Um estudo do problema de roteamento de veículos e implementações de algoritmos genéticos e meméticos

Luan Sandim Rohwedder Profa. Dra. Bianca de Almeida Dantas

12 de Dezembro de 2024

RESUMO

Este trabalho apresenta um estudo sobre o problema do roteamento de veículos capacitados, explorando a aplicação de Algoritmos Genéticos e Algoritmos Meméticos como alternativas para obter soluções de boa qualidade. Para aprimorar a eficiência dos algoritmos, implementou-se uma estratégia de inicialização mista, combinando métodos aleatórios e heurísticos, além de operadores de cruzamento, mutação e seleção otimizados. No Algoritmo Memético, incorporou-se um operador de busca local baseado em Recozimento Simulado (*Simulated Annealing*) para refinamento das soluções. Além disso, visando acelerar o processamento e explorar os recursos fornecidos pelos múltiplos núcleos de processamento dos processadores atuais, utilizou-se o *OpenMP* com o objetivo de acelerar a execução dos principais operadores evolutivos.

Palavras-chave: Algoritmos Genéticos; Algoritmos Meméticos; Problema do Roteamento de Veículos Capacitados, OpenMP.

ABSTRACT

This paper presents an analysis of the Capacitated Vehicle Routing Problem (CVRP), exploring the application of Genetic Algorithms (GAs) and Memetic Algorithms (MAs) as optimization strategies. To enhance algorithm efficiency, a hybrid initialization strategy was implemented, combining random and heuristic-based methods, along with optimized crossover, mutation, and selection operators. In the case of MAs, a local search operator based on Simulated Annealing was incorporated to refine solutions. Additionally, to accelerate processing and take advantage of the multiple cores available in modern processors, parallelization with OpenMP was employed to optimize the execution of key evolutionary operators.

Keywords: Genetic Algorithm; Memetic Algorithm; Capacitated Vehicles Routing Problem, OpenMP.

1 INTRODUÇÃO

O Problema de Roteamento de Veículos Capacitados (CVRP, do inglês *Capacitated Vehicle Routing Problem*) é um dos principais desafios de otimização combinatória enfrentados em áreas como logística e distribuição. Este problema visa determinar as rotas mais eficientes para uma frota de veículos, cada um com uma capacidade limitada, de forma a atender um conjunto de clientes com demandas específicas, minimizando os custos operacionais, como distância percorrida ou tempo de viagem. O CVRP surge comumente em cenários de transporte, no qual a demanda por entregas ou coletas precisa ser atendida de forma eficaz, respeitando as restrições de capacidade de cada veículo. Com o avanço das tecnologias e o aumento da demanda por soluções logísticas mais otimizadas, a resolução desse problema tornou-se cada vez mais relevante, especialmente em áreas como *e-commerce*, gestão de cadeias de suprimentos e planejamento urbano.

O Problema de Roteamento de Veículos (VRP, do inglês *Vehicle Routing Problem*) foi proposto por (Dantzig; Ramser, 1959), como uma extensão do Problema do Caixeiro Viajante. Naquela época, o problema era abordado principalmente por métodos lineares, refletindo as limitações computacionais e tecnológicas disponíveis. Atualmente diversos métodos foram propostos para resolvê-lo, incluindo abordagens exatas, como algoritmos de *branch-and-bound* (Toth; Vigo, 2002), assim como diversas variantes para o problema foram introduzidas, como o VRP com janela de tempo (VRPTW).

Os algoritmos genéticos (AG) são meta-heurísticas consideradas algoritmos evolucionários, inspirados na teoria da evolução natural de Charles Darwin. Algumas das principais aplicações dos algoritmos incluem otimização de funções, otimização combinatória, escalonamento de tarefas e robótica, abordando de forma eficaz problemas complexos e desafiadores em diversas áreas (Hu, 2024).

Além dos AG, um método amplamente utilizado para otimização é o algoritmo memético (AM). Ele pode ser visto como uma extensão dos AGs tradicionais, integrando técnicas de busca local para melhorar as soluções geradas. Os algoritmos meméticos, por combinarem técnicas evolutivas com busca local, constituem uma abordagem híbrida eficaz, sendo capazes de oferecer soluções de alta qualidade para problemas de otimização complexos (Wilson et al., 2022). Dessa forma, o objetivo deste trabalho é apresentar como algoritmos genéticos e meméticos podem ser utilizados para obter soluções para o problema de roteamento de veículos capacitados em tempo viável e comparar os resultados obtidos entre si e com os de outros trabalhos da literatura.

2 REFERENCIAL TEÓRICO

Ao longo desta seção, são apresentados os fundamentos teóricos do problema do roteamento de veículos (Seção 2.1), o funcionamento do algoritmo genético (Seção 2.2) e do algoritmo memético (Seção 2.3).

2.1 Problema do Roteamento de Veículos

O VRP é uma questão central na logística, afetando tanto a eficiência das entregas quanto os custos operacionais. A otimização das rotas de entrega é fundamental para que as empresas se mantenham competitivas no mercado. (Mubarak; Mawengkang; Suwilo, 2024).

Nesse contexto, o problema pode ser descrito por um grafo direcionado $G(V, E)$, onde $V = \{0, 1, ..., n\}$ representa um conjunto de nós, e E corresponde ao conjunto de arestas. O depósito é indicado pelo nó $j = 0$, enquanto os nós $j = 1, 2, ..., n$ correspondem aos clientes, cada um com uma demanda $d_j > 0$. Cada aresta que conecta os nós i e j define uma rota, cujo peso $c_{ij} > 0$ representa o custo, medido em termos de distância ou tempo (Laporte, 1992). A Figura 1 ilustra a representação do VRP.

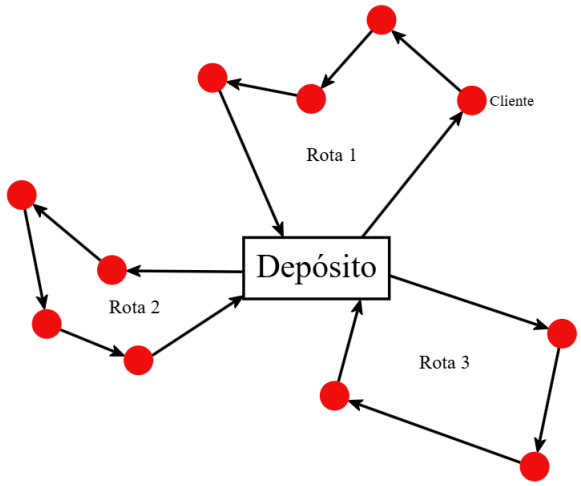

Figura 1 – Problema do Roteamento de Veículos.
Fonte: Elaborada pelo autor

Uma das variantes mais estudadas do VRP, e também o abordado neste artigo, é conhecido como problema do roteamento de veículos capacitados, que pode ser modelado da seguinte forma (Noon; Mittenthal; Pillai, 1994):

Notação

- $\mathcal{N}$: Conjunto de clientes, $\mathcal{N} = \{1, 2, \ldots, n\}$.
- V: Conjunto de veículos disponíveis, $V = \{1, 2, \ldots, V\}$.
- E: Conjunto de arcos possíveis, $E = \{(i, j) \mid i \neq j,\ i, j \in \{0, 1, \ldots, n\}\}$.
- c_{ij}^v: Custo de deslocamento do veículo v de i para j.
- x_{ij}^v: Variável binária que indica se o veículo v percorre a aresta (i, j).
- a_{ij}^v: Quantidade de recurso consumido pelo veículo v ao percorrer (i, j).
- b_v: Capacidade máxima do veículo v.
- $\mathcal{S} \subseteq \mathcal{N}$: Subconjunto de clientes usado na restrição de subtour.
- 0: Representa o depósito.

$$\min \sum_{v=1}^{V} \sum_{(i,j)\in E} c_{ij}^v x_{ij}^v \tag{1}$$

Sujeito a:

$$\sum_{v=1}^{V} \sum_{\substack{i=0 \\ i \neq j}}^{n} x_{ij}^{v} + \sum_{v=1}^{V} \sum_{\substack{k=0 \\ k \neq j}}^{n} x_{jk}^{v} = 2, \quad \forall j \in \mathcal{N} \tag{2}$$

$$\sum_{j=1}^{n} x_{0j}^{v} = 1 \tag{3}$$

$$\sum_{\substack{i=0 \\ i \neq j}}^{n} x_{ij}^{v} \leq 1, \quad \forall j \in \mathcal{N} \tag{4}$$

$$\sum_{i=1}^{n} x_{i0}^{v} = 1 \tag{5}$$

$$\sum_{\substack{i=0 \\ i \neq j}}^{n} x_{ij}^{v} - \sum_{\substack{k=0 \\ k \neq j}}^{n} x_{jk}^{v} = 0, \quad \forall j \in \mathcal{N} \tag{6}$$

$$\sum_{i \in \mathcal{S}} \sum_{\substack{j \in \mathcal{S} \\ j \neq i}} x_{ij}^{v} \leq |\mathcal{S}| - 1, \quad \forall \mathcal{S} \subseteq \mathcal{N}, |\mathcal{S}| \geq 2 \tag{7}$$

$$\sum_{(i,j) \in E} a_{ij}^{v} x_{ij}^{v} \leq b_{v} \tag{8}$$

$$x_{ij}^{v} \in \{0, 1\}, \quad \forall (i, j) \in E \tag{9}$$

O objetivo dessa formulação é minimizar o custo total das rotas (1). A restrição (2) garante que, para cada cliente j, o veículo entre e saia exatamente uma vez, assegurando que todos os clientes sejam atendidos. As restrições (3) a (9) são definidas individualmente para cada veículo $v \in V$. A restrição (3) assegura que cada veículo saia do depósito uma vez, enquanto a (4) garante que cada cliente seja visitado no máximo por um veículo, evitando assim a duplicidade de atendimento. A restrição (5) garante que os veículos retornem ao depósito após atender os clientes. A restrição (6) equilibra o fluxo de veículos em cada cliente, e a (7) impede a formação de sub-rotas entre subconjuntos de clientes. A capacidade dos veículos é respeitada pela restrição (8), e, por fim, a restrição (9) estabelece que as variáveis de decisão são binárias, indicando se uma rota é percorrida ou não.

2.2 Algoritmo Genético

Algoritmos genéticos são técnicas de busca heurística inspiradas na seleção natural, usadas para resolver problemas de otimização e busca. Eles funcionam evoluindo soluções ao longo de gerações, por meio de processos como seleção, cruzamento e mutação (Le; Wei; Tagalogon, 2023).

Os algoritmos genéticos seguem uma abordagem iterativa para evoluir soluções ao longo do tempo. Eles funcionam com uma população de soluções candidatas (chamadas de indivíduos), onde cada indivíduo é avaliado de acordo com uma função de aptidão (*fitness*), que mede o quão boa é a solução em relação ao problema a ser resolvido. A evolução

ocorre ao longo de várias gerações e, em cada geração, são aplicadas operações genéticas como seleção, cruzamento e mutação, visando criar uma nova população com melhores características.

O algoritmo genético envolve essencialmente seis etapas principais:

1. **Inicialização da população:** Uma população inicial de soluções é gerada aleatoriamente ou por algum método heurístico específico.
2. **Avaliação da população:** Cada indivíduo na população é avaliado com base em sua aptidão.
3. **Seleção:** Os indivíduos mais aptos são selecionados para participar do processo de reprodução, onde soluções melhores têm mais chances de sobreviver para a próxima geração.
4. **Cruzamento:** Duas soluções são combinadas para gerar novos indivíduos. Essa operação permite a troca de características entre soluções e aumenta a diversidade da população.
5. **Mutação:** Pequenas alterações são introduzidas em alguns indivíduos para explorar novas soluções no espaço de busca e evitar a estagnação em soluções locais.
6. **Nova geração:** Após a aplicação dos operadores de seleção, cruzamento e mutação, é formada uma nova população de indivíduos. Essa nova geração substitui parcial ou totalmente a geração anterior, dependendo da estratégia adotada.

O Algoritmo 1 apresenta o pseudocódigo geral de um algoritmo genético, enquanto a Figura 2 ilustra as etapas de um algoritmo genético.

Algoritmo 1: Algoritmo Genético Básico

```
Gerar a população inicial P;
enquanto critério de parada não satisfeito faça
    Avaliar os indivíduos de P;
    Selecionar indivíduos de P para reprodução;
    Aplicar cruzamento para gerar descendentes;
    Aplicar mutação nos descendentes;
    Formar nova população P com base nos descendentes e/ou pais;
fim
retorna melhor solução encontrada
```

No fluxograma, o algoritmo genético começa com a inicialização de uma população. Em seguida, essa população é avaliada, e o algoritmo verifica se a condição de término foi satisfeita. Se a condição de término ainda não foi alcançada, o processo de evolução é iniciado, onde ocorrem as operações de seleção, cruzamento e mutação, resultando na criação de uma nova geração. Isso continua até que uma solução satisfatória seja encontrada, momento em que o algoritmo retorna a melhor solução obtida.

Os componentes centrais do algoritmo genético são discutidos nas subseções seguintes, abrangendo aspectos como representação (Seção 2.2.1), inicialização (Seção 2.2.2), avaliação (Seção 2.2.3), seleção (Seção 2.2.4), cruzamento (Seção 2.2.5), mutação (Seção 2.2.6) e formação da nova geração (Seção 2.2.7).

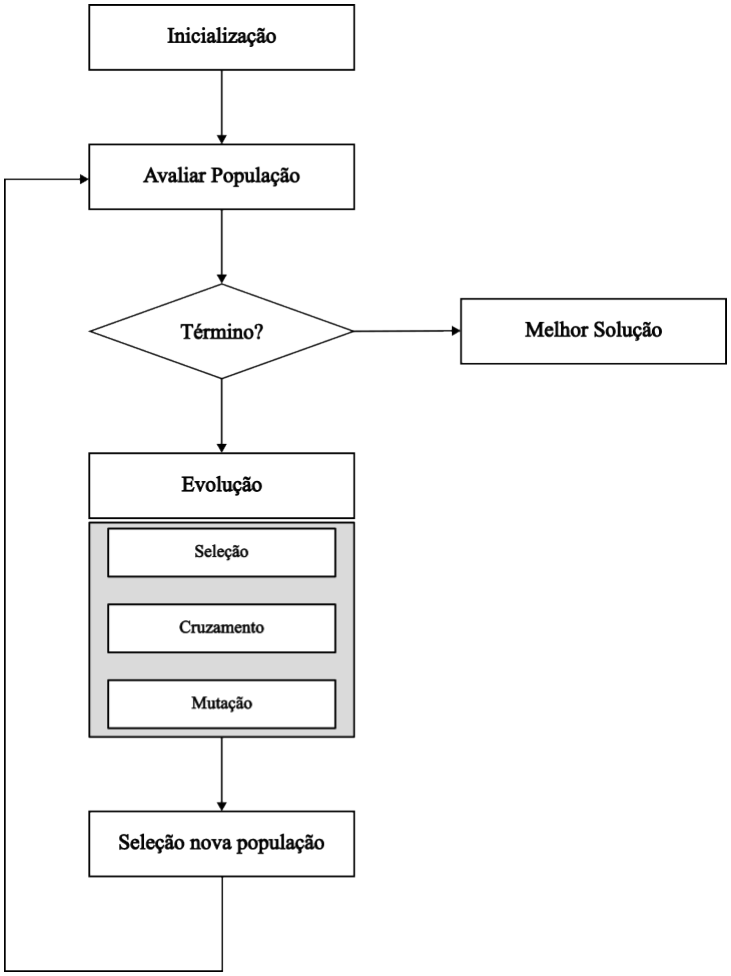

Figura 2 – Fluxograma básico de um Algoritmo Genético.
Fonte: Elaborada pelo autor

2.2.1 Representação

A representação de soluções em algoritmos genéticos é um dos aspectos fundamentais para o sucesso da abordagem. Ela define como cada indivíduo na população, ou seja, cada solução candidata, será codificado de modo que possa ser manipulado pelo algoritmo ao longo das iterações. Dependendo do problema, a representação pode variar, sendo comum a utilização de vetores binários, permutações ou até codificações mais complexas, como números reais.

No caso de uma representação binária, cada indivíduo é codificado como uma sequência de *bits* (0 e 1), na qual cada *bit* pode representar uma característica ou decisão específica no contexto do problema. Um exemplo típico de um indivíduo codificado como um vetor binário é mostrado na Figura 3.

1	0	0	1	0	1

Figura 3 – Indivíduo representado por uma sequência binária.
Fonte: Elaborada pelo autor

2.2.2 Inicialização

A Inicialização é a primeira etapa do algoritmo genético, na qual a população inicial de soluções é gerada. Existem várias maneiras de realizar a inicialização, sendo a mais comum a geração aleatória dos indivíduos. Esse método garante que a população inicial tenha uma boa diversidade, essencial para que o algoritmo explore diferentes áreas do espaço de soluções (Alam et al., 2020).

Entretanto, a inicialização totalmente aleatória pode levar a populações iniciais subótimas, resultando em uma convergência mais lenta e desempenho inferior ao longo do tempo, quando comparado a populações inicializadas por algoritmos de aproximação (Razip; Zakaria, 2017). Para mitigar esse problema, podem ser usadas técnicas mais sofisticadas, como inicialização baseada em heurísticas. Nessa abordagem, a população inicial é criada com base em uma heurística que já oferece uma solução viável, garantindo que o algoritmo genético comece sua busca a partir de uma população com soluções de qualidade razoável. Isso acelera o processo de convergência e melhora a eficiência do algoritmo.

Outra técnica usada é a inicialização mista, que combina a geração aleatória com heurísticas ou até com soluções preexistentes. Essa abordagem pode ser vantajosa porque mantém a diversidade inicial, necessária para evitar que o algoritmo convirja para soluções subótimas rapidamente, ao mesmo tempo em que incorpora boas soluções desde o começo.

Dessa forma, a fase de inicialização pode ser ajustada de acordo com as características do problema e o comportamento desejado do algoritmo, seja promovendo diversidade por meio de uma geração completamente aleatória ou já guiando o algoritmo com soluções mais direcionadas e otimizadas por heurísticas.

2.2.3 Avaliação

A Avaliação é uma etapa em que a aptidão de cada indivíduo da população é medida. A função de avaliação, ou função de aptidão, é responsável por determinar a qualidade de uma solução em relação ao problema sendo resolvido. Para cada indivíduo, um valor de aptidão é calculado, refletindo o quão bem essa solução atende aos critérios de otimização do problema.

Dependendo do problema, a função de avaliação pode variar bastante, sendo calculada a partir de critérios como distância, custo, tempo, entre outros. Em muitos casos, problemas complexos podem exigir funções de aptidão com múltiplos fatores, o que influencia diretamente na eficiência e no comportamento do algoritmo genético.

2.2.4 Seleção

A Seleção é uma fase crucial do algoritmo genético, na qual os indivíduos com melhor aptidão são escolhidos para se reproduzirem, gerando a próxima geração. O operador de seleção exerce papel fundamental ao direcionar o processo de busca, influenciando diretamente a velocidade de convergência e a qualidade das soluções encontradas (Shukla; Pandey; Mehrotra, 2015).

Um dos métodos de seleção mais utilizados é a Seleção por Torneio. Nesse método, uma quantidade de indivíduos é escolhida aleatoriamente da população para competir entre si. O indivíduo com a maior aptidão dentre os selecionados é o vencedor do torneio e, portanto, é escolhido para reprodução. O número de participantes de cada torneio (ou tamanho do torneio) pode variar, influenciando o grau de pressão seletiva no algoritmo.

A Figura 4 ilustra uma população inicial de cinco indivíduos com suas respectivas representações binárias e valores de aptidão. Três indivíduos foram selecionados aleatoriamente para participar do torneio. Destes, o indivíduo com a maior aptidão, 9, foi o vencedor e será o selecionado para reprodução.

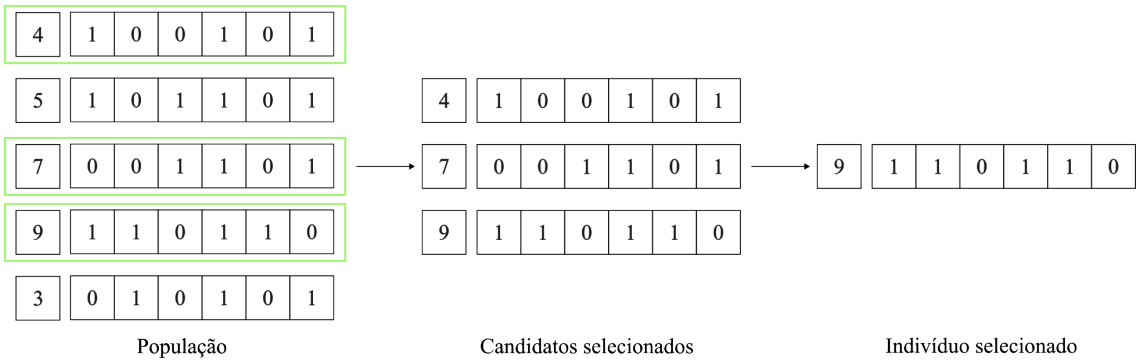

Figura 4 – Seleção por torneio. O primeiro valor indica aptidão.
Fonte: Elaborada pelo autor

A Seleção por Torneio é um método eficaz e amplamente utilizado em algoritmos genéticos, destacando-se pela simplicidade de implementação e pela eficiência em selecionar indivíduos com boa aptidão, uma vez que promove competições diretas entre candidatos da população (Oke et al., 2023).

2.2.5 Cruzamento

O Cruzamento (ou recombinação) é uma etapa essencial nos algoritmos genéticos, na qual soluções selecionadas (pais) são combinadas para produzir novos descendentes. Este operador pode tanto recombinar informações genéticas já existentes na população quanto introduzir novas características no espaço de busca, dependendo da sua implementação. Dessa forma, o cruzamento desempenha um papel crucial na evolução das soluções ao longo das gerações (Ezeani, 2015).

Durante o processo, dois indivíduos são selecionados para a reprodução, e suas cadeias de genes (representações binárias ou em outro formato) são combinadas para formar novos indivíduos. Existem diversas técnicas de cruzamento, entre elas o cruzamento de múltiplos pontos, em particular o cruzamento de dois pontos, no qual dois cortes são definidos aleatoriamente ao longo da cadeia genética dos pais. As seções entre esses pontos são trocadas entre os indivíduos, resultando em dois novos descendentes. Essa abordagem permite uma recombinação mais variada das características dos pais do que o cruzamento de ponto único, contribuindo para uma maior diversidade genética na população (Mendes, 2013).

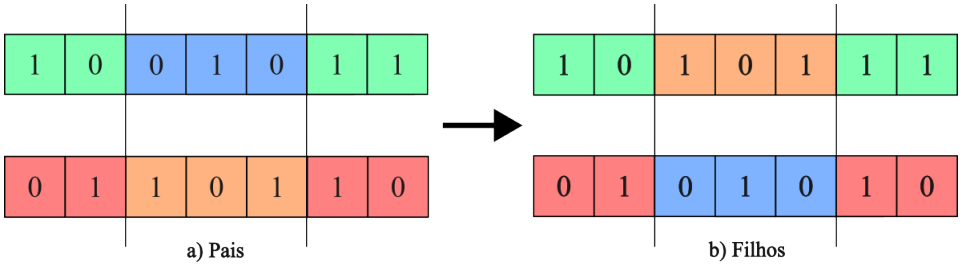

Figura 5 – Exemplo de cruzamento de dois pontos.
Fonte: Elaborada pelo autor

2.2.6 Mutação

A Mutação é uma etapa responsável por introduzir variações aleatórias nos indivíduos da população. Enquanto o cruzamento permite a recombinação de genes entre os pais, a mutação altera diretamente um ou mais genes de um indivíduo, garantindo que o algoritmo não fique preso em soluções subótimas e que novas regiões do espaço de soluções sejam exploradas. A mutação, além disso, desempenha um papel essencial na prevenção da convergência prematura para extremos locais, auxiliando na manutenção da diversidade genética ao longo das gerações (Isa et al., 2024).

O processo de mutação geralmente ocorre com uma probabilidade relativamente baixa, e pode ser visto como um mecanismo de correção que promove diversidade genética. Em representações binárias, que são amplamente utilizadas em algoritmos genéticos, a mutação é implementada invertendo o valor de um ou mais *bits*. Ou seja, se um *bit* for "0", ele será alterado para "1", e vice-versa. A Figura 6 ilustra o processo de mutação em um algoritmo genético simples.

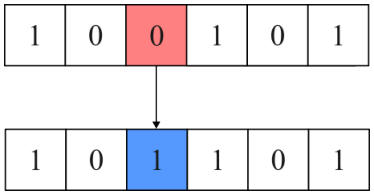

Figura 6 – Mutação em AG simples.
Fonte: Elaborada pelo autor

2.2.7 Nova Geração

A Seleção da Nova População, também conhecida como *Survive Selection*, é a etapa dos algoritmos genéticos responsável por decidir quais indivíduos da população atual, ou dos descendentes gerados, irão compor a nova população na próxima geração. Essa etapa é fundamental para garantir a continuidade do processo evolutivo, mantendo os indivíduos mais aptos e permitindo que o algoritmo melhore a solução ao longo do tempo.

Existem diferentes estratégias para realizar a seleção da nova população, cada uma com suas vantagens e desvantagens, dependendo da natureza do problema e dos objetivos do algoritmo. Entre as abordagens mais comuns estão:

- **Substituição Completa:** Toda a população antiga é substituída pelos novos indivíduos gerados a partir do cruzamento e da mutação.

- **Elitismo:** Para evitar a perda de boas soluções, o elitismo assegura que os melhores indivíduos da geração anterior sejam preservados na nova geração.

A seleção da nova população precisa garantir que as melhores soluções continuem no processo, mas também que novas soluções promissoras sejam exploradas. Esse equilíbrio é essencial para evitar que o algoritmo se estagne em mínimos locais ou perca a diversidade genética necessária para encontrar a solução ótima.

2.3 Algoritmo Memético

Algoritmos Meméticos (AM) são técnicas híbridas de otimização que integram os princípios dos algoritmos evolutivos com métodos de busca local. Essa abordagem permite que os AM explorem o espaço de soluções de forma mais eficiente, combinando estratégias de busca global e local. Essa sinergia entre diferentes abordagens de busca é um dos principais fatores que explicam o sucesso dos AM em diversos domínios de otimização (Moscato; Cotta, 2003).

Assim como os algoritmos genéticos, os algoritmos meméticos executam operações genéticas previamente definidas, promovendo a evolução da população ao longo de várias gerações. No entanto, em um AM, é possível incorporar uma etapa de busca local em algum ponto do processo evolutivo. Essa busca local tem o objetivo de refinar as soluções, melhorando sua qualidade ao explorar as vizinhanças próximas de cada solução atual.

As etapas do algoritmo memético seguem o mesmo fluxo do algoritmo genético, incluindo a mesma condição de término para o processo evolutivo. A busca local, por sua vez, é executada com uma condição de parada própria, que pode variar de acordo com sua implementação específica. O Algoritmo 2 apresenta o pseudocódigo de um algoritmo memético, enquanto a Figura 7 mostra um fluxograma com implementação de busca local após a mutação.

Algoritmo 2: Algoritmo Memético Básico

```
1  Gerar a população inicial P;
2  enquanto critério de parada não satisfeito faça
3  |  Avaliar os indivíduos de P;
4  |  Selecionar indivíduos de P para reprodução;
5  |  Aplicar cruzamento para gerar descendentes;
6  |  Aplicar mutação nos descendentes;
7  |  Busca Local;
8  |  Formar nova população P com base nos descendentes e/ou pais;
9  fim
10 retorna melhor solução encontrada
```

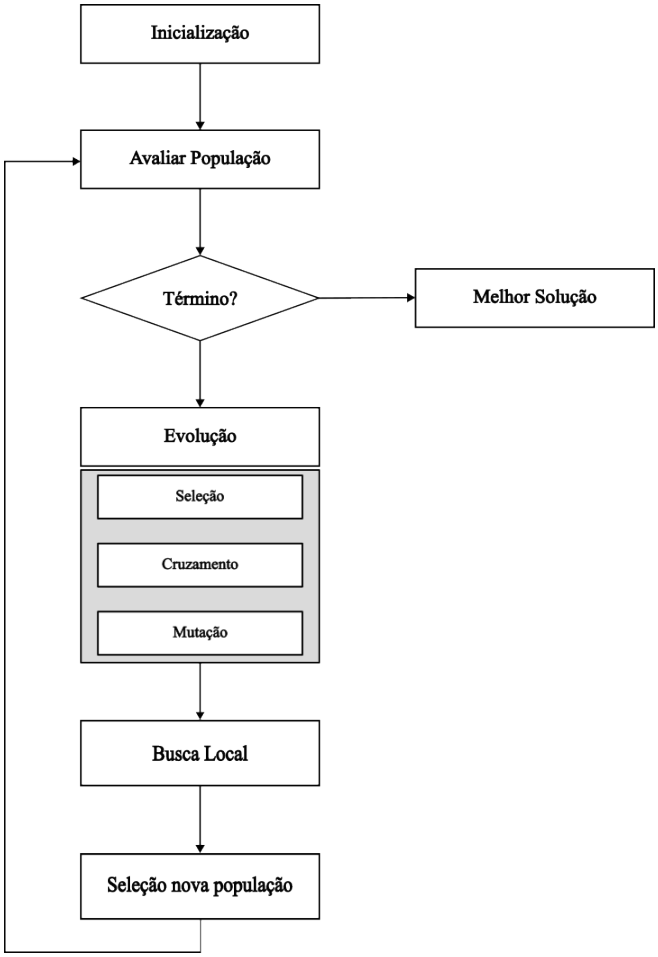

Figura 7 – Fluxograma de um AM com busca local após a mutação.
Fonte: Elaborada pelo autor

3 IMPLEMENTAÇÃO

Nesta seção, são discutidas as estratégias e técnicas utilizadas na implementação do sistema proposto, focando na aplicação de Algoritmos Genéticos e Algoritmos Meméticos. O desenvolvimento seguiu uma abordagem estruturada, que abrange desde a definição da população inicial até os mecanismos de evolução ao longo das gerações.

Os principais componentes do algoritmo são implementados com o intuito de garantir um equilíbrio entre a exploração do espaço de soluções e a intensificação das soluções já conhecidas, possibilitando a obtenção de soluções otimizadas. Além disso, técnicas de paralelismo com *OpenMP* são empregadas para melhorar o desempenho da execução, aproveitando o processamento concorrente para acelerar etapas específicas do algoritmo.

3.1 Representação

Neste trabalho, cada indivíduo é representado por uma sequência ordenada de números inteiros, indicando a ordem em que os clientes devem ser visitados, sem incluir explicitamente o depósito, que é sempre considerado o ponto de partida e de chegada de cada rota.

Além dessa sequência linear, as rotas são explicitamente separadas em um vetor de vetores, onde cada subvetor representa uma rota distinta, contendo os clientes que serão atendidos sequencialmente por um mesmo veículo. A separação é realizada automaticamente pelo procedimento *Split* (Vidal, 2016), que respeita a capacidade máxima do veículo, criando novas rotas sempre que a demanda acumulada ultrapassa essa capacidade.

Essa estrutura garante que todas as rotas sejam viáveis, ou seja, nenhuma rota excede a capacidade do veículo, não sendo admitidas soluções inviáveis. A atualização das rotas ocorre sempre que um novo indivíduo é criado ou modificado, como após operações de mutação, cruzamento ou aplicação de operadores locais. A representação explícita das rotas permite a aplicação direta de operadores de busca local inter-rotas, como o *Swap**. A Figura 8 ilustra a estrutura da representação utilizada.

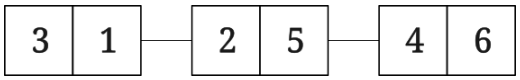

Figura 8 – Exemplo de representação por rotas adotado.
Fonte: Elaborada pelo autor

3.2 Avaliação

Para a função de avaliação, foi adotado o cálculo da distância euclidiana para medir a qualidade das soluções. Como descrito na seção anterior, a representação utilizada não inclui explicitamente o depósito, o que implica que cada indivíduo sempre inicia e termina sua rota no depósito.

A separação das rotas é realizada durante o cálculo da avaliação, levando em consideração o limite de capacidade do veículo. Sempre que esse limite é ultrapassado, uma nova rota é iniciada, garantindo a viabilidade da solução.

3.3 Inicialização

No presente estudo, fez-se uso de uma abordagem híbrida de inicialização, composta por técnicas de inicialização aleatória e heurística, que visam garantir a diversidade da população inicial e, ao mesmo tempo, inserir desde o início da evolução indivíduos com características que serão promissoras.

A população inicial é construída em partes iguais, sendo 50% da população gerada aleatoriamente e 50% utilizando heurísticas.

- **Inicialização Aleatória:** Cada indivíduo é gerado atribuindo uma permutação aleatória dos clientes, formando diferentes sequências de atendimento sem qualquer critério específico. Essa abordagem amplia a diversidade da população inicial, permitindo que o algoritmo explore uma gama variada de soluções desde o início.
- **Inicialização Baseada em Heurística:** Essa abordagem utiliza um processo estruturado para formar indivíduos de maneira mais eficiente. A estratégia adotada combina agrupamento via *K-means* e construção de rotas pelo método do Vizinho Mais Próximo (*Nearest Neighbour*). Primeiro, os clientes são agrupados com base em sua proximidade espacial utilizando o *K-means*. Em seguida, dentro de cada grupo formado, a ordem de visitação é definida utilizando o método do Vizinho Mais Próximo, criando sequências mais organizadas e potencialmente mais eficientes. Por fim, todas as rotas são unidas para formar um indivíduo completo.

As Subseções 3.3.1 e 3.3.2 detalham as estratégias adotadas para o funcionamento da inicialização por heurística.

3.3.1 K-means

O algoritmo *K-means* foi utilizado para o agrupamento dos usuários a partir de suas coordenadas geográficas, garantindo que os clientes que estão próximos sejam atendidos dentro de uma mesma rota. O método *K-means* é muito utilizado em problemas de agrupamento, pois minimiza a variação dentro de cada grupo e melhora a organização espacial das rotas geradas. Essa abordagem já foi aplicada com sucesso em algoritmos genéticos para problemas de roteamento, como o Problema do Caixeiro Viajante, demonstrando ganhos expressivos em qualidade da população inicial e desempenho da busca global (Deng; Liu; Zhou, 2015).

O funcionamento do *K-means* abrange os seguintes passos:

1. **Escolha dos Centros Iniciais:** Primeiramente, são selecionados aleatoriamente N clientes para servirem como os centros iniciais dos *clusters*.
2. **Atribuição dos Clientes aos *Clusters*:** Cada cliente é atribuído ao centro mais próximo, formando os primeiros agrupamentos de clientes.
3. **Atualização dos Centros:** Após a atribuição inicial, os centros dos *clusters* são recalculados como a média das coordenadas dos clientes pertencentes a cada grupo.
4. **Iteração:** O processo de atribuição e atualização dos centros é repetido até a convergência, ou seja, até que os centros dos *clusters* deixem de mudar significativamente entre uma iteração e outra.

Esse procedimento garante que os clientes dentro de um mesmo *cluster* estejam relativamente próximos uns dos outros, criando um ponto de partida mais estruturado para a construção das rotas.

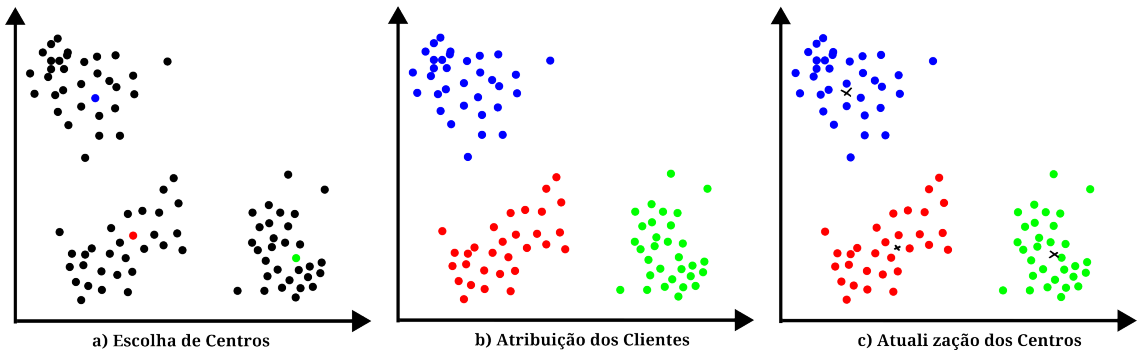

Figura 9 – Funcionamento do *K-means*.
Fonte: Elaborada pelo autor

3.3.2 Vizinho mais próximo

Após a formação dos *clusters* pelo algoritmo *K-means*, é necessário definir a sequência de visitação dos clientes dentro de cada grupo, de forma a estruturar percursos viáveis e eficientes. Para isso, foi adotado o método heurístico do Vizinho Mais Próximo

(*Nearest Neighbour*). Técnicas baseadas em vizinhos mais próximos têm sido amplamente utilizadas para a geração da população inicial em algoritmos genéticos aplicados a problemas de roteamento, pois favorecem a criação de indivíduos mais promissores desde o início, o que pode acelerar a convergência e melhorar a qualidade das soluções ao longo das gerações (Hassanat et al., 2018).

O algoritmo inicia a partir de um cliente aleatório escolhido dentro do *cluster*. A partir desse ponto, a rota é construída iterativamente, adicionando sempre o cliente mais próximo daquele que foi visitado por último, até que todos os clientes do *cluster* tenham sido incluídos na sequência. Dessa forma, cada trajeto se desenvolve priorizando visitas a clientes geograficamente mais próximos antes de considerar pontos mais distantes, resultando em percursos mais organizados e reduzindo a distância total percorrida.

A construção de rota pode ser observada na Figura 10, na qual, a partir de um nó inicial, seleciona-se iterativamente o nó mais próximo como o próximo destino. Esse procedimento é repetido sucessivamente até que todos os nós tenham sido visitados.

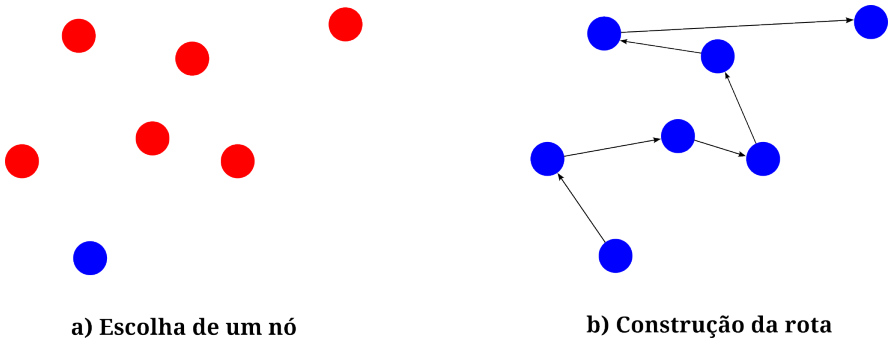

Figura 10 – Construção da rota utilizando o vizinho mais próximo.
Fonte: Elaborada pelo autor

Ao final do processo, as rotas geradas pelo método do Vizinho Mais Próximo são combinadas para formar um indivíduo completo. Como a representação adotada não inclui explicitamente o depósito, todas as soluções geradas são, por construção, viáveis dentro do contexto do problema.

3.4 Seleção

No processo de seleção, foi adotado o método de seleção por torneio, previamente descrito no capítulo anterior. Essa abordagem consiste em selecionar, de forma aleatória, um subconjunto de indivíduos da população e, a partir desse grupo, escolher os melhores candidatos para a reprodução com base em seus valores de aptidão.

Para este trabalho, definiu-se que o tamanho do subconjunto de candidatos ao torneio corresponde a um terço da população total, valor obtido após calibração de parâmetros. Dentro desse subconjunto, os indivíduos competem entre si e aquele com maior aptidão é selecionado como pai. A cada geração, são selecionados 4 indivíduos como pais, formando dois pares, cada um responsável por gerar dois filhos. Esse número foi escolhido de forma a manter uma taxa de substituição parcial, equilibrando a introdução de novos indivíduos com a preservação de soluções promissoras.

3.5 Cruzamento

O cruzamento foi implementado por meio do método de *Order Crossover* (OX), um operador de recombinação multiponto amplamente utilizado em representações baseadas em permutações. Nesse método, dada uma dupla de pais, $P1$ e $P2$, são selecionados aleatoriamente dois pontos dentro da sequência que delimitam um segmento a ser preservado. O primeiro descendente $P'1$ herda esse segmento diretamente de $P1$, enquanto o segundo descendente $P'2$ herda o mesmo segmento de $P2$.

Após a cópia do segmento central, a parte restante do cromossomo é preenchida com os genes do outro progenitor, iniciando-se pela posição imediatamente após o segundo ponto de corte e percorrendo circularmente o cromossomo. Para garantir que a permutação seja válida, os elementos ausentes são inseridos na mesma ordem em que aparecem no pai complementar, excluindo aqueles já presentes no segmento herdado. Dessa forma, $P'2$ recebe os genes restantes de $P1$, e $P'1$ recebe os genes restantes de $P2$, preservando a sequência relativa dos elementos e garantindo que não haja repetições.

A Figura 11 ilustra o funcionamento do método OX, com os pontos de corte definidos nos índices 2 e 5, ambos os descendentes herdam o mesmo segmento delimitado de um dos pais. Após a cópia desse trecho fixo, os genes restantes são preenchidos com os elementos do outro progenitor, respeitando a ordem em que aparecem e garantindo que nenhum gene seja repetido.

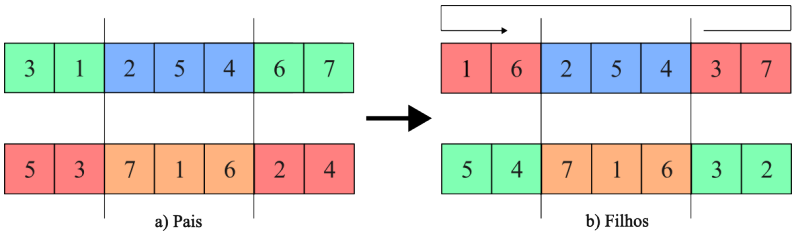

Figura 11 – Visualização do cruzamento OX.
Fonte: Elaborada pelo autor

3.6 Mutação

O operador de mutação adotado neste trabalho é a mutação por troca aleatória, ilustrada na Figura 12 . Nesse método, dois genes são selecionados aleatoriamente dentro da sequência global (considerando todas as rotas) e têm suas posições trocadas, modificando a ordem em que os clientes serão atendidos.

A probabilidade de ocorrência da mutação foi definida em 5%, com o objetivo de introduzir pequenas alterações nos indivíduos, promovendo a diversificação da população ao longo das gerações. Essa escolha buscou equilibrar a exploração do espaço de busca com a preservação das características estruturais das soluções.

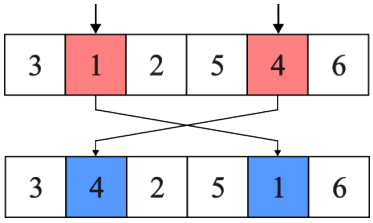

Figura 12 – Mutação de troca aleatória.
Fonte: Elaborada pelo autor

3.7 Nova Geração

Na etapa de seleção da nova geração, foi adotada a estratégia de elitismo, na qual dois terços dos melhores indivíduos da geração atual são preservados, enquanto os piores são substituídos pelos descendentes gerados durante o processo evolutivo. Essa abordagem garante que soluções de alta qualidade não sejam perdidas ao longo das iterações, mantendo a continuidade do aprimoramento da população.

3.8 Busca Local

No contexto do algoritmo memético proposto, a busca local desempenha um papel crucial na melhoria das soluções geradas pela fase global de busca, que é realizada por um algoritmo genético. Para essa etapa de busca local, optou-se pela utilização do Recozimento Simulado (SA, do inglês *Simulated Annealing*), uma meta-heurística que combina características de exploração e intensificação, permitindo escapar de ótimos locais e convergir para soluções de melhor qualidade.

O SA foi escolhido devido à sua capacidade de aceitar soluções piores durante o processo de busca, com uma probabilidade que diminui ao longo do tempo, simulando o processo de resfriamento de metais em metalurgia. A aceitação de soluções piores, controlada por um parâmetro de temperatura, permite que o algoritmo explore regiões do espaço de busca que poderiam ser negligenciadas por métodos de busca local mais convencionais (Kirkpatrick; Gelatt; Vecchi, 1983).

Após o processo de evolução do algoritmo genético, o SA é aplicado para refinar as soluções geradas. Para gerar novos vizinhos durante a busca local, foi utilizado o operador de busca local inter-rotas *Swap** (Vidal, 2022), que realiza trocas entre clientes de rotas diferentes, melhorando o potencial de intensificação e diversificação na busca por soluções de melhor qualidade.

3.9 Paralelização

A paralelização foi incorporada ao código com o objetivo de reduzir o tempo de execução dos algoritmos, sem comprometer a qualidade das soluções. Para isso, foi utilizada a biblioteca *OpenMP*, devido à sua integração com C++ e à simplicidade na inserção de diretivas de paralelização em regiões de código sequencial.

As regiões escolhidas para paralelização foram os trechos responsáveis pelo cruzamento e pela busca local. Esses pontos foram selecionados por apresentarem maior custo computacional, devido ao uso intensivo de laços *for*, e por permitirem a execução concorrente de iterações independentes. O Algoritmo 3 ilustra a aplicação do OpenMP na paralelização da busca local.

Algoritmo 3: Exemplo de OpenMP

```
#pragma omp parallel for;
para i ← 0 até |população| − 1 faça
    Busca Local;
fim
```

4 RESULTADOS

Nesta seção, são apresentados os parâmetros utilizados nos experimentos, bem como a descrição dos testes realizados e os resultados obtidos. Todos os experimentos foram executados em um computador equipado com um processador *Intel Core* I7-1185G7 (3 GHz, 4 núcleos, 8 *threads*), 16 GB de memória RAM, no sistema operacional *Linux Ubuntu*.

4.1 Descrição dos experimentos

Para os experimentos, foram usadas instâncias do conjunto de Uchoa et al. (2017). Esse conjunto de testes faz parte de um estudo que propôs novas instâncias para o CVRP, abordando limitações de *benchmarks* anteriores e oferecendo cenários mais desafiadores e realistas. Todas as instâncias utilizadas estão disponíveis em CVRPLIB (2025).

Para todas as instâncias, foram realizados dois experimentos para cada metaheurística: um utilizando a implementação sequencial e outro com a estratégia de paralelização proposta utilizando *OpenMP*. Esses testes tiveram como objetivo avaliar o impacto da paralelização na eficiência dos algoritmos em diferentes conjuntos de instâncias. Os parâmetros utilizados em todos os experimentos estão indicados na Tabela 1.

Tabela 1 – Parâmetros utilizados nos experimentos.

Parâmetros	**Valor**
Tamanho da População	25
Número de Gerações	1000
Mutação	5%
Tamanho do subconjunto da seleção por torneio	1/3 da população
Limite de gerações sem melhoria	200
Tempo máximo de execução	600 segundos
Temperatura inicial (*Simulated Annealing*)	10
Taxa de resfriamento (*Simulated Annealing*)	0.95

4.2 Comparação dos resultados

Nesta subseção, são analisados os resultados obtidos a partir dos experimentos descritos anteriormente. O objetivo é comparar o desempenho dos algoritmos genético e memético, avaliando sua eficácia e eficiência em diferentes conjuntos de instâncias. Além disso, também é analisado o impacto do uso do *OpenMP* no tempo de execução e na qualidade das soluções encontradas. Esta análise tem caráter descritivo e não tem como objetivo definir qual algoritmo é superior. A comparação conclusiva será realizada posteriormente por meio de testes estatísticos apropriados na Seção 4.3.

A avaliação dos algoritmos considera duas métricas principais:

- **Custo da solução:** representa a qualidade da solução encontrada, ou seja, a menor distância percorrida pelos veículos respeitando as restrições do problema.
- **Tempo de execução:** mede o tempo necessário para que o algoritmo encontre uma solução, permitindo avaliar se o uso de *OpenMP* possibilitou ganhos de tempo.

Além disso, a análise dos resultados será dividida em três partes abordadas nas Subseções 4.2.1, 4.2.2 e 4.2.3.

4.2.1 Comparação AG x AM

Para uma avaliação mais detalhada do desempenho dos algoritmos, cada teste foi executado 10 vezes, permitindo calcular a média das soluções obtidas. Inicialmente, foi utilizada a heurística de melhoria *2-opt* nas soluções geradas por meio do algoritmo memético, no entanto, os resultados não foram satisfatórios: além de aumentar o tempo de execução, não foram detectadas melhoras relevantes nas qualidades das soluções.

Diante desse panorama, o *2-opt* foi substituído pelo *Swap**, que se mostrou mais eficiente. O *Swap** apresentou melhor desempenho, alcançando soluções de qualidade superior, tornando-se a estratégia de busca local adotada na versão final do algoritmo memético. A Tabela 2 apresenta os resultados obtidos nesse experimento, onde os termos Inst., M.Sol., M.Res., Dif., T(s), correspondem, respectivamente, à instância, melhor solução conhecida, melhor resultado encontrado, diferença percentual em relação ao melhor valor conhecido e tempo de execução em segundos.

Tabela 2 – Resultados das Instâncias X.

Inst.	M. Sol.	Algoritmo Genético				Algoritmo Memético			
		Média	M.Res.	Dif.	T(s)	Média	M.Res.	Dif.	T(s)
X-n101-k25	27591	31177,37	30540,00	10,69%	1,4	29496,44	28668,40	3,90%	19,8
X-n106-k14	26362	28688,51	28259,60	7,20%	1,5	27273,24	27118,20	2,87%	27,0
X-n110-k13	14971	17800,56	17407,90	16,28%	1,4	16422,74	16249,40	8,54%	33,6
X-n115-k10	12747	15477,28	14990,40	17,60%	1,5	14409,40	14084,90	10,50%	32,4
X-n120-k6	13332	16446,73	16213,10	21,61%	1,5	15008,59	14454,90	8,42%	42,8
X-n125-k30	55539	63418,05	63010,10	13,45%	1,8	59034,47	58642,60	5,59%	26,9
X-n129-k18	28940	33875,96	33666,40	16,33%	1,7	31728,82	31166,00	7,69%	25,2
X-n134-k13	10916	13044,45	12794,30	17,21%	1,8	12095,63	11660,50	6,82%	39,8
X-n139-k10	13590	16908,05	16615,40	22,26%	1,8	15616,15	15014,70	10,48%	53,7
X-n143-k7	15700	20091,35	19616,40	24,95%	1,8	18741,75	18048,70	14,96%	41,0
X-n148-k46	43448	49815,63	49303,60	13,48%	2,2	45765,16	45266,50	4,19%	45,6
X-n153-k22	21220	27286,08	26990,00	27,19%	2,1	23381,55	22970,50	8,25%	47,7
X-n157-k13	16876	19231,74	19063,20	12,96%	2,0	17811,87	17498,50	3,69%	105,0
X-n162-k11	14138	17192,55	16947,30	19,87%	2,1	15974,12	15483,10	9,51%	70,7
X-n167-k10	20557	25033,86	24697,10	20,14%	2,2	23538,00	23139,50	12,56%	57,3
X-n172-k51	45607	51581,48	50152,30	9,97%	2,5	47733,67	47170,30	3,43%	56,9
X-n176-k26	47812	59818,38	59047,20	23,50%	2,4	51498,38	51061,90	6,80%	79,4
X-n181-k23	25569	28544,95	28200,30	10,29%	2,4	26491,74	26206,10	2,49%	129,9
X-n186-k15	24145	29039,92	28469,80	17,91%	2,4	26951,43	26542,40	9,93%	101,9
X-n190-k8	16980	20047,24	19550,80	15,14%	2,5	18846,18	18292,70	7,73%	84,9
X-n195-k51	44225	50074,76	49630,30	12,22%	2,8	47533,78	46717,70	5,64%	77,4
X-n200-k36	58578	64904,33	64380,10	9,90%	2,7	62215,57	61358,60	4,75%	89,0
X-n204-k19	19565	23064,55	22868,30	16,88%	2,7	22047,86	21267,70	8,70%	133,9
X-n209-k16	30656	36532,38	36184,00	18,03%	2,7	34346,75	33298,80	8,62%	137,9

Tabela 2 – Continuação.

Inst.	M. Sol.	Algoritmo Genético				Algoritmo Memético			
		Média	M.Res.	Dif.	T(s)	Média	M.Res.	Dif.	T(s)
X-n214-k11	10856	13898,64	13304,80	22,56%	2,8	12933,67	12565,10	15,74%	134,5
X-n219-k73	117595	120013,20	119649,00	1,75%	3,2	117840,40	117769,00	0,15%	308,5
X-n223-k34	40437	46107,94	45662,00	12,92%	3,0	44710,25	43918,60	8,61%	88,8
X-n228-k23	25742	32061,83	31272,60	21,48%	3,0	29478,78	28722,30	11,58%	132,8
X-n233-k16	19230	23639,71	23332,10	21,33%	3,1	22879,90	22585,30	17,45%	111,4
X-n237-k14	27042	32432,76	32171,50	18,97%	3,1	30289,04	29391,70	8,69%	267,9
X-n242-k48	82751	93100,05	92476,00	11,75%	3,4	89043,49	88343,90	6,76%	102,7
X-n247-k47	37274	44884,49	44298,10	18,84%	3,5	40391,60	40043,30	7,43%	116,2
X-n251-k28	38684	43986,13	43784,00	13,18%	3,4	42052,62	41435,00	7,11%	161,4
X-n256-k16	18839	22078,05	21568,20	14,49%	3,5	21481,69	20743,10	10,11%	147,9
X-n261-k13	26558	33126,83	32291,50	21,59%	3,4	31585,37	30961,40	16,58%	161,0
X-n266-k58	75478	83866,89	83181,50	10,21%	3,8	79929,29	79545,60	5,39%	199,3
X-n270-k35	35291	40493,26	40079,10	13,57%	3,7	38271,78	37696,00	6,81%	244,5
X-n275-k28	21245	24317,31	24069,40	13,29%	3,7	22910,63	22634,70	6,54%	291,8
X-n280-k17	33503	42473,16	42037,60	25,47%	3,7	39890,53	38285,80	14,28%	211,9
X-n284-k15	20215	24636,07	24398,10	20,69%	3,7	23734,31	23243,90	14,98%	201,3
X-n289-k60	95151	107391,20	107015,00	12,47%	4,1	103054,60	102504,00	7,73%	107,8
X-n294-k50	47161	54259,90	54006,40	14,51%	4,1	52336,37	51772,70	9,78%	131,6
X-n298-k31	34231	40871,81	40282,80	17,68%	4,0	39485,11	39008,90	13,96%	162,4
X-n303-k21	21736	26771,34	25980,30	19,53%	4,1	25227,01	24857,00	14,36%	248,4
X-n308-k13	25859	34095,65	33479,50	29,47%	4,2	31702,00	30121,10	16,48%	284,7
X-n313-k71	94043	106727,60	106346,00	13,08%	4,5	101426,20	100169,00	6,51%	178,4
X-n317-k53	78355	80972,98	80669,90	2,95%	4,4	79566,97	79183,00	1,06%	370,9
X-n322-k28	29834	35316,40	34940,40	17,12%	4,3	33910,96	32659,30	9,47%	270,5
X-n327-k20	27532	33585,66	33292,30	20,92%	4,4	32153,48	31638,80	14,92%	286,8
X-n331-k15	31102	37732,79	37259,30	19,80%	4,4	35212,76	33815,70	8,73%	357,3
X-n336-k84	139111	160877,20	159811,00	14,88%	5,0	151549,00	150230,00	7,99%	141,8
X-n344-k43	42050	48366,66	48117,00	14,43%	4,8	46285,23	45318,70	7,77%	373,1
X-n351-k40	25896	30049,64	29772,90	14,97%	4,9	29579,30	28897,20	11,59%	179,7
X-n359-k29	51505	59527,37	59178,60	14,90%	4,9	58288,35	57815,70	12,25%	151,5
X-n367-k17	22814	28693,35	28451,30	24,71%	5,1	27562,97	26917,50	17,99%	322,0
X-n376-k94	147713	152521,70	152308,00	3,11%	5,4	149125,50	148801,00	0,74%	600,6
X-n384-k52	65928	74597,85	73951,40	12,17%	5,3	71565,59	70969,50	7,65%	431,3
X-n393-k38	38260	44286,11	43988,10	14,97%	5,9	42977,24	42638,30	11,44%	365,4
X-n401-k29	66154	74033,60	73428,50	11,00%	5,7	72397,57	71636,90	8,29%	266,9
X-n411-k19	19712	25575,47	24996,80	26,81%	5,8	24486,63	23865,50	21,07%	419,8
X-n420-k130	107798	123651,90	122972,00	14,08%	6,8	115257,10	114416,00	6,14%	364,3
X-n429-k61	65449	73572,42	73143,50	11,76%	6,1	71666,50	70910,80	8,35%	325,4
X-n439-k37	36391	41733,25	41491,60	14,02%	6,1	40488,30	39898,40	9,64%	504,1
X-n449-k29	55233	65137,06	64594,80	16,95%	6,2	64183,98	63397,20	14,78%	252,4
X-n459-k26	24139	29543,48	29325,80	21,49%	6,4	28908,38	28338,20	17,40%	336,2
X-n469-k138	221824	251966,90	250757,00	13,04%	7,3	237375,70	235472,00	6,15%	500,8
X-n480-k70	89449	99103,19	98697,40	10,34%	7,1	96391,69	95788,70	7,09%	493,9
X-n491-k59	66483	75792,37	75249,00	13,19%	7,2	74971,24	74054,30	11,39%	297,0
X-n502-k39	69226	73787,84	73470,80	6,13%	7,7	71922,11	71527,60	3,32%	600,8
X-n513-k21	24201	30776,88	30441,00	25,78%	7,7	30172,62	29703,20	22,74%	430,4
X-n524-k153	154593	187874,30	186553,00	20,67%	8,4	164482,90	162966,00	5,42%	558,8
X-n536-k96	94846	106348,40	105954,00	11,71%	8,5	104405,00	102952,00	8,55%	480,7
X-n548-k50	86700	94594,53	94254,20	8,71%	7,8	92646,03	91856,20	5,95%	594,3
X-n561-k42	42717	51195,18	50634,30	18,53%	8,1	50420,97	49658,90	16,25%	400,1
X-n573-k30	50673	59345,63	58696,60	15,83%	8,5	57578,76	56956,40	12,40%	491,5
X-n586-k159	190316	213496,90	212787,00	11,81%	9,4	205489,00	204666,00	7,54%	579,0
X-n599-k92	108451	121348,50	120824,00	11,41%	9,0	118504,40	117787,00	8,61%	593,2
X-n613-k62	59535	69819,05	69336,60	16,46%	9,0	69194,97	68299,50	14,72%	381,9
X-n627-k43	62164	70153,09	69572,60	11,92%	10,0	68798,78	68076,70	9,51%	533,6
X-n641-k35	63682	74180,02	73873,60	16,00%	9,6	72610,55	72253,00	13,46%	583,0
X-n655-k131	106780	110566,00	110372,00	3,36%	10,5	109144,00	108759,00	1,85%	601,8
X-n670-k126	146332	182437,40	181131,00	23,78%	10,7	168284,70	166589,00	13,84%	588,6

Tabela 2 – Continuação.

Inst.	M. Sol.	Algoritmo Genético				Algoritmo Memético			
		Média	M.Res.	Dif.	T(s)	Média	M.Res.	Dif.	T(s)
X-n685-k75	68205	78929,90	78527,40	15,13%	10,5	78412,59	77462,00	13,57%	420,3
X-n701-k44	81923	93931,98	93597,30	14,25%	10,6	93499,03	92730,10	13,19%	455,7
X-n716-k35	43373	50694,86	50442,10	16,30%	11,4	50426,10	50048,50	15,39%	510,8
X-n733-k159	136187	154260,60	153632,00	12,81%	11,9	152215,10	151463,00	11,22%	492,5
X-n749-k98	77269	87786,44	87303,90	12,99%	11,9	87486,98	87145,00	12,78%	469,3
X-n766-k71	114417	140431,40	138460,00	21,01%	12,2	136784,10	134554,00	17,60%	526,2
X-n783-k48	72386	85049,72	**84082,90**	**16,16%**	12,1	84953,25	84229,40	16,36%	492,2
X-n801-k40	73305	84474,07	84086,50	14,71%	12,4	83071,70	82610,10	12,69%	601,7
X-n819-k171	158121	174972,00	174334,00	10,25%	14,4	174339,00	174012,00	10,05%	596,9
X-n837-k142	193737	213428,50	212812,00	9,85%	13,7	212017,30	211000,00	8,91%	600,1
X-n856-k95	88965	97799,94	97372,20	9,45%	13,7	96454,95	95908,30	7,80%	603,3
X-n876-k59	99299	110617,70	110018,00	10,79%	14,7	110464,30	109896,00	10,67%	495,6
X-n895-k37	53860	65935,50	65631,00	21,85%	14,4	65231,76	64389,10	19,55%	603,1
X-n916-k207	329179	365624,70	364024,00	10,59%	15,9	361027,10	358095,00	8,78%	602,2
X-n936-k151	132715	168159,20	166842,00	25,71%	16,2	162844,00	160901,00	21,24%	602,6
X-n957-k87	85465	94400,43	94129,00	10,14%	17,0	93413,63	92984,90	8,80%	604,4
X-n979-k58	118976	130711,00	**129897,00**	**9,18%**	20,3	130519,70	130012,00	9,28%	602,7
X-n1001-k43	72355	85579,63	85128,40	17,65%	16,5	85369,25	84734,60	17,11%	602,6

Com os resultados, observa-se que, na maioria dos casos, o algoritmo memético obteve soluções com menor custo (os casos em que algoritmo genético obteve menor custo estão destacados em negrito), conseguindo escapar de ótimos locais nas quais o algoritmo genético tende a ficar preso. No entanto, esse ganho teve como contrapartida um aumento relevante no tempo de execução. Em todos os casos, o algoritmo memético foi mais lento que o algoritmo genético. Observa-se ainda que, considerando as médias dos custos obtidos para cada instância, o algoritmo memético obteve uma diferença percentual média favorável de aproximadamente 4,21% em relação ao algoritmo genético, indicando uma tendência geral de melhor desempenho.

4.2.2 Paralelização

Nesse experimento, cada teste foi executado 10 vezes. O principal objetivo foi avaliar o impacto da paralelização no tempo de execução dos algoritmos e verificar se houve uma variação significativa na qualidade das soluções com relação aos experimentos sequenciais.

Tabela 3 – Resultados das Instâncias X com paralelização.

Inst.	M. Sol.	Algoritmo Genético				Algoritmo Memético			
		Média	M.Res.	Dif.	T(s)	Média	M.Res.	Dif.	T(s)
X-n101-k25	27591	31357,92	31131,00	12,83%	0,9	29576,43	28926,90	4,84%	15,5
X-n106-k14	26362	28898,02	28576,60	8,40%	0,7	27208,07	27044,70	2,59%	29,9
X-n110-k13	14971	17874,34	17521,80	17,04%	0,8	16586,65	16353,50	9,23%	22,2
X-n115-k10	12747	15430,78	15026,10	17,88%	0,8	14527,98	14209,20	11,47%	27,3
X-n120-k6	13332	16721,64	16323,20	22,44%	0,8	15363,50	14766,60	10,76%	26,4
X-n125-k30	55539	63403,57	62914,40	13,28%	1,1	59283,42	58813,90	5,90%	20,1
X-n129-k18	28940	33870,28	33517,90	15,82%	0,9	31535,03	31146,50	7,62%	24,1
X-n134-k13	10916	13023,18	12886,70	18,05%	0,9	12060,25	11824,00	8,32%	43,6
X-n139-k10	13590	16717,11	16278,70	19,78%	0,9	15661,19	15121,70	11,27%	51,1
X-n143-k7	15700	20066,08	19514,60	24,30%	0,9	18712,37	18328,40	16,74%	32,6

Tabela 3 – Continuação.

Inst.	M. Sol.	Algoritmo Genético				Algoritmo Memético			
		Média	M.Res.	Dif.	T(s)	Média	M.Res.	Dif.	T(s)
X-n148-k46	43448	49741,69	48824,80	12,38%	1,4	45849,19	45439,20	4,58%	34,5
X-n153-k22	21220	27135,71	26626,50	25,48%	1,2	23236,49	22730,70	7,12%	58,9
X-n157-k13	16876	19288,27	19136,20	13,39%	1,0	17775,13	17460,20	3,46%	83,5
X-n162-k11	14138	17120,73	16844,60	19,14%	1,0	16149,66	15759,80	11,47%	50,7
X-n167-k10	20557	25078,62	24885,00	21,05%	1,1	23325,76	22558,00	9,73%	84,4
X-n172-k51	45607	51824,54	50896,90	11,60%	1,6	47844,35	47345,40	3,81%	43,3
X-n176-k26	47812	59684,23	58397,50	22,14%	1,4	51485,15	50791,50	6,23%	64,4
X-n181-k23	25569	28555,27	28178,50	10,21%	1,3	26391,66	26154,90	2,29%	110,8
X-n186-k15	24145	29041,18	28853,20	19,50%	1,2	27049,98	26309,60	8,97%	81,4
X-n190-k8	16980	20053,82	19789,90	16,55%	1,2	18913,21	18641,70	9,79%	76,4
X-n195-k51	44225	50032,34	49512,30	11,96%	1,7	47634,08	47292,90	6,94%	48,7
X-n200-k36	58578	65058,08	64656,80	10,38%	1,5	62311,64	61282,20	4,62%	66,9
X-n204-k19	19565	23093,32	22847,50	16,78%	1,4	21890,03	21385,90	9,31%	119,4
X-n209-k16	30656	36536,77	36079,10	17,69%	1,4	34511,30	34024,50	10,99%	79,5
X-n214-k11	10856	13947,37	13493,50	24,30%	1,4	13072,55	12682,20	16,82%	92,1
X-n219-k73	117595	120018,90	119810,00	1,88%	1,9	117909,00	117763,00	0,14%	283,2
X-n223-k34	40437	46151,59	45437,00	12,36%	1,6	44764,42	44220,20	9,36%	66,0
X-n228-k23	25742	31791,11	30821,90	19,73%	1,6	29295,86	28446,80	10,51%	116,5
X-n233-k16	19230	23617,31	23360,60	21,48%	1,6	22995,99	22300,20	15,97%	67,4
X-n237-k14	27042	32421,80	31830,90	17,71%	1,5	30231,12	29958,20	10,78%	192,8
X-n242-k48	82751	93051,66	92001,20	11,18%	1,9	89142,88	88207,70	6,59%	65,5
X-n247-k47	37274	44624,81	44351,80	18,99%	2,1	40353,44	39796,30	6,77%	116,4
X-n251-k28	38684	44034,77	43436,20	12,28%	1,8	42184,38	41165,00	6,41%	141,9
X-n256-k16	18839	22078,67	21759,30	15,50%	1,7	21480,90	21133,80	12,18%	158,2
X-n261-k13	26558	33238,31	32637,20	22,89%	1,7	31578,28	30653,30	15,42%	117,0
X-n266-k58	75478	83859,32	82955,10	9,91%	2,2	79813,15	79153,40	4,87%	139,8
X-n270-k35	35291	40586,47	40314,10	14,23%	2,0	38289,90	37446,00	6,11%	206,5
X-n275-k28	21245	24386,62	24143,40	13,64%	1,9	23053,02	22850,40	7,56%	229,6
X-n280-k17	33503	42166,50	41493,70	23,85%	1,9	40117,88	39091,20	16,68%	134,9
X-n284-k15	20215	24793,07	24660,70	21,99%	1,8	23611,84	23003,70	13,80%	219,2
X-n289-k60	95151	107334,60	106657,00	12,09%	2,4	102776,20	101964,00	7,16%	110,8
X-n294-k50	47161	54216,39	53847,60	14,18%	2,3	52485,69	51497,30	9,19%	115,9
X-n298-k31	34231	41012,54	40667,00	18,80%	2,1	39519,57	39023,60	14,00%	175,7
X-n303-k21	21736	26776,75	26426,50	21,58%	2,1	25418,88	24631,50	13,32%	198,2
X-n308-k13	25859	34282,78	33732,80	30,45%	2,1	32012,27	30983,70	19,82%	211,7
X-n313-k71	94043	106912,70	106204,00	12,93%	2,7	101469,20	100833,00	7,22%	117,0
X-n317-k53	78355	80907,48	80787,40	3,10%	2,4	79430,09	79207,20	1,09%	339,8
X-n322-k28	29834	35312,08	35005,00	17,33%	2,2	33705,02	32860,50	10,14%	222,0
X-n327-k20	27532	33635,81	33310,40	20,99%	2,2	31805,45	31098,80	12,96%	290,2
X-n331-k15	31102	37577,65	36789,00	18,28%	2,2	35575,21	34611,50	11,28%	426,0
X-n336-k84	139111	160871,40	160044,00	15,05%	3,0	151208,30	150100,00	7,90%	107,3
X-n344-k43	42050	48515,40	48076,90	14,33%	2,6	46303,91	45026,60	7,08%	312,9
X-n351-k40	25896	29996,78	29583,50	14,24%	2,6	29488,98	28871,70	11,49%	159,1
X-n359-k29	51505	59444,93	58945,80	14,45%	2,5	58174,31	57609,90	11,85%	172,6
X-n367-k17	22814	28736,72	28337,90	24,21%	2,5	27458,20	26932,80	18,05%	224,2
X-n376-k94	147713	152418,10	151968,00	2,88%	3,1	148998,10	148642,00	0,63%	582,2
X-n384-k52	65928	74688,43	74334,90	12,75%	2,9	71487,70	70661,60	7,18%	411,3
X-n393-k38	38260	44162,16	43837,10	14,58%	3,3	42793,61	42067,70	9,95%	319,8
X-n401-k29	66154	74057,45	73548,40	11,18%	2,9	72703,76	71800,30	8,54%	169,8
X-n411-k19	19712	25629,47	25384,50	28,78%	3,0	24552,72	24026,10	21,89%	350,9
X-n420-k130	107798	123494,00	122395,00	13,54%	4,3	115407,40	114612,00	6,32%	252,3
X-n429-k61	65449	73403,13	72915,40	11,41%	3,3	71270,99	70566,50	7,82%	356,9
X-n439-k37	36391	41836,62	41571,40	14,24%	3,2	40244,85	39897,70	9,64%	510,7
X-n449-k29	55233	64962,29	64484,80	16,75%	3,2	64113,91	63249,20	14,51%	231,1
X-n459-k26	24139	29534,55	29123,60	20,65%	3,2	28817,41	28367,30	17,52%	348,4
X-n469-k138	221824	251725,70	250421,00	12,89%	4,6	237746,30	236069,00	6,42%	386,2
X-n480-k70	89449	98850,11	97974,10	9,53%	4,0	96246,66	95538,30	6,81%	368,2
X-n491-k59	66483	76092,60	75290,40	13,25%	3,8	75168,20	74005,80	11,32%	224,5

Tabela 3 – Continuação.

Inst.	M. Sol.	Algoritmo Genético				Algoritmo Memético			
		Média	M.Res.	Dif.	T(s)	Média	M.Res.	Dif.	T(s)
X-n502-k39	69226	73763,01	73369,10	5,98%	4,3	71909,04	71651,30	3,50%	583,6
X-n513-k21	24201	30901,63	30574,30	26,33%	4,1	30021,86	29554,00	22,12%	373,0
X-n524-k153	154593	187568,60	186722,00	20,78%	5,4	164220,10	162895,00	5,37%	508,0
X-n536-k96	94846	106418,70	105803,00	11,55%	5,1	104647,30	103553,00	9,18%	333,9
X-n548-k50	86700	94714,07	94452,70	8,94%	4,1	92480,57	91774,80	5,85%	592,1
X-n561-k42	42717	51279,48	50952,90	19,28%	4,3	50565,12	50210,90	17,54%	363,9
X-n573-k30	50673	59437,36	58736,50	15,91%	4,6	57634,45	56743,40	11,98%	408,6
X-n586-k159	190316	213681,40	212678,00	11,75%	5,9	204679,90	203511,00	6,93%	547,6
X-n599-k92	108451	121412,40	120672,00	11,27%	5,1	118201,60	117131,00	8,00%	558,7
X-n613-k62	59535	69913,68	69741,50	17,14%	5,0	69019,38	68287,40	14,70%	320,3
X-n627-k43	62164	70125,18	69806,40	12,29%	5,7	68457,05	67953,20	9,31%	570,9
X-n641-k35	63682	74137,97	73642,20	15,64%	5,1	72380,95	71699,30	12,59%	555,3
X-n655-k131	106780	110595,80	110344,00	3,34%	6,3	108859,10	108398,00	1,52%	601,1
X-n670-k126	146332	182459,10	180854,00	23,59%	6,6	168049,00	166633,00	13,87%	565,8
X-n685-k75	68205	79064,93	78578,90	15,21%	6,0	78133,94	77519,10	13,66%	403,6
X-n701-k44	81923	94048,79	93083,60	13,62%	5,7	93219,83	92833,20	13,32%	348,9
X-n716-k35	43373	50453,73	**49984,30**	**15,24%**	6,2	50486,50	50128,50	15,58%	413,4
X-n733-k159	136187	154302,70	153405,00	12,64%	7,3	152443,40	150155,00	10,26%	462,9
X-n749-k98	77269	87818,22	87419,90	13,14%	7,0	87473,77	86887,90	12,45%	361,2
X-n766-k71	114417	141193,30	139975,00	22,34%	7,0	136236,00	134291,00	17,37%	539,6
X-n783-k48	72386	85163,32	84657,80	16,95%	6,6	84733,45	83569,20	15,45%	418,9
X-n801-k40	73305	84421,80	83907,60	14,46%	6,7	82915,89	82118,70	12,02%	601,2
X-n819-k171	158121	174872,40	174363,00	10,27%	9,2	173660,50	173154,00	9,51%	567,7
X-n837-k142	193737	213706,90	213053,00	9,97%	8,3	211557,80	210810,00	8,81%	578,1
X-n856-k95	88965	97700,24	97259,90	9,32%	8,0	96654,77	95777,10	7,66%	602,6
X-n876-k59	99299	110619,10	**109515,00**	**10,29%**	8,6	110530,80	110114,00	10,89%	428,7
X-n895-k37	53860	65935,04	65650,10	21,89%	8,0	65261,06	64669,10	20,07%	601,9
X-n916-k207	329179	365802,00	365348,00	10,99%	10,0	360417,50	358433,00	8,89%	599,7
X-n936-k151	132715	168061,10	166504,00	25,46%	10,2	162660,30	160610,00	21,02%	602,0
X-n957-k87	85465	94449,87	94225,90	10,25%	10,2	93693,46	92940,80	8,75%	602,0
X-n979-k58	118976	130529,30	129807,00	9,10%	12,8	130553,80	129618,00	8,94%	544,5
X-n1001-k43	72355	85523,63	**85142,50**	**17,67%**	9,2	85571,21	85200,40	17,75%	590,2

Observando a Tabela 3, a maioria dos resultados permaneceu próxima aos valores obtidos na execução sequencial, tanto para o algoritmo genético quanto para o algoritmo memético. Em relação ao tempo de execução, o algoritmo genético apresentou redução em todas as instâncias. No caso do algoritmo memético, o tempo permaneceu próximo ao observado da versão sequencial na maioria dos casos, possivelmente em razão da complexidade do refinamento e a restrições como o limite de tempo e estagnação.

Observa-se também que, com os resultados obtidos na versão paralelizada, o algoritmo memético continuou apresentando melhor desempenho médio em relação ao genético, com uma diferença percentual média de aproximadamente 4,23% entre os valores médios de custo das soluções. Ressalta-se que a significância dessas diferenças será avaliada posteriormente por meio de testes estatísticos.

Assim, os resultados sugerem que a paralelização foi eficaz na redução do tempo de execução do algoritmo genético, mas teve impacto limitado no algoritmo memético. Além disso, as soluções obtidas com paralelismo permaneceram semelhantes às geradas sem paralelismo, indicando que a paralelização não afetou a qualidade das soluções.

4.2.3 Comparação com Trabalhos Relacionados

Para avaliar a qualidade das soluções obtidas, os melhores resultados dos algoritmos implementados são comparados com os de três métodos de referência da literatura. O primeiro é o algoritmo genético híbrido desenvolvido por Vidal (2022), o HGS-CVRP. O segundo método considerado é o algoritmo ILS-SP, descrito por Uchoa et al. (2017) no estudo de referência sobre os novos benchmarks do CVRP. O terceiro é o algoritmo POP, uma matheurística baseada em POPMUSIC com resolução de subproblemas via *branch-cut-and-price*, conforme proposto por Queiroga, Sadykov e Uchoa (2021). Os resultados utilizados correspondem à variante POP_2^2 com 2 horas de inicialização e 30 horas de execução. Além disso, a melhor solução conhecida para cada instância também é incluída, permitindo uma análise mais detalhada do desempenho dos algoritmos implementados em relação ao estado da arte. As comparações podem ser observadas na Tabela 4.

Tabela 4 – Comparação dos resultados com trabalhos relacionados.

Inst.	HGS-CVRP	ILS-SP	POP_2^2	AG	AM	M.Sol
X-n101-k25	27591	27591		30540,00	28668,40	27591
X-n106-k14	26364	26362		28259,60	27118,20	26362
X-n110-k13	14971	14971		17407,90	16249,40	14971
X-n115-k10	12747	12747		14990,40	14084,90	12747
X-n120-k6	13332	13332		16213,10	14454,90	13332
X-n125-k30	55539	55539		63010,10	58642,60	55539
X-n129-k18	28940	28948		33666,40	31166,00	28940
X-n134-k13	10916	10916		12794,30	11660,50	10916
X-n139-k10	13590	13590		16615,40	15014,70	13590
X-n143-k7	15700	15726		19616,40	18048,70	15700
X-n148-k46	43448	43448		49303,60	45266,50	43448
X-n153-k22	21225	21340		26990,00	22970,50	21220
X-n157-k13	16876	16876		19063,20	17498,50	16876
X-n162-k11	14138	14138		16947,30	15483,10	14138
X-n167-k10	20557	20562		24697,10	23139,50	20557
X-n172-k51	45607	45607		50152,30	47170,30	45607
X-n176-k26	47812	48140		59047,20	51061,90	47812
X-n181-k23	25569	25569		28200,30	26206,10	25569
X-n186-k15	24145	24145		28469,80	26542,40	24145
X-n190-k8	16980	17085		19550,80	18292,70	16980
X-n195-k51	44225	44225		49630,30	46717,70	44225
X-n200-k36	58578	58626		64380,10	61358,60	58578
X-n204-k19	19565	19570		22868,30	21267,70	19565
X-n209-k16	30656	30667		36184,00	33298,80	30656
X-n214-k11	10856	10985		13304,80	12565,10	10856
X-n219-k73	117595	117595		119649,00	117769,00	117595
X-n223-k34	40437	40471		45662,00	43918,60	40437
X-n228-k23	25742	25743		31272,60	28722,30	25742
X-n233-k16	19230	19266		23332,10	22585,30	19230
X-n237-k14	27042	27042		32171,50	29391,70	27042
X-n242-k48	82771	82774		92476,00	88343,90	82751
X-n247-k47	37274	37289		44298,10	40043,30	37274
X-n251-k28	38684	38727		43784,00	41435,00	38684
X-n256-k16	18839	18880		21568,20	20743,10	18839
X-n261-k13	26558	26706		32291,50	30961,40	26558
X-n266-k58	75478	75478		83181,50	79545,60	75478
X-n270-k35	35303	35324		40079,10	37696,00	35291
X-n275-k28	21245	21245		24069,40	22634,70	21245
X-n280-k17	33506	33624		42037,60	38285,80	33503
X-n284-k15	20231	20295		24398,10	23243,90	20215

Tabela 4 – Continuação.

Inst.	HGS-CVRP	ILS-SP	POP_2^2	AG	AM	M.Sol
X-n289-k60	95242	95315		107015,00	102504,00	95151
X-n294-k50	47167	47190		54006,40	51772,70	47161
X-n298-k31	34231	34239		40282,80	39008,90	34231
X-n303-k21	21739	21812	21738	25980,30	24857,00	21736
X-n308-k13	25862	25901	25859	33479,50	30121,10	25859
X-n313-k71	94045	94192	94044	106346,00	100169,00	94043
X-n317-k53	78355	78355	78355	80669,90	79183,00	78355
X-n322-k28	29834	29877	29834	34940,40	32659,30	29834
X-n327-k20	27532	27599	27532	33292,30	31638,80	27532
X-n331-k15	31102	31105	31102	37259,30	33815,70	31102
X-n336-k84	139205	139197	139125	159811,00	150230,00	139111
X-n344-k43	42061	42146	42050	48117,00	45318,70	42050
X-n351-k40	25924	26021	25896	29772,90	28897,20	25896
X-n359-k29	51566	51706	51505	59178,60	57815,70	51505
X-n367-k17	22814	22902	22814	28451,30	26917,50	22814
X-n376-k94	147713	147713	147713	152308,00	148801,00	147713
X-n384-k52	65997	66116	65941	73951,40	70969,50	65928
X-n393-k38	38260	38298	38260	43988,10	42638,30	38260
X-n401-k29	66209	66453	66178	73428,50	71636,90	66154
X-n411-k19	19716	19792	19712	24996,80	23865,50	19712
X-n420-k130	107810	107798	107798	122972,00	114416,00	107798
X-n429-k61	65484	65563	65455	73143,50	70910,80	65449
X-n439-k37	36395	36395	36395	41491,60	39898,40	36391
X-n449-k29	55306	55761	55262	64594,80	63397,20	55233
X-n459-k26	24139	24209	24139	29325,80	28338,20	24139
X-n469-k138	221916	221909	221824	250757,00	235472,00	221824
X-n480-k70	89498	89694	89449	98697,40	95788,70	89449
X-n491-k59	66569	66965	66489	75249,00	74054,30	66483
X-n502-k39	69230	69284	69226	73470,80	71527,60	69226
X-n513-k21	24201	24332	24201	30441,00	29703,20	24201
X-n524-k153	154646	154709	154593	186553,00	162966,00	154593
X-n536-k96	95040	95524	94915	105954,00	102952,00	94846
X-n548-k50	86710	86710	86700	94254,20	91856,20	86700
X-n561-k42	42726	42952	42717	50634,30	49658,90	42717
X-n573-k30	50757	51092	50733	58696,60	56956,40	50673
X-n586-k159	190470	190612	190316	212787,00	204666,00	190316
X-n599-k92	108605	109056	108453	120824,00	117787,00	108451
X-n613-k62	59636	60229	59536	69336,60	68299,50	59535
X-n627-k43	62238	62783	62223	69572,60	68076,70	62164
X-n641-k35	63782	64462	63763	73873,60	72253,00	63682
X-n655-k131	106785	106780		110372,00	108759,00	106780
X-n670-k126	146640	147045		181131,00	166589,00	146332
X-n685-k75	68288	68646		78527,40	77462,00	68205
X-n701-k44	82075	82888		93597,30	92730,10	81923
X-n716-k35	43459	44021		50442,10	50048,50	43373
X-n733-k159	136323	136832		153632,00	151463,00	136187
X-n749-k98	77563	77952		87303,90	87145,00	77269
X-n766-k71	114679	115443		138460,00	134554,00	114417
X-n783-k48	72704	73447		84082,90	84229,40	72386
X-n801-k40	73396	73830		84086,50	82610,10	73305
X-n819-k171	158391	159164		174334,00	174012,00	158121
X-n837-k142	194103	194804		212812,00	211000,00	193737
X-n856-k95	88986	89060		97372,20	95908,30	88965
X-n876-k59	99596	100177		110018,00	109896,00	99299
X-n895-k37	54023	54712		65631,00	64389,10	53860
X-n916-k207	329572	330639		364024,00	358095,00	329179
X-n936-k151	133121	133592		166842,00	160901,00	132715
X-n957-k87	85506	85697		94129,00	92984,90	85465
X-n979-k58	119180	119994		129897,00	130012,00	118976

Tabela 4 – Continuação.

Inst.	HGS-CVRP	ILS-SP	POP_2^2	AG	AM	M.Sol
X-n1001-k43	72678	73776		85128,40	84734,60	72355

4.3 Testes Estatísticos

Esta seção tem como objetivo verificar se as diferenças nos resultados e nos tempos de execução observadas entre os algoritmos são estatisticamente significativas. Para isso, foram aplicado testes estatísticos como *Shapiro-Wilk*, teste t pareado e teste de *Wilcoxon*. Os testes foram aplicados em pares, conforme descrito a seguir:

- **Algoritmo Genético vs. Algoritmo Memético:** comparação dos resultados e tempos de execução entre os dois algoritmos.
- **Algoritmo Genético vs. Algoritmo Genético Paralelo:** comparação entre a versão sequencial e paralela do algoritmo genético.
- **Algoritmo Memético vs. Algoritmo Memético Paralelo:** comparação entre a versão sequencial e paralela do algoritmo memético.
- **Algoritmo Genético Paralelo vs. Algoritmo Memético Paralelo:** comparação entre as versões paralelas dos dois algoritmos.

4.3.1 Testes Utilizados

Nesta subseção, são apresentados os testes estatísticos empregados para verificar se as diferenças nos resultados (qualidade das soluções) e nos tempos de execução entre os algoritmos são estatisticamente significativas.

Teste de *Shapiro-Wilk*: este teste foi utilizado para verificar a normalidade das diferenças entre os pares de amostras (por exemplo, diferença entre os resultados do Algoritmo Genético e do Algoritmo Memético para cada instância). A hipótese nula do teste assume que os dados seguem uma distribuição normal. Se o p-valor for menor que o nível de significância ($\alpha = 0{,}05$), rejeita-se a hipótese nula, indicando que os dados não seguem uma distribuição normal.

Teste t pareado: caso os dados tenham sido considerados normalmente distribuídos (isto é, o teste de *Shapiro-Wilk* não rejeitou a hipótese nula), aplicou-se o teste t pareado. Esse teste é paramétrico e compara as médias de duas amostras dependentes (ou relacionadas), avaliando se há evidência estatística de que a média das diferenças entre os pares é diferente de zero. Ele assume que as diferenças seguem uma distribuição normal.

Teste de *Wilcoxon*: quando o teste de *Shapiro-Wilk* indicou que os dados não seguiam uma distribuição normal, foi utilizado o teste não paramétrico de *Wilcoxon* para amostras pareadas. Esse teste avalia se há uma diferença significativa entre duas distribuições relacionadas, sem assumir normalidade.

A escolha entre os testes t pareado e *Wilcoxon* foi feita individualmente para cada par de algoritmos e para cada métrica (resultado e tempo), conforme o resultado do teste de normalidade. O nível de significância adotado para todos os testes foi de 0,05.

4.3.2 Resultados

Os testes estatísticos realizados para comparar a qualidade das soluções obtidas entre os diferentes pares de algoritmos indicaram que, nas comparações entre o AG e o AM, bem como entre o AG paralelo e o AM paralelo, a maioria das instâncias apresentou diferença estatisticamente significativa ($p < 0{,}05$). Em todas as situações em que houve diferença significativa, o AM chegou a melhores resultados. Mesmo nos casos em que a diferença não foi estatisticamente significativa, o AM alcançou soluções de qualidade superior, embora, nesse cenário, não se possa afirmar uma diferença relevante.

Na comparação entre o AG e sua versão paralela, a maioria das instâncias não apresentou diferença estatisticamente significativa, indicando que as soluções obtidas foram semelhantes. O mesmo comportamento foi observado na comparação entre o AM e o AM paralelo, sugerindo que o paralelismo, neste caso, não impactou de forma relevante a qualidade das soluções encontradas.

Em relação ao tempo de execução, os testes indicaram diferenças estatisticamente significativas em todas as instâncias comparadas entre o AG e o AM, sendo que o AG apresentou tempos de execução inferiores em todas as situações. Na comparação entre o AG e o AG paralelo, todas as instâncias também indicaram diferença estatística, com o AG paralelo apresentando desempenho superior. Ou seja, nessa situação a utilização de OpenMP se mostrou vantajosa pois levou a uma redução de tempo de execução com resultados sem diferença significativa. O mesmo padrão foi observado na comparação entre o AG paralelo e o AM paralelo, com diferença significativa em todas as instâncias e vantagem para o AG paralelo.

Por fim, na comparação entre o AM e o AM paralelo, a maioria das instâncias não apresentou diferença estatisticamente significativa quanto ao tempo de execução, o que sugere que o paralelismo não proporcionou ganho relevante de desempenho para o AM nesse aspecto.

5 CONCLUSÃO

Neste trabalho, foi apresentada uma comparação de desempenho entre o Algoritmo Genético e o Algoritmo Memético para resolver o Problema de Roteamento de Veículos Capacitados. Para resolver o problema, adotou-se uma estratégia de inicialização mista, combinando métodos aleatórios e heurísticos, visando iniciar o processo com uma população inicial de boa qualidade. No Algoritmo Memético, foi utilizada uma combinação de *Simulated Annealing* com o operador de vizinhança *Swap**, possibilitando o refinamento das soluções. Além disso, ambos os algoritmos foram paralelizados com o uso de *OpenMP*, visando reduzir o seu tempo de execução.

Para avaliar o comportamento das estratégias implementadas, foram utilizadas instâncias propostas por Uchoa et al. (2017), que oferecem cenários mais realistas e desafiadores. Os resultados obtidos indicaram que, na maioria dos casos, o Algoritmo Memético obteve soluções melhores, embora com um tempo de execução significativamente maior. O mesmo padrão se repetiu nos testes com paralelização, nos quais, na maioria dos casos, o Algoritmo Memético apresentou melhor desempenho em termos de qualidade da solução.

A paralelização levou a uma redução nos tempos de execução somente no Algoritmo Genético. A estratégia de paralelização adotada para ambos algoritmos não foi adequada para o Algoritmo Memético, cujas versões sequencial e paralela obtiveram tempos de execução semelhantes. Esse comportamento pode ser explicado pelo baixo aproveitamento do paralelismo e pelas definições dos parâmetros, como o tempo máximo de execução e o limite de estagnação.

De modo geral, observou-se que o Algoritmo Memético apresentou desempenho superior ao Algoritmo Genético na obtenção de soluções de melhor qualidade. Embora a análise estatística tenha indicado uma diferença significativa entre os tempos de execução das versões sequencial e paralela, na prática, essa diferença foi pequena. Isso se deve a limitações no aproveitamento do paralelismo e à configuração dos parâmetros do algoritmo.

Como trabalhos futuros, recomenda-se explorar estratégias mais eficientes de paralelização para o Algoritmo Memético, em especial para a etapa de busca local, que atualmente representa a parte mais custosa do algoritmo. Além disso, sugere-se explorar parâmetros que possam melhorar o desempenho do Algoritmo Memético, como o limite de tempo, que restringiu tanto a qualidade dos resultados obtidos quanto o tempo de execução.

REFERÊNCIAS

ALAM, T. et al. **Genetic Algorithm: Reviews, Implementations, and Applications**. 2020. 7

CVRPLIB. **CVRPLIB - All Instances**. 2025. Disponível em: <http://vrp.galgos.inf.puc-rio.br/index.php/en/>. 17

DANTZIG, G. B.; RAMSER, J. H. The truck dispatching problem. **Management Science**, 1959. 2

DENG, Y.; LIU, Y.; ZHOU, D. An improved genetic algorithm with initial population strategy for symmetric tsp. **Mathematical Problems in Engineering**, v. 2015, n. 1, p. 212794, 2015. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1155/2015/212794>. 13

EZEANI, I. Comparative performances of crossover functions in genetic algorithms. **International Journal of Scientific & Engineering Research**, Volume 6, p. 1371–1394, 02 2015. 8

HASSANAT, A. B. et al. An improved genetic algorithm with a new initialization mechanism based on regression techniques. **Information**, v. 9, n. 7, 2018. ISSN 2078-2489. Disponível em: <https://www.mdpi.com/2078-2489/9/7/167>. 14

HU, H. System parameter optimization based on genetic algorithm. **International Journal of Mechanical and Electrical Engineering**, v. 2, p. 11–16, 05 2024. 2

ISA, F. M. et al. A review of genetic algorithm: Operations and applications. **Journal of Advanced Research in Applied Sciences and Engineering Technology**, v. 40, n. 1, p. 1–34, Feb. 2024. Disponível em: <https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/article/view/3337>. 9

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. **Science**, v. 220, n. 4598, p. 671–680, 1983. Disponível em: <https://www.science.org/doi/abs/10.1126/science.220.4598.671>. 16

LAPORTE, G. The vehicle routing problem: An overview of exact and approximate algorithms. **European Journal of Operational Research**, v. 59, n. 3, p. 345–358, 1992. ISSN 0377-2217. Disponível em: <https://www.sciencedirect.com/science/article/pii/037722179290192C>. 3

LE, K.; WEI, T.; TAGALOGON, J. **Genetic Algorithm and Application**. 2023. 4

MENDES, J. A comparative study of crossover operators for genetic algorithms to solve the job shop scheduling problem. **WSEAS Transactions on Computers**, v. 12, p. 164–173, 04 2013. 8

MOSCATO, P.; COTTA, C. A gentle introduction to memetic algorithms. In: **Handbook of metaheuristics**. [S.l.]: Springer, 2003. p. 105–144. 10

MUBARAK, A.; MAWENGKANG, H.; SUWILO, S. Enhancing vehicle routing efficiency through branch and bound and heuristic methods. **sinkron**, v. 8, p. 1463–1472, 07 2024. 2

NOON, C. E.; MITTENTHAL, J.; PILLAI, R. A tssp + 1 decomposition strategy for the vehicle routing problem. **European Journal of Operational Research**, 1994. 3

OKE, S. A. et al. Exploiting tournament selection-based genetic algorithm in integrated ahp-taguchi analyses-ga method for wire electrical discharge machining of az91 magnesium alloy. **IJIEM (Indonesian Journals of Industrial Engineering and Management)**, v. 4, n. 1, p. 1–1, 2023. 8

QUEIROGA, E.; SADYKOV, R.; UCHOA, E. A POPMUSIC matheuristic for the capacitated vehicle routing problem. **Computers and Operations Research**, Elsevier, v. 136, n. 105475, dez. 2021. Disponível em: <https://inria.hal.science/hal-02994210>. 23

RAZIP, H.; ZAKARIA, M. N. Combining approximation algorithm with genetic algorithm at the initial population for np-complete problem. In: . [S.l.]: IEEE, 2017. 7

SHUKLA, A.; PANDEY, H.; MEHROTRA, D. Comparative review of selection techniques in genetic algorithm. In: **Comparative Review of Selection Techniques in Genetic Algorithm**. [S.l.: s.n.], 2015. 7

TOTH, P.; VIGO, D. **The Vehicle Routing Problem**. [S.l.]: Society for Industrial and Applied Mathematics, 2002. 2

UCHOA, E. et al. New benchmark instances for the capacitated vehicle routing problem. **European Journal of Operational Research**, v. 257, n. 3, p. 845–858, 2017. ISSN 0377-2217. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0377221716306270>. 17, 23, 27

VIDAL, T. Technical note: Split algorithm in o(n) for the capacitated vehicle routing problem. **Computers & Operations Research**, v. 69, p. 40–47, 2016. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054815002798>. 12

VIDAL, T. Hybrid genetic search for the cvrp: Open-source implementation and swap* neighborhood. **Computers & Operations Research**, v. 140, p. 105643, 2022. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S030505482100349X>. 16, 23

WILSON, A. J. et al. A review on memetic algorithms and its developments. **Electrical and Automation Engineering**, v. 1, n. 1, p. 7–12, 2022. 2