Implementação do Algoritmo Genético para Resolver o Problema do Caixeiro Viajante (TSP)

João Pedro Ospedal dos Santos

[1]Universidade Tuiuti do Paraná
Curitiba – PR

{joao.santos16}@utp.edu.br

Resumo. *Este trabalho explora o uso de Algoritmos Genéticos (AGs) na resolução do Problema do Caixeiro Viajante (TSP), que busca o menor caminho para visitar um conjunto de cidades uma única vez e retornar à origem. Foram testadas três configurações de AGs, variando operadores de crossover, taxas de mutação, formas de inicialização e critérios de parada. Os testes foram realizados sobre duas instâncias com cinco e seis cidades, cujas coordenadas foram fornecidas pelo usuário. Todas as configurações encontraram a mesma solução ótima em termos de distância, mas apresentaram tempos de execução distintos. A configuração com inicialização heurística e critério de parada por convergência foi a mais eficiente, reduzindo significativamente o tempo de processamento. Os resultados mostram que ajustes nos operadores genéticos podem melhorar a eficiência do algoritmo sem comprometer a qualidade da solução.*

1. Introdução

O Problema do Caixeiro Viajante (Travelling Salesman Problem – TSP) é um problema clássico de otimização combinatória no qual se busca determinar o caminho de menor custo que permita visitar um conjunto de cidades exatamente uma vez, retornando à cidade de origem. Este problema pertence à classe NP-difícil, o que significa que não existe, até o momento, algoritmo de tempo polinomial conhecido que resolva todas as instâncias de forma exata [GareyJohnson1979].

O TSP possui ampla aplicação prática em áreas como logística, planejamento de rotas, redes de distribuição e bioinformática [Applegate2006]. Entretanto, sua complexidade torna inviável a aplicação de algoritmos exatos para instâncias de grande porte. Dessa forma, estratégias heurísticas e metaheurísticas têm sido amplamente utilizadas para encontrar soluções aproximadas de boa qualidade em tempo viável.

Entre essas abordagens, destacam-se os Algoritmos Genéticos (AGs), introduzidos por Holland [Holland1975], que se baseiam nos princípios da evolução natural. AGs operam sobre uma população de soluções candidatas, aplicando operadores como seleção, crossover e mutação para gerar novas soluções a cada geração. Eles são especialmente úteis em problemas com grandes espaços de busca e características complexas, como o TSP [Mitchell1998].

Neste trabalho, investiga-se o uso de um Algoritmo Genético para resolver instâncias do TSP com diferentes configurações de operadores. Avaliam-se os impactos da escolha do tipo de crossover, taxa de mutação, forma de inicialização da população

e critério de parada. A justificativa para o uso de AGs reside na sua robustez, simplicidade de implementação e bom desempenho em problemas combinatórios [Reeves1995].

2. Metodologia

Este trabalho tem como objetivo avaliar o desempenho de um Algoritmo Genético (AG) aplicado ao Problema do Caixeiro Viajante (TSP), sob diferentes configurações de operadores genéticos. A seguir são descritos os elementos principais da metodologia utilizada, incluindo a implementação do algoritmo, os parâmetros configuráveis, as instâncias utilizadas e o critério de avaliação.

Cada indivíduo na população é representado por uma permutação dos índices das cidades a serem visitadas. Essa codificação é adequada ao TSP, pois garante que cada cidade será visitada exatamente uma vez, respeitando as restrições do problema.

Como configurações, foram usadas variações de crossover, taxa de mutação, inicialzação de população e critério de parada:

- **Crossover:**
  - Um ponto: corta a permutação em um ponto fixo e realiza troca parcial entre os pais.
  - Dois pontos: similar ao anterior, mas utiliza dois pontos de corte para maior recombinação.
  - Uniforme: seleciona cada posição do filho com base em uma probabilidade de herança de cada pai.
- **Taxa de Mutação:**
  - Baixa (1%)
  - Média (5%)
  - Alta (10%)
- **Inicialização da população:**
  - Aleatória: indivíduos gerados por permutações aleatórias das cidades.
  - Heurística (vizinho mais próximo): inclui indivíduos gerados por um algoritmo do tipo "vizinho mais próximo", que tende a produzir soluções iniciais de melhor qualidade.
- **Critério de parada:**
  - Número fixo de gerações (300): o algoritmo é executado por 300 gerações.
  - Convergência (50 gerações sem melhora): a execução é interrompida quando não há melhora no melhor indivíduo por 50 gerações consecutivas.

Foram utilizadas duas instâncias do TSP com coordenadas bidimensionais, descritas a seguir. Cada cidade está posicionada em um plano cartesiano.

Tabela 1. Coordenadas da Instância 1 (5 cidades)

Cidade	Coordenada X	Coordenada Y
1	27.50411189625751	56.99942629466494
2	61.62193366776722	53.98227359108739
3	28.587006473481114	86.23250613115717
4	22.035022128799476	35.765199514518685
5	0.0004412443822543466	59.15669815093235

Tabela 2. Coordenadas da Instância 2 (6 cidades)

Cidade	Coordenada X	Coordenada Y
1	5.460289212750835	43.34624376876254
2	92.80066976122855	56.70918986981631
3	28.873266683965447	19.20086445130401
4	14.496087068534314	92.64654629337083
5	89.09326279801039	55.23874333104428
6	96.43113373162713	32.10508157079847

Três configurações distintas foram definidas para avaliar o impacto dos operadores genéticos. Cada configuração combina um tipo de crossover, uma taxa de mutação, uma estratégia de inicialização da população e um critério de parada.

Tabela 3. Configurações de Algoritmo Genético testadas

ID	Crossover	Mutação	Inicialização	Parada
A	Um ponto	Baixa (1%)	Aleatória	Gerações (300)
B	Dois pontos	Média (5%)	Heurística	Convergência (50)
C	Uniforme	Alta (10%)	Aleatória	Gerações (300)

Cada configuração foi executada uma vez para cada instância, utilizando diferentes sementes aleatórias. Para cada execução, foram coletados os seguintes dados:

- Caminho encontrado pelo algoritmo representado pela sequência de números;
- Distância total do melhor caminho encontrado;
- Tempo de execução (em segundos);

Todos os testes foram realizados utilizando a linguagem Python 3.10.

3. Resultados

Os testes realizados permitiram avaliar o impacto das diferentes combinações de operadores genéticos na qualidade das soluções e no tempo de execução do algoritmo. A Tabela 4 apresenta os resultados para a Instância 1 (5 cidades), e a Tabela 5 mostra os dados para a Instância 2 (6 cidades).

Tabela 4. Resultados para Instância 1 (5 cidades)

Configuração	Melhor Caminho Encontrado	Distância Total	Tempo de Execução (s)
A	1, 2, 3, 5, 4	173.85	0.2570
B	1, 2, 3, 5, 4	173.85	0.0330
C	4, 5, 3, 2, 1	173.85	0.2620

Tabela 5. Resultados para Instância 2 (6 cidades)

Configuração	Melhor Caminho Encontrado	Distância Total	Tempo de Execução (s)
A	6, 3, 1, 4, 5, 2	264.84	0.2800
B	6, 3, 1, 4, 5, 2	264.84	0.0380
C	6, 3, 1, 4, 5, 2	264.84	0.2900

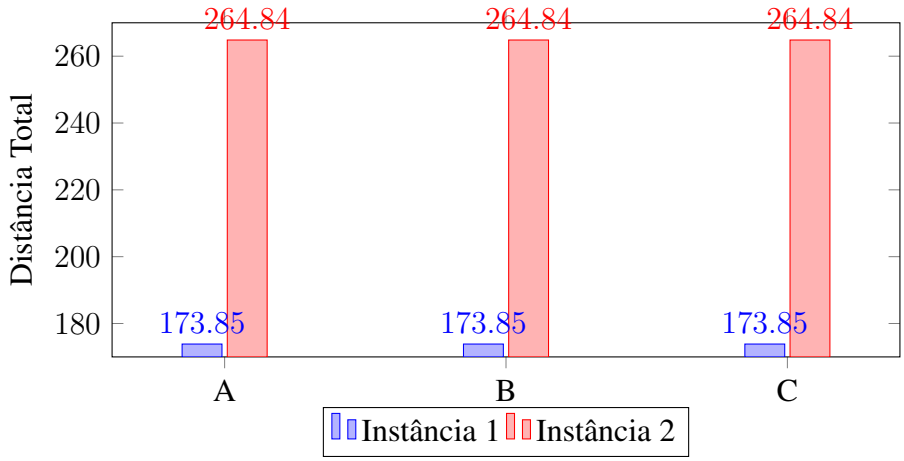

Figura 1. Comparação da distância total encontrada por configuração

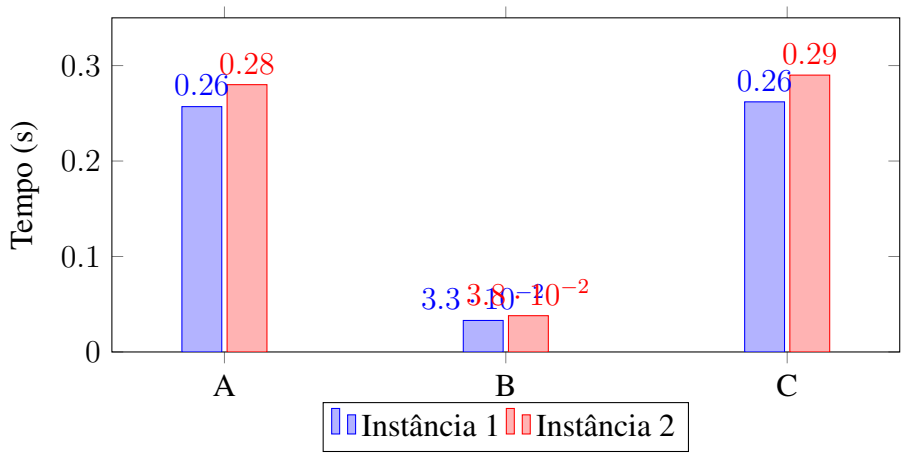

Figura 2. Comparação dos tempos de execução por configuração

4. Discussão

Observa-se que, para ambas as instâncias, todas as configurações encontraram a mesma solução ótima (mesma distância), o que indica que o algoritmo é robusto nas configurações testadas. No entanto, há diferenças significativas no tempo de execução, especialmente entre a configuração B (com inicialização heurística e critério de convergência) e as demais. Isso sugere que o uso de uma população inicial mais qualificada, aliado a um critério de parada eficiente, pode reduzir consideravelmente o tempo necessário para alcançar uma solução de qualidade.

A Configuração B apresentou o melhor desempenho em termos de tempo em ambas as instâncias, sendo quase 8 vezes mais rápida que as configurações A e C na Instância 1. Isso demonstra a efetividade da estratégia heurística e da convergência antecipada em evitar processamento desnecessário.

5. Conclusão

Os resultados demonstraram que, embora todas as configurações tenham encontrado a mesma solução ótima em termos de distância total para ambas as instâncias, houve

diferenças significativas nos tempos de execução. A Configuração B, que utilizou inicialização heurística e critério de parada por convergência, apresentou desempenho superior em termos de tempo, sendo até 8 vezes mais rápida que as demais.

A análise evidencia que a qualidade da solução pode ser mantida mesmo com redução expressiva no tempo de processamento, desde que operadores e estratégias adequadas sejam escolhidas. Isso reforça a importância da seleção criteriosa dos parâmetros do AG, sobretudo em aplicações com maiores volumes de dados, onde o tempo de execução é crítico.

Referências

[1] HOLLAND, J. H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[2] GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[3] GREFENSTETTE, J. J. Genetic algorithms for traveling salesman problems. In: *Proceedings of the 1st International Conference on Genetic Algorithms*, 1985. p. 160–168.

[4] TSALGATIDIS, A.; VRAKAS, K.; BASSILIADES, N.; VLAHAVAS, I. A Comparative Study of Genetic Operators for the TSP. *Applied Soft Computing*, v. 96, 2020. DOI: 10.1016/j.asoc.2020.106608.

[5] DAVIS, L. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, 1991.