Heurística de Busca de Vizinhança Variável para Otimização do Problema de Roteamento de Veículos

ABSTRACT

This work investigates a computational method for the optimization of the Enabled Vehicle Routing Problem. To propose the method, the variable neighborhood search meta-heuristic (Variable Neighborhood Search-VNS) was used. The proposed heuristic contains the implements of 5 neighborhood structures, 4 of them built with deterministic methods, to explore the search space in order to obtain local minima. In addition to these strategies, a search with random factors is also presented, to try to escape from local minima. For the computational tests, 4 groups of instances were used, with different characteristics, seeking to evaluate the heuristic in different scenarios. In the results it was found that the heuristic was able to obtain good results, with an error of up to 8% of the solution.

KEYWORDS

Problema de Roteamento de Veículos, *Variable Neighborhood Search*, Otimização.

1 Introdução

Em virtude da sua aplicabilidade, importância e extensões, o Problema de Roteamento de Veículos (PRV) é um dos problemas mais estudados na área de otimização combinatória. Sua aplicabilidade existe em muitas áreas como a distribuição de manufaturados, de bebidas, de alimentos, de produtos químicos, de derivados do petróleo, de gás, entrega de correspondência, recolhimento de materiais, roteamento em linhas aéreas e de distribuição de vagões ferroviários transporte de pessoas, e outros. Esse problema tem um papel fundamental na área de gerenciamento da distribuição e logística [1].

De maneira geral, o roteamento de veículos pode ser definido como o atendimento de clientes geograficamente dispersos. Cada cliente é representado como um nó de demanda em um grafo, e para cada ligação entre um par de nós, existe uma distância e custo associados. Para atendê-los, utiliza-se uma frota de veículos que partem e retornam a um depósito central.

Apesar do seu enunciado ser relativamente simples, o problema apresenta elevada complexidade computacional, pois é do tipo NP-difícil, o que significa que a dificuldade para achar a solução ótima cresce de maneira exponencial à medida que o número de elementos do problema aumenta [2]. O objetivo do PRV é determinar o conjunto de rotas de menor custo, que atenda às necessidades dos nós, sujeito a restrições operacionais, tais como veículos com capacidade limitada, período determinado para chegadas e partidas, duração da jornada de trabalho, duração da rota, existência de múltiplos depósitos para carga e descarga, necessidade de levar e trazer cargas, dentre outras [3].

Os problemas de roteamento de veículos podem ser vistos como problemas de múltiplos caixeiros viajantes com restrições de capacidade e outras restrições que dependem de cada aplicação [3]. As rotas devem iniciar e terminar no depósito, sendo que cada cliente pertence somente a uma rota e a demanda de todos os nós deve ser atendida. A demanda é determinística em cada nó e cada veículo possui capacidade conhecida. A Figura 1, apresenta um exemplo onde as rotas são estabelecidas para que 5 veículos atendam os pontos (clientes) a partir de um único depósito.

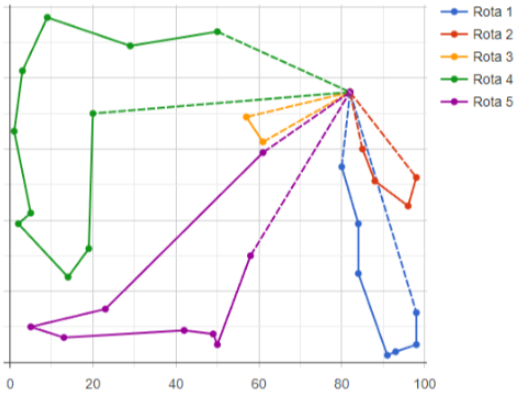

Figura 1: Ilustração de um PRV com 5 veículos e 31 clientes.

O objetivo desse trabalho é desenvolver um método computacional que possa ser aplicado em cenários reais do problema de roteamento de veículos. Assim, nesse trabalho propomos um algoritmo heurístico baseado na meta-heurística de busca de vizinhança variável (*Variable Neighborhood Search* VNS [4]), onde nossa contribuição está na diferença das estruturas de vizinhanças propostos. No trabalho serão apresentadas 5 vizinhanças: sendo 4 estratégias de busca determinísticas, com o objetivo de melhoria das soluções e manter a convergências das mesmas; e 1 de busca aleatória, para tentar escapar de mínimos locais. Os resultados numéricos mostram que a heurística proposta foi capaz de se adaptar para obter soluções de boa qualidade em diferentes cenários. Adicionalmente, nos testes de escalabilidade, obteve resultados que indicam a possibilidade de uso em sistemas reais.

O restante do artigo está dividido da seguinte forma: na Seção 2 são apresentados trabalhos correlatos; na Seção 3 estão descritas as vizinhanças para o VNS proposto para o problema de roteamento de veículos; na Seção 4 contém a descrição dos testes

computacionais e instâncias utilizadas; na Seção 5 são apresentados resultados numéricos e as análises dos mesmos; e finalmente, na Seção 6 temos as conclusões finais e apontamentos para trabalhos futuros.

2 Trabalhos Correlatos

Vários algoritmos de otimização têm sido utilizados em diversos estudos para resolver o PRV [5]. Em [6] foi proposto um algoritmo heurístico (HA) baseado em busca tabu (TS) e busca de vizinhança adaptativa. No trabalho apresentado em [7] é introduzido um algoritmo de otimização baseado em reatores químicos, combinado com um algoritmo de busca tabu unificada adaptável. Em [8] foi desenvolvido uma nova estratégia de divisão de rotas e a integrou com busca de vizinhança variável.

No trabalho apresentado em [9] é proposto a otimização de colônias de formigas (ACO) para resolver o PRV capacitado, onde um novo algoritmo de membrana (MA_ACO) é apresentado. Assim, no trabalho são analisadas duas versões de ACO. Em [10] foi usado o ACO como uma busca de vizinhança para fornecer um melhor nível de diversificação. No trabalho foi estudado nos três algoritmos meta-heurísticos híbridos, sendo o sistema elitista-formiga, big bang-big crunch e busca de dispersão. Também utilizando uma ACO, o algoritmo proposto em [11]. permitiu que cada formiga representasse todos os caminhos de uma solução viável em vez de um único caminho. Adicionalmente, usando a estratégia proposta em [11], o trabalho apresentado em [12] propõem uma melhora na diversidade de ACO usando o algoritmo do vaga-lume.

Na literatura existem vários algoritmos genéticos (AG) desenvolvidos para resolver VRP, entre eles destacamos o [13], que também considera a capacidade do veículo. Ainda do AG, citamos também [14] onde é proposto um GA hibridizado com o algoritmo ACO. No trabalho a população inicial foi construída pelo algoritmo ACO e melhorada pelo GA.

Utilizando a meta-heurística de enxame de partículas (*Particle Swarm Optimization* - PSO) destacamos o trabalho [15], onde também é considerada a restrição da capacidade do veículo. Os autores alegaram que o uso de um algoritmo de varredura adaptativa para construir a população inicial para o VRP é melhor do que o algoritmo de varredura padrão. Quatro mapas de caos diferentes foram incorporados com um algoritmo discreto de enxame de partículas. Os resultados computacionais mostraram que o uso de mapas de caos com um algoritmo PSO foi significativamente melhor do que geradores de números pseudoaleatórios.

Considerando a meta-heurística VRP, o trabalho [16] aborda o problema de roteamento de veículos com janelas de tempo (VRPTW). No trabalho é proposto uma formulação matemática de programação linear inteira, bem como uma meta-heurística geral de busca de vizinhança variável (VNS) para resolver grandes instâncias do problema. Para o VNS, inicialmente é criado uma solução inicial usando a heurística de Solomon, depois busca-se minimizar o número de rotas usadas e, em seguida, a distância total percorrida por todos os veículos é minimizada. Os resultados computacionais mostram a eficiência da abordagem proposta.

3 VNS Proposto para o Problema de Roteamento de Veículos (VNS-PRV)

A quantidade de veículos de uma frota, a capacidade dos veículos, os custos para se chegar em cada cliente e as demandas deles são as informações necessárias para o algoritmo executar e retornar a melhor rota encontrada. As instâncias utilizadas nesse trabalho foram retiradas do site CVRPLIB da PUC-Rio [17]. Nesse site existem diversas instâncias para o Problema do Roteamento de Veículos, variando a quantidade de veículos e clientes atendidos, também fornecendo o melhor custo para essas instâncias. Essas instâncias são fornecidas em arquivos do tipo txt que possuem as informações de quantidade de veículos, capacidades, demandas e custos (distâncias entre clientes).

A busca de vizinhança variável (VNS) é uma meta-heurística para resolver problemas de otimização combinatória cuja ideia básica é uma mudança sistemática de vizinhança, tanto dentro de uma fase de descida para encontrar um ótimo local, quanto em uma fase de perturbação para sair do vale correspondente. A seguir apresentamos as estruturas de vizinhanças propostas nesse trabalho. Foram propostas 5 vizinhanças, sendo 4 determinísticas, variando a posição do sequenciamento dos clientes em um veículo e dos clientes entre os veículos, e uma com busca aleatória variando a posição da sequência se dois clientes em dois veículos.

3.1 Vizinhanças

Para as variações nas vizinhanças considere o nó 0 (zero) como a garagem, ou seja, a rota deve iniciar e terminar no nó 0.

3.1.1 2-Opt dos Clientes de um Veículo (V1)

Está vizinhança busca encontrar a melhor rota possível de um veículo apenas fazendo a mudança na ordem de visita dos clientes pelo veículo. Nessa busca a troca na ordem de atendimento é feita pelo cliente adjacente. Assim, são realizadas todas as trocas de pares de nós adjacentes. A Tabela 1 ilustra o funcionamento dessa vizinhança. Considere a rota inicial de um veículo com custo de 253,8. Os vizinhos, utilizando a estratégia 2-Opt dos Clientes de um Veículo estão na Tabela 1. O vizinho escolhido por essa estratégia foi destacado na tabela.

Tabela 1: Estratégia de Busca para a vizinhança 2-Opt dos Clientes de um Veículo

	Rota	**Custo**
Rota Inicial	[0, 9, 7, 6, 14, 0]	253.8
1	[0, 7, 9, 6, 14, 0]	231.9
2	[0, 6, 7, 9, 14, 0]	250.7
3	[0, 14, 6, 7, 9, 0]	253.8

4	**[0, 9, 6, 7, 14, 0]**	**229.8**
5	[0, 9, 14, 6, 7, 0]	244.1
6	[0, 9, 7, 14, 6, 0]	281.6

3.1.2 Troca de Clientes entre Veículos (V2)

Essa vizinhança busca melhorar o custo total trocando os clientes entre dois veículos. Nessa estratégia cada cliente de um veículo é retirado e testado em todas as posições de sequenciamento do outro veículo. Na Tabela 2 é ilustrado as trocas realizadas pela busca, a partir de uma solução inicial.

Tabela 2: Estratégia de Busca para a vizinhança Troca de Clientes entre Veículos

	Rota	**Custo**
Rota Inicial	[[0, 9, 7, 6, 0], [0, 12, 13, 1, 0]]	366.1
1	[[0, 7, 6, 0], [0, 12, 9, 13, 1, 0]]	382.7
2	[[0, 7, 6, 0], [0, 12, 13, 9, 1, 0]]	387.9
3	[[0, 7, 6, 0], [0, 12, 13, 1, 9, 0]]	380.4
4	**[[0, 9, 6, 0], [0, 12, 7, 13, 1, 0]]**	**316.0**
5	[[0, 9, 6, 0], [0, 12, 13, 7, 1, 0]]	317.9
6	[[0, 9, 6, 0], [0, 12, 13, 1, 7, 0]]	328.5
7	[[0, 9, 7, 0], [0, 12, 6, 13, 1, 0]]	365.5
8	[[0, 9, 7, 0], [0, 12, 13, 6, 1, 0]]	368.3

3.1.3 Circ-Opt dos Clientes de um Veículo (V3)

Está vizinhança busca também encontrar a melhor rota possível de um veículo apenas fazendo a mudança de ordem de visita dos clientes pelo veículo, porém diferente da vizinhança 2-opt, a troca não é entre dois clientes. A troca da ordem de atendimento dos clientes será feita como uma lista circular, o último cliente passará a ser o primeiro e o restante terá a posição do sequenciamento adicionado de uma posição. A Tabela 3 ilustra essa estratégia de busca de boas soluções. O vizinho escolhido por essa vizinhança foi encontrado na terceira troca do algoritmo e melhorou o custo em 111.3 pontos.

Tabela 3: Estratégia de Busca para a vizinhança Circ-Opt dos Clientes de um Veículo

	Rota	**Custo**
Rota Inicial	[0, 9, 7, 6, 8, 0]	327.6
1	[0, 8, 9, 7, 6, 0]	266.2
2	[0, 6, 8, 9, 7, 0]	284,7
3	**[0, 7, 6, 8, 9, 0]**	**216.3**

3.1.4 Realocação de Cliente entre Veículos (V4)

Está vizinhança busca melhorar o custo total realocando o cliente de um veículo em todas as posições de sequenciamento do outro veículo, e vice-versa. Nota-se que a vizinhança Troca de Clientes entre Veículos (3.1.2) é um subconjunto dessa aqui apresentada, sendo esse consideravelmente maior. Em instâncias maiores essa diferença influenciará no tempo de processamento, de tal forma que o uso sucessivo da vizinhança 3.1.4 tornaria o tempo de resposta da heurística inviável para o uso sistemas reais. Assim, a aplicação do VNS proposto em um sistema comercial deve dimensionar o tempo de processamento, baseando-se no tamanho da instância. Feito isso, o sistema deve limitar a aplicação da vizinhança Realocação de Cliente entre Veículos (3.14). Contudo, destacamos que o escopo desse trabalho se limitou a investigar a convergência do método e a qualidade das soluções obtidas.

A Tabela 4 ilustra a vizinhança Realocação de Cliente entre Veículos de uma solução.

Tabela 4: Estratégia de Busca para a vizinhança Realocação de Cliente entre Veículos

	Rota	**Custo**
Rota Inicial	[[0, 9, 7, 6, 0], [0, 12, 13, 1, 0]]	366.1
1	[[0, 7, 6, 0], [0, 12, 9, 13, 1, 0]]	382.7
2	[[0, 7, 6, 0], [0, 12, 13, 9, 1, 0]]	387.9
3	[[0, 7, 6, 0], [0, 12, 13, 1, 9, 0]]	380.4
4	**[[0, 9, 6, 0], [0, 12, 7, 13, 1, 0]]**	**316.0**
5	[[0, 9, 6, 0], [0, 12, 13, 7, 1, 0]]	317.9
6	[[0, 9, 6, 0], [0, 12, 13, 1, 7, 0]]	328.5
7	[[0, 9, 7, 0], [0, 12, 6, 13, 1, 0]]	365.5
8	[[0, 9, 7, 0], [0, 12, 13, 6, 1, 0]]	368.3
9	[[0, 9, 7, 0], [0, 12, 13, 1, 6, 0]]	381.3
10	[[0, 12, 13, 1, 0], [0, 9, 7, 6, 0]]	366.1
11	[[0, 13, 1, 0], [0, 9, 12, 7, 6, 0]]	391.9
12	[[0, 13, 1, 0], [0, 9, 7, 12, 6, 0]]	395.2
13	[[0, 13, 1, 0], [0, 9, 7, 6, 12, 0]]	380.5
14	[[0, 12, 1, 0], [0, 9, 13, 7, 6, 0]]	336.7

15	[[0, 12, 1, 0], [0, 9, 7, 13, 6, 0]]	334.5
16	[[0, 12, 1, 0], [0, 9, 7, 6, 13, 0]]	347.1
17	[[0, 12, 13, 0], [0, 9, 1, 7, 6, 0]]	384.9
18	[[0, 12, 13, 0], [0, 9, 7, 1, 6, 0]]	385.8
19	[[0, 12, 13, 0], [0, 9, 7, 6, 1, 0]]	383.3

3.1.5 Recombinação Aleatória de Veículos (V5)

Esta vizinhança utiliza o conceito de Mutação da Computação Evolucionária para tentar sair de mínimos locais do espaço de busca. Primeiramente são selecionados dois veículos, após a escolha dos veículos um cliente de cada veículo é selecionado aleatoriamente para serem trocados. Esse processo é repetido por algumas iterações e o melhor resultado é retornado.

A Tabela 5 apresenta algumas iterações, considerando uma solução inicial.

Tabela 5: Estratégia de Busca para a vizinhança Recombinação Aleatória de Veículos

Iteração	Rota	Custo
Rota Inicial	[[0, 9, 7, 10, 6, 0], [0, 8, 12, 13, 1, 0]]	651.3
1	[[0, 8, 12, 10, 6, 0], [0, 9, 7, 13, 1, 0]]	642.9
2	[[0, 8, 12, 13, 1, 10, 6, 0], [0, 9, 7, 0]]	662.1
3	[[0, 12, 13, 10, 6, 0], [0, 8, 9, 7, 1, 0]]	520.6
4	[[0, 13, 1, 10, 6, 0], [0, 8, 12, 9, 7, 0]]	691.5
5	[[0, 8, 12, 6, 0], [0, 9, 7, 10, 13, 1, 0]]	689.8
6	[[0, 8, 12, 13, 0], [0, 9, 7, 10, 6, 1, 0]]	668.4
7	[[0, 12, 13, 1, 0], [0, 8, 9, 7, 10, 6, 0]]	522.7
8	[[0, 9, 8, 12, 13, 6, 0], [0, 7, 10, 1, 0]]	568.8
9	[[0, 9, 8, 12, 13, 1, 6, 0], [0, 7, 10, 0]]	549.1
10	[[0, 9, 8, 12, 0], [0, 7, 10, 6, 13, 1, 0]]	513.0

3.1.6 Calibração da Heurística

A calibração de uma heurística consiste em obter uma combinação de parâmetros das funções do método que geram melhores resultados. Para os testes calibração do método proposto utilizamos a instância P-n50-k10 encontrada em [17]. Essa instância foi escolhida por possuir uma quantidade de clientes e veículos próxima de um caso real. Nos experimentos, as combinações de parâmetros foram executadas 5 vezes, sendo selecionado o melhor resultado obtido. Dois parâmetros foram calibrados, a ordem de execução das vizinhanças e a quantidade de execuções de cada vizinhança. Para a quantidade de execuções das vizinhanças foram feitos testes com: 10, 20, 30, 50, 100, 500, 1000. Foi constatado que a ordem de execução das vizinhanças não influencia na qualidade do resultado, sendo assim utilização na ordem sequencial [V1, V2, V3, V4 E V5] nos demais testes. Adicionalmente, não foram constatadas melhorias significativas após 500 iterações em cada busca, sendo esse o valor utilizado no restante dos experimentos.

4 Instâncias para os Testes Computacionais

A fim de avaliar a heurística proposta utilizamos algumas instâncias da base em [17]. Foram avaliadas 24 instâncias, divididas em 4 grupos, que buscam analisar as soluções obtidas pelas heurísticas propostas (VNS-PRV) em diferentes situações. O primeiro grupo (G1) busca analisar um cenário com uma pequena variação na quantidade de clientes e um número fixo de veículos. O segundo grupo (G2) avalia um cenário com uma grande variação de clientes e um número fixo de veículos. O terceiro grupo (G3) verifica o desempenho do algoritmo em um cenário de quantidade de clientes fixo com uma variação de número de veículos. E por fim, o quarto grupo (G4) avalia a escalabilidade da heurística para verificar se seria possível implantar o método proposto para a solução de problemas reais, com instâncias maiores.

A Tabela 6 apresenta os cenários investigados com as instâncias divididas em 3 grupos, denominados G1 – G3. No G1 o número de veículos foi fixado em 5, enquanto o número de clientes de cada instância começa em 32 e termina com uma instância de 51 clientes. No G2, o número de veículos foi fixado em 8, enquanto o número de clientes de cada instância começa em 16 e termina com uma instância de 101 clientes. Já no G3 a variação é feita no número de veículos, onde o número de veículos utilizados será 4, 8, 10 e 14, enquanto o número de clientes é fixado em 101.

A Tabela 7 apresenta as instâncias investigadas do grupo G4. Neste grupo, 5 instâncias foram utilizadas para investigar a escalabilidade da heurística VNS-PRV e seu comportamento com um grande número de clientes e veículos.

Tabela 6: Descrição das instâncias dos grupos G1, G2 e G3

Grupo	Número de Clientes	Número de Veículos
G1	[32, 33, 34, 36, 37, 38, 39, 45, 51]	5
G2	[16, 23, 50, 62, 76, 101]	8
G3	101	[4, 8, 10, 14]

Tabela 7: Descrição das instância do grupo G4

Número de Clientes	Número de Veículos
101	10

121	7
151	12
200	16
200	17

5 Resultados Numéricos

Este capítulo busca discutir os resultados alcançados pelos experimentos descritos no capítulo anterior.

Os experimentos foram realizados usando um computador com processador Ryzen 3200G - 3.60 GHz com quatro núcleos e 16GB de memória RAM em um sistema operacional Windows 11 Pro. A heurística foi implementada em Python 3.8.10.

A Tabela 8 apresenta os resultados dos grupos G1-G3. O G1, que buscava validar a eficiência do algoritmo para casos com uma pequena diferença na quantidade de clientes e o mesmo número de veículos, apresentou bons resultados com uma diferença de aproximadamente 7% dos resultados ótimos das instâncias mantendo um tempo de execução razoavelmente baixo para o tamanho das instâncias.

O G2, validou a eficiência do algoritmo para casos com uma diferença maior na quantidade de clientes e com o mesmo número de veículos, este grupo também apresentou bons resultados com uma diferença também de aproximadamente 7% dos resultados ótimos, mantendo um tempo de execução razoavelmente baixo para o tamanho das instâncias. Os testes começaram com uma instância de 16 clientes e foi até uma com 101, isto mostra que a heurística pode dar suporte a uma frota de tamanho considerável e atender uma grande quantidade de clientes.

O G3 diferentemente do G1 e G2, buscava avaliar casos com o mesmo número de clientes, mas com diferentes frotas de veículos. Os testes começaram com 4 veículos e foram até 14 veículos. A instância P-n101-k4 apresentou algumas limitações da heurística, em um cenário com 101 clientes e 4 veículos a heurística levou aproximadamente 30 minutos para retornar o resultado encontrado. Isso ocorreu devido ao uso da vizinhança V4 (Realocação de Cliente entre Veículos). Entretanto, a heurística encontrou um resultado apenas 3% mais custoso que o custo ótimo. No geral este grupo também apresentou bons resultados com uma diferença média de aproximadamente 6% dos resultados ótimos das instâncias.

Por fim, o G4, que buscou validar a escalabilidade do VNS-PRV para instâncias com muitos clientes e veículos, também obteve resultados satisfatórios, com uma diferença entre o custo encontrado e o custo ótimo de 13%. Os resultados foram um pouco maior que os outros grupos, contudo ainda assim baixo e com o tempo de execução razoavelmente aplicável. Esses resultados podem ser vistos na Tabela 9.

Estes resultados mostram que a heurística proposta lidou bem com diversos cenários comuns do roteamento de veículos e que pode ser aplicado para casos reais e melhorar os custos de uma frota real.

Tabela 8: Resultados para os grupos G1, G2 e G3.

Grupo	Instância	Custo Ótimo	Custo Encontrado	Tempo de execução (seg)
G1	A-n32-k5	784	836,06	8,36
G1	A-n33-k5	661	676,1	7,08
G1	A-n34-k5	778	796,16	7,46
G1	A-n36-k5	799	875,17	10,48
G1	A-n37-k5	669	734,44	15,55
G1	A-n38-k5	730	763,95	11,32
G1	A-n39-k5	822	865,64	13,3
G1	B-n45-k5	751	842,66	22,71
G1	E-n51-k5	521	577,3	37,25
G2	A-n62-k8	1288	1410,94	23,85
G2	E-n101-k8	817	870,79	133,08
G2	E-n76-k8	735	748,32	40,46
G2	P-n16-k8	450	452,94	0,78
G2	P-n23-k8	529	537,55	1,71
G2	P-n50-k8	631	778,65	9,17
G3	E-n101-k14	1071	1166,02	35,01
G3	E-n101-k8	817	863,95	112,37
G3	M-n101-k10	820	883,24	64,71
G3	P-n101-k4	681	700,82	1933,38

Tabela 9: Resultados para os grupos G4.

Grupo	Instância	Custo Ótimo	Custo Encontrado	Tempo em segundos
G4	M-n101-k10	820	926,4	61,17
G4	M-n121-k7	1034	1168,67	264,79
G4	M-n151-k12	1053	1152,71	148,32
G4	M-n200-k16	1274	1623,36	175,09
G4	M-n200-k17	1373	1425,96	180,07

6 Conclusões

Este trabalho tratou um problema de otimização bastante conhecido na literatura, o Problema do Roteamento de Veículos Capacitados. Esse problema possui inúmeras aplicação em logística de transporte, tendo assim uma gama muito grande de variações com diferentes restrições.

Para a obtenção de soluções de boa qualidade para o problema, foi desenvolvida uma heurística baseada no método de busca de vizinhança variável, chamada de VNS-PRV, onde foram implementadas 5 vizinhanças buscando explorar diversos cenários para melhorar as rotas dos veículos. As vizinhanças desenvolvidas buscam melhorar a rota de um veículo, através de mudanças do sequenciamento do atendimento aos clientes, melhorar as rotas através da troca de clientes entre os veículos, e trocas aleatórias. Das 5 vizinhanças, 4 utilizam estratégias determinísticas buscando uma decida até um ótimo local e uma estratégia usando busca aleatória, para tentar escapar de um ótimo local e, posteriormente, encontrar o ótimo global.

Para avaliarmos a eficiência da heurística, analisamos seus resultados utilizando instâncias encontradas na literatura.

Os resultados mostraram que essas simples 5 vizinhanças foram suficientes para encontrar boas soluções, com os custos apenas 8% maiores que os custos ótimos das instâncias na média. O tempo de execução da heurística também se mostrou bastante razoável para o tamanho das instâncias. Uma limitação encontrada foi a de instâncias com muitos clientes e poucos veículos. Para esses casos o tempo de execução foi significativamente mais alto que a média, entretanto os custos para esses casos se mantiveram bem próximo do custo ótimo.

No geral os resultados mostram que a heurística obteve soluções bastante satisfatórias. Contudo, considerando o espaço que ainda há para melhoria das soluções, em trabalhos futuros buscaremos desenvolver uma heurística híbrida para iniciar o processo de busca por soluções de melhor qualidade.

Agradecimento

Os autores agradecem ao apoio da FAPES e CAPES (processo 2021- 2S6CD, nº FAPES 132/2021) por meio do PDPG (Programa de Desenvolvimento da Pós-Graduação, Parcerias Estratégicas nos Estados).

Referências

[1] Marcos Arenales, et. al., Pesquisa operacional . Ed. Campus, 2006.
[2] Gilbert Laporte, The vehicle routing problem: An overview of exact and approximate algorithms, European Journal of Operational Research, Volume 59, Issue 3, Pages 345-358, 1992.
[3] Wujun Cao e Wenshui Yang. A Survey of Vehicle Routing Problem. MATEC Web of Conf., 2017. Doi: 100. 01006. 10.1051/matecconf/201710001006.
[4] Pierre Hansen, et. al., Variable Neighborhood Search, Handbook of Metaheuristics, Springer International Publishing, pp 57-97, 2019.
[5] Kris Braekers, Katrien Ramaekers e Inneke Van Nieuwenhuyse, The vehicle routing problem: State of the art classification and review, Computers & Industrial Engineering, Vol. 99, pp. 300-316, 2016.
[6] Kır, S., Yazgan, H.R. e Tüncel, E. A novel heuristic algorithm for capacitated vehicle routing problem. *J Ind Eng Int* **13**, 323–330 (2017).
[7] Dam, TL., Li, K. e Fournier-Viger, P. Chemical reaction optimization with unified tabu search for the vehicle routing problem. *Soft Comput* **21**, 6421–6433 (2017).
[8] Mouaouia Cherif Bouzid, Hacene Aït Haddadene e Said Salhi, An integration of Lagrangian split and VNS: The case of the capacitated vehicle routing problem, Computers & Operations Research, pp 513-525, Vol. 78, 2017,
[9] Yunyun Niu, et al., A novel membrane algorithm for capacitated vehicle routing problem, Soft Computing, pp 471-482, vol. 19, 2015.
[10] Sener Akpinar, Hybrid large neighbourhood search algorithm for capacitated vehicle routing problem, Expert Systems with Applications, Volume 61, Pages 28-38, 2016.
[11] Wang, X, et. al., Novel Ant Colony Optimization Methods for Simplifying Solution Construction in Vehicle Routing Problems, IEEE Transactions on Intelligent Transportation Systems, Vol, 17, I. 11, 2016.
[12] Rajeev Goel e Maini Raman, A hybrid of ant colony and firefly algorithms (HAFA) for solving vehicle routing problems, Journal of Computational Science, vol 25, pp 28-37, 2018.
[13] Muhammad Luthfi Shahab, Daryono Budi Utomo, and Isa Irawan Mohammad, Decomposing and Solving Capacitated Vehicle Routing Problem (CRP) using Two-Step Genetic Algoritm(TSGA), Journal of Theoretical & Applied Information Technology, vol87, n.3, 2016.
[14] Arash Mazidi, Fakhrahmad Mostafa e Hadi Sadreddini Morammad, A meta-heuristic approach to CVRP problem: local search optimization based on GA and ant colony, pp 1-22, 2016.
[15] Akhand, M. A. H., Jannat Peya Zahrul e Kazuyuki Murase, Capacitated vehicle routing problem solving using adaptive sweep and velocity tentative PSO, international journal of advanced computer science and applications, v. 8, n. 12, 2017.
[16] Dhahri Amine, et. al., A VNS-based Heuristic for Solving the Vehicle Routing Problem with Time Windows and Vehicle Preventive Maintenance Constraints, Procedia Computer Science, Vol. 80, pp. 1212-1222, 2016.
[17] CVRPLIB, http://vrp.galgos.inf.puc-rio.br/index.php/en/, Acessado em 31/10/2022.