

SOLUCIONANDO O PROBLEMA DE ROTEAMENTO DE VEÍCULOS CAPACITADO COM ALGORITMOS GENÉTICOS E PARALELISMO

Rodrigo Raider de Oliveira

Instituto Federal do Espírito Santo - Serra
Av. dos Sabiás, 330, Morada de Laranjeiras - Serra, ES
rodrigoraider@gmail.com

Leandro Colombi Resendo

Instituto Federal do Espírito Santo - Serra
Av. dos Sabiás, 330, Morada de Laranjeiras - Serra, ES
leandro@ifes.edu.br

RESUMO

Problemas de roteamento de veículos (VRP) são clássicos da otimização combinatória, altamente aplicáveis no mundo real, que lidam com a definição de rotas para clientes e demandas em situações tipicamente restritivas. Neste trabalho, propomos uma abordagem baseada em Algoritmos Genéticos para a resolução do CVRP, aprimorando-o com o uso do modelo de ilhas, explorando *multithreading* e memória compartilhada. Um algoritmo foi implementado e *benchmarks* serviram como base de testes para explorá-lo. Os resultados revelaram uma melhoria na qualidade das soluções e uma redução na variância à medida que mais ilhas foram adicionadas. O método proposto mostrou-se promissor para cenários onde as soluções convencionais encontram limitações. Ele se destaca como alternativa através da escalabilidade horizontal dos recursos computacionais.

PALAVRAS CHAVE. CVRP, Algoritmo Genético, Modelo de Ilhas.

Tópicos: MH – Metaheurísticas. OC – Otimização Combinatória. L&T – Logística e Transportes.

ABSTRACT

Vehicle routing problems (VRP) are classics of combinatorial optimization that are highly applicable in the real world, dealing with the definition of routes for customers and demands in typically restrictive situations. In this work, we propose an approach based on Genetic Algorithms for solving the CVRP, improving it by using the island model, exploiting multithreading and shared memory. An algorithm was implemented and benchmarks were used to explore it. The results showed an improvement in the quality of the solutions and a reduction in variance as more islands were added. The proposed method proved promising for scenarios where conventional solutions encounter limitations. It stands out as an alternative due to the horizontal scalability of computing resources.

KEYWORDS. CVRP, Genetic Algorithm, Island Model.

Paper topics: MH – Metaheuristics. OC - Combinatorial Optimization. L&T – Logistics and Transportation.

1. Introdução

Solucionar problemas de roteamento de veículos se tornou uma importante e extensa área de estudo na Pesquisa Operacional, originada da necessidade de superar desafios e tornar mais eficiente e econômica a cadeia de suprimentos. Esses desafios se acentuam com as rotinas cada vez mais aceleradas, integrando o fluxo de informações, produtos e serviços. A seleção da modalidade de transporte, bem como a roteirização e programação dos veículos, são algumas das principais atividades estratégicas nesse contexto [Ballou, 2006]. O Problema de Roteamento de Veículos Capacitado (CVRP - *Capacitated Vehicle Routing Problem*) é uma das variantes dessa classe de problemas abordada neste trabalho.

Classificado como NP-Difícil, a obtenção de soluções ótimas para o CVRP é limitada pelo tempo de processamento. As abordagens exatas mais típicas envolvem a programação dinâmica e busca em árvore, como: *branch-and-bound*, *branch-and-cut*, *branch-and-price* e *branch-price-and-cut* [Christofides et al., 1981; Augerat et al., 1995; Blasum e Hochstättler, 2004; Lysgaard et al., 2004]. Elas demonstram alto esforço no tratamento de versões quando se aproximam de uma centena de clientes, necessitando de muitas horas de processamento, o que dá espaço para abordagens aproximadas, como: Busca Tabu [Toth e Vigo, 2003], *Neighborhood Search* [Amous et al., 2017], Redes Neurais [Hottung e Tierney, 2019], Algoritmo do Vagalume [Altabeeb et al., 2019], além de diversas soluções híbridas que mesclam meta-heurísticas.

Por outra perspectiva, alguns estudos têm explorado a extensão do uso de *hardware* na resolução de problemas de natureza combinatória. Essa expansão envolve principalmente duas grandes vertentes: (a) delegar cálculos complexos e/ou (b) realizar busca local paralela. A primeira vertente (a) é notável pela distribuição do trabalho computacional intensivo, designando a múltiplos núcleos a tarefa de processar funções com alto custo, como no cálculo de funções de aptidão. A segunda vertente (b) é empregada em técnicas que permitem superar interdependências de dados ou funções, realizando buscas locais de forma paralela e independente [Crainic e Toulouse, 2003]. O modelo mestre-escravo e o modelo granular são as aplicações mais comuns, ambos pertencentes ao conceito de Modelo de Ilhas.

Explorado neste trabalho, o Modelo de Ilhas é uma alternativa promissora para contornar os desafios da complexidade computacional e da convergência prematura presentes em técnicas de busca em populações, como nos Algoritmos Genéticos [Cantú-Paz, 1998]. Suas principais características envolvem a utilização de múltiplas populações e a migração genética. Tal técnica permite que sistemas multiprocessados sejam utilizados para aumentar a capacidade computacional, com o uso de processadores multinúcleo, unidades de processamento gráfico (GPU) ou *clusters*.

O objetivo deste trabalho é apresentar um método de otimização para o CVRP que visa encontrar soluções de boa qualidade com o recurso de paralelização, voltado para a obtenção de resultados em um tempo computacional aplicável. Para isso, propomos o uso da meta-heurística Algoritmo Genético (AG), explorando o modelo de ilhas, com o objetivo de utilizar o paralelismo como uma alternativa para ampliar o espaço de busca. Testes com instâncias clássicas da literatura demonstraram que o acréscimo de ilhas aumentou consideravelmente a eficiência dos resultados obtidos pelo AG, com a contrapartida de um pequeno aumento no tempo de processamento.

O restante deste artigo está organizado da seguinte forma: na Seção 2, são apresentados os principais conceitos necessários para o entendimento do CVRP; na Seção 3, é detalhada a heurística proposta e a estratégia de paralelização; na Seção 4, são descritos o projeto de experimentos, a calibração da heurística, os resultados e sua análise. Finalmente, na Seção 5, são apresentadas as conclusões e os trabalhos futuros.

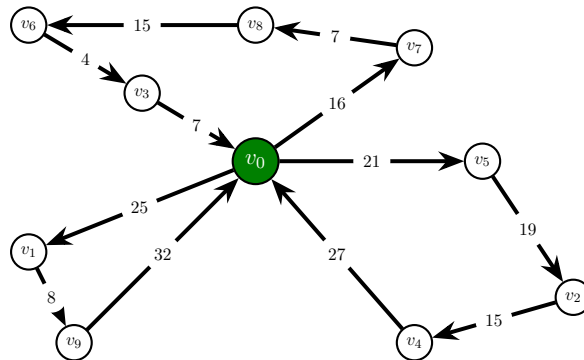
2. Descrição do Problema

Formalmente, o CVRP é descrito por Toth e Vigo [2002] através de uma representação como um grafo completo não direcionado $G = (V, A)$, onde cada vértice (V) representa um destino, e cada aresta (A) o caminho entre eles. Os vértices $i = \{1, \dots, n\}$ correspondem aos clientes, e o vértice 0 ($i = 0$) representa um depósito. As arestas (A) definem um grafo fortemente conexo, geralmente completo, onde todos os vértices são adjacentes aos outros.

Uma função de custo não negativo $c_{i,j} : A \rightarrow \mathbb{Z}^+$ é associada a cada arco de G e uma função de demanda $d : V \rightarrow \mathbb{Z}^+$ é associada a seus vértices. O depósito possui uma demanda $d_0 = 0$. Ciclos (*loops*) são proibidos, pois o veículo não deve passar duas vezes pelo mesmo cliente.

Uma frota de veículos homogênea que possui capacidade Q é inicializada no depósito (v_0). O objetivo é encontrar um conjunto de rotas com custo mínimo tal que: (I) Todas as rotas começam e terminam no depósito; (II) Todas as demandas dos clientes são atendidas exatamente uma vez por um único veículo; (III) Todos os $|K|$ veículos são utilizados, cada um associado a apenas uma rota; (IV) A capacidade de cada veículo não é excedida. A Figura 1 ilustra um exemplo com 3 veículos, sendo dois deles atendendo 3 clientes e um atendendo 4 clientes. Também é apresentado o custo das arestas.

Figura 1: Exemplo do grafo que representa um roteamento aplicado ao CVRP.



Segundo [Ralphs et al., 2003], a solução do CVRP está associada ao relaxamento de dois problemas combinatórios difíceis, o Problema do Caixeiro Viajante Múltiplo (PCVM) e o Problema do Empacotamento (BPP). No relaxamento relacionado ao PCVM, cria-se instâncias virtuais do depósito não conectados para cada veículo extra ($k - 1$), convertendo-a em uma instância do PCV. Para saber se uma instância respeita as restrições de carga, existe o tratamento de uma instância do BPP. A solução do CVRP é obtida a partir da solução do PCV que satisfaz as restrições de carga do BPP de cada k segmento das rotas obtidas.

O modelo de programação linear inteira proposto por [Borcinova, 2017] pode ser usado como formalização do problema:

$$\text{Minimizar} \quad \sum_{r=1}^p \sum_{i=0}^n \sum_{j=0, i \neq j}^n c_{ij} x_{rij} \quad (1)$$

Sujeito a

$$\sum_{r=1}^p \sum_{i=0, i \neq j}^n x_{rij} = 1, \quad \forall j \in \{1, \dots, n\} \quad (2)$$

$$\sum_{j=1}^n x_{r0j} = 1, \quad \forall r \in \{1, \dots, p\} \quad (3)$$

$$\sum_{i=0, i \neq j}^n x_{rij} = \sum_{i=0}^n x_{rji}, \quad \forall j \in \{0, \dots, n\}, r \in \{1, \dots, p\} \quad (4)$$

$$\sum_{i=0}^n \sum_{j=1, i \neq j}^n d_j x_{rij} \leq Q, \quad \forall r \in \{1, \dots, p\} \quad (5)$$

$$\sum_{r=1}^n \sum_{i \in S} \sum_{j \in S, i \neq j} x_{rij} \leq |S| - 1, \quad \forall S \subseteq \{1, \dots, n\} \quad (6)$$

$$x_{rij} \in 0, 1, \quad \forall r \in \{1, \dots, p\}, i, j \in \{1, \dots, n\}, i \neq j \quad (7)$$

A Expressão (1) representa a função objetivo, que visa a minimização do custo total das rotas. A Equação (2) garante que cada cliente seja visitado por exatamente um veículo. As equações (3) e (4) garantem que cada veículo deixe o depósito uma única vez, e que a quantidade de veículos que deixa depósito e visita os clientes devem ser igual ao número de veículos que voltam. A Expressão (5) garante que a capacidade dos veículos sejam respeitadas. A Expressão (6) garante que não haverá sub-rotas na solução. Por fim, a Expressão (7) determina o domínio das variáveis.

3. Heurística Proposta

Nesse trabalho é proposto um Algoritmo Genético paralelo para o CVRP. Utilizando o conceito de ilhas na paralelização do Algoritmo Genético, descrito a seguir, este algoritmo foi projetado para fazer uso das múltiplas *threads* presentes nos processadores multinúcleos amplamente disponíveis hoje.

3.1. Algoritmo Genético (AG)

Para representar as soluções de um problema, os AGs utilizam o conceito de cromossomo, que é um conjunto finito de símbolos representando uma solução viável. Durante o processo de busca na otimização essas soluções (indivíduos) são transformadas pelos operadores genéticos. Em nosso trabalho, cada indivíduo é representado no formato de permutação sequencial. A Figura 2 ilustra sua aplicação.

Figura 2: Exemplo de representação para o CVRP usando permutação.

Clientes (10)	1	2	3	4	5	6	7	8	9	10			
Divisores de Rota (3)	11	12	13										
Exemplo de Cromossomo	6	9	8	11	7	10	12	3	1	2	13	5	4
Rota 1:	depósito → cliente 6 → cliente 9 → cliente 8 → depósito												
Rota 2:	depósito → cliente 7 → cliente 10 → depósito												
Rota 3:	depósito → cliente 3 → cliente 1 → cliente 2 → depósito												
Rota 4:	depósito → cliente 5 → cliente 4 → depósito												

As populações, após a inicialização (I), dentro do contexto do AG são transformadas através de ciclos que tipicamente contém: seleção (II), cruzamento (III) e mutação (IV). Para a inicialização (I), foi adotado um embaralhamento aleatório de cromossomos; Na seleção (II), um par de cromossomos são selecionados aleatoriamente. Em seguida, um valor dentro do intervalo (0, 1) é escolhido de forma aleatória, representando a porcentagem de preferência pelo cromossomo mais

apto entre os dois selecionados. Após essa seleção, ambos os cromossomos retornam à população original, e esse processo é repetido quantas vezes forem necessárias para gerar o número adequado de indivíduos que avançarão para a próxima etapa; Para realizar o cruzamento (III), foi escolhido a abordagem *partially-mapped crossover* (PMX). Para a mutação (IV) foram selecionados 4 métodos, citados por [Eiben e Smith, 2015] conforme a Figura (3): (a) por troca; (b) por inserção; (c) por embaralhamento; (d) por inversão.

Figura 3: Exemplo dos métodos de mutação usados.

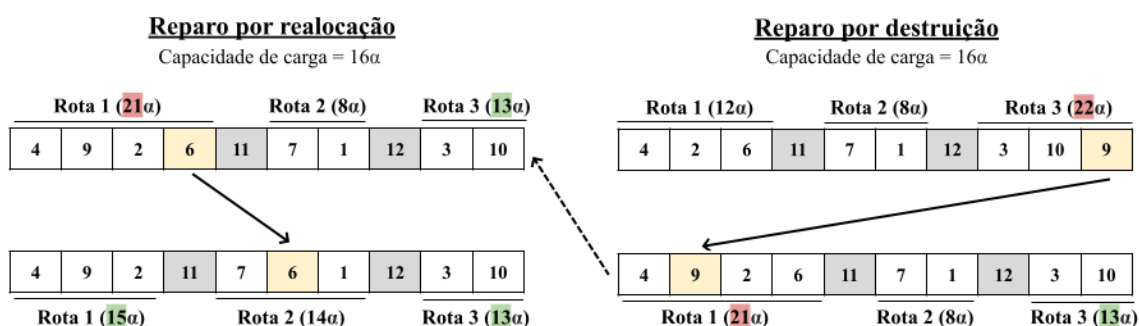
(a) troca	<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table> → <table><tr><td>1</td><td>5</td><td>3</td><td>4</td><td>2</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table>	1	2	3	4	5	6	7	8	9	1	5	3	4	2	6	7	8	9
1	2	3	4	5	6	7	8	9											
1	5	3	4	2	6	7	8	9											
(b) inserção	<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table> → <table><tr><td>1</td><td>2</td><td>5</td><td>3</td><td>4</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table>	1	2	3	4	5	6	7	8	9	1	2	5	3	4	6	7	8	9
1	2	3	4	5	6	7	8	9											
1	2	5	3	4	6	7	8	9											
(c) embaralhamento	<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table> → <table><tr><td>1</td><td>3</td><td>5</td><td>4</td><td>2</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table>	1	2	3	4	5	6	7	8	9	1	3	5	4	2	6	7	8	9
1	2	3	4	5	6	7	8	9											
1	3	5	4	2	6	7	8	9											
(d) inversão	<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table> → <table><tr><td>1</td><td>5</td><td>4</td><td>3</td><td>2</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table>	1	2	3	4	5	6	7	8	9	1	5	4	3	2	6	7	8	9
1	2	3	4	5	6	7	8	9											
1	5	4	3	2	6	7	8	9											

Adicionalmente, as fases de inicialização, cruzamento e mutação podem gerar soluções que violam a restrição de carga, sendo essas soluções inviáveis. Para abordar esse problema, optou-se por implementar uma operação de reparação. Como operador de reparação, visando manter a diversidade genética, foi escolhido a utilização de duas estratégias: reparação por realocação e reparação por destruição. A Figura 4 ilustra os métodos de reparação usados.

- **Reparo por realocação:** envolve a transferência de um cliente para outra rota. De maneira iterativa, quando ocorre uma violação de restrição ao exceder a capacidade de uma rota, a solução procura uma rota alternativa que possa acomodar esse cliente. Se uma rota alternativa for encontrada, o algoritmo insere o cliente nessa rota em uma posição aleatória e continua a busca por outras violações de restrições.
- **Reparo por destruição:** situações podem surgir em que a inserção de um nó em qualquer outra rota seja inviável devido a quebra de restrições. Nesse cenário, a abordagem consiste em realocar aleatoriamente o nó problemático e reiniciar o processo de reparação.

O algoritmo de reparação é usado para corrigir as rotas inválidas antes de juntar as gerações, a atual e seus filhos. Além disso, também são excluídos os novos elementos que são iguais a outros já existentes. Em seguida, são escolhidos os n indivíduos com o melhor *fitness*.

Figura 4: Exemplo dos métodos de reparo usados.



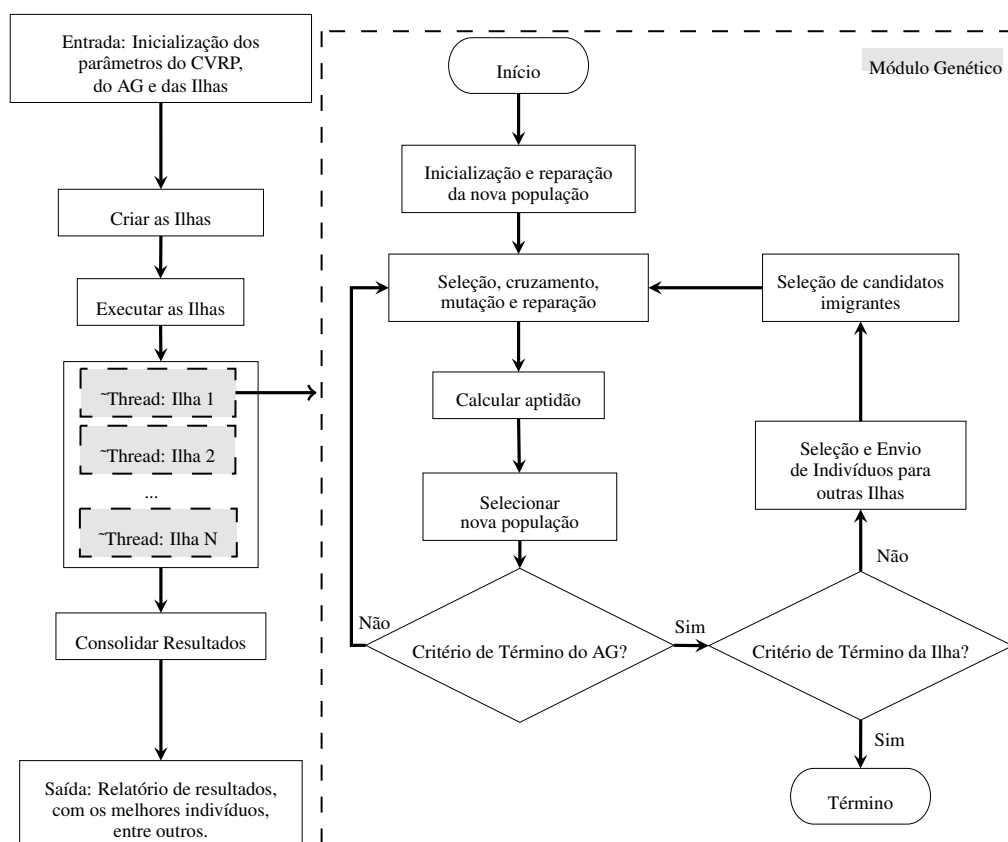
3.2. Modelo de Ilhas

A migração é o principal operador do modelo, garantindo que haja uma pressão global, buscando contornar o fenômeno da convergência prematura em cada domínio. Conforme [Cantú-Paz, 2000], sua implementação depende da definição da quantidade de indivíduos e o número de ilhas, da topologia que as interligam, da taxa de migração e a frequência com que ocorrem. Além disso, deve ser definida a política de migração, que determina quais indivíduos migram, e quais são substituídos na ilha receptora.

Nossa política de troca de material genético entre as ilhas ocorre em duas fases distintas e assíncronas: (I) fase de envio e (II) fase de recebimento. No envio (I), um parâmetro determina quantos indivíduos são selecionados e entregues nas ilhas que possuem conexão. Em (II), uma população candidata de imigrantes pode estar esperando em uma fila (enviados por outras ilhas, conforme o passo I). Quando isso ocorre, na incorporação os novos indivíduos não podem ser iguais a algum outro presente na população, sendo removidos; e, a substituição ocorre com elitismo. A migração ocorre periodicamente, determinado por parâmetro de quantidade de gerações.

A Figura 5 ilustra o modelo proposto, alinhado com as ideias discutidas até o momento. Um módulo externo (à esquerda) apresenta os dados de entrada, com informações sobre a instância a ser resolvida e as configurações (parâmetros) necessários para o AG e as ilhas. Para cada Módulo Genético (à direita), uma *thread* é construída, seguindo o fluxo clássico do AG, com a adição da operação de reparação e a implementação do conceito de ilhas, permitindo que indivíduos sejam periodicamente enviados e recebidos.

Figura 5: Representação da Visão Geral da Heurística Proposta.



4. Avaliação Computacional

Com o objetivo de avaliar a efetividade do modelo proposto, um algoritmo foi implementado e um plano de testes foi definido.

Um código do algoritmo proposto foi desenvolvido utilizando a linguagem de programação C, com uso da biblioteca *pthread*. Instâncias consolidadas da literatura serviram de base para a execução de testes. Todos os testes foram realizados em um ambiente de execução Windows 10 64-bits, processador AMD Ryzen 5 5600 3,5 GHz x 6 núcleos e 12 *threads*, em uma máquina com 16 GB de memória RAM 2667 GHz. A seguir é apresentado os cenários de testes e posteriormente os resultados computacionais.

4.1. Plano de Experimentos

Para avaliar a heurística, foram utilizadas instâncias construídas por [Augerat et al., 1995] (conjuntos A, B e P), [Christofides e Eilon, 1969] (conjunto E), [Fisher, 1994] (conjunto F) e [Christofides et al., 1979] (conjunto M). Tais instâncias estão disponibilizadas em [CVRPLIB, 2014].

Em relação à função aptidão, seguimos a proposta da literatura. Os dados informados representam pontos em um plano 2D. Calculada através da função de distância Euclidiana $d(Euc) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. A matriz de custo para sair de i para j é armazenado com custo inteiro de cada enlace disponível, seguindo os trabalhos desenvolvidos com essas bases, usando a função *round* disponível pela biblioteca *math.h*. O valor total da aptidão é obtido com a somatória das rotas de todos os veículos.

Para a realização de testes, foram selecionadas 24 instâncias, organizadas em grupos, visando proporcionar análises mais específicas dos resultados e suas relações com as diferentes características dos cenários analisados. Quatro principais perspectivas são encontradas nos CVRP: (I) quantidade de clientes que precisam ser atendidos; (II) quantidade de veículos que irão resolver as demandas; (III) tipo de distribuição dos dados, que se refere ao formato de distribuição (espalhamento) dos clientes no espaço geográfico, tipicamente aleatórios ou agrupados; e, (IV) e o *tightness* (ou *tigh*, tradução literal aperto ou rigidez), que é a relação entre a demanda total com o total da capacidade dos veículos. Uma instância é dita apertada quando o *tightness* se aproxima de 1, indicando que os veículos estarão carregados ao limite. Quanto menor essa razão, maior será a flexibilidade para alocação dos clientes.

O primeiro grupo (G1) considera uma quantidade baixa de veículos e uma distribuição aleatória de clientes. O objetivo desse grupo é avaliar o desempenho à medida que o número de clientes aumenta significativamente. No segundo grupo (G2), mantém-se uma padronização no número de veículos, enquanto pares aproximados de clientes são usados, variando o formato de distribuição. Isso permite verificar se pequenos agrupamentos de clientes influenciam nos resultados. No terceiro grupo (G3), a quantidade de clientes foi fixada, variando principalmente o *tightness*. O principal objetivo desse grupo é analisar o impacto dessa variação nos resultados a partir de problemas semelhantes. O quarto grupo (G4) apresenta situações em que os clientes são iguais em quantidade e posicionamento no plano, com variação na quantidade de veículos e no *tightness*. Esse grupo simula um cenário de tomada de decisão em relação ao número e capacidade dos veículos, considerando uma demanda específica a ser atendida. Por fim, o quinto grupo (G5) foca em problemas de maior escala, variando as características. O objetivo é observar o comportamento da heurística em problemas mais complexos, tanto em termos de qualidade das soluções quanto no tempo de execução. A Tabela 1 lista todos os casos selecionados e suas características básicas. A coluna *tight* refere-se ao conceito de *tightness* (aperto, em português), que é a razão entre o somatório de todas as demandas e a capacidade total dos veículos, Fisher [1994].

Para cada caso de teste, abordamos a realização de 10 execuções do algoritmo, variando o número de ilhas, contemplando configurações de 1, 2, 4 e 6 ilhas. Esse procedimento totalizou 960

execuções, abrangendo todas as combinações de problemas, repetições e ilhas.

Tabela 1: Instâncias de testes.

Grupo	Nome	Qnt de Clientes	Qnt de Veículos	Distribuição	Tigh
G1	E-n22-k4	21	4	Aleatório	0,94
	E-n33-k4	32	4	Aleatório	0,92
	F-n45-k4	44	4	Aleatório	0,9
	F-n72-k4	71	4	Aleatório	0,96
	P-n101-k4	100	4	Aleatório	0,91
G2	B-n34-k5	33	5	Cluster	0,91
	A-n34-k5	33	5	Aleatório	0,92
	B-n39-k5	38	5	Cluster	0,88
	A-n39-k5	38	5	Aleatório	0,95
	P-n45-k5 ¹	44	5	Aleatório	0,92
	B-n45-k5 ²	44	5	Cluster	0,97
G3	P-n45-k5 ¹	44	5	Aleatório	0,92
	B-n45-k5 ²	44	5	Cluster	0,97
	A-n45-k6	44	6	Aleatório	0,99
	B-n45-k6	44	6	Cluster	0,99
G4	P-n76-k4	75	4	Aleatório	0,97
	E-n76-k7	75	7	Aleatório	0,89
	E-n76-k10	75	10	Aleatório	0,97
	E-n76-k14	75	14	Aleatório	0,97
G5	M-n101-k10	100	10	Cluster	0,91
	E-n101-k14	100	14	Aleatório	0,93
	M-n121-k7	120	7	Cluster	0,98
	F-n135-k7	134	7	Aleatório	0,95
	M-n151-k12	150	12	Aleatório	0,93
	M-n200-k16	199	16	Aleatório	1
	M-n200-k17	199	17	Aleatório	0,94

4.2. Calibração de Parâmetros

A calibração apropriada dos parâmetros do Algoritmo Genético (AG) e do modelo de ilhas é uma etapa crítica que envolve o balanceamento entre busca local e global. O objetivo principal é prevenir a convergência prematura e garantir uma exploração eficaz do espaço de soluções. Para isso, testes preliminares foram realizados usando a instância *E-n76-k10* com uso de 4 ilhas. Foram avaliados variações na topologia unidirecional e bidirecional: para a quantidade de gerações foram testadas $[500, \dots, 5000] \pm 250$; na etapa de seleção, foram usados os $[50\%, \dots, 90\%] \pm 5\%$ mais apto; a mutação variou em $[0.5\%, \dots, 10\%] \pm 0.5\%$; a quantidade de indivíduos filhos que poderão ser utilizados na nova geração variou de $[5\%, \dots, 100\%] \pm 5\%$; o intervalo de migração em gerações foi $[100, \dots, 300] \pm 50$; por fim, a quantidade de imigrantes entre ilhas variou em $[1, \dots, 10] \pm 1$. A Tabela 2 apresenta o resultado dos experimentos de calibração do algoritmo.

Tabela 2: Resumo dos Parâmetros Calibrados.

	Parâmetro	Calibração Escolhida
Ilha	Vizinhança	Unidirecional
	Intervalo de Migração (em gerações)	300
	Imigrantes	2
	Tamanho das Populações	100
Alg. Genético	Quantidade de Gerações	4500
	Seleção	80%
	Mutação	7%
	Nova Geração	10%

4.3. Resultados

A Tabela 3 contém os resultados a partir das médias obtidas com a execução das instâncias. Os resultados foram representados com: média aritmética (μ); desvio padrão (σ); melhor resultado (*Best*); e tempo em segundos ($t(s)$). Tais métricas estão apresentadas para todas as quantidades de ilhas testadas. Adicionalmente, também é apresentado o nome da instância, na literatura, e o ótimo.

Tabela 3: Resultados dos testes realizados

Grupo	Instância	Ótimo	1 ilha				2 ilhas				4 ilhas				6 ilhas			
			μ	σ	<i>Best</i>	$t(s)$	μ	σ	<i>Best</i>	$t(s)$	μ	σ	<i>Best</i>	$t(s)$	μ	σ	<i>Best</i>	$t(s)$
G1	E-n22-k4	375	385,5	13,8	375	0,96	378,3	4,5	375	0,99	375,7	2,21	375	1,05	375,7	2,21	375	1,16
	E-n33-k4	835	853,8	10,77	839	0,9	851,5	12,68	836	0,96	848	9,45	835	0,96	839,5	5,21	835	1,09
	F-n45-k4	721	788,1	63,86	721	0,93	740,8	13,51	727	0,98	732,3	12,12	721	0,98	728,9	5,86	721	1,1
	F-n72-k4	237	283,6	16,16	260	1,02	271,9	14,81	243	1,06	250,9	8,76	241	1,06	259,2	11,57	245	1,23
	P-n101-k4	681	756,9	14,07	735	1,02	728,4	17,18	697	1,08	734,1	16,49	703	1,1	724,3	9,53	704	1,28
G2	B-n34-k5	788	802,5	10,65	789	0,89	792,3	2,41	788	0,92	789,1	2,13	788	0,9	789,1	1,66	788	1,11
	F-n34-k5	778	793,3	8,63	780	0,91	792,7	11,68	778	0,93	788,1	7,82	778	0,93	788,4	5,48	778	1,08
	B-n39-k5	549	578,7	20,74	549	0,89	564,5	14,36	549	0,94	560,4	9,26	549	0,97	553,6	4,84	549	1,13
	A-n39-k5	822	877,2	45,91	836	0,92	857,8	21,36	832	0,94	843,7	21,68	827	0,94	842,7	14,28	822	1,13
	P-n45-k5	510	537,3	18,8	516	0,95	521	9,09	510	0,98	526,2	9,35	515	0,98	517,6	5,02	510	1,12
G3	B-n45-k5	751	861,2	70,46	770	0,92	787,6	13,3	767	0,96	785,9	26,51	753	0,95	779,5	12,24	759	1,07
	P-n45-k5	510	537,3	18,8	516	0,95	521	9,09	510	0,98	526,2	9,35	515	0,98	517,6	5,02	510	1,12
	B-n45-k5	751	861,2	70,46	770	0,92	787,6	13,3	767	0,96	785,9	26,51	753	0,95	779,5	12,24	759	1,07
	A-n45-k6	944	1198,90	48,16	1123	0,95	1143,60	76,08	1057	0,96	1097,00	62,02	972	1,02	1073,40	48,15	1010	1,14
	B-n45-k6	678	836,7	56,64	770	0,94	801,5	21,86	759	0,94	754,6	22,82	726	0,95	753,9	21,16	716	1,12
G4	P-n76-k4	593	681,2	37,35	624	0,94	673,5	22,9	636	0,95	658,3	16,53	632	0,95	638,6	15,01	613	1,15
	E-n76-k7	682	740,7	20,76	709	1,05	720,4	12,38	699	1,06	713,6	13,01	689	1,16	709,2	8,22	700	1,26
	E-n76-k10	830	975,8	41,28	914	1,05	962,8	47,4	860	1,07	938,8	33,81	908	1,07	928,5	21,36	900	1,25
	E-n76-k14	1021	1171,00	31,13	1133	1,07	1160,30	28	1124	1,08	1122,60	36,51	1075	1,09	1129,20	26,87	1100	1,27
	M-n101-k10	820	975,9	24,74	940	1,13	941,7	28,88	907	1,16	917,6	35,83	862	1,18	917,4	29,24	875	1,36
G5	E-n101-k14	1067	1217,40	18,51	1197	1,15	1201,70	11,72	1184	1,19	1183,70	11,21	1159	1,24	1171,30	16,31	1149	1,42
	M-n121-k7	1034	1503,50	46,03	1440	1,22	1483,30	64,34	1417	1,32	1444,70	61,54	1355	1,32	1433,40	33,23	1393	1,54
	F-n135-k7	1162	1521,00	90,59	1404	1,26	1439,10	54,68	1360	1,3	1412,40	41,58	1359	1,35	1377,80	63	1282	1,59
	M-n151-k12	1015	1316,80	43,12	1283	1,4	1314,30	22,03	1290	1,44	1302,90	30,21	1266	1,48	1276,20	35,7	1214	1,72
	M-n200-k16	1274	2797,70	54,2	2739	1,96	2693,30	88,22	2543	1,97	2700,50	46,89	2594	2,02	2635,10	76,68	2513	2,36
	M-n200-k17	1275	1893,90	19,9	1867	1,71	1874,00	24,92	1833	1,75	1848,20	54,36	1773	1,8	1793,30	41,73	1735	2,08

De maneira geral, observa-se uma melhoria progressiva tanto na média quanto no desvio padrão à medida que mais ilhas foram adicionadas. A distância da média dos resultados em relação a aptidão ótima se aproxima em 19,79% (1), 16,22% (2), 14,40% (4) e 13,23% (6). Esses dados indicam uma tendência do modelo em aprimorar o desempenho, embora haja o aumento correspondente no tempo de processamento e no consumo de recursos computacionais. O tempo de execução (em segundos) aumentou em média 3,07% (2), 4,89% (4), e 21,21% (6). Contudo, destacamos que tais valores referem-se a menos que 1 segundo.

Sobre o G1, o algoritmo foi capaz de encontrar soluções ótimas ou próximas, com uma tendência de degradação na medida em que os problemas ficam maiores, como em *F-n72-k5* e *P-n101-k4*. O crescimento das ilhas na maioria dos casos representou um avanço no sentido de uma menor dispersão (σ). A maioria dos testes se aproxima da aptidão ótima com o aumento de ilhas. Além das observações globais, outro ponto que chama atenção foi que a instância *F-n72-k5* gerou os piores resultados do grupo, apesar de não ser o maior caso de teste. Esse item indica fortemente o impacto do *tigh* no cenário.

No grupo G2, não foram encontradas evidências conclusivas sobre a influência do formato de distribuição dos clientes no desempenho do algoritmo, em instâncias de tamanho médio com certa flexibilidade. Em quase todos os testes, foi possível obter uma solução igual ou próxima do ótimo.

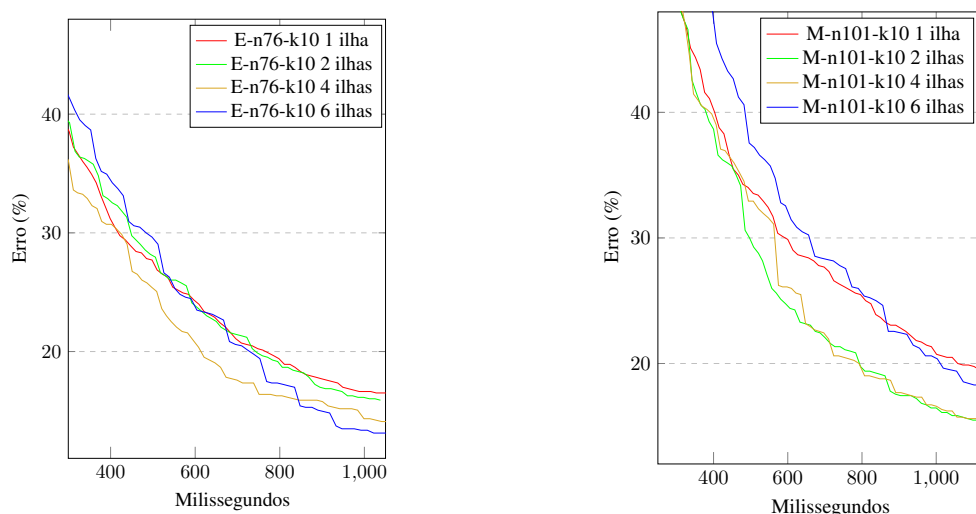
As médias do grupo G4 seguiram o padrão de melhoria com o acréscimo das ilhas observada nos demais conjuntos. O acréscimo de veículos representou uma melhoria na distância em relação ao ótimo. Mais veículos carregam menos carga e têm um cromossomo maior, ocasionando maior dificuldade em realizar a busca. Além disso, a instância *E-n76-k7*, que é menos restritiva, obteve o melhor resultado do grupo, se aproximando do ótimo.

G3 e G5 corroboram com a ideia de que um dos desafios para o algoritmo é a resolução de problemas com *tigh* apertado. Os dados encontrados demonstram uma melhoria com o uso de ilhas, porém há uma clara queda de desempenho nos resultados em problemas com pouca margem. Além disso, esses dados demonstram que apesar da maior dificuldade em realizar a busca, esse fator não gerou alterações que representassem um maior tempo de processamento.

Em relação ao Grupo 5, observou-se uma progressão dos resultados com o uso de mais ilhas, sendo possível ver uma curva de melhora em todos os casos. Como segunda observação, percebe-se que houve uma diferença entre as instâncias com margem (*tigh*) e as sem margem. Um comparativo entre *M-n200-k16* e *M-n200-k17*, com margem respectivas de 1 e 0,94, confirma uma diferença consistente na capacidade do método em melhorar as soluções com essas características. O mesmo *outlier* ocorre olhando para *M-n121-k7*, com resultados comparáveis a *M-n151-k12*, cujo problema é consideravelmente maior.

Para ilustrar a evolução da qualidade da aptidão, a Figura 6 apresenta a progressão da porcentagem de erro no eixo Y, calculada pela fórmula $\frac{Aptidao_{obtida} - Aptidao_{otimo}}{Aptidao_{otimo}} * 100$, em função do tempo (em milissegundos) no eixo X, para duas instâncias do plano de testes: *E-n76-k10* e *M-n101-k10*. Nos instantes iniciais do algoritmo, observa-se que a instância com 1 ilha apresenta uma aptidão comparável às demais, enquanto a instância com 6 ilhas se destaca negativamente, exibindo a pior média. À medida que o tempo avança, ocorre uma degradação na qualidade da instância com 1 ilha, tornando-se a pior em todos os casos por volta dos 900 milissegundos. No primeiro caso (*E-n76-k10*), a instância com 6 ilhas passa a apresentar a melhor aptidão. No segundo caso (*M-n101-k10*), apesar de a instância com 6 ilhas mostrar uma melhora contínua na qualidade da solução, ela não alcança o desempenho das instâncias com 2 ou 4 ilhas, devido à maior complexidade do problema. Em todos os cenários ocorreu uma melhor aproximação com o acréscimo de ilhas.

Figura 6: Evolução da aptidão para as instâncias *E-n76-k10* e *M-n101-k10*



Adicionalmente, observamos uma adaptação limitada aos parâmetros ajustados do algoritmo, com foco especial na quantidade de gerações. Essa limitação se refletiu no tempo de execução dos testes, onde se observou uma amplitude de apenas 1,47 segundos entre as instâncias. Apesar disso, ao nível de comparação, outros trabalhos que utilizaram algumas das instâncias desse trabalho, encontraram valores ótimos para casos do grupo A e B com tempos próximos de 1 minuto. Como [Faiz et al., 2018] que obteve um tempo médio superiores a 1 minutos; e em [Sbai et al., 2020] onde os autores alcançaram a solução ótima da instância *P-n45-k5* em 37,65 segundos. Destacamos que em nosso trabalho conseguimos essa convergência em alguns dos testes, porém com

média de 1,12 segundos.

5. Conclusão

Neste trabalho, propomos a implementação de uma heurística baseada em Algoritmos Genéticos para resolver o problema de roteamento de veículos capacitados. Para otimizar a eficiência da implementação, exploramos uma abordagem paralela assíncrona do modelo de ilhas, que utiliza processador multinúcleo, *threads* e memória compartilhada.

Foram conduzidos testes para avaliar o método construído, variando a quantidade de ilhas, e mantendo constante os parâmetros genéticos e migratórios. Os resultados revelaram uma melhoria das soluções com baixo custo de processamento, ao comparar o uso de uma única ilha com múltiplas ilhas, permitindo um melhor aproveitamento do sistema multinúcleo. Em 87,5% das instâncias avaliadas, a melhor média foi alcançada com o uso de 6 ilhas, enquanto nos outros 12,5%, 4 ilhas foram mais eficazes. Para isso, houve um aumento de custo médio de 4,84% (4 ilhas) e 21,09% (6 ilhas) no tempo (em segundos) em relação ao AG tradicional (1 ilha). Contudo, cabe destacar que esse aumento computacional é menor do que 1 segundo. Além disso, o acréscimo de ilhas resultou em uma redução significativa no desvio padrão.

No geral, o modelo se demonstrou promissor como alternativa para a construção de sistemas para a resolução de problemas de roteamento de veículos. Assim, considerando uma expansão horizontal de *hardware*, pode ser uma ferramenta útil para a otimização de problemas complexos.

Trabalhos futuros podem explorar o impacto na diversidade no modelo. Parâmetros ou técnicas distintas podem ser usados nas ilhas, tanto no sentido de criar ambientes propícios para diversidade de formato de exploração, no sentido de adaptabilidade quando se detecta uma dificuldade em realizar uma busca local, e no sentido de explorar outras heurísticas. Além disso, a arquitetura pode ser expandida para ambientes de computação distribuída, visando solucionar problemas maiores, mais complexos ou técnicas de soluções mais custosas que justifiquem seu uso.

6. Agradecimento

Os autores agradecem a FAPES/UnAC (Nº FAPES 1228/2022 P 2022-CD0RQ, Nº SIA-FEM 2022-CD0RQ) pelo apoio financeiro dado por meio do Sistema UniversidaES.

Referências

- Altabeeb, A. M., Mohsen, A. M., e Ghallab, A. (2019). An improved hybrid firefly algorithm for capacitated vehicle routing problem. *Applied Soft Computing*, 84:105728. ISSN 1568-4946. URL <https://doi.org/10.1016/j.asoc.2019.105728>.
- Amous, M., Toumi, S., Jarboui, B., e Eddaly, M. (2017). A variable neighborhood search algorithm for the capacitated vehicle routing problem. *Electronic Notes in Discrete Mathematics*, 58:231–238. ISSN 1571-0653. URL <https://doi.org/10.1016/j.endm.2017.03.030>. 4th International Conference on Variable Neighborhood Search.
- Augerat, P., Naddef, D., Belenguer, J., Benavent, E., Corberan, A., e Rinaldi, G. (1995). Computational results with a branch and cut code for the capacitated vehicle routing problem.
- Ballou, R. H. (2006). *Gerenciamento da Cadeia de Suprimentos/Logística Empresarial*. Bookman, 5ª edition.
- Blasum, U. e Hochstättler, W. (2004). Application of the branch and cut method to the vehicle routing problem. *Zentrum für Angewandte Informatik Köln*.
- Borcinova, Z. (2017). Two models of the capacitated vehicle routing problem. *Croatian Operational Research Review*, 8:463–469.

- Cantú-Paz, E. (1998). A survey of parallel genetic algorithms. *Calculateurs Paralleles, Reseaux et Systems Repartis*, 10(2):141–171.
- Cantú-Paz, E. (2000). Efficient and accurate parallel genetic algorithms. *Boston, Kluwer Academic Publishers*.
- Christofides, N., Mingozzi, A., e Toth, P. (1979). The vehicle routing problem. *Combinatorial Optimization*. Wiley, Chichester, p. 315–338.
- Christofides, N. e Eilon, S. (1969). An algorithm for the vehicle-dispatching problem. *Journal of the Operational Research Society*, 20:309–318. URL <https://api.semanticscholar.org/CorpusID:62174543>.
- Christofides, N., Mingozzi, A., e Toth, P. (1981). State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11(2):145–164. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/net.3230110207>.
- Crainic, T. G. e Toulouse, M. (2003). *Parallel Strategies for Meta-Heuristics*, p. 475–513. Springer US, Boston, MA. ISBN 978-0-306-48056-0. URL https://doi.org/10.1007/0-306-48056-5_17.
- CVRPLIB (2014). Cvrplib - all instances. URL <http://vrp.galgos.inf.puc-rio.br/>.
- Eiben, A. E. e Smith, J. E. (2015). *Introduction to Evolutionary Computing*. Springer Publishing Company, Incorporated, 2nd edition. ISBN 3662448734.
- Faiz, A., Subiyanto, S., e Arief, U. (2018). An efficient meta-heuristic algorithm for solving capacitated vehicle routing problem. *International Journal of Advances in Intelligent Informatics*, 4 (3):212–225. ISSN 2548-3161.
- Fisher, M. L. (1994). Optimal solution of vehicle routing problems using minimum k-trees. *Operations Research*, 42(4):626–642. ISSN 0030364X, 15265463. URL <http://www.jstor.org/stable/171617>.
- Hottung, A. e Tierney, K. (2019). Neural large neighborhood search for the capacitated vehicle routing problem. *arXiv*. URL <https://arxiv.org/abs/1911.09539>.
- Lysgaard, J., Letchford, A. N., e Eglese, R. W. (2004). A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100:423–445. ISSN 0025-5610.
- Ralphs, T., Kopman, L., Pulleyblank, W., e Trotter, L. (2003). On the capacitated vehicle routing problem. *Mathematical Programming*, 94:343–359. ISSN 0025-5610.
- Sbai, I., Krichen, S., e Limam, O. (2020). Two meta-heuristics for solving the capacitated vehicle routing problem: the case of the tunisian post office. *Operational Research*, 22(1):507–549. URL <https://doi.org/10.1007/s12351-019-00543-8>.
- Toth, P. e Vigo, D. (2002). *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics. URL <https://epubs.siam.org/doi/abs/10.1137/1.9780898718515>.
- Toth, P. e Vigo, D. (2003). The granular tabu search and its application to the vehicle-routing problem. URL <http://dx.doi.org/10.1287/ijoc.15.4.333.24890>.