UNIVERSIDADE FEDERAL DE ALAGOAS
Instituto de Computação

Matheus Gomes de Oliveira

Uma Abordagem do Algoritmo Genético com Foco na Inicialização para o Problema do Caixeiro Viajante

Maceió - AL
2023

Matheus Gomes de Oliveira

Uma Abordagem do Algoritmo Genético com Foco na Inicialização para o Problema do Caixeiro Viajante

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal de Alagoas – UFAL, Campus A. C. Simões.

Orientadora: Profª. Dr. Roberta Vilhena Vieira Lopes

Maceió - AL
2023

Catalogação na Fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico

Bibliotecário: Marcelino de Carvalho Freitas Neto – CRB-4 - 1767

O48a Oliveira, Matheus Gomes de.
Uma abordagem do algoritmo genético com foco na inicialização para o problema do caixeiro viajante/ Matheus Gomes de Oliveira. – 2023.
57 f. : il.

Orientadora: Roberta Vilhena Vieira Lopes.
Monografia (Trabalho de conclusão de curso em Ciência da Computação) – Universidade Federal de Alagoas, Instituto de Computação. Maceió, 2023.

Bibliografia: f. 55-57.

1. Algoritmos genéticos - Inicialização. 2. Problema do caixeiro viajante. 3. Logística - Alagoas. I. Título.

CDU: 004.421

Agradecimentos

Agradeço primeiramente, e acima de tudo, aos meus pais: Joel Silva de Oliveira e Maria Bernadete Gomes dos Santos de Oliveira por sempre acreditarem em mim, me apoiar, aconselhar e incentivar nessa caminhada.

A UFAL e todo corpo docente do curso de Ciência da Computação, por me proporcionar um ambiente acolhedor e de muito aprendizado e crescimento profissional.

Aos meus orientadores, Roberta Vilhena Viera Lopes e João Victor Ribeiro Ferro que me ajudaram em todo o percurso desse trabalho, me orientando, mostrando os melhores caminhos e acima de tudo acreditando na minha capacidade. Em especial ao amigo João Victor Ferro Ribeiro que aceitou, mesmo com seus afazares e compromissos, me orientar e me ajudou em todos os momentos desta caminhada do início ao fim deste trabalho.

Aos professores Evandro de Barros Costa e Victor Heuer pelas oportunidades e disponibilidade no desenvolvimento da minha trajetória acadêmica.

Agradeço aos meus amigos e colegas de curso, com quem convivi durante os últimos anos, que tornaram essa jornada mais divertida e única. Em especial a José Rubens da Silva Brito, Ramon Basto Callado, João Victor Ferro Ribeiro ,Lucas Albuquerque Lisboa, William Gabriel da Paz Rosendo, Antero Arcanjo de Assunção Júnior, Wagner Anthony de Medeiros Silva, Michael Miller Rodrigues Cardoso.

Agradeço ao corpo técnico, em especial a Ana Ferreira e Marcelo Gusmão pelo apoio fornecido nas demandas burocráticas.

Agradeço também a minha namorada Millena Thalyne Silva Ramos que esteve ao meu lado e me ajudou também nesta caminhada me incentivando e dando apoio no que precisei.

Agradeço aos professores que aceitaram compor a banca examinadora.

O primeiro requisito para uma ordem social melhor é o retorno à liberdade irrestrita de pensamento e de expressão.

(Ludwig Von Mises)

Resumo

Os Algoritmos Genéticos são métodos de busca heurística que simulam o processo evolutivo conforme explicado na Teoria da Evolução de Charles Darwin. Segundo essa teoria, os indivíduos que possuem adaptações mais vantajosas ao ambiente são os que têm maior probabilidade de sobreviver. Este método de busca é particularmente eficaz na resolução de um problema clássico da literatura, conhecido como o Problema do Caixeiro Viajante, que se enquadra na categoria NP-Difícil. Este problema tem ampla aplicação em cenários que envolvem roteamento, logística e otimização de recursos. Neste trabalho, será introduzida uma abordagem inovadora para a geração da população inicial dos Algoritmos Genéticos. O principal objetivo é apresentar um novo método de inicialização específico para o problema em discussão e, ao mesmo tempo, realizar uma simulação abordando um cenário de distribuição logística com estoque predefinido no estado de Alagoas. Para a avaliação do algoritmo proposto, será comparado os resultados do mesmo com outra implementação simples do Problema do Caixeiro Viajante, utilizando as mesma métricas e base de dados. Será realizada uma avaliação considerando o desempenho de cada algoritmo, levando em consideração o seu *score* por geração, o número de gerações em que ambos encontraram a melhor solução e o tempo de execução que cada algoritmo demandou para atingir essa solução. O estudo em pauta apresentou resultados favoráveis nas duas primeiras métricas de avaliação, apenas demonstrando um desempenho inferior em relação ao aspecto de tempo.

Palavras-chaves: Algoritmos Genéticos, Alagoas, Logística, TSP, Problema do Caixeiro Viajante.

Abstract

Genetic Algorithms are heuristic search methods that replicate the evolutionary process as explained in Charles Darwin's Theory of Evolution. According to this theory, individuals with more advantageous adaptations to their environment are more likely to survive. This search method is particularly effective in solving a classic problem in the literature known as the Traveling Salesman Problem, which falls into the NP-Complete category. This problem has broad applications in scenarios involving routing, logistics, and resource optimization. In this work, we introduce an innovative approach based on Genetic Algorithms. The primary goal is to present a new initialization method specific to the discussed problem while simultaneously conducting a simulation addressing a logistics distribution scenario with predefined stock in the state of Alagoas. To evaluate the proposed algorithm, we will compare its results with another simple implementation of the Traveling Salesman Problem, using the same metrics and dataset. An assessment will be conducted considering the performance of each algorithm, taking into account their score per generation, the number of generations in which they found the best solution, and the execution time required for each algorithm to reach that solution. The study at hand yielded favorable results in the first two evaluation metrics, with only a relatively lower performance in terms of execution time.

key-words: AG, Genetic Algorithm, Logistic ,Alagoas, TSP, Travelling Sales Problem.

Sumário

1 Introdução **12**
1.1 Motivação e Justificativa . . . 13
1.2 Objetivo Geral . . . 16
1.2.1 Objetivos Específicos . . . 16
1.3 Estrutura do Trabalho . . . 16

2 Fundamentação Teórica **18**
2.1 Algoritmo Genético de Holland . . . 18
2.1.1 Representação do individuo . . . 20
2.1.2 Tipos de Inicialização . . . 20
2.1.3 Função de adaptação . . . 21
2.1.4 População . . . 22
2.1.5 Operadores Genéticos . . . 22

3 Revisão da Literatura e Análise Bibliométrica **27**
3.1 Revisão da Literatura . . . 27
3.2 Metodologia . . . 29
3.3 Análise bibliométrica . . . 30
3.4 Análise dos Resultados . . . 30
3.4.1 Q1 - Quais são os principais autores e instituições que publicam na área de estudo? . . . 31
3.4.2 Q2 - Quais são os países que trabalham no assunto? . . . 32
3.4.3 Q3 - Quais são as principais revistas e conferências que publicam sobre o assunto? . . . 33
3.4.4 Q4 - Quais são os principais temas de pesquisa na área? . . . 34
3.4.5 Q5 - Quais são os trabalhos mais citados? . . . 36
3.5 Considerações Finais . . . 37

4 Metodologia **38**
4.1 Ambiente de Teste . . . 38
4.2 Descrição do problema . . . 40
4.3 Descrição das técnicas utilizadas . . . 42
4.4 Processamento dos Dados . . . 42
4.5 O algoritmo proposto . . . 43
4.6 Métricas de avaliação . . . 44

5 Resultados e Discussão **45**
5.1 Comparação entre algoritmos . . . 45
5.2 Teste de Hipótese . . . 49

6 Conclusão e Trabalhos Futuros **53**
6.1 Conclusão 53
6.2 Trabalhos Futuros 54

Referências bibliográficas **55**

Lista de Figuras

1.1 Espaço amostral (TSP) – Fonte: Autor (2023) 15
1.2 Espaço amostral Proposto com Quadrantes – Fonte: Autor (2023) 15

2.1 Máximo Global e Local em uma função hipotética – Adaptado (LINDEN, 2008) 19
2.2 Representação do pseudocódigo de AG – Adaptado (KATOCH; CHAUHAN; KUMAR, V., 2021) 19
2.3 Operação de cruzamento (Adaptado de (YANG, X.-S., 2020)) 23
2.4 Mutação por Permutação (Adaptado de (KATO R. PAIVA V., s.d.)) 24
2.5 Exemplo de Seleção Elitista – (Adaptado de (FERRO, 2022)) 26

3.1 Protocolo Prisma dos outputs da RSL – Fonte: Autor 29
3.2 Autores e Instituições (Autor, 2023) 31
3.3 Países (Autor, 2023) 33
3.4 Palavras chaves dos Artigos (Autor, 2023) 34

4.1 Representação do Problema do Caixeiro Viajante – Adaptado (ANDERS WIRÉN, s.d.) 40
4.2 Mapa geopolítico de Alagoas – Fonte: IBGE 41

5.1 Boxplot ilustrativo dos Resultados – Fonte: Autor (2023) 47
5.2 Gráfico de linhas com a relação Score/Geração – Fonte: Autor (2023) 48
5.3 Gráfico de linhas com a relação Tempo/Execução – Fonte: Autor (2023) 49
5.4 Gráfico de linhas com a relação Geração/Execução – Fonte: Autor (2023) . . . 50

Lista de Tabelas

3.1 Questões para Extração de Dados 31
3.2 Principais Autores 32
3.3 Principais Instituições 32
3.4 Principais Países 33
3.5 Conferências e Publicações Sobre o Escopo da Pesquisa 34
3.6 Artigo e Citações 36

5.1 Estatísticas Descritivas 46
5.2 Teste de normalidade de Shapiro-Wilk 50
5.3 T student para amostras independentes ($H0 \neq H1$) 51
5.4 T student para amostras independentes (H0 > H1) 52

Lista de Abreviaturas e Siglas

ACO	*Ant Colony Optimization*
AG	*Algoritmo Génetico*
CAPES	*Coordenação de Aperfeiçoamento de Pessoal de Nível Superior*
CBTU	*Companhia Brasileira de Trens Urbanos*
csv	*comma-separated-values*
DER	Departamento de Estradas e Rodovias
DF	*Data-Frame*
DP	Desvio Padrão
GL	Grau de Liberdade
GWO	*Gray Wolf Optimization*
PRISMA	*Principais Itens para Relatar Revisões Sistemáticas e Meta-análises*
PSO	*Particle Swarm Optimization*
TSP	*Traveling Salesman Problem*

Capítulo 1

Introdução

Os algoritmos genéticos emulam o processo de evolução das espécies conforme descrito na teoria de Darwin (DARWIN, 2004). Eles combinam esses princípios com conceitos da biologia genética, como hereditariedade, seleção, cruzamento e mutação (POZO et al., 2005). O objetivo é explorar o espaço de soluções em busca de artifícios ótimos globais para problemas complexos, realizando tanto uma busca ampla quanto uma busca detalhada no espaço de investigação do problema.

Entre vários componentes que constitui um Algoritmo Genético, um dos mais importantes é a inicialização, isto é, o processo de criar a primeira geração de indivíduos, conhecida como população inicial. Essa etapa é fundamental, pois define o ponto de partida da busca por soluções em um problema específico. A maneira como os indivíduos são inicializados pode ter um impacto significativo no desempenho do algoritmo genético.

A escolha da estratégia de inicialização depende do problema em questão. Em muitos casos, é útil experimentar diferentes abordagens de inicialização para determinar qual funciona melhor para a otimização de um problema específico. Além disso, a qualidade da inicialização pode afetar na busca da solução satisfatória e no tempo que o algoritmo genético leva para convergir. Portanto, a inicialização é uma etapa crítica na configuração de um algoritmo genético eficaz.

O problema que será abordado em questão é o do Caixeiro Viajante, em inglês conhecido como *Traveling Salesman Problem* – TSP, que se trata de um dos problemas mais estudados em otimização combinatória e é frequentemente usado como um desafio em ciência da computação e matemática. Ele pode ser formulado da seguinte maneira: dado um conjunto de cidades e as distâncias entre todas as combinações possíveis de pares de cidades, o objetivo do TSP é

encontrar o caminho mais curto que começa em uma cidade, visita cada cidade exatamente uma vez e retorna à cidade de origem. Em outras palavras, o Problema do Caixeiro Viajante responde à pergunta: "Qual é o caminho mais curto que um caixeiro viajante deve seguir para visitar todas as cidades em sua lista, exatamente uma vez cada, e retornar ao ponto de partida?"Uma forma para resolver esta problemática é justamente utilizando uma instanciação do Algoritmo Genético para encontrar o melhor caminho ao percorrar todas as cidades e retorne ao ponto inicial.

Neste estudo, o foco está em abordar o problema do Caixeiro Viajante de forma mais ampla, com foco na sua inicialização, de forma que aumente a precisão do *Score* (pontuação usada como métrica para definir a precisão da execução em relação ao caminho percorrido) por meio de uma melhor inicialização, utilizando quadrantes pré-definindos, individualmente, que possibilitem o espaço amostral para cada cidade. Além disso, como se trata de um problema logístico, foi utilizado parâmetros a fim de simular uma situação real de distribuição com estoque.

Como resultado preliminar desta pesquisa, foram comparados o Algoritmo Genético comum e o algoritmo com a inicialização proposta, com a finalidade de validar os resultados obtidos e mostrar a superioridade em relação ao Problema do Caixeiro Viajante. Como resultados, como pode ser observado no Capítulo 5, foi mostrado que realmente temos uma melhora do valor de *score* do algoritmo apresentado em relação a uma simples implementação relacionado ao TSP. No entanto, também foi observado que o método ainda pode ser melhorado quando usa-se como métrica de avaliação o tempo de execução em que o algoritmo comum ainda pode ser recomendado quando não se dispõe de um poder computacional robusto.

1.1 Motivação e Justificativa

A logística no Brasil é um aspecto crítico para a economia do país, dado o seu tamanho geográfico e a sua importância como uma das maiores economias da América Latina. A logística envolve o transporte, a armazenagem, a distribuição e a gestão de mercadorias e recursos em toda a cadeia de suprimentos, desempenhando um papel crucial no comércio interno e internacional. O Brasil possui uma extensa rede de transporte que inclui rodovias, ferrovias, hidrovias, portos e aeroportos, mesmo que a qualidade e a capacidade dessa infraestrutura varie significativamente em todo o país. As rodovias muitas vezes enfrentam problemas de

manutenção e congestionamento, enquanto o transporte ferroviário e hidroviário não são tão amplamente utilizados quanto poderiam ser.

No cenário do estado de Alagoas não é diferente, tem-se uma predominância do modelo rodoviário, já que nossa malha ferroviária é quase nula, que só conta com 35 km de extensão segundo o Portal da *Companhia Brasileira de Trens Urbanos* – CBTU de Maceió (MACEIÓ, s.d.), o estado como um todo também só conta com um Porto e um Aeroporto, todos centralizados na capital do estado. Sendo assim, pode-se dizer que majoritariamente o escoamento de qualquer coisa vinda de portos e aeroportos sai pelas estradas e isso requer uma estrátegia de logística mais aperfeiçoada, pois os custos por transporte rodoviário é mais elevado que as demais opções citadas.

Neste contexto, o mercado de distribuição tem um grande potencial de crescimento para empresas de logísticas, visto que a maioria das grandes distribuidoras firmam suas sedes na capital e, delas, suas mercadorias são enviadas para os demais munícipios do estado. Isso é facilmente visto em um anúncio divulgado pela Prefeitura de Maceió (NIVIANE RODRIGUES, s.d.) onde 3 mil empregos diretos serão gerados por meio de grandes empresas de distribuição e logística no bairro da Cidade Universitária, corroborando com o que foi mencionado. Com isso, se faz necessário terem mecânismos estratégicos para o melhor uso destes recursos resultando em um bom caminho logístico de tudo que chega no estado, pois, atualmente, se encontra centralizado em uma única cidade.

Com o objeto do problema definido, tem-se como uma solução computacional para este conflito, que se encaixa na problemática do Caixeiro viajante, um algoritmo genético que cálcula o melhor caminho para percorrer os munícipios e voltar a cidade inicial. Na implementação comum do problema do TSP, considera-se como espaço amostral o apresentado na seguinte figura:

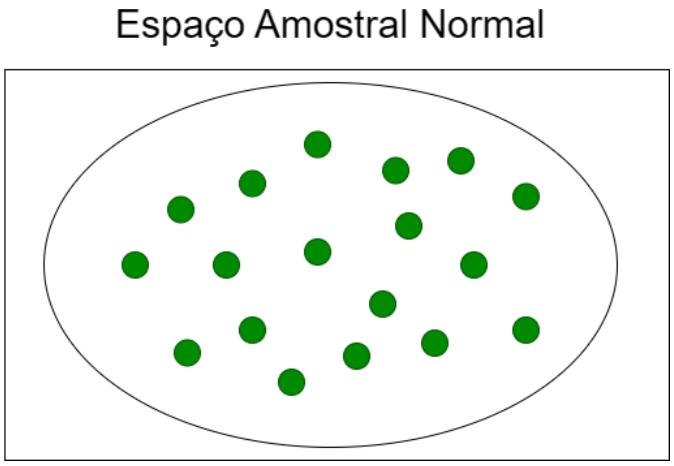


Figura 1.1: Espaço amostral (TSP) – Fonte: Autor (2023)

Na Figura 1.1, tem-se a disposição normal dos pontos em um problema clássico do TSP, onde pode-se imaginar como sendo os pontos das diversas cidades presentes no estado de Alagoas. Nela, um algoritmo genético comum irá percorrer todos os 101 munícipios e voltar para capital. Isso já pode ser utilizado como uma solução, contudo na proposta desta monografia, a disposição dos pontos será feitos por quadrantes, que, de forma individual, calculariam qual seria o melhor caminho percorrido. Dessa forma, pode-se exemplificar como os quadrantes da abordagem apresentada funcionará, conforme pode ser observado na Figura 1.2.

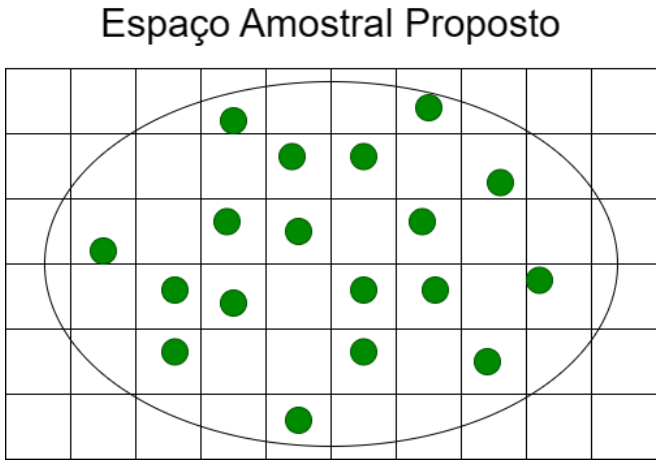


Figura 1.2: Espaço amostral Proposto com Quadrantes – Fonte: Autor (2023)

Como pode ser observado na figura, é apresentado uma individualização de cada ponto com o intuito de trata-los de maneira única, antes de construir a solução. O objetivo dessa abordagem é ter um representante de cada quadrante na população inicial. Isso, como pôde ser observado nos resultados, tem um custo computacional, porém se justifica ao apresentar

melhores resultados que uma implementação comum, já conhecida, de um *Algoritmo Génetico* – AG, para o problema proposto.

1.2 Objetivo Geral

O objetivo geral deste trabalho é propor uma nova abordagem para o problema do Caixeiro Viajante a partir de um novo tipo de inicialização da população, com o objetivo de ter um representante de cada quadrante na mesma.

1.2.1 Objetivos Específicos

Este trabalho tem como objetivos específicos:

- Investigar na literatura o que há de implementado em algoritmos genéticos para o TSP, focando em modelos de inicialização, por meio de uma revisão sistemática.
- Construir uma base de dados com as coordenadas decimais de cada município alagoano e suas respectivas distâncias euclidianas entre eles.
- Realizar as execuções dos algoritmos a fim de consolidar resultados de forma consistente, e, com um número relevante estatisticamente de execuções,para que assim, se possa realizar as análises necessárias
- Analisar os resultados obtidos e compará-los com um algoritmo genético convencional para validar o algoritmo proposto.

1.3 Estrutura do Trabalho

Este documento está estruturado em seis capítulos. No primeiro, será apresentado uma introdução sobre o trabalho, contextualizando o tema da pesquisa, motivando a sua realização, justificando-a, identificando a problemática e as contribuições principais, bem como os objetivos para a sua execução. Os demais capítulos são descritos a seguir:

- **Capítulo 2:** Apresentar a fundamentação teórica que servirá de pilar para o entendimento da problemática e justificativa para proposta apresentada.

- **Capítulo 3:** Apresentar a revisão sistemática e bibliométrica da literatura em decorrência à utilização de técnicas de otimização na inicialização no TSP.
- **Capítulo 4:** Apresentar a metodologia da pesquisa com o passo a passo para sua realização.
- **Capítulo 5:** Apresentar resultados obtidos com o objetivo de validar a abordagem proposta e, também, apresentar testes de hipóteses que justifiquem sua realização.
- **Capítulo 6:** Apresentar a conclusão do trabalho e as considerações finais obtidas com a abordagem proposta, além de apresentar encaminhamentos de trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Neste capitulo, serão abordados os conceitos que fundamentam os assuntos ligados a este estudo, seguindo uma ordem que demonstra a importância de cada bloco para a construção do algoritmo genético implementado no problema central deste trabalho.

2.1 Algoritmo Genético de Holland

O algoritmo genético desenvolvido por Holland é uma abordagem utilizada para encontrar a solução próxima da ótima global, geralmente utilizada em problemas complexos, ou seja, que exigem um tempo computacional muito alto. Geralmente é utilizado para otimizar problemas combinatoriais, esse algoritmo se comporta como uma busca estocástica, guiada pela adaptação dos indivíduos que fazem parte da população atual (MARIANO et al., 2021). Por outro lado, quando aplicado à otimização de problemas funcionais, esse algoritmo é considerado uma técnica de otimização global, pois não utiliza a primeira ou segunda derivada da função, como ocorre no algoritmo *hill climbing* (MIRANDA, 2007). Essa característica permite que ele escape dos máximos locais e seja empregado em funções descontínuas (LINDEN, 2008). Abaixo, na Figura 2.1 temos uma representação que pode melhor ilustrar esse comportamento:

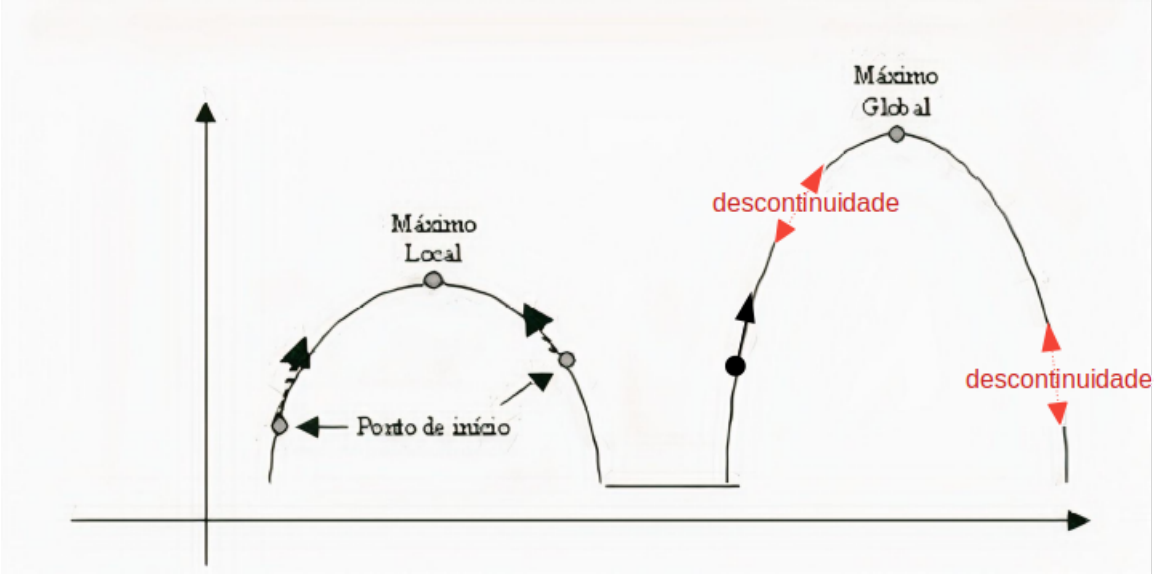


Figura 2.1: Máximo Global e Local em uma função hipotética – Adaptado (LINDEN, 2008)

Sendo assim, este algoritmo é um método iterativo no qual a população de indivíduos evolui ao longo do tempo, sendo que cada indivíduo representa uma possível solução para o problema em questão. Essas iterações são chamadas de gerações do algoritmo genético, e é durante essas gerações que ocorre o processo de evolução da população. Em cada iteração, os indivíduos são selecionados para gerar descendentes, que, por sua vez, são submetidos aos operadores genéticos e, por fim, compõem a nova população (MARIANO et al., 2021). Abaixo, na Figura 2.2, temos um fluxograma que demonstra melhor este funcionamento.

Entrada:
Tamanho da população, n
Número máximo de iterações, MAX
Saída:
Melhor solução global, Y_{bt}

Início
Gere uma população inicial de n cromossomos Y_i (i=1,2, ..., n)
Defina o contador de iterações t = 0
Calcule o valor de aptidão de cada cromossomo
Enquanto *(t < MAX)*
Selecione um par de cromossomos da população inicial baseado na sua aptidão
Aplique a operação de cruzamento no par de cromossomos selecionados com a probabilidade de cruzamento
Aplique a operação de mutação nos descendentes com a probabilidade de mutação
Substitua a antiga população pela nova população gerada
Incremente o contador t em 1
fim enquanto
retorne a melhor solução, Y_{bt}
fim

Figura 2.2: Representação do pseudocódigo de AG – Adaptado (KATOCH; CHAUHAN; KUMAR, V., 2021)

2.1.1 Representação do individuo

No algoritmo de Holland, os indivíduos são representados por meio de estruturas chamadas de cromossomos. Este cromossomo é um dos componentes fundamentais dos AG, visto que é nesse momento que começa a modelagem do problema proposto (EIBEN; SMITH, 2015). Os cromossomos são compostos por genes, que representam as características ou parâmetros do problema que está sendo otimizado. A representação dos indivíduos pode variar dependendo do tipo de problema e das características que estão sendo otimizadas. Alguns exemplos comuns de representação incluem:

1. **Representação Binária**: Os cromossomos são compostos por uma sequência de bits, onde cada bit representa uma característica ou parâmetro do problema. Por exemplo, em um problema de otimização de um vetor binário, cada gene do cromossomo pode representar um elemento do vetor, sendo 0 ou 1.

2. **Representação numérica**: Os cromossomos são compostos por sequências de valores numéricos. Cada gene do cromossomo representa um parâmetro numérico do problema. Por exemplo, em um problema de otimização de uma função matemática, cada gene pode representar um valor real.

3. **Representação de cadeias de caracteres**: Os cromossomos são compostos por sequências de caracteres. Cada gene do cromossomo representa uma parte ou característica do problema. Por exemplo, em um problema de otimização de uma sequência de palavras, cada gene pode representar uma palavra.

2.1.2 Tipos de Inicialização

Existem várias estratégias comuns de inicialização em algoritmos genéticos, (GUIMARAES et al., 2017) sendo as principais:

- **Inicialização aleatória**: Nessa abordagem, os indivíduos da população inicial são criados aleatoriamente. Isso proporciona uma exploração ampla do espaço de busca, mas pode levar mais tempo para convergir para soluções de alta qualidade.

- **Inicialização baseada em heurísticas**: Às vezes, é possível usar informações específicas do problema para criar uma população inicial que já tenha algum conhecimento

sobre a natureza do problema. Isso pode acelerar a convergência para soluções de melhor qualidade.

- **Inicialização baseada em soluções conhecidas**: Se você possui informações prévias sobre boas soluções para o problema, pode usar essas soluções como ponto de partida para a população inicial. Isso pode ser útil para melhorar rapidamente a qualidade das soluções.

- **Inicialização baseada em amostras**: Em alguns casos, é possível usar amostras de dados reais ou conhecimento sobre o domínio do problema para criar a população inicial. Isso ajuda a iniciar o algoritmo com uma população que reflita melhor as características do problema real.

2.1.3 Função de adaptação

A função de aptidão desempenha um papel fundamental no algoritmo genético, pois indica o quão bem um indivíduo se adapta ao problema em questão. Ela atribui um aumento positivo na aptidão aos indivíduos que pertencem ao domínio do problema, enquanto aqueles que não estão adaptados recebem um aumento negativo. Essa função desempenha um papel crucial ao guiar a busca do algoritmo genético em direção à solução ótima global do problema. Uma vez que os algoritmos genéticos são baseados em estratégias de maximização, é importante que o valor de aptidão do cromossomo seja maior para soluções consideradas melhores em comparação com outras. Por exemplo, se o cromossomo x_1 representa uma solução superior ao cromossomo x_2, então o valor de aptidão atribuído a x_1 deve ser maior do que o atribuído a x_2 (LINDEN, 2008). Dessa forma, a função de aptidão desempenha um importante papel ao orientar a evolução da população, permitindo que os indivíduos mais adaptados tenham maior probabilidade de serem selecionados como pais para a próxima geração, contribuindo para a busca pela solução ótima do problema.

É necessário ter cuidado ao lidar com problemas de otimização que envolvem a busca pelo valor mínimo. Isso ocorre porque os algoritmos genéticos são projetados para buscar os cromossomos mais adaptados, por definição. Portanto, ao aplicar um algoritmo genético a esse tipo de problema, é importante considerar que o novo valor de aptidão será:

- A negação do valor real de aptidão do problema;

- Uma fração em que o numerador é 1 e o denominador é o valor real de aptidão do problema representado pelo cromossomo.

Essas abordagens permitem que o algoritmo genético seja ajustado para encontrar o valor mínimo, ao invés do valor máximo, conforme exigido pelo problema de otimização em questão.

2.1.4 População

Nos Algoritmos Genéticos, a população é representada por um conjunto limitado de cromossomos, contendo um tamanho fixo ou variável. Ela armazena os possíveis resultados encontrados até o momento para o problema em questão, conforme descrito por Eiben e Smith (EIBEN; SMITH, 2015). Esta é utilizada como entrada para o algoritmo genético e, dependendo da natureza do problema, a população inicial pode ser gerada aleatoriamente ou obtida do ambiente. A população é definida como um vetor de dimensão, no qual cada linha representa um cromossomo, sendo estabelecida como:

$$\vec{p}_1 = \langle p_1, p_2, p_3 \rangle \, onde, \forall \epsilon \, \{1, 2, 3\} \, p_j \epsilon C_v \tag{2.1}$$

Onde, explicando de maneira simples e objetiva, o nosso P_1 seria o individuo da população em que sua inicialização passa diretamente pela inclusão dos demais membros da população onde cada p_1, p_2, p_3 está dentro do conjunto da Caixeiro.

2.1.5 Operadores Genéticos

Os operadores genéticos têm como objetivo transformar a população de indivíduos, representada por x, em um resultado satisfatório, gerando novos indivíduos com base no material genético de seus pais. Esses operadores trabalham sob o princípio da tensão entre *exploration* e *explotation*, sendo *exploration* a busca por novas adaptações e *explotation* a manuntenção dessas adaptações feitas na geração atual, conforme denominado por Holland (LINDEN, 2008).

Embora grande parte das ações realizadas pelos operadores genéticos possa ser considerada destrutiva, por se tratar de uma seleção que apenas prioriza os melhores indivíduos e por consequência eliminando os demais, elas se enquadram na categoria de exploração. Isso significa que ocorre a busca por indivíduos com avaliação melhor do que a de seus pais (LINDEN, 2008). Os operadores genéticos desempenham um papel fundamental na geração de diversi-

dade e na melhoria gradual da qualidade dos indivíduos ao longo das gerações, permitindo a exploração de diferentes regiões do espaço de busca procurando a solução mais promissora.

Operador de Cruzamento

A etapa de cruzamento, também conhecida como *crossover*, é empregada para criar novas populações por meio da combinação de informações presentes em dois cromossomos existentes. Essa combinação ocorre com uma probabilidade p determinada, simulando a recombinação de material genético observada na reprodução sexuada. Holland (BOOKER; GOLDBERG; HOLLAND, 1989) introduziu essa operação que desempenha um papel importante no algoritmo genético ao promover a diversificação e a exploração de diferentes combinações de características genéticas.A Figura a seguir ilustra o processo de cruzamento entre dois cromossomos:

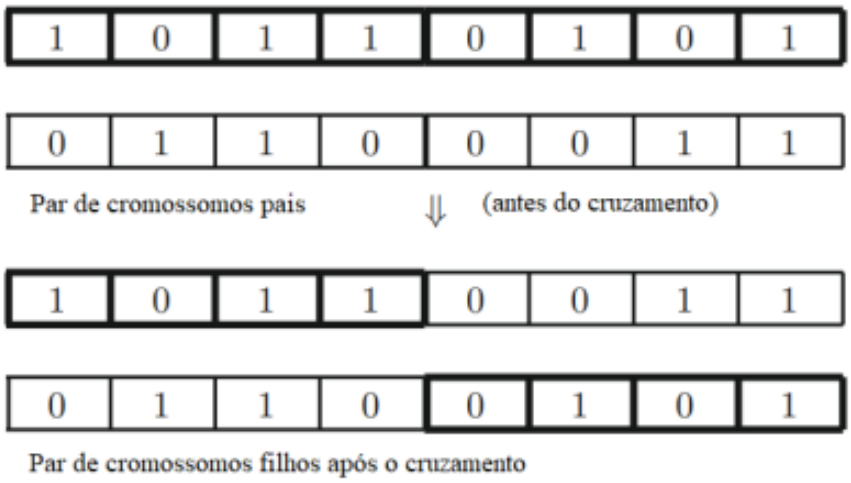


Figura 2.3: Operação de cruzamento (Adaptado de (YANG, X.-S., 2020))

Com base na ilustração, os dois cromossomos pais trocam material genético, resultando na geração de dois cromossomos filhos. Cada filho preserva um filamento de informação dos seus respectivos pais. Esse processo cria duas novas soluções que combinam características das soluções originais. O resultado é uma população com maior diversidade genética, o que pode facilitar a convergência para soluções mais otimizadas (GOLDBERG, 1989). Durante a operação de cruzamento, são definidos pontos de corte onde ocorre a extração do material genético dos pais. Portanto, nesta figura, foi utilizado apenas um ponto de corte na posição 3, sendo definidos de forma prévia ou gerados aleatoriamente. Além disso, é possível que um cromossomo possua múltiplos pontos de corte em algoritmos genéticos, o que aumenta sua eficiência evolutiva (YANG, X.-S., 2020).

Operador de Mutação

Para gerar variabilidade genética, que é similar ao surgimento de novos indivíduos na natureza, seja por exposição a radiação que causa modificações no DNA, seja por erros durante a troca de informações genéticas durante a reprodução sexuada, o algoritmo genético utiliza a operação de mutação (NUNES, 2018). Diversos tipos de mutação são descritos por SONI e KUMAR (SONI; KUMAR, T., 2014), como a mutação baseada na posição, a mutação baseada na ordem, a mutação de mistura e a por complemento. Cada uma delas recebe um cromossomo como entrada, altera seu material genético de acordo com um propósito específico e retorna um novo cromossomo diferente do cromossomo original. Essa operação de mutação é fundamental para introduzir diversidade genética na população e explorar regiões do espaço de busca que não seriam alcançadas apenas por meio do cruzamento dos cromossomos. Neste trabalho será tratado Operadores de Mutação baseado na permutação.

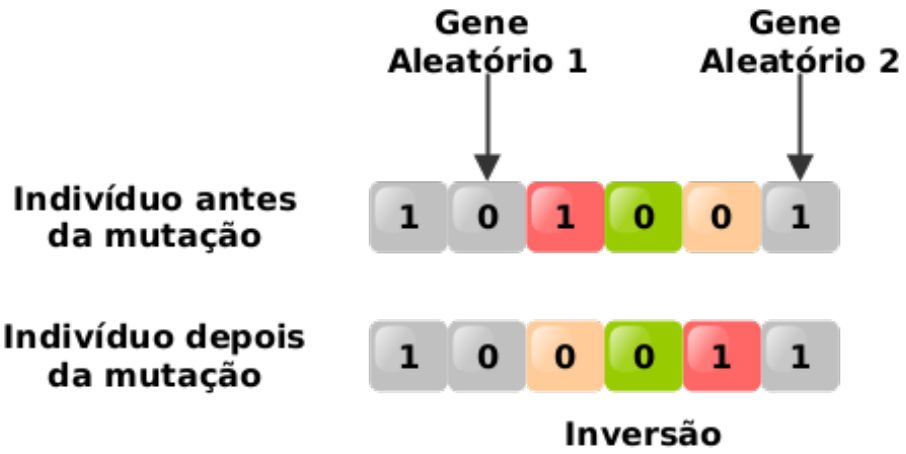


Figura 2.4: Mutação por Permutação (Adaptado de (KATO R. PAIVA V., s.d.))

Operador de mutação por permutação consiste em substituir os elementos no caso de problemas relacionados a ordenação, roteamento e escalonamento. Esses operadores permitem a manipulação e a geração de soluções novas a partir de permutações presentes nos cromossomos. Nesse contexto, é fundamental que o operador de mutação assegure que o novo indivíduo gerado seja uma permutação válida. Uma abordagem comum é a utilização de operadores de mutação que selecionam 2, 3 ou k pontos de corte no cromossomo e tentam trocar os segmentos entre esses pontos. No entanto, essa classe de operadores pode ser computacionalmente custosa. Nesse sentido, é importante considerar alternativas mais eficientes para a mutação em permutações. É possível explorar operadores de mutação baseados em técnicas como inver-

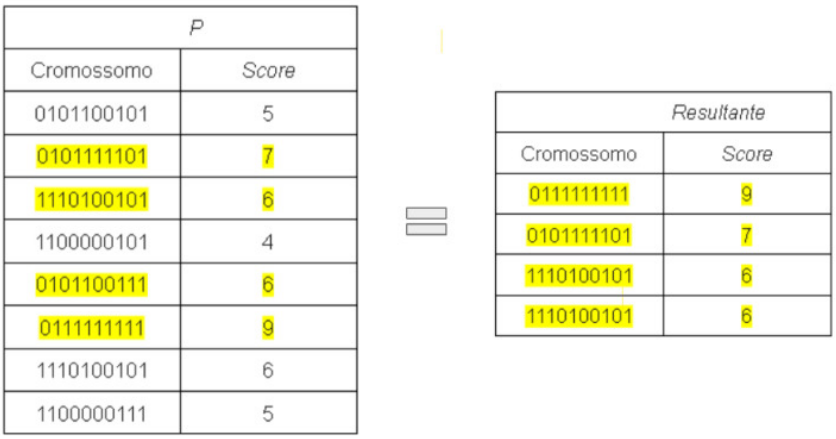
são de sub-sequências ou transposição de elementos, que são capazes de introduzir variações nas permutações de forma menos custosa, computacionalmente. Essas abordagens mais eficientes permitem a geração de novos indivíduos permutados de maneira válida, preservando a estrutura e as propriedades das trocas originais, ao mesmo tempo em que minimizam o custo computacional envolvido na operação de mutação(BÄCK; FOGEL; MICHALEWICZ, 2018). Os principais operadores genéticos de permutação são: crossover de permutação e a mutação por permutação.

Operador de Seleção

Assim como na Teoria da Evolução de Charles Darwin (DARWIN, 2004), a seleção no contexto dos algoritmos genéticos afirma que os indivíduos mais adaptados ao ambiente possuem maior probabilidade de sobrevivência e de transmitir suas características genéticas às gerações futuras por meio de seus descendentes (POZO et al., 2005). Esse princípio de seleção é fundamental para o funcionamento dos algoritmos genéticos, pois encaminha a busca em direção a soluções de maior qualidade. Os indivíduos com maior adaptação, ou seja, com valores de aptidão mais elevados, têm maior probabilidade de serem selecionados para a reprodução, possibilitando a transferência de suas características genéticas para a próxima geração. Dessa forma, ao longo das iterações do algoritmo genético, a seleção favorece a progressiva melhoria da população, aproximando-a de soluções ótimas ou de alta qualidade para o problema em questão. A seleção desempenha um papel crucial na busca por soluções mais adaptadas, impulsionando o processo evolutivo e permitindo a convergência para soluções promissoras.

Existem diferentes tipos de seleção que são utilizados em algoritmos genéticos. Neste trabalho, será adotada apenas a Seleção Elitista, conforme proposto por Eiben e Smith (EIBEN; SMITH, 2015). A escolha desse tipo de seleção é baseada na estratégia de gerenciamento da diversidade da população, isto é, tem como objetivo reduzir a diversidade da população, limitando a possibilidade de sobrevivência ou reprodução de indivíduos menos adaptados. Isso significa que apenas os indivíduos mais adaptados, os melhores em termos de aptidão, têm maiores chances de serem selecionados para a próxima geração. Essa abordagem visa direcionar o processo evolutivo em direção a soluções de melhor qualidade, favorecendo a convergência para soluções próximas da ótima. Ao restringir a diversidade da população, preserva as características dos indivíduos mais adaptados, permitindo que suas características genéticas sejam transmitidas para as gerações futuras. Sendo assim, a Seleção Elitista desempenha

um papel importante no algoritmo genético, garantindo que os indivíduos mais aptos tenham maior influência na evolução da população e contribuam para a busca por soluções de maior qualidade.



P	
Cromossomo	*Score*
0101100101	5
0101111101	7
1110100101	6
1100000101	4
0101100111	6
0111111111	9
1110100101	6
1100000111	5

Resultante	
Cromossomo	*Score*
0111111111	9
0101111101	7
1110100101	6
1110100101	6

Figura 2.5: Exemplo de Seleção Elitista – (Adaptado de (FERRO, 2022))

Na Figura 2.5, temos um exemplo em que uma população inicial P com 8 indivíduos passa por um processo de seleção elitista. Nesse processo, apenas 4 membros são escolhidos como ancestrais para a próxima geração, como ilustrado na população resultante. Esses membros selecionados têm a responsabilidade de realizar o processo de cruzamento.

O resultado disto na construção no Algoritmo Genético, em um contexto para a resolução do TSP, é a construção de um *Score*, que é formado pela soma do caminho percorrido pelo algoritmo através dos pontos (indivíduos) de toda a população. Essa pontuação que é obitda ao final da execução do AG é classificada em ordem descrecente, ou seja, quanto menor o valor de *Score* melhor foi a performance do algoritmo e isso tem uma explicação simples e objetiva: se o caminho é construído pelo peso das arestas e o objetivo é fazer o menor caminho possível em todos os pontos, logo quanto menor for esse número, melhor vai ser o caminho percorrido pelo Caixeiro.

Capítulo 3

Revisão da Literatura e Análise Bibliométrica

Este capítulo do trabalho tem por objetivo realizar uma compreensão da literatura com o intuito de apresentar os principais estudos e descobertas relacionados ao tema proposto. Essa revisão permitirá um embasamento sólido e uma compreensão abrangente do contexto em que o trabalho se insere.

3.1 Revisão da Literatura

Com o objetivo de obter uma visão abrangente do que já foi discutido na literatura sobre os temas abordados nesta monografia, foi conduzido uma revisão sistemática nas principais bases de conhecimento relevantes para a área de estudo no campo da computação. Para isso, foram exploradas duas bases de conhecimento amplamente utilizadas na comunidade acadêmica, acessíveis por meio do periódico *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* – CAPES: *Web of Science* e *Scopus*. Essa abordagem de pesquisa permitiu acessar uma gama diversificada de artigos científicos relevantes para o campo de estudo em questão. Ao utilizar essas bases de conhecimento renomadas, foi possível obter informações atualizadas e de alta qualidade, provenientes de fontes confiáveis e reconhecidas na comunidade acadêmica (NEPOMUCENO; COSTA; DARAIO, 2023). Dessa forma, a revisão sistemática da literatura desempenhou um papel fundamental na obtenção de um panorama completo e consistente sobre os assuntos abordados nesta monografia como, também, enriquecendo o embasamento teórico e contribuindo para a construção de um trabalho sólido e fundamentado.

Foi utilizado, em todas as bases citadas, duas *strings* de busca: uma de aspecto macro e outra específica. A *string* macro tem por objetivo pegar o máximo de trabalhos relacionados a área, que cita o problema do Caixeiro viajante (*Travelling Salesman Problem – TSP*), e, também, algoritmos genéticos em geral. A macro string pensada, e utilizada nas bases, foi: *("travelling salesman problem"OR "tsp") AND "genetic algorithm"*. Já para a *string* específica, obteve-se uma lista de termos mais específicos para priorizar os trabalhos que mencionasse uma nova abordagem relacionada com a inicialização da população dentro do problema do caixeiro viajante e, também, sempre procurando selecionar trabalhos correlatos ao problema da logística junto do TSP. Portanto, temos como resultado a *string específica* com os seguintes termos: *("travelling salesman problem"OR "tsp") AND ("genetic algorithm") AND ("initialization").*

Além das *strings*, também foram realizadas algumas filtragens que são importantes para delimitar o extrato dos resultados obtidos a fim de direcionar o máximo possível desejado: encontrar trabalhos relevantes em algoritmos genéticos que desenvolvem o problema do caixeiro viajante dentro de uma mutação específica. Então, além das três bases de conhecimentos, também restringimos os seguintes campos:

- Língua: somente a língua inglesa;
- Tipos desejados para o documento: Artigos em periódicos;
- Ter sido públicados nos últimos 5 anos;
- Áreas de conhecimento não desejadas: Engenharia, Ciências Humanas, Biológicas e Medicinais;
- Áreas de conhecimento desejadas: Computação, Data Science e Computação Evolucionária;
- Ser *open access*;
- Ter sido revisado por pares;

Alguns pontos interessantes que merecem menção acerca dos critérios de inclusão e exclusão são: foi priorizado trabalhos de acesso aberto a fim de que o acesso aos artigos fossem o mais democrático possível, além de restringir aos últimos 5 anos para deixar a pesquisa alinhada no que há de mais atual em relação a problemática.

3.2 Metodologia

Para dar mais validade a Revisão Sistemática da Literatura realizada para esta monografia, vou utilizar um protocolo *Principais Itens para Relatar Revisões Sistemáticas e Meta-análises* – PRISMA nos resultados obtidos. O PRISMA é um método criado para conduzir revisões sistemáticas da literatura de forma transparente e rigorosa. Ele fornece diretrizes detalhadas e padronizadas para o planejamento, execução e relato das revisões. O objetivo é promover a qualidade, transparência e replicabilidade dos estudos, garantindo uma seleção imparcial e um relato completo dos resultados (DARAIO et al., 2019). Abaixo temos o fluxograma do protocolo aplicado no estudo:

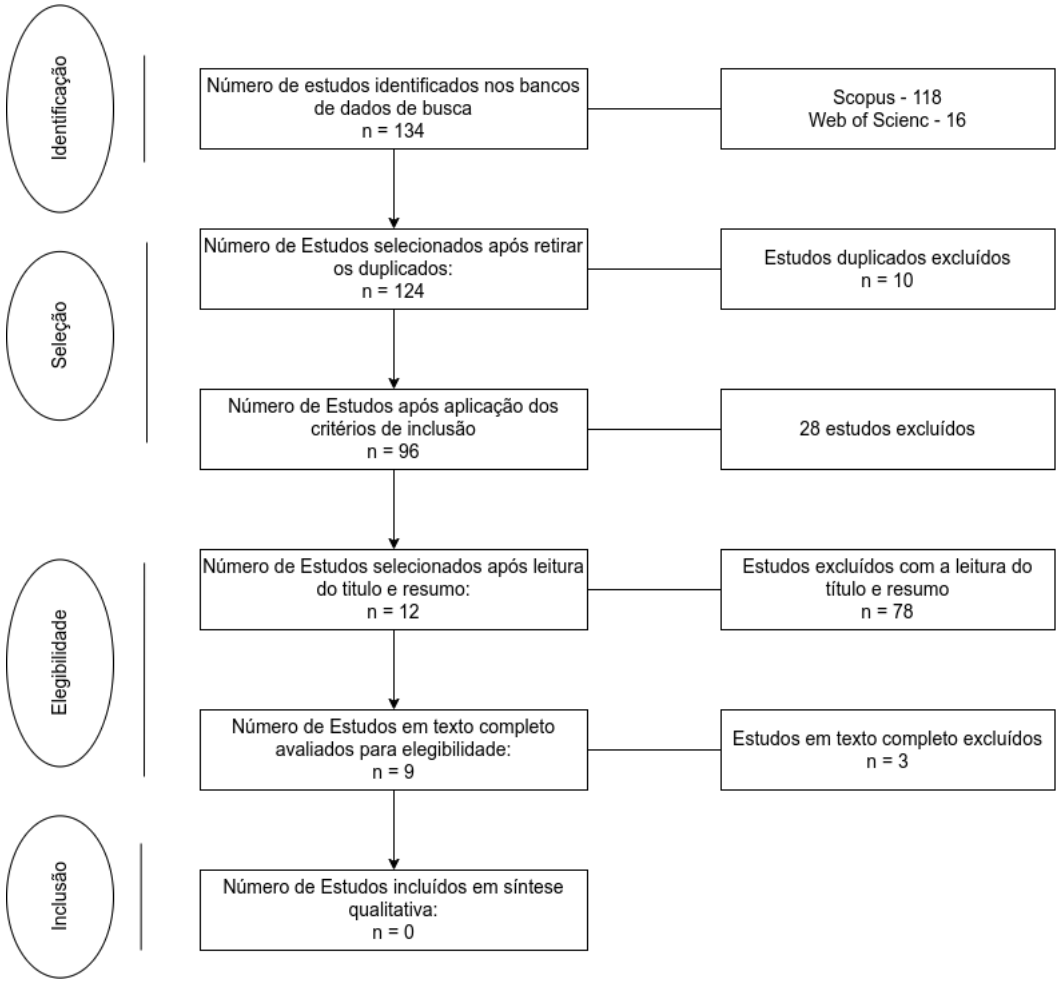


Figura 3.1: Protocolo Prisma dos outputs da RSL – Fonte: Autor

Como pode-se observar nos *steps* do protocolo PRISMA, foi obtido um total de 134 selecionados para *screening* seguindo os critérios e filtros mencionados na seção anterior. Destes 134 artigos, 118 artigos foram extraídos do *Scopus* e 16 da *Web of Science*. Foram encontradas 10 duplicidades de artigos, sobrando 124 para analíse de seus títulos e resumos. Destes, 12 arti-

gos foram aceitos seguindo critérios de inclusão que serão abordados na Análise Bibiométrica. Não foram incluídos nenhum artigo fora das bases selecionadas e nenhum artigo selecionado foi posteriormente excluído.

3.3 Análise bibliométrica

Nesta seção, serão realizadas análises bibliométricas considerando as citações, autores e co-autores dos artigos pesquisados. Além disso, serão identificados pontos de congruência entre os estudos, buscando elucidar suas correlações com o tema desta monografia. Essas análises proporcionarão uma compreensão mais abrangente do panorama bibliográfico relacionado ao tema em questão, fornecendo *insights* valiosos sobre as tendências, influências e relações existentes na literatura científica (NEPOMUCENO; COSTA; DARAIO, 2023).

Dessa forma, com o intuito de facilitar a explicação e o entendimento do andamento desta seleção, nestes trabalhos, foram criadas as seguintes perguntas:

- **RQ1:** Está relacionado ao problema do Caixeiro viajante?
- **RQ2:** Está relacionado a logística ou distribuição?
- **RQ3:** O artigo menciona ou propõe um novo tipo de inicialização?

Com essas perguntas, foi selecionado, em um universo de 124 artigos, 12 trabalhos que estão no escopo dos questionamentos acima. Muitos dos trabalhos elegidos para *screening*, apenas utilizavam o TSP a fim de resolver problemas de rotas para veículos autonomos, ou para determinar os melhores caminhos para algum drone, ou robô automatizado, em um contexto de indústria 4.0. Fora isso, muitos trabalhos respondiam positivamente a primeira e a segunda pergunta, porém não mencionava nada acerca da inicialização da população, que, para este trabalho, se faz muito importante, pois essa abordagem difere do que a monografia propõe. Abaixo, se encontra a tabela que será usada para a análise dos resultados obtidos a fim colaborar para trabalhos futuros que possam utilizar esta pesquisa:

3.4 Análise dos Resultados

Nesta seção, será abordado, de forma objetiva, os resultados obtidos nas bases de conhecimento selecionadas. Para isso, com o intuito de sanar de forma pragmatíca, foram analisados os

Tabela 3.1: Questões para Extração de Dados

	Questões
Q1	Quais são os principais autores e instituições que publicam na área de estudo?
Q2	Quais são os países que trabalham no assunto?
Q3	Quais são as principais revistas e conferências que publicam sobre o assunto?
Q4	Quais são as principais técnicas e abordagens de cada artigo que foi selecionado?
Q5	Quais são os trabalhos mais citados ?

resultados com base nas perguntas apresentadas na Tabela 3.1, mostrando o que foi extraído, em cada uma delas, com representações e dados estátisticos inseridos no VOSviewer [1] e o Google Sheets [2].

3.4.1 Q1 - Quais são os principais autores e instituições que publicam na área de estudo?

Com base nos artigos selecionados, foram coletadas as seguintes informações acerca dos autores, e instituições, que publicaram na área de estudo, voltado para cenário dos algoritmos genéticos atrelado ao problema do Caixeiro viajante vide a Figura3.2. Nela, pode-se observar os autores selecionados e as respectivas instituições nas quais seus *papers* foram publicados.

(a) Autores (b) Instituições

Figura 3.2: Autores e Instituições (Autor, 2023)

Além disso, temos a Tabela 3.2 com os cinco principais autores ordenados de forma decrescente pelo grau de relevância avaliado em cima do número de artigos na área e as citações que ele teve em seu trabalho.

[1] VosViewer: https://www.vosviewer.com/
[2] Google Sheets: https://www.google.com/sheets/about/

Autor	Artigos	Citações
Hassanat A.B.	1	38
Xu X.	1	33
Chen X.	1	27
Wu C.	1	23
Emambocus B.A.S.	1	10

Tabela 3.2: Principais Autores

Foi realizado um procedimento semelhante para identificar as principais instituições, onde os pesquisadores associados a cada instituição, ordenados de forma decrescente, com base no número de citações. Os resultados dessa análise estão apresentados na Tabela 3.3.

Instituição	Citações
Department of Information Technology, Mutah University	38
Department of Computer Science, Nanjing University of Posts and Telecommunications	33
College of Information and Computer Science, Southwest University	27
School of Management Science and Engineering, Shandong Institute of Business and Technology	23
Department of Computing and Information Systems, Sunway University	10

Tabela 3.3: Principais Instituições

3.4.2 Q2 - Quais são os países que trabalham no assunto?

O país com mais publicações na área de Algoritmos Genético, com foco no problema do caixeiro viajante e na inicialização, é a China, seguido da Jordânia, como pode ser observado na Figura 3.3, ilustrado pelo tamanho de seu círculo, que representa a densidade de pesquisadores de cada país. Além disso, é possível observar a diversidade de países que vem cooperando para o avanço no tema e que nenhum deles se concentra em um lugar específico do mundo. Na Tabela 3.4, é mostrado os cincos principais países em ordem decrescente, levando em consideração o número de publicações na área e as citações dos artigos de cada país.

País	Publicações	Citações
China	6	105
Jordânia	2	39
Estados Unidos	1	38
Reino Unido	1	33
Malásia	1	10

Tabela 3.4: Principais Países

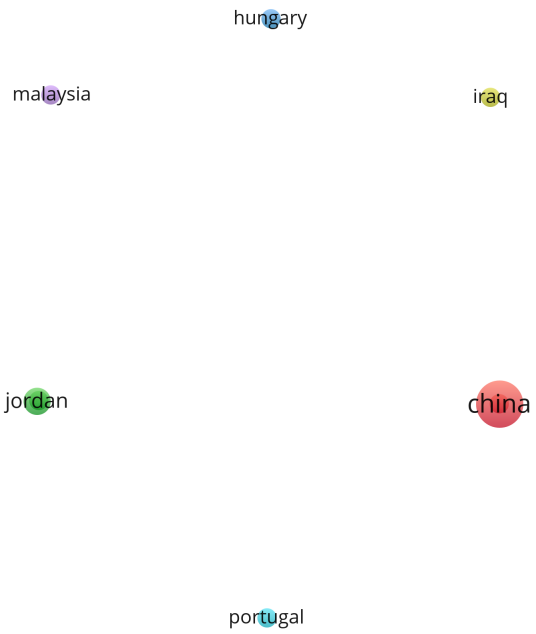


Figura 3.3: Países (Autor, 2023)

3.4.3 Q3 - Quais são as principais revistas e conferências que publicam sobre o assunto?

É mostrado na Tabela 3.5, todas as revistas e conferências em ordem decrescente, com base no número de publicações na área. Vale destacar a "IEEE Access", com quatro publicações na área, sendo a maior revista que nesta análise publicou sobre o assunto. Ademais, todas as outras revistas e *journals* só possuem um artigo, cada, publicado. O único ponto de semelhança, por localidade, que é possível extrair, são duas revistas da Suíça, a *Information* e a *Eletronics*.

Conferências | Publicações	Quantidade
IEEE Access	4
Soft Computing	1
Info-communications Journal	1
International Journal of Advanced Computer Science and Applications	1
Indonesian Journal of Electrical Engineering and Computer Science	1
Eurasip Journal on Embedded Systems	1
Electronics (Switzerland)	1
International Journal of Cognitive Informatics and Natural Intelligence	1
Information (Switzerland)	1

Tabela 3.5: Conferências e Publicações Sobre o Escopo da Pesquisa

3.4.4 Q4 - Quais são os principais temas de pesquisa na área?

Na figura 3.4, foi feita a extração de todas as palavras-chave indexadas, tanto pelo autores, como pela Scopus/Web of Science, a fim de investigar quais são os principais temas e assuntos tratados pelos *papers* selecionados. Na construção dos principais temas abordados, foi realizada a leitura de cada artigo a fim de conhecer quais foram as principais técnicas e assuntos abordados nos trabalhos.

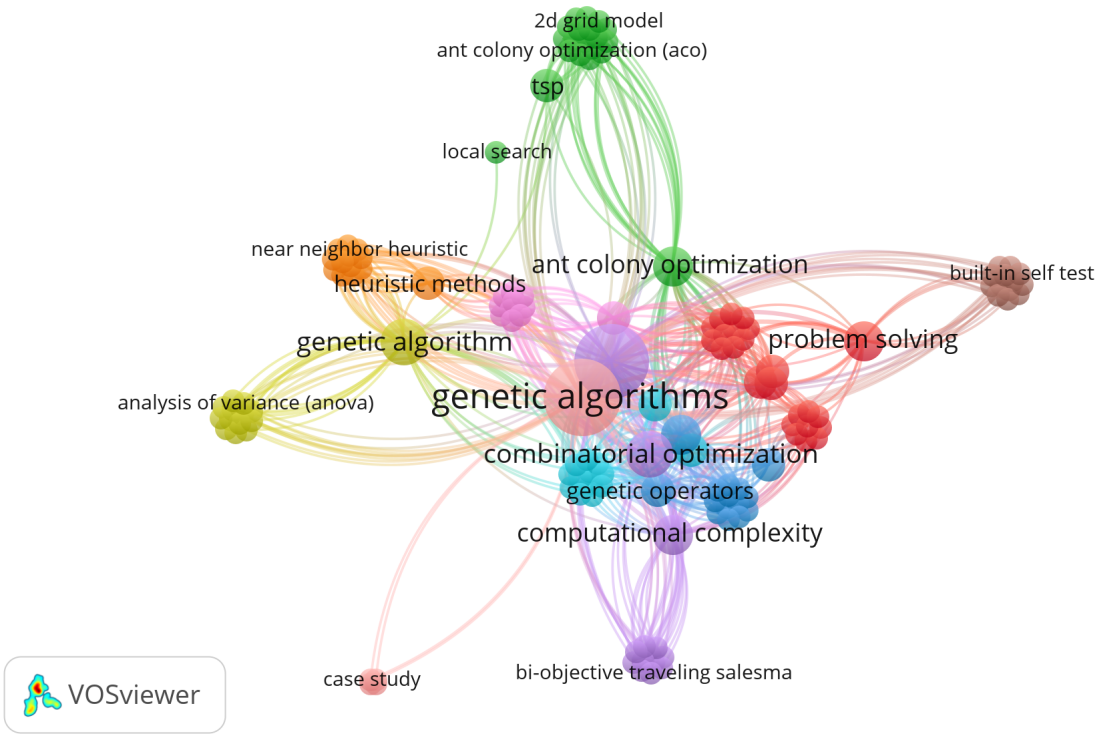


Figura 3.4: Palavras chaves dos Artigos (Autor, 2023)

Como pode-se ver, as principais palavras encontradadas nos artigos foram: *genetivcs algorithms*, *tsp*, *combinational optimization* e *ant colony optimization*. Já relacionado aos principais temas tratados nos artigos, pode-se elencar: *Ant Colony Optimization* – ACO, *Gray Wolf Optimization* – GWO, e o *Particle Swarm Optimization* – PSO como principais atores dentro dos trabalhos selecionados. Além disso, também vale mencionar, que em sua maioria, os artigos estão procurando a otimização no caminho do problema do caixeiro viajante e, também, grande parte dos trabalhos buscam solucionar problemas na área de logística, seja de distribuição ou de recursos humanos, utilizando-se de Algoritmos genéticos. Abaixo, será discutido um pouco de cada técnica utilizada, mencionando os principais pontos abordados pelo *papers*.

Ant Colony Optimization

O *Ant Colony Optimization* (ACO) é uma técnica de otimização inspirada no comportamento das formigas em busca de alimentos (IBADA ALI; TÜŰ-SZABÓ; KÓCZY, 2022). Ela utiliza feromônios para marcar caminhos e compartilhar informações entre formigas. As formigas constroem soluções iterativamente, preferindo caminhos com maior concentração de feromônios. O algoritmo é eficiente para problemas complexos, sendo aplicado em otimização de redes, roteamento e agendamento. Obteve-se uma recorrência de 5 trabalhos que abordam ou mencionam essa técnica, o que corrobora com a ideia de que a otimização do caminho é um ponto crucial, como refenciado no trabalho *A novel path planning algorithm for warehouse robots based on a two-dimensional grid model* (YANG, B. et al., 2020), nele, tem-se a construção de um caminho ponderando a inicialização da população antes mesmo de encontrar o melhor percurso.

Gray Wolf Optimization

O *Gray Wolf Optimization* (GWO) é uma técnica de otimização inspirada no comportamento social de lobos cinzentos, que, por sua vez, representam soluções e seguem ações baseadas na hierarquia social da matilha (ALI, Z. A.; RASHEED; ALI, N. N., 2020). O algoritmo explora o espaço de busca e converge para soluções melhores ao longo do tempo. É eficaz para resolver problemas complexos em diversas áreas, como otimização de funções, engenharia e aprendizado de máquina. Sua inspiração no comportamento dos lobos torna-o uma técnica promissora para resolver problemas desafiadores. Da mesma forma que a ACO, essa técnica foi bastante explorada para encontrar uma solução direta para otimizar o caminho proposto no problema

do Caixeiro Viajante, como pode ser visto no trabalho *Applying a genetic algorithm to a m-tsp: Case study of a decision support system for optimizing a beverage logistics vehicles routing problem* (GOMES et al., 2021) e, também, encontrado no trabalho do IBADA Ali (IBADA ALI; TÜŰ-SZABÓ; KÓCZY, 2022), para otimizar a montagem inicial da população.

Particle Swarm Optimization

Particle Swarm Optimization (PSO) é uma técnica de otimização inspirada no comportamento social de bandos de pássaros ou cardumes de peixes. As soluções são representadas por partículas que se movem em busca das melhores posições, com base em sua experiência pessoal e nas informações compartilhadas com outras partículas (WU et al., 2021). O PSO é amplamente utilizado em problemas de otimização contínua, e não linear, em várias áreas, devido à sua eficácia e simplicidade de implementação. Esta foi uma das técnincas mais presentes nos trabalhos selecionados, pois quando aplicada ao TSP temos uma melhora consideravel, como pode ser observado no trabalho *A Novel Particle Swarm Optimization With Genetic Operator and Its Application to TSP* (WEI et al., 2021). Nele, a abordagem do PSO melhora a precisão da convergência do algoritmo e a diversidade da população no problema proposto.

3.4.5 Q5 - Quais são os trabalhos mais citados?

Na Tabela 3.6, mostra os cinco principais artigos da área, o número de citações, e os cita de forma que possa ser consultado nas referências desta monografia, para o estudo de quem interessar.

Artigo	N. Citações
(HASSANAT et al., 2018)	38
(XU et al., 2018)	33
(CHEN et al., 2019)	27
(WU et al., 2021)	23
(EMAMBOCUS et al., 2021)	10

Tabela 3.6: Artigo e Citações

3.5 Considerações Finais

A revisão sistemática dos estudos realizados entre 2017 e 2022 permitiu uma análise abrangente do estado atual de pesquisa no que diz respeito à aplicação de técnicas de otimização para a inicialização do Problema do Caixeiro Viajante. Inicialmente, por meio da análise bibliométrica, consegue-se identificar os principais periódicos responsáveis pela maioria das publicações, quantificar o número de artigos publicados e identificar os autores mais proeminentes nesse campo. Além disso, foi categorizado os artigos por macroárea, o que fornece *insights* valiosos para orientar futuras investigações nessa área de estudo, categorizando as principais universidades, revistas, países e relevância acadêmica presente no item 3.4 deste trabalho. Essas descobertas contribuem para o direcionamento de pesquisas posteriores e fornecem um panorama do estado atual da pesquisa sobre o assunto.

No contexto deste artigo, é identificado as principais áreas de foco que englobam o uso de algoritmos para aprimorar a inicialização. Isso inclui técnicas, como: o Enxame de Partículas (WEI et al., 2021), que busca aprimorar iterativamente uma solução, seguindo uma medida de qualidade específica. Além disso, abrange a otimização de algoritmos através do ajuste de hiperparâmetros e da seleção de atributos para otimização. O objetivo aqui é melhorar o desempenho do TSP de acordo com os modelos de otimização disponíveis.

A análise e categorização dos artigos revelam que os estudos mais recentes têm colocado um forte foco na combinação de modelos e técnicas para otimizar Algoritmos Genéticos aplicados ao Problema do Caixeiro Viajante (TSP). Isso ocorre devido à complexidade inerente a problemas desse tipo, que frequentemente envolvem dados não lineares, especialmente no contexto da logística. Apesar dos desafios a serem enfrentados, a sistematização das melhorias obtidas pela combinação de algoritmos contribui significativamente para aumentar a precisão na exploração de todas as soluções possíveis na população. Esse avanço é particularmente relevante no contexto em que os autores direcionam sua atenção para a logística, já que ajuda a encontrar rotas mais eficientes para distribuição, além de reduzir custos.

Capítulo 4

Metodologia

Esta seção é dividida, primeiramente, com a descrição do ambiente de teste e as bibliotécas e linguagem utilizada para realização do algoritmo proposto e em seguida em quatro subseções que são:

(1) Descrição do Problema.

(2) Descrição das Técnicas Utilizadas.

(3) Processamento dos Dados.

(4) Algoritmo Utilizado.

(5) Métricas de Avaliação.

4.1 Ambiente de Teste

O ambiente de teste utilizado para fazer as comparações foi o Google Colab (*Google Computer Engine*), [1] que possui as seguintes configurações:

- RAM: aproximadamente 13 GB;
- HD: aproximadamente 108 GB;

Foi utilizado o *Python* 3, como também foram utilizadas algumas bibliotecas, tais como:

[1] `https://colab.research.google.com`

- Pandas: O pandas é uma biblioteca Python 3 usada para manipular e analisar dados tabulares, oferecendo estruturas de dados como DataFrames e Series. Ele permite carregar, limpar, transformar e analisar dados de forma eficiente, com recursos como manipulação de dados, leitura/escrita de dados, tratamento de dados ausentes e visualização. É amplamente utilizado em análise de dados e ciência de dados (NUMFOCUS, s.d.).

- Numpy: O NumPy é uma biblioteca essencial do Python 3 para computação científica, oferecendo suporte para arrays multidimensionais eficientes e funções matemáticas de alto desempenho. Ele permite realizar cálculos numéricos, manipular dados multidimensionais e é amplamente utilizado em ciência de dados, análise numérica e processamento de dados (NUMPY, s.d.).

- Statistics: A biblioteca ‘statistics‘ do Python 3 é um módulo padrão que oferece funções para realizar cálculos estatísticos básicos em dados univariados. Isso inclui cálculos como média, mediana, moda, variância, desvio padrão e outras estatísticas simples. É útil para tarefas estatísticas simples (PYTHON, s.d.).

- Itertools: A biblioteca ‘itertools‘ do Python 3 é um módulo padrão que oferece ferramentas para criação de iteradores e manipulação eficiente de sequências de dados. Ela inclui funções para criar iteradores infinitos, gerar combinações e permutações, agrupar elementos, filtrar dados e criar iteradores personalizados, tornando mais fácil e eficiente lidar com tarefas de iteração e manipulação de sequências em Python. É especialmente útil para problemas que envolvem combinações, permutações e agrupamento de elementos em coleções de dados (PYTHON, s.d.).

- Secrets: A biblioteca ‘secrets‘ do Python 3 é usada para gerar números aleatórios seguros em situações de segurança, como geração de senhas e chaves criptográficas. Ela fornece funções para gerar bytes aleatórios, strings hexadecimais e inteiros aleatórios, garantindo maior segurança em comparação com a biblioteca ‘random‘ padrão (PYTHON, s.d.).

- Random: A biblioteca ‘random‘ no Python 3 oferece funções para geração de números pseudoaleatórios em programas Python. Ela pode gerar números aleatórios, inteiros, selecionar aleatoriamente elementos de sequências e embaralhar sequências (PYTHON, s.d.).

- Time: A biblioteca ‘time‘ no Python 3 é usada para lidar com operações relacionadas ao tempo, incluindo medições de tempo, pausas no programa e manipulação de datas e horas. Ela é útil para controlar o fluxo do programa com base no tempo e para formatar datas e horas. Pode ser usada para medir o tempo decorrido, criar temporizadores, entre outras tarefas relacionadas ao tempo (PYTHON, s.d.).

4.2 Descrição do problema

O Problema do Caixeiro Viajante (TSP) é um desafio de otimização combinatória que busca encontrar o caminho mais curto que visite todas as cidades exatamente uma vez, e, retorne à cidade inicial. É conhecido por ser NP-difícil, sendo comum o uso de algoritmos aproximados para encontrar soluções eficientes em instâncias grandes, enquanto que instâncias menores podem ser resolvidas por métodos exatos em um tempo razoável. Tendo em vista essa problemática, no problema proposto nessa monografia foi pensado em criar uma implementação do TSP visando a inicialização individual de cada individuo com o intuito de melhorar a performance do algoritmo. O contexto do problema está baseado em como é possível melhorar o sistema de logística, seja de uma empresa com distribuição local ou até mesmo uma distribuição do estado para insumos médicos ou itens de necessidade urgente que precise ser entregue com a melhor rota possível. Abaixo, é apresentada uma ilustração base que mostra a premissa do problema do Caixeiro Viajante.

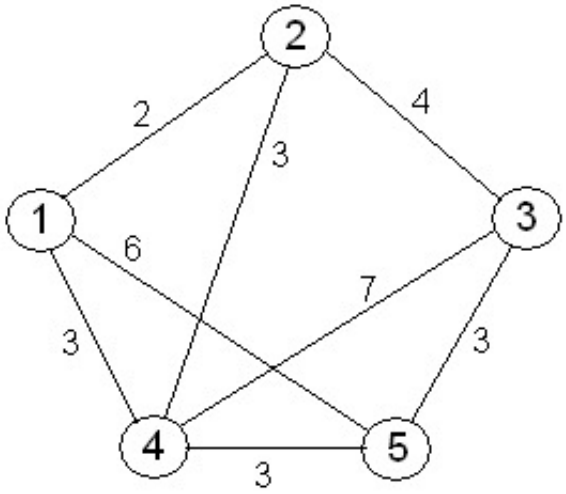


Figura 4.1: Representação do Problema do Caixeiro Viajante – Adaptado (ANDERS WIRÉN, s.d.)

Para isso, foi feito uma implementação do caixeiro viajante utilizando dados do Departamento de Estradas e Rodovias – DER-AL [2] onde foi feita a coleta das latitudes e longitudes de cada munícipio de alagoas e usamos métricas cartesianas para calcular essas distâncias de forma individual, e, utilizando suas coordenadas decimais para que cada cidade vire um ponto, para que de fato se possa usá-las no AG proposto para o TSP. Cada aresta neste problema é representado pela distância euclidiana juntamente com uma relação do estoque e o custo para ir de determinado ponto a outro. Foi pensado na seguinte relação: uma cidade perto, porém, com um estoque mais cheio pode significar uma logística falha do ponto de vista da proposta. Então, em um cenário onde o algoritmo tem que decidir entre duas cidades com estoques distintos, é feita uma avaliação referente a relação de estoque/distância para que a decisão seja feita em conformidade com o melhor, no quesito de priorizar o que tem estoque baixo, e a menor distância olhando sempre para o melhor cenário dentro dos 101 munícipios alagoanos, e, retornando para a capital do estado, Maceió. Para melhor visualização, temos a Figura 4.2 que ilustra a distribuição das cidades.

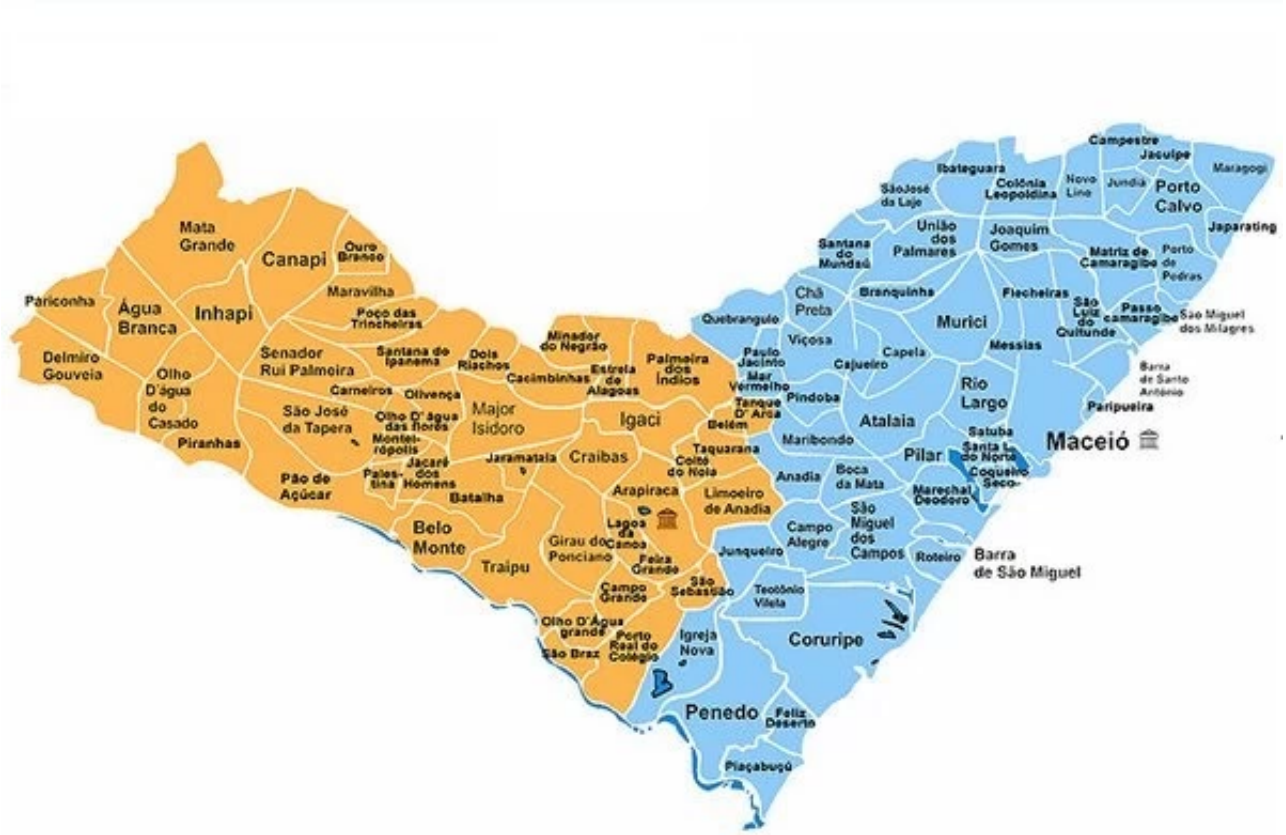


Figura 4.2: Mapa geopolítico de Alagoas – Fonte: IBGE

[2]Fonte:http://www.der.al.gov.br/distancia-entre-cidades

4.3 Descrição das técnicas utilizadas

Para apresentar uma solução do problema do Caixeiro Viajante utilizamos um Algoritmo Genético simples utilizando os municipios de alagoas como arestas. Esse algoritmo genético tem por objetivo priorizar a individualização de cada individuo no processo da inicialização. Ademais, o algoritmo tem técnicas de extração dos dados do tipo *comma-separated-values* – csv utilizando a biblioteca Pandas, onde foi criado um *data-frame* com todas informações de cada município.

Logo em seguida, foi definido o cromossomo simples adapatado ao problema e feita uma seleção elitista, que resumidamente, seleciona os melhores individuos para as próximas gerações sempre procurando a melhor solução. O operador genético de Mutação usado no algoritmo proposto foi o de permutação com uma taxa de 75%, onde em cada iteração o operador verifica se houve alguma mutação, tendo uma mutação constatada é feita a permutação em cima do individuo mutado. A operação de *crossover* foi feita de maneira uniforme, ou seja, não foi feita a utilização de pontos de cruzamento e sim com um parâmetro global que determina qual a probabilidade da variável ser trocada entre seus pais, no algoritmo proposto, está variavel é denominada CROSSOVER-RATE que tem, como valor igual a 90, para ter uma taxa de cruzamento bem alta entre os individuos.

E por último, temos o *score* obtido na geração do AG proposto, que é onde são implementadas as métricas referentes a custos, estoque e processa as distâncias euclidianas de cada municipio em forma de arestas, representando a distância entre os pontos (municipios).

4.4 Processamento dos Dados

Para o processamento dos dados foi utilizado a biblioteca Pandas, extraindo as latitudes e longitudes dos dados cedidos pelo DER de Alagoas. Com estes dados, convertemos para uma métrica euclidiana das distâncias dos entre munícipios na unidade de medida de coordenadas decimais. Feita essa conversão tivemos a distância dos pontos sem levar em consideração a distância oficial que é dada pelo próprio DER em quilomêtros a partir da capital Maceió. Não foi utilizada esta métrica pois ela só leva em consideração como ponto de partida a capital do estado e isso não se aplicaria ao problema proposto porque temos diferentes pontos de inicialização para a problemática do TSP, em que a abordagem tomada, esse ponto de inicialização pode ser qualquer cidade que apresente o melhor percurso. Utilizando somente os dados

centralizados na capital perderíamos diferentes referenciais de partidas que com a proposta de multiplas inicializações seria limitante ao algoritmo genético aplicado ao problema do Caixeiro viajante. Foi utilizada a seguinte equação para fazer essa conversão:

$$DD = d + \frac{min}{60} + \frac{sec}{3600} \tag{4.1}$$

Como podemos observar na equação 4.1, a conversão é feita seguindo as proporções de minutos e segundos, onde um minuto equivale a 60 segundos e cada grau equivale a uma hora, que, por sua vez, são 60 minutos, ou como por conveniência e para padronização das unidades de medida, definimos como 3600 segundos.

Feito os devidos tratamentos das distâncias, os dados são armazenados em um *dataframe*, os nomes de cada munícipio é extraído e jogado em uma lista para ser usada como os indíces. Os demais itens armazenados no *Data-Frame* – DF é processado posteriormente no *crossover* do AG. Além disso, para este algoritmo foram usados números aleatórios para definir o estoque de cada cidade utilizando uma biblioteca *Random* do *Python*. O objetivo de randomizar os estoque é ter diferentes resultados que possam testar o algoritmo, sempre colocando em teste sua eficiência. O custo, por outro lado, foi baseado no preço médio do combustível no ano virgente (2023), [3] a fim de que isto seja padronizado para toda execução do AG.

4.5 O algoritmo proposto

O algoritmo proposto por esta monografia tem por objetivo melhorar a perfomance do problema do Caixeiro Viajante utilizando Algoritmo Genético, com foco em abordar a problemática da inicialização da população. No caso, esta implementação utilizou os referentes ao Estado de Alagoas. Este é o principal objetivo, já o contexto aplicado é a problemática de fazer uma melhor distribuição de recursos a partir de um ponto fixo. Neste algoritmo, utilizamos um caso hipotético de uma distribuição de insumos da saúde em um caso pândemico como exemplo, visando encontrar o melhor caminho para esta distribuição utilizando métricas como o estoque disponível na cidade e o custo para chegar em determinado local.

Foram utilizados para o algoritmo alguns fatores como:

- uma taxa de mutação de 75% para garantir uma boa diversidade da população;

[3] Preço médio da Gasolina no momento da escrita: R$ 5,61 no estado de Alagoas

- um cruzamento de um corte na população como mencionado na seção de técnicas utilizadas com um crossover de 90;
- seleção elitista da população para que os melhores indivíduos sejam priorizados

Como o foco deste trabalho foi propor uma nova abordagem de inicialização, as características do Algoritmo Genético criado foram feitas para priorizar sempre uma alta diversidade da população, atrelado com uma taxa de cruzamento alta sempre escolhendo os melhores indíviduos gerados por sua execução.

4.6 Métricas de avaliação

As métricas escolhidas neste trabalho para a realização da comparação foram: a quantidade de gerações, o tempo de execução necessário para o algoritmo genético encontrar a solução do problema,e, o score do cromossomo mais adaptado por intervalo de geração.

Capítulo 5

Resultados e Discussão

Neste capítulo, será descrito os resultados obtidos com o algoritmo proposto e, também, irão ser feitas comparações com outra implementação comum do problema do Caixeiro viajante, utilizando Algoritmo Genético com a mesma base de dados e métricas de um AG, sendo a mesma taxa de mutação, o mesmo cruzamento de um ponto de corte, somente diferenciando do algoritmo proposto, sua inicialização individual. Por último, será apresentado um teste de hipótese a fim de validar o algoritmo e mostrar que de fato há relevância atráves do teste de T student (LIVINGSTON, 2004). O repositório do projeto se encontra no Google Colab[1] para visualização.

5.1 Comparação entre algoritmos

Nesta seção do trabalho, será comparado dois Algoritmos Genéticos: o proposto e uma implementação comum para o TSP. Foi utilizado o *software* Jamovi como ferramenta para analisar todos os resultados obtidos nesta seção (TEAM, s.d.), gerar alguns gráficos e dados estátisticos que serão apresentados a seguir. Além do Jamovi, também foi utilizado python para a criação de alguns gráficos, também vistos neste capítulo. Como metodologia para obtenção dos resultados de cada algoritmo apresentado, foi executado ambos os algoritmos 30 vezes para, assim, poder ter um nível de significância relevante e uma confiabilidade consistente no que foi obtido. Na tabela a seguir, é observado as estátisticas descritivas de cada algoritmo executado, utilizando como parâmetro o *score* obtido:

[1] Algoritmo proposto: `https://drive.google.com/file/d/172Bj93Hu9qZPSGw4odn1qvY_rpimV-Ps/view?usp=sharing`

Algoritmo	Nº de Gerações	Média	Mediana	DP	Erro-padrão
Algoritmo Normal	30	6073	6131	297	54.2
Algoritmo Proposto	30	5882	5955	310	56.6

Tabela 5.1: Estatísticas Descritivas

Como é possível observar na tabela 5.1, o algoritmo proposto performa melhor em relação a um algoritmo genético comum implementado. Fatores básicos, como a média e a mediana, o algoritmo apresentado mostra uma pequena melhora em relação a um AG comum, olhando exclusivamente o *score* de cada execução, ressaltando que a construção do *score* se dar pela soma das arestas percorrido durante o caminho feito pelo Caixeiro, logo, a melhor pontuação, neste contexto, se dar por um *score* menor e por isso que é constatado um melhor resultado para o algoritmo apresentado. Não obstante a isto, o Desvio Padrão – DP do algoritmo proposto é maior que o algoritmo normal utilizado, ou seja, a dispersão dos *scores* obtidos no algoritmo normal são melhores que o do algoritmo proposto, indicando que o conjunto do mesmo é menos homogênio e irregular que uma implementação comum. Além disso, o erro-padrão, ou seja, a precisão da média do *score* em relação ao conjunto de dados, também apresenta uma melhor precisão por parte do AG Comum, o que faz sentido, levando em consideração o Desvio Padrão obtido em relação a ambos. Para corroborar com as constatações acima citadas, é apresentado na Figura 5.1 um BoxPlot representativo destes dados a fim de exemplificar o que foi descrito:

No BoxPlot apresentado, vê-se que o algoritmo normal tem uma aplitude interquartílica menor em seus quartis e seu valor máximo e mínimo são menores, ou seja, menos dispersos que o algoritmo proposto, apresentando uma normalidade maior dos dados, (observado no paragráfo acima), em relação ao seu erro e desvio padrão. Além disso é possível observar que a mediana apresentada pela linha de corte na caixa de cada um são bem semelhantes nas duas amostragens, o que indica que seus valores de média em cada quartil são distribuídos de forma parecida, ainda que suas amplitudes sejam diferentes. Abaixo, será apresentado o gráfico de linha de cada algoritmo em relação ao número de gerações, mostrando como cada um performou na relação geração/score. Com ele, será fácil de ver como o algoritmo proposto teve um desempenho mais coeso e com valores mais constantes e satisfatórios em relação ao *score* comparado ao Algoritmo Genético normal.

Na figura 5.2, é visualizado uma ilustração de um gráfico de linhas, com a performance de ambos os algoritmos usados, para efeito de comparação. É possível ver tudo que já foi abordado

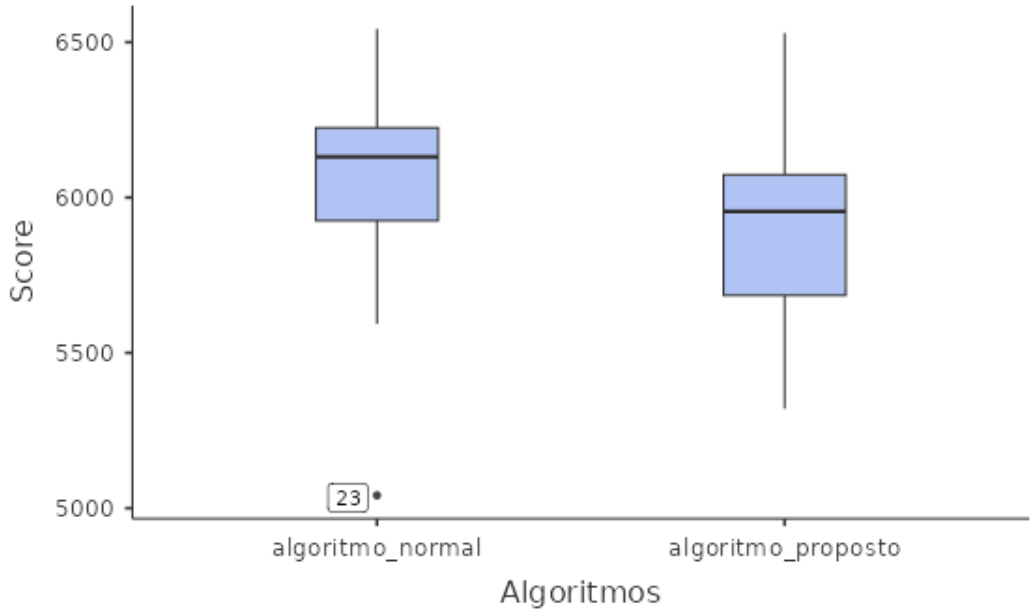


Figura 5.1: Boxplot ilustrativo dos Resultados – Fonte: Autor (2023)

em relação ao melhor desempenho do *score* do algoritmo proposto em relação ao algoritmo genético normal para o problema do Caixeiro viajante, bem como identificar a inferioridade em relação ao desempenho de seu *score*, ou seja, o AG convencional sempre se mantem com sua execução apresentando valor maior em relação ao proposto onde, é possível observar que apenas um pequeno intervalo de gerações ele ultrapassa a superioridade do algoritmo proposto. Contudo, também é possível observar que nas gerações finais conseguimos ver que a abordagem convencional do algoritmo genético estaciona e mantem sua linearidade até o fim da execução, enquanto que o algoritmo com a nova proposta de inicialização continua apresentando melhores resultados a partir da décima geração e só estaciona em um único valor de score ao final das gerações necessárias para encontrar o melhor caminho. Vale ressaltar, entretanto, que como pode-se observar no gráfico, em nenhum momento o pior *score* do algoritmo proposto foi inferior ao mais alto do que a implementação comum. Outra informação que conseguimos extrair do gráfico, é que a implementação do AG normal já apresenta o primeiro valor de score relativamente bem mais alto que a implementação proposta. Além disso, apenas o intervalo entra a oitava e nona geração o algoritmo convencional nos apresenta um resultado melhor contudo, já na nona geração o algoritmo proposto volta a apresentar resultados melhores e se mantem até o final com um score com 100 pontos de diferença da melhor execução do algoritmo convencional, melhorando consideravelmente a partir da décima segunda geração e se

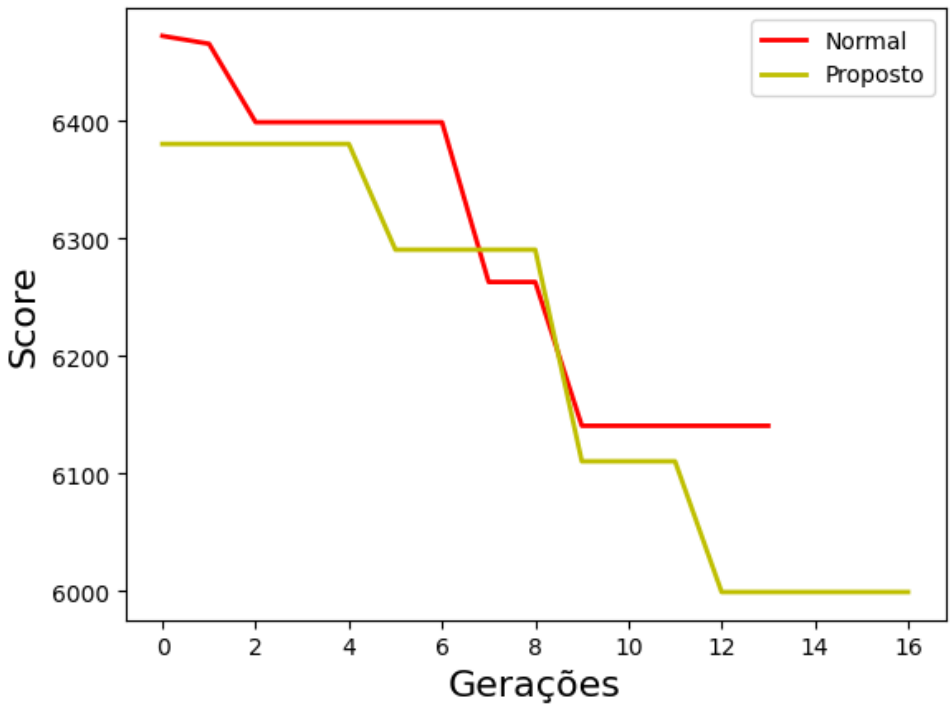


Figura 5.2: Gráfico de linhas com a relação Score/Geração – Fonte: Autor (2023)

mantendo com um bom resultado até a última geração. Portanto, conseguimos observar que o algoritmo apresentado é mais regular, pois demonstra resultados menos lineares e com menores valores de *score* (visto que, do ponto de vista de avaliação, como mencionado, é melhor), em relação a seu concorrente.

Além do gráfico de linhas mostrando como ambos os algoritmos se saíram em relação ao seu *score*, também será apresentado, de forma a complementar e dar mais lucidez em relação aos resultados, o tempo da execução de cada algoritmo nas trinta vezes em que foram executados e, também, o número de gerações em que cada um encontrou o melhor caminho na resolução do TSP. Primeiro, será mostrado um gráfico de linhas do tempo de cada execução e, logo em seguida, um gráfico de linha das gerações. Foi escolhido, em ambos, o gráfico de linhas, por melhor ilustrar o desempenho em relação a cada execução.

Como pode-se ver na Figura 5.3, o algoritmo proposto apresenta um tempo de execução (medido em segundos) mais uniforme, em relação ao algoritmo normal, contudo, também apresenta o maior pico de execução de 4400 segundos (cerca de 73 minutos e 18 segundos), enquanto que o algoritmo normal executou em seu maior tempo de 3900 segundos (precisamente 65 minutos). Dessa forma, entende-se que o algoritmo apresentado não demonstra um bom desempenho no quesito tempo, não obstante a isso, a diferença entre o pior tempo de ambos não é algo muito discrepante, levando em consideração que o algoritmo deste trabalho

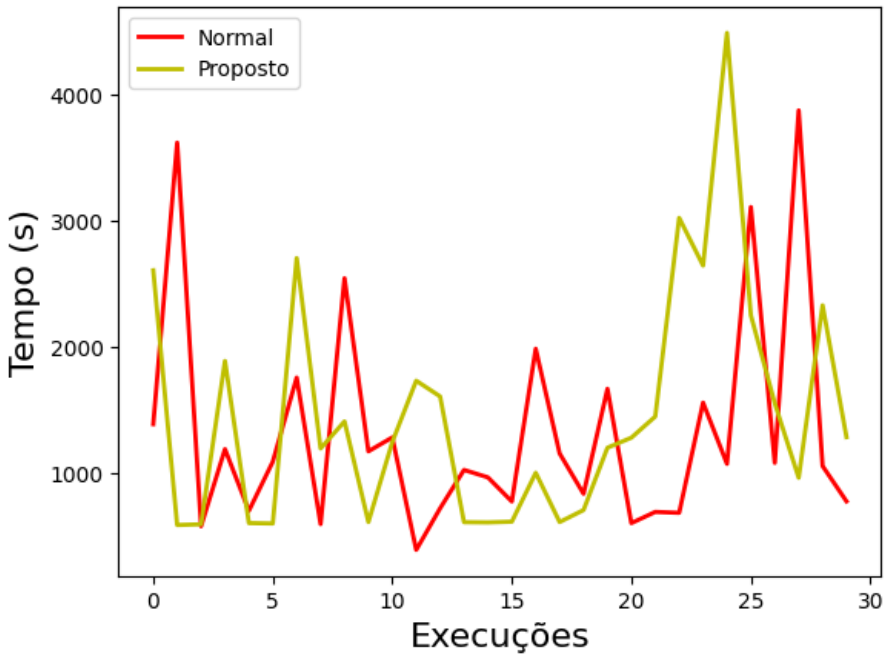


Figura 5.3: Gráfico de linhas com a relação Tempo/Execução – Fonte: Autor (2023)

tem, como já mencionado, um equílibrio maior em relação a tempo. Uma explicação plausível para isso pode se dar pela necessidade de que o algoritmo mostrado precisa de um pouco mais de processamento para inicializar de forma individual cada ponto, para, assim, encontrar a melhor resolução do TSP, fazendo com que ele apresente percas no quesito tempo.

No que se refere as gerações necessárias para busca do melhor caminho, a Figura 5.4 mostra que o algoritmo proposto apresenta melhores resultados em relação a seu algoritmo concorrente. Dessa maneira, observa-se no gráfico que a quantidade de execuções em que o algoritmo encontrou na primeira geração, ou seja, no valor 0, é bem maior que o algoritmo normal. No mais, ambos se comportam de maneira bem irregular, contudo o algoritmo normal apresentou o maior pico de geração mais alta para encontrar o caminho; sendo necessárias 30 gerações para encontrar o caminho do percurso. Outro ponto interessante a se notar, observando o gráfico, é que, a partir da décima execução, ambos os algoritmos demoram a mesma quantidade de gerações até o final, com uma leve diferenciação positiva no algoritmo proposto com uma geração a menos na última vez executado.

5.2 Teste de Hipótese

Nesta seção, será abordado os testes de hipóteses realizados para validade do estudo, apresentando as hipóteses utilizadas no teste T student, descrevendo seus respectivos H0 e H1, ex-

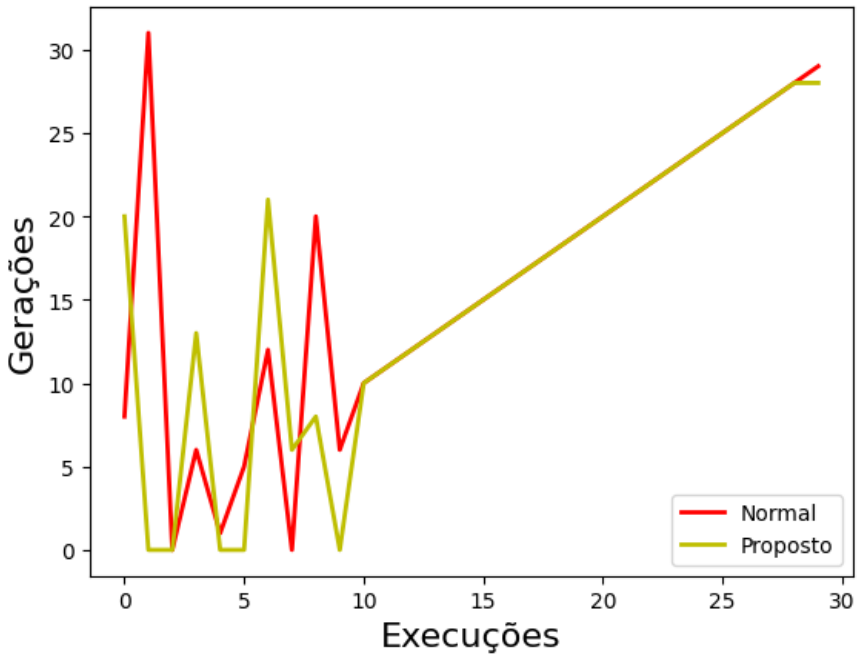


Figura 5.4: Gráfico de linhas com a relação Geração/Execução – Fonte: Autor (2023)

plicando seus resultados e apresentando os argumentos que o validam para esta monografia. Além disso, também será apresentado a normalidade dos dados através do teste de Shapiro-Wilk (GHASEMI; ZAHEDIASL, 2012), para verificar a normalidade dos dados apresentados. Abaixo, foi feito uma tabela que ilustra o resultado da normalidade aplicada pelo teste realizado:

	W de Shapiro-wilk	p de Shapiro-Wilk
Algoritmo Normal	0.901	0.009
Algoritmo Proposto	0.957	0.262

Tabela 5.2: Teste de normalidade de Shapiro-Wilk

Como pode-se observar na Tabela 5.2, o teste realizado de Shapiro-wilk mostra que ambos os resultados possuem valores semelhantes no que se refere ao nível de significância associado (W), contudo o valor p do teste nos mostra que o algoritmo comum não apresenta uma distribuição normal dos dados, ou seja, ela não chega ser simétrica em seus extremos de média. Isso se dar por conta de seu p ser maior que 0,05, mostrando, assim, que os dados gerados por este algoritmo não apresentam uma normalidade gaussiana em sua distribuição. Ao encontro disto, o valor p do algoritmo proposto apresenta um valor maior que o corte para definir se a distribuição é normal ou não. Nesse caso, o p do algoritmo apresentado é maior, portanto a distribuição desses dados pelo algoritmo é normal, o que já mostra um pequeno indício que,

nos testes de hipóteses, estes dados apresentaram uma diferença estátistica em um primeiro momento, pois, de imediato, ambos apresentam distribuições diferentes dos dados. É importante ressaltar que o teste de T Student permite realizar comparações de amostras independentes com distribuições análogas desde que a amostra tenha um tamanho maior ou igual a trinta (GRAVETTER et al., 2020) .Abaixo, temos o primeiro teste de hipótese de T student para amostras independentes:

	Estatística	gl	p
t de Student	2.43	58.0	0.018

Tabela 5.3: T student para amostras independentes ($H0 \neq H1$)

Primeiro, é importante entender quais são os argumentos utilizados para realizar o teste de hipóteses, definindo, assim, quais são os H0 e H1 que foram utilizados para o resultado apresentado na Tabela 5.3. As hipóteses levantadas foram as seguintes:

- H0: os algoritmos não apresentam diferença entre si
- H1: os algoritmos apresentam uma diferença entre si

Nessa análise bi-caudal, onde iremos verificar se o teste apresenta diferença entre os grupos com as hipóteses acima citadas, vê-se que o p resultante da análise se torna verdadeiro, apresentando um valor significativo de 0.018. Lembrando que, para existir uma diferença, é necessário que a seguinte relação seja respeitada, utilizando um intervalor de confiança de de 95%, em que: p < 0,05. No caso deste trabalho, isso foi comprovado estatísticamente com o p passando no teste T Student apresentado, ou seja, os algoritmos apresentam uma diferença entre si em seus respectivos resultados obtidos de *score*. Outras informações importantes que poder extrair destes resultados é que o Grau de Liberdade – GL utilizado de 58.0, apresentando uma distribuição normal padrão satisfatória em relação as duas amostragem usadas. Já o valor estátistico que diferencia o desvio padrão da população, também mostra algo interessante em relação as populações: por mais que no teste de Shapiro-wilk eles tenham apresentado diferença, a diferença real é pequena.

O próximo teste de T student, realizado para amostras independentes, busca verificar se o *score* do algoritmo normal apresenta uma superioridade em relação ao do algoritmo proposto. Lembrando que o *score* em um AG sempre tende a ser melhor quando apresenta valores baixos, ou seja, se o teste aplicado apresentar uma veracidade, significa dizer que o algoritmo proposto

teve uma melhor performance de pontuação. Abaixo, temos as hipóteses utilizadas para este teste:

- H0: o algoritmo normal tem um Score superior ao algoritmo proposto
- H1: o algoritmo normal tem um Score inferior ao algoritmo proposto

	Estatística	gl	p
t de Student	2.43	58.0	0.009

Tabela 5.4: T student para amostras independentes (H0 > H1)

Como visto na Tabela 5.4, e olhando o que foi dito no paragráfo anterior, o *score* de uma implementação básica do TSP, apresenta um resultado bem acima do algoritmo proposto, o que mostra que sim, o algoritmo proposto tem uma taxa de *score* melhor que um AG simples para o problema citado. Os valores de grau de liberdade e a diferença média dos desvios padrões continuam iguais, contudo o valor p de significância do teste, que é o valor que realmente prova que um resultado é superior ao outro, é muito inferior ao corte de $p < 0{,}05$ como é conhecido no teste de T student, sendo o p obtido de 0,009.

Capítulo 6

Conclusão e Trabalhos Futuros

Neste capítulo, será apresentado as conclusões acerca dos resultados obtidos, todos os passos percorridos até então, quais conclusões e contribuições, foram obtidas por meio desta monografia. Também será abordado o que pode ser feito em relação aos trabalhos futuros com o objetivo de mostrar quais são as principais melhorias, caminhos que podem ser tomados a partir do algoritmo e proposta apresentada.

6.1 Conclusão

O trabalho propôs a implementação do Algoritmo Genético aplicado ao problema do Caixeiro Viajante com foco logístico no estado de Alagoas, apresentando uma nova forma de inicialização. O espaço amostral de soluções foi dividido em quadrantes, com o objetivo de ter um representante de cada quadrante na população inicial. Na construção do conjunto de dados (*dataset*), foram selecionados parâmetros, como a distância em estoque disponível da cidade e o preço médio de combustível a fim de gerar um grafo completo. Isso significa que todos os vértices estão interligados e o cálculo entre cidades leva em conta os parâmetros mencionados anteriormente.

Os resultados indicam que o Algoritmo Genético proposto apresenta melhor desempenho em comparação com o AG convencional. Esse desempenho superior é evidenciado não apenas pela quantidade de gerações, mas, também, pelo *score* obtido em cada execução do algoritmo, bem como pelos resultados do teste de hipótese que confirmam essa afirmação. Entretanto, é importante observar que os gráficos do Capítulo 5 revelam que o Algoritmo Genético proposto

requer um maior tempo de execução em comparação com o AG convencional, o que o torna menos eficiente em termos de tempo de convergência.

6.2 Trabalhos Futuros

Embora o objetivo deste estudo seja de natureza científica, a abordagem apresentada tem o potencial de servir como um protótipo para futuros desenvolvimentos de uma ferramenta prática de otimização de custos logísticos. Esta ferramenta futura poderia permitir a integração de locais reais e oferecer rotas otimizadas. A ideia é criar uma Interface de Programação de Aplicativos (API) que faça uso gratuito de mapas e serviços de análise de tráfego. Isso seria valioso para auxiliar pequenos empresários a reduzir seus custos operacionais, gerando economia que poderia ser investida em seus negócios.

Em complemento a isto, também pode ser feito estudos e implementações que otimizem o algoritmo atual com algumas das técnicas citadas na seção anterior, e ainda há, também, a possibilidade de propor um novo tipo de inicialização por quadrante que utilize um Algortimo Genético para cada indivíduo do sub-espaço amostral retornando o melhor deste encontrado e tornando a solução ainda mais satisfatória.

Referências bibliográficas

ALI, Z. A.; RASHEED, S. A.; ALI, N. N. An enhanced hybrid genetic algorithm for solving traveling salesman problem. **Indonesia Journal of Electrical Engineering and Computuper Science**, v. 18, n. 2, p. 1035–1039, 2020.

ANDERS WIRÉN, P. nobis. **Example The travelling salesman problem**. [S.l.: s.n.]. `https://commons.wikimedia.org/wiki/File:Example_The_travelling_salesman_problem_(TSP).gif`. Acessado em: 19 de Setembro.

BÄCK, T.; FOGEL, D. B.; MICHALEWICZ, Z. **Evolutionary computation 1: Basic algorithms and operators**. [S.l.]: CRC press, 2018.

BOOKER, L. B.; GOLDBERG, D. E.; HOLLAND, J. H. Classifier systems and genetic algorithms. **Artificial intelligence**, Elsevier, v. 40, n. 1-3, p. 235–282, 1989.

CHEN, X. et al. A new evolutionary multiobjective model for traveling salesman problem. **Ieee Access**, IEEE, v. 7, p. 66964–66979, 2019.

DARAIO, C.; KERSTENS, K. H.; NEPOMUCENO, T. C. C.; SICKLES, R. Productivity and efficiency analysis software: an exploratory bibliographical survey of the options. **Journal of Economic Surveys**, Wiley Online Library, v. 33, n. 1, p. 85–100, 2019.

DARWIN, C. **On natural selection**. [S.l.]: Penguin UK, 2004.

EIBEN, A. E.; SMITH, J. E. **Introduction to evolutionary computing**. [S.l.]: Springer, 2015.

EMAMBOCUS, B. A. S. et al. An enhanced swap sequence-based particle swarm optimization algorithm to solve TSP. **IEEE Access**, IEEE, v. 9, p. 164820–164836, 2021.

FERRO, J. V. R. Um Algoritmo Genético auto Adaptável pela Fuzzificação da Taxa de Mutação. **International journal of endocrinology and metabolism**, Universidade Federal de Alagoas, p. 22, 2022.

GHASEMI, A.; ZAHEDIASL, S. Normality tests for statistical analysis: a guide for non-statisticians. **International journal of endocrinology and metabolism**, Brieflands, v. 10, n. 2, p. 486, 2012.

GOLDBERG, D. E. Optimization, and machine learning. **Genetic algorithms in Search**, Addison-wesley, 1989.

GOMES, D. E. et al. Applying a genetic algorithm to a m-TSP: case study of a decision support system for optimizing a beverage logistics vehicles routing problem. **Electronics**, MDPI, v. 10, n. 18, p. 2298, 2021.

GRAVETTER, F. J.; WALLNAU, L. B.; FORZANO, L.-A. B.; WITNAUER, J. E. **Essentials of statistics for the behavioral sciences**. [S.l.]: Cengage Learning, 2020.

GUIMARAES, A. C. d. B. P. et al. Uma análise comparativa sobre as estratégias de inicialização da população em algoritmos genéticos para o problema de eficiência energética em trens urbanos. Universidade Federal da Paraıba, p. 26, 2017.

HASSANAT, A. B. et al. An improved genetic algorithm with a new initialization mechanism based on regression techniques. **Information**, MDPI, v. 9, n. 7, p. 167, 2018.

IBADA ALI, J.; TÜŰ-SZABÓ, B.; KÓCZY, T. L. Effect of the initial population construction on the DBMEA algorithm searching for the optimal solution of the traveling salesman problem. **INFOCOMMUNICATIONS JOURNAL**, v. 14, n. 3, p. 72–78, 2022.

KATO R. PAIVA V., I. S. **Algoritmos Genéticos**. [S.l.: s.n.]. `https://bioinfo.com.br/algoritmos-geneticos/`. Acessado em: 19 de Setembro.

KATOCH, S.; CHAUHAN, S. S.; KUMAR, V. A review on genetic algorithm: past, present, and future. **Multimedia Tools and Applications**, Springer, v. 80, p. 8091–8126, 2021.

LINDEN, R. **Algoritmos genéticos (2a ediçao)**. [S.l.]: Brasport, 2008.

LIVINGSTON, E. H. Who was student and why do we care so much about his t-test? 1. **Journal of Surgical Research**, Elsevier, v. 118, n. 1, p. 58–65, 2004.

MACEIÓ, P. C. **Malha Viária Maceió**. [S.l.: s.n.]. `https://www.cbtu.gov.br/index.php/pt/sistemas-cbtu/maceio`. Acessado em: 13 de Setembro.

MARIANO, D. et al. Revista Brasileira de Bioinformática e Biologia Computacional, 2021.

MIRANDA, M. N. de. **Algoritmos Genéticos: Fundamentos e Aplicações**. [S.l.: s.n.], 2007.

NEPOMUCENO, T. C. C.; COSTA, A. P. C. S.; DARAIO, C. Theoretical and Empirical Advances in the Assessment of Productive Efficiency since the introduction of DEA: A Bibliometric Analysis. **International Journal of Operational Research**, Inderscience Publishers (IEL), v. 46, n. 4, p. 505–549, 2023.

NIVIANE RODRIGUES, S. F. e. **Prefeito JHC autoriza instalação de empresas que vão gerar mais de 3 mil empregos em Maceió**. [S.l.: s.n.]. `https://maceio.al.gov.br/noticias/gp/prefeito-jhc-autoriza-instalacao-de-empresas-que-vao-gerar-mais-de-3-mil-empregos-em-maceio`. Acessado em: 13 de Setembro.

NUMFOCUS, I. **Pandas Documentation**. [S.l.: s.n.]. `https://pandas.pydata.org/pandas-docs/stable/`. Acessado em: 19 de Setembro.

NUMPY, I. **NumPy Documentation**. [S.l.: s.n.]. `https://numpy.org/doc/`. Acessado em: 19 de Setembro.

NUNES, J. R. d. O. **Avaliação de taxas de cruzamento e mutação em um algoritmo genético baseado em ordem aplicado ao problema do caixeiro viajante**. 2018. B.S. thesis – Universidade Federal do Rio Grande do Norte.

POZO, A. et al. Computação evolutiva. **Universidade Federal do Paraná, 61p.(Grupo de Pesquisas em Computação Evolutiva, Departamento de Informática-Universidade Federal do Paraná)**, 2005.

PYTHON, O. **Python Documentation**. [S.l.: s.n.]. `https://www.python.org/doc/`. Acessado em: 19 de Setembro.

SONI, N.; KUMAR, T. Study of Various Mutation Operators in Genetic Algorithm. **International Journal of Computer Science and Information Technologies**, v. 5, n. 3, p. 4519–4521, 2014.

TEAM, J. **The Jamovi Project (2023)**. [S.l.: s.n.]. `https://www.jamovi.org/`. Acessado em: 2 de Agosto.

WEI, B.; XING, Y.; XIA, X.; GUI, L. A novel particle swarm optimization with genetic operator and its application to tsp. **International Journal of Cognitive Informatics and Natural Intelligence (IJCINI)**, IGI Global, v. 15, n. 4, p. 1–17, 2021.

WU, C.; FU, X.; PEI, J.; DONG, Z. A novel sparrow search algorithm for the traveling salesman problem. **IEEE Access**, IEEE, v. 9, p. 153456–153471, 2021.

XU, X.; YUAN, H.; LIPTROTT, M.; TROVATI, M. Two phase heuristic algorithm for the multiple-travelling salesman problem. **Soft Computing**, Springer, v. 22, p. 6567–6581, 2018.

YANG, B. et al. A novel path planning algorithm for warehouse robots based on a two-dimensional grid model. **IEEE Access**, IEEE, v. 8, p. 80347–80357, 2020.

YANG, X.-S. **Nature-inspired optimization algorithms**. [S.l.]: Academic Press, 2020.