ALGORITMO HÍBRIDO PARA RESOLUÇÃO DO PROBLEMA DE ROTEAMENTO DE VEÍCULOS CAPACITADO

Otávio Martins Vasconcelos, Edgar Ancioto
Instituto Federal de Minas Gerais – Campus Congonhas
Av. Michel Pereira de Souza, 3007 - Campinho, Congonhas – MG
Otavio_vasconcelos@outlook.com, junior.ancioto@gmail.com

Luiza Real
Instituto Federal de Minas Gerais – Campus Congonhas
Av. Michel Pereira de Souza, 3007 - Campinho, Congonhas – MG
Luiza.real@ifmg.edu.br

RESUMO

O Problema do Roteamento de Veículos é um dos problemas clássicos da otimização e tem grande relevância na indústria. Este problema é classificado na literatura como NP-Difícil e as formas de resolvê-lo estão em constante evolução. Dessa forma, este trabalho propõe um algoritmo genético híbrido com o *Simulated Annealing* para resolução da derivação capacitada do problema do roteamento de veículos. Para avaliar a viabilidade da hibridização proposta, 72 instâncias da literatura foram testadas e os resultados foram comparados com o melhor valor conhecido. O experimento computacional mostrou que a hibridização conseguiu resultados próximos às melhores soluções conhecidas com tempo de execução razoável.

PALAVRAS CHAVE. Problema do Roteamento de Veículos, Algoritmos Híbridos, Metaheurísticas, Algoritmo Genético, *Simulated Annealing*.

Tópicos: MH – Metaheurística, OC – Otimização Combinatória, **L&T – Logística e Transporte**

ABSTRACT

The Vehicle Routing Problem is one of the classic optimization problems and has great relevance in the industry. This problem is NP-Hard and its solution methods are in constant evolution. Thus, this article proposes a hybrid genetic algorithm with Simulated Annealing to solve the capacitated version of the vehicle routing problem. To assess the feasibility of the proposed hybridization, 72 instances from the literature were simulated and the results were compared with the best-known value in the literature. The computational experiment showed that the hybridization achieved results very close to the best-known solutions with reasonable time.

KEYWORDS. Vehicle Routing Problem, Hybrid Algorithms, Metaheuristics, Genetic Algorithms, Simulated Annealing

Paper Topics: MH – Metaheuristic, CO – Combinatorial Optimization, **L&T – Logistics and Transport**

1. Introdução

O Problema de Roteamento de Veículos (PRV) é um dos problemas clássicos da otimização e apresenta grande relevância na indústria devido à sua grande aplicabilidade em casos reais. Segundo Bodin (1983), o PRV consiste em traçar um conjunto de rotas em que uma frota de veículos atenda um conjunto de clientes, satisfazendo algumas restrições. Segundo Simas e Gómez (2007), as rotas a serem traçadas devem, de alguma forma, considerar a minimização de custos, como: custo total do roteamento, número de veículos a serem utilizados, tempo total gasto, entre outros.

Devido à sua capacidade de otimizar rotas, o PRV também é amplamente utilizado no setor de logística, fundamental para toda a cadeia produtiva. De acordo com Ballou (2006), "A movimentação de cargas absorve de um a dois terços dos custos logísticos em uma empresa" e no Brasil o setor representa 6,6% do PIB nacional (ILOS, 2017).

O setor logístico também tem ganhado cada vez mais importância devido ao crescimento do e-commerce nos últimos anos. Fazer a entrega dos produtos num curto período de tempo é parte crucial na percepção de qualidade do serviço por parte do cliente. Dessa forma, os meios de otimizar tanto custo quanto qualidade, como o PRV é capaz, devem ser explorados ao máximo.

Conforme destacam Kramer, Subramanian e Penna (2016), o PRV é considerado NP-difícil. Isso significa que ainda não foi desenvolvido um algoritmo de tempo polinomial para o problema. Dessa forma, é fundamental que melhorias nos modelos de resolução do problema estejam em constante evolução, para atender em tempo hábil as exigências de problemas cada vez maiores.

Dentre as várias formas de se resolver o PRV, destacam-se as metaheurísticas, que podem ser entendidas como métodos de solução que coordenam procedimentos de busca locais com estratégias de mais alto nível, de modo a criar um processo capaz de escapar de mínimos locais e realizar uma busca robusta no espaço de soluções de um problema (GLOVER, F.; KOCHENBERGER, G. A., 2003).

Nesse contexto, este trabalho desenvolve um algoritmo híbrido baseado na teoria de evolução de Charles Darwin (algoritmo genético) e na metaheurística *Simulated Annealing* (SA), para resolver a variação capacitada do problema de roteamento de veículos com frota homogênea. Portanto, o objetivo deste estudo é: (i) propor a aplicação híbrida entre um algoritmo genético e uma metaheurística bio-inspirada; e (ii) comparar os resultados desse algoritmo híbrido com os melhores valores da literatura em instâncias de tamanhos variados.

Este artigo está dividido da seguinte forma: na segunda seção, é apresentado o referencial teórico que foi utilizado como base para o entendimento do problema do roteamento de veículos e para o desenvolvimento do algoritmo híbrido, a terceira seção traz uma revisão bibliométrica do tema e as explicações acerca do algoritmo proposto no trabalho, a quarta seção apresenta os experimentos computacionais de avaliação do algoritmo em relação às instâncias da literatura. A conclusão é apresentada na sexta seção.

2. Referencial Teórico

2.1 Heurísticas

As heurísticas, segundo Souza (2022), podem ser definidas como sendo uma técnica inspirada em processos intuitivos que procuram uma boa solução a um custo computacional aceitável, sem, no entanto, estar capacitada a garantir sua otimalidade, bem como garantir quão próximo ela está da solução ótima. Portanto, o objetivo de se aplicar uma heurística em Problemas NP-difíceis, como o PRV, é obter uma boa resposta em um tempo razoável, permitindo que essa boa resposta seja aplicada em um contexto.

As heurísticas que resolvem o PRV podem ser divididas em heurísticas construtivas e heurísticas de melhoria. As heurísticas construtivas começam sem nenhuma rota pré-estabelecida de forma a desenvolver a rota ponto a ponto, adicionando o novo nó de acordo com regras preestabelecidas. Um exemplo de heurísticas de construção é a inserção do vizinho mais próximo, onde a cada iteração é inserido na rota o ponto com a menor distância do nó atual que não foi

adicionado ainda. Já as heurísticas de melhoria começam a partir de uma rota gerada aleatoriamente que vai sendo melhorada através de trocas entre os pontos da solução inicial. Um exemplo de heurística de melhoria é a 2-opt, que foi aplicada neste trabalho como método de busca local do SA.

2.2 Metaheurísticas

As metaheurísticas podem ser entendidas como métodos de solução que coordenam procedimentos de busca locais com estratégias de mais alto nível, de modo a criar um processo capaz de escapar de ótimos locais e realizar uma busca robusta no espaço de soluções de um problema (GLOVER, F. e KOCHENBERGER, G. A. ,2003)

Dentre as metaheurísticas que surgiram ao longo das últimas décadas, destacam-se: Algoritmos Genéticos (GAs), *Simulated Annealing* (SA), Busca Tabu (BT, *Tabu Search*), *Greedy Randomized Adaptive Search Procedures* (GRASP), Busca Local Iterada (ILS, *Iterated Local Search*), Busca em Vizinhança Variável (VNS, *Variable Neighborhood Search*), *Late Acceptance Hill-Climbing* (LAHC), Colônia de Formigas (ACO, *Ant Colony Optimization*), Busca Dispersa (SS, *Scatter Search*) e Otimização por Nuvem de Partículas (DPSO, *Discrete Particle Swarm Optimizatio*n) (SOUZA, 2022).

2.3 Algoritmo Genético

Os algoritmos genéticos fazem parte da computação evolutiva, que se baseia em mecanismos encontrados na natureza (ARTERO, 2008) e também nas teorias de Darwin em relação a reprodução e sobrevivência dos indivíduos de uma população. Para entender o funcionamento do algoritmo genético, é necessário entender algumas definições: o tamanho da população indica a quantidade de indivíduos (cromossomos); gerações é o máximo de iterações que o algoritmo simula; cromossomo é uma solução viável do problema simulado; cruzamento é o procedimento que gera um novo cromossomo; mutação é o processo de mudança das características de um cromossomo; aptidão é a qualidade do cromossomo; elitismo é o processo de manutenção dos indivíduos com melhores aptidões na população e seleção é a retirada dos elementos com as piores aptidões da população. A Figura 3 traz o pseudocódigo do GA.

```
procedimento GA (geracoes, novos_individuos, tamanho_populacao, sobreviventes)
Saída: Individuo_mais_apto

inicio
        populacao <- gerador_populacao(tamanho_populacao)
        contador_geracoes <- 0
        enquanto (contador_geracoes < geracoes) faça:
                calcula_aptidao(populacao)
                cruzamento(populacao)
                mutacao(novos_individuos, populacao)
                elitismo()
                selecao_individuos_mais_aptos(sobreviventes)
                populacao <- gerador_populacao(tamanho_populacao)
                contador_geracoes <- contador_geracoes + 1
        fim-enquanto
fim
```

Figura 3- Funcionamento do Algoritmo Genético

Os dados de entrada do GA são o número de gerações, a quantidade de novos indivíduos, o tamanho da população e sobreviventes. A população inicial é gerada e o contador que indica a geração atual é definido como zero. É inicializado também o critério de parada que define a quantidade de evoluções realizadas na população. Calcula-se o valor da aptidão para cada indivíduo da população. Ocorre então o processo de cruzamento e a mutação desses indivíduos para gerar novos membros para a população aproveitando as características das gerações anteriores. Passa-se para a etapa de elitismo e seleção dos indivíduos mais aptos daquela geração para serem mantidos na população, enquanto os menos aptos são eliminados. São criados então novos cromossomos

para garantir que a população mantenha o tamanho original e o contador de gerações é incrementado. Esse processo se repete até que o contador de gerações atinja critério de parada.

Devido à grande aplicabilidade do algoritmo genético, várias formas de melhorar os resultados já foram desenvolvidas. Uma delas é a criação do indivíduo especialista, que consiste em inserir na população inicial um indivíduo que é gerado através de alguma heurística que não impacte de maneira relevante o tempo de execução e segundo Ancioto (2019), possibilita que em meio a uma população totalmente probabilística, exista um indivíduo com conhecimento especialista e melhore ainda mais os resultados.

2.4 Simualated Annealing

O SA é um algoritmo de busca local proposto em 1983 por Kirkpatrick (SOUSA JÚNIOR, 2007). Segundo Ancioto (2019), computacionalmente, o SA pode ser visto como um processo de determinação estocástica da organização dos átomos de um sólido, buscando a energia mínima. Isso torna o algoritmo bastante útil para encontrar ótimos locais em problemas de otimização combinatória. O pseudocódigo da Figura 4 indica o funcionamento do algoritmo:

```
procedimento SA(f(.),N(.),α,SAmax,T0,s)
1  s* ← s;                    {Melhor solução obtida até então}
2  IterT ← 0;                 {Número de iterações na temperatura T}
3  T ← T0;                    {Temperatura corrente}
4  enquanto (T > 0) faça
5      enquanto (IterT < SAmax) faça
6          IterT ← IterT + 1;
7          Gere um vizinho qualquer s' ∈ N(s);
8          Δ = f(s') − f(s);
9          se (Δ < 0)
10             então
11                 s ← s';
12                 se (f(s') < f(s*)) então s* ← s';
13             senão
14                 Tome x ∈ [0,1];
15                 se (x < e^(−Δ/T)) então s ← s';
16         fim-se;
17     fim-enquanto;
18     T ← α × T;
19     IterT ← 0;
20 fim-enquanto;
21 s ← s*;
22 Retorne s;
fim SA;
```

Figura 4 - Algoritmo Simulated Annealing, Fonte: Souza (2022)

Os dados de entrada do algoritmo são dados pelos parâmetros temperatura inicial, coeficiente alfa de redução de temperatura e o número máximo de iterações; e, pelo conjunto de solução inicial. A solução inicial é definida como melhor solução. São inicializados os critérios de parada que garantem a temperatura maior que zero e o número de repetições menor que o parâmetro de entrada. Ocorre então a perturbação na solução inicial, gerando outra solução. Calcula-se o valor da energia de ambas as soluções, que no caso do PRV é a distância total percorrida na rota, e as compara. Se a energia do indivíduo perturbado for menor que a do indivíduo original, esta solução passa a ser a atual. Contudo, se a energia da solução perturbada for maior, um número aleatório de 0 a 1 é gerado e comparado ao resultado da equação indicada na linha 15 do pseudocódigo. Se o número aleatório for menor que o resultado da equação, a solução perturbada com maior energia é aceita como solução atual. O processo se repete até que o contador atinja a condição de parada. Em seguida, ocorre a redução da temperatura através da multiplicação do seu valor atual pelo coeficiente alfa e o contador de repetições é zerado. O algoritmo irá se repetir enquanto a temperatura for maior que zero.

2.5 Algoritmos Híbridos

O aumento da complexidade dos problemas de otimização evidenciou que os métodos heurísticos sozinhos podem muitas vezes não trazer resultados satisfatórios. Então, essas soluções passaram a ser hibridizadas de forma a potencializar as características positivas e reduzir o efeito das características negativas das técnicas envolvidas e chegar a um resultado mais próximo da melhor solução. Segundo Raidl (2006), as pesquisas dessa área não têm foco na utilização de uma heurística específica, mas de heurísticas capazes de combinar suas estratégias com outras, deixando de seguir a filosofia de heurística "pura".

Em relação a classificação dos algoritmos híbridos, Talbi (2002) indica a existência de duas classes de hibridização: alto nível e baixo nível. Na primeira classe, os métodos heurísticos envolvidos são autônomos e um gera dados para o outro. No contexto do PRV, um exemplo de metaheurística de alto nível seria um algoritmo genético gerando a solução inicial para o SA. Na segunda classe, um método heurístico é utilizado por outro método durante o seu processo, criando uma relação de composição entre os envolvidos. No contexto do PRV seria, por exemplo, o SA sendo usado como método de mutação dos cromossomos num algoritmo genético.

3. Metodologia

3.1 Revisão Bibliométrica

Para entender quais os métodos de solução mais abordados para resolver o PRV dentre as várias técnicas possíveis, foi realizada uma análise bibliométrica dos artigos da base de artigos *Web Of Science*. Os filtros utilizados na busca avançada foram feitos através das palavras chave: "*vehicle routing problem*". Foram encontradas 8246 publicações e todas elas foram consideradas neste trabalho.

Através do software *VOSViewer*, foram criados mapas de redes bibliométricas com os metadados extraídos da base de artigos. A relação entre os assuntos é dada a partir da correlação entre os assuntos na base de dados, que é representada no diagrama pela distância entre as esferas. Dessa forma, quanto mais perto as esferas estiverem uma das outras, mais relacionados os temas estão na base de dados. Com isso, é possível inferir quais assuntos são mais associados a esfera do termo central da pesquisa, bem como a relação entre os próprios temas.

Outros dois fatores a serem considerados são as cores do diagrama, que são definidas através de uma clusterização realizada pelo software e que reforça a relação entre os termos; e o tamanho de cada texto e esfera, sendo que o tamanho de ambos é diretamente proporcional a quantidade de vezes que o termo aparece na base de dados avaliada. A Figura 5 traz o resultado da análise.

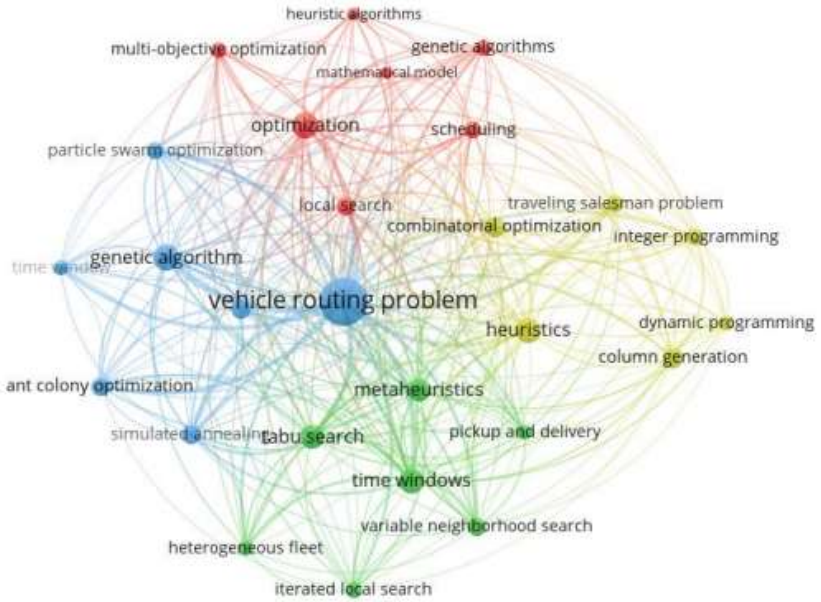

Figura 5 – Diagrama Customizado do VRP

Fica evidente que os métodos de solução mais utilizados estão relacionados com metaheurísticas bio-inspiradas, busca por tabu e algumas heurísticas. Entre as metaheurísticas, as principais citadas para resolver o PRV foram o algoritmo genético, otimização por exame de partículas, por colônia de formigas (bio-inspiradas) e o SA.

Em relação as variações do PRV, as mais presentes foram a *pick-up and delivery*, *time windows*. Devido à proximidade dessas palavras com o termo metaheurísticas, pode-se concluir que as publicações indicam que metaheurísticas são mais comumente usadas para resolver essas variações.

O *VOSViewer* também faz uma avaliação a respeito do ano de publicação mais comum em cada tema, como indicado na Figura 6.

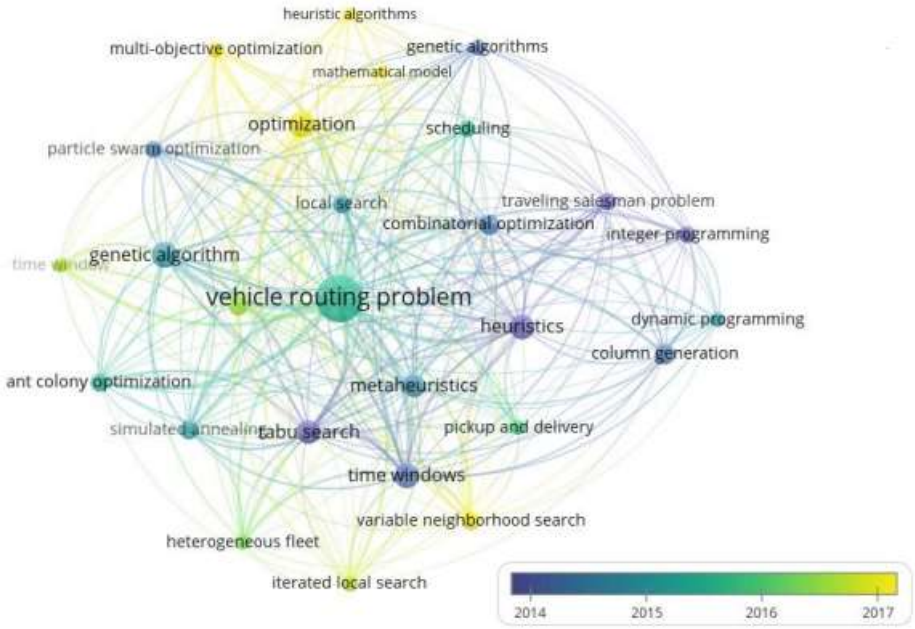

Figura 6 – Diagrama Temporal do VRP

As heurísticas foram mais publicadas num período próximo de 2014. Em seguida, as metaheurísticas foram mais exploradas e atualmente, estão sendo mais publicados soluções usando otimização multiobjetivos e a *Variable Neighbordhood Search*. Dessa forma, este trabalho se propõe a hibridizar dois algoritmos relevantes na literatura: Algoritmo Genético e SA.

3.2 Proposta de Algoritmo

O algoritmo proposto neste trabalho para resolver o PRV capacitado foi uma hibridização considerada de baixo nível, de acordo com a classificação de Talbi (2002), entre um Algoritmo Genético e o SA. O fluxograma exposto na Figura 7 indica o funcionamento do algoritmo:

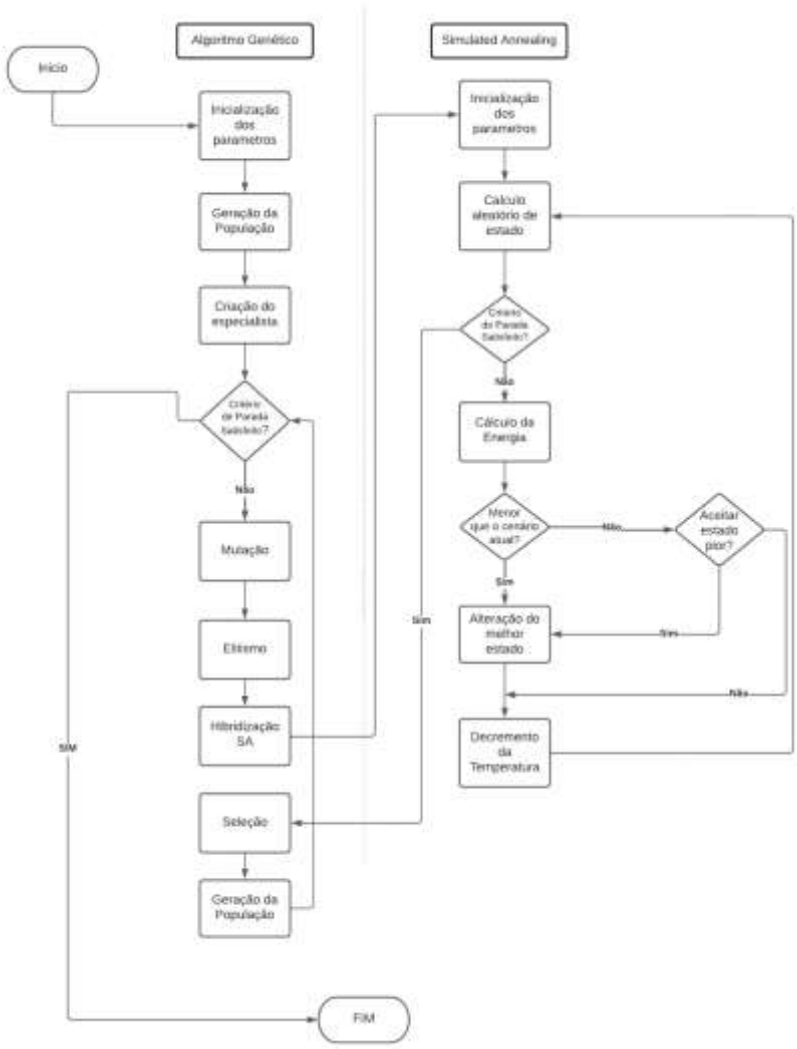

Figura 7 – Funcionamento do Algoritmo Proposto

O algoritmo começa com a inicialização dos parâmetros gerações, novos indivíduos, tamanho da população e sobreviventes do algoritmo genético e os parâmetros temperatura e alfa do SA, parâmetros esses explicados na Seção 2. Em seguida, é gerada a população de indivíduos (ou cromossomos) através de um embaralhamento aleatório da ordem dos nós do vetor base recebido na instância, sem considerar o nó do depósito. Esse procedimento é repetido para a criação de cada indivíduo da população até que o seu tamanho alcance o parâmetro de entrada tamanho da população. A Figura 8 exemplifica a criação aleatória dos indivíduos:

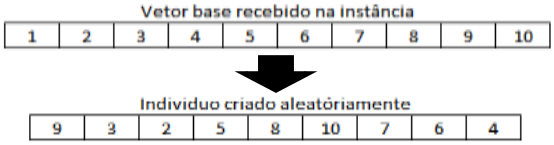

Figura 8 – Geração Aleatória da População

Após a criação da população inicial, ocorre a criação de um indivíduo especialista. Essa criação seleciona um nó aleatoriamente do vetor base, sem considerar o nó de depósito, e seleciona o próximo nó da rota através da heurística do vizinho mais próximo, até que toda a instância seja atendida. Ao final, esse especialista é inserido na população, que fica com um indivíduo a mais que o parâmetro tamanho da população.

Durante a criação de cada indivíduo, é feito também o cálculo da distância total percorrida pela rota (fitness). Esse cálculo do fitness divide todos os nós das instâncias em sub-rotas considerando a restrição de capacidade dos veículos e da demanda dos pontos e então, acrescenta ás distâncias dos primeiros e últimos pontos até o depósito. Um exemplo com 10 nós e 3 veículos de capacidade de 300 unidades é apresentado na Figura 9. O indivíduo 2-3-4-5-6-7-8-9-10 implica em 3 rotas: (i) 1-2-3-4-1; (ii) 1-5-6-7-8-1; e, (iii) 1-9-10-1. O fitness dessa solução é dado pela soma das distâncias percorridas em cada uma dessas rotas.

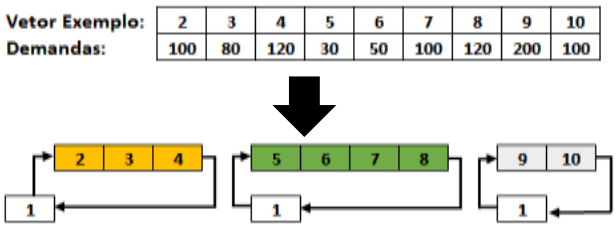

Figura 9 – Cálculo do Fitness

O contador de gerações é inicializado e então ocorre a mutação dos indivíduos. Esse processo seleciona um indivíduo aleatório da população e para cada indivíduo escolhido, dois nós também são selecionados aleatoriamente para realizar a troca 2-opt e dar origem a um novo indivíduo, que é inserido na população. A quantidade de mutações realizadas é definida através do parâmetro de entrada novos indivíduos e após todas as mutações realizadas o tamanho da população é dado pela soma dos parâmetros novos indivíduos e tamanho da população. A Figura 10 demonstra a mutação de um indivíduo pela heurística de troca 2-opt.

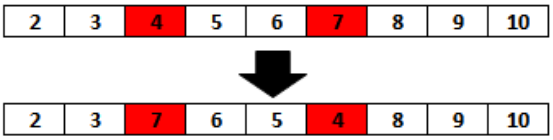

Figura 10 – Mutação 2-opt

Após a mutação, é realizado o elitismo através de uma ordenação da população para que os indivíduos com menor fitness, ou mais aptos, estejam no início dessa população e aqueles com maior fitness, ou menos aptos, estejam no fim. Na sequência, ocorre a hibridização com o SA, que recebe como solução inicial o indivíduo mais apto daquela geração e executa todas as etapas definidas na Figura 4, com a única diferença de que a única condição de parada utilizado no algoritmo proposto foi a temperatura negativa. Ao final do SA, um novo indivíduo é retornado e adicionado na população. Por fim, ocorre a seleção natural para retirar os indivíduos com maior fitness, deixando na população a quantidade de indivíduos indicada no parâmetro de entrada sobreviventes. Ao final da iteração são criados novos indivíduos com o mesmo método da geração da população inicial para completar o número de indivíduos da população original e o contador é atualizado. Esse processo se repete até o contador alcançar o número de gerações e por fim, o algoritmo retorna a melhor solução e o seu valor de fitness.

4 Experimentos Computacionais

O algoritmo híbrido foi implementado na linguagem de programação Python versão 3.9. Os testes computacionais foram executados em um computador que possui o Sistema Operacional Windows 10 – 64 bits com a CPU Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz 2.30 GHz .

Todos os testes foram feitos com os algoritmos recebendo os mesmos parâmetros de calibração, exibidos no Quadro 1.

Algoritmo	Parâmetros	Valor Utilizado
Simulated Annealing	Temperatura	1000
	Alfa	0,9
Algoritmo Genético	Gerações	1000
	Novos Indivíduos	320
	Tamanho da População	100
	Sobreviventes	50

Quadro 1- Parâmetros de calibração do algoritmo

Todas as instâncias testadas foram retiradas da biblioteca CVRPLIB, que é um repositório online amplamente utilizado que contém instâncias de problemas e conjuntos de dados relacionados ao PRV, além dos melhores resultados da literatura para essas instâncias. A biblioteca pode ser acessada pelo link http://vrp.galgos.inf.puc-rio.br/index.php/en/. Dessa forma, o ótimo global considerado na avaliação do algoritmo proposto são aqueles indicados pela biblioteca.

Para quantificar essa avaliação foi utilizado o percentual de diferença M = 100% x (Dr – Ds) / Dr, sendo Dr o melhor resultado de acordo com a biblioteca CVRPLIB e Ds o resultado obtido pelo algoritmo.

A Tabela 1 apresenta os resultados obtidos pelo algoritmo e a comparação com o melhor resultado conhecido. A primeira coluna traz o nome da instância avaliada, onde ao lado direito da letra "n" é informada a quantidade de nós e ao lado da letra "k" é informado o número de veículos disponíveis. Por exemplo, a instância “P-n23-k8” considera 23 nós e 8 veículos. Na segunda coluna, é apresentado o melhor resultado de acordo com a biblioteca CVRPLIB. Na terceira coluna, é informado o resultado obtido pelo algoritmo. Já a quarta coluna apresenta o tempo de execução do algoritmo em segundos. Por fim, a quinta coluna calcula o percentual de diferença em relação ao ótimo de acordo com a Equação 1.

Tabela 1: Resultados do Algoritmo

Instância	Melhor Resultado (Km)	Resultado Algoritmo (Km)	Tempo(s)	GAP(%)
P-n16-k8	450	450	6,10	0%
P-n19-k2	212	212	6,08	0%
P-n20-k2	216	217	6,65	-1%
P-n21-k2	211	213	7,22	-1%
E-n22-k4	375	377	6,51	0%
P-n22-k2	216	218	7,14	-1%
P-n22-k8	603	603	8,36	0%
E-n23-k3	569	569	6,83	0%
P-n23-k8	529	530	9,04	0%
E-n30-k3	534	535	9,19	0%
A-n32-k5	784	808	9,90	-3%
A-n33-k5	661	662	11,71	0%
A-n33-k6	742	746	10,49	-1%
E-n33-k4	835	840	12,20	-1%
A-n34-k5	778	781	12,31	0%
A-n36-k5	799	816	10,01	-2%
A-n37-k5	669	690	15,83	-3%
A-n37-k6	949	990	15,90	-4%
A-n38-k5	730	735	13,53	-1%
A-n39-k5	822	847	17,59	-3%
A-n39-k6	831	843	17,65	-2%
P-n40-k5	458	475	17,33	-4%
A-n44-k6	937	964	18,55	-3%
A-n45-k6	944	976	21,71	-3%
A-n45-k7	1146	1181	19,76	-3%
F-n45-k4	724	728	23,76	-1%
P-n45-k5	510	530	25,07	-4%

Tabela 1: Resultados do Algoritmo (Continuação)

Instância	Melhor Resultado (Km)	Resultado Algoritmo (Km)	Tempo(s)	GAP(%)
A-n46-k7	914	925	22,13	-1%
A-n48-k7	1073	1118	27,82	-4%
P-n50-k10	696	717	35,11	-3%
P-n50-k7	554	583	30,75	-5%
P-n50-k8	631	650	38,93	-3%
E-n51-k5	521	543	34,71	-4%
P-n51-k10	741	753	29,18	-2%
A-n53-k7	1010	1032	29,61	-2%
A-n54-k7	1167	1191	35,76	-2%
A-n55-k9	1073	1085	24,65	-1%
P-n55-k10	694	710	22,65	-2%
P-n55-k15	989	1003	26,85	-1%
P-n55-k7	568	574	23,76	-1%
P-n55-k8	588	595	19,85	-1%
B-n57-k7	1153	1159	28,16	-1%
A-n60-k9	1354	1422	20,10	-5%
P-n60-k10	744	773	28,26	-4%
P-n60-k15	968	992	27,79	-3%
A-n61-k9	1034	1053	26,05	-2%
A-n62-k8	1288	1320	21,73	-2%
A-n63-k10	1314	1346	26,51	-2%
A-n63-k9	1616	1644	29,75	-2%
A-n64-k9	1401	1441	22,96	-3%
A-n65-k9	1174	1209	22,84	-3%
P-n65-k10	792	818	30,94	-3%
A-n69-k9	1159	1192	31,64	-3%
P-n70-k10	827	858	48,35	-4%
F-n72-k4	237	251	32,42	-6%
E-n76-k10	830	887	47,35	-7%
E-n76-k14	1021	1066	30,60	-4%
E-n76-k7	682	730	35,09	-7%
E-n76-k8	735	779	46,90	-6%
P-n76-k4	593	620	51,88	-5%
P-n76-k5	627	670	43,17	-7%
A-n80-k10	1763	1828	42,98	-4%
E-n101-k14	1071	1142	87,51	-7%
E-n101-k8	817	869	83,71	-6%
E-n101-k14	1071	1145	95,97	-7%
E-n101-k8	817	872	61,17	-7%
M-n101-k10	820	867	89,82	-6%
P-n101-k4	681	727	72,17	-7%
M-n121-k7	1034	1059	113,19	-2%
F-n135-k7	1162	1214	110,89	-5%
M-n151-k12	1053	1122	162,06	-7%
M-n200-k17	1373	1444	212,75	-5%

Tabela 1: Comparativo entre resultados do algoritmo proposto e os melhores valores da literatura

Pode-se observar pela Tabela 1 que o algoritmo conseguiu atingir o valor ótimo para instâncias até 23 nós, com tempo de execução menor que 10 segundos. Para as outras instâncias, o algoritmo não alcançou a melhor solução, mas conseguiu valores próximos, sendo a pior diferença em 10% numa instância de 151 nós. Em relação ao tempo de execução, o algoritmo conseguiu encontrar resultados em grandes instâncias com, no máximo, 3,5 minutos. Em média o GAP alcançado e o tempo de resolução ficaram em -3% e 25,94 segundos, respectivamente.

4 Conclusão

O PRV tem grande aplicação em diversos setores do cotidiano e sua relevância tem se tornado cada vez maior por conta da capacidade de se adequar a problemas reais com o acréscimo de poucas restrições e também pelo aumento da importância das aplicações nos setores que podem ser otimizados com esse problema, como a logística.

Contudo, o PRV e suas derivações são classificados como NP-difícil e por isso, são estudados a cada dia formas de se obter boas soluções no menor tempo de execução possível. Dentre essas formas se destacam os algoritmos heurísticos. Todavia, com o aumento da complexidade das aplicações, os algoritmos heurísticos sozinhos não são capazes de satisfazer as soluções e por isso, o estudo da hibridização desses métodos tem sido a forma estudada para atender as maiores demandas.

Desta forma, este trabalho teve como objetivo apresentar uma hibridização de baixo nível desenvolvida na linguagem de programação Python de um algoritmo genético e o SA sendo usado para fazer uma mutação no melhor cromossomo de cada geração. Para validar a eficácia dessa configuração de hibridização, os resultados do algoritmo foram comparados com a melhor solução de instâncias do site CVRPLIB. O algoritmo híbrido desenvolvido apresentou resultados próximos ao ótimo conhecido em um tempo de execução aceitável em todos os tamanhos de instâncias avaliados.

Portanto, é possível concluir que, com às configurações utilizadas, a hibridização proposta não se mostrou eficaz em relação a encontrar o valor ótimo conhecido, mas conseguiu encontrar soluções próximas ao ótimo num curto tempo de execução.

Para trabalhos futuros, é interessante realizar uma análise estatística para definir os melhores parâmetros de calibração de ambos os algoritmos e verificar se os resultados obtidos neste trabalho podem ser melhorados ou aplicar técnicas onde os próprios algoritmos ajustem os parâmetros durante a execução do código.

Referências

Ancioto Junior, Edgar Marcos. Plataforma Web Hybroo: ambiente experimental voltado à hibridização de algoritmos de otimização. 2019. 101 f. Dissertação (Programa de Pós-Graduação STRICTO SENSU em Engenharia de Produção e Sistemas) - Pontifícia Universidade Católica de Goiás, Goiânia.

ARTERO, A. O. Inteligência artificial teórica e prática. São Paulo: São Paulo, 2008

BALLOU, R. H. Gerenciamento da cadeia de suprimentos/logística empresarial. 5. ed. Porto Alegre: Bookman, 2006

BODIN, L. D. Routing and scheduling of vehicles and crews - The state of the art. [s.l.] Pergamon Press, 1983. v. 10

GLOVER, F. e KOCHENBERGER, G. A. (2003). Handbook of Metaheuristics. Kluwer Academic Publishers, Boston.

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by Simulated Annealing. Science, v. 220, n. 4598, p. 671–680, 1983.

KRAMER, R. H. F. R.; SUBRAMANIAN, A.; PENNA, P. H. V. Asymmetric vehicle routing problem with heterogeneous limited fleet: a case study in a beverage industry. Gestão & Produção, v. 23, n. 1, Jan./Mar.. 2016.

Lucas Daniel Padia Rocha; Flávio V. Cruzeiro Martins. Um Algoritmo Genético aplicado ao Problema de Roteamento de Veículos Capacitados. In: ANAIS DO SIMPóSIO BRASILEIRO DE PESQUISA OPERACIONAL, 2019, Limeira. Anais eletrônicos... Campinas, Galoá, 2019. Disponível em: <https://proceedings.science/sbpo/sbpo-2019/trabalhos/um-algoritmo-genetico-aplicado-ao-problema-de-roteamento-de-veiculos-capacitados?lang=pt-br> Acesso em: 10 abr. 2023.

OLIVEIRA, Marcelus Xavier et al. HEURÍSTICA GRASP APLICADA AO PROBLEMA DE ROTEAMENTO DE VEÍCULO COM BACKHAULS E FROTA HETEROGÊNEA FIXA. In: A PESQUISA OPERACIONAL NA BUSCA DE EFICIêNCIA NOS SERVIÇOS PðBLICOS E/OU PRIVADOS, 45., 2012, Natal. Simpósio Brasileiro de Pesquisa Operacional. Natal: Sbpo, 2012. p. 1642-1653. Disponível em: http://www.din.uem.br/sbpo/sbpo2013/pdf/arq0339.pdf. Acesso em: 24 maio 2023.

RAIDL, G. R. A Unified View on Hybrid Metaheuristics. European RTN ADONET, p. 1–12, 2006.

SIMAS, E. P. L.; GÓMEZ, A. T. Comparying a tabu search process. ICINCO 2007 - International Conference on Informatics in Control, Automation and Robotics, p. 77–84, 2007.

Souza, M. J. F. Inteligência Computacional para Otimização: metaheurísticas. Departamento de Computação, Universidade Federal de Ouro Preto, Ouro Preto, Minas Gerais, 2022. Disponível em http://www.decom.ufop.br/prof/marcone/Disciplinas/InteligenciaComputacional/InteligenciaComputacional.pdf

TALBI, E. G. A taxonomy of hybrid metaheuristics. Journal of Heuristics, v. 8, n. 5, p. 541–564, 2002.