Uma Abordagem Híbrida de Algoritmo Genético com Simulated Annealing na Solução do Problema do Roteamento de Veículos Capacitados

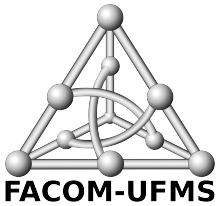
Caio Miguel G. Darzi

Felipe dos Santos Moreira

Vitor de Assis Ramos

Trabalho de Conclusão de Curso

Orientação
Prof. Dra. Bianca de Almeida Dantas



Faculdade de Computação
Universidade Federal de Mato Grosso do Sul

Campo Grande - MS
2025

Resumo

O Problema de Roteamento de Veículos Capacitado (PRVC) é um desafio de otimização combinatória NP-difícil, fundamental para a eficiência logística, que busca minimizar os custos de rotas de entrega respeitando a capacidade de carga dos veículos. Dada a sua complexidade computacional, a utilização de meta-heurísticas é uma abordagem eficaz. Este trabalho propõe uma metodologia híbrida em duas fases para a resolução do PRVC. Primeiramente, um Algoritmo Genético (AG) é executado para realizar uma busca global no espaço de soluções, explorando diversas regiões para gerar uma solução inicial de alta qualidade. Posteriormente, esta solução gerada pelo AG é utilizada como ponto de partida para o algoritmo *Simulated Annealing* (SA), que aplica um processo de busca local intensiva para refinar a rota, explorando sua vizinhança e utilizando seu mecanismo probabilístico para tentar escapar de ótimos locais. A hibridização busca aliar o poder de exploração do Algoritmo Genético com a capacidade de intensificação do *Simulated Annealing*, visando encontrar soluções de maior qualidade do que as obtidas pela aplicação isolada de cada meta-heurística.

Palavras-chave: *PRVC*, *Algoritmo Genético*, *Simulated Annealing*, *Hibridização de Meta-heurísticas*.

Abstract

The Capacitated Vehicle Routing Problem (CVRP) is an NP-hard combinatorial optimization challenge, fundamental to logistical efficiency, which seeks to minimize the costs of delivery routes while respecting vehicle load capacities. Given its computational complexity, the use of metaheuristics is an effective approach. This work proposes a two-phase hybrid methodology for solving the CVRP. First, a Genetic Algorithm (GA) is executed to perform a global search of the solution space, exploring diverse regions to generate a high-quality initial solution. Subsequently, the solution generated by the GA is used as the starting point for the Simulated Annealing (SA) algorithm, which applies an intensive local search process to refine the route, exploring its neighborhood and using its probabilistic mechanism to escape local optima. This hybridization aims to combine the exploratory power of the Genetic Algorithm with the intensification capability of Simulated Annealing, with the goal of finding higher-quality solutions than those obtained by the isolated application of each metaheuristic.

Keywords: *CVRP*, *Route Optimization*, *Genetic Algorithm*, *Simulated Annealing*, *Metaheuristic Hybridization*.

CAPÍTULO 1

Introdução

No cenário econômico atual, caracterizado por uma alta competitividade e pela globalização dos mercados, a eficiência logística tornou-se um fator determinante para o sucesso das organizações. O transporte de mercadorias representa uma parcela significativa dos custos logísticos totais, impactando diretamente o preço final dos produtos e o nível de serviço oferecido aos clientes. Neste contexto, a otimização das rotas de distribuição é essencial não apenas para a redução de custos operacionais, como consumo de combustível e manutenção de frota, mas também para o cumprimento de prazos e a satisfação do consumidor final.

Dentre os modelos matemáticos que visam solucionar questões de transporte, destaca-se o Problema de Roteamento de Veículos Capacitado (PRVC), ou do inglês *Capacitated Vehicle Routing Problem* (CVRP). Formalmente proposto por Dantzig and Ramser (1959), o PRVC é uma generalização do clássico Problema do Caixeiro Viajante (*Traveling Salesman Problem* – TSP), adicionando restrições de capacidade à frota de veículos. O objetivo central consiste em determinar um conjunto de rotas que atenda à demanda de todos os clientes, respeitando a capacidade máxima de carga de cada veículo e minimizando o custo total, geralmente associado à distância percorrida (Toth and Vigo, 2014a).

Apesar de sua ampla aplicabilidade prática, o PRVC pertence à classe de problemas de otimização combinatória conhecidos como NP-difíceis (*NP-hard*). Isso significa que, à medida que o número de clientes aumenta, a complexidade para encontrar a solução exata cresce exponencialmente, tornando inviável o uso de métodos exatos para instâncias de grande porte encontradas em operações reais. Christofides et al. (1979) já apontavam que, para tamanhos onde alternativas exatas teriam tempo de execução proibitivo, o uso de métodos aproximativos é necessário.

Para contornar essa complexidade computacional, a literatura tem se voltado para o uso de meta-heurísticas. Estas técnicas não garantem a solução ótima global, mas são capazes de encontrar soluções de boa qualidade em um tempo computacional aceitável (Bräysy and Gendreau, 2005). Dentre as meta-heurísticas mais estudadas, destacam-se o Algoritmo Genético (AG), inspirado na evolução biológica e capaz de realizar uma busca global eficiente explorando diversas regiões do espaço de soluções (Goldberg, 1989), e o *Simulated Annealing* (SA), baseado na termodinâmica, que se destaca pela

capacidade de intensificação (busca local) e refinamento de soluções (Kirkpatrick et al., 1983).

No entanto, a aplicação isolada dessas meta-heurísticas pode apresentar limitações. O AG, embora excelente na exploração (*exploration*), pode ter dificuldades em realizar o ajuste fino da solução final. Por outro lado, o SA é poderoso na intensificação (*exploitation*), mas pode ficar preso em ótimos locais se a solução inicial for de baixa qualidade.

Diante desse cenário, este trabalho propõe uma metodologia híbrida sequencial que utiliza o Algoritmo Genético como gerador de uma solução inicial de alta qualidade. Essa solução, após passar pelo processo evolutivo do AG, serve como ponto de partida para o *Simulated Annealing*, que aplica sua capacidade de intensificação para refinar o resultado e buscar um ótimo local de melhor qualidade.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo geral deste trabalho é desenvolver e avaliar uma metodologia híbrida sequencial, combinando um Algoritmo Genético e o *Simulated Annealing*, para a resolução do Problema de Roteamento de Veículos Capacitado.

1.1.2 Objetivos Específicos

Para alcançar este objetivo, foram definidas as seguintes etapas específicas:

- Realizar uma revisão bibliográfica sobre o PRVC e as meta-heurísticas de Algoritmo Genético e *Simulated Annealing*;
- Implementar o Algoritmo Genético como um método para gerar uma solução inicial robusta para o PRVC;
- Implementar o algoritmo *Simulated Annealing* como um método de busca local para o refinamento de uma solução dada;
- Integrar os algoritmos em um fluxo sequencial, no qual a saída do AG serve como entrada para o SA;
- Validar a abordagem proposta através da aplicação do modelo em instâncias de teste conhecidas na literatura, como os conjuntos CMT (Christofides et al., 1979) e X-n (Uchoa et al., 2017);
- Analisar os resultados obtidos e discutir a eficácia da hibridização proposta.

1.2 Estrutura do Trabalho

Este trabalho está organizado em cinco capítulos, além desta introdução.

O **Capítulo 2** apresenta a contextualização teórica, detalhando formalmente o Problema de Roteamento de Veículos Capacitado, assim como os fundamentos das meta-heurísticas Algoritmo Genético e *Simulated Annealing*.

O **Capítulo 3** descreve a metodologia, detalhando a abordagem híbrida proposta, a representação da solução e os passos do algoritmo desenvolvido.

O **Capítulo 4** apresenta os resultados experimentais obtidos pela aplicação do modelo, incluindo a descrição do ambiente de teste e a análise comparativa dos resultados.

O **Capítulo 5** traz a conclusão do trabalho, sumarizando as contribuições, discutindo as limitações e propondo direções para trabalhos futuros.

CAPÍTULO 2

Contextualização

Este capítulo apresenta a fundamentação teórica necessária para a compreensão do trabalho. Inicialmente, a Seção 2.1 define formalmente o Problema de Roteamento de Veículos Capacitado (PRVC), detalhando sua representação em grafos, formulação matemática e complexidade computacional. Em seguida, a Seção 2.2 descreve o Algoritmo Genético (AG), seus operadores e funcionamento. Por fim, a Seção 2.3 aborda a metaheurística *Simulated Annealing* (SA), destacando sua analogia com a termodinâmica e mecanismo de busca local.

2.1 Problema de Roteamento de Veículos Capacitado

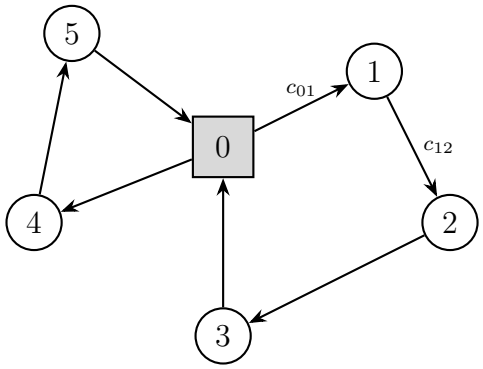
O Problema de Roteamento de Veículos (*Vehicle Routing Problem* – VRP) é uma questão central na logística, afetando diretamente tanto a eficiência das entregas quanto os custos operacionais das empresas. Proposto originalmente por Dantzig and Ramser (1959), o VRP é uma generalização do clássico Problema do Caixeiro Viajante (TSP).

Dentre as diversas variantes do VRP, o Problema de Roteamento de Veículos Capacitado (PRVC) é uma das mais estudadas e aplicadas no mundo real. O PRVC busca 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 a distância total percorrida ou o tempo de viagem. Todas as rotas devem iniciar e terminar em um único depósito central.

2.1.1 Representação em Grafos

Formalmente, o PRVC 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 (arcos). O nó $j = 0$ representa o depósito central, enquanto os nós $j = 1, 2, ..., n$ correspondem aos clientes (Toth and Vigo, 2014a).

Cada cliente j possui uma demanda conhecida $d_j > 0$ que deve ser atendida. Cada aresta $(i, j) \in E$ que conecta os nós i e j define um segmento de rota, cujo peso $c_{ij} > 0$ representa o custo associado (distância ou tempo). Uma frota de K veículos idênticos está disponível no depósito, cada um com uma capacidade máxima de carga Q.



0: Depósito 1-5: Clientes

Figura 2.1: Representação esquemática do grafo $G(V, E)$ com depósito e clientes
Fonte: Elaborado pelos autores.

2.1.2 Formulação Matemática

A formulação matemática do PRVC, baseada na notação de Noon et al. (1994), utiliza as seguintes definições:

- $\mathcal{N} = \{1, 2, ..., n\}$: Conjunto de clientes.
- $\mathcal{V} = \{1, 2, ..., K\}$: Conjunto de veículos disponíveis.
- $E = \{(i, j) \mid i \neq j, i, j \in \{0, 1, ..., n\}\}$: Conjunto de arcos possíveis.
- c_{ij}: Custo de deslocamento de i para j.
- x_{ij}^k: Variável de decisão binária, que assume valor 1 se o veículo k percorre a aresta (i, j), e 0 caso contrário.
- d_i: Demanda do cliente i.
- Q: Capacidade máxima do veículo.

O objetivo do PRVC é minimizar o custo total de deslocamento de todos os veículos, conforme a Equação 2.1:

$$\min \sum_{k=1}^{K} \sum_{(i,j) \in E} c_{ij} x_{ij}^k \tag{2.1}$$

A função objetivo está sujeita às seguintes restrições:

$$\sum_{k=1}^{K}\sum_{i=0}^{n} x_{ij}^{k} = 1, \quad \forall j \in \mathcal{N} \tag{2.2}$$

$$\sum_{i=0}^{n} x_{ip}^{k} - \sum_{j=0}^{n} x_{pj}^{k} = 0, \quad \forall p \in \mathcal{N}, \forall k \in \mathcal{V} \tag{2.3}$$

$$\sum_{j=1}^{n} x_{0j}^{k} \leq 1, \quad \forall k \in \mathcal{V} \tag{2.4}$$

$$\sum_{i=0}^{n}\sum_{j=1}^{n} d_j x_{ij}^{k} \leq Q, \quad \forall k \in \mathcal{V} \tag{2.5}$$

$$\sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}} x_{ij}^{k} \leq |\mathcal{S}| - 1, \quad \forall \mathcal{S} \subseteq \mathcal{N}, |\mathcal{S}| \geq 2, \forall k \in \mathcal{V} \tag{2.6}$$

$$x_{ij}^{k} \in \{0, 1\}, \quad \forall (i, j) \in E, \forall k \in \mathcal{V} \tag{2.7}$$

A restrição (2.2) garante que cada cliente seja visitado exatamente uma vez. A restrição (2.3) assegura a conservação de fluxo (todo veículo que entra em um cliente deve sair). A restrição (2.4) garante que cada veículo saia do depósito no máximo uma vez. A restrição (2.5) impõe o limite de capacidade de cada veículo. A restrição (2.6) elimina sub-rotas desconectadas do depósito. Finalmente, (2.7) define a natureza binária das variáveis.

2.1.3 Complexidade Computacional

O PRVC é classificado como NP-difícil (*NP-hard*). Isso significa que não se conhece nenhum algoritmo capaz de encontrar a solução ótima garantida em tempo polinomial, e o tempo necessário cresce exponencialmente com o aumento do número de clientes. Para instâncias realistas, métodos exatos como *branch-and-bound* tornam-se inviáveis, justificando o uso de meta-heurísticas como Algoritmos Genéticos e *Simulated Annealing* para buscar boas soluções em tempo hábil.

2.2 Algoritmo Genético

Os Algoritmos Genéticos (AGs) constituem uma técnica de busca e otimização inspirada nos princípios da teoria da evolução Darwin (1859); Holland (1975); Goldberg (1989). Os AGs baseiam-se em um processo de busca probabilístico, que simula os mecanismos de seleção natural e genética biológica.

O ponto de partida de um Algoritmo Genético é a formação de uma população inicial, composta por um conjunto de possíveis soluções para o problema em estudo. Cada solução individual dentro dessa população é denominada cromossomo, sendo comumente representada por uma lista ou cadeia de dados.

A evolução da população ocorre ao longo de diversas gerações, por meio de um ciclo iterativo que compreende as seguintes etapas:

1. **Inicialização da população**: Geração de soluções iniciais, que podem ser totalmente aleatórias, baseadas em heurísticas, ou ainda parcialmente aleatórias a partir da combinação de diferentes métodos.

2. **Avaliação da população**: Cada indivíduo é analisado por meio de uma função de aptidão *(fitness function)*, que quantifica a qualidade de sua solução em relação ao problema proposto.

3. **Seleção**: Os indivíduos com melhor desempenho são selecionados para a próxima etapa, possuindo maior probabilidade de participar do processo de reprodução.

4. **Cruzamento *(crossover)***: Os indivíduos selecionados são combinados em pares para trocar características entre si, promovendo diversidade e possibilitando a criação de novas soluções potencialmente superiores.

5. **Mutação**: Pequenas alterações aleatórias são introduzidas em alguns indivíduos, a fim de explorar novas regiões do espaço de busca e evitar a convergência prematura do algoritmo.

6. **Formação da nova geração**: Após as etapas anteriores, uma nova população é formada, composta por indivíduos mais aptos e diversificados, que passam a representar a próxima geração do processo evolutivo.

Esse ciclo se repete até que um critério de parada seja atendido, como o alcance de um número máximo de gerações, a convergência das soluções ou a obtenção de um nível de aptidão satisfatório, resultando em uma solução otimizada para o problema analisado.

O Algoritmo 1 apresenta o pseudocódigo geral de um algoritmo genético.

Algoritmo 1: Pseudocódigo de um Algoritmo Genético

```
1 Gerar a população inicial P;
2 enquanto o critério de parada não for satisfeito Avaliar a aptidão dos
    indivíduos em P;
3 Selecionar indivíduos de P para reprodução;
4 Realizar cruzamento para gerar novos descendentes;
5 Aplicar mutação sobre os descendentes;
6 Atualizar a população P combinando pais e descendentes;
7 retorna P;
```

O **cruzamento** (*crossover*) representa o principal operador dos AGs, sendo responsável pela recombinação das informações genéticas de dois indivíduos da população. Durante esse processo, características dos pais são misturadas e transmitidas aos descendentes, o que possibilita o surgimento de novas combinações de soluções. Dessa forma, o cruzamento atua como o principal mecanismo de herança e propagação de boas características, favorecendo a evolução da população ao longo das gerações. Normalmente, ele é aplicado com uma probabilidade associada à *taxa de crossover*, a qual costuma ser superior à taxa de mutação, uma vez que o cruzamento exerce maior influência sobre o desempenho global do algoritmo.

A **mutação**, por outro lado, exerce papel complementar ao cruzamento. Ela tem a função de introduzir variações aleatórias nos indivíduos, alterando um ou mais de

seus componentes genéticos. Essa aleatoriedade permite que novas regiões do espaço de busca sejam exploradas, evitando que o processo evolutivo fique preso a regiões limitadas ou a ótimos locais. Além de preservar a diversidade genética da população, a mutação garante que qualquer ponto do espaço de soluções possa, em teoria, ser alcançado. Geralmente, a mutação é aplicada com uma probabilidade menor, definida pela *taxa de mutação* (P_m), visto que seu objetivo é promover ajustes sutis na busca, e não transformações drásticas.

De modo geral, o *crossover* e a *mutação* funcionam como mecanismos de exploração do espaço de busca, favorecendo a descoberta de novas regiões e potenciais soluções. Em contraste, o operador de seleção atua como um processo de *exploitation*, direcionando a evolução para áreas já identificadas como promissoras. O equilíbrio entre aproveitamento e exploração é o que permite ao AG alcançar soluções de alta qualidade sem perder diversidade ao longo do tempo.

Em consequência desse equilíbrio, o algoritmo pode convergir para um ótimo local — seja ele mínimo ou máximo — dentro do espaço de busca. Ainda assim, os AGs se destacam por diversas vantagens, entre as quais se incluem:

- Capacidade de realizar buscas paralelas em diferentes regiões do espaço de soluções;
- Aplicabilidade a problemas com variáveis contínuas ou discretas;
- Facilidade de implementação e adaptação em diferentes contextos computacionais;
- Flexibilidade para lidar com múltiplos objetivos e restrições simultâneas, tornando-os adequados para otimização multiobjetivo.

A eficiência de um Algoritmo Genético depende fortemente de como seus parâmetros são definidos. Compreender a influência desses parâmetros é essencial para ajustar o comportamento do algoritmo de acordo com as características do problema e os recursos computacionais disponíveis. Entre os principais **parâmetros**, destacam-se:

- **Tamanho da População:** define a quantidade de indivíduos avaliados em cada geração. Populações pequenas podem comprometer a diversidade genética e reduzir a cobertura do espaço de busca, aumentando o risco de convergência prematura. Já populações maiores proporcionam melhor amostragem do espaço de soluções, mas demandam maior tempo de execução e capacidade computacional.
- **Taxa de Cruzamento:** controla a frequência com que o operador de recombinação é aplicado. Taxas elevadas tendem a acelerar a evolução, introduzindo rapidamente novas estruturas genéticas; entretanto, valores excessivos podem eliminar indivíduos de alta qualidade. Por outro lado, taxas muito baixas tornam o processo evolutivo mais lento e menos eficaz.
- **Taxa de Mutação:** regula o grau de variação aleatória introduzido nas soluções. Uma taxa baixa é útil para evitar a estagnação e preservar boas características, enquanto uma taxa muito alta pode tornar o processo essencialmente randômico, comprometendo a convergência.

2.3 *Simulated Annealing*

O algoritmo *Simulated Annealing* foi apresentado por **Kirkpatrick, Gelatt e Vecchi** em 1983 com o trabalho *"Optimization by Simulated Annealing"*. A fórmula principal do algoritmo é a replicação da ideia empregada no recozimento de metais, o controle ao resfriar e aquecer os metais de maneira a otimizar a estrutura ao menor custo energético possível. O SA se encaixa na categoria de metaheurísticas baseadas em trajetória, isso significa que a busca se dá através da busca pela melhor solução candidata ao longo de todo o tempo de execução.

No processo físico de recozimento (*annealing*), um sólido é primeiro aquecido até derreter. Em seguida, ele é resfriado muito lentamente, passando um tempo considerável perto do ponto de congelamento. Esse processo cuidadoso permite que os átomos se organizem em um estado de equilíbrio, formando um cristal com energia mínima (o estado fundamental). Se o resfriamento for muito rápido (um processo chamado *quenching* ou têmpera), o material fica preso em estados metaestáveis e imperfeitos, como um vidro, que não representam o ótimo global.

De maneira direta ao recozimento, o SA faz uso da otimização de energia (E), assim o problema gira em torno dessa variável. Deste modo, uma configuração ou estado do sistema representa uma possível solução para o problema proposto. Dentro do mesmo escopo, um novo parâmetro é introduzido ao algoritmo: a temperatura (T), que é um parâmetro de controle que faz a gestão do processo de busca por uma melhor solução à candidata atual.

A grande inovação do SA é a absorção do algoritmo de Metropolis Metropolis et al. (1953); Kirkpatrick et al. (1983). Esse algoritmo generaliza a melhoria iterativa comum. A melhoria iterativa comum apenas aceita movimentos que fazem o custo total da solução diminuir, o que faz com que a solução gerada pelo algoritmo possa ficar presa em ótimos locais. Assim, o algoritmo de Metropolis se apresenta como uma melhor opção, já que o mesmo possui regras de aceitação baseadas em probabilidade Kirkpatrick et al. (1983); Aarts and Korst (1989):

1. A partir da solução atual, um movimento aleatório (como um pequeno deslocamento de um átomo) gera uma nova solução vizinha;
2. Calcula-se a variação no custo (energia), ΔE;
3. Se $\Delta E \leq 0$ (a nova solução é melhor ou igual), ela é sempre aceita;
4. Se $\Delta E > 0$ (a nova solução é pior), ela pode ser aceita com uma probabilidade $P(\Delta E) = \exp(-\Delta E/T)$ (onde k_B é a constante de Boltzmann, geralmente omitida ou ajustada no parâmetro T).

O *Simulated Annealing* tem sua execução iniciada com uma temperatura T muito alta e, assim, com a execução dos processos, ela ocorre em derreter, com seus passos iniciais quase sempre sendo aceitos. À medida que a temperatura T vai diminuindo progressivamente, também é reduzida a mudança de estágios:

Para altas temperaturas, o algoritmo tem altas probabilidades de aceitação para realizar buscas no espaço de possibilidades, o que permite que rapidamente ele escape de ótimos locais. Para baixas temperaturas, a probabilidade de que um movimento

pior seja aceito diminui vertiginosamente, assim o algoritmo começa a se comportar de maneira mais lenta, com melhorias iterativas e fazendo apenas pequenas melhorias com mais segurança.

Em síntese, o SA pode ser descrito como um algoritmo que simula o processo físico de resfriamento lento para encontrar o estado de energia mínima (solução ótima) de um problema complexo. O algoritmo a seguir descreve os principais passos para a implementação do *Simulated Annealing*.

Algoritmo 2: *Simulated Annealing (SA)*

```
Inicialize uma solução candidata aleatória S_atual;
Defina uma temperatura inicial alta T = T_inicial;
Defina um cronograma de resfriamento;
repita
    repita
        Gere uma solução vizinha S_nova a partir de S_atual;
        Calcule ΔE = Custo(S_nova) − Custo(S_atual);
        se ΔE ≤ 0 então
            S_atual ← S_nova;
        senão
            Gere um número aleatório r ∈ [0, 1];
            se r < exp(−ΔE/T) então
                S_atual ← S_nova;
        Memorize a melhor solução encontrada até agora;
    até Atingir o número de iterações para a temperatura T (equilíbrio);
    Reduza a temperatura T (conforme o cronograma de resfriamento);
até Condição de parada (sistema "congelado", T muito baixa);
```

A principal força do SA é a sua capacidade de sair de ótimos locais por meio da aceitação controlada de piores soluções propostas. A sequência de temperaturas e o número de iterações em cada configuração formam um poderoso cronograma de ações, o que é essencial para que o SA seja assertivo, porém precisa ser cuidadosamente configurado para cada problema a ser resolvido.

Diante destas caracteristicas, o *Simulated Annealing* funcionará neste presente trabalho como um mecanismo de intensificação local. De maneira particular, após a execução do Algoritmo Genético, com uma tarefa de ser o realizador de uma busca mais aprofundada em sua vizinhança com aceitações de pioras controladas e aproveitando de sua valia de escapar de otimos locais e promover o refinamento final.

CAPÍTULO 3

Metodologia

A metodologia adotada neste trabalho descreve, de forma estruturada, os passos necessários para o desenvolvimento e aplicação da abordagem híbrida proposta. Neste capítulo são apresentados os fundamentos que orientam o método, bem como as etapas que compõem sua implementação. Inicialmente, apresenta-se uma visão geral da estratégia de hibridização adotada, destacando a integração entre Algoritmo Genético e Simulated Annealing. Em seguida, detalha-se a representação da solução utilizada, que define a forma como os indivíduos são codificados e interpretados pelo algoritmo.

Na sequência, são discutidos os procedimentos de avaliação das soluções e a implementação do Algoritmo Genético, contemplando seus operadores e mecanismos internos. Também é descrita a integração do método Simulated Annealing como etapa de refinamento local, evidenciando seu papel na intensificação da busca. Por fim, são apresentados o processo de ajuste dos parâmetros, bem como as ferramentas e o ambiente utilizados para a execução dos experimentos, garantindo transparência e reprodutibilidade ao estudo.

3.1 Visão Geral da Abordagem Híbrida

O princípio da metodologia aplicada neste trabalho é a estratégia de hibridização entre as meta-heurísticas Algoritmo Genético (AG) e *Simulated Annealing* (SA). A motivação para a escolha destas duas meta-heurísticas foi pela forma complementar com que as duas atuam: AG com exploração global do espaço de busca e SA como mecanismo de intensificação local, refinando soluções previamente encontradas. **O fluxo geral da abordagem híbrida proposta é apresentado na Figura 3.1.**

O fluxo geral da abordagem proposta pode ser descrito com a seguinte representação diagramática:

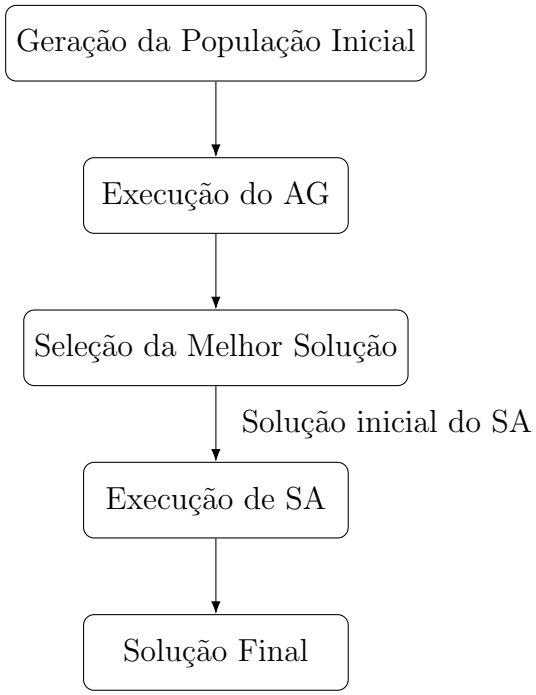


Figura 3.1: Fluxo de abordagem híbrida AG + SA.

Conforme indicado no diagrama, esta abordagem está organizada, basicamente, em duas etapas sequenciais; na primeira, o AG gera e evolui um grupo de soluções viáveis, para tal são realizadas operações de seleção, cruzamento e mutação para que seja realizada a exploração e escolha das rotas. Em um segundo momento, a melhor solução encontrada é submetida ao SA que realiza as melhorias locais, através de movimentações na vizinhança e avaliações por meio de seu critério probabilístico. Essa lógica híbrida de combinar um algoritmo genético com uma etapa de intensificação local é amplamente utilizada no PRVC, pois equilibra diversificação e refinamento, como discutido na *Hybrid Genetic Search* proposta por Vidal et al. (2021).

3.2 Representação da Solução (Codificação)

Em relação ao PRVC, é necessário definir como uma solução é representada dentro do algoritmo. Em problemas reais, uma solução pode ser entendida como uma coleção de rotas que saem do depósito, visitam os clientes uma única vez e retornam ao depósito. Ao longo do processo, deve-se garantir que a capacidade do veículo não seja excedida, mantendo sempre a viabilidade das rotas encontradas (Toth and Vigo, 2014b; Laporte, 2009).

Neste trabalho, as rotas não são representadas diretamente nos cromossomos. Cada indivíduo é codificado como uma permutação única dos clientes, formando uma sequência em que todos os nós aparecem exatamente uma vez. Essa forma de representação é conhecida como *giant tour* e, por preservar a estrutura de permutação, facilita a aplicação dos operadores de crossover e mutação sem comprometer a viabilidade básica do indivíduo. Essa ideia é amplamente utilizada em trabalhos sobre PRVC desde Prins (Prins, 2004), além de compor métodos de hibridização como o HSG de Vidal et al. (Vidal et al., 2013, 2021).

Aqui é necessário definir dois níveis de solução, já que a estrutura possui dois níveis distintos. No AG, a solução é representada como um vetor/lista de inteiros contendo os clientes ordenados. Esse modelo de permutação combina naturalmente com operadores de ordem, como o *crossover OX*, pois preserva a estrutura sequencial relativa ao gerar os filhos. Após o passo de decodificação, a sequência gerada é transformada em uma lista de rotas que segue o mesmo padrão — um vetor de clientes — mas agora representando efetivamente um caminho. É essa lista de rotas que será utilizada pelo SA em sua etapa de execução e refinamento.

A capacidade se torna relevante no momento em que o cromossomo é convertido em rotas. A sequência dos clientes é percorrida ordenadamente, somando-se as demandas até que o limite do veículo seja alcançado. Quando isso ocorre, a rota é encerrada e uma nova é iniciada, garantindo assim a viabilidade da solução. Dessa forma, o AG opera livremente nas ordens globais de visita, e sua representação permutável faz com que raramente sejam criadas soluções inviáveis.

Já no SA, a viabilidade é verificada a cada vizinho gerado. Em cada movimento local — seja troca, realocação, troca entre rotas ou uma reinserção curta — a viabilidade é novamente avaliada e, caso o movimento resulte em uma solução inviável, ele é descartado.

3.3 Avaliação da Solução

A avaliação de uma solução no problema de roteamento tem como objetivo determinar a qualidade das rotas propostas, permitindo que o algoritmo compare alternativas e selecione aquelas mais eficientes. No contexto do PRVC, uma solução é composta por um conjunto de rotas que partem do depósito, atendem clientes específicos e retornam à origem. A métrica utilizada neste trabalho para avaliar a qualidade das soluções é a *distância total percorrida*, abordagem amplamente consolidada na literatura clássica de VRP (Laporte, 1992; Toth and Vigo, 2002).

O cálculo do custo segue a formulação tradicional do VRP: para cada rota, somam-se as distâncias entre todos os pares consecutivos de clientes, incluindo o deslocamento do depósito ao primeiro cliente e do último cliente de volta ao depósito, conforme discutido por Toth and Vigo (2002). Assim, se uma rota atende os clientes $v_1, v_2, \ldots, v_n$, o custo associado é dado por:

$$C(r) = d(0, v_1) + \sum_{i=1}^{n-1} d(v_i, v_{i+1}) + d(v_n, 0),$$

e o custo total da solução é obtido pela soma dos custos de todas as rotas pertencentes ao indivíduo avaliado.

Embora restrições operacionais sejam fundamentais no Capacitated VRP (PRVC) (Golden et al., 2008), neste trabalho tais restrições não são incorporadas diretamente à função de custo. Em vez disso, a verificação de viabilidade quanto à capacidade é tratada de forma estritamente restritiva: caso a demanda total de uma rota exceda a capacidade permitida, a solução é considerada inválida e descartada pelo algoritmo durante o processo evolutivo.

Dessa forma, a função de custo adotada neste trabalho permanece estritamente

baseada na distância descrita, enquanto a verificação de viabilidade é tratada separadamente e de maneira rígida pelo algoritmo. Essa abordagem garante simplicidade na avaliação, transparência no modelo de custo e compatibilidade com o comportamento observado no código implementado.

3.4 Implementação do Algortimo Genético

A implementação do Algoritmo Genético utilizada neste trabalho foi estruturada de modo a garantir um equilíbrio adequado entre exploração e intensificação durante o processo evolutivo. Nesta subseção são apresentadas as etapas que compõem o funcionamento do AG, contemplando desde a criação das soluções iniciais até o processo de renovação da população a cada geração.

Inicialmente, descreve-se o método empregado para a geração da população inicial, responsável por estabelecer a diversidade e a qualidade básica das primeiras soluções. Em seguida, são detalhados os mecanismos de seleção utilizados para determinar os indivíduos que participarão da reprodução.

Na sequência, apresentam-se os operadores genéticos de crossover e mutação, responsáveis pela criação de novos indivíduos e pela introdução controlada de variabilidade no processo evolutivo. Por fim, discute-se a estratégia de substituição adotada, que define como a nova população é formada e como as melhores soluções são preservadas ao longo das gerações.

3.4.1 População Inicial

A definição da população inicial desempenha um papel central na eficiência e na qualidade das soluções geradas por algoritmos evolutivos aplicados ao Problema de Roteamento de Veículos (VRP). No algoritmo implementado, cada indivíduo da população corresponde a um cromossomo composto por uma permutação dos clientes, que posteriormente é convertido em um conjunto de rotas factíveis por meio do processo de divisão gulosa *(split)*. Essa abordagem segue princípios amplamente aceitos na literatura, segundo os quais a representação indireta — em que o cromossomo não contém explicitamente as rotas, mas apenas a sequência de visitação — permite maior flexibilidade e evita a quebra de restrições estruturais essenciais do VRP.

A geração das soluções iniciais inicia-se com a construção de um cromossomo base contendo todos os clientes em ordem crescente. A partir dele, parte da população (aproximadamente 20%) é produzida por uma heurística gulosa do vizinho mais próximo, estratégia tradicionalmente reconhecida por fornecer soluções iniciais razoáveis e de rápida obtenção em problemas de roteamento (Laporte, 1992). Os demais indivíduos são gerados por embaralhamento aleatório completo da permutação de clientes, garantindo diversidade estrutural desde o início da evolução. Cada cromossomo produzido passa pelo processo de conversão em rotas por meio do operador *split*, que verifica, a cada novo cliente, se sua inserção ultrapassa a capacidade do veículo. Caso isso ocorra, a rota atual é encerrada e uma nova rota é iniciada. Esse procedimento assegura que toda solução construída seja válida em relação à restrição de capacidade ou, caso uma estratégia de penalização esteja ativada, que o custo reflita adequadamente possíveis violações, como sugerido nas formulações clássicas de funções de penalidade para VRP

(Bräysy and Gendreau, 2005).

Um aspecto adicional considerado no projeto da população inicial é a preocupação com a duplicação de indivíduos, fator que pode prejudicar a diversidade genética e provocar convergência prematura do algoritmo. Embora o espaço de permutações dos clientes seja extremamente grande — reduzindo naturalmente a probabilidade de duplicações — a implementação adota mecanismos implícitos de diversificação, como a aleatorização independente de cada permutação e o uso de heurísticas gulosas apenas para uma fração da população, o que reduz a chance de que múltiplos indivíduos compartilhem padrões idênticos.

3.4.2 Seleção

A etapa de seleção do algoritmo genético tem como objetivo escolher, dentro da população atual, os indivíduos que terão maior probabilidade de produzir descendentes nas próximas gerações, direcionando a busca evolutiva para regiões promissoras do espaço de soluções. No algoritmo implementado, foi utilizado o método Seleção por Torneio, uma técnica amplamente adotada na literatura devido à sua simplicidade, eficiência computacional e capacidade de controlar o nível de pressão seletiva de forma direta. Segundo Mitchell (1998), a seleção por torneio consiste em escolher aleatoriamente um conjunto de indivíduos da população e selecionar o melhor entre eles para reprodução, repetindo esse processo conforme necessário. Esse mecanismo permite que a competição local entre candidatos reduza a probabilidade de indivíduos muito fracos serem escolhidos, enquanto mantém diversidade adequada, uma vez que mesmo indivíduos não ótimos ainda possuem chance de participar do torneio (Goldberg, 1989).

No código implementado, cada torneio seleciona um conjunto de indivíduos de forma aleatória, definido pelo parâmetro *tournamentSize*, e o indivíduo de menor custo (melhor aptidão) é escolhido como vencedor. Essa abordagem segue exatamente o procedimento descrito por Eiben and Smith (2003), onde o tamanho do torneio controla diretamente o nível de elitismo natural do método: torneios maiores aumentam a pressão seletiva, acelerando a convergência, enquanto torneios menores aumentam a diversidade. Assim, o parâmetro utilizado possibilita regular o equilíbrio entre exploração e exploração, um aspecto essencial em problemas combinatórios como o VRP, frequentemente caracterizados por múltiplos ótimos locais (Reinelt, 1994).

A escolha da seleção por torneio se justifica por diversas razões. Primeiramente, o método é independente da escala de valores da função de custo, evitando problemas comuns associados ao método da roleta, como saturação ou sensibilidade a diferenças pequenas de aptidão (Goldberg and Deb, 1991). Em segundo lugar, é um método extremamente eficiente, exigindo apenas comparações diretas entre custos sem cálculos adicionais, o que é desejável para problemas de grande dimensão. Além disso, sua implementação é simples e se integra de maneira natural com operadores como elitismo e *crossover* ordenado, utilizados no algoritmo desenvolvido. Como argumentam Blickle and Thiele (1995), a seleção por torneio tende a apresentar desempenho superior em contextos onde as diferenças relativas entre indivíduos são significativas, como é o caso do VRP com cromossomos representados por permutações.

3.4.3 *Crossover*

O operador de *crossover* é um dos mecanismos centrais dos algoritmos genéticos, sendo responsável por combinar informações provenientes de dois indivíduos da população para gerar novos descendentes com potencial de melhorar a qualidade das soluções. Inspirado nos processos de recombinação genética observados na biologia, o *crossover* atua como principal força exploratória do algoritmo, permitindo que características favoráveis presentes em diferentes indivíduos sejam reunidas em uma única solução (Holland, 1975). Em termos gerais, a lógica do *crossover* consiste em selecionar dois pais e combinar partes de seus cromossomos, preservando estruturas relevantes enquanto introduz diversidade controlada, conforme discutido por Goldberg (1989).

Existem diversos tipos de *crossover*, cada um adequado a uma representação específica do cromossomo. Para problemas de permutação — como o VRP, em que as soluções são formadas por sequências de clientes — destacam-se operadores especialmente projetados para evitar duplicações ou perdas de elementos. Entre os mais utilizados (Pereira et al., 2023) encontra-se o *Partially Mapped Crossover (PMX)*, que cria uma correspondência entre regiões dos pais para garantir consistência posicional (Goldberg and Lingle, 1985). Outro operador importante é o *Cycle Crossover (CX)*, que identifica ciclos de posição entre os pais e os transfere integralmente para os descendentes, mantendo a estrutura de dependência entre posições (Oliver et al., 1987). Existe também o *Edge Recombination Crossover (ERX)*, projetado para preservar adjacências frequentes entre elementos, sendo particularmente útil em problemas em que a proximidade entre clientes é determinante para a qualidade da solução (Whitley et al., 1989).

No algoritmo desenvolvido, o operador escolhido foi o *Ordered Crossover (OX)*, amplamente referenciado na literatura como uma das opções mais eficazes para problemas de roteamento e permutação (Davis, 1985). O *OX* funciona selecionando um segmento contínuo de um dos pais e preservando essa estrutura no descendente, enquanto o restante das posições é preenchido com os valores do segundo pai na mesma ordem em que aparecem. Esse método mantém a ordem relativa entre os clientes, preservando sequências potencialmente eficientes, ao mesmo tempo em que evita duplicações e garante que todos os elementos da permutação estejam presentes exatamente uma vez. Essa característica torna o *OX* particularmente adequado para o VRP, uma vez que a ordem de visita aos clientes está diretamente relacionada ao custo total das rotas, como apontado por Gendreau and Potvin (2010).

3.4.4 Mutação

A mutação é um componente essencial dos algoritmos genéticos, atuando como mecanismo primário de diversificação da população e prevenindo a convergência prematura para ótimos locais. Inspirada nos processos biológicos de alteração espontânea do material genético, a mutação introduz pequenas perturbações nos cromossomos, permitindo que o algoritmo explore regiões do espaço de busca que não seriam acessíveis apenas pela recombinação entre indivíduos. Segundo Goldberg (1989), a mutação desempenha um papel fundamental no equilíbrio entre exploração (busca global de novas soluções) e intensificação (refinamento das soluções já boas), garantindo dinamismo ao processo evolutivo.

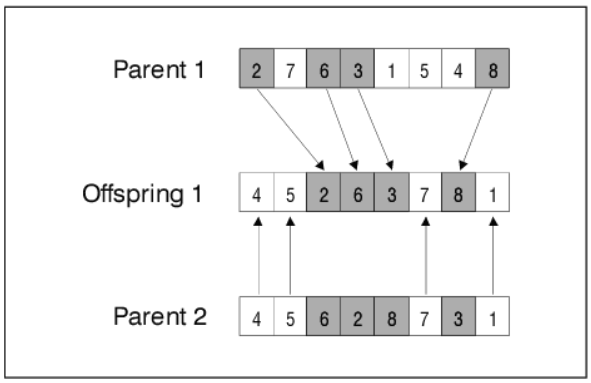


Figura 3.2: Exemplo de uma operação de *Ordered Crossover*
Fonte: Radiah, Hasnah & Omar, 2013

Em problemas de permutação, como o VRP, a escolha do operador de mutação é particularmente importante, pois a representação exige que cada cliente apareça exatamente uma vez no cromossomo. Operadores tradicionais como a mutação por inversão ou *scramble* são frequentemente utilizados na literatura (Reeves, 1993), mas o algoritmo desenvolvido adotou a *swap mutation*, uma das abordagens mais simples e eficazes para esse tipo de representação. Esse operador consiste em selecionar aleatoriamente duas posições do cromossomo e trocar seus elementos. A simplicidade do *swap mutation* é acompanhada de robustez, sendo amplamente recomendado em problemas de roteamento e ordenação por manter a integridade estrutural da permutação e introduzir perturbações leves que evitam destruição excessiva de padrões produtivos (Michalewicz, 1996).

Um aspecto fundamental após a aplicação de operadores de mutação em permutações é o tratamento de viabilidade da solução. Em muitos problemas, a mutação pode gerar cromossomos inválidos, como duplicação de clientes ou ausência de algum elemento. No caso do operador *swap* utilizado, esse problema não ocorre, pois a troca de posições não altera o conjunto de elementos presentes, garantindo automaticamente que todos os clientes continuarão aparecendo exatamente uma vez. Essa característica torna o *swap mutation* especialmente adequado, eliminando a necessidade de mecanismos adicionais de correção estrutural, frequentemente exigidos quando se utilizam operadores mais disruptivos, como *scramble* ou *insertion*(Potvin, 1996). Entretanto, mesmo com a preservação da permutação, a viabilidade logística da solução — particularmente quanto à divisão de rotas e respeito à capacidade dos veículos — é tratada posteriormente na conversão do cromossomo para uma solução completa, quando o algoritmo reconstrói as rotas e verifica se a carga acumulada não ultrapassa o limite permitido. Esse processo, conforme recomendado em aplicações clássicas do AG ao VRP, garante que mesmo soluções mutadas preservem a consistência com as restrições do problema.

3.4.5 Substituição

A etapa de substituição é responsável por definir quais indivíduos comporão a próxima geração do algoritmo genético após o processo de seleção, cruzamento e mutação. No método adotado, utilizou-se uma estratégia que combina elitismo com a reconstrução

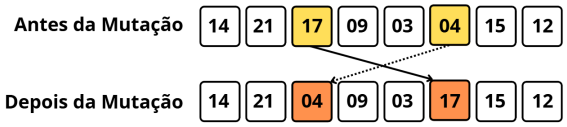


Figura 3.3: Exemplo de Operação de *Swap Mutation*
Fonte: Elaborada pelos autores

da população por meio dos indivíduos gerados ao longo da iteração. O tamanho total da população permaneceu constante durante toda a execução, característica comum em algoritmos genéticos clássicos (Goldberg, 1989; Eiben and Smith, 2003).

O elitismo garante que uma quantidade previamente definida dos melhores indivíduos da geração atual seja preservada sem alterações. Esses indivíduos, avaliados como os de maior aptidão segundo a função objetivo, são copiados diretamente para a próxima geração. Essa prática evita que soluções de alta qualidade sejam descartadas devido à aleatoriedade das operações genéticas, proporcionando maior estabilidade e promovendo uma convergência mais eficiente.

Após reservar os espaços destinados aos indivíduos elitistas, a substituição é completada pelos novos indivíduos produzidos pelo cruzamento e pela mutação. Assim, a nova geração é formada pela união desses dois grupos: os indivíduos preservados e os indivíduos recém-criados. Caso a quantidade de indivíduos produzidos ultrapasse o número necessário para completar a população, apenas os primeiros gerados são adicionados até alcançar o tamanho estabelecido; caso seja inferior, indivíduos adicionais são gerados ou selecionados conforme o mecanismo implementado.

Esse esquema de substituição equilibra exploração e preservação: mantém soluções promissoras já encontradas enquanto introduz variação genética suficiente para que o algoritmo continue explorando o espaço de busca. Dessa forma, evita-se tanto a perda prematura de boas soluções quanto a estagnação em regiões subótimas.

3.5 Implementação do *Simulated Annealing*

Após o passo evolutivo com o AG, passamos ao componente local baseado em *Simulated Annealing*. Aqui, recebemos a solução gerada pelo AG que, em termos gerais, já é uma boa solução inicial, porém ainda com espaço para melhoria em suas rotas. O SA se destaca, nesta busca local, pela melhoria das rotas com uma busca intensa, usando a vizinhança clássica do PRVC e uma aceitação baseada em probabilidade, permitindo aceitar pioras locais controladas, da mesma maneira que no modelo original proposto por (Kirkpatrick et al., 1983). Este tipo de solução com SA é frequentemente utilizado na aplicação ao roteamento, uma vez que ele encontra novas respostas possíveis sem que seja necessária a construção do zero (Liu and Kozan, 2018).

3.5.1 Ponto de Aplicação do SA

Nesta aplicação de um fluxo híbrido, o SA é realmente utilizado na aplicação apenas após toda a execução de AG ser finalizada com todas as suas gerações, ou seja, ele não é exercido dentro de um ciclo evolutivo, inicialmente o AG evolui completamente sua

população e ao final desta execução o seu melhor produto é selecionado. Essa solução encontrada já apresenta um formato de rotas viáveis e assim ela é usada de ponto de partida para o SA. Apenas um indivíduo de AG é melhorado por SA. Essa escolha agrupa a ideia sequencial com AG atuando na diversificação e SA no refinamento final (Sajid et al., 2021; Talbi, 2023).

3.5.2 Vizinhança

O cerne da aplicação de SA é a definição de vizinhança. Na execução do algoritmo, a solução corrente, é continuamente modificada por movimentos locais simples (operadores de vizinhança), escolhidos de forma aleatória, que geram soluções vizinhas a partir de pequenas alterações na rota atual. Assim, as iterações do código tentam gerar vizinhos com o uso de um dos impulsos abaixo:

- ***Swap* intra-rota**: Faz a seleção de dois clientes de uma mesma rota criada e faz a inversão de posições entre eles.
- ***Relocate* (realocação)**: Faz a seleção de um cliente de uma das rotas e insere este usuário em outra rota caso a sua demanda caiba na capacidade disponível.
- ***Exchange* (troca inter-rota)**: Realiza o *Swap* porém agora são de clientes presentes em rotas diferentes mantendo a viabilidade da solução.
- ***Or-opt* (reinserção curta)**: Remove um pequeno segmento de 1 ou 2 clientes e reinsere em outra posição da rota.

Estas operações são bem conhecidas para execuções de PRVC uma vez que elas conseguem de maneira satisfatória realizar pequenas mudanças locais, elas também são amplamente utilizadas em heurísticas e meta-heurísticas de roteamento. Em relação a validação da viabilidade, todo vizinho gerado passa por uma avaliação para que somente seja aceito se continuar sendo possível. Como as rotas já estão nítidas, é realizada a checagem de maneira direta somando a demanda em cada rota gerada, em caso de alguma estar acima do possível este movimento é descartado imediatamente e outra mudança é tentada. Esta verificação constante permite que exista um padrão em SA quando aplicado a PRVC e sempre gera soluções possíveis.

3.5.3 Parâmetros do SA

O SA foi configurado seguindo o modelo mais clássico, com o resfriamento geométrico, comumente utilizado em roteamento, e seguem os principais parâmetros:

- **Temperatura Inicial ($T_{\max}$)**: Faz o controle do quanto o algoritmo irá aceitar pioras nas execuções iniciais, permitindo maior exploração do espaço de possibilidades.
- **Temperatura Final ($T_{\min}$)**: Define o limite inferior no qual o processamento é finalizado (quando $T \leq T_{\min}$).

- **Taxa de Resfriamento (α)**: Reduz a temperatura gradualmente seguindo a atualização geométrica:
$$T \leftarrow \alpha \cdot T$$
- **Número de Tentativas por Temperatura**: A cada nível de temperatura, o algoritmo avalia vários vizinhos antes de iniciar o próximo resfriamento.

Esta abordagem é amplamente reportada visto que ela tem um início mais flexível e conforme é executado se torna um algoritmo com características mais conservadoras, que é uma marca registrada do SA. Para execuções de PRVC o uso dos parâmetros $T_{\min}$, $T_{\max}$ e α juntamente com o resfriamento geométrico é o padrão mais encontrado, pois apresenta um equilíbrio ótimo entre intensificação e custo computacional (Harmanani et al., 2007; Liu and Kozan, 2018). Para critérios de parada usamos:

1. A temperatura atual da execução chegar ao nível mínimo expresso por $T_{\min}$, este encontro de valores indica que o sistema se encontra em um estado que pode ser dito quase que como congelado e estático não valendo mais aceitar pioras;
2. Limitar o número de iterações de vizinhos em cada nível de temperatura, decisão ao qual garante que o controle do tempo a ser executado e que o SA fique dentro de um período controlado.

Estes critérios atualmente são os mais consistentes para uma aplicação de SA no contexto de PRVC e amplamente utilizado na literatura recente (Liu and Kozan, 2018).

3.6 Ajuste de Parâmetros

No presente trabalho, os parâmetros do algoritmo híbrido foram definidos com base em recomendações da literatura e em experimentações preliminares de caráter exploratório. Em vez de uma bateria extensa de testes robustos, foram realizados pequenos casos piloto e ajustes manuais, com o objetivo de verificar se os valores escolhidos produziam um comportamento coerente com o esperado para o método e com a escala do PRVC estudada. Assim, esses testes serviram como uma calibração inicial, sem a pretensão de conduzir uma análise estatística completa de sensibilidade. Dessa forma, buscou-se um conjunto de parâmetros que apresentasse desempenho consistente para a proposta, equilibrando custo computacional, capacidade exploratória e qualidade das soluções obtidas.

No Algoritmo Genético, adotou-se uma população composta por vinte indivíduos, quantidade suficiente para manter diversidade sem comprometer o tempo de execução, considerando que instâncias do PRVC podem exigir múltiplas avaliações de custo. Trabalhos clássicos indicam que populações pequenas, quando combinadas com mecanismos de controle de diversidade, podem ser eficientes em problemas combinatórios (Goldberg, 1989; Eiben and Smith, 2003).

O número de gerações foi fixado em cem, valor intencionalmente moderado para evitar convergência excessiva do Algoritmo Genético. Essa escolha impede que a população evolua até um ponto em que todos os indivíduos tornem-se demasiadamente semelhantes, o que poderia limitar a capacidade do *Simulated Annealing* de explorar regiões promissoras do espaço de busca. Em outras palavras, a limitação deliberada no

número de gerações mantém a solução obtida pelo AG em um nível intermediário de refinamento, preservando variabilidade suficiente para que o SA possa atuar de forma mais significativa no processo de melhoria local. Dessa forma, o papel do AG permanece como fornecedor de boas soluções iniciais, enquanto o SA se responsabiliza pela etapa de intensificação, reforçando a sinergia característica dos métodos híbridos.

A taxa de crossover utilizada foi de 0.85, seguindo a recomendação de que o cruzamento deve ser o principal operador de exploração em algoritmos genéticos, apresentando alta probabilidade de aplicação. Já a taxa de mutação adotada, igual a 0.02, alinha-se aos valores tradicionalmente utilizados na literatura, uma vez que mutações raras contribuem para evitar convergência prematura sem comprometer estruturas adaptativas importantes (Mitchell, 1996). O processo de seleção por torneio empregou três competidores, promovendo uma pressão seletiva moderada, enquanto a estratégia de elitismo preservou dois indivíduos por geração, garantindo que soluções de alta qualidade não fossem descartadas durante o processo evolutivo.

Para a etapa de *Simulated Annealing*, definiu-se uma temperatura inicial de 1000, valor suficientemente alto para permitir ampla aceitação de movimentos desfavoráveis nas primeiras iterações, característica fundamental para escapar de mínimos locais, conforme descrito por Kirkpatrick et al. (1983). A temperatura final foi definida como 0.01, indicando encerramento apenas quando a probabilidade de aceitar movimentos piores torna-se praticamente nula, garantindo precisão na fase final do refinamento. A taxa de resfriamento de 0,98 estabelece um decaimento térmico lento e cuidadoso, adequado para a complexidade do PRVC. O número de iterações por temperatura foi configurado proporcionalmente ao número de clientes da instância, multiplicado por vinte, garantindo escalabilidade e exploração adequada por nível térmico. Além disso, definiu-se um limite de trinta tentativas para geração de vizinhos válidos, evitando sobrecarga computacional em situações onde a vizinhança apresenta baixa viabilidade.

A combinação desses parâmetros resulta em um equilíbrio apropriado entre exploração global, conduzida pelo Algoritmo Genético, e intensificação local, realizada pelo *Simulated Annealing*. Ao controlar deliberadamente o grau de convergência do AG e deixar espaço para atuação efetiva do SA, o método híbrido alcança resultados robustos em termos de custo total das rotas e número de veículos utilizados, reforçando a complementaridade entre as duas abordagens.

3.7 Ferramentas e Ambiente de Execução

O ambiente de execução utilizado para o desenvolvimento e avaliação do algoritmo consistiu em uma combinação de *hardware* e *software* capaz de oferecer uma boa capacidade de processamento e compatibilidade com as ferramentas empregadas. A implementação foi realizada em linguagem C++, escolhida pela sua eficiência, controle de memória e desempenho superior em aplicações que exigem múltiplas iterações e manipulação intensiva de estruturas de dados, características típicas em algoritmos evolutivos e de otimização combinatória.

Os experimentos foram executados em um computador equipado com um processador AMD Ryzen 5 4600G, com *clock* base de 3.7 GHz, cuja arquitetura de múltiplos núcleos favorece a execução de rotinas paralelizáveis, além de 16 GB de memória RAM, permitindo lidar, de forma regular, com as instâncias do problema e com estruturas

populacionais utilizadas no algoritmo genético e no *Simulated Annealing*. A máquina também possui *Radeon Graphics* integrada, suficiente para o ambiente de desenvolvimento e para visualização de resultados. O sistema operacional utilizado foi o Windows 10, escolhido por sua estabilidade, ampla compatibilidade com compiladores C++ e boa integração com ferramentas de desenvolvimento.

Para o desenvolvimento do projeto, utilizou-se o Visual Studio Code (VS Code), um ambiente leve, modular e altamente configurável, que permite integração direta com extensões de C++, depuradores e ferramentas de análise. Essa combinação proporcionou um fluxo de programação eficiente, desde a escrita do código até testes e validação. O conjunto de *hardware* e *software* adotado garantiu um ambiente adequado para o desenvolvimento e execução do algoritmo, assegurando confiabilidade nas etapas de implementação, depuração e experimentação.

CAPÍTULO 4

Análise de Resultados

Este capítulo apresenta os resultados experimentais obtidos com a aplicação da abordagem híbrida proposta (AG+SA) para a resolução do Problema de Roteamento de Veículos Capacitado. Inicialmente, são descritas as instâncias de teste utilizadas e a configuração do ambiente experimental. Em seguida, são discutidas as métricas de avaliação adotadas. Por fim, realiza-se uma análise comparativa de desempenho entre o algoritmo híbrido e as meta-heurísticas originais (Algoritmo Genético e *Simulated Annealing*) executadas isoladamente.

4.1 Instâncias de Teste

Para a avaliação do desempenho dos algoritmos propostos, foi utilizado um conjunto de 17 instâncias do Problema de Roteamento de Veículos Capacitado (PRVC) amplamente reconhecido na literatura. Estas instâncias são divididas em dois grupos principais:

- **Conjunto CMT:** Instâncias clássicas propostas por Christofides et al. (1979). Deste conjunto, foram selecionadas 6 instâncias de médio porte: CMT1 a CMT5 e a CMT12. Estas instâncias variam em número de clientes e complexidade, servindo como base para validação inicial.
- **Conjunto X-n:** Foram utilizadas 11 instâncias de grande escala que representam *benchmarks* modernos para o PRVC (ex: X-n101-k25, X-n106-k14, ..., X-n1001-k43). Estas instâncias pertencem à coleção CVRPLIB (Uchoa et al., 2017) e desafiam os algoritmos com um número maior de clientes e restrições mais apertadas.

Os arquivos de dados das instâncias (extensão `.vrp`) foram obtidos a partir do repositório mantido por Vidal et al. (2021), que disponibiliza versões padronizadas das instâncias da CVRPLIB para testes de algoritmos de roteamento[1].

A Tabela 4.1 resume as instâncias utilizadas neste trabalho.

[1]Disponível em: `https://github.com/vidalt/HGS-CVRP/tree/main/Instances/CVRP`

Tabela 4.1: Instâncias de Teste Utilizadas

Conjunto	Instâncias Selecionadas	Fonte
CMT	CMT1–CMT5, CMT12	Christofides et al. (1979)
X-n	X-n101-k25 – X-n1001-k43 (11 instâncias)	Uchoa et al. (2017)

Fonte: Elaborado pelos autores.

4.2 Configuração Experimental

Esta seção detalha os parâmetros e a metodologia empregada para realizar a avaliação de desempenho dos algoritmos AG (Algoritmo Genético), SA (*Simulated Annealing*) e Híbrido (AG+SA).

Quantidade de Execuções: Devido à natureza estocástica das meta-heurísticas, cada um dos três algoritmos (AG, SA e Híbrido) foi executado 30 vezes de forma independente para cada uma das 17 instâncias de teste. Esse procedimento visa mitigar a variabilidade dos resultados e garantir a significância estatística das médias obtidas.

Critérios de Parada: O critério principal de parada foi o tempo máximo de processamento (CPU *time*), ajustado proporcionalmente à dimensão de cada instância (número de clientes). Adicionalmente, a execução era interrompida caso o algoritmo alcançasse o valor da *Best Known Solution* (BKS) — a melhor solução conhecida na literatura para aquela instância.

Métodos de Comparação:

- **Resultados Médios:** O desempenho primário é avaliado pela média dos custos finais das 30 execuções. Estes valores são comparados com a BKS e entre os algoritmos.
- **Melhoria Percentual (*Gap*):** Foi calculada a melhoria relativa do Algoritmo Híbrido em relação aos algoritmos base (AG e SA), utilizando a Equação 4.1:

$$\text{Melhoria}(\%) = \left(\frac{\text{Custo}_{\text{Base}} - \text{Custo}_{\text{Híbrido}}}{\text{Custo}_{\text{Base}}} \right) \times 100 \tag{4.1}$$

- **Análise Visual:** A estabilidade e o desempenho comparativo foram visualizados através de gráficos de colunas, permitindo identificar o comportamento dos algoritmos em diferentes escalas de instâncias.

4.3 Métricas Avaliadas

Para que este modelo de Hibridização seja avaliado quanto a seu desempenho, foram adotadas algumas métricas comumente utilizadas em estudos de PRVC envolvendo meta-heurísticas. Isso se mostra importante uma vez que estes algoritmos são estocásticos e assim não basta apenas uma visão focada no resultado de maneira isolada, mas também se mostra relevante notar e pontuar a qualidade média, estabilidade nas diversas execuções e o custo computacional. Diversos trabalhos publicados recentemente e revisões de VRP reafirmam exatamente esta conjuntura de análises como base para comparação da qualidade da solução proposta (Vidal, 2016; Ali et al., 2020).

4.3.1 Custo Total das Rotas

Em primeiro plano temos o **Custo Total** (total da distância percorrida entre clientes), o cálculo deste valor é feito através da soma dos custos de todas as rotas que estão presentes na solução final. Como o PRVC tem como norte minimizar diretamente este custo, ela representa frontalmente o objetivo de qualidade para esta solução proposta. Em cada instância o custo foi registrado ao final de sua execução e também agregado ao **custo médio**, valor usado para comparação entre algoritmos. Em disponibilidade deste valor de custo médio, podemos também o interpretar em relação a um valor de referência, ótimo conhecido ou BKS, isto nos permite realizar uma avaliação da proximidade que a abordagem realizada quanto o que já está posto na literatura. A CVRPLIB[2] organiza e disponibiliza essas instâncias e os valores de melhor solução conhecida (BKS) para servir como base de comparação.

4.3.2 Tempo de Execução

Outra importante métrica que foi medida é o **tempo de execução** para cada uma das instâncias. No contexto de softwares de roteamento, esta métrica de mostra muito importante em um momento onde não somente a qualidade tem relevância mas sim a comparação qualidade x custo computacional, este medição pode fazer com que uma solução proposta seja inexequível ou que uma solução com qualidade aquém ganhe relevância. No caso de modelos híbridos, o tempo representa o somatório das etapas de AG + SA, permitindo uma observabilidade mais detalhada.

4.3.3 Variabilidade / Estabilidade

Para que seja diminuída ao máximo aleatoriedades causadas pelo algoritmo e suas diversas escolhas causadas por população inicial, escolha dos vizinhos, aceitação probabilística, etc., avaliou-se também a presença de dispersão dos resultados, assim cada instância foi executada 30 vezes de maneira independente, esta forma mais direta de realmente observar os desvios possíveis. Este modelo de análise é recomendado na literatura por medir a robustez, assim um bom algoritmo não é somente aquele com uma média menor mas também aquele que tem uma pequena variância de resultado entre as suas execuções.

4.3.4 Melhoria Percentual

No fim, para realmente qualificar o real efeito do modelo híbrido, o cálculo feito foi a **melhoria percentual do modelo híbrido** em relação aos algoritmos base usados, assim temos a métrica calculada como:

$$\text{Melhoria}(\%) = \frac{C_{\text{base}} - C_{\text{híbrido}}}{C_{\text{base}}} \times 100$$

onde temos que C_{base} é o custo médio de AG ou se SA, este modelo de medidas é comumente utilizado em *benchmarks* de PRVC uma vez que deixa claro a efetividade do ganho e clarifica o efeito da proposta realizada.

[2]Disponível em: `https://vrp.atd-lab.inf.puc-rio.br/index.php/en/`

4.4 Resultados Obtidos

Esta seção detalha os resultados das execuções dos algoritmos Genético (AG), *Simulated Annealing* (SA) e Híbrido (AG+SA) nas instâncias de teste CMT e X-n. A análise é dividida entre o desempenho médio de custo e a robustez da distribuição dos resultados.

4.4.1 Comparação de Desempenho Médio

Inicialmente, o desempenho dos algoritmos foi avaliado com base no custo médio final obtido após 30 execuções independentes para cada instância. A Tabela 4.2 apresenta os valores médios de custo obtidos por cada abordagem, comparando-os com a melhor solução de Referência (BKS – *Best Known Solution*) disponível na literatura.

Tabela 4.2: Resultados Médios dos Algoritmos por Instância (30 Execuções)

Instância	**Referência (BKS)**	**AG (Média)**	**SA (Média)**	**Híbrido (Média)**
		Conjunto CMT		
CMT1	524.61	620.28	582.10	**570.44**
CMT2	835.26	933.65	883.77	**880.12**
CMT3	826.14	979.38	865.70	**864.12**
CMT4	1028.42	1247.98	**1121.97**	1125.23
CMT5	1291.29	1599.09	1448.82	**1433.08**
CMT12	819.56	940.58	875.76	**859.74**
		Conjunto X-n		
X-n101-k25	27591.00	29986.31	28788.50	**28700.91**
X-n106-k14	26362.00	27813.49	**27124.81**	27159.98
X-n110-k13	14971.00	16788.17	**15998.39**	16004.42
X-n115-k10	12747.00	14633.01	14131.88	**14068.52**
X-n120-k6	13332.00	15441.24	**14344.52**	14401.25
X-n125-k30	55539.00	60552.76	57984.36	**57774.63**
X-n129-k18	28940.00	31657.95	**31073.88**	31115.55
X-n134-k13	10916.00	12325.84	**11728.74**	11847.64
X-n139-k10	13590.00	16164.76	14940.02	**14869.71**
X-n228-k23	25742.00	31184.54	28134.10	**27874.74**
X-n1001-k43	72355.00	**85188.12**	93802.46	86332.99

Fonte: Elaborado pelos autores.

Ao analisar a Tabela 4.2, observa-se que o Algoritmo Híbrido (AG+SA) obteve o menor custo médio na maioria das instâncias do conjunto CMT e em diversas instâncias do conjunto X-n.

A abordagem híbrida demonstrou superioridade consistente em relação ao Algoritmo Genético (AG) puro em quase todas as instâncias, com exceção notável da instância de grande porte X-n1001-k43, onde o AG isolado apresentou melhor desempenho. Em relação ao *Simulated Annealing* (SA), houve uma maior competição: o Híbrido superou o SA na maioria das instâncias CMT e nas instâncias X-n mais complexas (como X-n125 e X-n228), mas o SA obteve médias ligeiramente melhores em casos específicos como CMT4 e X-n106-k14.

4.4.2 Eficácia do Algoritmo Híbrido (Melhoria Percentual)

Para quantificar o ganho de eficiência obtido pela hibridização, a Tabela 4.3 apresenta a melhoria percentual (*gap*) do Algoritmo Híbrido em relação aos métodos base isolados (AG e SA). O cálculo baseia-se na fórmula apresentada na Seção 4.2, onde valores positivos indicam superioridade do Híbrido e valores negativos indicam que o algoritmo base obteve um desempenho médio superior.

Tabela 4.3: Melhoria Percentual do Algoritmo Híbrido em Relação aos Algoritmos Base

Instância	**Melhora Híbrido/AG (%)**	**Melhora Híbrido/SA (%)**
	Conjunto CMT	
CMT1	8.04%	2.00%
CMT2	5.73%	0.41%
CMT3	11.77%	0.18%
CMT4	9.84%	-0.29%
CMT5	10.38%	1.09%
CMT12	8.59%	1.83%
	Conjunto X-n	
X-n101-k25	4.29%	0.30%
X-n106-k14	2.35%	-0.13%
X-n110-k13	4.67%	-0.04%
X-n115-k10	3.86%	0.45%
X-n120-k6	6.74%	-0.40%
X-n125-k30	4.59%	0.36%
X-n129-k18	1.71%	-0.13%
X-n134-k13	3.88%	-1.01%
X-n139-k10	8.01%	0.47%
X-n228-k23	10.61%	0.92%
X-n1001-k43	-1.34%	7.96%

Fonte: Elaborado pelos autores.

A análise da Tabela 4.3 revela que o ganho do método Híbrido é significativamente mais expressivo quando comparado ao Algoritmo Genético puro. O pico de melhoria foi observado na instância **CMT3**, com um ganho de **11.77%**, seguido pela instância de grande porte **X-n228-k23** com **10.61%**.

Entretanto, há exceções importantes. Na instância massiva **X-n1001-k43**, o Híbrido teve um desempenho inferior ao AG (-1.34%), o que sugere que, para espaços de busca extremamente amplos, o custo computacional da busca local do SA pode reduzir a eficiência da evolução global se o tempo de execução for restrito.

Em comparação ao *Simulated Annealing*, os ganhos são mais modestos e, em alguns casos (indicados pelos valores negativos), o SA puro apresentou médias ligeiramente melhores (ex: CMT4 e X-n134-k13). Contudo, na instância mais complexa (X-n1001-k43), a hibridização garantiu uma melhoria robusta de 7.96% sobre o SA, indicando que a população inicial fornecida pelo AG foi crucial para evitar que o SA ficasse preso em ótimos locais de baixa qualidade.

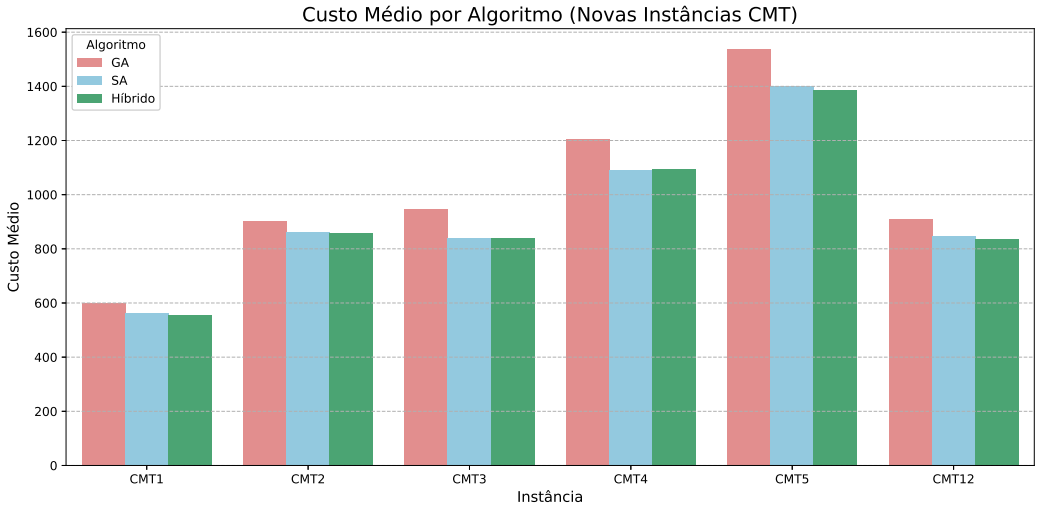
4.4.3 Comparação Visual das Médias de Custo

Para complementar a análise da Tabela 4.2, o desempenho dos algoritmos é visualizado através de gráficos de coluna, que permitem a comparação direta dos custos médios obtidos, separados por conjuntos de instâncias.

Instâncias CMT

A Figura 4.1 ilustra o custo médio final para as instâncias de menor porte (CMT).

Figura 4.1: Custo Médio dos Algoritmos (AG, SA, Híbrido) nas Instâncias CMT



Fonte: Elaborado pelo autor.

A análise visual das colunas na Figura 4.1 confirma que, na maioria das instâncias CMT, o algoritmo Híbrido (AG+SA) alcançou o menor custo médio (barras verdes), superando consistentemente o AG. Nota-se uma disputa acirrada com o SA (barras azuis), especialmente na instância CMT4, onde o SA obteve uma ligeira vantagem.

Instâncias X-n

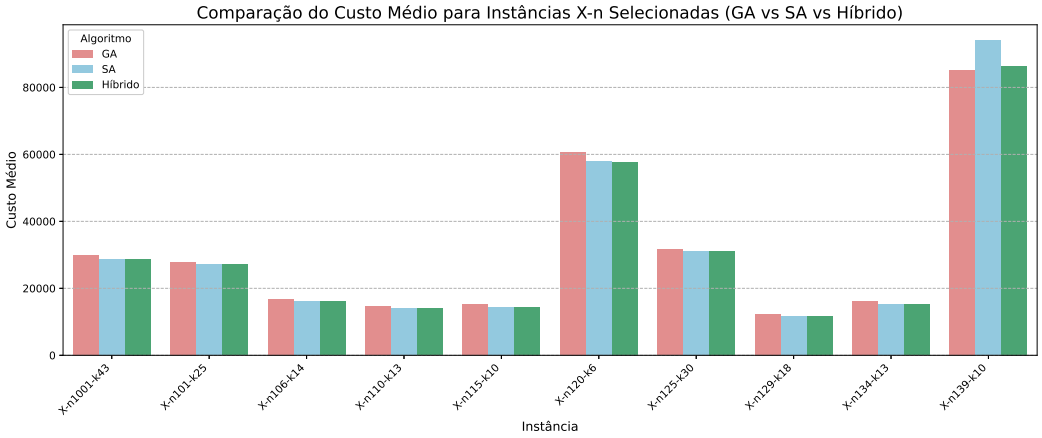
Para as instâncias de maior porte (X-n), a Figura 4.2 compara o custo médio entre as abordagens.

Observa-se na Figura 4.2 que, nas instâncias maiores, a abordagem híbrida mantém sua competitividade, frequentemente apresentando as barras mais baixas. A comparação visual reitera que a hibridização é uma estratégia eficaz para reduzir o custo médio final, equilibrando a exploração do AG com o refinamento do SA, exceto em casos extremos de escala (como X-n1001) onde o comportamento varia.

4.5 Discussão dos Resultados

A avaliação de desempenho do algoritmo híbrido proposto foi conduzida por meio de uma comparação direta com as versões canônicas do Algoritmo Genético (GA) e do *Simulated Annealing (SA)*. Os resultados apresentados na Tabela 4.3, expressos em

Figura 4.2: Custo Médio dos Algoritmos (AG, SA, Híbrido) nas Instâncias X-n



Fonte: Elaborado pelo autor.

termos da melhoria percentual alcançada pelo método híbrido, evidenciam comportamentos distintos conforme a heurística de referência utilizada, reforçando as vantagens decorrentes da combinação entre abordagens populacionais e de trajetória.

Ao comparar o híbrido com o Algoritmo Genético puro, verifica-se uma superioridade clara e consistente da abordagem híbrida. Nas instâncias da classe CMT, o ganho de desempenho manteve-se sempre positivo, variando entre 5% e 11%. Esses valores sugerem que a incorporação do operador de busca local inspirado no *Simulated Annealing* foi determinante para superar a limitação clássica dos algoritmos genéticos quanto ao refinamento das soluções finais. Apesar de apresentar boa capacidade de exploração global, o GA puro tende a falhar na convergência para ótimos locais dentro das regiões promissoras do espaço de busca. A hibridização corrigiu essa deficiência ao realizar o ajuste fino dos indivíduos após as etapas de cruzamento e mutação. Observa-se, contudo, uma exceção na instância de grande escala X-n1001-k43, em que o desempenho do híbrido foi ligeiramente inferior ao do GA (-1,34%). Esse resultado indica que, em espaços de busca extremamente amplos, o custo computacional adicional da busca local pode reduzir a quantidade de gerações evolutivas, impactando negativamente a diversidade populacional quando há restrição de tempo de processamento.

A comparação entre o algoritmo híbrido e o *Simulated Annealing* puro revela um cenário mais equilibrado, caracterizado por compromissos entre qualidade de solução e robustez. Em várias instâncias da série X, o híbrido apresentou pequenas perdas relativas, geralmente inferiores a 1%, indicando que o SA bem parametrizado é bastante competitivo em problemas de porte médio. Entretanto, a principal vantagem da abordagem híbrida torna-se evidente quando se analisa a estabilidade dos resultados. Em situações nas quais o SA isolado apresentou desempenho insatisfatório, como na instância X-n1001-k43, em que o híbrido obteve cerca de 8% de melhoria, a presença dos operadores genéticos impediu estagnações em ótimos locais de baixa qualidade. Esse comportamento confirma a maior suscetibilidade do SA, enquanto metaheurística de solução única, a aprisionamento prematuro, especialmente em instâncias grandes e complexas. Já o componente populacional do híbrido forneceu diversidade suficiente para evitar tais falhas.

De modo geral, a discussão dos resultados demonstra que a hibridização atua de

forma verdadeiramente sinérgica, combinando a capacidade de exploração global do GA com o poder de intensificação do SA. Embora o método híbrido não supere o SA em todas as instâncias de menor escala, ele se mostra mais robusto e confiável no conjunto geral de problemas analisados, reduzindo significativamente os piores cenários observados nas heurísticas isoladas.

CAPÍTULO 5

Conclusão

A partir da análise comparativa realizada, é possível concluir que o algoritmo híbrido proposto apresenta um comportamento diferenciado frente às metaheurísticas canônicas que o compõem. Os resultados evidenciaram que a cooperação entre os mecanismos evolutivos do Algoritmo Genético e a busca local inspirada no *Simulated Annealing* produz efeitos positivos principalmente na etapa de refinamento das soluções.

Quando comparado ao GA puro, o híbrido demonstrou ganhos consistentes na maior parte das instâncias analisadas, confirmando que a inclusão de uma estratégia de intensificação contribui para superar a limitação natural dos algoritmos genéticos em convergir para ótimos locais de forma precisa. A exceção observada na instância de maior escala reforça apenas que, em problemas muito extensos, o custo adicional da busca local pode impactar o número total de gerações, gerando pequenas perdas de desempenho.

Em relação ao SA, os resultados mostraram um cenário mais equilibrado: em diversas instâncias, o desempenho do híbrido foi muito próximo ou ligeiramente inferior ao SA puro, o que evidencia a eficiência do método de solução única quando bem parametrizado. Entretanto, também se verificou que o componente evolutivo do híbrido desempenhou papel importante em casos específicos onde o SA apresentou forte degradação de desempenho, sobretudo em instâncias de grande porte. Nesse sentido, o híbrido não se mostrou superior ao SA de forma geral, mas sim mais estável em situações particulares que favorecem a diversidade populacional.

De maneira geral, o estudo indica que a hibridização atua como um equilíbrio entre exploração ampla e intensificação local, oferecendo um comportamento consistente ao longo de diferentes classes de instâncias. O método híbrido não substitui nem supera completamente o GA ou o SA em todos os cenários, mas se apresenta como uma alternativa viável quando se busca uma abordagem que combine refinamento, diversidade e menor risco de falhas críticas em problemas de maior complexidade.

5.1 Trabalhos Futuros

Uma direção natural para trabalhos futuros consiste em avaliar a metodologia proposta em um conjunto ainda mais amplo de instâncias, incluindo aquelas de outras coleções clássicas, como *Golden, Augerat, Uchoa-Large* VRP, entre outras presentes na CVRPLIB. Isso permitiria identificar padrões adicionais de comportamento dos algoritmos, bem como validar a consistência da hibridização em cenários com diferentes características geométricas, níveis de dispersão espacial e demandas variadas. Além disso, seria interessante explorar instâncias com janelas de tempo, múltiplos depósitos ou frota heterogênea, ampliando o escopo para variantes do VRP amplamente estudadas na literatura e presentes em problemas reais.

Outra possibilidade relevante é a investigação de técnicas de hibridização adaptativa, nas quais o peso relativo das fases exploratória (AG) e intensificadora (SA) seja ajustado dinamicamente durante a execução. Tal adaptação poderia ajudar a superar a limitação observada em instâncias muito grandes, onde o SA se torna menos eficiente. Estratégias como autoajuste de parâmetros, mecanismos reativos baseados no histórico da busca ou até a integração com outras meta-heurísticas — como GRASP, VNS ou ILS — representam caminhos promissores.

Também seria valioso estudar a adoção de estruturas de vizinhança mais avançadas, como *Large Neighborhood Search (LNS)*, ou técnicas de intensificação guiada por aprendizagem, como *reinforcement learning* para escolha adaptativa de operadores. Esses métodos podem complementar o papel do SA, tornando sua atuação mais eficaz mesmo em espaços de busca de alta dimensionalidade.

Por fim, uma linha adicional de investigação consiste em desenvolver uma versão paralelizada ou distribuída do método híbrido, explorando arquiteturas multicore ou GPU. Essa abordagem poderia reduzir significativamente o tempo de execução em instâncias de grande escala e potencialmente modificar o comportamento observado, permitindo que a fase de intensificação se torne novamente competitiva mesmo em problemas com milhares de nós.

Referências Bibliográficas

Aarts, E. and Korst, J. (1989). *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*. Wiley.

Ali, I., Turky, A., and Ayob, M. (2020). Recent Advances in Hybrid Metaheuristics for Vehicle Routing Problem. *Algorithms*, 13(8):1–20. Comparativo de métricas de desempenho entre híbridos.

Blickle, T. and Thiele, L. (1995). A Comparison of Selection Schemes Used in Evolutionary Algorithms. TIK-Report 11, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.

Bräysy, O. and Gendreau, M. (2005). Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms. *Transportation Science*, 39(1):104–118.

Christofides, N., Mingozzi, A., and Toth, P. (1979). The Vehicle Routing Problem. In Christofides, N. e. a., editor, *Combinatorial Optimization*, pages 315–338. Wiley.

Dantzig, G. B. and Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, 6(1):80–91.

Darwin, C. (1859). *On the Origin of Species*. John Murray, London.

Davis, L. (1985). Applying Adaptive Algorithms to Epistatic Domains. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Eiben, A. E. and Smith, J. E. (2003). *Introduction to evolutionary computing*. Springer.

Gendreau, M. and Potvin, J.-Y. (2010). Tabu search. In *Handbook of metaheuristics*, pages 165–186. Springer.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.

Goldberg, D. E. and Deb, K. (1991). A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. In Rawlins, G., editor, *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann.

Goldberg, D. E. and Lingle, R. (1985). Alleles, Loci, and the Traveling Salesman Problem. In *Proceedings of the First International Conference on Genetic Algorithms*.

Golden, B., Raghavan, S., and Wasil, E., editors (2008). *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer.

Harmanani, H., Mansour, N., and Kanso, A. (2007). A Simulated Annealing Algorithm for the Capacitated Vehicle Routing Problem. In *Proceedings of the 19th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 155–162.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, 220(4598):671–680.

Laporte, G. (1992). The Vehicle Routing Problem: An Overview of Exact and Approximate Algorithms. *European Journal of Operational Research*, 59(3):345–358.

Laporte, G. (2009). Fifty Years of Vehicle Routing. *Transportation Science*, 43(4):408–416.

Liu, M. and Kozan, E. (2018). A Simulated Annealing Based Approach for the Vehicle Routing Problem. *European Journal of Operational Research*, 267(2):841–861.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, 3 edition.

Mitchell, M. (1996). An Introduction to Genetic Algorithms. Technical report, University of Michigan.

Mitchell, M. (1998). *An Introduction to Genetic Algorithms*. MIT Press.

Noon, C. E., Mittenthal, J., and Pillai, R. (1994). Advances in Modeling and Solving Vehicle Routing Problems. *Journal of Transportation Engineering*, 120(3):387–398.

Oliver, I. M., Smith, D. J., and Holland, J. R. (1987). A Study of Permutation Crossover Operators on the Traveling Salesman Problem. In *Proceedings of the Second International Conference on Genetic Algorithms*.

Pereira, J. et al. (2023). Pmx — partially mapped crossover. CRAN R Project Documentation. Disponível em: `https://search.r-project.org/CRAN/refmans/adana/html/pmx.html`. Acesso em: 7 dez. 2025.

Potvin, J.-Y. (1996). Genetic Algorithms for the Traveling Salesman Problem. *Annals of Operations Research*, 63:337–370.

Prins, C. (2004). A Simple and Effective Evolutionary Algorithm for the Vehicle Routing Problem. *Computers & Operations Research*, 31(12):1985–2002.

Reeves, C. R., editor (1993). *Modern Heuristic Techniques for Combinatorial Problems*. Blackwell Scientific.

Reinelt, G. (1994). *The Traveling Salesman: Computational Solutions for TSP Applications*. Springer.

Sajid, A., Khan, S. A., and Shah, S. A. A. (2021). A Hybrid Simulated Annealing Algorithm for Vehicle Routing Problems. *Applied Sciences*, 11(9):1–20.

Talbi, E.-G. (2023). *Metaheuristics for Optimization: Algorithms and Applications*. Springer.

Toth, P. and Vigo, D., editors (2002). *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics (SIAM).

Toth, P. and Vigo, D. (2014a). *Vehicle routing: problems, methods, and applications*. Society for Industrial and Applied Mathematics.

Toth, P. and Vigo, D. (2014b). *Vehicle routing: problems, methods, and applications*. Society for Industrial and Applied Mathematics.

Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., and Subramanian, A. (2017). New Benchmark Instances for the Capacitated Vehicle Routing Problem. *European Journal of Operational Research*, 257(3):845–858.

Vidal, T. (2016). Split algorithm in O(n) for the capacitated vehicle routing problem. *Computers & Operations Research*, 69:40–47. Métrica de complexidade computacional e avaliação de fitness.

Vidal, T., Crainic, T. G., Gendreau, M., and Prins, C. (2013). A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research*, 40(1):475–489.

Vidal, T., Roberti, R., and Ropke, S. (2021). Implicit Solution Encoding for Vehicle Routing Problems. *European Journal of Operational Research*, 290(3):777–794.

Whitley, D., Starkweather, T., and Fuquay, D. (1989). Scheduling Problems and Traveling Salesman: The Genetic Edge Recombination Operator. In *Proceedings of the Third International Conference on Genetic Algorithms*, pages 133–140.