	Nº veículos	37	37	36
	Nº tripulantes	73	73	75
	Viagem Morta	1334	1334	1460
	Horas Extras	2361	2361	1118
	Tempo de Terminal	1302	1302	994
	Tempo Ocioso	2260	2260	1862
507	**Função Avaliação**	115.284	115.622	115.147
	% acima MS	0,12%	0,41%	MS
	Nº veículos	39	38	39
	Nº tripulantes	64	65	64
	Viagem Morta	1049	1240	1049
	Horas Extras	3684	3120	3606
	Tempo de Terminal	970	928	970
	Tempo Ocioso	1848	2974	1867

Tabela A.11 – Instâncias de teste entre 538 e 1.038 viagens diárias conjunto de dados utilizando abordagem sequencial tradicional.

Nº de viagens	Métodos	VNS	VNS-LR	VNDS
538	**Função Avaliação**	92.426	93.416	91.396
	% acima MS	1,13%	2,21%	MS
	Nº veículos	24	24	24
	Nº tripulantes	55	58	56
	Viagem Morta	675	675	675
	Horas Extras	3835	2549	2787
	Tempo de Terminal	956	956	956
	Tempo Ocioso	3450	4012	3516
706	**Função Avaliação**	192.967	192.160	185.937
	% acima MS	3,78%	3,35%	MS
	Nº veículos	64	65	62
	Nº tripulantes	114	114	116
	Viagem Morta	1647	1356	1482
	Horas Extras	4104	3590	926
	Tempo de Terminal	2265	2310	1683
	Tempo Ocioso	1200	958	1438
769	**Função Avaliação**	157.024	150.810	149.518
	% acima MS	5,02%	0,86%	MS
	Nº veículos	46	45	46
	Nº tripulantes	98	90	88
	Viagem Morta	1312	1485	1198
	Horas Extras	1816	2393	2892
	Tempo de Terminal	2824	3172	2234
	Tempo Ocioso	3944	4882	5104
1038	**Função Avaliação**	220.901	220.181	217.433
	% acima MS	1,59%	1,26%	MS
	Nº veículos	72	74	73
	Nº tripulantes	131	122	128
	Viagem Morta	4194	4081	3410
	Horas Extras	2228	6112	2945
	Tempo de Terminal	1164	2004	1001
	Tempo Ocioso	3893	1791	2722

APÊNDICE B

B.1. COMPARAÇÃO ENTRE OS PACOTES DE OTIMIZAÇÃO LINEAR PARA O VSBPP

Os resultados apresentados na Tabela B.1 mostram a comparação entre o consumo de memória RAM do pacote de otimização GUROBI 4 e do pacote de otimização IBM ILOG CPLEX 12. Estes resultados são relativos aos problemas de pequeno porte com o número de entregas a serem alocadas aos veículos variando entre 10 e 40. A solução ótima foi obtida resolvendo o modelo matemático apresentado na seção 5.1. A Tabela B.1 também apresenta a comparação entre os tempos computacionais utilizados pelos dois pacotes de otimização.

Os resultados apresentados na Tabela B.1 mostram que em 14 das 17 instâncias testadas o consumo de memória RAM foi igual para ambos os pacotes de otimização linear e o tempo computacional do CPLEX foi, em média, 2,83% (entre 0% e 5,88%) acima do tempo computacional do GUROBI. Nas demais instâncias (3 instâncias) o consumo de memória RAM do CPLEX foi, em média, 5,13% (entre 4,54% e 5,56%) acima do consumo de memória RAM do GUROBI.

Os resultados apresentados na Tabela B.2 mostram a comparação entre o consumo de memória RAM do pacote de otimização linear GUROBI 4 e do pacote de otimização linear IBM ILOG CPLEX 12. Estes resultados são relativos aos problemas de médio porte com o número de entregas a serem alocadas aos veículos variando entre 30 e 500. A solução ótima foi obtida resolvendo o modelo matemático apresentado na seção 5.1. A Tabela B.2 também apresenta a comparação entre as soluções obtidas.

Os resultados apresentados na Tabela B.2 mostram que em 2 das 23 instâncias testadas o consumo de memória RAM foi igual para ambos os pacotes de otimização linear, nas demais instâncias (21 instâncias) o consumo de memória RAM do CPLEX foi, em média, 5,02% (entre 3,45% e 8,33%) acima do consumo de memória RAM do GUROBI. Estas instâncias foram consideradas como de médio porte devido a nenhum dos pacotes de otimização linear conseguir obter uma solução em menos de uma hora de processamento.

A solução apresentada na coluna *Custo Sol* da Tabela B.2 apresenta a solução obtida após uma hora de processamento, a coluna *GAP* apresenta a porcentagem que esta

solução está acima do limite inferior calculado pelo GUROBI ou pelo CPLEX.

Tabela B.1 – Comparação entre o GUROBI 4 e CPLEX 12 (pequeno porte).

Número de objetos	GUROBI 4		CPLEX 12	
	Consumo de memória RAM (MB)	Tempo CPU (seg)	Consumo de memória RAM (MB)	Tempo CPU (seg)
10	5	1,7	5	1,8
12	5	2,1	5	2,2
12	6	3,9	6	4,1
15	5	0,0	5	0,0
20	6	0,8	6	0,8
25	7	7,6	7	7,9
40	8	7,6	8	7,6
40	8	0,8	8	0,8
40	11	40,7	11	42,3
40	9	2,5	9	2,5
40	10	8,5	10	8,6
40	9	5,1	9	5,3
40	19	1784	20	1854,1
40	15	1238	15	1287,9
40	18	1417	19	1471,8
40	22	2393	23	2477,6
40	9	65,3	9	67,9
Média	10,1	410,5	10,3	426,1

Tabela B.2 – Comparação entre o GUROBI 4 e CPLEX 12 (médio porte).

Nº. de Objetos	GUROBI 4			CPLEX 12		
	Custo Sol	GAP (%)	Consumo RAM (MB)	Custo Sol	GAP (%)	Consumo RAM (MB)
30	1.240	1,32	8	1.240	1,38	8
35	1.400	2,24	7	1.400	2,32	7
40	1.500	3,26	12	1.500	3,44	13
45	2.000	1,28	25	2.000	1,34	26
50	2.040	1,60	28	2.040	1,65	29
55	2.040	1,82	29	2.040	1,85	30
60	2.400	1,58	36	2.400	1,60	38
70	2.860	2,33	42	2.860	2,35	44
120	5.000	1,26	59	5.000	1,26	62
120	5.040	1,91	56	5.050	2,02	59
120	4.720	1,36	45	4.720	1,42	47
120	4.760	1,29	54	4.760	1,36	56
120	4.760	1,29	53	4.760	1,36	56
250	9.680	1,53	197	9.680	1,55	209
250	9.880	1,38	245	890	1,42	260
250	9.820	1,17	185	9.820	1,18	196
250	9.680	0,86	161	9.680	0,87	168
250	9.880	1,25	223	9.890	1,32	233
500	19.780	4,43	3.994	19.820	4,63	4.201
500	20.060	3,62	3.686	20.070	3,66	3.831
500	20.260	3,72	3.789	20.280	3,83	4.005
500	20.400	2,69	3.277	20.420	2,81	3.417
500	20.540	3,88	3.379	20.570	4,04	3.559
Média		2,05	851,7		2,12	893,7

APÊNDICE C

C.1. EXPERIMENTOS COMPUTACIONAIS PARA O VSBPP

Tabela C.1 – Resultados obtidos para as instâncias de pequeno porte – GUROBI e VNS.

Número de Objetos	***Lower Bound***	**Solução Ótima**					**Solução VNS**				
		Custo Sol	**Tempo**	**Veic Grande**	**Veic Médio**	**Veic Pequeno**	**Custo Sol**	**Tempo**	**Veic Grande**	**Veic Médio**	**Veic Pequeno**
10	371	400	1,7	1	2	1	400	0,22	1	2	1
12	443	460	2,1	3	1	0	460	0,31	3	1	0
12	344	360	3,9	3	0	0	360	0,39	3	0	0
15	589	600	0,0	5	0	0	600	0,33	5	0	0
20	681	700	0,8	5	1	0	700	0,34	5	1	0
25	1.037	1.060	7,6	8	1	0	1.060	0,53	8	1	0
40	1.634	1.660	7,6	13	1	0	1.660	1,02	13	1	0
40	1.542	1.560	0,8	13	0	0	1.580	1,09	11	1	2
40	1.635	1.660	40,7	13	1	0	1.660	1,03	13	1	0
40	1.542	1.560	2,5	13	0	0	1.580	0,88	9	5	0
40	1.634	1.660	8,5	13	1	0	1.680	0,92	14	0	0
40	1.592	1.620	5,1	12	1	1	1.640	0,99	12	2	0
40	1.530	1.540	1.784,0	12	1	0	1.560	0,93	13	0	0
40	1.496	1.520	1.238,0	11	2	0	1.520	0,92	11	2	0
40	1.633	1.660	1.417,0	13	1	0	1.660	0,93	13	1	0
40	1.551	1.560	2.393,0	13	0	0	1.600	0,94	11	2	1
40	1.503	1.520	65,3	11	2	0	1.540	0,78	12	1	0
Total	**20.757**	**21.100**	**6.979**	**162**	**15**	**2**	**21.260**	**12,55**	**157**	**21**	**4**

Tabela C.2 – Resultados obtidos para as instâncias de pequeno porte – VNS-LR e VNDS.

Número de Objetos	*Lower Bound*	Solução VNS-LR					Solução VNDS				
		Custo Sol	Tempo	Veic Grande	Veic Médio	Veic Pequeno	Custo Sol	Tempo	Veic Grande	Veic Médio	Veic Pequeno
10	371	400	0,21	1	2	1	400	0,21	1	2	1
12	443	460	0,30	3	1	0	460	0,29	3	1	0
12	344	360	0,38	3	0	0	360	0,37	3	0	0
15	589	600	0,32	5	0	0	600	0,31	5	0	0
20	681	700	0,33	5	1	0	700	0,32	5	1	0
25	1.037	1.060	0,51	8	1	0	1.060	0,50	8	1	0
40	1.634	1.660	0,99	13	1	0	1.660	0,97	13	1	0
40	1.542	1.560	1,06	13	0	0	1.560	1,04	13	0	0
40	1.635	1.660	1,00	13	1	0	1.660	0,98	13	1	0
40	1.542	1.560	0,85	13	0	0	1.560	0,83	13	0	0
40	1.634	1.660	0,89	13	1	0	1.660	0,87	13	1	0
40	1.592	1.620	0,96	12	1	1	1.620	0,94	12	1	1
40	1.530	1.560	0,90	13	0	0	1.540	0,88	12	1	0
40	1.496	1.520	0,89	11	2	0	1.520	0,87	11	2	0
40	1.633	1.660	0,90	13	1	0	1.660	0,88	13	1	0
40	1.551	1.600	0,92	11	2	1	1.560	0,89	13	0	0
40	1.503	1.540	0,76	12	1	0	1.520	0,74	11	2	0
Total	**20.757**	**21.180**	**12,18**	**162**	**15**	**3**	**21.100**	**11,92**	**162**	**15**	**2**

Tabela C.3 – Resultados obtidos para as instâncias de médio porte – GUROBI e VNS.

Nº. de Objetos	*Lower Bound*	Solução GUROBI					Solução VNS				
		Custo Sol	GAP (%)	Veic Grande	Veic Médio	Veic Pequeno	Custo Sol	Tempo	Veic Grande	Veic Médio	Veic Pequeno
30	1.215	1.240	1,32	7	4	0	1.260	0,70	8	3	0
35	1.362	1.400	2,24	11	0	1	1.400	0,87	10	2	0
40	1.444	1.500	3,26	11	1	1	1.480	1,03	10	2	1
45	1.966	2.000	1,28	15	2	0	2.000	1,36	15	2	0
50	2.002	2.040	1,60	17	0	0	2.060	1,28	14	3	1
55	1.998	2.040	1,82	17	0	0	2.060	1,33	13	5	0
60	2.356	2.400	1,58	20	0	0	2.420	1,55	16	5	0
70	2.783	2.860	2,33	23	1	0	2.840	1,91	22	2	0
120	4.921	5.000	1,26	41	0	1	5.000	5,14	41	0	1
120	4.930	5.040	1,91	42	0	0	5.040	6,88	37	6	0
120	4.644	4.720	1,36	38	0	2	4.720	4,22	36	4	0
120	4.687	4.760	1,29	39	0	1	4.760	5,10	38	2	0
120	4.687	4.760	1,29	39	0	1	4.760	4,25	38	2	0
250	9.507	9.680	1,53	80	0	1	9.620	14,47	78	1	2
250	9.737	9.880	1,38	81	0	2	9.860	15,37	79	3	1
250	9.659	9.820	1,17	81	1	0	9.780	14,43	79	3	0
250	9.572	9.680	0,86	80	0	1	9.700	15,33	77	3	2
250	9.731	9.880	1,25	81	0	2	9.860	15,30	79	3	1
500	18.968	19.780	4,43	159	7	0	19.160	61,96	154	6	1
500	19.282	20.060	3,62	164	3	1	19.460	81,48	158	5	0
500	19.339	20.260	3,72	164	5	1	19.540	67,02	159	3	2
500	16.567	20.400	2,69	165	6	0	19.720	72,13	161	4	0
500	19.691	20.540	3,88	167	5	0	19.900	80,60	160	7	0
Total	**181.048**	**189.740**	**2,05**	**1.542**	**35**	**15**	**186.400**	**473,71**	**1.482**	**76**	**12**

Tabela C.4 – Resultados obtidos para as instâncias de médio porte – VNS-LR e VNDS.

Nº. de Objetos	*Lower Bound*	Solução VNS-LR					Solução VNDS				
		Custo Sol	Tempo	Veic Grande	Veic Médio	Veic Pequeno	Custo Sol	Tempo	Veic Grande	Veic Médio	Veic Pequeno
30	1.215	1.240	0,68	7	4	0	1.240	0,66	7	4	0
35	1.362	1.400	0,85	11	0	1	1.400	0,82	11	0	1
40	1.444	1.480	1,00	9	4	0	1.480	0,98	10	2	1
45	1.966	2.000	1,32	15	2	0	2.000	1,29	15	2	0
50	2.002	2.040	1,25	17	0	0	2.040	1,22	17	0	0
55	1.998	2.040	1,29	17	0	0	2.040	1,26	17	0	0
60	2.356	2.400	1,50	20	0	0	2.400	1,47	20	0	0
70	2.783	2.840	1,86	22	2	0	2.840	1,81	22	2	0
120	4.921	5.000	4,97	41	0	1	5.000	4,87	41	0	1
120	4.930	5.040	6,70	37	6	0	5.040	6,55	37	6	0
120	4.644	4.720	4,10	36	4	0	4.720	3,99	36	4	0
120	4.687	4.760	4,94	38	2	0	4.760	4,86	38	2	0
120	4.687	4.760	4,10	38	2	0	4.760	4,02	38	2	0
250	9.507	9.620	14,09	78	1	2	9.620	13,72	78	1	2
250	9.737	9.860	14,95	79	3	1	9.860	14,55	79	3	1
250	9.659	9.780	14,05	79	3	0	9.780	13,69	79	3	0
250	9.572	9.680	14,90	80	0	1	9.680	14,55	80	0	1
250	9.731	9.860	14,85	79	3	1	9.860	14,56	79	3	1
500	18.968	19.160	60,15	154	6	1	19.160	59,13	154	6	1
500	19.282	19.460	79,06	158	5	0	19.460	77,76	158	5	0
500	19.339	19.540	65,29	159	3	2	19.540	63,66	159	3	2
500	16.567	19.720	69,81	161	4	0	19.720	68,63	161	4	0
500	19.691	19.900	77,99	160	7	0	19.900	76,41	160	7	0
Total	**181.048**	**186.300**	**459,7**	**1.495**	**61**	**10**	**186.300**	**450,5**	**1.496**	**59**	**11**

Tabela C.5 – Resultados obtidos para as instâncias de grande porte – VNS e VNS-LR.

Número de Objetos	*Lower Bound*	Solução VNS					Solução VNS-LR				
		FA	Tempo	Grande	Médio	Pequeno	FA	Tempo	Grande	Médio	Pequeno
1.000	38.249	38.560	315	315	6	2	38.540	307	314	7	2
1.000	38.905	39.260	278	319	9	1	39.200	270	320	8	0
1.000	39.380	39.720	271	322	10	1	39.700	264	322	9	2
1.000	39.444	39.760	271	325	6	2	39.760	264	325	6	2
1.000	38.087	38.420	538	312	9	1	38.420	519	312	9	1
1.000	38.256	38.600	342	312	10	2	38.540	333	313	9	1
1.000	37.844	38.140	346	309	9	2	38.140	334	309	9	2
1.000	38.704	39.000	271	322	2	2	38.960	264	320	4	2
1.000	38.250	38.540	347	315	5	3	38.500	336	314	5	4
1.000	38.105	38.400	295	313	6	3	38.400	287	313	6	3
1.000	38.337	38.640	297	314	8	2	38.620	289	316	7	0
1.000	38.450	38.740	259	319	3	2	38.740	250	319	3	2
1.000	37.655	38.000	458	307	10	2	38.000	444	307	10	2
1.000	37.947	38.260	491	310	9	2	38.220	477	309	9	3
1.000	37.814	38.160	261	310	8	2	38.160	253	310	8	2
1.000	38.574	38.900	492	313	11	3	38.860	478	314	11	1
1.000	38.691	39.000	259	319	4	4	38.820	251	318	5	2
1.000	38.765	39.140	274	321	3	4	39.100	266	320	3	5
1.000	38.227	38.520	427	311	8	5	38.480	412	313	6	4
1.000	38.336	38.660	268	314	5	6	38.660	260	314	5	6
Total	**768.020**	**774.420**	**6.760**	**6.302**	**141**	**51**	**773.820**	**6.558**	**6.302**	**139**	**46**

Tabela C.6 – Resultados obtidos para as instâncias de grande porte – VNDS.

Número de Objetos	***Lower Bound***	**Solução VNDS**				
		FA	**Tempo**	**Grande**	**Médio**	**Pequeno**
1.000	38.249	38.540	299	314	7	2
1.000	38.905	39.200	265	321	6	1
1.000	39.380	39.700	258	323	7	3
1.000	39.444	39.760	258	325	6	2
1.000	38.087	38.420	509	312	9	1
1.000	38.256	38.540	323	313	9	1
1.000	37.844	38.140	329	309	9	2
1.000	38.704	38.960	258	320	4	2
1.000	38.250	38.500	331	314	5	4
1.000	38.105	38.400	279	313	6	3
1.000	38.337	38.620	284	316	7	0
1.000	38.450	38.740	245	319	3	2
1.000	37.655	38.000	434	308	8	3
1.000	37.947	38.220	464	309	9	3
1.000	37.814	38.160	248	310	8	2
1.000	38.574	38.860	468	314	11	1
1.000	38.691	38.820	246	318	5	2
1.000	38.765	39.100	261	320	3	5
1.000	38.227	38.480	404	313	6	4
1.000	38.336	38.660	254	314	5	6
Total	**768.020**	**773.820**	**6.416**	**6.305**	**133**	**49**

APÊNDICE D

D.1. EXPERIMENTOS COMPUTACIONAIS PARA O BID-VSBPP

Tabela D.1 – Resultados obtidos para as instâncias com 120 objetos – GUROBI e VNS.

Lower Bound	Solução GUROBI					Solução VNS				
	FA	GAP (%)	Grande	Médio	Pequeno	FA	Tempo	Grande	Médio	Pequeno
4.529	4.740	4,43	38	1	1	4.720	17,52	28	12	2
4.611	4.760	2,04	39	0	1	4.820	14,34	33	7	2
4.348	4.520	2,62	37	0	1	4.480	8,22	30	8	1
4.662	4.740	1,27	38	1	1	4.840	6,24	33	8	1
4.706	4.900	3,62	40	1	0	4.960	8,47	30	12	2
4.558	4.640	1,17	38	0	1	4.720	8,16	32	4	6
4.567	4.680	2,33	39	0	0	4.720	6,98	32	8	1
4.668	4.860	2,82	39	1	1	4.880	8,32	32	8	3
4.785	4.920	2,17	41	0	0	4.980	17,77	33	7	4
4.396	4.600	4,07	37	0	2	4.700	7,58	31	9	1
4.915	5.140	1,69	42	1	0	5.220	16,60	33	11	2
4.638	4.880	2,88	39	2	0	4.900	8,02	32	9	2
4.595	4.860	3,18	38	3	0	4.820	8,46	32	9	1
4.609	4.900	2,13	40	1	0	5.000	10,59	33	8	3
4.718	4.900	1,97	40	1	0	4.980	7,61	34	9	0
4.545	4.760	2,45	39	0	1	4.760	9,59	34	6	1
4.920	5.340	1,94	43	1	1	5.340	20,50	33	9	6
4.929	5.280	1,50	44	0	0	5.260	8,75	36	7	3
4.643	4.800	2,78	40	0	0	4.920	16,56	32	10	1
4.686	4.780	1,53	39	1	0	4.900	8,48	34	5	4
93.028	**97.000**	**2,43**	**790**	**14**	**10**	**97.920**	**218,76**	**647**	**166**	**46**

Tabela D.2 – Resultados obtidos para as instâncias com 120 objetos – VNS-LR e VNDS.

Lower Bound	Solução VNS-LR					Solução VNDS				
	FA	Tempo	Grande	Médio	Pequeno	FA	Tempo	Grande	Médio	Pequeno
4.529	4.720	16,90	28	12	2	4.720	16,57	28	12	2
4.611	4.760	13,94	39	0	1	4.760	13,63	39	0	1
4.348	4.480	7,94	30	8	1	4.480	7,81	30	8	1
4.662	4.740	6,02	38	1	1	4.740	5,87	38	1	1
4.706	4.900	8,17	40	1	0	4.900	7,97	40	1	0
4.558	4.640	7,95	38	0	1	4.640	7,78	38	0	1
4.567	4.680	6,74	39	0	0	4.680	6,59	39	0	0
4.668	4.860	8,03	39	1	1	4.860	7,93	39	1	1
4.785	4.920	17,09	41	0	0	4.920	17,04	41	0	0
4.396	4.600	7,35	37	0	2	4.600	7,19	37	0	2
4.915	5.140	16,23	42	1	0	5.140	15,90	42	1	0
4.638	4.900	7,83	32	9	2	4.900	7,65	32	9	2
4.595	4.820	8,17	32	9	1	4.820	8,07	32	9	1
4.609	5.000	10,18	33	8	3	5.000	9,99	33	8	3
4.718	4.900	7,42	40	1	0	4.900	7,27	40	1	0
4.545	4.760	9,28	39	0	1	4.760	9,05	39	0	1
4.920	5.340	19,68	43	1	1	5.340	19,29	43	1	1
4.929	5.260	8,55	36	7	3	5.260	8,33	36	7	3
4.643	4.920	16,21	32	10	1	4.920	15,74	32	10	1
4.686	4.780	8,26	39	1	0	4.780	7,99	39	1	0
93.028	**97.120**	**211,95**	**737**	**70**	**21**	**97.120**	**207,66**	**737**	**70**	**21**

Tabela D.3 – Resultados obtidos para as instâncias com 250 objetos – GUROBI e VNS.

Lower Bound	Solução GUROBI					Solução VNS				
	FA	GAP (%)	Grande	Médio	Pequeno	FA	Tempo	Grande	Médio	Pequeno
9.461	9.920	4,37	82	0	1	9.960	42,60	66	10	13
9.506	9.800	2,73	81	0	1	10.000	26,50	69	14	4
9.736	10.320	4,20	86	0	0	10.400	42,23	70	16	5
9.544	9.960	3,91	83	0	0	9.980	39,90	66	15	7
9.658	10.080	3,92	84	0	0	10.140	57,40	68	15	6
9.679	10.280	4,54	85	0	1	10.240	52,22	67	18	5
9.698	10.060	3,34	83	1	0	10.140	42,16	67	17	5
9.867	10.500	5,75	85	3	0	10.460	63,30	70	15	7
10.071	10.660	5,27	88	1	0	10.620	52,46	72	15	6
9.619	9.980	2,98	81	1	2	10.140	23,54	69	13	7
10.019	10.520	4,99	83	4	2	10.580	47,18	69	19	5
9.668	10.320	3,41	86	0	0	10.440	42,27	71	16	4
10.073	10.680	3,32	89	0	0	10.620	68,82	70	15	9
9.788	10.520	4,54	87	0	1	10.320	44,64	71	14	5
9.520	9.900	5,37	81	1	1	10.040	51,26	69	12	7
10.062	10.800	4,75	86	4	1	10.640	44,02	74	16	2
9.265	9.660	4,57	78	3	0	9.700	43,13	71	7	6
9.520	9.920	3,03	82	0	1	9.980	62,03	68	15	4
9.571	10.160	4,38	84	0	1	10.280	64,74	75	8	6
9.730	10.420	5,18	86	1	0	10.300	47,25	72	11	7
194.055	**204.460**	**4,23**	**1.680**	**19**	**12**	**204.980**	**957,65**	**1.394**	**281**	**120**

Tabela D.4 – Resultados obtidos para as instâncias com 250 objetos – VNS-LR e VNDS.

Lower Bound	Solução VNS-LR					Solução VNDS				
	FA	Tempo	Grande	Médio	Pequeno	FA	Tempo	Grande	Médio	Pequeno
9.461	9.920	41,33	82	0	1	9.920	40,32	82	0	1
9.506	9.800	25,95	81	0	1	9.800	25,08	81	0	1
9.736	10.400	40,76	70	16	5	10.320	39,78	86	0	0
9.544	9.980	38,70	66	15	7	9.980	38,15	66	15	7
9.658	10.080	55,90	84	0	0	10.080	54,21	84	0	0
9.679	10.240	50,26	67	18	5	10.240	50,08	67	18	5
9.698	10.140	41,00	67	17	5	10.140	39,99	67	17	5
9.867	10.460	61,22	70	15	7	10.460	60,17	70	15	7
10.071	10.620	50,84	72	15	6	10.620	49,74	72	15	6
9.619	9.980	23,05	81	1	2	9.980	22,51	81	1	2
10.019	10.520	45,73	83	4	2	10.520	44,71	83	4	2
9.668	10.320	40,80	86	0	0	10.320	40,31	86	0	0
10.073	10.620	66,31	70	15	9	10.620	65,56	70	15	9
9.788	10.320	43,22	71	14	5	10.320	42,56	71	14	5
9.520	10.040	49,92	69	12	7	10.040	48,42	69	12	7
10.062	10.640	42,88	74	16	2	10.640	41,54	74	16	2
9.265	9.700	42,11	71	7	6	9.700	40,67	71	7	6
9.520	9.920	60,18	82	0	1	9.920	59,48	82	0	1
9.571	10.160	62,26	84	0	1	10.160	61,74	84	0	1
9.730	10.420	45,83	86	1	0	10.300	44,72	72	11	7
194.055	**204.280**	**928,21**	**1.516**	**166**	**72**	**204.080**	**909,75**	**1.518**	**160**	**74**

Tabela D.5 – Resultados obtidos para as instâncias com 500 objetos – GUROBI e VNS.

Lower Bound	Solução GUROBI					Solução VNS				
	FA	GAP (%)	Grande	Médio	Pequeno	FA	Tempo	Grande	Médio	Pequeno
18.967	20.640	6,29	172	0	0	19.840	516,07	143	22	6
19.281	21.820	11,29	169	9	8	20.400	390,17	147	22	7
19.338	22.080	8,06	176	8	2	20.900	338,10	153	15	13
19.566	21.680	10,46	177	2	3	20.620	299,83	150	23	4
19.690	22.520	12,58	172	10	11	20.700	279,73	151	17	11
19.688	21.720	5,82	181	0	0	21.360	197,99	161	14	8
19.861	22.540	11,42	176	7	9	21.080	460,15	153	20	9
19.582	21.780	8,40	176	1	7	20.940	639,94	150	19	13
18.785	21.360	8,36	172	4	4	20.280	332,35	146	22	7
19.301	21.600	5,40	169	6	9	20.240	406,67	144	24	7
19.110	21.880	5,51	166	10	12	20.200	518,19	149	20	4
19.145	21.860	8,98	168	9	10	20.320	240,10	147	22	6
19.067	21.500	8,79	171	5	6	20.120	403,21	143	24	7
18.776	20.460	10,34	169	1	1	19.460	194,05	136	25	8
19.490	22.300	8,41	172	7	12	20.760	274,35	152	18	9
19.212	21.060	5,87	174	1	1	20.000	263,98	144	20	9
19.296	21.140	10,26	173	3	1	20.300	920,82	139	29	9
18.953	20.400	6,46	170	0	0	19.800	391,00	140	26	5
19.324	21.980	8,65	174	7	5	20.500	338,97	148	21	8
18.780	21.460	9,56	162	9	14	19.840	407,75	138	24	11
385.212	**431.780**	**8,55**	**3.439**	**99**	**115**	**407.660**	**7.813,42**	**2.934**	**427**	**161**

Tabela D.6 – Resultados obtidos para as instâncias com 500 objetos – VNS-LR e VNDS.

Lower Bound	Solução VNS-LR					Solução VNDS				
	FA	Tempo	Grande	Médio	Pequeno	FA	Tempo	Grande	Médio	Pequeno
18.967	19.840	502,24	143	22	6	19.840	491,56	143	22	6
19.281	20.400	375,07	147	22	7	20.400	371,44	147	22	7
19.338	20.900	327,08	153	15	13	20.900	320,72	153	15	13
19.566	20.620	290,15	150	23	4	20.620	287,18	150	23	4
19.690	20.700	270,72	151	17	11	20.700	263,45	151	17	11
19.688	21.360	191,14	161	14	8	21.360	186,59	161	14	8
19.861	21.080	442,16	153	20	9	21.080	432,63	153	20	9
19.582	20.940	619,46	150	19	13	20.940	610,69	150	19	13
18.785	20.280	322,35	146	22	7	20.280	315,60	146	22	7
19.301	20.240	391,58	144	24	7	20.240	386,05	144	24	7
19.110	20.200	506,53	149	20	4	20.200	493,94	149	20	4
19.145	20.320	231,07	147	22	6	20.320	230,21	147	22	6
19.067	20.120	393,09	143	24	7	20.120	384,90	143	24	7
18.776	19.460	187,47	136	25	8	19.460	185,65	136	25	8
19.490	20.760	263,40	152	18	9	20.760	259,78	152	18	9
19.212	20.000	257,83	144	20	9	20.000	250,36	144	20	9
19.296	20.300	891,91	139	29	9	20.300	875,42	139	29	9
18.953	19.800	375,48	140	26	5	19.800	368,79	140	26	5
19.324	20.500	329,11	148	21	8	20.500	324,94	148	21	8
18.780	19.840	398,86	138	24	11	19.840	388,34	138	24	11
385.212	**407.660**	**7.566,69**	**2.934**	**427**	**161**	**407.660**	**7.428,24**	**2.934**	**427**	**161**

Tabela D.7 – Resultados obtidos para as instâncias com 1.000 objetos – VNS e VNS-LR.

Lower Bound	Solução VNS					Solução VNS-LR				
	FA	Tempo (seg)	Grande	Médio	Pequeno	FA	Tempo (seg)	Grande	Médio	Pequeno
38.249	40.600	3.081,60	298	34	18	40.600	2.990,38	298	34	18
38.904	41.100	2.355,87	306	35	11	41.100	2.274,83	306	35	11
39.379	41.880	1.219,76	313	36	9	41.880	1.179,75	313	36	9
39.443	42.060	2.296,82	310	39	12	42.060	2.245,60	310	39	12
38.087	40.080	1.301,93	291	42	12	40.080	1.272,51	291	42	12
38.255	40.600	1.606,11	298	34	18	40.600	1.559,85	298	34	18
37.843	40.100	2.711,80	289	43	14	40.100	2.626,38	289	43	14
38.703	41.800	1.486,76	312	34	12	41.800	1.455,84	312	34	12
38.249	40.680	2.879,11	295	40	16	40.680	2.813,75	295	40	16
38.105	40.920	1.087,78	305	32	14	40.920	1.055,69	305	32	14
38.336	40.580	3.000,51	301	35	12	40.580	2.933,90	301	35	12
38.449	40.820	1.412,08	292	49	11	40.820	1.371,13	292	49	11
37.655	39.720	1.277,21	286	46	10	39.720	1.239,28	286	46	10
37.946	40.700	955,74	300	39	10	40.700	923,63	300	39	10
37.813	40.420	1.540,18	306	33	5	40.420	1.480,42	306	33	5
38.573	40.960	1.191,37	319	22	6	40.960	1.158,61	319	22	6
38.690	41.320	2.945,28	294	46	18	41.320	2.831,30	294	46	18
38.764	41.300	1.017,08	312	29	12	41.300	978,13	312	29	12
38.226	40.600	2.873,54	303	32	13	40.600	2.784,46	303	32	13
38.336	41.220	1.402,14	319	23	8	41.220	1.348,58	319	23	8
768.005	**817.460**	**37.642,67**	**6.049**	**723**	**241**	**817.460**	**36.524,01**	**6.049**	**723**	**241**

Tabela D.8 – Resultados obtidos para as instâncias com 1.000 objetos – VNDS.

Lower Bound	**Solução VNDS**				
	FA	**Tempo (seg)**	**Grande**	**Médio**	**Pequeno**
38.249	40.600	2.943,24	298	34	18
38.904	41.100	2.241,37	306	35	11
39.379	41.880	1.157,06	313	36	9
39.443	42.060	2.190,94	310	39	12
38.087	40.080	1.225,64	291	42	12
38.255	40.600	1.516,33	298	34	18
37.843	40.100	2.582,99	289	43	14
38.703	41.800	1.425,51	312	34	12
38.249	40.680	2.757,04	295	40	16
38.105	40.920	1.032,52	305	32	14
38.336	40.580	2.857,99	301	35	12
38.449	40.820	1.351,22	292	49	11
37.655	39.720	1.204,79	286	46	10
37.946	40.700	912,73	300	39	10
37.813	40.420	1.464,56	306	33	5
38.573	40.960	1.143,24	319	22	6
38.690	41.320	2.805,67	294	46	18
38.764	41.300	975,28	312	29	12
38.226	40.600	2.708,89	303	32	13
38.336	41.220	1.326,00	319	23	8
768.005	**817.460**	**35.823,00**	**6.049**	**723**	**241**

APÊNDICE E

E.1. EXPERIMENTOS COMPUTACIONAIS PARA O BID-VSBPP COM OS *BINS* MODIFICADOS

Tabela E.1 – Solução Ótima para as instâncias com 120 objetos.

Solução GUROBI, VNS, VNS-LR, VNDS			
FA	**Veíc Grande**	**Veíc Médio**	**Veíc Pequeno**
2.720	7	1	0
2.840	7	1	1
2.840	7	1	1
3.080	8	1	0
2.840	7	1	1
2.880	8	0	0
2.880	8	0	0
2.720	7	1	0
2.880	8	0	0
2.720	7	1	0
2.880	8	0	0
2.720	7	1	0
2.840	7	1	1
2.640	7	0	1
2.880	8	0	0
2.720	7	1	0
2.720	7	1	0
2.720	7	1	0
2.720	7	1	0
2.880	8	0	0
56.120	**147**	**13**	**5**

Tabela E.2 – Solução Ótima para as instâncias com 250 objetos.

Solução GUROBI, VNS, VNS-LR, VNDS			
FA	**Veíc Grande**	**Veíc Médio**	**Veíc Pequeno**
5.720	15	1	1
5.760	16	0	0
5.760	16	0	0
5.720	15	1	1
5.520	15	0	1
5.600	15	1	0
5.760	16	0	0
5.880	16	0	1
5.520	15	0	1
5.520	15	0	1
5.760	16	0	0
5.600	15	1	0
5.720	15	1	1
5.720	15	1	1
5.760	16	0	0
5.760	16	0	0
5.880	16	0	1
5.600	15	1	0
5.720	15	1	1
5.520	15	0	1
113.800	**308**	**8**	**11**

Tabela E.3 – Solução Ótima para as instâncias com 500 objetos.

Solução GUROBI, VNS, VNS-LR, VNDS			
FA	**Veíc Grande**	**Veíc Médio**	**Veíc Pequeno**
11.280	31	0	1
11.360	31	1	0
10.760	29	1	1
11.280	31	0	1
11.360	31	1	0
11.160	31	0	0
11.480	31	1	1
11.520	32	0	0
11.520	32	0	0
11.640	32	0	1
11.160	31	0	0
11.360	31	1	0
11.520	32	0	0
11.280	31	0	1
11.280	31	0	1
11.000	30	1	0
11.360	31	1	0
11.360	31	1	0
10.920	30	0	1
11.480	31	1	1
226.080	**620**	**9**	**9**

Tabela E.4 – Solução Ótima para as instâncias com 1.000 objetos.

Solução GUROBI, VNS, VNS-LR, VNDS			
FA	**Veíc Grande**	**Veíc Médio**	**Veíc Pequeno**
22.520	62	1	0
22.320	62	0	0
22.640	62	1	1
22.320	62	0	0
22.320	62	0	0
22.440	62	0	1
22.680	63	0	0
22.800	63	0	1
22.320	62	0	0
22.320	62	0	0
22.280	61	1	1
22.160	61	1	0
22.320	62	0	0
22.320	62	0	0
ND	ND	ND	ND
22.320	62	0	0
22.320	62	0	0
22.680	63	0	0
22.440	62	0	1
22.520	62	1	0
426.040	**1.179**	**5**	**5**

APÊNDICE F

F.1. COMPARAÇÃO ENTRE OS PACOTES DE OTIMIZAÇÃO LINEAR PARA O BID-VSBPP

Os resultados apresentados na Tabela F.1 mostram a comparação entre o consumo de memória RAM do pacote de otimização linear GUROBI 4 e do pacote de otimização linear IBM ILOG CPLEX 12.

Estes resultados são relativos aos problemas de pequeno porte, ou seja, 120 objetos. A solução ótima foi obtida resolvendo o modelo matemático apresentado na seção 6.1. A Tabela F.1 também apresenta a comparação entre os tempos computacionais utilizados pelos dois pacotes de otimização linear. Os resultados apresentados na Tabela F.1 mostram que em 13 das 20 instâncias testadas o consumo de memória RAM foi igual para ambos os pacotes de otimização linear e o tempo computacional do CPLEX foi, em média, 0,2 segundos (23,5%) acima do tempo computacional do GUROBI. Nas demais instâncias (7 instâncias) o consumo de memória RAM do CPLEX foi, em média, 1 MB (2,1%) acima do consumo de memória RAM do GUROBI.

Os resultados apresentados na Tabela F.2 mostram a comparação entre o consumo de memória RAM do pacote de otimização linear GUROBI 4 e do pacote de otimização linear IBM ILOG CPLEX 12. Estes resultados são relativos aos problemas com instâncias de 250 objetos. A solução ótima foi obtida resolvendo o modelo matemático apresentado na seção 6.1. A Tabela F.2 também apresenta a comparação entre as soluções obtidas.

Os resultados apresentados na Tabela F.2 mostram que em todas as instâncias testadas o consumo de memória RAM do CPLEX foi, em média, 4,17% (entre 2,79% e 5,0%) acima do consumo de memória RAM do GUROBI.

Os resultados apresentados na Tabela F.3 mostram a comparação entre o consumo de memória RAM do pacote de otimização linear GUROBI 4 e do pacote de otimização linear IBM ILOG CPLEX 12. Estes resultados são relativos aos problemas com instâncias de 500 objetos. Os resultados apresentados na Tabela F.3 mostram que em todas as instâncias testadas o consumo de memória RAM do CPLEX foi, em média, 6,69% (entre 5,94% e 7,99%) acima

do consumo de memória RAM do GUROBI.

Tabela F.1 – Comparação entre o GUROBI 4 e CPLEX 12 (120 objetos).

	GUROBI 4		**CPLEX 12**	
Número de objetos	**Consumo de memória RAM (MB)**	**Tempo CPU (seg)**	**Consumo de memória RAM (MB)**	**Tempo CPU (seg)**
120	19	1	19	1
120	17	1	17	1
120	18	1	19	1
120	20	1	21	1
120	16	1	17	1
120	17	0	17	1
120	17	1	17	1
120	19	1	19	1
120	20	0	20	0
120	20	2	20	2
120	16	0	16	0
120	17	1	18	1
120	17	1	18	2
120	17	1	17	1
120	19	1	19	1
120	18	1	19	2
120	19	1	19	1
120	20	1	20	2
120	16	0	16	0
120	16	1	17	1
Média	**17,9**	**0,9**	**18,3**	**1,1**

O tempo computacional em uma das instâncias foi igual para ambos os pacotes de otimização. Em 19 das 20 instâncias, o tempo computacional utilizado pelo CPLEX foi, em média, 7,03% (entre 5,96% e 11,11%) acima do tempo gasto pelo GUROBI.

Os resultados apresentados na Tabela F.4 mostram a comparação entre o consumo de memória RAM do pacote de otimização linear GUROBI 4 e do pacote de otimização linear IBM ILOG CPLEX 12. Estes resultados são relativos aos problemas com instâncias de 1.000

objetos. Para estas instâncias de maior porte o GUROBI conseguiu encontrar a solução ótima em 19 das 20 instâncias testadas, enquanto que o CPLEX não conseguiu encontrar a solução ótima em nenhuma das instâncias testadas.

Tabela F.2 – Comparação entre o GUROBI 4 e CPLEX 12 (250 objetos).

	GUROBI 4		**CPLEX 12**	
Número de objetos	**Consumo de memória RAM (MB)**	**Tempo CPU (seg)**	**Consumo de memória RAM (MB)**	**Tempo CPU (seg)**
250	239	64	247	66
250	249	0	259	0
250	209	5	219	5
250	234	15	244	16
250	245	8	254	8
250	200	31	210	32
250	176	5	184	5
250	191	32	197	34
250	161	10	168	10
250	231	32	240	33
250	237	1	247	1
250	153	19	160	20
250	169	45	176	47
250	203	48	211	49
250	211	4	221	4
250	182	31	191	32
250	229	9	240	9
250	236	8	246	8
250	213	43	222	45
250	179	32	184	33
Média	**207,4**	**22,1**	**216,0**	**22,9**

Tabela F.3 – Comparação entre o GUROBI 4 e CPLEX 12 (500 objetos).

	GUROBI 4		CPLEX 12	
Número de objetos	**Consumo de memória RAM (MB)**	**Tempo CPU (seg)**	**Consumo de memória RAM (MB)**	**Tempo CPU (seg)**
500	815	260	877	278
500	751	144	811	154
500	709	310	755	330
500	748	145	800	156
500	784	293	832	311
500	790	44	844	47
500	721	125	766	135
500	758	61	803	65
500	785	9	834	10
500	808	237	860	254
500	728	299	775	322
500	794	276	851	294
500	739	8	783	8
500	682	117	726	126
500	684	124	731	133
500	715	163	763	174
500	729	319	780	344
500	798	218	846	231
500	700	199	745	213
500	722	175	779	189
Média	**748,0**	**176,3**	**798,1**	**188,7**

Tabela F.4 – Comparação entre o GUROBI 4 e CPLEX 12 (1.000 objetos).

	GUROBI 4		**CPLEX 12**	
Número de objetos	**Consumo de memória RAM (MB)**	**Tempo CPU (seg)**	**Consumo de memória RAM (MB)**	**Tempo CPU (seg)**
1.000	7.142	3.598	ND	ND
1.000	5.538	296	ND	ND
1.000	4.212	3.707	ND	ND
1.000	4.915	66	ND	ND
1.000	7.652	308	ND	ND
1.000	5.618	2.002	ND	ND
1.000	5.149	2.475	ND	ND
1.000	3.965	1.626	ND	ND
1.000	6.947	55	ND	ND
1.000	7.404	2.401	ND	ND
1.000	4.958	2.016	ND	ND
1.000	5.304	2.942	ND	ND
1.000	5.549	66	ND	ND
1.000	6.439	3.094	ND	ND
1.000	ND	32.334	ND	ND
1.000	5.865	61	ND	ND
1.000	7.355	69	ND	ND
1.000	7.158	62	ND	ND
1.000	5.215	4.736	ND	ND
1.000	4.861	2.269	ND	ND
Média	**5.855,1**	**3.209,2**	**ND**	**ND**