

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO DE JOINVILLE
CURSO DE ENGENHARIA DE TRANSPORTES E LOGÍSTICA

NATAN BISSOLI

MÉTODOS DE OTIMIZAÇÃO PARA O PROBLEMA DE CARPOOLING

Joinville
2018

NATAN BISSOLI

MÉTODOS DE OTIMIZAÇÃO PARA O PROBLEMA DE CARPOOLING

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia de Transportes e Logística, no curso Engenharia de Transportes e Logística da Universidade Federal de Santa Catarina, Centro Tecnológico de Joinville.

Orientador: Prof^a. Dra. Silvia Lopes de Sena Taglialenha

Joinville
2018

NATAN BISSOLI

MÉTODOS DE OTIMIZAÇÃO PARA O PROBLEMA DE CARPOOLING

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do título de Bacharel em Engenharia de Transportes e Logística, na Universidade Federal de Santa Catarina, Centro Tecnológico de Joinville.

Joinville, 21 de novembro de 2018.

Dr.

Coordenador do Curso

Banca Examinadora:

Dra. Silvia Lopes de Sena Tagliapietra
Orientadora

Dra. Christiane Wenck Nogueira Fernandes
1º Membro

Dr. Pablo Andretta Jaskowiak
2º Membro

RESUMO

Neste trabalho propõe-se a aplicação de métodos de otimização para solução do problema de carpooling, no qual consiste em pessoas com destinos em comum que se locomovem juntas em veículos compartilhados. Apresenta-se modelos aproximados, heurística gulosa, meta heurística simulated annealing, variable neighborhood descent (VND) e variable neighborhood search (VNS), além de um modelo exato. Estes buscam maximizar a quantidade de usuários no sistema como passageiros. Leva-se em consideração as informações dos usuários, dentre elas quais são os proprietários de veículos, as janelas de tempo disponível para se alcançar o destino final de cada motorista, a capacidade de cada veículo e o tempo de início da viagem. Apresenta-se a formulação dos métodos utilizados bem como a aplicação de tais métodos para um problema com 66 usuários com destino à uma universidade. Os métodos apresentaram quais usuários são motoristas e quais são as caronas, bem como o trajeto que maximiza a quantidade de passageiros. Os métodos aproximados utilizados foram programados na linguagem de programação C/C++. Apresenta-se análises comparativas entre os métodos nos diferentes cenários propostos. Os métodos apresentam soluções satisfatórias para o problema de carpooling.

Palavras-chave: Carpooling. Métodos heurísticos. Métodos de otimização. Mobilidade.

ABSTRACT

This work proposes the application of optimization methods to solve the problem of carpooling, which consist of people with common destinations that move together in shared vehicles. We present approximate models, greedy heuristics, meta-heuristics simulated annealing, variable neighborhood descent (VND) and variable neighborhood search (VNS), in addition to an exact model, which seek to maximize the number of users in the system as passengers. Users' information is taken into account, including vehicle owners, windows of time available to reach the final destination of each driver, the capacity of each vehicle and the time of commencement of the trip. Presents the formulation of the methods used as well as the application of such methods to a problem with 66 users destined to a university. The methods presented which users are drivers and which are the rides, as well as the route that maximizes the number of passengers. The approximate methods used were programmed in the C / C ++ programming language. We present comparative analyzes between the different scenarios proposed. The methods present satisfactory solutions to the problem of carpooling.

Keywords: Carpooling. Heuristics Methods. Optimization Methods. Mobility.

AGRADECIMENTOS

Agradeço, primeiramente, a Deus pelas conquistas alcançadas ao longo desta jornada. Agradeço ao meu pai e minha mãe, que são a base do meu caráter e exemplos para toda minha vida.

Agradeço a minha família, a qual sempre me apoiou em minhas decisões e compreenderam as minhas dificuldades.

Agradeço a todos os professores que fizeram parte da minha caminhada acadêmica. Em especial a minha orientadora e aos professores do curso de transportes e logística, que me apresentaram novos horizontes para a vida.

Agradeço aos amigos que fiz nessa jornada, que sempre estiveram ao meu lado e foram um forte auxílio em momentos difíceis e se mostraram fiéis e companheiros pelo caminho.

"Temos o destino que
merecemos. O nosso destino
está de acordo com os nossos
méritos."

Albert Einstein

LISTA DE ILUSTRAÇÕES

Figura 1 – Exemplo de grafo	15
Figura 2 – Exemplo de lista de adjacência	16
Figura 3 – Efeito da inserção do destino final no grafo do carpooling	17
Figura 4 – Representação do sistema de carpooling	19
Figura 5 – Algoritmo VNS	26
Figura 6 – Funcionamento do Algoritmo VNS	26
Figura 7 – Algoritmo VND	27
Figura 8 – Fluxograma do procedimento SA	29
Figura 9 – Fluxograma da Metodologia Utilizada	31
Figura 10 – Solução para o exemplo proposto	36
Figura 11 – Representação da estrutura de um grafo	37
Figura 12 – Algoritmo guloso proposto para o problema do carpooling	38
Figura 13 – Codificação para as meta heurísticas	40
Figura 14 – Modificações consideradas para a meta heurística VND	41
Figura 15 – Algoritmo VND modificado	42
Figura 16 – Vizinhanças consideradas para a meta heurística VNS	43
Figura 17 – Algoritmo VNS modificado	44
Figura 18 – Fluxograma do procedimento para gerar perturbações (Busca local)	45
Figura 19 – Possíveis trocas com posição escolhida já ocupada	46
Figura 20 – Possíveis trocas com posição escolhida não ocupada	47
Figura 21 – Gráfico para determinação da temperatura inicial	52
Figura 22 – Resultados da aplicação do modelo exato para o cenário 1	53
Figura 23 – Resultado da aplicação do algoritmo VNS para o cenário 2	56
Figura 24 – Tempo de processamento dos métodos para o cenário 1	57
Figura 25 – Tempo de processamento dos métodos para o cenário 2	58
Figura 26 – Tempo de processamento dos métodos para o cenário 3	60
Figura 27 – Evolução do tempo de processamento nos cenários propostos	60
Figura 28 – Síntese dos resultados para o cenário 1	61
Figura 29 – Síntese dos resultados para o cenário 2	62
Figura 30 – Resultados das diferentes soluções iniciais utilizadas	63
Figura 31 – Síntese dos resultados para o cenário 3	63
Figura 32 – Quantidade de motoristas e taxa de ocupação para o cenário 1	65
Figura 33 – Quantidade de motoristas e taxa de ocupação para o cenário 2	65

Figura 34 – Quantidade de motoristas e taxa de ocupação para o cenário 3 com as restrições (4.6) e (4.7)	66
Figura 35 – Quantidade de motoristas e taxa de ocupação para o cenário 3 sem as restrições (4.6) e (4.7)	67
Figura 36 – Distância total percorrida pelos motoristas no cenário 1	68
Figura 37 – Distância total percorrida pelos motoristas no cenário 2	68
Figura 38 – Distância total percorrida pelos motoristas no cenário 3 com e sem as restrições (4.6) e (4.7)	69

LISTA DE TABELAS

Tabela 1 – Vetor do tempo inicial dos passageiros	35
Tabela 2 – Matriz de tempos de viagem do exemplo	36
Tabela 3 – Cenários considerados	48
Tabela 4 – Solução - Carpooling 40 usuários	49
Tabela 5 – Rotas dos motoristas pelo algoritmo guloso para 40 usuários	50
Tabela 6 – Rota dos motoristas pelo algoritmo VND para 40 usuários	50
Tabela 7 – Rotas dos motoristas VNS para 40 usuários	51
Tabela 8 – Resultado da aplicação do algoritmo SA para 40 usuários	52
Tabela 9 – Rotas dos motoristas pelo algoritmo guloso para 66 usuários	54
Tabela 10 – Rota dos motoristas VND para 66 usuários	54
Tabela 11 – Rotas dos motoristas VNS para 66 usuários	55
Tabela 12 – Complexidade do cenário 3 com e sem as restrições de horário de saída - 20 usuários	59

SUMÁRIO

1	INTRODUÇÃO	12
1.1	Objetivo Geral	13
1.2	Objetivos Específicos	13
1.3	Justificativas	13
1.4	Estrutura do trabalho	14
2	REVISÃO TEÓRICA	15
2.1	Grafos	15
2.2	Carpooling	17
2.2.1	Modelo Carpooling - Maximização de Compatibilidade	18
2.2.2	Modelo Carpooling - Minimização de Custo	19
2.2.3	Modelo Carpooling - Maximização do Número de Usuários	21
2.3	Complexidade	23
2.4	Heurísticas	23
2.4.1	Heurística Gulosa	24
2.5	Meta Heurísticas	25
2.5.1	Meta Heurísticas VNS e VND	25
2.5.2	Simulated Annealing	28
3	METODOLOGIA	31
4	MÉTODOS DE SOLUÇÃO	33
4.1	Solução via Modelo Exato	33
4.1.1	Cenário de Validação	35
4.2	Solução via heurística gulosa	36
4.3	Solução via Meta heurística VND	39
4.4	Solução via Meta heurística VNS	42
4.5	Solução via Meta Heurística Simulated Annealing	44
5	ANÁLISE DE RESULTADOS	48
5.1	Rotas dos Motoristas	49
5.1.1	Cenário 1	49
5.1.2	Cenário 2	53
5.2	Tempo de processamento	57
5.2.1	Cenário 1	57
5.2.2	Cenário 2	57
5.2.3	Cenário 3	58

5.3	Quantidade de Passageiros Transportados	60
5.3.1	Cenário 1	61
5.3.2	Cenário 2	62
5.3.2.1	Cenário 2 - Considerando-se diferentes soluções iniciais	62
5.3.3	Cenário 3	63
5.4	Quantidade de Veículos e Taxa de Ocupação	64
5.4.1	Cenário 1	64
5.4.2	Cenário 2	65
5.4.3	Cenário 3	66
5.5	Distâncias Percorridas	67
5.5.1	Cenário 1	67
5.5.2	Cenário 2	68
5.5.3	Cenário 3	69
6	CONCLUSÕES	70
6.1	Trabalhos Futuros	71
	REFERÊNCIAS	72

1 INTRODUÇÃO

Devido à crescente quantidade de veículos nas vias, diversos novos problemas são encontrados ao se tentar atender à demanda, sejam eles de mobilidade, ambientais ou econômicos, nota-se então a necessidade de novos modos de locomoção. A construção de novas vias não se apresenta como alternativa eficiente uma vez que logo depois de construídas, muitas vezes, já estão saturadas. O transporte coletivo, em algumas regiões, também se mostra ineficaz devido a vários fatores, como custo, frequência, acessibilidade, abrangência, entre outros. Sendo assim, o uso de outras formas de locomoção está cada vez mais em evidência, por exemplo, os transportes ativos. Contudo, para os mesmos, ainda é necessária uma infraestrutura que permita segurança aos usuários, mas esta nem sempre é garantida.

Outra forma de locomoção que tem se destacado é a prática de carpooling, também conhecido por *ride-sharing* ou *lift-sharing* e refere-se à partilha de um veículo privado, pertencente a um dos usuários, que se deslocam sob a forma de boléia. Reúnem-se no mesmo automóvel usuários que efetuam ou pretendem efetuar pelo menos uma parte do mesmo percurso e as despesas de deslocamento são partilhadas entre todos (HARTMAN et al., 2014).

Dentre as principais vantagens do carpooling, pode-se citar a diminuição do número de carros nas vias, aumento na frequência de vagas para estacionar, redução no número de acidentes, do consumo de combustível e da emissão de poluentes.

O veículo privado individual possibilita diversas vantagens ao usuário, relacionadas principalmente à flexibilidade e conveniência (ABRAHAMSE; KEALL, 2012). Sendo assim, existe grande tendência para que os veículos individuais sejam adotados pelos cidadãos. Porém, Abrahamse e Keall (2012), afirmam que “[...] o crescimento na demanda pelas viagens de carros contribuem para uma variedade de problemas relacionados à qualidade ambiental e saúde pública [...]” (ABRAHAMSE; KEALL, 2012, p. 45). Desta forma, o carpooling se mostra como uma alternativa.

Bruglieri et al. (2011) explicam que “[...] usualmente o motorista decide buscar passageiros de maneira a conseguir utilizar uma via de veículos de alta capacidade ou dividir os gastos da viagem [...]” (BRUGLIERI et al., 2011, p. 558, tradução nossa). Além das vantagens para a cidade de maneira geral, o carpooling também demonstra vantagens individuais para os usuários. “No carpooling, os indivíduos precisam coordenar, negociar e na maior parte dos casos adaptar suas agendas (escala diária) para permitir a cooperação.” (HUSSAIN et al., 2014, p. 397).

Sob o ponto de vista do utilizador, existem algumas desvantagens, tais como, perda da independência e rigidez do sistema, em particular no que diz respeito às horas das viagens, indução de stress devido ao compromisso de horários ou de presença, e perda de privacidade na partilha do veículo com outras pessoas.

Embora o carpooling apresente vantagens e desvantagens, é necessário esforço por parte dos indivíduos para que o sistema seja eficiente. Em (HARTMAN et al., 2014) o problema de carpooling (PC) é apresentado através de um problema de programação matemática para determinar quais indivíduos que possuem veículo serão motoristas e quais usuários serão alocados como passageiros no sistema, de maneira que alguns usuários que possuem veículos podem ser considerados como passageiros. Trata-se de tema recente e em desenvolvimento.

O objetivo deste trabalho é apresentar métodos de solução baseados em uma modelagem matemática para determinar a solução ótima do problema de carpooling e propor meta heurísticas para a resolução em tempo viável do mesmo. A modelagem se dá por uma função objetivo, na qual busca-se maximizar a quantidade de usuários do sistema carpooling e, por consequência, minimizar a quantidade de veículos nas vias. Utilizando-se de *A Mathematical Programming Language* (AMPL) e o Solver Gurobi é possível resolver o problema de maneira exata. O modelo proposto, além de determinar os motoristas e seus respectivos passageiros, também propõe rotas para os veículos.

1.1 Objetivo Geral

Analisar o desempenho de diferentes métodos de otimização para o problema de carpooling.

1.2 Objetivos Específicos

Os objetivos específicos são:

1. Apresentar um modelo matemático para o problema de carpooling;
2. Indicar modos de codificação para o problema de carpooling para aplicação de meta heurísticas;
3. Propor e comparar resultados de meta-heurísticas cabíveis ao problema;
4. Aplicar métodos exatos de solução;
5. Aplicar métodos heurísticos de solução;
6. Aplicar métodos meta heurísticos de solução.

1.3 Justificativas

O carpooling se apresenta como um sistema que fornece vantagens aos seus usuários (BRUGLIERI et al., 2011). Referente ao tráfego, o sistema permite que mais

pessoas sejam transportadas utilizando uma menor quantidade de veículos, permitindo uma melhor circulação dos veículos (BRUGLIERI et al., 2011). De acordo com Hartman et al. (2014), o carpooling também pode auxiliar na redução da emissão de poluentes e na questão econômica dos usuários.

Entretanto, Correia e Viegas (2011) afirmam que uma grande quantidade de viagens são realizadas por apenas um usuário. Tal fato mostra a existência de dificuldades para compatibilizar motoristas e passageiros (HARTMAN et al., 2014). Desta maneira, Ferrari et al. (2003) apresentam algumas abordagens heurísticas para solução do problema de carpooling com diferentes objetivos.

Algumas das abordagens recentes do problemas, como as de Hartman (2013) e Xia, Curtin e Zhao (2015), buscam maximizar a compatibilidade entre motoristas e passageiros ou minimizar o custo dos motoristas. O presente trabalho mostra sua importância acadêmica ao apresentar uma melhoria de uma modelagem matemática do problema de carpooling, proposta por Eccel e Tagliarenha (2015), visando refletir melhor a realidade do problema.

Devido à complexidade do problema de carpooling, o presente trabalho também apresenta propostas de métodos aproximados para solução do mesmo bem como seus resultados.

1.4 Estrutura do trabalho

Para alcançar o objetivo proposto, este trabalho foi organizado da seguinte maneira. No Capítulo 1 é realizada a introdução do trabalho que será detalhado nos demais capítulos. O Capítulo 2 apresenta o referencial teórico do mesmo, onde são apresentados os conceitos necessários para melhor entendimento do trabalho. No Capítulo 3 é mostrado a metodologia adotada para realização das análises. No Capítulo 4 são expostos os resultados individuais dos métodos de otimização escolhidos para solução do problema de carpooling. No Capítulo 5 é realizada a análise comparativa de desempenho entre os métodos abordados. Por fim, no Capítulo 6 são apresentadas as considerações finais do presente trabalho.

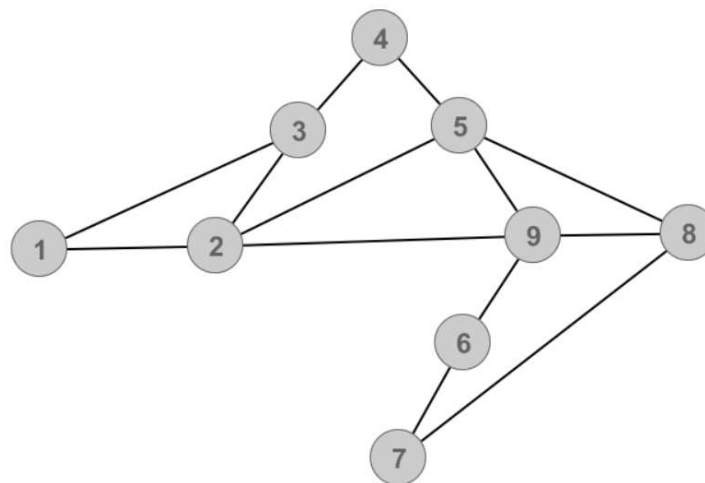
2 REVISÃO TEÓRICA

Neste capítulo são apresentados os conceitos necessários para entedimento do trabalho desenvolvido.

2.1 Grafos

Para o presente trabalho, é importante o conhecimento sobre alguns tópicos de grafos. Um grafo G pode ser considerado um par $G = (V, E)$, no qual V é considerado um conjunto, denominado conjunto de vértices, finito, discreto e não vazio, E é o conjunto de arestas, ou arcos, onde seus elementos são dados por $e = \{a, b\}$, no qual $a, b \in V$ (JUNGNICKEL, 2010). Um grafo pode ser representado visualmente, como na Figura 1, na qual pode-se observar que $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ e $E = \{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{2, 5\}, \{2, 9\}, \{3, 4\}, \{4, 5\}, \{5, 8\}, \{5, 9\}, \{6, 7\}, \{6, 9\}, \{7, 8\}, \{8, 9\}\}$.

Figura 1 – Exemplo de grafo



Fonte: Adaptado de Jungnickel (2010)

Para grafos maiores, a visualização dos vértices e arestas pode ser comprometida, desta forma, é mais interessante representá-lo por meio de uma lista de adjacência (BOAVENTURA NETTO, 2006). Existem duas maneiras de representar uma lista de adjacência, a primeira é criá-la a partir das arestas que saem de um vértice, a segunda é formá-la a partir dos arcos que chegam em um vértice. Para o grafo presente na Figura 1, a lista de adjacência podem ser verificadas na Figura 2.

Figura 2 – Exemplo de lista de adjacência

Vértices	
Origem	Destino
1	2,3
2	3,5,9
3	4
4	5
5	8,9
6	7,9
7	8
8	9
9	

Fonte: O Autor (2018)

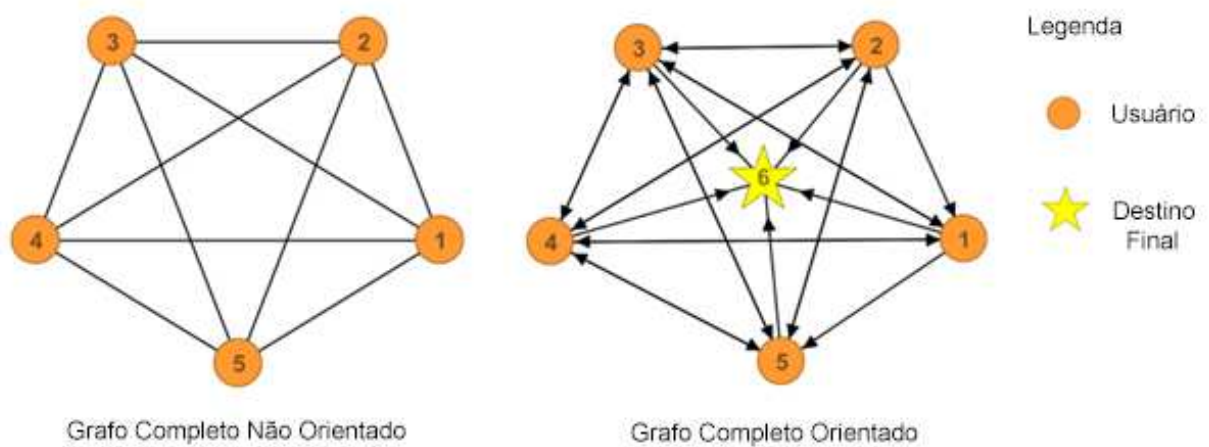
São apresentados a seguir alguns conceitos gerais dos grafos:

1. Valoração sobre as Arestas: Atribui-se o conjunto E a outro conjunto ou conjuntos de números por meio de uma ou mais funções;
2. Grau de Entrada de um Vértice (d_i): Quantidade de ligações que se destinam à um determinado vértice a ;
3. Grau de Saída de um Vértice (d_o): Quantidade de ligações que se originam de um determinado vértice a ;
4. Grafo Não Orientado: Para todo elemento $e \in E$ de G é possível escrever um G' de maneira que cada aresta de G é representada por duas arestas em sentidos opostos em G' ;
5. Grafo Orientado: Para todo elemento $e \in E$ de G não é possível escrever um G' de maneira que cada aresta de G é representada por duas arestas em sentidos opostos em G' ;
6. Grafo Simétrico: Para todo arco $\{a, b\}$ existe um correspondente $\{b, a\}$. Pode ser equivalente a um grafo não orientado;
7. Grafo Completo: Para todos elementos a e $b \in V$ existe ao menos uma aresta associada. Grafos Não Orientados Completos, recebem a notação K_n , na qual n é o número de vértices.

O grafo resultante do Problema do Carpooling, retirando o vértice de destino final, pode ser classificado como um Grafo Completo, podendo ser ou não, Simétrico. Ao inserir o vértice de destino final, o Grafo passa a ser um Grafo Orientado Completo. Na Figura 3 apresenta-se a transição de classificações ao ser inserido o vértice de destino no Grafo do Carpooling.

Pode-se observar na Figura 3 que as arestas presentes no Grafo Completo Não Orientado permanecem as mesmas quando é inserido o vértice de destino final. Porém, devido ao grau de saída do vértice de destino final ser zero e seu grau de entrada

Figura 3 – Efeito da inserção do destino final no grafo do carpooling



Fonte: O Autor (2018)

ser igual a n , que é o número de usuários, o Grafo passa a ser apenas Direcionado Completo.

Desta forma, os grafos que representam o problema de carpooling podem ser considerados como grafos completos orientados.

2.2 Carpooling

Qi, Wang e Wang (2016) afirmam que "nos últimos anos, cidades grandes enfrentam sérios problemas de congestionamento e poluição ambiental." (QI; WANG; WANG, 2016, p. 9294, tradução nossa). Qi, Wang e Wang (2016) ainda afirmam que na cidade de Shanghai, no ano de 2014, foram contabilizadas cerca de 55,5 milhões de viagens diárias, das quais 26,72 se referiam a propósitos diários. Desta forma, afim de mitigar os problemas gerados pelo excesso de tráfego, o sistema de carpooling tornou-se foco de estudos (QI; WANG; WANG, 2016).

De acordo com Hartman et al. (2014), o "Carpooling é um esquema segundo o qual várias pessoas partilham um veículo privado simultaneamente, afim de chegar a destinos comuns, ou próximos." (HARTMAN et al., 2014, p. 339), contribuindo assim, para a redução na quantidade de viagens diárias. Além de tal redução, Xia, Curtin e Zhao (2015) apontam, entre os benefícios do carpooling "custo reduzido do combustível, redução do custo de pedágio, redução no tempo para viagens diárias [...] e potencial redução de stress para os passageiros do veículo." (XIA; CURTIN; ZHAO, 2015, p. 1, tradução nossa).

Para que o sistema do carpooling seja estabelecido com sucesso, é necessário que passageiros e motoristas, diariamente, combinem seus horários de saída (HUSSAIN et al., 2014). Tal acordo pode gerar um desconforto e rigidez no sistema para

ambos. Aqueles usuários designados como motoristas, não possuem mais flexibilidade em sua rota diária. Já os passageiros, caso um motorista cancele sua viagem, correm o risco de não conseguirem utilizar o sistema de carpooling.

São apresentados a seguir, alguns modelos de otimização propostos por diferentes autores.

2.2.1 Modelo Carpooling - Maximização de Compatibilidade

Considerando $V = \{1, 2, \dots, n\}$ um conjunto de usuários do sistema de carpooling, Hartman (2013) propõe a seguinte modelagem matemática:

$$\text{Maximize } \sum_i \sum_j w_{i,j} x_{i,j} \quad (2.1)$$

$$y_i + \sum_j x_{i,j} \leq 1 \quad \forall i \in V \quad (2.2)$$

$$x_{i,j} \leq y_i \quad \forall i, j \in V \quad (2.3)$$

$$\sum_i x_{i,j} \leq c_j y_j \quad \forall j \in V \quad (2.4)$$

$$x_{i,j} \in \{0, 1\} \quad \forall i, j \in V \quad (2.5)$$

$$y_i \in \{0, 1\} \quad \forall i \in V \quad (2.6)$$

Em que:

- w_{ij} = Compatibilidade entre o passageiro i e o motorista j ;
- $x_{ij} = \begin{cases} 1, & \text{se o usuário } i \text{ viaja com o motorista } j \\ 0, & \text{c.c} \end{cases}$;
- c_j = Capacidade do veículo j ;
- $y_i = \begin{cases} 1, & \text{se o usuário } i \text{ é considerado motorista} \\ 0, & \text{c.c} \end{cases}$

A Equação (2.1) representa a função objetivo que busca maximizar compatibilidade entre motoristas e passageiros. A restrição (2.2) garante que um usuário indicado como motorista, não seja contabilizado como passageiro e caso selecionado como passageiro, não possa estar em outro veículo como motorista. A restrição (2.3) faz com que um usuário seja motorista caso exista alguém para ofertar carona e o mesmo possua um veículo. A restrição (2.4) não permite que a quantidade de pessoas em um veículo seja maior que sua capacidade. As restrições (2.5) e (2.6) se referem a integralidade das variáveis.

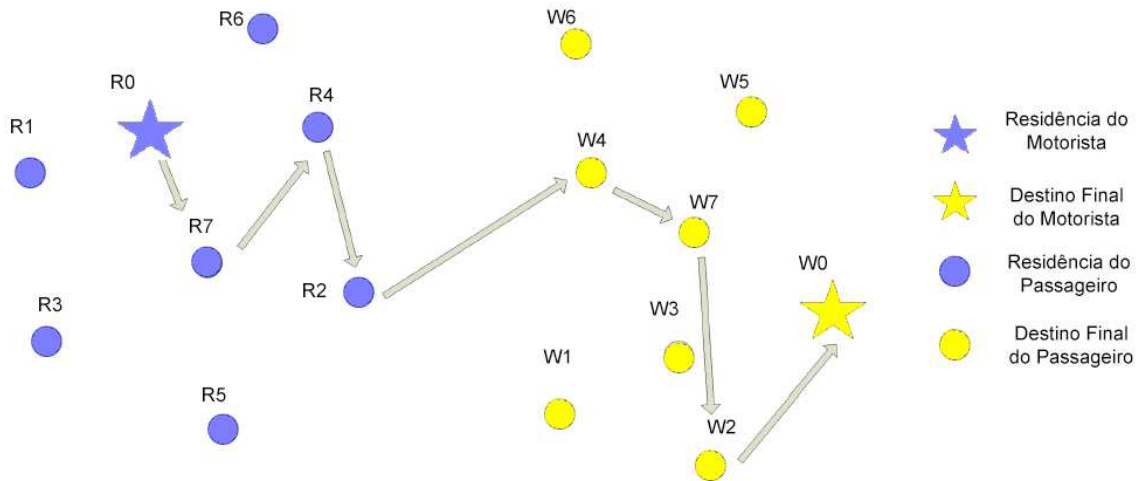
De acordo com Hartman et al. (2014), tal modelo permite que passageiros e motoristas possuam semelhanças entre diversos fatores, gosto musical, idade, gênero, etc, uma vez que os mesmos são contabilizados em w_{ij} . Em seu trabalho, Knapen et al.

(2015), demonstram as dificuldades quando este problema toma grandes proporções, uma vez que são necessários diversos procedimentos para encontrar a probabilidade do usuário i realizar a viagem com o motorista j .

2.2.2 Modelo Carpooling - Minimização de Custo

Em Xia, Curtin e Zhao (2015), propõe-se um modelo de minimização, onde otimiza-se principalmente o custo (distância, impedância, entre outros) entre usuários. Xia, Curtin e Zhao (2015) definem o problema de maneira que o motorista e seus passageiros não precisam ter o mesmo local de destino, como mostrado na Figura 4, na qual os vértices R0, R1, R2, R3, R4, R5, R6 e R7 representam as residências dos usuários e W0, W1, W2, W3, W4, W5, W6 e W7 seus respectivos destinos finais.

Figura 4 – Representação do sistema de carpooling



Fonte: Adaptado de Xia, Curtin e Zhao (2015)

Considerando p como o indexador dos candidatos a passageiros e motoristas, no qual $p = 1, 2, \dots, N$, em que N o número total de usuários do sistema. $k = 1, 2, \dots, N$ sendo as residências, $l = N + 1, N + 2, \dots, 2 * N$ os locais de trabalho. Seja também, $d_{i,j}$ o custo do arco entre os usuários i e j , que pode ser a distância, o tempo ou impedância, por exemplo. C representa a quantidade de passageiros no veículo (parâmetro definido pela capacidade do veículo). s representa a residência de um motorista. Os parâmetros r e f indicam arcos que compõem uma rota, de maneira que R é o número de arcos em uma rota de carpooling. A variável de decisão $x_{i,j,r}$ é binária e assume valor 1 caso o arco de i para j é percorrido no passo r de uma rota e 0 caso contrário. O modelo de programação linear proposto por Xia, Curtin e Zhao (2015) é descrito como:

$$\text{Minimize } \sum_{i=1}^{2*N} \sum_{j=1}^{2*N} \sum_{r=1}^R d_{i,j} x_{i,j} \quad (2.7)$$

Sujeito a:

$$\sum_{i=1}^{2*N} \sum_{k=1}^N \sum_{r=1}^R x_{i,k,r} = C \quad (2.8)$$

$$\sum_{i=1}^{2*N} \sum_{l=N+1}^{2*N} \sum_{r=1}^R x_{i,l,r} = C + 1 \quad (2.9)$$

$$\sum_{k=1}^N x_{s,k,l} = 1 \quad (2.10)$$

$$\sum_{l=N+1}^{2*N} x_{l,t,R} = 1 \quad (2.11)$$

$$\sum_{i=1}^{2*N} x_{i,k,r} - \sum_{i=1}^{2*N} \sum_{f=r+1}^R x_{i,(k+N),f} = 0 \quad \forall k = 1, \dots, N; k \neq s; r = 1, \dots, R - 1 \quad (2.12)$$

$$\sum_{i=1}^{2*N} \sum_{r=1}^R x_{i,j,r} \leq 1 \quad \forall j = 1, \dots, 2 * N \quad (2.13)$$

$$\sum_{j=1}^{2*N} \sum_{r=1}^R x_{i,j,r} \leq 1 \quad \forall i = 1, \dots, 2 * N \quad (2.14)$$

$$\sum_{i=1}^{2*N} x_{i,j,r} - \sum_{i=1}^{2*N} x_{j,i,(r+1)} = 0 \quad \forall j = 1, \dots, 2 * N; r = 1, \dots, R - 1 \quad (2.15)$$

$$\sum_{i=1}^{2*N} \sum_{j=1}^{2*N} x_{i,j,r} = 1 \quad \forall i = 1, \dots, 2 * N \quad (2.16)$$

A função objetivo, dada pela Equação (2.7), busca minimizar o custo global do sistema. Em (2.8), o motorista deve buscar uma quantidade C de passageiros e a Equação (2.9) faz com que C passageiros sejam levados aos seus destinos. A restrição (2.10) faz com que o início da rota seja a residência do motorista, já em (2.11) determina que o fim da rota seja o destino do motorista. Em (2.12), um passageiro não pode levar um passageiro ao seu destino, ao menos que o passageiro tenha sido buscado anteriormente. A restrição (2.13) garante que apenas um arco pode chegar em um vértice, já em (2.14) estabelece que apenas um arco pode sair de um vértice. A Equação (2.15) garante a continuidade da rota e em (2.16) garante que apenas um arco será escolhido por vez na rota.

O modelo de programação linear proposto por Xia, Curtin e Zhao (2015), embora busque minimizar a distância total percorrida pelos motoristas, não considera restrições referentes ao tempo de viagem máximo ou tempo de disponibilidade dos usuários.

2.2.3 Modelo Carpooling - Maximização do Número de Usuários

O modelo matemático (2.17) - (2.28), apresentado a seguir, proposto por Eccel e Tagliapietra (2015) mostra uma função objetivo baseada na maximização da quantidade de usuários como passageiros. O modelo é baseado em de Hartman et al. (2014), apresentado na Seção 2.2.1, com a adaptação da função objetivo e das restrições.

Considerando $x_{i,j,k}$ como sendo a variável de decisão, onde toma o valor de 1 caso o usuário i viaje até o passageiro j com o motorista k e 0 caso contrário; cap_k é a capacidade do veículo do motorista k ; $tmax_k$ se refere à janela de tempo máxima do motorista k ; $t_{i,j}$ o tempo de viagem do usuário i ao j ; y_k um variável de decisão que assume o valor de 1 caso o usuário k seja selecionado como um motorista e 0 caso contrário; B uma constante muito grande e S um parâmetro para impedir a formação de subciclos no grafo.

$$\text{Maximizar } \sum_i \sum_j \sum_k x_{i,j,k} \quad (2.17)$$

Sujeito a:

$$\sum_i \sum_j x_{i,j,k} \leq cap_k \quad \forall k \quad (2.18)$$

$$\sum_i \sum_k x_{i,j,k} \leq 1 \quad \forall j \quad (2.19)$$

$$\sum_j \sum_k x_{i,j,k} \leq 1 \quad \forall i \quad (2.20)$$

$$\sum_i \sum_j x_{i,j,k} t_{i,j} \leq tmax_k \quad \forall k \quad (2.21)$$

$$\sum_j x_{i,j,k} \leq \sum_j x_{j,i,k} \quad \forall i; \forall k \quad (2.22)$$

$$\sum_i \sum_j x_{i,j,k} \leq B(1 - \sum_1 x_{k,j,k}) \quad \forall j; \forall k \quad (2.23)$$

$$\sum_i \sum_j x_{i,j,k} \leq B y_k \quad \forall k \quad (2.24)$$

$$\sum_j x_{k,j,k} = y_k \quad \forall k \quad (2.25)$$

$$\sum_i x_{i,n+1,k} = y_k \quad \forall k \quad (2.26)$$

$$\sum_i \sum_j x_{i,j,k} \leq |S| - 1 \quad \forall S \subset K \quad (2.27)$$

$$x_{i,j,k}, y_k \in 0, 1 \quad \forall i; \forall j; \forall k \quad (2.28)$$

No qual a equação (2.17) representa a função objetivo do modelo, na qual busca-se maximizar a quantidade de usuários do carpooling como passageiros não

contabilizando os motoristas. Em (2.18) considerada-se a quantidade de vagas para passageiros do k -ésimo veículo, a qual não deve ser ultrapassada. A restrição (2.19) garante que o grau de entrada do vértice referente ao usuário i menor ou igual a 1 e (2.20) pode ser considerado o oposto da restrição (2.19), ou seja, garante que o grau de saída do vértice i seja menor ou igual a 1, juntas garantem que cada usuário é transportado por no máximo um veículo.

A restrição (2.21) considera a janela de tempo máxima de um motorista, na qual é contabilizado o tempo de ir de um usuário i a outro j e de j ao destino final $n + 1$ que é forçado pela restrição (2.26). Desta forma, caso o tempo de deslocamento de i a j e de j a $n + 1$ pelo motorista k seja superior a $tmax_k$ então a viagem não é realizada. Para o problema em questão, é considerado que um passageiro só pode viajar com determinado motorista se o mesmo estiver disponível.

A restrição (2.22) refere-se ao balanceamento de fluxo no grafo, ou seja, se um motorista chega a um passageiro, obrigatoriamente ele deve sair do mesmo. Tal restrição se aplica para todos os nós, exceto para o destino final, do qual os motoristas não podem ir a outro lugar.

Quando se maximiza a quantidade de usuários que viajam como passageiros, automaticamente reduz-se a quantidade de veículos circulando nas vias. Desta forma busca-se que nem todos os motoristas utilizem seus veículos, e uma vez que um motorista é selecionado como passageiro, seu veículo não pode ser utilizado. A restrição (2.23) garante tal situação. Em (2.24) permite-se alocar usuários em um veículo somente se usuário k é selecionado como motorista.

Em (2.25) os motoristas são colocados como ponto inicial da rota. A restrição (2.27) impede que sejam formados subciclos nas rotas dos motoristas. Por fim, a restrição (2.28) se refere a integralidade das variáveis de decisão.

Embora Eccel e Tagliarenha (2015) tenham elaborado um modelo matemático para o problema de carpooling o mesmo foi aplicado apenas para um problema pequeno com 5 usuários.

Neste trabalho apresenta-se um modelo de forma a considerar as restrições de janela de tempo que cada usuário está disposto a considerar ao participar do sistema, já considerando o tempo de espera e de deslocamentos de forma que os motoristas possuam uma janela de tempo máxima para buscar os passageiros. No entanto, quando se considera essa restrição, pode ocorrer que nem todos os possíveis usuários sejam admitidos no sistema, já que a soma do tempo da viagem do ponto de saída do motorista até o passageiro, e deste até o destino final deve ser inferior ao tempo máximo que o motorista possui para buscar os outros usuários. Tal modelo será melhor abordado na Seção 4.1.

2.3 Complexidade

De acordo com Hillier e Lieberman (2013), no âmbito da PO, modelos exatos fornecem a resposta ótima para problemas de otimização, porém este tipo de modelagem não tem êxito em todos os problemas.

Alguns problemas (e os modelos de PO correspondentes) são tão complicados que pode não ser possível encontrar uma solução ótima. Nessas situações, ainda é importante encontrar uma boa solução viável, que seja pelo menos razoavelmente próxima da solução ótima (HILLIER; LIEBERMAN, 2013, p. 581).

Portanto, os diferentes problemas de otimização possuem diferentes níveis de complexidade. A complexidade de um problema pode ser medida principalmente pelo tempo computacional necessário para se obter uma solução, ou pela quantidade de memória necessária. Contudo, podem ser utilizados outros critérios de desempenho, os quais o analista considere relevante (COLIN, 2013). Os problemas mais comuns se dividem em duas grandes classes: Problemas Polinomiais (P) e Não Polinomiais (NP). Os problemas da classe P são aqueles que possuem solução em tempo polinomial em função da quantidade de entrada de dados. Já os problemas da classe NP são os que não possuem solução em tempo polinomial.

Dentre os mais difíceis problemas da classe NP, estão os problemas NP-Hard e NP-Completo. De acordo com Goldberg e Luna (2005), os problemas da classe NP possuem um elevado número de combinações possíveis, o que torna os métodos de enumeração de soluções inviáveis. Desta forma, devido à grande explosão na quantidade de combinações ao se acrescentar variáveis, o tempo de solução desta classe de problemas é não polinomial (GOLDBARG; LUNA, 2005). Para ambos os casos o tempo de solução de um algoritmo é definido pelo tempo necessário para solucionar o problema dada uma entrada de dados de tamanho n , no pior caso (COLIN, 2013). Em geral, problemas inclusos nestas classes, possuem elevado tempo de processamento. Devido a este fato, a espera pela solução ótima torna-se inviável.

O problema do carpooling pode ser apresentado como um problema da classe NP-Hard, devido à sua natureza combinatorial (HARTMAN et al., 2014). Em seu trabalho, Hartman et al. (2014) demonstram, por meio de provas matemáticas, os motivos para o problema se comportar deste modo.

2.4 Heurísticas

Métodos Heurísticos são procedimentos utilizados para encontrar soluções viáveis para um problema de PO, tais métodos não possuem garantia otimalidade (HILLIER; LIEBERMAN, 2013).

Ainda, de acordo com Hillier e Lieberman (2013):

O procedimento também deve ser suficientemente capaz para lidar com problemas muito grandes. O procedimento normalmente é um algoritmo iterativo completo em que cada iteração envolve a condução da procura de uma nova solução que poderia ser melhor que a melhor solução encontrada previamente. Quando o algoritmo termina após um tempo razoável, a solução por ele fornecida é a melhor que foi encontrada durante qualquer iteração (HILLIER; LIEBERMAN, 2013, p. 581).

As heurísticas podem ser construtivas, de melhoria ou compostas (construtivas e de melhoria) (BODIN et al., 1983). As heurísticas construtivas buscam a partir de determinadas informações, construir a solução. Diferentemente das heurísticas de melhoria, nas quais se possui uma solução inicial e busca-se, por meio de determinadas operações, melhorar tal solução (BODIN et al., 1983). As heurísticas compostas, segundo Bodin et al. (1983), possuem os dois procedimentos em conjunto, em que se constrói uma solução inicial que é melhorada posteriormente.

Arenales et al. (2015) também mostra métodos heurísticos de busca local ou busca em vizinhança. Em tais métodos busca-se gerar um conjunto de soluções a partir de uma solução inicial por meio de alterações na solução inicial (ARENALES et al., 2015). Desta forma, escolhe-se algum vizinho que melhore a solução antecessora, a busca é encerrada quando não há melhoria da função objetivo (ARENALES et al., 2015).

Desta forma, neste capítulo será apresentado um método heurístico para solução do problema do carpooling.

2.4.1 Heurística Gulosa

Uma heurística gulosa, de modo geral, busca sempre encontrar uma solução por meio de soluções locais, busca-se sempre a melhor solução local para o procedimento guloso. No âmbito de problemas de roteirização, a heurística gulosa, também pode ser denominada como Heurística do Vizinho mais Próximo. De acordo com Taha (2008), a heurística possui como procedimento a escolha de um vértice inicial e a partir deste caminha-se para um próximo vértice que possui o menor custo ainda não inserido na solução, e a partir desta inserção repete-se o processo novamente, o método é repetido até que um ciclo hamiltoniano ¹ seja formado.

A heurística do vizinho mais próximo foi proposta por Bellmore e Nemhauser (1968) e pode ser considerada uma heurística construtiva que busca construir a solução por meio de soluções ótimas locais (ARENALES et al., 2015). Desta forma, em geral a solução encontrada é um ótimo local. O algoritmo de Bellmore e Nemhauser (1968), de acordo com Rosenkrantz, Stearns e Il (1977), aplicado ao problema do caixeiro viajante, pode ser descrito em três etapas:

¹ Um ciclo hamiltoniano consiste em um circuito em que todos os nós são visitados apenas uma vez (GOLDBARG; LUNA, 2005)

1. Comece com um vértice arbitrário;
2. Encontre um vértice ainda não inserido no caminho o qual é o mais próximo do último vértice adicionado e adicione a aresta que conecta estes vértices;
3. Repita o passo 2 até que todos os vértices tenham sido adicionados e adicione a aresta que conecta o último vértice adicionado ao vértice de início.

Tal algoritmo se demonstra facilmente executável para o problema do caixeiro viajante, problema que consiste em sair de um vértice inicial, percorrer todos os demais vértices do grafo e retornar ao vértice inicial. Entretanto, para o problema do carpooling, são necessárias modificações no algoritmo. As modificações necessárias para que a heurística gulosa seja executada para o problema de carpooling serão apresentadas na Seção 4.2.

2.5 Meta Heurísticas

Um procedimento meta heurístico pode ser classificado como uma metodologia que auxilia o desenvolvimento de métodos heurísticos por meio do fornecimento de estruturas e diretrizes de estratégias, atendendo as características do problema (HILLIER; LIEBERMAN, 2013). O principal objetivo da meta heurística está em tentar encontrar a solução ótima por meio da fuga de soluções ótimas locais.

Nesta Seção serão apresentados alguns métodos meta heurísticos utilizados no presente trabalho.

2.5.1 Meta Heurísticas VNS e VND

O procedimento *Variable Neighborhood Search* (VNS) é uma meta heurísticas de vizinhança proposta por Mladenovic e Hansen (1997) a qual busca encontrar uma solução viável de um problema a partir de variações pré-definidas. A meta heurística VNS busca explorar vizinhanças variadas mais distantes da solução incubente como meio de escapar de mínimos locais (HANSEN; MLADENOVIC, 2001). O principal passo do algoritmo realiza uma alteração na solução incubente, em uma determinada vizinhança, a partir desta solução alterada, faz-se uma busca local na vizinhança desta nova solução, tal busca local pode ser determinada de diversas formas, por tempo computacional até mesmo por regras determinísticas (MLADENOVIC; HANSEN, 1997).

Caso o vizinho encontrado seja melhor que a solução incubente atual, então a nova solução é aceita. Caso contrário, realiza-se uma nova alteração em uma vizinhança diferente. Tais passos são realizados até que um determinado número de vizinhanças tenha sido explorado. A estrutura do algoritmo VNS proposto por Mladenovic e Hansen (1997) está apresentada na Figura 5, o qual possui os parâmetros de entrada: x^* que representa a melhor solução encontrada até o momento (incubente) e k_{max} o qual indica a quantidade máxima de vizinhanças a serem exploradas. Os

Figura 5 – Algoritmo VNS

```

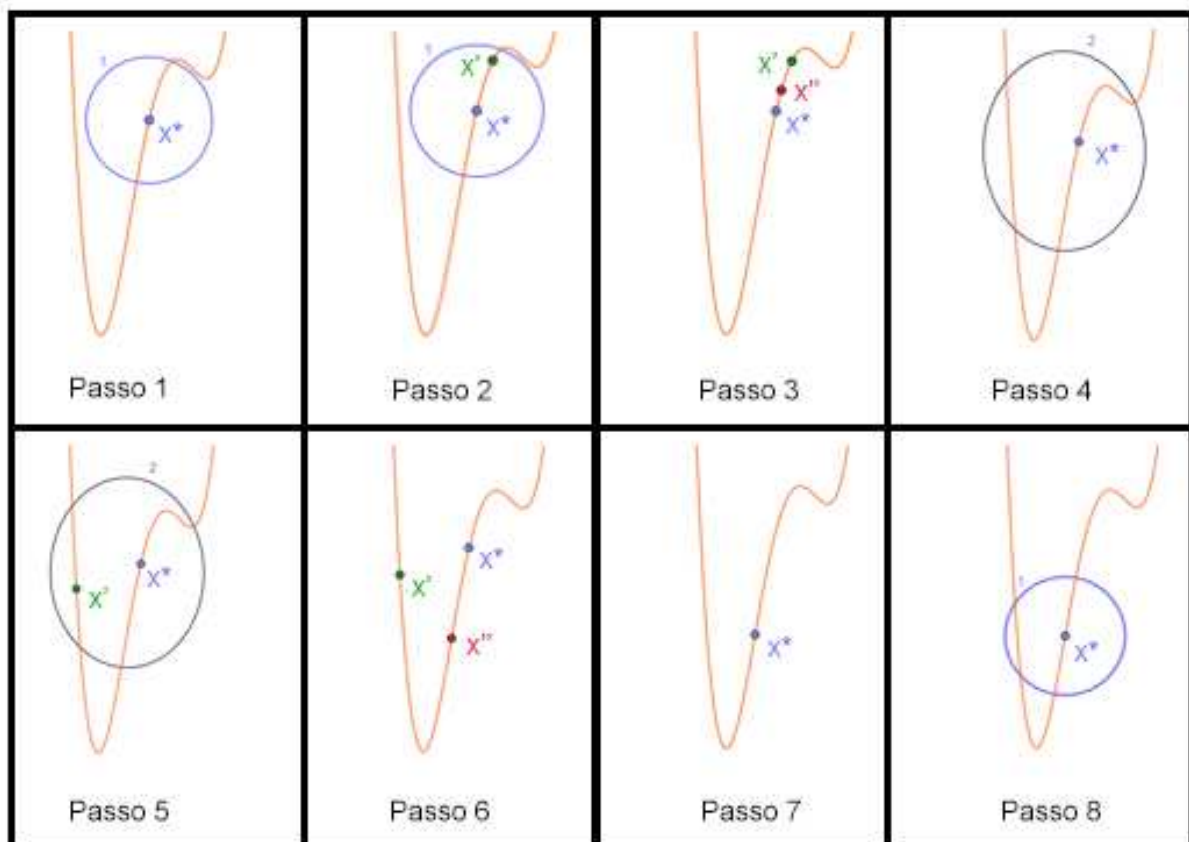
1 início
2    $k \leftarrow 1$ ;
3   enquanto  $k < k_{max}$  faça
4      $x' \leftarrow$  Vizinho aleatório de  $x^*$  na vizinhança  $k$ ;
5      $x'' \leftarrow$  Busca Local em  $x'$ ;
6     se  $x''$  Melhor que  $x^*$  então
7        $x^* \leftarrow x''$ ;
8        $k \leftarrow 1$ ;
9     fim
10    senão
11       $k \leftarrow k + 1$ ;
12    fim
13  fim
14 fim

```

Fonte: Adaptado de Mladenovic e Hansen (1997)

parâmetros k se refere a atual vizinhança que serão buscados vizinhos, x' é o vizinho aleatório da solução incubente e x'' é a melhor solução encontrada pela busca local em x' .

Figura 6 – Funcionamento do Algoritmo VNS



Fonte: O Autor (2018)

A Figura 6 apresenta o funcionamento do algoritmo VNS em um problema

genérico de minimização. O passo 1 mostra a região de vizinhança 1 selecionada para a solução incubente x^* . Em 2, escolhe-se o vizinho aleatório x' na vizinhança. O passo 3 apresenta a solução encontrada x'' a partir de x' por meio de um método de busca local. Como x'' não é melhor que x^* , então a solução incubente não é modificada e seleciona-se uma nova região de vizinhança 2 (passo 4). Nos passos 5 e 6 são repetidos os procedimentos utilizados nos passos 1 e 2, entretanto, nesta situação como a solução x'' é melhor que x^* , então a solução incubente é modificada para a posição de x'' e o procedimento é iniciado novamente a partir da vizinhança 1 (passos 7 e 8).

Quando utiliza-se a meta heurística VNS com mais de uma vizinhança, alguns questionamentos precisam ser levados em considerações, tais como: Quais e quantas vizinhanças serão utilizadas, qual deve ser a ordem da busca e qual a estratégia para mudar de vizinhança (MLADENOVIC; HANSEN, 1997). Desta maneira, Hansen e Mladenovic (2001) citam que diversas soluções podem ser geradas dependendo de como as vizinhanças são escolhidas e em alguns procedimentos podem ocorrer ciclos caso não forem selecionadas de maneira eficiente.

O algoritmo Variable Neighborhood Descent (VND) possui um procedimento semelhante ao VNS, porém a busca local não é aleatória em uma determinada vizinhança, mas a melhor solução naquela vizinha (HANSEN; MLADENOVIC, 2001). Desta forma, todas as possibilidades de uma vizinhança são determinadas de uma maneira determinística (GAO; SUN; GEN, 2008). O algoritmo para execução da meta heurística VND pode ser verificado na Figura 7, o qual possui os mesmos parâmetros de entrada e de processamento do algoritmo VNS. Vale indicar que a meta heurística VND pode ser utilizada como o método de procura local do algoritmo VNS (HANSEN; MLADENOVIC, 2001).

Figura 7 – Algoritmo VND

```

1 início
2    $k \leftarrow 1$ ;
3   enquanto  $k < k_{max}$  faça
4      $x' \leftarrow$  Melhor Vizinho de  $x^*$  na vizinhança  $k$ ;
5     se  $x'$  Melhor que  $x^*$  então
6        $x^* \leftarrow x'$ ;
7        $k \leftarrow 1$ ;
8     fim
9     senão
10       $k \leftarrow k + 1$ ;
11    fim
12   $k \leftarrow k + 1$ ;
13 fim
14 fim
```

Fonte: Adaptado de Hansen e Mladenovic (2001)

2.5.2 Simulated Annealing

A meta heurística Simulated Annealing (SA), ou Recozimento Simulado, consiste na simulação de um processo físico de recozimento aplicado à problemas combinatoriais proposto por Kirkpatrick, Gelatt e Vecchi (1983). "Metropolis et al. (1953), nos primórdios da computação científica, introduziram um algoritmo simples que pode ser usado para uma simulação eficiente da colisão de átomos no equilíbrio, dada uma temperatura."(KIRKIPATRICK; GELATT; VECCHI, 1983, p. 672), trabalho que posteriormente seria utilizado por Kirkpatrick, Gelatt e Vecchi (1983) para esquematizar a meta heurística SA.

O procedimento de recozimento, na metalurgia, consiste em aquecer, em geral, um metal à altas temperaturas, para posteriormente resfriá-lo de maneira controlada. O aquecimento permite que os átomos estejam com elevada energia, permitindo que se movam para posteriormente encontrarem o melhor posicionamento antes do endurecimento. A taxa de resfriamento depende do objetivo final do recozimento, quanto mais lento o resfriamento, mais organizados estarão os átomos ao final do processo.

De maneira análoga, o procedimento SA, descrito na Figura 8, parte de uma solução inicial S_0 , a qual pode ser completamente aleatória, ser solução parcial do problema, ou ser solução final de uma heurística, e uma temperatura inicial T_0 . Considerando ΔE^+ como a média aritmética da diferença de n soluções aleatórias e $\epsilon_0 = 0.8$ um valor empírico. Tal temperatura inicial pode ser determinada pela Equação (2.29), ou por meio de simulações em diferentes temperaturas.

$$T_0 = -\frac{\Delta E^+}{\ln(\epsilon_0)} \quad (2.29)$$

Para determinar o valor de ΔE^+ são geradas n soluções aleatórias e calculada a variação entre as soluções, da primeira para a segunda, da segunda para a terceira e assim por diante. A média das variações as soluções é o valor de ΔE^+ .

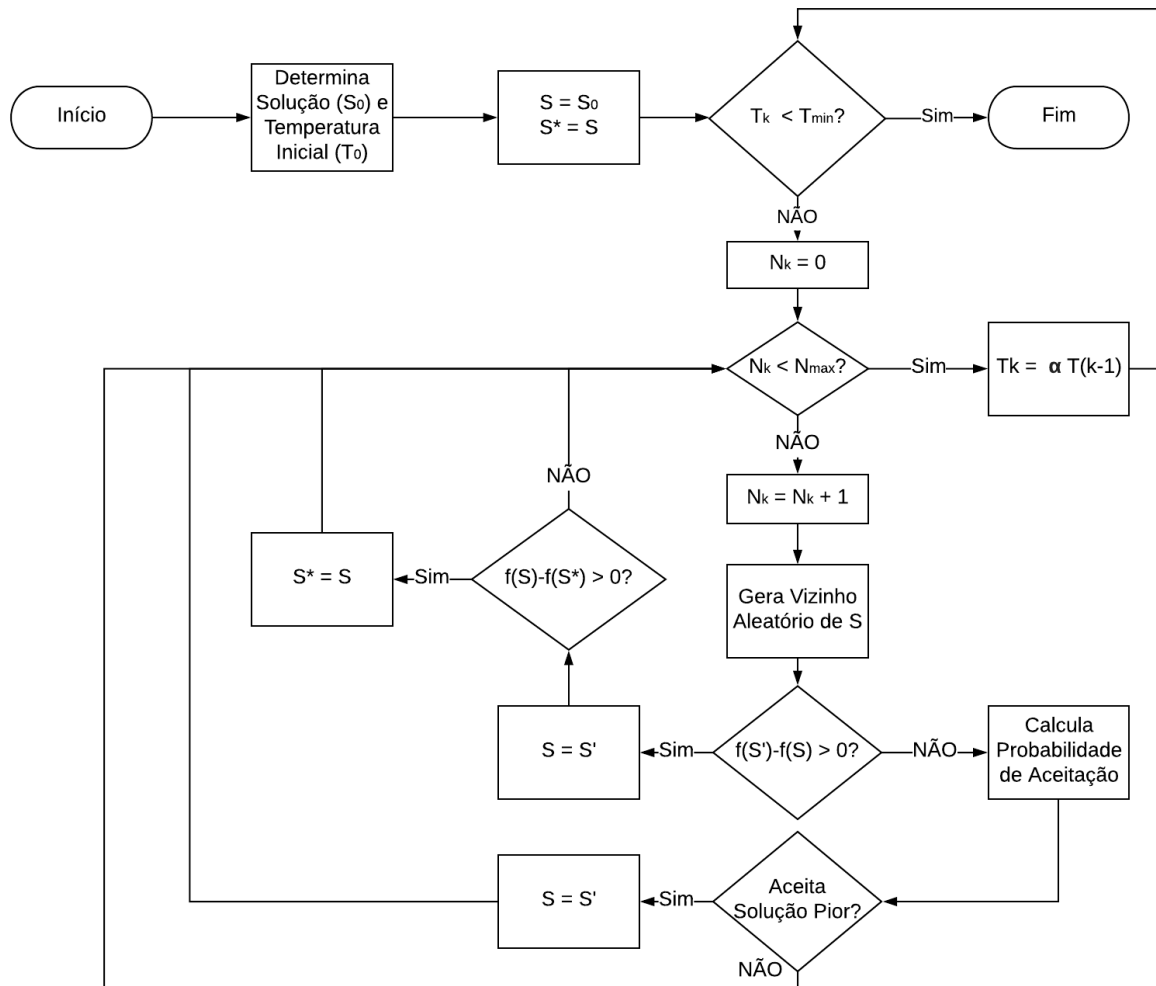
A solução inicial, em analogia ao processo físico, pode ser submetida em analogia ao metal em temperaturas elevada, que permitirá seus átomos se moverem.

A cada iteração k do procedimento, é determinada uma temperatura T_k que é atualizada por um fator de resfriamento α , o qual varia entre $\{0, 1\}$. É realizada uma perturbação na solução inicial que gera uma nova solução S' . As perturbações são movimentos válidos dentro das restrições do problema. Então calcula-se

$$\Delta E = f(S') - f(S) \quad (2.30)$$

o qual é a energia do movimento, para problemas de maximização, se a energia for positiva, a nova solução S' é aceita e torna-se a solução incubente. Em caso contrário, Kirkpatrick, Gelatt e Vecchi (1983), afirma que o caso deve ser tratado probabilisticamente, segundo a Equação (2.31) adaptada para o problema de

Figura 8 – Fluxograma do procedimento SA



Fonte: Adaptado de Araujo (2001)

maximização, pois a equação de probabilidade só será utilizada caso $\Delta E < 0$. Tal estratégia é utilizada para escapar de mínimos locais, permitindo que a função objetivo tenha um valor de piora (ARAUJO, 2001). A solução incubente é trocada caso a solução temporária obter um valor de função objetivo melhor.

$$P(S') = \begin{cases} 1, & \text{se } \Delta E \geq 0 \\ \exp(\Delta E/T_k), & \text{c.c} \end{cases} \quad (2.31)$$

Para cada temperatura são realizadas N_{max} iterações, na qual são realizadas as perturbações e geradas novas soluções S' . A Figura 8 esquematiza o procedimento do algoritmo SA para um problema de maximização. O procedimento termina quando uma temperatura mínima t_{min} é alcançada.

Pode-se observar que o algoritmo SA permite uma determinada flexibilidade na escolha dos vizinhos aleatórios. Desta maneira, para um mesmo problema, pode haver

diferentes formas de escolha de vizinhanças que podem levar a soluções melhores ou não.

3 METODOLOGIA

Para alcançar os objetivos propostos neste trabalho, buscou-se utilizar os procedimentos que podem ser verificados na Figura 9.

Figura 9 – Fluxograma da Metodologia Utilizada



Fonte: O Autor (2018)

- Etapa 1: Na Etapa 1 foi realizada uma pesquisa bibliográfica sobre os temas referentes ao trabalho e proposto: Grafos, carpooling, heurísticas e meta-heurísticas são alguns exemplos de temas relacionados ao trabalho.
- Etapa 2: As informações necessárias para aplicação do modelo são levantadas na Etapa 2. Foi escolhido o cenário da Universidade Federal de Santa Catarina (UFSC) Campus Joinville (CTJ). A pesquisa foi disponibilizada de maneira online apenas para os estudantes do CTJ. Inicialmente, questionou-se qual o atual modo de locomoção do indivíduo até a universidade (Ônibus, Carona, Transportes Ativos e Veículo Próprio). Para aqueles que possuíam veículo próprio foi questionado se estariam dispostos à ofertar caronas diárias, caso o pesquisado respondesse negativamente, a pesquisa era encerrada. Caso contrário, questionou-se os dados referentes à capacidade do veículo, tempo disponível para buscar os caroneiros, local de partida da rota, tipo de veículo utilizado e informações dos horários

de saída diários. Para aqueles que não possuíam veículo próprio, considerados possíveis passageiros, eram realizadas apenas perguntas referentes ao possível ponto de encontro com o motorista e sobre os horários de saída diários.

- Etapa 3: Na Etapa 3 foi realizada a elaboração de um modelo matemático de otimização no qual fosse possível refletir a realidade do problema. Buscou-se identificar todas as restrições que caracterizam o problema de carpooling.
- Etapa 4: Na Etapa 4, são aplicados os métodos de solução para o problema de carpooling. Foram considerados: o método exato e os métodos aproximados. Da gama de métodos aproximados estudados na literatura, isto é, algoritmo genético, colônia de formigas, busca tabu, entre outros, foram selecionados quatro para serem utilizados. Os métodos aproximados escolhidos para o problema de carpooling foram a heurística gulosa, a meta heurística Simulated Annealing, a meta heurística *Variable Neighborhood Search* (VNS) e *Variable Neighborhood Descent* (VND).
- Etapa 5: Para garantir que o modelo exato de otimização proposto, o qual será apresentado na Seção 4.1, estivesse refletindo a realidade, foi necessário testá-lo em um problema controlado, que seja possível encontrar a solução por meio de enumeração por exemplo. Este procedimento foi efetuado na Etapa 5.
- Etapa 6: Após realizada a Etapa 5, foi possível prosseguir para a Etapa 6, na qual os métodos de solução foram aplicados ao problema completo, com os dados levantados da Etapa 2 e o modelo proposto da Etapa 3. Os métodos aproximados, para aplicação do problema maior, foram programados na linguagem de programação C++.
- Etapa 7: A Etapa 7 consiste na análise de resultados dos métodos de solução propostos isoladamente, afim de apenas coletar informações para etapas posteriores. Os dados coletados de cada método foram, principalmente: valor da solução obtida e tempo de processamento do método.
- Etapa 8: O objetivo da Etapa 8 é avaliar o desempenho dos métodos de solução propostos por meio de critérios de avaliação apresentados na Etapa 7 e comparações entre os mesmos.

4 MÉTODOS DE SOLUÇÃO

Como descrito no Capítulo 3, foram aplicados os seguintes métodos de solução: Modelo Exato, Heurística Gulosa, Meta Heurística Simulated Annealing, Meta Heurística *Variable Neighborhood Search* e Meta Heurística *Variable Neighborhood Descent*, as quais serão detalhadas nas próximas seções. O computador utilizado para processar todos os métodos de solução tinha as seguintes configurações:

1. Processador Inter(R) Core(TM) i7-4700HQ CPU @ 2.40GHz
2. Memória RAM 8.00 GB (7,89 GB usáveis)
3. Tipo de sistema 64-bit
4. Sistema operacional Windows 8.1

4.1 Solução via Modelo Exato

O modelo matemático de carpooling de proposto (MMCP) neste trabalho foi baseado no modelo (2.17)-(2.28), apresentando em (ECCEL; TAGLIALENHA, 2015), acrescentando restrições dos usuários relacionados ao tempo de saída, de espera ou de deslocamento. Desta maneira, as restrições (4.6) e (4.7) apresentadas a seguir, foram acrescentadas ao modelo (2.28)-(2.17) apresentado na Seção 2.2.3, de modo que para o modelo matemático (4.1) - (4.16), apresentado a seguir, consideremos:

- m o número de motoristas;
- $M = 1, 2, \dots, m$ o conjunto de motoristas;
- n o número de usuários do sistema, motoristas e passageiros;
- $N = 1, 2, \dots, n$ o conjunto de usuários;
- $tinicial_i$ o tempo em que o usuário i estará disponível no ponto de encontro;
- $tespera$ o tempo máximo que um usuário está disposto a esperar por um motorista;

Vale ressaltar que $tinicial_i$ para a presente aplicação é expresso em minutos, portanto um usuário acessível às 08:00h, na base de dados seu $tinicial_i$ será 480 minutos. No modelo considera-se o destino final de todos como um passageiro artificial, o qual é denominado $n + 1$.

O MMCP para resolver o problema de carpooling neste trabalho pode ser representado pelas Equações (4.1)-(4.16) apresentadas a seguir:

$$\text{Maximizar } \sum_i^n \sum_j^n \sum_k^m x_{i,j,k} \quad (4.1)$$

Sujeito a:

$$\sum_i^n \sum_j^n x_{i,j,k} \leq cap_k \quad \forall k \in M \quad (4.2)$$

$$\sum_i^n \sum_k^m x_{i,j,k} \leq 1 \quad \forall j \in N \quad (4.3)$$

$$\sum_j^{n+1} \sum_k^m x_{i,j,k} \leq 1 \quad \forall i \in N \quad (4.4)$$

$$\sum_i^{n+1} \sum_j^{n+1} x_{i,j,k} t_{i,j} \leq tmax_k \quad \forall k \in M \quad (4.5)$$

$$tinicial_i + x_{i,j,k} t_{i,j} \geq x_{i,j,k} tinicial_j \quad \forall i, j \in N; k \in M \quad (4.6)$$

$$x_{i,j,k} (tinicial_i + t_{i,j} - tinicial_j) \leq tespera \quad \forall i, j \in N; k \in M \quad (4.7)$$

$$\sum_j^{n+1} x_{i,j,k} \leq \sum_j^n x_{j,i,k} \quad \forall i \in N; k \in M; i \neq k \quad (4.8)$$

$$\sum_i^n \sum_j^n x_{i,j,k} \leq BigM(1 - \sum_i^n x_{i,k,l}) \quad \forall k, l \in M; l \neq k \quad (4.9)$$

$$\sum_i^n \sum_j^n x_{i,j,k} \leq BigM y_k \quad \forall k \in M \quad (4.10)$$

$$\sum_j^n x_{k,j,k} = y_k \quad \forall k \in M \quad (4.11)$$

$$\sum_i^n x_{i,n+1,k} = y_k \quad \forall k \in M \quad (4.12)$$

$$x_{i,j,k} + x_{j,i,k} \leq 1 \quad \left\{ \begin{array}{l} \forall i = 1, \dots, n; j = 2, \dots, n \\ k \in M; i \neq j \end{array} \right. \quad (4.13)$$

$$x_{i,j,k} + x_{j,l,k} + x_{l,i,k} \leq 2 \quad \left\{ \begin{array}{l} \forall i = 1, \dots, n; j = 2, \dots, n \\ \forall l = 3, \dots, n; k \in M \\ i \neq j; j \neq l \end{array} \right. \quad (4.14)$$

$$x_{i,j,k} + x_{j,l,k} + x_{l,h,k} + x_{h,i,k} \leq 3 \quad \left\{ \begin{array}{l} \forall i = 1, \dots, n; j = 2, \dots, n \\ \forall l = 3, \dots, n; h = 4, \dots, n \\ k \in M \\ i \neq j; j \neq l; l \neq h \end{array} \right. \quad (4.15)$$

$$x_{i,j,k}, y_k \in 0, 1 \quad \forall i, j, k = 1, \dots, n+1 \quad (4.16)$$

Em (4.6) garante-se que o motorista k só realizará a viagem do nó i ao nó j se o usuário do nó j estiver disponível após o tempo de deslocamento do nó i ao nó j . Na

restrição (4.7) considera-se que um usuário só pode esperar um tempo máximo pelo motorista.

As restrições (4.13), (4.14) e (4.15) impedem que sejam formados subciclos, ressalta-se que aqui foram considerados ser formados subciclos com no máximo 4 passageiros devido à capacidade máxima dos veículos. Caso houvessem veículos com capacidades maiores, seriam necessárias restrições adicionais. Tais restrições equivalem a restrição (2.27) apresentada no modelo (2.17)-(2.28), no entanto, com menor complexidade. As demais restrições e parâmetros do modelo são especificados na Seção 2.2.3.

4.1.1 Cenário de Validação

Para validar o MMCP, aplicou-se o problema proposto para o conjunto de dados apresentado nas Tabelas 1 e 2, nas quais considera-se 9 usuários, dos quais os usuários 1 e 2 são motoristas.

A Tabela 1 mostra o momento que cada usuário i estará disponível para ser buscado por qualquer motorista, assim como a janela de tempo que os motoristas possuem para buscar os passageiros e a capacidade dos veículos.

Tabela 1 – Vetor do tempo inicial dos passageiros

Usuário	1	2	3	4	5	6	7	8	9	10
Tempo inicial	480	480	500	490	485	610	510	550	470	0
Janela de tempo (Motoristas)	20	50								
Capacidade	4	4								

Fonte: O Autor (2018)

Na Tabela 2 o elemento i, j da matriz apresenta o custo temporal (em minutos) do usuário i se locomover até o usuário j .

A solução ótima para o exemplo de nove usuários pode ser verificada na Figura 10. A rota encontrada: 2 - 5 - 1 - 4 - 10, indica que o motorista 2 irá sair de seu local de destino e se locomover até o local onde está o usuário 5, deste para o usuário 1, posteriormente para o usuário 4 e então se locomover até o destino final 10.

O modelo programado em AMPL e submetido ao Solver Gurobi encontra o melhor valor de função objetivo, contudo, ressalta-se que não há garantias da otimalidade da rota para os motoristas, uma vez que o modelo está maximizando a quantidade de usuários como passageiros.

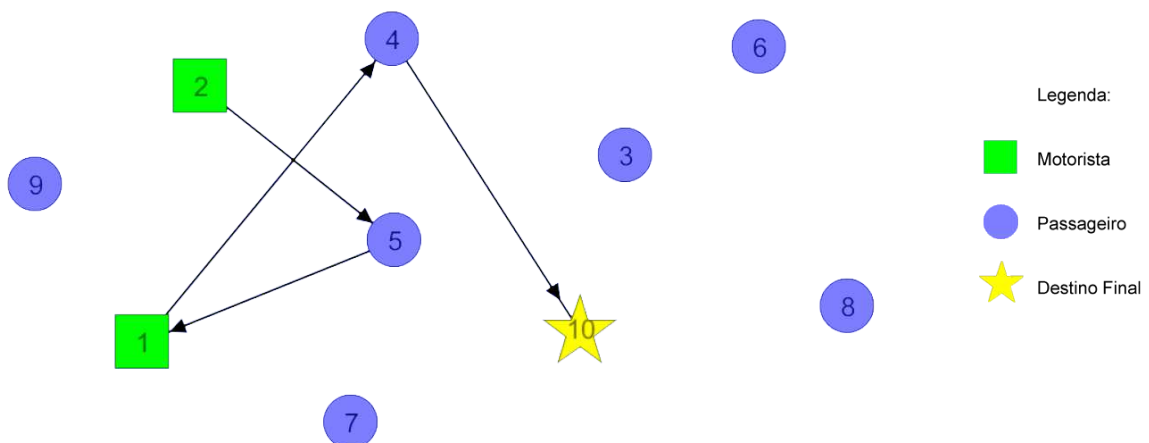
Considerando-se que com o modelo exato não foi possível resolver o problema com dimensões maiores, aplicou-se métodos heurísticos e meta heurísticos, descritos nas próximas seções.

Tabela 2 – Matriz de tempos de viagem do exemplo

Origem/ Destino	1	2	3	4	5	6	7	8	9	10
1	100	15	20	10	5	30	40	10	20	15
2	15	100	5	20	10	15	20	20	25	10
3	20	5	100	15	40	20	30	10	25	15
4	10	20	15	100	10	15	20	30	35	5
5	5	10	40	10	100	25	15	5	40	10
6	30	15	20	15	25	100	15	50	25	20
7	40	20	30	20	15	15	100	10	20	30
8	10	20	10	30	5	50	10	100	15	40
9	20	25	25	35	40	25	20	15	100	15
10	100	100	100	100	100	100	100	100	100	100

Fonte: O Autor (2018)

Figura 10 – Solução para o exemplo proposto



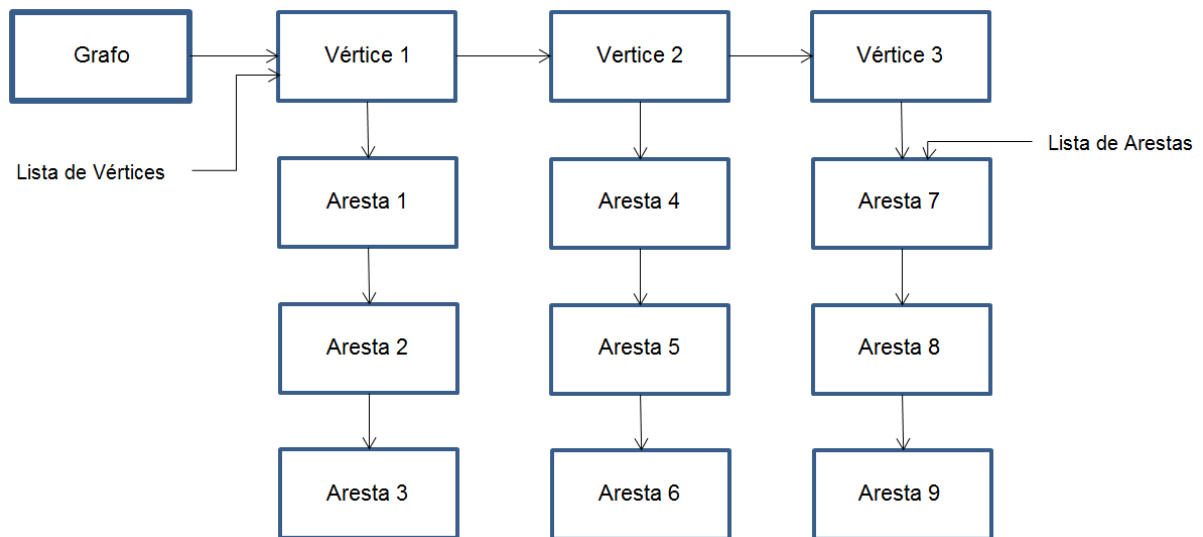
Fonte: O Autor (2018)

4.2 Solução via heurística gulosa

Uma das propostas para solução do problema do carpooling é a solução via heurística gulosa. Entretanto, devido às restrições do problema, foi necessário adaptar o algoritmo apresentado na Seção 2.4.1. Para entendimento do algoritmo é necessário verificar que o mesmo foi elaborado para a estrutura de dados de um grafo denominada lista de adjacência. Na Figura 11 é representado graficamente a interpretação de um grafo para o computador, no qual a estrutura do grafo possui uma lista para os vértices do mesmo. Um vértice, por sua vez, possui uma lista para as arestas que se originam do próprio vértice.

Para representar o problema, inicialmente são inseridos no grafo os vértices para que seja formada a lista de vértices. Cada elemento da lista possui as seguintes

Figura 11 – Representação da estrutura de um grafo



Fonte: O Autor (2018)

informações:

1. Identificador Único: Valor inteiro para identificação de cada usuário no grafo. Variando de 1 a $n + 1$;
2. Capacidade: Valor inteiro que indica capacidade do veículo;
3. Janela de Tempo: Valor do tipo float que informa a janela de tempo do usuário;
4. Status: Parâmetro booleano que indica se o usuários foi visitado;
5. Motorista: Parâmetro booleano que indica se o usuário é um motorista;
6. Tempo inicial: Valor do tipo float que apresenta o momento que o usuário estará disponível.

A partir dos vértices, é possível identificar as arestas do grafo, ou seja, as ligações entres os vértices. As arestas possuem as seguintes informações:

1. Identificador Único: Valor inteiro para identificação da aresta no grafo. Variando de 1 a n^2 ;
2. Vértice de origem: Valor inteiro que informa o vértice inicial da aresta. Variando de 1 a $n + 1$;
3. Vértice de destino: Valor inteiro que apresenta o vértice final da aresta. Variando de 1 a $n + 1$;
4. Custo: Valor do tipo float que indica o tempo de viagem entre o vértice de origem e de destino.

Vale ressaltar que o grafo gerado a partir desta estrutura de dados é completo, ou seja, há arestas entre todos os pares de vértice. Caso, em futuras aplicações, não exista uma conexão entre dois vértices do grafo, esta aresta deve possuir custo muito elevado. Os vértices que representam os passageiros possuem os parâmetros

Capacidade e Janela de Tempo iguais a zero e Status como falso. O vértice de destino, possui as mesmas características dos vértices dos passageiros com Tempo Inicial igual a zero.

Após determinado o grafo inicial completo é aplicada então a heurística gulosa proposta representada na Figura 12.

Figura 12 – Algoritmo guloso proposto para o problema do carpooling

```

1 início
2    $G_{solucao} \leftarrow VerticeDestino(G);$ 
3    $v_{destino} \leftarrow Visitado;$ 
4    $V_{mot} \leftarrow Motoristas(G);$ 
5   enquanto  $ExisteMotoristaNaoVisitado(V_{mot})$  faça
6      $v_{aux} \leftarrow SeleccionaMotorista(G);$ 
7      $F_{vert} \leftarrow v_{aux};$ 
8      $L_{are} \leftarrow \emptyset;$ 
9      $C_{total} \leftarrow 0;$ 
10    contador  $\leftarrow 0;$ 
11     $nvagas \leftarrow Vagas(v_{aux});$ 
12    enquanto contador < nvagas faça
13       $v_{proximo} \leftarrow VizinhoProximo(v_{aux}, V_{mot});$ 
14      se  $SatisfazAsRestricoes(v_{proximo})$  então
15         $C_{total} \leftarrow C_{total} + C_{aux,proximo};$ 
16         $F_{vert} \leftarrow v_{proximo};$ 
17         $L_{are} \leftarrow a_{aux,proximo};$ 
18         $v_{proximo} \leftarrow Visitado;$ 
19         $v_{aux} \leftarrow v_{proximo};$ 
20        contador  $\leftarrow$  contador + 1;
21      fim
22      senão
23         $v_{proximo} \leftarrow Visitado;$ 
24      fim
25      se  $!PossivelInserirPassageiro()$  então
26        Pare;
27      fim
28    fim
29     $L_{are} \leftarrow a_{aux,destino};$ 
30     $Imprima(F_{vert});$ 
31     $G_{aux} \leftarrow F_{vert};$ 
32     $G_{aux} \leftarrow L_{are};$ 
33     $F_{vert} \leftarrow \emptyset;$ 
34     $Desmarca(G, G_{aux});$ 
35  fim
36 fim

```

Para entendimento do pseudo-código, considera-se:

- G o grafo original e completo de todos os usuários do sistema;
- $G_{solucao}$ o grafo resultante da heurística;
- V_{mot} o conjunto de vértices de motoristas do grafo original;
- v_{aux} um vértice auxiliar para funcionamento da heurística;
- F_{vert} uma fila para guardar o caminho percorrido por um motorista;
- L_{are} uma lista que contem as arestas utilizadas pelo motorista;
- C_{total} , o custo acumulado de um determinado caminho;
- $v_{proximo}$ representa um vértice candidato para ser inserido em uma rota;
- $a_{aux,proximo}$ simboliza a aresta com origem v_{aux} e destino $v_{proximo}$;
- $nvagas$ a quantidade de vagas disponíveis no veículo;
- $contador$ a quantidade de vagas utilizadas no veículo.

A função $VerticeDestino(G)$ insere o vértice de destino em $G_{solucao}$. Na linha 4, é inicializado o conjunto com os motoristas de G . A condição de parada do algoritmo se dá quando todos os motoristas são visitados.

Na linha 6 é selecionado um motorista que ainda não foi visitado de V_{mot} . A função $Vagas(v_{aux})$ associa o número de vagas do motorista selecionado a $nvagas$. Em $VizinhoProximo(v_{aux}, V_{mot}, G)$ busca-se encontrar o vizinho mais próximo de v_{aux} que não tenha sido visitado em G e não pertença à V_{mot} . Tal vértice selecionado é atribuído a $v_{proximo}$. É necessário que $v_{proximo}$ atenda às restrições (4.5) - (4.7) do modelo. É possível que o veículo não tenha todos os seus espaços ocupados e ainda assim não seja possível inserir mais passageiros, pois todos os tempos de viagem extrapolam a janela de tempo do motorista. Desta forma, a função $PossivelInserirPassageiro(G)$ verifica essa situação e caso seja negativa, o laço é interrompido.

Quando não for possível inserir passageiros no veículo, seja pela condição estabelecida na linha 14 ou pela linha 25, é necessário inserir uma última aresta entre o último passageiro e o destino final. Feito isso, a fila com os vértices é impressa indicando o caminho realizado pelo motorista e quais usuários serão levados com ele. Então os vértices e arestas percorridos são inseridos em G_{aux} . A função $Desmarca(G, G_{aux})$ desfaz as marcações de visitação dos vértices de G não inseridos em G_{aux} .

4.3 Solução via Meta heurística VND

Outras meta heurísticas selecionadas para a resolução do problema de carpooling foram VND e VNS. Para que fosse possível encontrar as soluções pelos algoritmos apresentados na Seção 2.5.1 seriam necessários definir as vizinhanças.

A codificação utilizada para aplicação destas meta heurísticas foi a mesma utilizada nos procedimentos VND, VNS e *Simulated Annealing*, apresentado na Figura 13.

Figura 13 – Codificação para as meta heurísticas

Motoristas	1º	2º	3º	4º	Ordem dos Passageiros
1	5	6	-1	-1	
2	7	-1	-1	-1	
3	8	9	-1	-1	
4	-1	-1	-1	-1	
⋮	⋮	⋮	⋮	⋮	
n	-1	-1	-1	-1	

Fonte: O Autor (2018)

Para a resolução do problema via meta heurísticas, necessitou-se ter uma codificação, ou seja, uma representação de uma solução diferente da proposta para a heurística gulosa, apresentada na Seção 4.2. Utilizou-se de uma matriz, na qual a primeira coluna representa o motorista do veículo e as demais colunas representam os passageiros transportados pelo mesmo. Desta forma, o número na matriz é o identificador do usuário. Caso na posição não esteja presente o usuário, a posição recebe o valor -1 . A Figura 13 demonstra a codificação utilizada.

A matriz utilizada para codificar a solução do problema possui dimensão $n \times C$, em que n é o número de motoristas totais e C é a maior capacidade de um veículo. Embora o número de colunas seja C , para um motorista i são utilizadas apenas a quantidade de lugares disponíveis no veículo. Desta maneira, na Figura 13 é possível verificar que existe algum motorista com veículo de capacidade 4, uma vez que existem 5 colunas e os motoristas não são contabilizados.

Para o algoritmo VND foram determinadas cinco vizinhanças divididas em dois principais grupos: vizinhanças de locais não ocupados nos veículos e vizinhanças de locais já ocupados no veículos. Para a primeira, as modificações na matriz de solução são efetuadas nos locais disponíveis (iguais a -1). Já a segunda, realiza modificações em locais que já estão ocupados. Como apresentado na Seção 2.5.1, a sequência de vizinhanças é importante e afeta a solução final, sendo assim, foi escolhido alternar as vizinhanças entre os grupos. A Figura 14 apresenta a maneira que as vizinhanças foram formadas.

A sequência escolhida de vizinhanças foi:

1. Inserção de passageiros fora do sistema no fim da rota de um motorista (Fig 14a);
2. Troca de um passageiro i por outro passageiro j (Fig 14b) ;
3. Inserção de um passageiro i ao fim da rota de um motorista (Fig 14c);
4. Troca de um passageiro (Fig 14d) i por um motorista j ;
5. Inserção de um motorista (Fig 14e) i ao fim da rota de outro motorista;

De maneira que para o funcionamento da meta heurística VND, todas as combinações possíveis foram testadas. Isto é, por exemplo, para a Vizinhança V1 tentou-se inserir todos os passageiros fora do sistema em todos os motoristas. Para V2, todos os passageiros já alocados

Figura 14 – Modificações consideradas para a meta heurística VND

a-) Vizinhança V1 - Inserção de todos os usuários ao fim de todas as rotas

Matriz de Solução			⇒	Matriz de Solução Alterada 1			⇒	Matriz de Solução Alterada 2		
1	4	-1		1	4	7		1	4	8
2	-1	-1		2	-1	-1		2	-1	-1
3	5	6		3	5	6		3	5	6

b-) Vizinhança V2 - Troca de todos os passageiros já alocados

Matriz de Solução			⇒	Matriz de Solução Alterada 1			⇒	Matriz de Solução Alterada 2		
1	4	-1		1	5	-1		1	6	-1
2	-1	-1		2	-1	-1		2	-1	-1
3	5	6		3	6	-1		3	5	-1

c-) Vizinhança V3 - Inserção de todos os passageiros já alocados ao fim de todas as rotas

Matriz de Solução			⇒	Matriz de Solução Alterada 1			⇒	Matriz de Solução Alterada 2		
1	4	-1		1	4	5		1	4	6
2	-1	-1		2	-1	-1		2	-1	-1
3	5	6		3	6	-1		3	5	-1

d-) Vizinhança V4 - Troca de passageiros já alocados por todos os motoristas

Matriz de Solução			⇒	Matriz de Solução Alterada 1			⇒	Matriz de Solução Alterada 2		
1	4	-1		1	2	-1		1	3	-1
2	-1	-1		-1	-1	-1		2	-1	-1
3	5	6		3	5	6		-1	-1	-1

e-) Vizinhança V5 - Inserção de todos os motoristas ao fim de todas as rotas

Matriz de Solução			⇒	Matriz de Solução Alterada 1			⇒	Matriz de Solução Alterada 2		
1	4	-1		1	4	2		1	4	3
2	-1	-1		-1	-1	-1		2	-1	-1
3	5	6		3	5	6		-1	-1	-1

Legenda		
		Posição a ser alterada
		Alteração efetuada
		Posições modificadas devido a alteração

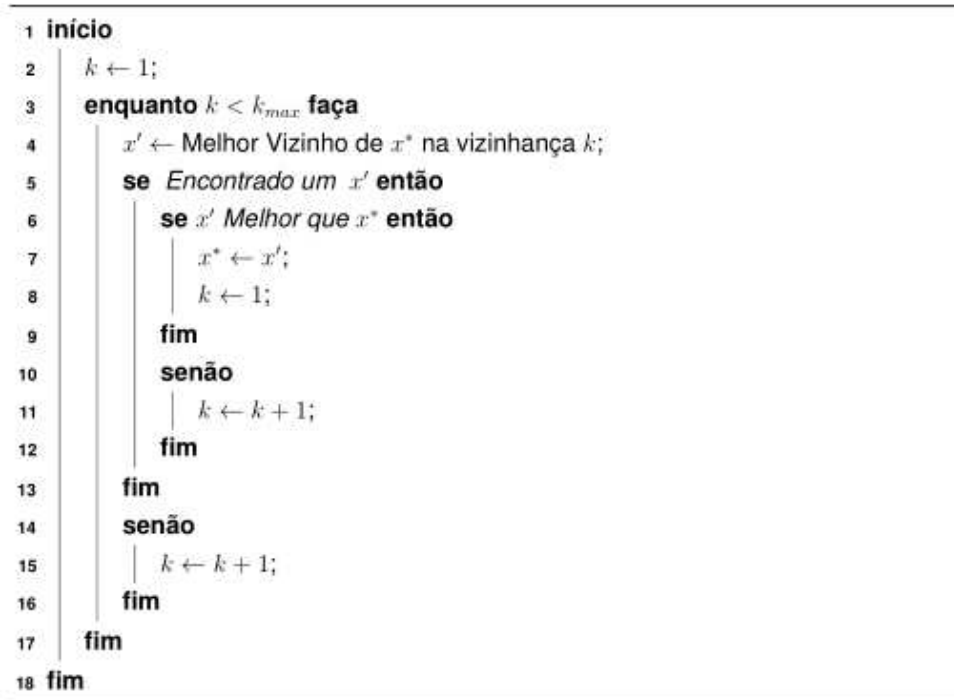
Fonte: O Autor (2018)

foram trocados por todos os demais usuários em que o primeiro usuário é retirado do sistema. Os demais casos também foram testados em todas as combinações possíveis. Vale expor, que para o grupo de vizinhanças de locais não ocupados, caso o local ocupado esteja em um veículo que não possui motorista, o mesmo será alocado para efetuar a troca ou inserção. De modo que, caso o motorista esteja alocado como passageiro, o mesmo deixa de participar da rota onde estaria inserido.

Ressalta-se que o algoritmo proposto na Figura 7 foi modificado. A modificação se deu pelo fato de que as vizinhanças escolhidas, caso a quantidade de usuários seja pequena, por exemplo, é possível que em alguma das vizinhanças escolhidas não exista uma solução válida. Para evitar tal situação, foi inserido uma condição caso a vizinhança não gerasse nenhuma

solução válida. Neste caso, a condição permitia que a vizinhança fosse modificada para dar a oportunidade de encontrar uma solução válida em outra vizinhança definida. Se todas as vizinhanças fossem exploradas e não fosse encontrada uma solução válida, então o algoritmo era encerrado. O algoritmo modificado por ser verificado na Figura 15.

Figura 15 – Algoritmo VND modificado



Fonte: O Autor (2018)

4.4 Solução via Meta heurística VNS

Observa-se que a meta heurística VNS, embora parecida com a VND, pode ter sua busca local de maneira aleatória. De maneira semelhante à meta heurística VND, é necessário determinar vizinhanças para a meta heurística VNS. De modo que as vizinhanças aqui apresentadas tem por objetivo gerar os vizinhos aleatórios que serão utilizados como solução inicial para a busca local do algoritmo. A busca local escolhida para o algoritmo VNS foi a meta heurística VND uma vez que apresentou soluções satisfatórias.

Foram definidas três vizinhanças para gerar os vizinhos aleatórios do procedimento VNS. As vizinhanças podem ser verificadas na Figura 16. A primeira vizinhança foi determinada a partir da troca de passageiros já alocados, são selecionados dois passageiros e trocados entre si. A segunda foi a partir da troca de um passageiro alocado com um motorista, buscando-se tentar reduzir a quantidade de veículos com esta vizinhança. Já a terceira, busca simplesmente inserir um passageiro qualquer em uma posição qualquer, tentando-se aumentar o valor da função objetivo. Tais trocas e inserções foram feitas de maneira aleatória, de maneira que as mesmas considerações feitas nas meta heurísticas VND e SA, referentes aos motoristas, também foram levadas em conta para o procedimento do VNS.

Figura 16 – Vizinhanças consideradas para a meta heurística VNS

a-) Vizinhança V1 - Troca de posição entre passageiros já alocados

Matriz de Solução			⇒	Matriz de Solução Alterada 1		
1	4	-1		1	6	7
2	-1	-1		2	-1	-1
3	5	6		3	5	4

b-) Vizinhança V2 - Troca de passageiro com motorista alocado

Matriz de Solução			⇒	Matriz de Solução Alterada 1		
1	4	-1		-1	-1	-1
2	-1	-1		2	-1	-1
3	5	6		3	1	-1

c-) Vizinhança V3 - Inserção de passageiro aleatório em posição aleatória

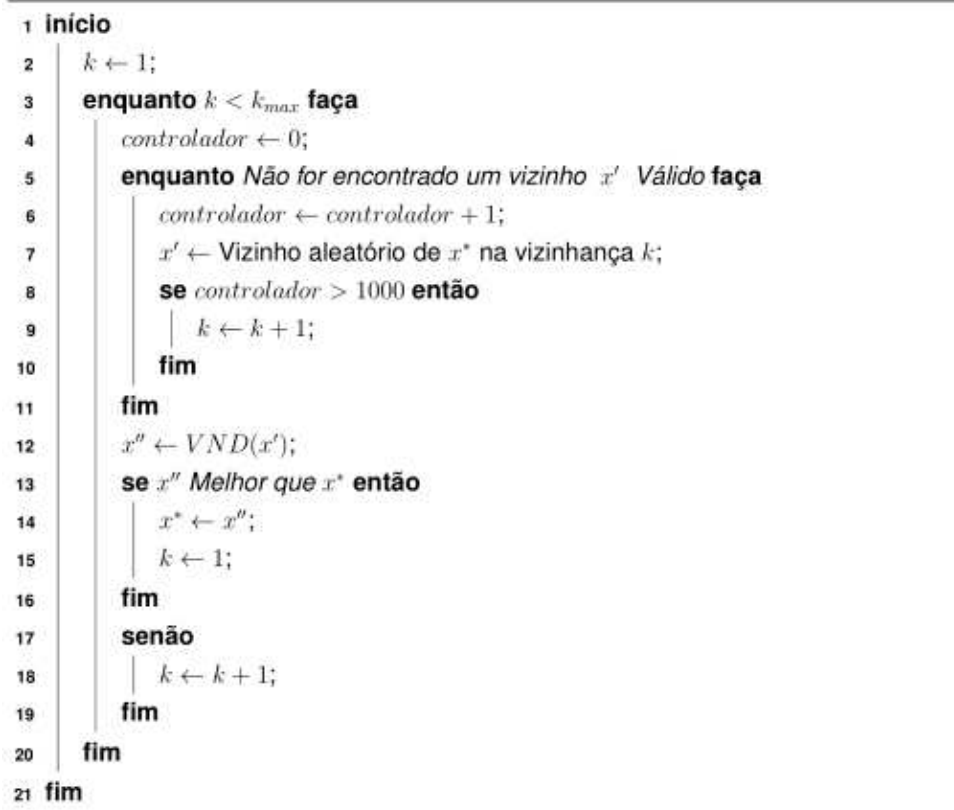
Matriz de Solução			⇒	Matriz de Solução Alterada 1		
1	4	-1		1	4	5
2	-1	-1		2	7	-1
3	5	6		3	6	-1

Legenda		
		Posição a ser alterada
		Alteração efetuada
		Posições modificadas devido a alteração

Fonte: O Autor (2018)

Assim como a meta heurística VND, foi necessário alterar o procedimento VNS da Figura 5. Foi acrescentado um controlador para caso um vizinho não seja encontrado seja permitido avançar à outra vizinhança possibilitando que outro vizinho possa ser encontrado. De maneira que o algoritmo modificado do VNS pode ser visualizado na Figura 17.

Figura 17 – Algoritmo VNS modificado



Fonte: O Autor (2018)

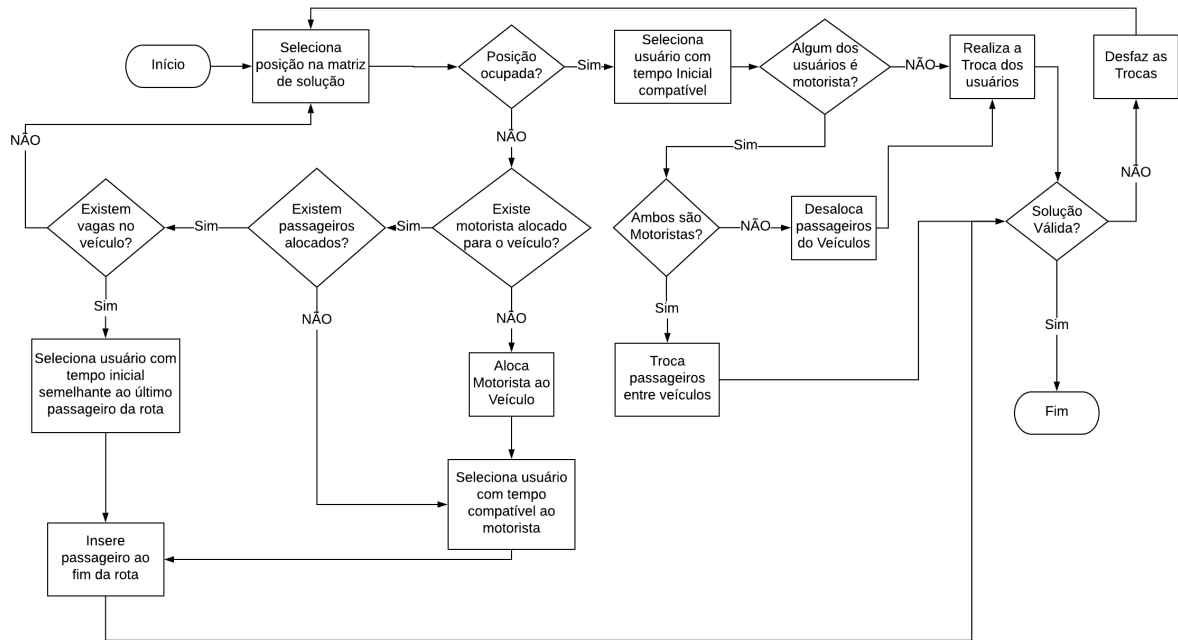
4.5 Solução via Meta Heurística Simulated Annealing

A codificação utilizada para solução via Meta Heurística Simulated Annealing (SA) pode ser verificada na Figura 13. A escolha desta codificação para o problema se deu pela facilidade em realizar as perturbações na solução no processo de busca local do algoritmo SA. Entretanto, devido às restrições (4.6)-(4.7), as perturbações necessitaram ser controladas para que seja possível encontrar somente soluções viáveis.

O fluxograma apresentado na Figura 8, na Seção 2.5.2, indica a sequência lógica para aplicação do método SA. As soluções vizinhas encontradas pelo algoritmo são aleatórias. Entretanto, como mencionado anteriormente, para que fosse possível encontrar soluções viáveis, foi necessário controlar a escolha das perturbações, gerando-se vizinhos aleatoriamente controlados, apresentado na Figura 18.

O procedimento busca encontrar usuários com tempos iniciais compatíveis para que depois sejam realizadas as trocas. O processo de seleção de uma posição na matriz de solução, apresentado na Figura 18, seleciona uma posição aleatória na matriz de solução representada na Figura 13. A posição escolhida pode ou não possuir um passageiro alocado. Desta forma, caso já exista um passageiro alocado, escolhe-se outro usuário, que pode (Caso 1 da Figura 19) ou não (Caso 2 da Figura 19) estar alocado, e efetua-se a troca de lugares entre ambos. Se um dos usuários escolhidos é um motorista, então faz-se necessário desalocar seus passageiros,

Figura 18 – Fluxograma do procedimento para gerar perturbações (Busca local)



Fonte: O Autor (2018)

caso estejam no veículo (Caso 3 da Figura 19). O mesmo ocorre se o lugar não estiver ocupado por um passageiro (Caso 3 - Variação da Figura 20). Se ambos os usuários forem motoristas, então seus passageiros são trocados (Caso 4 da Figura 19).

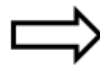
Caso o local escolhido aleatoriamente na matriz de solução não possua nenhum usuário alocado, seu identificador é igual a -1 , como apresentado na Figura 13, então outras questões devem ser analisadas. É necessário verificar se há um motorista alocado para aquele veículo antes de selecionar um usuário para ser realocado neste local. Em caso negativo, aloca-se o motorista e então é selecionado um passageiro (Caso 5 da Figura 20). Caso haja passageiros no veículo, é necessário verificar a existência de locais válidos, isto trata o caso de veículos com capacidade menor que C (Caso 6 da Figura 20). Em caso positivo, é selecionado um usuário com tempo compatível ao último passageiro da rota (Caso 7 da Figura 20). As Figuras 19 e 20 apresentam os possíveis casos que foram discutidos anteriormente, para uma situação hipotética.

A solução inicial considerada para a aplicação do método SA foi a solução obtida por meio da heurística Gulosa apresentada na Seção 4.2. Para determinar a temperatura inicial gerou-se 5 soluções aleatórias e encontrou-se a média na variação de energia. Desta forma, com o auxílio da Equação (2.29), foi possível determinar que uma temperatura inicial adequada. Entretanto, também realizou-se uma simulação com diversas temperaturas. Iniciou-se o procedimento com temperaturas baixas, e posteriormente com temperaturas mais altas. Para cada temperatura, o procedimento SA foi executado 100 vezes. A taxa de resfriamento utilizada foi de 0,98.

Figura 19 – Possíveis trocas com posição escolhida já ocupada

Caso 1

Matriz de Solução			
1	4	5	6
2	7	8	9
3	10	11	12



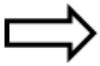
Nova Matriz de Solução			
1	11	5	6
2	7	8	9
3	10	4	12

Posição Escolhida: 1 - 1

Usuário Escolhido: 11

Caso 2

Matriz de Solução			
1	4	5	6
2	7	8	9
3	10	11	12



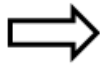
Nova Matriz de Solução			
1	15	5	6
2	7	8	9
3	10	11	12

Posição Escolhida: 1 - 1

Usuário Escolhido: 15

Caso 3

Matriz de Solução			
1	4	5	6
2	7	8	9
3	10	11	12



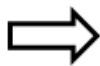
Nova Matriz de Solução			
1	3	5	6
2	7	8	9
-1	-1	-1	-1

Posição Escolhida: 1 - 1

Usuário Escolhido: 3

Caso 4

Matriz de Solução			
1	4	5	6
2	7	8	9
3	10	11	12



Nova Matriz de Solução			
1	10	11	12
2	7	8	9
3	4	5	6

Posição Escolhida: 1 - 0

Usuário Escolhido: 3

Fonte: O Autor (2018)

Figura 20 – Possíveis trocas com posição escolhida não ocupada

Caso 3 - Variação

Matriz de Solução			
1	-1	-1	-1
2	3	8	-1
-1	-1	-1	-1



Matriz de Solução			
1	2	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1

Posição Escolhida: 1 - 1

Usuário Escolhido: 2

Caso 5

Matriz de Solução			
1	4	-1	-1
2	3	8	-1
-1	-1	-1	-1



Nova Matriz de Solução			
1	4	-1	-1
2	8	-1	-1
3	10	-1	-1

Posição Escolhida: 3 - 2

Usuário Escolhido: 10

Caso 6

Matriz de Solução			
1	4	-1	-1
2	3	8	-1
-1	-1	-1	-1



Nova Matriz de Solução			
1	4	-1	-1
2	3	8	-1
-1	-1	-1	-1

Posição Escolhida: 1 - 3

Usuário Escolhido: 10

Caso 7

Matriz de Solução			
1	4	-1	-1
2	3	8	-1
-1	-1	-1	-1



Nova Matriz de Solução			
1	4	-1	-1
2	3	8	10
-1	-1	-1	-1

Posição Escolhida: 2 - 3

Usuário Escolhido: 10

Fonte: O Autor (2018)

5 ANÁLISE DE RESULTADOS

Neste capítulo serão apresentados os resultados e as análises quanto ao desempenho computacional dos métodos, qualidade da solução obtida (quantidade de passageiros inseridos no sistema), taxa de ocupação dos veículos e distâncias percorridas dos métodos de otimização utilizados.

Na pesquisa realizada foram coletadas informações de 72 indivíduos, após o tratamento de dados, foram dispensados seis usuários devido a falta de informações, totalizando 66 possíveis usuários do sistema de carpooling. O Solver Gurobi, disponível remotamente na plataforma NEOS Server em <https://neos-server.org/neos/>, não obteve uma solução ao processar o MMCP para os 66 usuários aptos, alcançando o limite de memória máxima permitida por submissão. A mesma abordagem foi inserida ao mesmo solver em um computador, apresentado na Seção 4, com a versão completa do software AMPL. Com o computador utilizado também não foi possível obter a solução ótima.

Desta maneira, reduziu-se a quantidade de possíveis passageiros no sistema para um valor possível de ser processado no Solver Gurobi. Ao reduzir o problema para 40 usuários, o Solver Gurobi da plataforma NEOS Server conseguiu encontrar a solução ótima. Desta forma, no presente trabalho, para o MMCP, serão consideradas apenas as soluções obtidas por meio da plataforma NEOS Server.

Tabela 3 – Cenários considerados

Cenário	Número de Usuários	Número de Motoristas
Cenário de Validação	9	2
Cenário 1 (C1)	40	16
Cenário 2 (C2)	66	16
Cenário 3 (C3)	20	16

Fonte: O Autor (2018)

Nas seções seguintes apresenta-se a aplicação do MMCP para os cenários apresentados na Tabela 3. Determinou-se o cenário 1 para realizar a comparação de resultados entre a solução do modelo exato e dos modelos aproximados. O cenário 2 considera o problema com todos os possíveis usuários levantados. Para realizar uma análise referente as restrições acrescidas ao MMCP foi determinado o cenário 3.

5.1 Rotas dos Motoristas

Nesta seção são apresentados os resultados referentes as rotas obtidas dos motoristas por cada método de otimização em cada cenário.

5.1.1 Cenário 1

Aplicou-se o modelo para o problema do carpooling com as informações de 40 usuários (Cenário 1). As rotas determinadas como solução, são apresentadas na Tabela 4, na qual a linha um indica que o usuário 1 é um motorista e transportará os usuários 14, 33, 5 e 12, nesta ordem, até o destino 41. Já o usuário 4 transportará os usuários 26, 21 e 11 até o destino final 41 e assim por diante para os demais motoristas.

Tabela 4 – Solução - Carpooling 40 usuários

Motorista	Rota				
1	14	33	5	12	41
4	26	21	11	41	
7	25	19	41		
9	27	32	2	41	
15	29	39	34	23	41
16	20	24	37	41	
Motoristas que viajam sozinhos	3	6	8	10	13
Número de passageiros não alocados	10				

Fonte: O Autor (2018)

Pode-se notar que na solução ótima determinada pelo método exato nem todos os usuários utilizam o sistema do carpooling. Embora a quantidade de lugares nos veículos (61 lugares) supere a quantidade de usuários total (40 usuários), devido as restrições de janela de tempo consideradas no MMCP apenas 19 pessoas foram alocadas como passageiros. Ainda que o número de passageiros tenha sido inferior a 50% dos passageiros totais, o modelo ainda mostrou uma redução de cinco veículos nas vias, uma vez que os motoristas: 2, 5, 11, 12 e 14 são considerados passageiros.

Aplicando-se o algoritmo guloso para o Cenário 1, tem-se que a solução obtida pode ser verificada na Tabela 5. A função objetivo alcançou o valor de 14 passageiros dispostos em 7 veículos.

Tabela 5 – Rotas dos motoristas pelo algoritmo guloso para 40 usuários

Motorista	Rota				
2	27	41			
4	21	41			
5	33	37	26	41	
7	25	19	41		
9	32	41			
15	29	28	39	34	41
16	20	24	41		
Motoristas que Viajam Sozinhos	1	3	6	8	10
	11	12	13	14	
Número de Passageiros Não Alocados	10				

Fonte: O Autor (2018)

Ao aplicar algoritmo VND para o Cenário 1, por se tratar de uma heurística determinística, não é necessário executar o procedimento diversas vezes, pois as soluções encontradas, para uma dada solução inicial, serão as mesmas. O método alcançou valor de função objetivo 16, os passageiros foram alocados em oito veículos. A alocação dos passageiros pode ser verificada na Tabela 6.

Tabela 6 – Rota dos motoristas pelo algoritmo VND para 40 usuários

Motorista	Rota				
1	14	41			
2	27	41			
4	21	11	41		
5	33	37	26	41	
7	25	19	41		
9	32	41			
15	28	39	29	34	41
16	20	24	41		
Motoristas que Viajam Sozinhos	3	6	8	10	12
	13				
Número de Passageiros Não Alocados	10				

Fonte: O Autor (2018)

Aplicou-se o algoritmo VND para uma solução inicial vazia para verificar o comportamento da meta heurística. De modo que para a solução inicial vazia a meta heurística obteve o mesmo resultado que a solução inicial gulosa. A diferença foi no posicionamento do usuário 24, o qual não viajou com o motorista 5, mas com o 16, sendo incluído na rota após o usuário 20. Sendo assim, a meta heurística VND, aplicada a este problema, com a codificação e estrutura de vizinhanças apresentadas, apresenta-se eficiente para ser utilizada como uma

meta heurística construtiva.

Tabela 7 – Rotas dos motoristas VNS para 40 usuários

Motorista	Rota				
4	11	41			
5	33	37	26	21	41
7	25	19	41		
9	32	27	2	41	
14	1	41			
15	28	39	29	34	41
16	20	24	41		
Motoristas que viajam sozinhos	3	6	8	10	12
	13				
Número de passageiros não alocados	10				

Fonte: O Autor (2018)

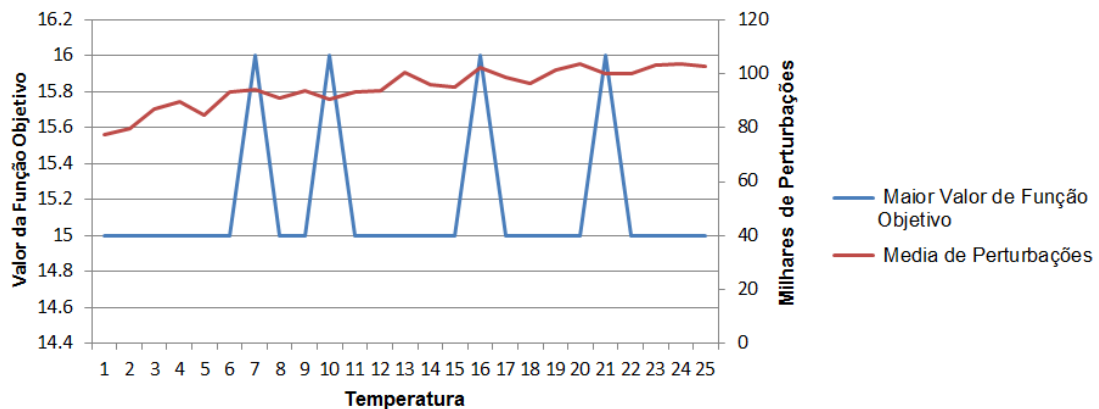
A roteirização dos veículos por meio do método VNS pode ser verificada na Tabela 7. Devido a aleatoriedade do algoritmo, foram realizadas 200 repetições do procedimento com a solução inicial do algoritmo guloso. O maior valor de função objetivo encontrado nestas 200 repetições foi de 17 passageiros alocados em sete veículos.

Tentou-se também a abordagem com a solução inicial vazia e os resultados obtidos foram semelhantes aos obtidos por meio da solução inicial gulosa, porém com um tempo de processamento muito superior.

Desta maneira, pode-se notar que a meta heurística VNS, mesmo utilizando o procedimento VND como busca local do algoritmo, alcançou uma solução melhor que o algoritmo VND isoladamente, devido ao aumento na quantidade de passageiros transportados. Tal fato, pode ter ocorrido por conta das mudanças nas vizinhanças evitando que a solução fique em um mínimo local. Porém, outros procedimentos de busca local podem gerar outras soluções melhores que a apresentada no presente trabalho. Entretanto, para as presentes aplicações, não serão apresentados outros métodos de busca local, devido o presente procedimento ter mostrado respostas satisfatórias.

O procedimento SA possui como um de seus parâmetros a temperatura inicial. Para determinar tal temperatura inicial gerou-se as 5 soluções aleatórias e encontrou-se a média na variação de energia de 2,3 graus. Desta forma, foi possível determinar que a temperatura inicial adequada seria de aproximadamente 10 graus. As simulações foram executadas considerando $1 \leq T_0 \leq 25$. Não foram executados mais procedimentos ou com outras temperaturas devido à insuficiência de memória do computador utilizado. Pode-se observar na Figura 21 que os melhores valores da função objetivo foram encontrados nas temperaturas 7, 10, 16 e 21 graus. Sendo assim, estas temperaturas foram selecionadas, para 200 repetições do algoritmo SA. Pode-se verificar também na Figura 21 a quantidade média de perturbações por temperatura. Um dos motivos para a variação na quantidade de perturbações se dá pela taxa de resfriamento escolhida e pela aleatoriedade do processo.

Figura 21 – Gráfico para determinação da temperatura inicial



Fonte: O Autor (2018)

Após aplicar o algoritmo para as temperaturas selecionadas a função objetivo não demonstrou melhorias. Desta maneira, a solução obtida com o SA para o presente problema foi obtida considerando-se a solução inicial determinada pela heurística gulosa, com a temperatura inicial de 10 graus e taxa de resfriamento de 0,98.

Para o problema de carpooling do cenário 1, o procedimento SA foi repetido 200 vezes, com uma média de 95.271 perturbações. A função objetivo teve seu maior valor de 16 passageiros alocados em sete veículos.

Tabela 8 – Resultado da aplicação do algoritmo SA para 40 usuários

Motorista	Rota				
	1	14	41		
4	21	41			
5	33	37	26	41	
7	25	19	41		
9	32	27	2	41	
15	29	28	39	34	41
16	20	24	41		
Motoristas que Viajam Sozinhos	3	6	8	10	11
	12	13			
Número de Passageiros Não Alocados	10				

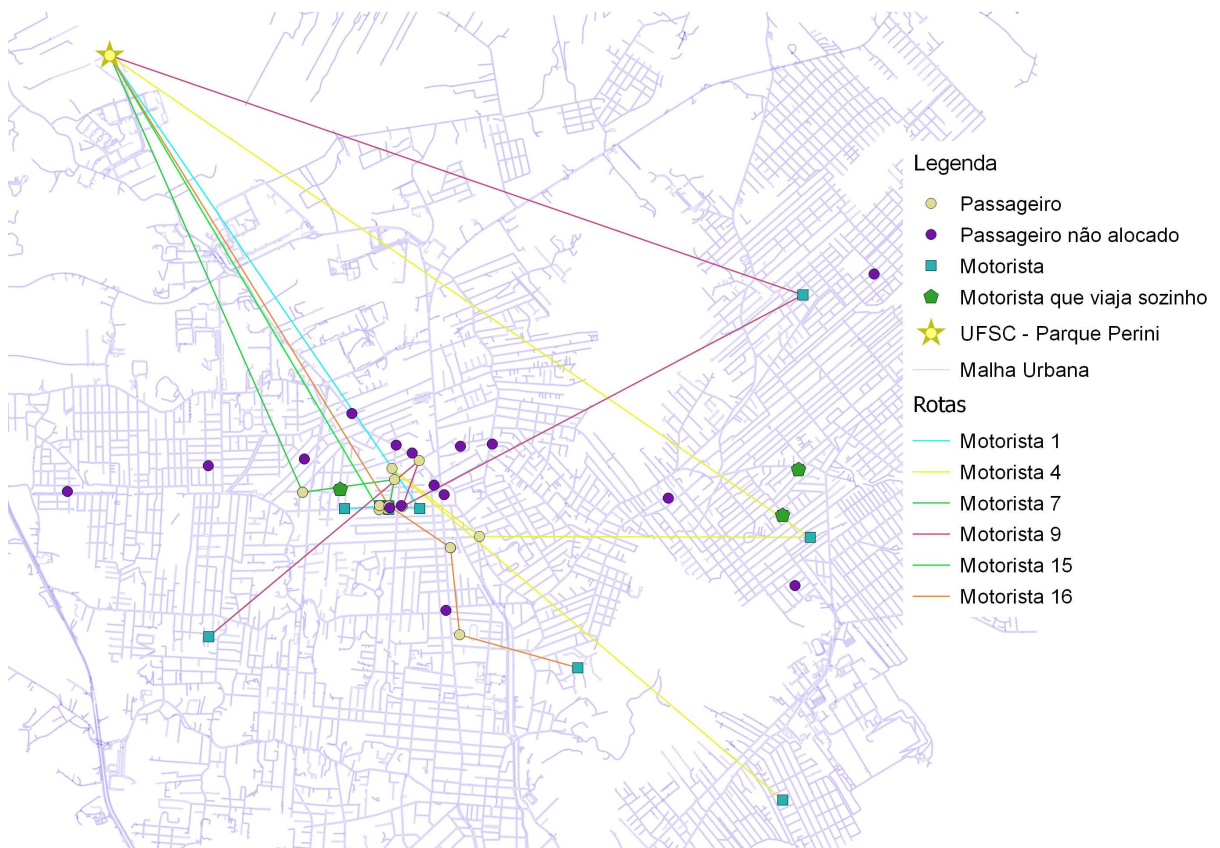
Fonte: O Autor (2018)

A solução obtida pelo procedimento SA mostrou-se muito semelhante à obtida por meio do algoritmo guloso, como pode ser observado na Tabela 8, pois a qualidade da solução inicial do procedimento SA impacta diretamente na qualidade da solução final obtida. Desta maneira, tentou-se utilizar o procedimento SA como uma heurística construtiva de forma que a cada iteração se constrói uma etapa da solução final. Para tal, foi utilizada uma matriz de solução inicial completamente vazia, com todos seus elementos sendo iguais a -1 , uma vez que a implementação permite que os motoristas sejam alocados à medida que são inseridos

novos passageiros. A temperatura inicial escolhida foi 10 graus, entretanto a solução inicial vazia não se mostrou eficaz. Repetiu-se o procedimento SA 200 vezes e foi escolhida o melhor valor de função objetivo, o qual foi seis usuários alocados como passageiros em quatro veículos. Desta forma, utilizar o procedimento SA como heurística construtiva para o presente trabalho não se mostrou como alternativa viável.

A melhor rota obtida no Cenário 1 pode ser verificada na Figura 22 que representa a solução obtida por meio do método exato de otimização.

Figura 22 – Resultados da aplicação do modelo exato para o cenário 1



Fonte: O Autor (2018)

5.1.2 Cenário 2

Realizou-se a aplicação dos métodos de otimização para o cenário 2, o qual considerava todos os possíveis 66 usuários. Entretanto, para este cenário não foi possível aplicar o MMCP, uma vez que o Solver Gurobi, na plataforma NEOS Server, não conseguiu encontrar uma solução válida com a quantidade de memória disponível para cada aplicação.

Aplicou-se então o algoritmo guloso para a base de dados completa com todos os 66 usuários e a solução é apresentada na Tabela 9. O valor da função objetivo encontrado foi de 30 passageiros em 12 veículos. Observa-se que existe um melhor aproveitamento dos veículos, uma vez que apenas quatro veículos não possuem passageiros alocados. Desta forma, embora pela heurística não há uma grande redução na quantidade de motoristas, existe a melhora no

aproveitamento dos espaços nos veículos.

Tabela 9 – Rotas dos motoristas pelo algoritmo guloso para 66 usuários

Motorista	Rota				
1	53	33	37	42	67
2	27	67			
3	51	67			
4	21	50	67		
5	47	26	67		
6	56	67			
7	25	19	67		
9	60	32	5	63	67
12	66	65	46	54	67
14	43	67			
15	29	28	39	34	67
16	20	24	58	59	67
Motoristas que Viajam Sozinhos	8	10	11	13	
Número de Passageiros Não Alocados	20				

Fonte: O Autor (2018)

Para o problema de carpooling aplicado ao cenário 2, o resultado da meta heurística VND é mostrado na Tabela 10. Embora o valor tenha sido o mesmo que aqueles obtidos nas pelo método guloso a roteirização dos motoristas foi diferente.

Tabela 10 – Rota dos motoristas VND para 66 usuários

Motorista	Rota				
1	53	33	37	42	67
2	27	67			
3	51	67			
4	21	50	67		
5	47	26	67		
6	56	67			
7	25	19	67		
9	55	32	60	63	67
12	61	65	46	54	67
14	43	67			
15	28	39	29	34	67
16	20	24	58	59	67
Motoristas que viajam sozinhos	8	10	11	13	
Número de passageiros não alocados	20				

Fonte: O Autor (2018)

Pode-se notar que as soluções obtidas por meio do algoritmo VND são semelhantes às aquelas obtidas pelo algoritmo guloso. Indicando, novamente, que a solução inicial fornecida para o procedimento VND, impacta diretamente na solução final. Desta maneira, uma vez mais, foi utilizada uma matriz de solução inicial nula para o algoritmo VND. O valor da função objetivo permaneceu em 30 passageiros alocados em 12 veículos, porém algumas rotas dos motoristas foram alteradas. Portanto, para este caso específico, foi analisado alguns fatores adicionais além do valor de função objetivo de quantidade de veículos utilizados.

A quantidade de passageiros transportados em ambas as soluções foi a mesma. Sendo assim foi analisado também a distância percorrida pelos motoristas. Na solução obtida pela solução inicial do algoritmo guloso, os motoristas percorreram um total de 154,74 quilômetros, soma de todas as distâncias percorridas, percurso que tomou um total de 232,11 minutos, soma de todos os tempos gastos. Já na solução obtida pela matriz de solução nula, os motoristas percorreram um total de 158,30 quilômetros, tomando um tempo de 237,47 minutos. Desta maneira, embora ambas soluções tenham tido o mesmo valor da função objetivo, a solução obtida em conjunto com a heurística gulosa se mostrou um pouco superior nos quesitos aqui apresentados.

Tabela 11 – Rotas dos motoristas VNS para 66 usuários

Motorista	Rota				
1	43	67			
2	27	67			
3	51	67			
4	21	42	50	67	
5	61	47	46	58	67
6	56	67			
7	25	19	67		
9	55	32	60	63	67
12	33	37	26	67	
14	53	66	65	54	67
15	28	39	29	34	67
16	41	20	24	59	67
Motoristas que Viajam Sozinhos	8	10	11	13	
Número de Passageiros Não Alocados	18				

Fonte: O Autor (2018)

Aplicou-se o algoritmo VNS para os dados do Cenário 2 com a solução inicial gulosa obtendo-se um resultado de 30 passageiros alocados em 12 veículos. Entretanto, também foi realizada a abordagem com a matriz inicial nula e o resultado desta abordagem se mostrou superior a solução com a matriz gulosa. Em ambas as abordagens foram realizadas 200 iterações do algoritmo VNS e selecionada a melhor solução encontrada, a solução obtida por meio da matriz inicial vazia apresentou o valor da função objetivo de 32 passageiros em 12 veículos, de maneira que a Tabela 11 apresenta as rotas dos motoristas. Tal resultado, mostra a

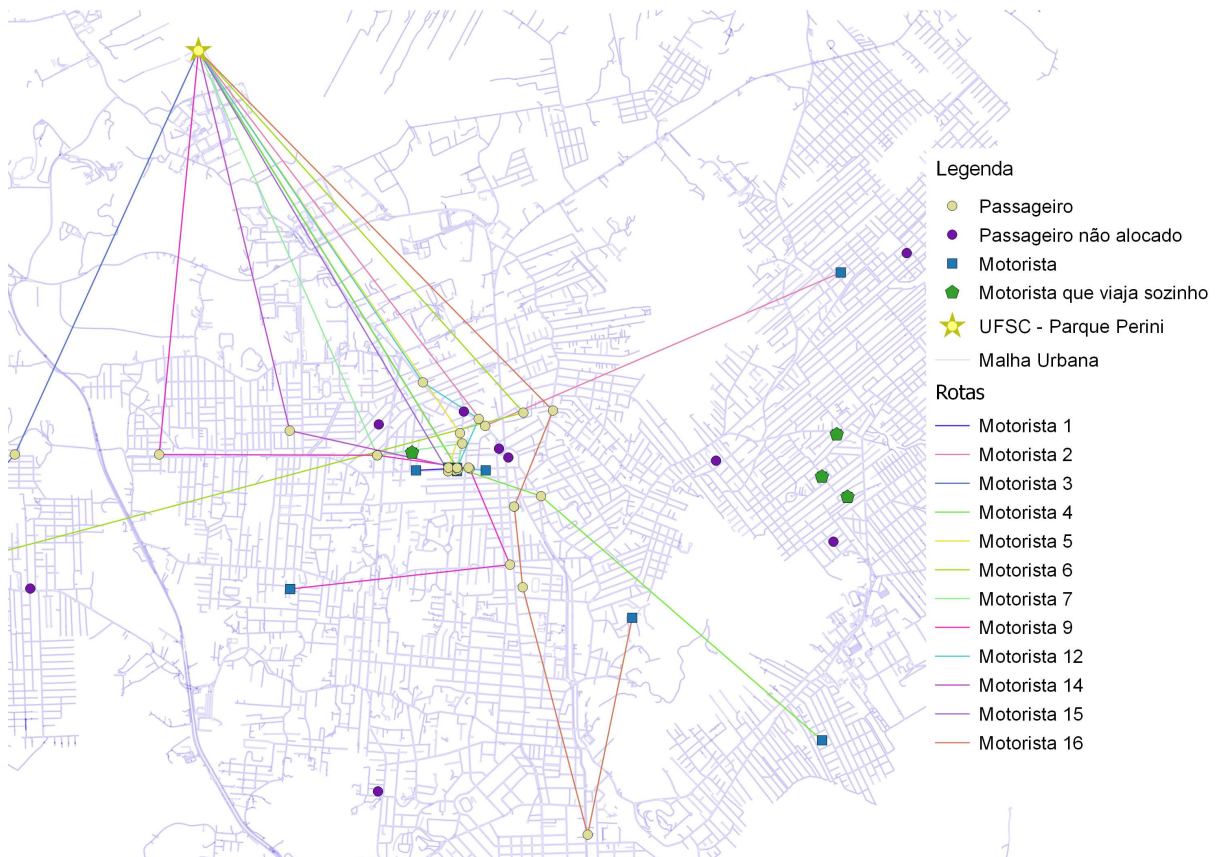
necessidade de, além de outras formas de busca local, novas abordagens para a solução inicial do algoritmo VNS.

A meta heurística SA também foi aplicada ao Cenário 2, realizando os mesmos procedimentos, em relação à temperatura inicial e quantidade de reproduções, que foram descritos anteriormente para o Cenário 1. Ao aplicar a meta heurística com a solução inicial da heurística gulosa, a função objetivo alcançou valor de 30 passageiros alocados em 12 veículos. A temperatura inicial escolhida foi de 34 graus. A solução inicial não teve melhorias significativas dadas as perturbações utilizadas. Portanto, a solução obtida por meio da heurística gulosa e da meta heurística SA são as mesmas para o problema com 66 usuários, as soluções são apresentadas na Tabela 9.

Tentou-se novamente realizar o algoritmo SA com a matriz de solução inicial vazia. A solução obtida teve um valor máximo de 9 passageiros nas 200 iterações. Tal resultado, reforça que a meta heurística SA, da maneira apresentada no presente trabalho, não é viável quando utilizada de maneira construtiva.

A melhor solução do Cenário 2 pode ser verificada na Figura 23, a qual consiste na solução obtida por meio da meta heurística VNS.

Figura 23 – Resultado da aplicação do algoritmo VNS para o cenário 2



5.2 Tempo de processamento

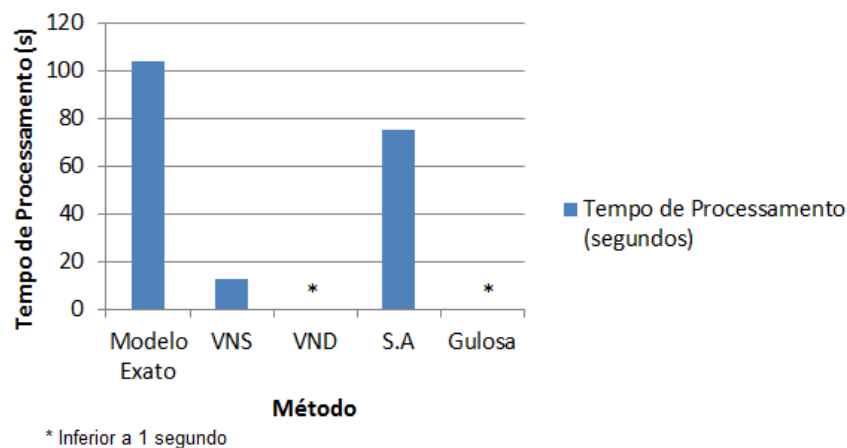
Os resultados referentes ao tempo de processamento dos métodos propostos para os cenários estabelecidos na Seção 4 são apresentados a seguir.

5.2.1 Cenário 1

Para o cenário 1, no qual haviam 40 usuários, o tempo de processamento dos métodos de solução pode ser verificado na Figura 24.

O valor apresentado pelo método VNS apresenta o tempo de processamento com a solução inicial gulosa, o qual foi de 13,032 segundos. O método VNS com a solução inicial vazia teve seu tempo de processamento na ordem dos 350 segundos (média de 340,29 segundos).

Figura 24 – Tempo de processamento dos métodos para o cenário 1



Fonte: O Autor (2018)

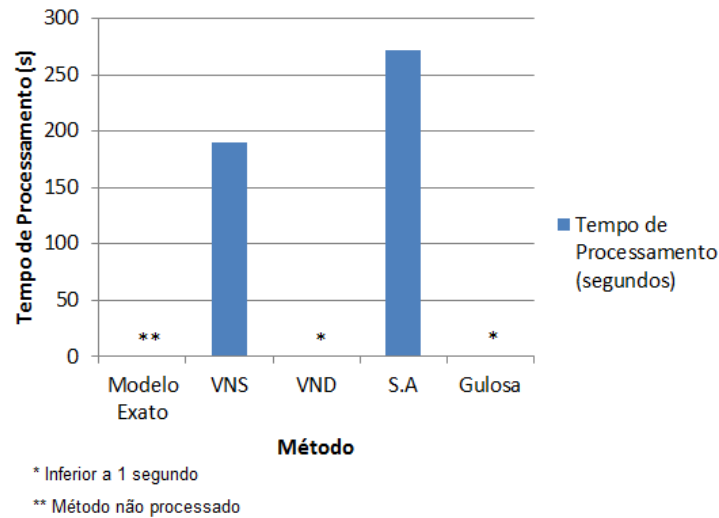
Pode-se notar pela Figura 24 que a heurística gulosa e a meta heurística VND foram os métodos que possuíram os melhores tempos de processamento. O modelo exato apresentou-se como o método mais lento dentre os analisados, seguido pela meta heurística SA. A meta heurística VNS apresentou um tempo de processamento satisfatório para o cenário, uma vez que não se mostrou tão elevado quanto ao modelo exato e a meta heurística SA.

5.2.2 Cenário 2

Os tempos de processamento do cenário 2, com 66 usuários, podem ser verificados na Figura 25.

Idêntico ao cenário 1, a heurística gulosa e a meta heurística VND tiveram os melhores tempos de processamento. As meta heurística VNS e SA possuíram um tempo de processamento superior a 150 segundos. Tal fato reforça a complexidade do problema do carpooling, uma vez que a tendência é que para entradas maiores o tempo computacional aumente ainda mais. O MMCP não pode ser processado para o cenário 2.

Figura 25 – Tempo de processamento dos métodos para o cenário 2



Fonte: O Autor (2018)

5.2.3 Cenário 3

Para mostrar que a restrição de horários de saída, para os dados utilizados, é a mais restritiva no MMCP, simulou-se um cenário no qual todos os usuários saem no mesmo horário, eliminando-se as restrições (4.6) e (4.7). Entretanto, o modelo aplicado para o problema com 40 usuários não obteve sucesso ao ser processado sem as restrições, uma vez que a quantidade de memória máxima por submissão do Solver Gurobi foi alcançada novamente. Desta forma, tentou-se reduzir a quantidade de usuários para 30 e, uma vez mais, não foi possível chegar a uma solução. Ao reduzir o problema para 20 usuários, foi possível encontrar uma solução ótima por meio do Solver Gurobi.

Sendo assim, para poder comparar resultados, foi simulado o Cenário 3 de duas maneiras. A primeira considerando-se que os 20 usuários estavam disponíveis no mesmo horário, sem as restrições (4.6) e (4.7). A segunda na qual eram consideradas as restrições do horário de saída e tempo de espera máximo. Os parâmetros escolhidos para avaliação dos resultados do MMCP foram: tempo de processamento no solver Gurobi, quantidade de variáveis no problema ajustado e valor da função objetivo. O tempo de processamento no solver Gurobi se refere apenas ao tempo utilizado pelo solver para encontrar a solução ótima. A linguagem de programação matemática AMPL aplica por padrão uma pré-solução dos problemas de otimização antes de enviá-los ao solver. Desta forma, o método busca reduzir a quantidade de restrições e variáveis para encontrar uma solução de maneira mais rápida. Tais simplificações são desfeitas quando o problema encontra um solução, verificando se todas as restrições do problema original foram satisfeitas. O comparativo entre cenários pode ser verificado na Tabela 12.

Observa-se na Tabela 12, que se houvesse a possibilidade de todos os usuários estarem disponíveis ao mesmo tempo, para o cenário 3, a função objetivo aumentaria de

Tabela 12 – Complexidade do cenário 3 com e sem as restrições de horário de saída - 20 usuários

	Tempo de Processamento (s)	Número de Variáveis do Problema Ajustado	Valor da Função Objetivo
Cenário 3 Sem Restrições	164.24	5054	15
Cenário 3 Com Restrições	0.015	206	5
Aumento (Razão)	10511	25	3

Fonte: O Autor(2018)

cinco para 15 passageiros. Entretanto, o tempo de processamento também aumentaria consideravelmente.

Também aplicou-se o algoritmo guloso para o problema de 20 usuários, sem as restrições (4.6) e (4.7), referentes a disponibilidade dos usuários e tempo de espera máximo, respectivamente. O algoritmo guloso, para os dois cenários (com e sem as restrições) tomou um tempo de processamento de 0,02 segundos.

O algoritmo VND, para o Cenário 3 com as restrições teve um tempo de processamento de 0,042 segundos. Já o Cenário 3 sem as restrições teve o tempo de processamento um pouco superior de 0,076 segundos.

Aplicou-se o algoritmo VNS para o cenário 3 com e sem as restrições referentes ao tempo inicial dos usuários e tempo máximo de espera. Para o cenário com as restrições o tempo de processamento foi de 5,960 segundos. Já para o cenário 3 sem as restrições, o tempo de processamento de 9,791 segundos da VNS foi superior ao tempo de processamento da VND.

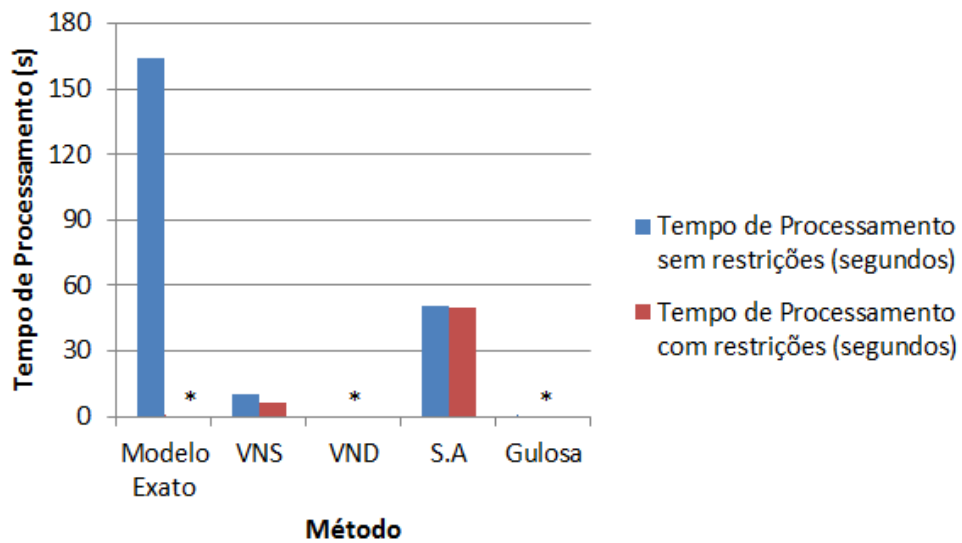
O tempo de processamento da meta heurística SA para o Cenário 3 com e sem as restrições foi semelhante, na ordem de 50 segundos.

Os informações referentes ao tempo de processamento dos métodos para o cenário de 20 usuários (Cenário 3), com e sem as restrições de disponibilidade e tempo de espera máximo, podem ser visualizadas na Figura 26.

Pode-se verificar que o desempenho do MMCP para o cenário 3 com restrições, da heurística gulosa e da meta heurística VND se mostraram muito semelhantes. Porém quando analisado o MMCP para o cenário 3 sem as restrições o mesmo apresentou um tempo de processamento elevado. As meta heurísticas SA e VNS apresentaram melhores resultados quando comparados com os cenários anteriores.

A Figura 27 apresenta a evolução do tempo de processamento dos métodos de otimização propostos para os cenários propostos. Pode-se notar que o tempo de processamento das meta heurísticas VNS e SA tendem a aumentar de maneira muito rápida quando acrescenta-se novos usuários no sistema, indicando a necessidade de novas estratégias para problemas

Figura 26 – Tempo de processamento dos métodos para o cenário 3

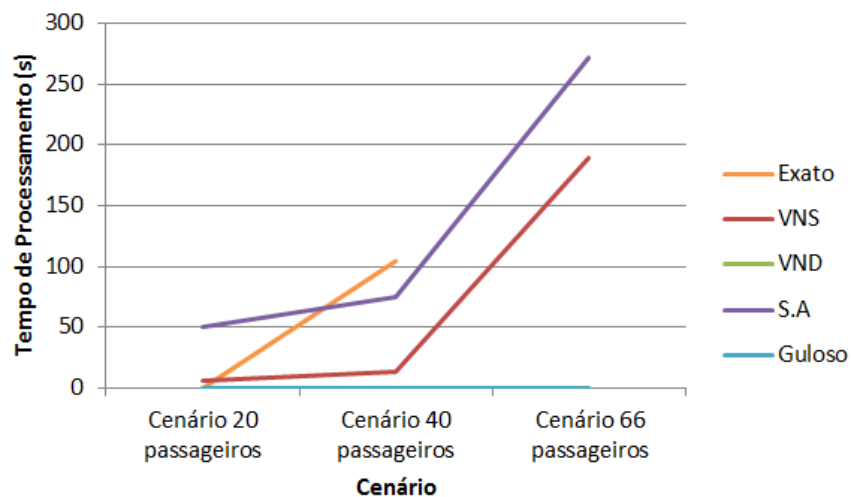


* Inferior a 1 segundo

Fonte: O Autor (2018)

com uma maior quantidade de possíveis usuários.

Figura 27 – Evolução do tempo de processamento nos cenários propostos



Fonte: O Autor (2018)

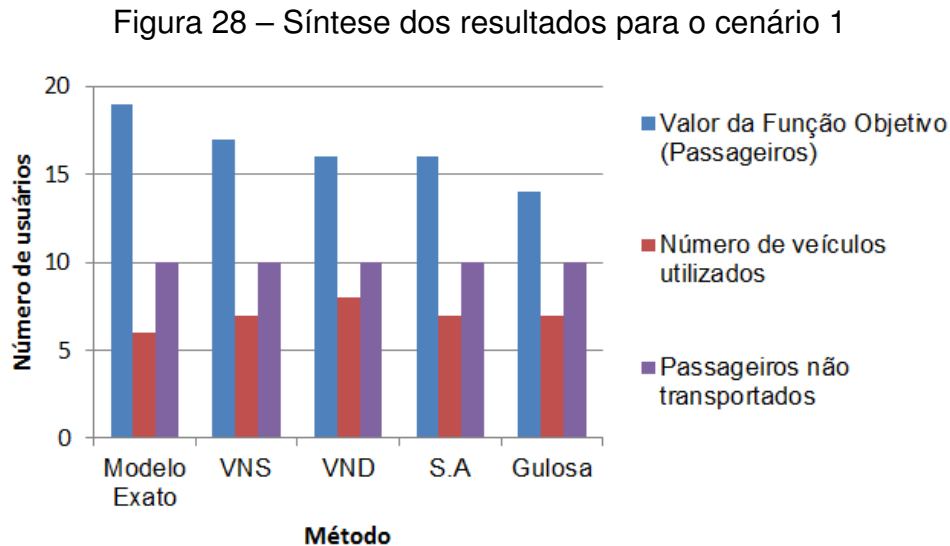
5.3 Quantidade de Passageiros Transportados

Não é possível analisar o tempo de processamento como um indicador isolado uma vez que a quantidade de passageiros transportados no sistema carpooling foi o principal objetivo do MMCP. Uma solução pode possuir seu tempo de processamento baixo, mas pode não haver qualidade na resposta obtida. Ainda, pode que a solução possua qualidade, mas o tempo de processamento a torne inviável. Desta maneira, a seguir são apresentados os resultados

referentes a qualidade da solução dos diferentes cenários.

5.3.1 Cenário 1

A Figura 28 apresenta uma síntese dos tempos de processamento e maiores valores da função objetivo alcançados para o cenário 1.



Fonte: O Autor (2018)

Pode-se notar na Figura 28 que o modelo exato, como esperado, teve o maior valor da função objetivo. O tempo de processamento do requerido mostrou-se satisfatório dada a complexidade do problema, entretanto não foi possível aplicar o problema para mais de 40 usuários no sistema, demonstrando assim a limitação da aplicação do modelo.

A meta heurística VNS foi, entre os métodos aproximados, o que obteve o melhor resultado e mais próximo do resultado otimizado. O tempo de processamento requerido foi considerado o melhor devido a solução obtida neste tempo. Tal fato pode ser devido às vizinhanças escolhidas, uma vez que diferentes vizinhanças podem determinar vizinhos diferentes, impactando no tempo de busca.

Um fato observado ao se analisar os resultados foi que todos os métodos deixaram de alocar exatamente 10 passageiros. Os passageiros: 17, 18, 22, 30, 31, 35, 36, 38 e 40 não foram alocados por nenhum método de otimização. O passageiro 23 foi alocado apenas na solução do modelo exato e não nos demais métodos. Já o passageiro 28 foi alocado em todos os métodos aproximados e não no método exato. Como abordado anteriormente, para os passageiros não alocados em nenhum método de otimização, não foi possível inserir os passageiros na rota de nenhum motorista devido ao horário de saída não ser compatível, ou porque os mesmos possuíam tempos de viagem muito elevados.

As meta heurística VND e Simulated Annealing, no quesito de quantidade de passageiros, se mostraram satisfatórias, uma vez que ambas encontraram soluções viáveis e não distantes da melhor solução dos métodos aproximados. A meta heurística VND se mostrou superior a SA com relação ao tempo de processamento quase 800 vezes menor. Porém, vale

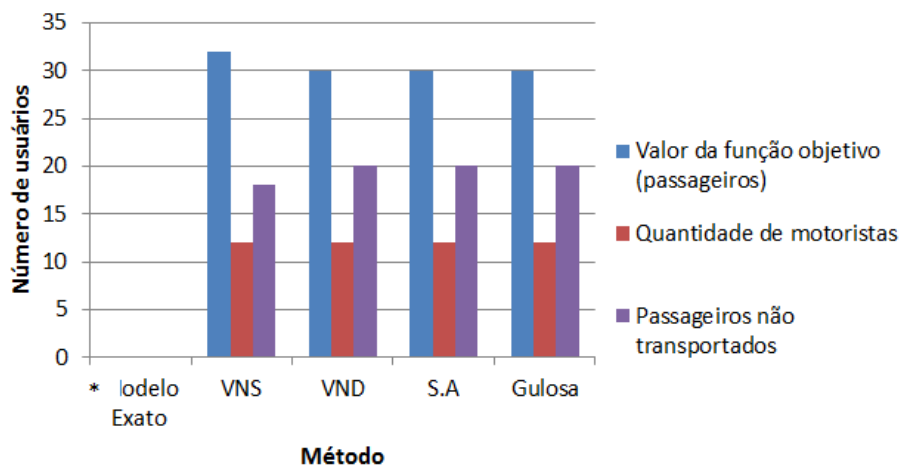
ressaltar que isto se aplica ao problema presente neste trabalho, uma vez que para problemas maiores, testar todas as combinações possíveis (procedimento VND), pode se tornar inviável, ainda que haja também problemas com o procedimento SA, pois em problemas maiores pode haver uma dificuldade ainda maior de encontrar vizinhos controlados.

A heurística gulosa apresentou um resultado razoável em comparação aos demais. A heurística ainda se mostra eficiente como solução inicial para os demais procedimentos, uma vez que as meta heurísticas encontraram soluções próximas à ótima. Contudo, como abordado nas seções anteriores, pode que diferentes soluções iniciais forneçam soluções melhores, desta forma, são necessários outros estudos para análise da qualidade da solução inicial dos algoritmos apresentados.

5.3.2 Cenário 2

A Figura 29 apresenta a síntese do valor da função objetivo, quantidade de motoristas utilizados e passageiros não transportados.

Figura 29 – Síntese dos resultados para o cenário 2



* Método não processado

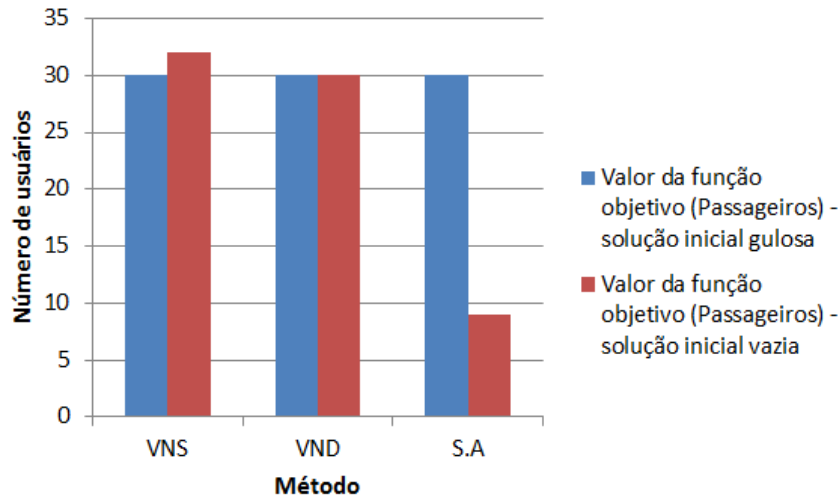
Fonte: O Autor (2018)

Encontrou-se, com a meta heurística VNS, uma solução melhor que as soluções dos demais métodos. Entretanto não é possível afirmar a qualidade desta solução em comparação com a solução ótima uma vez que o modelo exato não pode ser processado. Ainda pode-se notar que todos os métodos aproximados obtiveram soluções semelhantes. Tal fato reforça o fato de que a solução inicial influencia diretamente na qualidade da solução final.

5.3.2.1 Cenário 2 - Considerando-se diferentes soluções iniciais

Buscou-se, para o cenário 2, realizar a análise referente à influência da solução inicial utilizada pelos métodos meta heurísticos na solução final dos mesmos. Para o presente trabalho foram consideradas as soluções iniciais vazias e da heurística gulosa. A Figura 30 apresenta os resultados da análise da influência da solução inicial.

Figura 30 – Resultados das diferentes soluções iniciais utilizadas



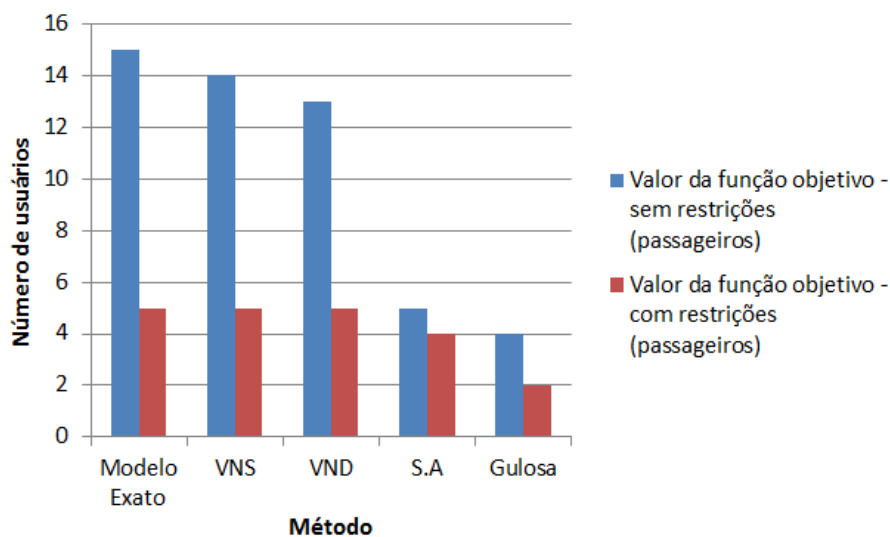
Fonte: O Autor (2018)

As respostas obtidas por meio da solução inicial gulosa apresentaram uma convergência para o valor de 30 usuários. No entanto, a melhor solução encontrada foi por meio da solução inicial vazia aplicada a meta heurística VNS. Para a meta heurística SA ao utilizar a solução inicial vazia, o valor da função objetivo foi reduzido drasticamente de 30 para 9 passageiros. Portanto, salienta-se a importância da necessidade de estudos referentes a solução inicial para as meta heurísticas utilizadas.

5.3.3 Cenário 3

Os métodos de otimização propostos também foram analisados levando em consideração o cenário 3 com e sem as restrições de horário de saída e tempo de espera máximo. Tem-se os resultados na Figura 31.

Figura 31 – Síntese dos resultados para o cenário 3



Fonte: O Autor (2018)

A Figura 31 mostra que o valor da função objetivo sem tais restrições tende a aumentar de modo considerável nos métodos Exato, VNS e VND. Observa-se que as meta heurísticas VND e VNS apresentaram os resultados mais próximos ao resultado pelo modelo exato. A heurística gulosa não obteve desempenho melhor devido à implementação sugerida, apresentada na Seção 4.2, que aloca pessoas apenas se forem passageiros. A meta heurística SA não apresentou resultados satisfatórios para o cenário 3.

5.4 Quantidade de Veículos e Taxa de Ocupação

Nos resultados apresentados na Seção 5.3, pode ser notado que algumas soluções obtiveram a mesma quantidade de passageiros no sistema. Porém, pode ser que existam diferenças entre as soluções e devido a fatores além da quantidade de passageiros exista uma solução melhor que a outra.

Desta forma, foi analisado a quantidade de veículos utilizados em cada solução, fator secundário no modelo o qual implicitamente é assumido que quanto maior a quantidade de usuários como passageiros, menor a quantidade de veículos circulando nas vias. Tal fato é considerado verdadeiro, quando motoristas do sistema são alocados como passageiros.

A taxa de ocupação do sistema pode ser definida como $tx = \frac{npessoas}{nvagas}$, na qual considera-se $npessoas$ o número de pessoas transportadas no veículo e $nvagas$ a quantidade de vagas totais no veículo. A taxa de ocupação média considera a média da taxa de ocupação dos veículos utilizados e dos veículos dos motoristas que viajam sozinhos, os quais possuem $tx = 0$. A taxa média dos veículos utilizados considera apenas a média de tx dos veículos utilizados na solução. Sendo assim, mesmo que uma solução tenha uma quantidade maior de veículos, ainda é possível que sua taxa de ocupação seja maior, ou seja, existe um melhor aproveitamento dos espaços dos veículos.

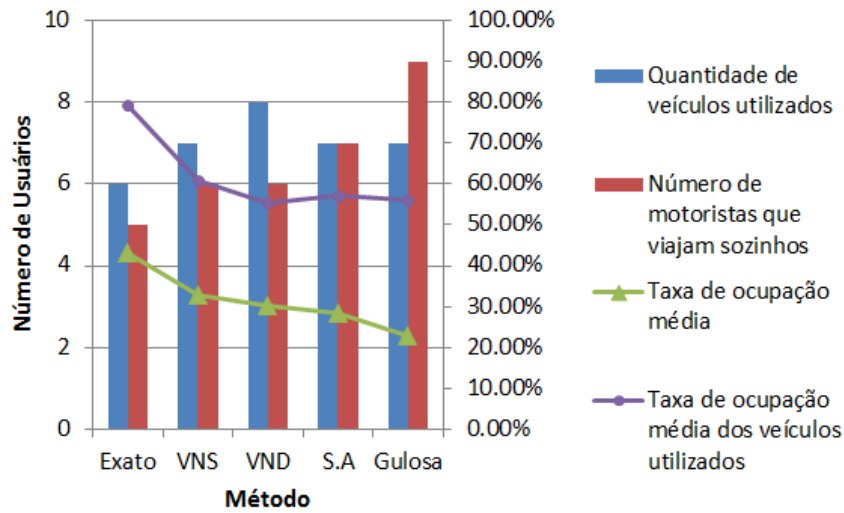
Vale ressaltar que, para o presente trabalho, apenas o motorista 2 possui um veículo de capacidade diferente de 4. A capacidade do veículo do motorista 2 é de apenas um passageiro. Desta forma, caso o motorista 2 possua um passageiro sua taxa de ocupação será $tx = 1$ ou de 100%.

5.4.1 Cenário 1

Ao aplicar-se os métodos de otimização para o cenário 1 foram obtidos os resultados apresentados na Figura 32 referentes a taxa de ocupação e quantidade de veículos.

Pode-se notar pela Figura 32 que o método exato teve uma melhor ocupação dos veículos, indicando que houve um melhor aproveitamento dos lugares nos veículos. Já a heurística gulosa apresenta uma grande quantidade de motoristas que viajam sozinhos e os passageiros que são transportados estão divididos em vários veículos diferentes. A solução SA apresenta uma maior quantidade de motoristas que viajam sozinhos quando comparada com a solução VND, entretanto quando observa-se a taxa de ocupação média dos veículos utilizados, a solução SA teve um desempenho levemente melhor. Tal fato indica que, embora ambas soluções mostraram o mesmo resultado na função objetivo, a solução SA apresentou um

Figura 32 – Quantidade de motoristas e taxa de ocupação para o cenário 1



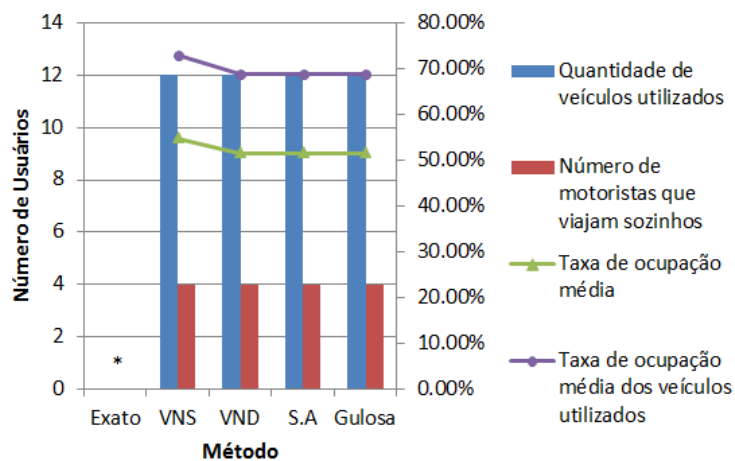
Fonte: O Autor (2018)

melhor aproveitamento dos lugares nos veículos. Entre as meta heurísticas, a VNS se destaca por possuir uma taxa de ocupação melhor, possuindo uma quantidade de veículos igual a VND e uma quantidade de motoristas que viajam sozinhos como a SA.

5.4.2 Cenário 2

Para o cenário 2 apresenta-se na Figura 33 os resultados para os métodos de otimização.

Figura 33 – Quantidade de motoristas e taxa de ocupação para o cenário 2



* Método não processado

Fonte: O Autor (2018)

Pode-se notar pela Figura 33 que a taxa de ocupação nos veículos para todos os métodos processados foi idêntica. O aumento na taxa de ocupação da solução VNS se deu pelo fato de alocar dois passageiros a mais na mesma quantidade de veículos que os demais métodos. É possível afirmar que o parâmetro que é alterado entre as soluções é a roteirização

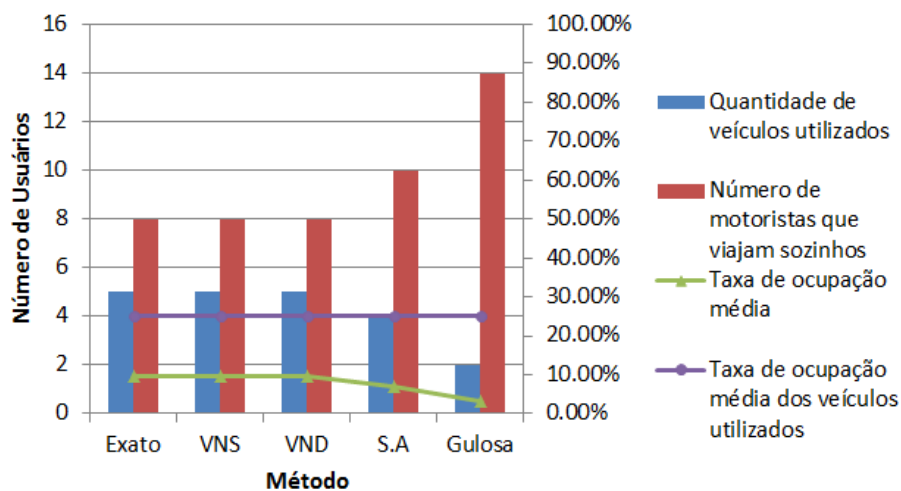
dos motoristas

5.4.3 Cenário 3

O cenário 3 é apresentado em dois momentos. O primeiro é o cenário com as restrições (4.6) e (4.7) de tempo de saída dos usuários e janela de tempo máxima, respectivamente. O segundo é sem tais restrições.

A Figura 34 apresenta os resultados para o cenário 3 com as restrições referidas anteriormente.

Figura 34 – Quantidade de motoristas e taxa de ocupação para o cenário 3 com as restrições (4.6) e (4.7)



Fonte: O Autor (2018)

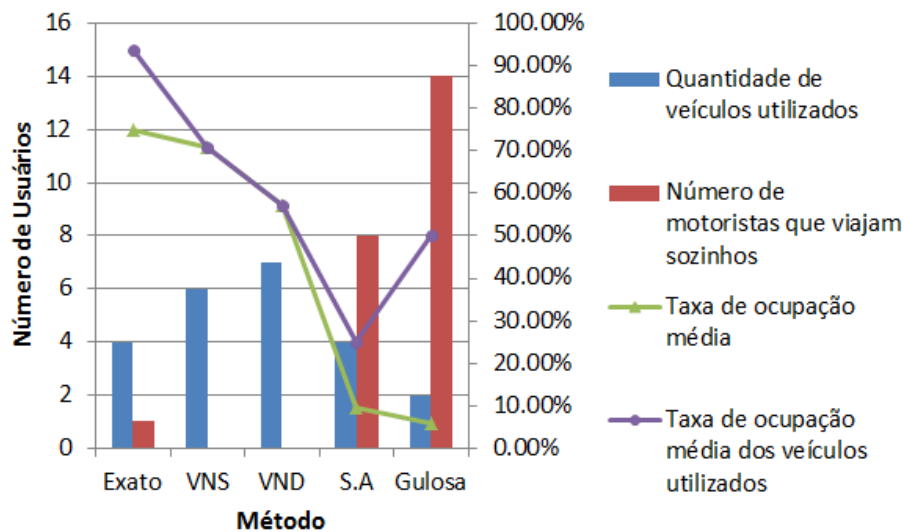
Nota-se que para todos os métodos a taxa de ocupação média dos veículos foi a mesma, uma vez que todos os veículos utilizaram apenas 25% da suas capacidades. Pode-se notar que houve uma grande quantidade de veículos não utilizados devido as restrições.

A Figura 35 apresenta os resultados para o cenário 3 sem as restrições (4.6) e (4.7).

Pode-se notar pela Figura 35 que ao retirar as restrições de tempo de saída e tempo máximo de espera a quantidade de motoristas que viajam sozinhos, para o modelo exato e para as meta heurísticas VNS e VND, foi muito baixa quando comparada com o cenário 3 com as restrições. Os veículos utilizados na solução pelo modelo exato possuíam quase 100% de seus lugares ocupados, mostrando que quase todos os usuários poderiam se locomover com poucos motoristas. Embora as meta heurísticas VNS e VND não tenham motoristas viajando sozinhos, possuem taxas de ocupação menores devido a necessidade de alocar mais motoristas para todos os passageiros.

Vale ressaltar que as meta heurísticas VND e VNS deixaram de alocar 6 e 7 passageiros, respectivamente. Desta forma, embora foi reduzida a quantidade de veículos circulando nas vias, não foi possível alocar todos os usuários nos veículos utilizados. A heurística gulosa teve a maior quantidade de motoristas não utilizados, desta forma, sua taxa de ocupação média foi a menor de todas. Entretanto, quando comparada com a meta heurística SA, teve-se

Figura 35 – Quantidade de motoristas e taxa de ocupação para o cenário 3 sem as restrições (4.6) e (4.7)



Fonte: O Autor (2018)

uma melhor ocupação nos veículos utilizados, ou seja, a meta heurística SA alocou mais motoristas para transportar mais passageiros, porém nem todos os veículos estavam com uma alta ocupação.

5.5 Distâncias Percorridas

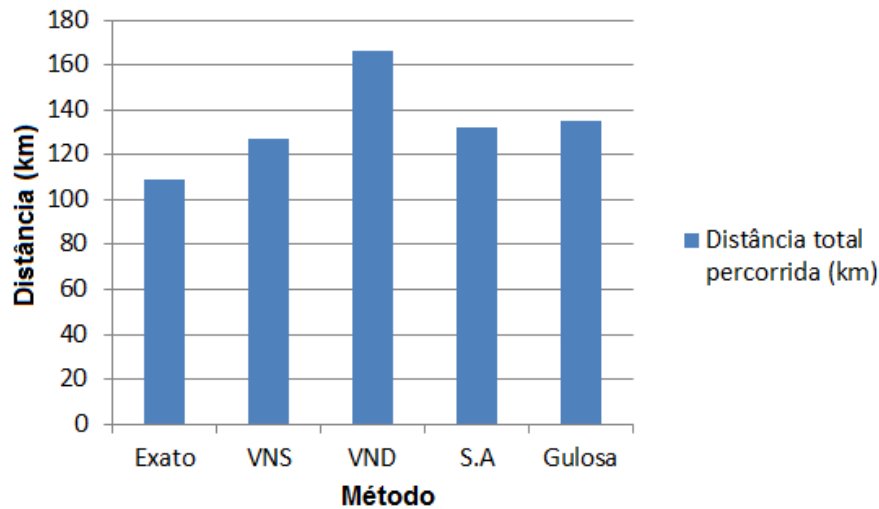
Outro critério de análise considerado foi a distância percorrida pelos motoristas, uma vez que foi mostrado nas seções anteriores, que os valores da função objetivo e a taxa de ocupação dos veículos são as mesmas. As distâncias foram obtidas por meio do Software QGIS, o qual, por meio das informações georreferenciadas, determina a distância euclidiana entre os pontos, ou seja, em linha reta.

5.5.1 Cenário 1

As informações presentes na Figura 36 referem-se ao cenário de 40 usuários. A distância total percorrida se refere a distância percorrida pelos motoristas em seus trajetos determinados pelo respectivo método de otimização somados à distância dos motoristas que viajam sozinhos até o destino final.

Pode-se notar que a solução obtida pelo modelo exato possui a menor distância percorrida, ou seja, embora exista a possibilidade da rota de um determinado motorista não estar otimizada, a distância percorrida por todos os motoristas será menor. Observa-se que a heurística VND teve a maior distância percorrida, possivelmente pela quantidade de veículos utilizados neste cenário ter sido maior que os demais métodos. A meta heurística SA se mostrou melhor que a VND neste quesito, uma vez que ambas transportam a mesma quantidade de passageiros, mas a solução pelo SA apresenta uma distância percorrida total menor.

Figura 36 – Distância total percorrida pelos motoristas no cenário 1



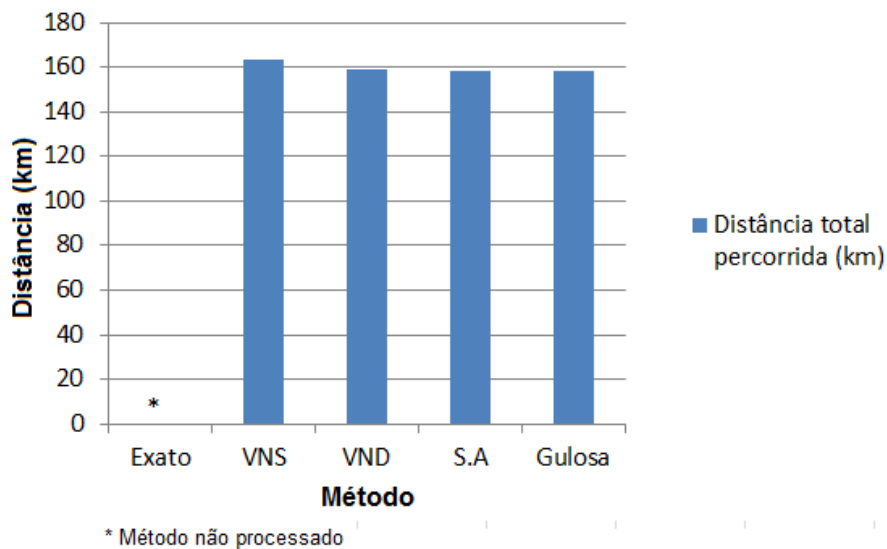
Fonte: O Autor (2018)

Apresentou-se, na meta heurística VNS, resultados satisfatórios, uma vez que é o método aproximado que mais alocou passageiros e teve a menor distância percorrida. O resultado da heurística gulosa se mostrou inferior aos demais métodos, ainda que não possua uma distância percorrida tão elevada como a VND, a heurística aloca menos usuários e percorre uma distância semelhante aos demais métodos aproximados.

5.5.2 Cenário 2

A Figura 37 apresenta os resultados referentes as distâncias para o cenário 2.

Figura 37 – Distância total percorrida pelos motoristas no cenário 2



Fonte: O Autor (2018)

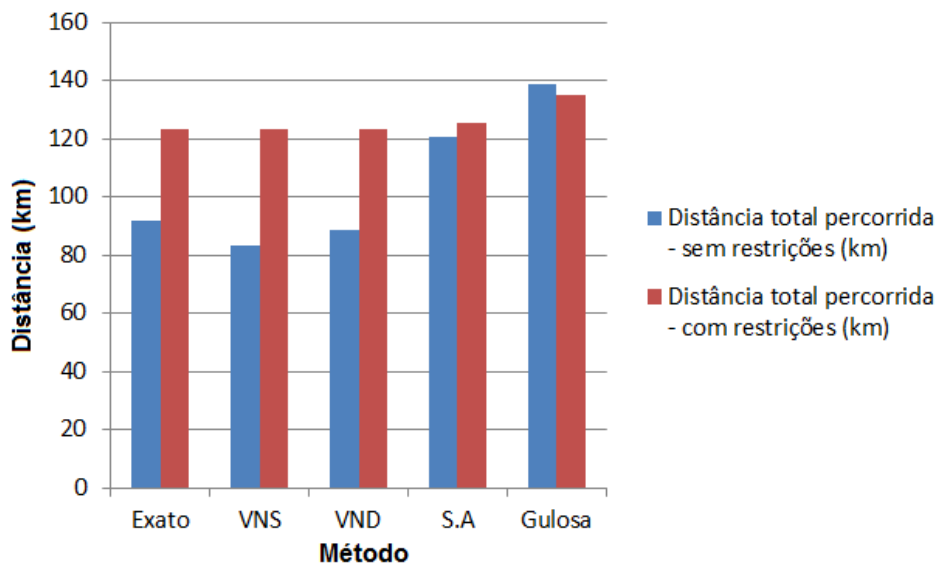
Pode-se notar que para todos os métodos processados a distância percorrida total foi semelhante. O fato se dá pelo motivos de homogeneidade das rotas apresentadas pelos

métodos, nas quais muda-se apenas alguns passageiros de posição. A meta heurística VNS apresentou uma distância percorrida maior devido aos dois passageiros adicionais em sua solução, motivo que gerou uma roteirização diferente aos motoristas. Desta forma, não há vantagens significativas para a distância entre os métodos aproximados.

5.5.3 Cenário 3

Na Figura 38 apresenta-se a comparação da aplicação dos métodos para o cenário 3 com e sem as restrições de tempo inicial de viagem e tempo de espera máximo, restrições (4.6) e (4.7) respectivamente.

Figura 38 – Distância total percorrida pelos motoristas no cenário 3 com e sem as restrições (4.6) e (4.7)



Fonte: O Autor (2018)

Observa-se que quando o problema é relaxado (desconsideradas as restrições) a distância total percorrida pelos motoristas, do método exato e das meta heurísticas VND e VNS, são reduzidas drasticamente. Isto se dá por alocar uma quantidade maior de motoristas como passageiros, reduzindo a distância percorrida pelos motoristas que viajam sozinhos. A distância total percorrida do método exato é superior à meta heurística VNS, pois na solução pelo modelo exato há um motorista viajando sozinho, fato que não ocorre na solução obtida pela VNS.

A solução gulosa apresentou um aumento na distância total percorrida, pois a solução sem as restrições aloca dois passageiros adicionais em comparação a solução com as restrições e nenhum destes passageiros é um motorista. Desta forma, os motoristas alocados para buscar os passageiros adicionais necessitam percorrer uma distância maior. A solução SA não apresentou vantagens significativas uma vez que na solução relaxada ainda existem muitos motoristas que viajam sozinhos.

6 CONCLUSÕES

O sistema de carpooling apresenta-se como um sistema eficiente para a locomoção de pessoas em uma cidade. O sistema apresenta vantagens nas reduções de veículos circulando nas vias e, conseqüentemente, de tempo de viagem, poluentes, entre outros. Uma forma de melhorar esta eficiência é por meio da otimização de passageiros e motoristas.

Desta forma, o presente trabalho mostrou a aplicação de métodos de otimização para um problema de carpooling com o objetivo de maximizar a quantidade de usuários como passageiros.

No modelo matemático proposto para o problema de carpooling apresentou-se melhorias para que o mesmo refletisse melhor a realidade do problema de carpooling, considerando-se as restrições de tempo inicial e tempo máximo de espera dos usuários.

As dificuldades no desenvolvimento do trabalho se deram na evolução do modelo exato e na escolha e formulação dos métodos aproximados, uma vez que o problema do carpooling apresenta-se com peculiaridades específicas.

Uma problemática foi a formulação de soluções iniciais a serem consideradas para o problema, as quais necessitam outros estudos para determinar a melhor solução inicial para os métodos aproximados e as vizinhanças escolhidas para os mesmos.

Destaca-se outra dificuldade referente aos tempos de viagem considerados entre os usuários. O qual não é constante em todo o trajeto, variando conforme o fluxo no meio urbano.

Ainda com tais dificuldades, os resultados dos métodos se apresentaram satisfatórios, sendo possível realizar a avaliação de desempenho entre os métodos. Mostrou-se que os métodos aproximados, da maneira que foram apresentados, podem ser considerados aceitáveis para a solução do problema de carpooling, uma vez que foi possível apresentar um aumento na quantidade de usuários como passageiros de carpooling.

A meta heurística VNS, para o cenário 3, apresentou um valor de função objetivo de 32 usuários como passageiros, um total de aproximadamente 48,5% de todos os 66 usuários, em um tempo de processamento de 189,96 segundos. Os demais métodos aproximados apresentaram uma solução de 30 usuários como passageiros, um total de aproximadamente 45% dos usuários.

Com tais resultados obtidos pode-se concluir que as vizinhanças escolhidas, embora necessitem de mais estudos, se apresentaram eficientes na solução do problema de carpooling. Contudo, os métodos heurísticos e meta heurísticos não possuem garantia da otimalidade da solução, sendo assim sugere-se, sempre que possível, a utilização de métodos exatos de solução.

6.1 Trabalhos Futuros

Para trabalhos futuros propõe-se:

1. Estudar o impacto da solução inicial na solução final de métodos meta heurísticos;
2. Apresentar novas vizinhanças para os métodos meta heurísticos apresentados;
3. Propor novas restrições, ou melhoria das restrições existentes, no modelo matemático de carpooling proposto;
4. Melhorar a maneira de quantificar as distâncias entre os usuários;
5. Avaliar a quantidade de soluções inválidas obtidas pelos algoritmos propostos.

REFERÊNCIAS

- ABRAHAMSE, W.; KEALL, M. Effectiveness of a web-based intervention to encourage carpooling to work: A case study of wellington, new zealand. **Transporte Policy**, v. 21, p. 45–51, 2012.
- ARAUJO, H. A. d. **Algoritmo Simulated Annealing: Uma Nova Abordagem**. Dissertação (mathesis) — Universidade Federal de Santa Catarina, Florianópolis, maio 2001.
- ARENALES, M. et al. **Pesquisa operacional: para cursos de engenharia**. 2. ed. Rio de Janeiro: Elsevier, 2015. 723. ISBN 978-85-352-7161-4.
- BELLMORE, M.; NEMHAUSER, G. L. The traveling salesman problem: A survey. **Operations Research**, v. 16, n. 3, p. 538–558, 1968.
- BOAVENTURA NETTO, P. O. **Grafos: teoria, modelos, algoritmos**. 4. ed. São Paulo: E. Blucher, 2006. ISBN 85-212-0391-8.
- BODIN, L. et al. **Routing and Scheduling of vehicles and crews: the state of the art**. England: Pergamon Press, 1983. v. 10.
- BRUGLIERI, M. et al. Poliunipool: a carpooling system for universities. **Procedia Social and Behavioral Sciences**, v. 20, p. 558–567, 2011.
- COLIN, E. C. **Pesquisa operacional : 170 aplicações em estratégia, finanças, logística, produção, marketing e vendas**. Rio de Janeiro: LTC, 2013.
- CORREIA, G.; VIEGAS, J. M. Carpooling and carpool clubs: Clarifying concepts and assessing value enhancement possibilities through a Stated Preference web survey in Lisbon, Portugal. **Transportation Research Part A: Policy and Practice**, v. 45, n. 2, p. 81–90, 2011.
- ECCEL, R. A. L.; TAGLIALENHA, S. L. d. S. Problema de caronas: um modelo matemático considerando um nova abordagem. In: **XXIX ANPET - Congresso de Pesquisa e Ensino em Transportes, 2015**. Ouro Preto, MG: [s.n.], 2015.
- FERRARI, E. et al. The car pooling problem: Heuristic algorithms based on savings functions. **Journal of Advanced Transportation**, v. 37, n. 3, p. 243–272, 2003.
- GAO, J.; SUN, L.; GEN, M. A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. **Computers & Operations Research**, v. 35, p. 2892–2907, 2008.
- GOLDBARG, M. C.; LUNA, H. P. L. **Otimização combinatória e programação linear : modelos e algoritmos**. 2. ed. Rio de Janeiro: Elsevier, 2005.
- HANSEN, P.; MLADENOVIC, N. Variable neighborhood search: Principles and applications. **European Journal of Operational Research**, v. 130, p. 449–467, 2001.
- HARTMAN, I. B.-A. Optimal assignment for carpooling. **Draft**, 2013.

HARTMAN, I. B.-A. et al. Theory and practice in large carpooling problems. **Procedia Computer Science**, v. 32, p. 339–347, 2014.

HILLIER, F. S.; LIEBERMAN, J. G. **Introdução à pesquisa operacional**. 9. ed. Porto Alegre: AMGH, 2013. 1005p.

HUSSAIN, I. et al. Organizational and agent-based automated negotiation model for carpooling. **Procedia Computer Science**, v. 37, p. 396–403, 2014.

JUNGNICKEL, D. **Graphs, Networks and Algorithms**. 3rd. ed. [S.l.]: Springer, 2010. v. 5. ISBN 978-3-642-09186-5.

KIRKIPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. **Science**, v. 220, n. 4598, p. 671–680, maio 1983.

KNAPEN, L. et al. Scalability issues in optimal assignment for carpooling. **Journal of Computer and System Sciences**, v. 81, n. 3, p. 568 – 584, 2015. ISSN 0022-0000. Special Issue on selected papers from the 4th International Conference on Ambient Systems, Networks and Technologies (ANT 2013).

METROPOLIS, N. et al. Equation of state calculations by fast computing machines. **The Journal of Chemical Physics**, v. 21, n. 6, p. 1087–1092, jun. 1953.

MLADENOVIC, N.; HANSEN, P. Variable neighborhood search. **Location Science**, v. 5, n. 4, p. 207 – 226, 1997.

QI, X.; WANG, L.; WANG, X. Optimization of carpooling based on complete subgraphs. **35th Chinese Control Conference (CCC)**, p. 9294–9299, jul. 2016.

ROSENKRANTZ, D. J.; STEARNS, R. E.; LEWIS II, P. M. An analysis of several heuristics for the traveling salesman problem. **SIAM J. Comput.**, v. 6, n. 3, p. 563–581, 1977.

TAHA, H. A. **Pesquisa operacional: uma visão geral**. 8. ed. São Paulo: Pearson Prentice Hall, 2008. 359 p. ISBN 978-85-7605-150-3.

XIA, J.; CURTIN, K. M.; ZHAO, Y. A new model for a carpool matching service. **PLOS ONE**, v. 10, n. 6, p. 1–23, jun. 2015.