INSTITUTO FEDERAL DO ESPÍRITO SANTO
CURSO SUPERIOR DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

RODRIGO RAIDER DE OLIVEIRA

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

Serra
2023

RODRIGO RAIDER DE OLIVEIRA

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

Trabalho de Conclusão de Curso apresentado à Coordenadoria do Curso de Bacharelado em Sistemas de Informação do Instituto Federal do Espírito Santo, Campus Serra, como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof.º Dr.º Leandro Colombi Resendo

Serra
2023

Dados Internacionais de Catalogação na Publicação (CIP)

O48s Oliveira, Rodrigo Raider de
2023 Solucionando o problema de roteamento de veículos capacitado com uso de algoritmos genéticos e paralelismo / Rodrigo Raider de Oliveira. - 2023.
60 f.; il.; 30 cm

Orientador: Prof. Dr. Leandro Colombi Resendo.

Monografia (graduação) - Instituto Federal do Espírito Santo, Coordenadoria de Informática, Curso de Bacharelado em Sistemas de Informação, 2023.

1. Algoritmo genérico. 2. CVRP. 3. Paralelismo. 4. I. Resendo, Leandro Colombi. II. Instituto Federal do Espírito Santo. III. Título.

CDD 004

Bibliotecário: Valmir Oliveira de Aguiar - CRB6/ES 566

RODRIGO RAIDER DE OLIVEIRA

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

Trabalho de Conclusão de Curso apresentado como parte das atividades para obtenção do título de Bacharel em Sistemas de Informação, do curso de Bacharelado em Sistemas de Informação do Instituto Federal do Espírito Santo.

Aprovado em 05 de dezembro de 2023.

COMISSÃO EXAMINADORA

_____

Prof. Dr. Leandro Colombi Resendo (Orientador)
Instituto Federal do Espírito Santo - Campus Serra

_____

Prof. Dr. Sérgio Nery Simões
Instituto Federal do Espírito Santo - Campus Serra

_____

Prof. Dr. Daniel Cruz Cavalieri
Instituto Federal do Espírito Santo - Campus Serra

MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL DO ESPÍRITO SANTO
SISTEMA INTEGRADO DE PATRIMÔNIO, ADMINISTRAÇÃO E CONTRATOS

FOLHA DE ASSINATURAS

Emitido em 05/12/2023

FOLHA DE ROSTO Nº 93/2023 - SER-CGEN (11.02.32.01.08.02)

(Nº do Protocolo: NÃO PROTOCOLADO)

(Assinado digitalmente em 05/12/2023 18:08)
DANIEL CRUZ CAVALIERI
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
SER-CGEN (11.02.32.01.08.02)
Matrícula: 1986870

(Assinado digitalmente em 05/12/2023 16:16)
LEANDRO COLOMBI RESENDO
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
SER-CGEN (11.02.32.01.08.02)
Matrícula: 1687072

(Assinado digitalmente em 06/12/2023 18:19)
SERGIO NERY SIMOES
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
SER-CGEN (11.02.32.01.08.02)
Matrícula: 1418911

Visualize o documento original em https://sipac.ifes.edu.br/documentos/ informando seu número: **93**, ano: **2023**, tipo: **FOLHA DE ROSTO**, data de emissão: **05/12/2023** e o código de verificação: **40ea06ab9d**

AGRADECIMENTOS

Gostaria de expressar minha gratidão a todas as pessoas que diretamente e indiretamente participaram deste processo.

Ao IFES e aos seus colaboradores que abriram essas portas, aos professores do campus Serra que contribuíram durante esta trajetória.

À minha família por me ajudar a superar os desafios que, de muitas formas, foram fundamentais para que este trabalho pudesse acontecer.

Ao professor orientador deste, Leandro Colombi Resendo, pelo direcionamento, pela disponibilidade, pelo compartilhamento, pela paciência, pelo empenho, e por abrir as portas para o tema deste trabalho através do ensino.

Em especial, agradeço a minha mãe pelo incentivo, pela insistência e pela resiliência. Uma pessoa ímpar na minha vida, isso não seria possível sem você.

RESUMO

O Problema de Roteamento de Veículos Capacitado (CVRP) é um clássico problema de otimização combinatória que lida com o roteamento de veículos para atender às demandas de clientes. Rotas são projetadas para partir de um único depósito, formando um conjunto de circuitos hamiltonianos que visa minimizar o custo total, obedecendo às restrições de capacidade de carga e atomicidade. O CVRP é classificado como NP-difícil, onde o espaço de solução cresce exponencialmente com o número de clientes, tornando inviável obter soluções ótimas em tempo hábil. O uso de heurísticas tem se mostrado uma alternativa na construção de algoritmos para a sua resolução. Dentro desse contexto, o Algoritmo Genético (AG) surge como opção promissora, explorada por trabalhos acerca do tema. Neste, propomos uma abordagem baseada no AG para a resolução do CVRP, aprimorando-o com o uso de paralelismo. Um algoritmo foi implementado usando o modelo de ilhas, explorando *multithreading* e memória compartilhada. Um total de 24 instâncias, divididas em grupos, foram utilizadas como experimentos para a solução desenvolvida, selecionadas a partir de *benchmarks* na literatura. Foram realizadas variações no número de ilhas durante os testes para avaliar o impacto do paralelismo. Os resultados revelaram uma melhoria na qualidade das soluções à medida que mais ilhas foram adicionadas, apesar do aumento no custo computacional. O AG padrão apresentou uma distância média de 19,79% em relação às soluções ótimas. No entanto, com 2 ilhas, essa distância diminuiu para 16,22%, com 4 ilhas para 14,40%, e com 6 ilhas para 13,23%. O aumento médio no custo de processamento, medido em segundos, variou entre 1,08(1), 1,11(2), 1,13(4) e 1,31(6). Além disso, os resultados também demonstraram uma redução na variância, resultando em maior consistência com o uso de mais ilhas. O método proposto mostrou-se promissor na abordagem de problemas de maior complexidade, particularmente em cenários onde as soluções convencionais encontram limitações. Ele se destaca como alternativa através da escalabilidade horizontal dos recursos computacionais.

Palavras-chave: Problema de Roteamento de Veículos, Algoritmo Genético, Paralelismo, Modelo de Ilhas.

ABSTRACT

The Capacitated Vehicle Routing Problem (CVRP) is a classic combinatorial optimization problem that involves routing vehicles to meet customer demands. Routes are designed to originate from a single depot, forming a set of Hamiltonian circuits aimed at minimizing the total cost while adhering to load capacity and atomicity constraints. CVRP is classified as NP-hard, where the solution space grows exponentially with the number of customers, making it impossible to obtain optimal solutions in a timely manner. In this context, the use of heuristics has proven to be an alternative in constructing algorithms to solve it. Within this framework, the Genetic Algorithm (GA) has emerged as a promising option, as explored by studies on the subject. In this article, we propose a GA-based approach to solving the CVRP, enhancing it with the use of parallelism. An algorithm was implemented using the island model, leveraging multithreading and shared memory. A total of 24 instances, divided into groups, were used as experiments for the developed solution, selected from benchmarks in the literature. Variations were made to the number of islands during the tests to assess the impact of parallelism. The results demonstrated an improvement in the quality of solutions as more islands were added, despite the increase in computational cost. The standard GA exhibited an average distance of 19.79% from the optimal solutions. However, with 2 islands, this distance decreased to 16.22%, with 4 islands to 14.40%, and with 6 islands to 13.23%. The average increase in processing cost, measured in seconds, varied between 1.08(1), 1.11(2), 1.13(4), and 1.31(6). Additionally, the results also indicated a reduction in variance, leading to greater consistency with the use of more islands. The proposed method has demonstrated promise in addressing problems of greater complexity, particularly in scenarios where conventional solutions encounter limitations. It stands out as an alternative due to the horizontal scalability of computational resources.

Keywords: Vehicle Routing Problem, Genetic Algorithm, Parallelism, Island Model.

LISTA DE FIGURAS

LISTA DE SIGLAS

AG	– Algoritmo Genético
BPP	– *Bin Packing Problem*
PCV	– Problema do Caixeiro Viajante
PCVM	– Problema do Caixeiro Viajante Múltiplo
PMX	– *Partially-Mapped Crossover*
VRP	– *Vehicle Routing Problem*
CVRP	– *Capacitated Vehicle Routing Problem*

SUMÁRIO

1 INTRODUÇÃO

Nos últimos séculos, o gerenciamento integrado da logística se tornou uma necessidade para as organizações. À medida em que as Tecnologias da Informação e da Comunicação (TICs) foram sendo aprimoradas, houve uma globalização dos serviços e produtos, que cada vez mais estão fragmentados. Hoje, suas relações são baseadas em fluxos, de informações (do cliente para o fornecedor) e de materiais (do fornecedor ao cliente), onde todos são clientes e fornecedores simultâneamente. Supondo uma empresa que necessita de *commodities* para montar seu produto final, é bem provável que nos dias atuais encontre fornecedores com preços mais atrativos para muitos desses, do que se ela própria a produzisse. Essa redução de preços pode se passar pela especialização na produção, mas também é fortemente impactada pela redução nos custos logísticos. As instituições, principalmente as privadas, precisaram transformar seus meios de produção para suportar a competição a partir desta perspectiva (BALLOU, 2006).

No Brasil, a forte predominância do transporte rodoviário acentua ainda mais a importância na obtenção de soluções que visem a redução e otimização de custos de logística. De acordo com o último Plano de Transporte e Logística divulgado pelo CNT (2018), o país transportou no ano de 2017 um total de 61,1% das suas cargas pelo modal rodoviário. Em 2016, o custo do transporte correspondeu a cerca de 55% dos custos totais de logística, que envolvem também estocagem, armazenagem e administrativo. Ainda segundo o estudo, esses valores representaram 12,3% do produto interno bruto (PIB) brasileiro no período, enquanto nos Estados Unidos esse custo era de 7,8%, que serve a nível de comparação pelas suas dimensões continentais.

A integração de diversas áreas relacionadas que impactam e são impactadas pelos serviços logísticos é conhecida como cadeia de suprimentos. Segundo Ballou (2006), o planejamento eficaz dessa cadeia requer o alinhamento de estratégias de estoque, transporte e localização, que são fortemente dependentes. Pode ser visto como um sistema logístico expresso como redes abstratas de ligações e nós, onde essas precisam ser gerenciadas com eficácia, pois impactam diretamente nos custos de transporte. A seleção da modalidade e a roteirização e programação dos veículos fazem parte das principais atividades que compõem as decisões estratégicas dentro da cadeia de suprimentos.

O Problema de Roteamento de Veículos (*Vehicle Problem Routing* - VRP) é um conceito genérico para a classe de problemas de roteirização e programação. Consiste na determinação de rotas a serem percorridas por veículos de uma rede de serviços, sob a perspectiva da otimização, sujeitos a um conjunto de recursos e um conjunto de restrições, visando o atendimento de demandas de clientes (LEE; UENG, 1999). Ao roteirizar, deseja-se alcançar uma meta, que geralmente é a minimização de um ou mais objetivos.

Muitas variações para o VRP são citadas na literatura. Os formatos mais clássicos envolvem: depósito único ou múltiplo, veículos homogêneos ou heterogêneos, restrições de tempo como janelas de tempo, tipo de operação que podem envolver coleta ou entrega (ou ambos), uma ou mais viagens, entre outros. Mais recentemente, com a mudança na dinâmica da distribuição de bens e serviços, aplicações em tempo real demandam que os sistemas de roteamento sejam capazes de reajustar, trazendo ainda mais a necessidade de sistemas capazes de recalcular rapidamente, com dados estocásticos e dinâmicos.

1.1 CVRP

Neste trabalho, olharemos para a roteirização, sob as características que formam o Problema de Roteamento de Veículos Capacitado (*Capacitated Vehicle Problem Routing* - CVRP). Dantzig e Ramser (1959) foi um dos primeiros autores a abordar o problema na perspectiva de roteirização de veículos, onde um conjunto de veículos homogêneos deveriam realizar a distribuição de gasolina entre diversos postos, considerando que a quantidade deixada em cada posto era unidimensional (litros). O autor define esse problema como uma extensão particular do Problema do Caixeiro Viajante. Esse trabalho é considerado precursor no desenvolvimento de modelagem de programação matemática e de abordagem algorítmica, se tornando um dos principais assuntos no ramo da otimização combinatória desde então.

Os principais elementos que caracterizam o CVRP estão condensados na Tabela 1.

Tabela 1 – Elementos presentes no CVRP.

Características do VRP	**Opções do CVRP**
Tamanho da frota	Múltiplos veículos
Tipos de veículos	Homogêneos, com restrição de capacidade
Origem dos veículos	Um único depósito
Tipo de demanda	Demanda determinística conhecida
Localização da demanda	Em cada cliente (vértices)
Características da rede	Não dirigido Custo simétrico
Tipo de operação	Somente entrega
Objetivos	Minimizar a distância percorrida (custos)
Restrições	Cada cliente é abastecido por um único veículo Todos os veículos devem iniciar e terminar no depósito Um veículo não pode parar duas vezes no mesmo cliente Nenhum veículo pode exceder sua capacidade máxima

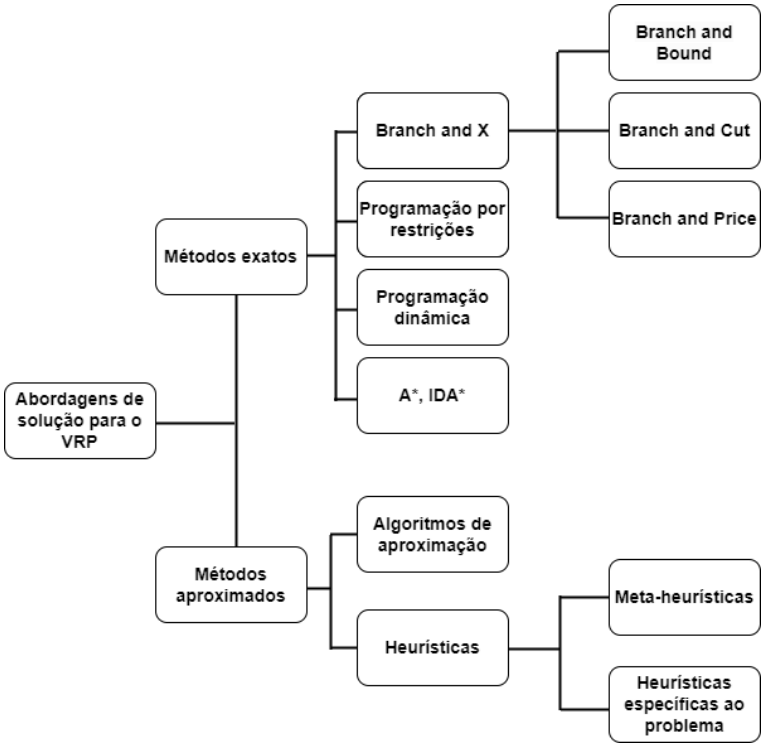
Fonte: Elaborado pelo autor (2022).

Nota: Características típicas do Problema de Roteamento de Veículos básico proposto pela primeira vez por (DANTZIG; RAMSER, 1959)

Classificado como NP-Difícil, a obtenção de soluções exatas para o CVRP ficam limitadas no sentido do tempo de processamento. Trabalhos voltados para solução exata demonstram alto esforço no tratamento de versões com muitos clientes, se limitando a resolver problemas simples, com menos de uma centena de clientes e uma dezena de veículos, em muitas horas de processamento. Essa limitação fomentou ao longo das últimas décadas abordagens

alternativas, priorizando também o requisito de tempo. Clarke e Wright (1964) propôs uma abordagem heurística gulosa muito referenciada na literatura, marcando o início de uma série de trabalhos desenvolvidos no contexto das heurísticas e de algoritmos de aproximação. A Figura 1 exemplifica, em uma visão genérica, o mapeamento das abordagens encontradas na literatura.

Figura 1 – Visão geral das abordagens propostas para o VRP.



Fonte: Adaptado de (TAN; YEH, 2021).

Heurística, de forma macro, é uma junção da necessidade de simplificar critérios para atingir um objetivo, com o desejo de que essa, vista na perspectiva da tomada de decisão, consiga discriminar corretamente boas e más escolhas (PEARL, 1984). Meta-heurística podem ser vistas como heurísticas em um nível superior. Condensam estratégias de melhoria local, ao mesmo tempo que provém procedimentos estratégicos que visam superar os mínimos locais. São abordagens mais genéricas, utilizadas para problemas complexos de natureza combinatória (GLOVER; KOCHENBERGER, 2003). Em computação, ambas são utilizadas com o objetivo de aumentar a velocidade de resolução de problemas, abrindo mão da garantia da otimalidade ou da integridade.

Para o desenvolvimento deste trabalho, foi definido a utilização do Algoritmo Genético (AG). O AG é uma meta-heurística que se baseia na busca por solução utilizando múltiplos candidatos, e o processo visa melhorar a qualidade deles, chamada de população. Caracterizada como evolucionária, ela se baseia na ideia de evolução biológica, onde os indivíduos melhores adaptados tendem a transferir suas boas qualidades às novas gerações, sendo essa melhor adaptada ao meio. Um código é desenvolvido para gerar possíveis

soluções dada uma função objetivo que determina a qualidade da solução, chamada de aptidão bruta. Para isso, o algoritmo genético é executado diversas vezes, até que a solução ótima seja obtida, o que nem sempre é possível, ou um valor n de gerações seja alcançado. Dada a complexidade do problema tratado, busca obter soluções para instâncias maiores em um tempo computacional viável, o AG proposto nesse trabalho foi projetado para usar processamento paralelo.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Desenvolver método de otimização para o CVRP clássico baseado em algoritmo genético implementado com recursos de paralelização.

Para isso, segue os objetivos específicos a seguir:

1.2.2 Objetivos Específicos

1. Modelar e implementar um algoritmo genético utilizando o conceito de ilhas com paralelismo e migração, com armazenamento de resultados e relatórios de execução.

2. Selecionar e testar instâncias a partir de *benchmark* da literatura, adequando-os como entrada válida para o algoritmo desenvolvido.

3. Apresentar uma síntese dos resultados obtidos, a partir da perspectiva da minimização do custo total em comparação com os resultados ótimos.

4. Avaliar custos computacionais e analisar resultados obtidos nos testes, localizando-a sob os demais objetivos desejáveis nessa categoria de problema.

1.3 ORGANIZAÇÃO DO TRABALHO

O restante deste trabalho está dividido da seguinte forma: No Capítulo 2 está apresentado os principais conceitos que envolvem o CVRP necessários para entendimento do problema. Nele também é abordado os conceitos necessários para o entendimento dos Algoritmos Genéticos, bem como as estratégias para a sua paralelização. Adicionalmente, nesse capítulo também é apresentado trabalho correlatos atuais; No Capítulo 3 está apresentado os detalhes da heurística proposta nesse trabalho; No Capítulo 4, o projeto de experimentos e a calibração da heurística. No Capítulo 5, os resultados e análises. Finalmente, no Capítulo 6 estão apresentadas as principias conclusões e trabalhos futuros.

2 REFERENCIAL TEÓRICO

O Problema de Roteamento de Veículos (VRP) é uma área da Pesquisa Operacional que se desenvolveu em larga escala ao longo de quase toda o último século, mais precisamente desde os anos 60. Se tornou um alvo para o setor científico, visando o desenvolvimento de técnicas e abordagens para sua solução, uma vez que esses possuem alta aplicabilidade e resultam em economia real para instituições que desejam usar esse arcabouço na construção de *softwares* para uso em suas operações de transporte.

Suas primeiras versões e modelos tratavam de entrega de produtos a clientes, mas não se limitou a isso. Aos poucos novos tipos de operação foram adicionadas para atender aos mais diversos requisitos logísticos. O que era só coleta ou entrega, passou a envolver ambos simultaneamente. Mais tarde, também surgiu a observação que o transporte de pessoas poderia se tornar uma ramificação, como o *dial-a-ride*. A abordagem que iniciou-se com frotas homogêneas, passou a também ser heterogênea. Permitindo também a utilização de multi-depósitos, divisão de carga, quantidade múltiplas de viagens, regulamentação dos motoristas, restrições de tempo, entre várias outras características.

Como principais elementos, podemos citar:

Clientes são requerentes de serviços que estão alocados em locais geograficamente dispersos, que possuem ou requerem determinada demanda.

Demandas é a essencia do VRP, que pode ser uma ordem de transporte, como a entrega de uma encomenda ou o recolhimento de um item, ou, um serviço a ser realizado, por exemplo uma manutenção. Podem possuir requisitos de tempo, como limitação de horário de atendimento e intervalos, tipos de veículos que podem realizar o serviço, entre outras características.

Veículos de transporte com algum grau de mobilidade, genericamente representados por caminhões ou ônibus, e mais recentemente algumas abordagens tratam de drones e veículos autônomos. Possui um ou mais depósitos, capacidade limitada, podem possuir compartimentação. Também, restrições diversas como acesso limitado a determinadas rotas, clientes, e a tipo de produtos. Ainda, possuem custos de operação.

Objetivos é a relação dos clientes e veículos, através do trabalho realizado na definição das rotas, em relação aos objetivos primários. Tipicamente deseja-se minimizar os custos de deslocamento total dos veículos.

Recursos e restrições se entrelaçam como uma extensa lista, formando variantes do VRP. Drexl (2012) propõe uma divisão na perspectiva de requisições, frota, estrutura das rotas,

objetivos e escopo de planejamento. Eksioglu, Vural e Reisman (2009) aborda os problemas ricos apresentando mais detalhes a partir de dados coletados em diversos estudos da área, alocando-os sob cinco macrocategorias: tipo de estudo, características físicas do problema, do cenário, da informação e dos dados. A Tabela 2 ilustra um resumo das categorias e subcategorias que formam problemas ricos de roteamento na perspectiva de Lahyani, Khemakhem e Semet (2015), que classifica em características do cenário e do problema.

Tabela 2 – Taxonomia de VRP aplicáveis a problemas ricos

Características do cenário	**Características físicas do problema**
Dados de entrada	Veículos
Estático	Tipo
Dinâmico	Homogêneo
Determinístico	Heterogêneo
Estocástico	Número
Componentes de gerenciamento de decisão	Fixo
Roteamento	Sem limite
Inventário e roteamento	Estrutura
Localização e roteamento	Compartimentado
Roteamento e agendamento de motorista	Não compartimentado
Produção e planejamento de distribuição	Restrições de capacidade
Número de depósitos	Política de carregamento
Único	Ordem cronológica
Múltiplos	Nenhuma política
Tipo de operação	Regulamentos dos motoristas
Coleta ou entrega	Restrições de tempo
Coleta e entrega	Restrição ao cliente
Backhauls	Restrição de acesso rodoviário
Dial-a-ride	Restrição no depósito
Restrições de divisão de carga	Tempo de serviço
Divisão permitida	Tempo de espera
Divisão não permitida	Estrutura da janela de tempo
Período de planejamento	Janela de tempo única
Período único	Janela de tempo múltipla
Períodos múltiplos	Restrições de incompatibilidade
Uso múltiplo de veículos	Restrições específicas
Viajem única	Função objetivo
Múltiplas viagens	Objetivo único
	Múltiplos objetivos

Fonte: Adaptado de (LAHYANI; KHEMAKHEM; SEMET, 2015)

As características do cenário surgem como elementos modificadores a nível estratégico e tático, enquanto as características físicas trazem ao problema mais possibilidades Lahyani, Khemakhem e Semet (2015). Mais definições acerca da taxonomia e outras características podem ser consultados em Toth e Vigo (2002), Eksioglu, Vural e Reisman (2009), Drexl (2012) e Lahyani, Khemakhem e Semet (2015).

2.1 O PROBLEMA DE ROTEAMENTO DE VEÍCULOS CAPACITADO

O Problema de Roteamento de Veículos Capacitado ou CVRP, foi a primeira variante do VRP, definido pela primeira vez por (DANTZIG; RAMSER, 1959). Suas principais características: (a) há um depósito único; (b) os veículos são homogêneos, possuindo o mesmo limite de carga; (c) tipo de operação é entrega; (d) os dados são determinísticos, permitindo o planejamento de antemão; (e) tem como objetivo minimizar o custo total, que pode ser obtido através de funções de custo associadas a distância euclidiana entre os pontos.

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

Uma função de custo não negativo $c_{i,j} : A \rightarrow \mathbb{Z}^+$ é associado a cada arco de G e uma função de demanda $d : V \rightarrow \mathbb{Z}^+$ é associada aos seus vértices. O depósito possui uma demanda $d_0 = 0$. Ciclos (loops) são proibidos , isso geralmente é tratado ao definir $c_{i,i} = +\infty$.

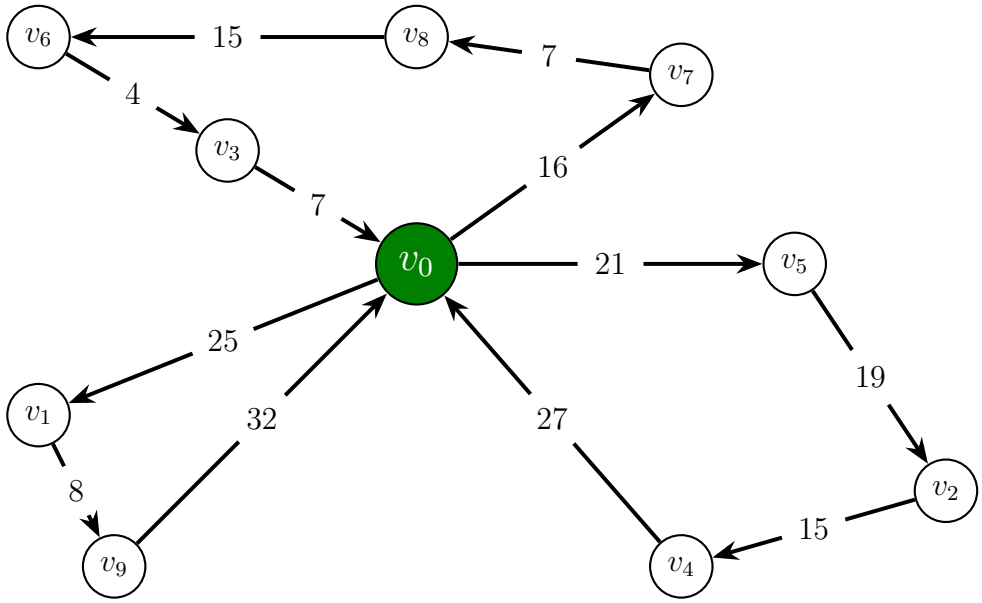
Uma frota de veículos homogênea que possui capacidade Q é inicializada no depósito (v_0). O objetivo é encontrar um conjunto de rotas com custo mínimo tal que (LAPORTE et al., 2000):

- Todas as rotas começam e terminam no depósito.
- Todas as demandas dos clientes são atendidas exatamente uma vez por um único veículo.
- Todos os $|K|$ veículos são utilizados, cada um associado a apenas uma rota.
- A capacidade de cada veículo não é excedida.

A Figura 2 demonstra um exemplo da aplicação da roteirização, v_0 (em verde) representando o depósito de onde três veículos deverão percorrer uma sequência visando o custo mínimo.

Segundo Ralphs et al. (2003), sua solução está associada ao relaxamento de dois problemas combinatórios difíceis, o Problema do Caixeiro Viajante Múltiplo (PCVM) e o Problema do Empacotamento (BPP). No relaxamento relacionado ao PCVM, cria-se instâncias virtuais do depósito não conectados para cada veículo extra $(k - 1)$, convertendo-a em uma instância do PCV. Para saber se uma instância respeita as restrições de carga, existe

Figura 2 – Exemplo do grafo que representa um roteamento aplicado ao CVRP.



Fonte: Elaborado pelo autor (2022).
Nota: O grafo demonstra uma solução. Representado ao centro, o depósito (em verde). Três subrotas que estão indicadas com as setas, mostrando a precedência da visita de cada vértice.
Nota: O custo total do exemplo é 196, onde os veículos percorrem na ordem $k_0 = \{0, 7, 8, 6, 3, 0\}$, $k_1 = \{0, 1, 9, 0\}$ e $k_2 = \{0, 5, 2, 4, 0\}$.
Nota: A demanda não está expressa nos vértices, abstraindo que elas não ultrapassaram a limitação de carga dos veículos.

o tratamento de uma instância do BPP. A solução do CVRP é obtida a partir da solução do PCV que satisfaz as restrições de carga do BPP de cada k segmento das rotas obtidas.

Um modelo de programação linear de três índices proposto por Borcinova (2017) pode ser usado como base para a ilustração do problema:

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

Sujeito a

$$\sum_{r=1}^{p}\sum_{i=0,i\neq j}^{n} x_{rij} = 1, \qquad \forall j \in \{1, ..., n\} \tag{2}$$

$$\sum_{j=1}^{n} x_{r0j} = 1, \qquad \forall r \in \{1, ..., p\} \tag{3}$$

$$\sum_{i=0,i\neq j}^{n} x_{rij} = \sum_{i=0}^{n} x_{rji}, \qquad \forall j \in \{0,...,n\}, r \in \{1,...,p\} \tag{4}$$

$$\sum_{i=0}^{n} \sum_{j=1,i\neq j}^{n} d_j x_{rij} \leq Q, \qquad \forall r \in \{1,...,p\} \tag{5}$$

$$\sum_{r=1}^{n} \sum_{i\in S} \sum_{j\in S,i\neq j} x_{rij} \leq |S| - 1, \qquad \forall S \subseteq \{1,...,n\} \tag{6}$$

$$x_{rij} \in 0,1, \qquad \forall r \in \{1,...,p\}, i,j \in \{1,...,n\}, i \neq j \tag{7}$$

(1) representa a função objetivo que visa a minimização do custo total. (2) garante que cada cliente seja visitado por exatamente um veículo. (3) e (4) garantem que cada veículo deixa o depósito uma única vez, e que a quantidade de veículos que deixa depósito e visita os clientes é igual ao número de veículos que voltam. (5) garante que a capacidade dos veículos sejam respeitas. (6) garante que a solução não sub-rotas externas ao depósito. (7) determina o domínio das variáveis.

A definição original proposta no trabalho de Dantzig e Ramser (1959) foi ganhando mais características ao longo dos anos, adicionando algumas extensões:

- Problema de Roteamento de Veículos Simétrico: na perspectiva dos custos, transitar entre os nós é simétrico quando $c_{i,j} = c_{j,i}$, custos iguais.

- Problema de Roteamento de Veículos Assimétrico: na perspectiva dos custos, transitar entre os nós é assimétrico quando $c_{i,j} \neq c_{j,i}$, custos diferentes. Usado para representar custos diferentes se os trajetos são diferentes.

- Problema de Roteamento de Veículos com Restrição de Distância: na perspectiva da distância, os veículos possuírem limitação máxima percorrida. Por exemplo, a limitação de quilometros que um veículo pode percorrer. Pode servir para evitar a necessidade de reabastecimento em operação.

- Problema de Roteamento de Veículos com Restrição de Tempo: na perspectiva do tempo percorrido, os veículos possuírem limitação máxima de tempo em operação. Exemplo, uma rota não pode demorar mais que um determinado valor. Pode ser usado como estratégia para respeitar os limites de horário de carga dos funcionários.

- 2L-CVRP: na perspectiva da dimensionalidade dos recursos transportados, os veículos possuem restrições geométricas bidimensionais (x, y), assim como os recursos, requerendo o tratamento do BPP enquanto validador de solução. Usado para programar as rotas respeitando a capacidade de volume dos veículos.

- 3L-CVRP: em similaridade com o 2L-CVRP, considera-se entretanto, a geometria tridimensional (x, y, z).

2.1.1 Abordagens Exatas

As abordagens exatas visam a garantir que a solução ótima seja obtida. Para isso, é necessário percorrer todo o espaço de busca, ou, localizar alternativas metodológicas que sejam capazes de limitar esse processo. Deve-se garantir que tudo aquilo que não foi investigado minuciosamente, de fato, não seja o valor buscado pelo objetivo de maximização ou minimização.

Naturalmente, muitas das abordagens exatas tem como princípio soluções desenvolvidas que são voltadas para o Problema do Caixeiro Viajante (PCV). Christofides, Mingozzi e Toth (1981) foram os precursores ao abordar uma solução exata no contexto do VRP, através da técnica de *branch-and-bound* (do português, ramificar e limitar). Como fundamento, tem-se o propósito de enumerar todo o espaço de busca em formato de árvore, utilizando a relaxação Lagrangiana. Implementa-se um mecanismo baseado em limites que visam podar a busca exaustiva por todo o espaço de soluções. Sempre que o algoritmo encontra uma solução de pior qualidade ou uma solução infactível, ela encerra a busca daquela ramificação.

O *branch-and-bound* serviu como inspiração para métodos mais eficientes: *branch-and-cut*, *branch-and-price* e *branch-price-and-cut*.

A estratégia *branch-and-cut* tem como princípio a ramificação baseada em desigualdades válidas, com o objetivo de fazer cortes mais eficientes em relação ao *branch-and-bound*. A escolha dos critérios, para isso, impacta diretamente nos algoritmos desenvolvidos, e por isso o método foi sendo aprimorado, resolvendo problemas com cada vez mais veículos e clientes, com dificuldades inerentes aos seus diversos cenários. Se destacam na literatura trabalhos como de Augerat et al. (1995), Blasum e Hochstättler (2000) e Lysgaard, Letchford e Eglese (2004). Augerat et al. (1995) utilizou critérios heurísticos que permitiram que instâncias ainda não solucionadas com otimalidade pudessem ser alcançadas. Blasum e Hochstättler (2000) seguiu investigando classes de desigualdades, também obtendo resultados inéditos. Lysgaard, Letchford e Eglese (2004) explorou mais planos de cortes, fornecendo algoritmos mais novos e eficientes.

2.1.2 Abordagens Aproximadas

Até o momento, as soluções exatas disponíveis na literatura não alcançam tempos razoáveis que as tornem úteis para problemas de tamanhos e complexidade do mundo real, por isso muitas abordagens aproximadas são encontradas na literatura. Heurísticas e meta heurísticas são exploradas nesse sentido. No fim não se garante a obtenção do ótimo global, mas visa-se uma redução no custo de processamento.

Por ser um problema bem difundido, é possível encontrar os mais diversos tipos de trabalho em cima das principais meta-heurísticas disponíveis e abordagens mistas que as mesclam com outros formatos. Toth e Vigo (2003) foi um dos primeiros trabalhos a utilizar a meta-heurística Busca Tabu, olhando a partir da perspectiva da diminuição do tempo computacional. Busca por vizinhança (*neighborhood search*) é explorada com diversas variações na literatura, mais recentemente temos como exemplos o formato variável em Amous et al. (2017), e com uso de rede neural (HOTTUNG; TIERNEY, 2019). Resultados consistentes com uso hibrido da meta-heurística do vagalume podem ser encontradas em Altabeeb, Mohsen e Ghallab (2019).

Algumas soluções partem do princípio do agrupamento (*clustering*) dos nós clientes, formando subproblemas. Esses subproblemas podem ser abordados internamente, então, como versões do Problema do Caixeiro Viajante, cujo princípio é reduzir a busca em espaços menores onde técnicas exatas podem ser exploradas. A qualidade na utilização deste tipo de técnica é também impactada pela parte de processamento voltada na obtenção desses agrupamentos, relacionando-as com a quantidade de veículos disponíveis e os limites de carga suportados, bem como a visitação de mais de um cluster (BATTARRA; ERDOGAN; VIGO, 2014).

Além disso, heurísticas construtivas são bastante referenciadas, como a Busca do Vizinho mais Próximo (*Nearest Neighbor Search*), que se baseia na formação das rotas adicionando iterativamente os clientes mais próximos, é um método naturalmente rápido de construir uma solução; e a Economia de Clarke and Wright, que parte do princípio de gerar agrupamentos que maximizem a economia.

2.2 ALGORITMOS GENÉTICOS

Algoritmos genéticos (GA) é um algoritmo de busca, desenvolvido por John Holland durante os anos 60 e 70, que visa projetar sistemas artificiais que abstraiam mecanismos importantes da biologia evolutiva (GOLDBERG, 1989). A noção de população é um dos pilares dessa inspiração, onde existe um conjunto de indivíduos que convivem em ambiente de competição, em que os mais adaptados tendem a transmitir suas características às novas gerações (MAN; TANG; KWONG, 1996).

Muitos problemas computacionais exigem *adaptabilidade*, pois o ambiente pode sofrer mudanças e precisa manter um bom desempenho; *inovação*, quando um algoritmo precisa realizar uma tarefa ou descoberta com potencial inovativo, como para uso de descobertas científicas; ou requerer uma *solução complexa*, quando a programação de um conjunto de regras não é trivial. Os algoritmos genéticos conseguem condensar essas características (MITCHELL, 1998). São recomendados para uso quando há a exigência de busca com grande quantidade de cenários possíveis, onde os algoritmos exatos não conseguem convergir para a solução ótima, por ausência de técnica capaz de alcançá-la analiticamente, ou limitação computacional de percorrer todo o espaço de busca, o que torna o tempo de execução muito alto para sua aplicabilidade em problemas reais.

Segundo (GOLDBERG, 1989), sistemas robustos artificiais precisam alinhar eficiência e eficácia para serem úteis aos mais diversos tipos de ambientes. Alguns algoritmos são desenvolvidos baseando-se em um processo que visa melhorar uma solução, partindo de uma arbitrária, conhecida como busca local ou *hill climbing*. Entretanto, sua qualidade está fortemente associada a essa localização inicial, ficando presa ao máximo (ou mínimo) local, o que não garante a eficácia. A utilização de uma população aumenta as chances de que as diversas potenciais soluções estejam melhor distribuídas, mesmo elas sendo estocásticas. A utilização de gerações, no processo de evolução, ajuda na manutenção daquilo que pode ser visto como boa solução (dado aquele momento), ao mesmo tempo que a distribuição em diversas possíveis soluções ajudam a monitorar um conjunto maior do espaço de busca. Ainda segundo Goldberg, a inspiração nos mecanismos biológicos, base da computação evolucionária, contém recursos para auto-reparo, auto-orientação e reprodução, que são elementos presentes nos sistemas artificiais sofisticados.

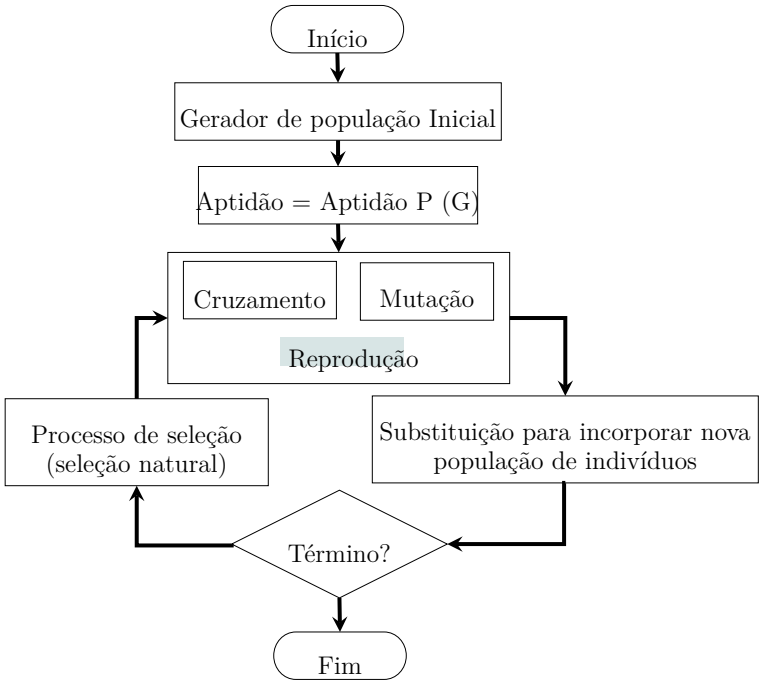
Os GA pegam emprestado a nomenclatura da biologia nos seus componentes, embora não tragam o mesmo significado literal, e sim seu princípio no processo. Um GA é mantém uma população $P(t) = \{x_1^t, ..., x_1^n\}$ por cada ciclo (ou geração) t. Cada indivíduo x representa uma solução em potencial codificada, também chamada de cromossomo, que carrega um conjunto de genes, que são as características dessa (MICHALEWICZ, 1996).

O fluxo de processamento dessas populações é realizado através de ciclos que contém: seleção (ver 2.2.4), cruzamento (ver 2.2.5) e mutação (ver 2.2.6). A Figura 3 a seguir demonstra o fluxograma de um GA tradicional.

2.2.1 Parâmetros do Algoritmo

Os parâmetros são pontos fundamentais na construção da solução de algoritmo genético. São fatores que influenciam diretamente no seu desempenho. Um equilíbrio entre eles é necessário para evitar gargalos, ou para evitar uma convergência prematura. Alguns deles são citados a seguir.

Figura 3 – Fluxograma básico de um Algoritmo Genético tradicional.



Fonte: Adaptado de (SHUKLA; PANDEY; MEHROTRA, 2015).

Tamanho da população o algoritmo deve manter uma população de tamanho definido. O custo do processamento dos operadores, do cálculo de aptidão e o tamanho dos cromossomos impactam diretamente nessa escolha. Deseja-se a maior população possível, mas a fluidez das gerações é fundamental.

Número de gerações pode ser usado como critério para limitar a execução do algoritmo. O número de gerações precisa ser suficiente para buscas locais, ao mesmo tempo que deve fornecer probabilidades de que operadores, como a mutação, consiga evoluir os indivíduos. Entretanto, gerações excedentes podem se tornar desperdício de recursos.

Taxa de cruzamento determina a probabilidade de um cruzamento ocorrer. Um balanço é necessário, pois taxas de cruzamento muito altas podem aumentar as chances de que uma solução com potencial seja destruída antes de realizar uma busca local. Ao mesmo que, uma taxa baixa pode tornar a convergência lenta.

Taxa de mutação determina a probabilidade de uma mutação ocorrer. Uma taxa muito baixa pode reduzir a taxa de convergência, dado que o fator de aleatoriedade muitas vezes é um escape importante de soluções muito aquém da otimalidade. Uma taxa muito alta é trazer muita inconsistência ao método, onde as soluções podem tender a aleatoriedade.

2.2.2 Representação

Para alcançar o objetivo de otimização do problema faz-se necessário a codificação dos indivíduos em um conjunto finito de símbolos que representem as potenciais soluções em formato genético. Esses indivíduos codificados precisam passar pelos operadores genéticos (ver adiante) e ainda serem capazes de manter uma solução viável. Esse processo de conversão das variáveis do problema em nível computacional é conhecido como codificação. Após o processamento do algoritmo, as variáveis são convertidas ao seu estado original, processo chamado de decodificação.

Segundo Linden (2012), a representação é arbitrária a quem está o modelando, entretanto prioriza-se o atendimento dos requisitos mínimos:

- A representação deve ser simples.
- Soluções proibidas ao problema não devem fazer parte da representação
- As condições do problema devem estar implícitas na representação.

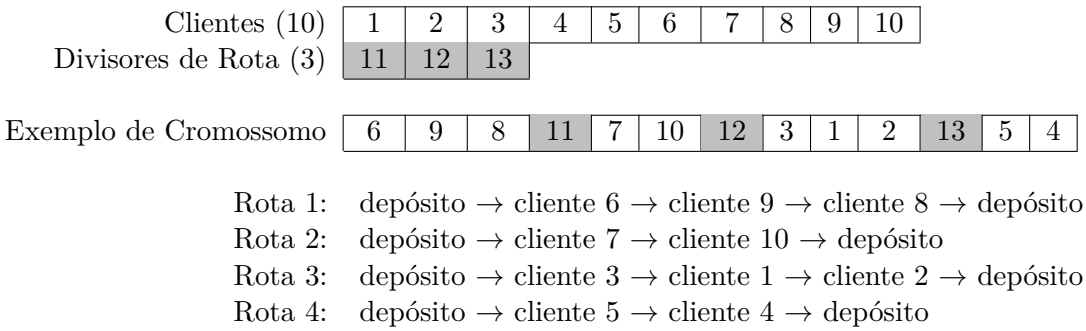
Na perspectiva do algoritmo genético, enquanto utilizado para otimização, codifica-se as variáveis de decisão como uma cadeia de caracteres (*string*), vetor, ou qualquer outro tipo de estrutura de dados. Segundo Mitchell (1998), o conceito de fenótipo está associado ao formato da solução na natureza do problema. Genótipo é um subespaço mapeado do fenótipo.

O formato mais tradicional na literatura é a **codificação binária**. Geralmente é utilizada em problemas de natureza binária, ou através da conversão do problema não-binário em uma sequencia binária de caracteres utilizando a técnica código de Gray (*Gray code*). O tipo é amplamente utilizado para problemas de estimativa de parâmetros numéricos, problemas de seleção de subconjuntos e problemas de sequenciamento (LUCASIUS; KATEMAN, 1994).

Mais recentemente outras formas de abordagens são trabalhadas na literatura. As **variáveis discretas** podem ser usadas para representar problemas de permutação (REEVES; ROWE, 2002). **Variáveis não discretas** são opções, com uso de vetor com valores de ponto flutuante. Michalewicz (1996) realizou experimentos que demonstraram que sob a perspectiva de conjuntos grandes de variáveis, quando a quantidade de bits necessários se tornam excepcionalmente grandes, a representação em ponto flutuante demonstrou maior consistência e precisão em um comparativo utilizando também valores binários. Esse cenário é caracterizado principalmente quando há um espaço de busca com alta dimensionalidade, e requer operadores genéticos que sejam capazes de tratá-los.

Em abordagens que tratam de soluções em grafos, como no CVRP, é comum a adoção da representação com permutação sequencial. Cada cromossomo indica a ordem em que os clientes devem ser visitados. Tal formato exige que cada gene contenha exatamente uma instância dos nós-clientes e dos divisores de rota no cromossomo. Uma vez que os operadores tradicionais produzem descendentes que duplicam ou omitem genes, deve-se incorporar mecanismos que contornem esse problema, sendo essa restrição uma das dificuldades naturais ao lidar com subproblemas do TSP em AG (FOX; MCMAHON, 1991). Cada permutação contém $[1...n]$, onde n é o somatório do número de clientes $[1...c]$, com a quantidade de divisores de rotas $[c + 1...n]$. Os divisores de rotas são representados pela quantidade de veículos, com um veículo subtraído, visto que as extremidades do cromossomo têm acesso virtual ao nó de origem. As rotas são obtidas a partir das subsequências obtidas da separação desses divisores. A Figura 4 ilustra a representação de rotas para o CVRP.

Figura 4 – Exemplo de representação para o CVRP usando permutação.



Fonte: Elaborado pelo autor (2023).

2.2.3 Inicialização

Inicializar a população requer a definição do seu tamanho e de um método. Segundo Reeves e Rowe (2002), a escolha do tamanho da população precisa levar em consideração a combinação entre eficiência e eficácia. Utilizar uma população muito pequena não permite explorar com eficácia o espaço de busca, e em contrapartida uma população grande não geraria um algoritmo eficiente, uma vez que o processo computacional é mais custoso.

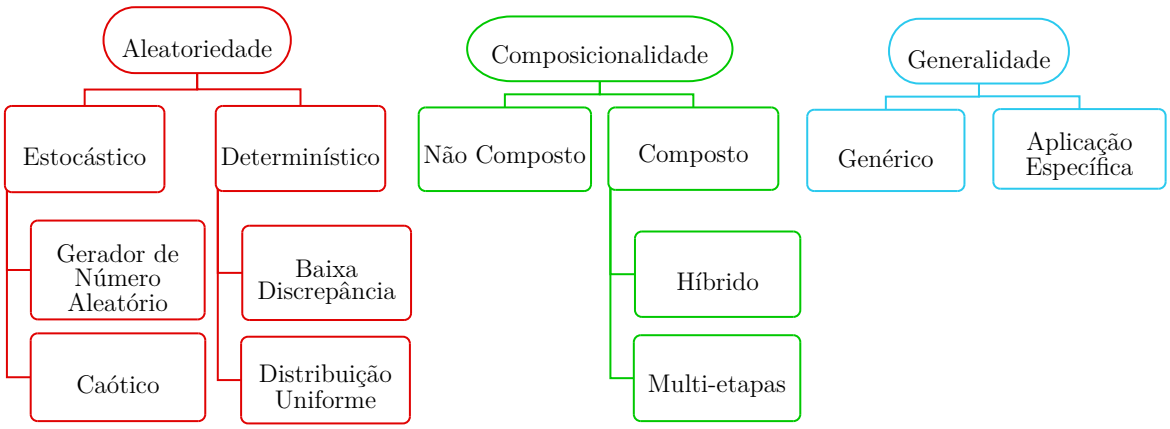
Vrajitoru (2000) explorou em seu trabalho um comparativo entre a relação do tamanho da população contra a quantidade de gerações, visando um balanço entre os dois parâmetros. Segundo o autor, para problemas com espaço de busca grande os resultados são mais promissores se partem da perspectiva de populações maiores, em relação a utilização de mais gerações.

A inicialização tradicional é através de geração de uma população pseudoaleatória. Entretanto, outros tipos de abordagens também podem ser encontradas na literatura. A utilização de estratégias de distribuição podem cobrir uma área melhor do espaço de busca.

Para isso, pode ser útil a utilização de conhecimento específico do domínio (SASTRY; GOLDBERG; KENDALL, 2005), bem como a exploração de heurísticas de inicialização. A Figura 5 mostra as principais abordagens de inicialização de populações para o AG.

Um estudo mais completo acerca dos métodos podem ser consultados em Kazimipour, Li e Qin (2014).

Figura 5 – Categorização dos métodos de inicialização.



Fonte: Adaptado de (KAZIMIPOUR; LI; QIN, 2014).

2.2.4 Seleção

A seguir, apresentamos alguns critérios utilizados para a seleção de indivíduos durante a fase de seleção. Outros métodos podem ser consultados nas revisões realizadas por Goldberg e Deb (1991) e Shukla, Pandey e Mehrotra (2015).

A seleção **aleatória** é formato mais simples que pode ser explorado. Extrai-se da população de entrada um conjunto n de indivíduos necessários para o próximo passo dos AG. Todos os indivíduos da família possuem a mesma probabilidade de serem escolhidos, através de sorteios.

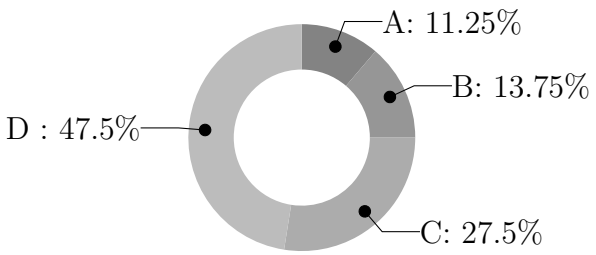
O método da **roleta proporcional** é outra forma tradicional de seleção que utiliza a ideia de sorteio, dando uma proporção a cada indivíduo da população de entrada. A forma proporcional tem uma maior aproximação ao conceito de evolução, onde a distribuição de probabilidades são realizadas de acordo com seus valores de aptidão, tornando os mais fortes mais propensos a serem sorteados, na medida em que os menos aptos terão menores possibilidades de serem escolhidos para os próximos operadores.

Um esquema de implementação é proposto por (SASTRY; GOLDBERG; KENDALL, 2005) a seguir:

1. Avaliar a aptidão de cada indivíduo da população.

Figura 6 – Representação do método da roleta proporcional.

Indivíduo	Fitness (Frequência Absoluta)	Frequência Relativa (%)
A	45	11,25
B	55	13,75
C	110	27,50
D	190	47,50
Total	400	100,0



Fonte: Elaborado pelo autor (2022).

2. Calcular a probabilidade p_i, de selecionar cada membro da população: $p_i = f_i / \sum_{j=1}^{n} f_j$, onde n é o tamanho da população.

3. Calcular a probabilidade cumulativa, q_i, para cara indivíduo: $q_i = \sum_{j=1}^{i} p_j$.

4. Gerar um número aleatório uniforme, $r \epsilon (0, 1]$

5. Se $r < q_1$, selecione o primeiro cromossomo, caso contrário, selecione o indivíduo x_i tal que $q_{i-1} < r \leq q_i$.

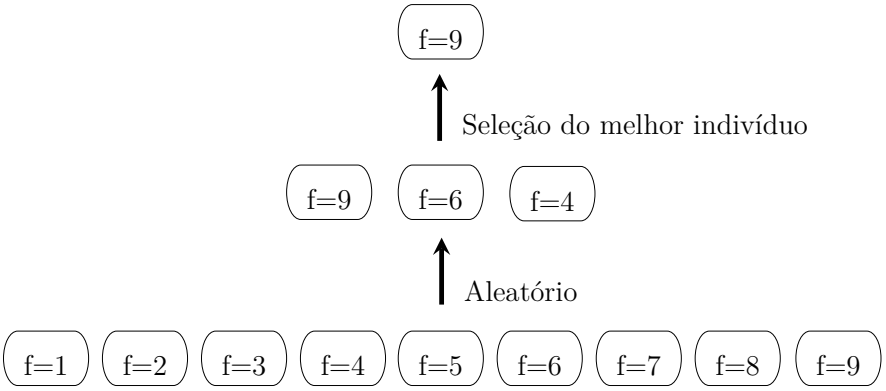
6. Repetir os passos 4 e 5 n vezes para criar n candidatos para o cruzamento.

Sua forma de busca tradicional possui complexidade $\mathcal{O}(n^2)$, entretando é possível uma abordagem com uso de busca binária para sua redução, alcançando nível logarítmico da ordem $\mathcal{O}(n \log n)$ (GOLDBERG; DEB, 1991). Outros métodos, como por classificação linear e classificação exponencial são abordagens disponíveis na literatura que visam alterar a dinâmica do método da roleta para lidar com as desvantagens de aptidão.

Seleção por **torneio** é realizada através da competição de um grupo n de indivíduos sorteados aleatoriamente, ou, selecionados seguindo algum outro critério. Desse grupo, forma-se confrontos entre dois (torneio binário) ou mais indivíduos. Do torneio, deseja-se alcançar um determinado subconjunto resultante, cujos vencedores entram na lista dos indivíduos selecionados para a próxima etapa do algoritmo genético. É realizado uma quantidade de torneios equivalente ao tamanho da população desejada de indivíduos, ou

um único torneio cuja a quantidade seja seu resultante. O critério tradicional de escolha de vencedor de embates é a melhor aptidão do indivíduo. A Figura 7 ilustra esse processo.

Figura 7 – Representação da seleção de um indivíduo através de torneio.



Fonte: Adaptado de (SHUKLA; PANDEY; MEHROTRA, 2015).

O método por torneio é uma estratégia com maior eficiência computacional do que as demais técnicas (desconsiderando o método aleatório), principalmente levando em consideração que não há a necessidade de classificar os indivíduos, possuindo complexidade $\mathcal{O}(n)$. Além disso, sua implementação é fácil, dado que os indivíduos podem ser comparados utilizando a função de aptidão do algoritmo, e permite avaliação com paralelismo (GOLDBERG; DEB, 1991; SHUKLA; PANDEY; MEHROTRA, 2015).

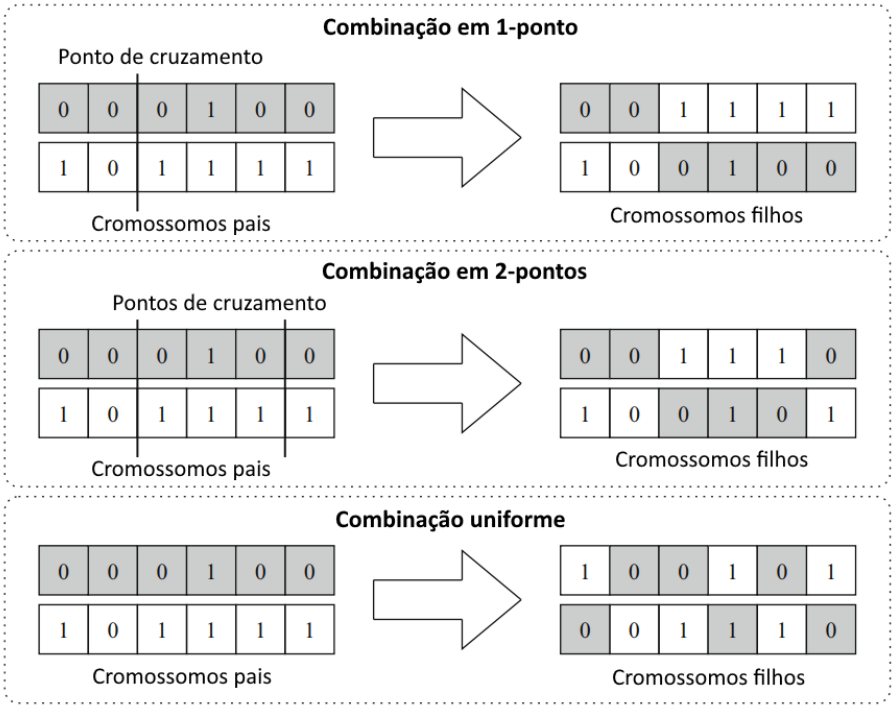
2.2.5 Cruzamento

O Cruzamento é o operador genético responsável por mesclar partes de dois ou mais indivíduos distintos, criando novos que mesclem suas características. As técnicas mais tradicionais de cruzamento são a uniforme e a por pontos (k-pontos).

No **cruzamento uniforme**, posição a posição, cada alelo pode ser trocado entre os filhos, dada uma certa probabilidade do evento ocorrer. Já no **cruzamento por k-pontos**, pares de cromossomos pais são combinados para a formação de um ou mais filhos. k pontos com o tamanho limite da *string* que representa o cromossomo, são escolhidos dos pais. Pode ser realizado de forma aleatória ou com posições pré-estabelecidas. Os alelos selecionados dos pais são concatenados nas posições selecionadas, gerando filhos. Esta técnica geralmente é adotada com uso de 1 ou 2 pontos dos pais, e culminam na geração de 2 filhos. A ilustração desse processo pode ser visto na Figura 8.

Para representações não lineares, algumas adaptações aos métodos de combinação precisam ser reinterpretados (REEVES; ROWE, 2002). No sistema binário, trocar bits significa alterar um valor que está dentro do escopo da representação. Quando há uso de valores inteiros para representar uma rota destino do problema do caixeiro viajante, impossibilita

Figura 8 – Ilustração da combinação em k-pontos e da combinação uniforme.

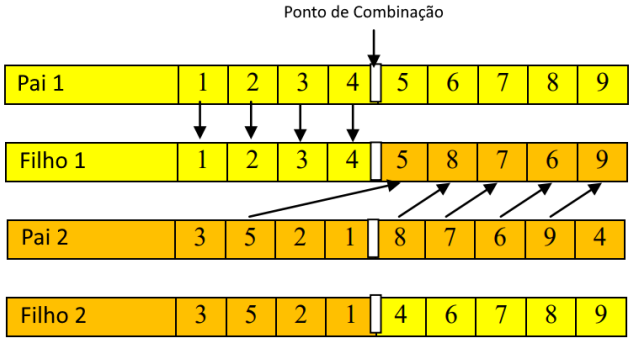


Fonte: Adaptado de (SASTRY; GOLDBERG; KENDALL, 2005).

que uma operação de troca k-pontos seja executada, pois poderá gerar cromossomos com clientes duplicados, fora de escopo.

Uma abordagem não linear em 1-ponto pode ser resumida em 4 passos: (1) Seleção aleatória de um ponto de corte entre dois genes em cada cromossomo pai, resultando em substrings antes e depois desse ponto. (2) Copiar a primeira substring do Pai 1 e inseri-la nas posições apropriadas do Filho 1. (3) Em seguida, copiar individualmente os genes do Pai 2, evitando duplicatas, e inseri-los no Filho 1. (4) Trocar os papéis dos pais para gerar o Filho 2. A Figura 9 exemplifica essa abordagem.

Figura 9 – Ilustração da combinação em k-pontos com representação não linear.



Fonte: Adaptado de (KUMAR; PANNEERSELVAM, 2017).

Para instâncias representadas com permutação de valores, outros métodos se destacam: *order crossover* e *partially-mapped crossover* (PMX). O PMX tem como princípio criar

filhos que preservem ordem e posição da subsequência de um dos pais, adaptando o outro progenitor, a fim de manter boa parte da ordem e posição (FOX; MCMAHON, 1991). Se baseia na técnica de recombinação em dois pontos, onde duas posições aleatórias dos indivíduos são selecionadas, gerando subsequências (substring) de dois parentes. Essas sequências são trocadas, com o objetivo de formar a identidade dos filhos. Suas etapas são descritas por (LARRAÑAGA et al., 1999 apud GOLDBERG; LINGLE, 1985), que está representada na Figura 10, e descritas a seguir:

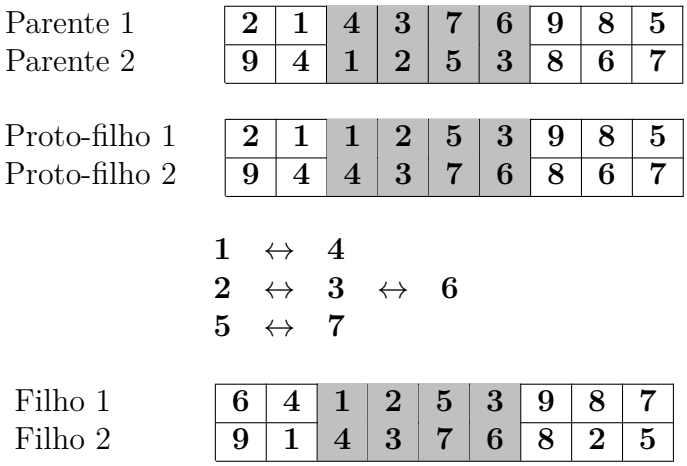
- Selecionar o intervalo: seleciona aleatoriamente dois pontos que representam duas posições ao longo do cromossomo dos pais, formando uma seção de mapeamento selecionada.

- Trocar as subsequências: é gerado duas cópias idênticas dos pais, chamado de proto-filhos. Após isso, a seção selecionada pela etapa anterior dos parentes 1 e 2 são trocados nos filhos.

- Mapear relacionamentos: Ao realizar a troca, é possível que indivíduos inválidos sejam gerados, possuindo mais de uma referência do mesmo nó. Para resolver esse problema, o PMX trata do mapeamento dos nós substituídos nessa etapa.

- Reparar os filhos: Após mapear a relação dos nós substituídos, resolve-se o problema dos genes duplicados fazendo a troca pelo que foi mapeado na etapa anterior.

Figura 10 – Exemplo da Aplicação do PMX.



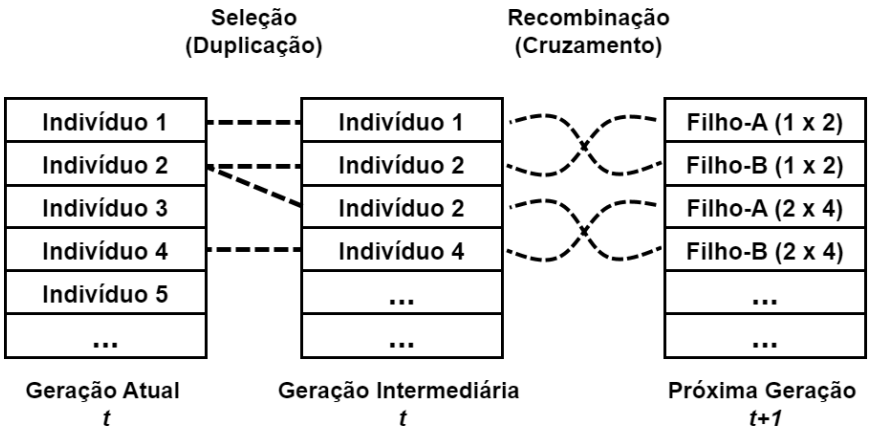
Fonte: Adaptado de (LARRAÑAGA et al., 1999).

Outros exemplos da reprodução em k-pontos, além de outros métodos, podem ser encontrados em (KUMAR; PANNEERSELVAM, 2017).

2.2.6 Mutação

A Figura 11 ilustra como o processo ocorre tipicamente até esse ponto. Após realizar a seleção e a recombinação, uma geração de filhos é obtida.

Figura 11 – Visão geral do processamento da cadeia de indivíduos.



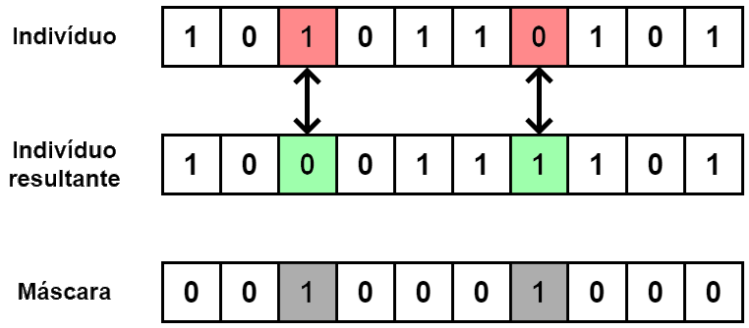
Fonte: Adaptado de (WHITLEY, 1994).

Uma das limitações do operador de cruzamento é a baixa diversidade da população quando o algoritmo começa a convergir para uma solução local, chamada de homogeneidade. Assim como na natureza, as melhores soluções tendem a sobreviver no ambiente de competição, afunilando o processo de busca local. A tendência é de que durante a execução do operador de cruzamento, os pais serão mais parecidos, o que poderá reduzir sua diversidade. Para solucionar esse problema, existe a operação de mutação. Seu fundamento é agregar diversidade, permitindo um melhor monitoramento do espaço de busca, introduzindo outro formato de aleatoriedade (SASTRY; GOLDBERG; KENDALL, 2005; MIRJALILI, 2019). Encontrar o equilíbrio nos parâmetros de cruzamento e mutação são desafios com pouco direcionamento (REEVES; ROWE, 2002).

Mutação por inversão é a metodologia mais citada na literatura. Nela, um ou mais genes são escolhidos aleatoriamente, e tem seus valores invertidos. Baixas taxas de probabilidade costumam ser usadas durante a escolha dos genes a sofrerem inversão. É uma abordagem conveniente para a representação binária, uma vez que a mudança nos bits do cromossomo ainda continua representando uma solução dentro do escopo do problema. A Figura 12 ilustra a aplicação da inversão com variáveis binárias.

Para a abordagem com representação numérica, destacam-se a mutação por troca e por inserção. Ambas se baseiam na seleção de 2 posições do cromossomo. Na mutação por troca, o valor é retirado da primeira, e é trocado com a segunda. Na seleção por inserção, retira-se a primeira posição, e insere esse valor entre a posição da segunda e a próxima (se houver), empurrando uma parte do cromossomo.

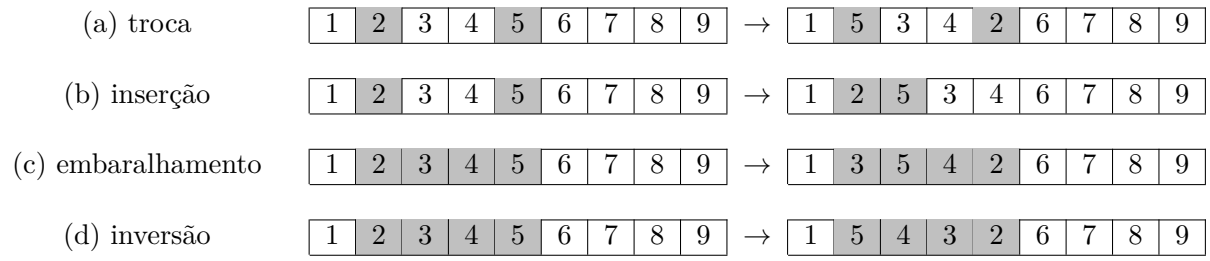
Figura 12 – Exemplo da mutação por inversão baseado na representação binária.



Fonte: Elaborado pelo autor (2022).

Para indivíduos representados por permutação, algumas opções são apresentadas em Eiben e Smith (2015): (a) mutação por troca, método onde dois alelos aleatórios diferentes são trocados no cromossomo; (b) mutação por inserção, onde dois alelos diferentes e aleatoriamente selecionados são reposicionados um ao lado do outro, movendo o segundo para o primeiro; (c) mutação por embaralhamento, onde um subconjunto do cromossomo é selecionado e embaralhado; (d) mutação por inversão, onde um subconjunto do cromossomo é invertido. A Figura 13 exemplifica esses métodos aplicados.

Figura 13 – Exemplo dos métodos de mutação usados.



Fonte: Adaptado de (EIBEN; SMITH, 2015).

2.2.7 Nova Geração

Após a execução do processo de mutação, o AG tradicional demanda que dentre a geração anterior e a nova, o parâmetro da quantidade de indivíduos a ser mantido deve ser respeitado. Para isso, um parâmetro k representa um número de seleções da nova geração a serem escolhidas, onde os melhores substituem os piores da geração anterior, formando uma população com indivíduos de ambas.

Elitismo consiste em manter a porção com melhor aptidão entre as gerações subsequentes. Nela, junta-se a geração t com $t+1$, e retira-se a quantidade excedente do tamanho original da população.

Pode ser usado, também, o domínio do problema para a construção de critérios visando penalizar soluções que possam se estagnar em um espaço de solução não ótimo.

2.3 ALGORITMOS GENÉTICOS PARALELOS

Problemas de otimização de maior escala e/ou maior complexidade demandam um aumento significativo na carga computacional, o que, por sua vez, pode resultar em uma execução mais demorada de sistemas genéticos. Diversos fatores podem requerer um elevado poder de processamento em AGs: problemas com alta dimensionalidade, granularidade reduzida, a utilização de populações extensas, a presença de múltiplas restrições não lineares, funções de aptidão intrinsecamente complexas, a manipulação de grandes conjuntos de dados, a necessidade de aplicar operadores personalizados, entre outros.

Além disso, na abordagem convencional, a busca pode frequentemente se encontrar em um estágio de convergência prematura. Esses sistemas devem equilibrar cuidadosamente a necessidade de explorar amplamente o espaço de busca com o custo computacional da pesquisa realizada em torno das potenciais soluções.

Usar implementações paralelas é uma alternativa que se demonstra promissora para contornar os desafios da complexidade computacional e da convergência prematura (GOODMAN; PUNCH; LIN, 1994) (ADAMIDIS; GREECE, 1995) (CANTÚ-PAZ, 1998). Seu princípio é dividir uma tarefa em partes, e resolvê-las simultaneamente utilizando múltiplos processos, permitindo tanto que funções dispendiosas sejam paralelizadas, quanto que o espaço de busca possa ser melhor explorado. Algumas outras vantagens podem ser obtidas com o seu uso, conforme Tabela 3.

Tabela 3 – Vantagens de usar algoritmos genéticos paralelos

Vantagens de usar algoritmos genéticos paralelos
Trabalha sobre uma codificação do problema (menos restritiva)
Independente do problema (robustez)
Pode produzir soluções alternativas
Pesquisa paralela a partir de múltiplos pontos no espaço
Fácil paralelização como ilhas ou vizinhanças
Melhor pesquisa, mesmo que não seja utilizado hardware paralelo
Maior eficiência e eficácia do que os AGs sequenciais
Fácil cooperação com outros procedimentos de pesquisa

Fonte: Adaptado de (ALBA; TROYA, 1999)

Os AGs são naturalmente habilitados para processamento paralelo, tornando-o mais um degrau próximo daquilo que a biologia interpreta a natureza, expresso em um modelo artificial. Porém, há um desafio ao abordar o paralelismo no contexto das meta-heurísticas, que é a forte presença de processos sequenciais com interdependências significativas, seja do ponto de vista dos dados ou das funções envolvidas (CRAINIC; TOULOUSE, 2003). Sua implementação é fortemente ligada a adoção de estratégias que visem superar tal dependência. Segundo o autor, essas estratégias podem partir do ponto de vista de um paralelismo de baixo nível ou da execução paralela de partes ou subpartes do AG.

No contexto do paralelismo de baixo nível, o objetivo é otimizar os cálculos de uma interação do método heurístico ao executar operações ou avaliações de forma concorrente. Isso é feito focando especificamente naquelas tarefas que podem ser realizadas simultaneamente, sem interromper o progresso da heurística. Seu desempenho está fortemente associado ao conceito de *speedup*. O *speedup* é uma métrica usada para relacionar o ganho de desempenho na execução de um programa utilizando um processador (serial) e vários (paralelo) (SCHMIDT et al., 2018).

Já no contexto da execução paralela que atinge o nível da arquitetura da heurística, destaca-se na literatura uma divisão baseada em: modelo global, modelo granular fino (*fine grained*) e modelo granular grosso (*coarse grained*) (CANTÚ-PAZ, 1998) (HARADA; ALBA, 2020):

- A abordagem de **modelo global** utiliza técnicas para acelerar as operações paralelizáveis de um AG baseado em uma população global. Seu foco é manter a estrutura sequencial de um AG clássico. O formato mestre-escravo (do inglês, *master-slave*) é o mais tradicional nesse tipo de modelo. Se baseia na divisão em dois níveis. O nível mestre, que é responsável por controlar a comunicação, classificação e emparelhamento de um conjunto de soluções provisórias (população). Cada solução é avaliada paralelamente pelo nível dos escravos (HAUPT; HAUPT, 2004). Nesse modelo, a seleção e o cruzamento são mantidos pelo mestre. A mutação e partes do cruzamento podem ser tratadas nos escravos.

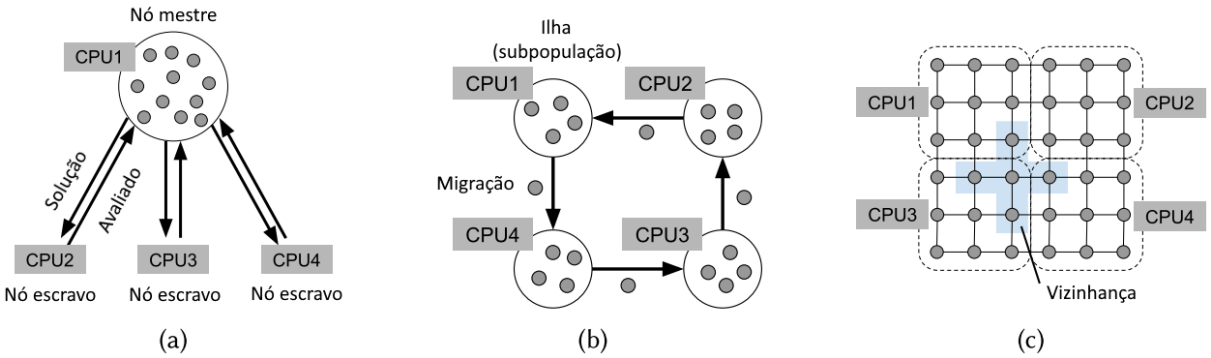
- A abordagem de **granularidade grossa** tem seu foco nas políticas de evolução, particionamento de variáveis e busca independente. Nessa estratégia, a população é dividida em subconjuntos particionado, cada um evoluindo de forma paralela e implementando algum mecanismo de migração de indivíduos entre esses subgrupos. A busca independente se baseia em múltiplas explorações simultâneas, com uso de *threads* ou sistemas multiprocessados integrados. Podem partir de soluções iguais ou diferentes, bem como cooperarem através de comunicação dinâmica. O modelo de ilhas é a aplicação mais conhecida para essa granularidade, que será abordado em 2.3.2.

- A abordagem de **granularidade fina**, em contraste com a granularidade grossa, concentra-se em um nível mais individual, analisando indivíduos e suas interações em grande detalhe. Essa abordagem envolve o gerenciamento de uma grande quantidade de indivíduos e, portanto, requer uma infraestrutura massivamente paralela para suportá-la. As interações são estritamente reguladas pela topologia das regiões em que esses indivíduos estão inseridos. Um dos principais desafios dessa abordagem é lidar com o *overhead* causado pela necessidade de comunicação excessiva para permitir a migração entre as diferentes subpopulações. A topologia comum para essa abordagem

é geralmente implementada em forma de grade ou malha. Um exemplo notável de aplicação desse modelo é o modelo celular, que explora essas interações individuais com grande detalhe.

A Figura 14 demonstra o exemplo dos métodos mais usuais de PGA.

Figura 14 – Modelos de Algoritmos Genéticos Paralelos usuais.



Fonte: Adaptado de (HARADA; ALBA, 2020).
Nota: (a) modelo mestre-escravo, (b) modelo de ilhas, (c) modelo celular

Vale ressaltar a relevância no desenvolvimento híbrido que combinam elementos dessas diferentes abordagens. Esses modelos são amplamente discutidos na literatura e representam estratégias que permitem a integração de outras técnicas de resolução de problemas, como a aplicação de métodos exatos em subproblemas e a fusão de heurísticas, entre outras abordagens complementares.

Além da programação paralela e dos modelos teóricos, a arquitetura de *hardware* representa outro elemento fundamental no desenvolvimento de sistemas paralelos no contexto. Na arquitetura de hardware, dois aspectos críticos que merecem destaque são a configuração do sistema multiprocessado e o gerenciamento de memória (SCHMIDT et al., 2018). Na configuração do sistema multiprocessado, destacam-se os processadores multinúcleo (CPU *multicores*), unidades de processamento gráfico (GPU) e *clusters*. Em relação ao gerenciamento de memória, pode ser realizado com memória distribuída ou memória compartilhada. A combinação dessas configurações de hardware está intrinsecamente ligada ao modelo de programação paralela escolhido e seus requisitos de desempenho, determinando como as tarefas são divididas e como os recursos de processamento são alocados.

2.3.1 Tipos de Sincronização

Outro ponto-chave do paralelismo se destaca o tipo de sincronização de gerações. No AG clássico, a geração corrente é respeitada de acordo com seu modelo sequencial. Nas abordagens paralelizadas, uma parte do algoritmo eventualmente está sendo concluído

à frente das outras, o que pode se diferir do modelo sequencial. Nesse sentido, há duas perspectivas:

- No síncrono (ou abordagem geracional) o conceito de geração da população global é respeitado. Nesse sentido, é realizada através de alguma tática onde a transferência de indivíduos são interpretadas regulares, ocorrendo ao mesmo tempo (RUCIŃSKI; IZZO; BISCANI, 2010).

- No assíncrono (ou estado estacionário) não há uma dependência entre as abstrações de processamento no sentido de uma geração global. É uma abordagem um pouco diferente da tradicional, pois há uma mesclagem contínua entre soluções novas e antigas.

Em ambientes síncronos, a utilização da computação distribuída pode resultar em custos significativos, o que, por sua vez, pode impactar negativamente a escalabilidade do sistema (RUCIŃSKI; IZZO; BISCANI, 2010). Além disso, em sistemas que apresentam características heterogêneas, existe o risco de surgirem gargalos de processamento, levando a uma subutilização de recursos em algumas partes do sistema. Segundo Chipperfield e Fleming (1996), sua eficiência depende de um forte equilíbrio entre os custos das partes paralelas e das sequenciais, e ele pode se tornar ineficiente se houver muita variação nas funções de avaliação. A economia do paralelismo vê forte vantagem se o cálculo das funções de aptidão são suficientemente complexos, que justifiquem a criação de instâncias escravas.

Por outro lado, em cenários assíncronos, o monitoramento do estado global do algoritmo torna-se uma tarefa mais complexa e imprevisível. Nesse contexto, a coordenação entre os processos ou *threads* pode ser desafiadora, uma vez que não há um controle global rígido sobre a ordem de execução das tarefas. Sua diferenciação é que produz uma convergência mais rápida em relação ao modelo tradicional panmítico (HARADA; ALBA, 2020).

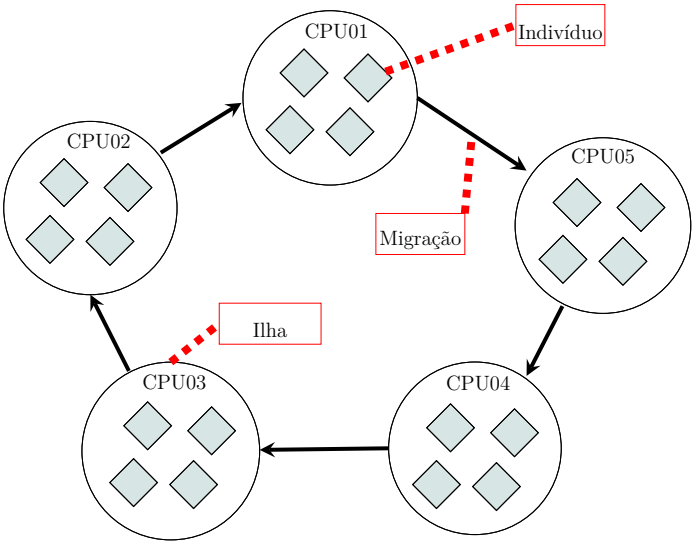
2.3.2 Modelo de Ilhas

Geograficamente os indivíduos na natureza estão separados em grupos, por motivos diversos e em contextos diferentes. O modelo de ilhas utiliza esse fator como inspiração na construção de soluções. Tem como características a utilização de múltiplas populações e a migração genética (CANTÚ-PAZ, 2000).

Nele, a população é dividida em várias subpopulações, chamadas de ilhas. Cada ilha executa uma heurística de forma independente, evoluindo suas próprias populações. Diferentemente do modelo mestre-escravo ou do AG tradicional, que possuem uma única população centralizada, esse modelo pressupõe uma colaboração entre domos que evoluem sob suas condições internas e trocam informações genéticas de forma periódica. Técnica que se

demonstra, segundo (ALBA; TROYA, 1999) e (CANTÚ-PAZ; GOLDBERG, 2000), mais eficiente e robusta em relação ao AG padrão.

Figura 15 – Exemplo de modelo de ilha.



Fonte: Elaborado pelo autor (2023).
Nota: A Figura demonstra um exemplo de cinco ilhas alocadas em CPUs distintas que desenvolvem seus indivíduos e realizam a migração circular, com topologia no padrão anelar (*ring*) unidirecional.

Inicialmente foi uma técnica voltada para o desenvolvimento de algoritmos genéticos, porém diversos trabalhos demonstram que é uma técnica capaz de absorver outras heurísticas. Duarte, Lemonge e Goliatt (2017), por exemplo, explora utilizando implementações com 5 diferentes técnicas em cada ilha: AG, colônia de formigas, otimização por enxame de partículas, colônia de abelhas, evolução diferencial e *social spider algorithm*, demonstrando a versatilidade que o método possui.

A migração é o principal operador do modelo, garantindo que haja uma pressão global, buscando contornar o fenômeno da convergência prematura em cada domínio. Conforme Cantú-Paz (2000), sua implementação depende da difícil definição dos seguintes parâmetros: a quantidade de indivíduos e o número de ilhas (2.3.2.1), a topologia que interliga as ilhas (2.3.2.2), a taxa de migração que controla quantos indivíduos são enviados e a frequência com que as migrações ocorrem (2.3.2.3), e a política de migração que determina quais indivíduos migram e quais são substituídos na ilha receptor (2.3.2.4).

2.3.2.1 Número e tamanho das ilhas

O número de ilhas é escolhido dependendo da arquitetura e modelo adotado. O tamanho de uma ilha é determinado pela quantidade de indivíduos alocados. Algumas abordagens permitem que esses valores sejam dinâmicos ao longo da evolução.

Em sistemas síncronos, visa-se minimizar qualquer desbalanço que possa ocasionar gargalos de processamento. Esse desbalanço pode ser ocasionado pela heterogeneidade de sistemas, de um desequilíbrio de processamento oriundo de trechos do algoritmo, ou, do tamanho das populações. Já nos assíncronos, construir mecanismos que permitam que a quantidade de gerações concorrentes estejam o mais próximo possível da paridade de processamento. Isso evita que um subconjunto esteja significativamente à frente dos outros em termos de progresso na quantidade de gerações.

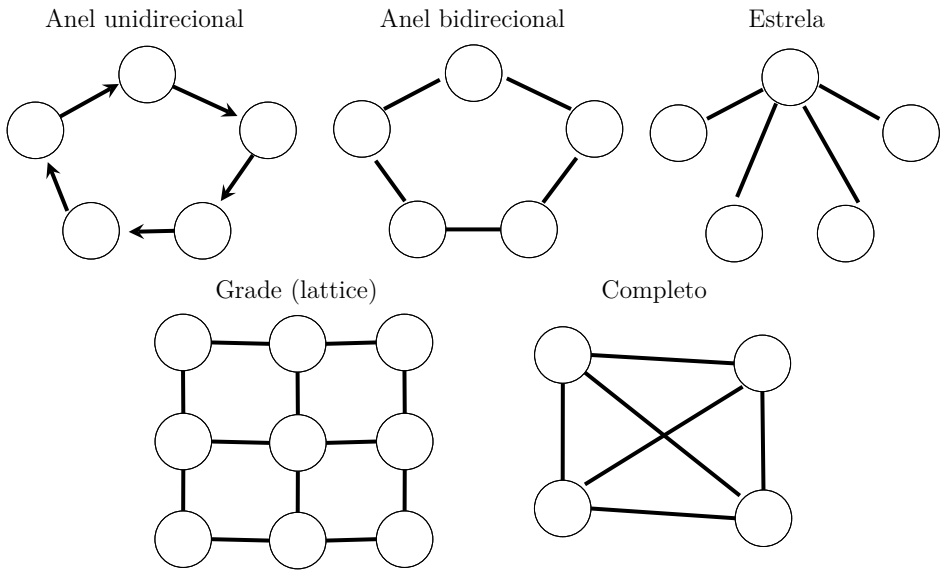
Fernandez, Tomassini e Vanneschi (2003) realizou um trabalho comparativo com uso de *benchmark*, demonstrando que para cada tipo de problema existe um ponto de corte onde o aumento da população se demonstra ineficiente ou prejudicial.

2.3.2.2 Topologia de migração

A topologia de migração determina os caminhos viáveis de migração entre as ilhas. Pode ser expressas como as arestas em um grafo, que representa suas conexões. Nessas conexões, se formam fluxos de troca de indivíduos. Esses fluxos, podem ser estáticos ou dinâmicos, variando de acordo com a implementação desejada.

Diversas topologias são encontradas na literatura. No estudo realizado por Ruciński, Izzo e Biscani (2010) é possível localizar a maioria dos formatos estáticos presentes na literatura. Também pode ser consultado (CANTÚ-PAZ, 2000) para mais detalhamento. Os mais tradicionais são os formatos em anel (unidirecional e bidirecional), estrela, grade (*lattice*) e grafo completo (*full connected*).

Figura 16 – Exemplos de topologias.



Fonte: Elaborado pelo autor (2023).

Em termos gerais, busca-se uma topologia que mantenha a diversidade genética, ao mesmo tempo em que permita a troca de material genético para promover o progresso em nível global. Essa configuração visa encontrar um equilíbrio entre a exploração de novas soluções e a preservação da variabilidade genética.

Alba e Troya (1999) cita que topologias muito conexas ou muito centralizadas, exemplo de completa e estrela respectivamente, apresentam problemas de paralelização e escalabilidade na medida em que dependem de uma conexão estreitada.

Tang et al. (2004) comparou a utilização de uma topologia aleatória com uma topologia em anel, em um problema quadrático de alocação de larga escala. Verificou que a utilização da topologia em anel se demonstrou mais adequada para equilibrar exploração e convergência em relação ao aleatório. Ruciński, Izzo e Biscani (2010) desenvolveu um estudo comparativo com as topologias tradicionais em diferentes problemas de otimização com diferentes base de algoritmos, demonstrando que a topologia afeta a qualidade das soluções e o ritmo da convergência consideravelmente. Algumas técnicas se beneficiaram mais e menos a depender do algoritmo escolhido, em muitos casos o excesso de arestas (grafo completo) torna a convergência prematura.

As topologias dinâmicas generalizam os parâmetros básicos do modelo de ilhas, a um nível de autoadaptação capaz de serem reajustados a cada ciclo migratório (LARDEUX; GOËFFON, 2010). Isso pode incluir estratégias diversas que envolve tanto a forma, a quantidade e frequência de migração, quanto nos algoritmos e população dentro das próprias ilhas. O autor cita que conceitos como a probabilidade de migração podem ser ajustados baseado em recompensa e penalidade, a partir nos níveis de melhora a cada interação.

2.3.2.3 Taxa e frequência de migração

Frequência de migração (ou intervalo de migração) é um parâmetro que determina períodos sucessivos em que trocas entre as subpopulações são realizadas. Tipicamente, são realizadas em intervalos bem determinados ou com uso de probabilidade de ocorrência (ALBA; TROYA, 1999).

Já a taxa de migração, é um parâmetro que determina o número de indivíduos que migra em cada interação da frequência de migração (ALBA; TROYA, 1999).

Ambos os parâmetros estão diretamente relacionados ao formato e características do algoritmo paralelo (RUCIŃSKI; IZZO; BISCANI, 2010).

Experimentos realizados por Skolicki e Jong (2005) demonstraram que o modelo se beneficia da diversidade genética. Essa diversidade é degradada quando há uma queda de

desempenho, por pelo menos três fatores: (1) quando a taxa de migração é muito alta; (2) quando a frequência é alta; (3) e quando a frequência é muito baixa. O autor cita que o modelo depende de pequenas migrações em frequência mediana.

Além de taxas e intervalos fixos, alguns estudos tratam ambos os parâmetros dinamicamente. Mambrini e Sudholt (2014) explorou uma adaptação do intervalo de migração baseado no índice de melhoria da aptidão, onde o parâmetro é reajustado quando as interações internas das ilhas não encontram melhorias. O autor sugeriu que sua abordagem é capaz de reduzir a comunicação sem degradar os resultados obtidos utilizando intervalos fixos. Em Lin et al. (2012), um comparativo entre esquemas migratórios sob o problema da mochila binária, usando uma abordagem de migração adaptativa. Explorou ajustes dinâmicos na taxa e frequência de migração, e uma abordagem de ajuste a partir da comparação da diversidade entre as subpopulações. Em seus resultados, todos os experimentos dinâmicos demonstraram mais eficazes em relação ao uso de valores fixos.

2.3.2.4 Políticas de migração

A migração também pode ser implementada de forma dinâmica e ajustável, com o objetivo de melhorar o desempenho. Lardeux et al. (2019) em seu trabalho demonstrou tal flexibilidade ao ajustar as políticas migratórias ao nível dos indíviduos, demonstrando que é possível que eles cooperem dinamicamente se ajustados com algum mecanismo de aprendizagem de máquina.

2.4 TRABALHOS CORRELATOS

Abbasi et al. (2020) propõe um projeto de AG paralelo para a solução do Problema do Caixeiro Viajante (PCV) baseado na construção do processamento em *kernels*, em que parte pudesse ser executado concorrentemente. No seu trabalho, realizou um comparativo da execução da solução com uso de sistemas de processador único (cpu), multiprocessador (*multi-core*) e processadores com muitos núcleos (*many-core*). Como resultado, demonstrou que a paralelização reduz o tempo de processamento para populações grandes e para uma grande quantidade de cidades, quanto maiores, mais vantagens os sistemas com paralelismo se apresentaram. Também, que os AG são afetados pela eficiência do escalonamento de recursos de hardware, que aumentou no comparativo com um único núcleo e em relação à concorrência com uso de *threads*, onde os *many-cores* se destacam dos *multi-cores*.

Borcinova (2018) propôs uma abordagem baseada em micro AG paralelo, com uso de populações pequenas. Como critério para seleção foi utilizado uma versão da heurística de Clarke-Wright. Para seleção, torneio binário entre indivíduo 1x2 e 3x4, formando uma dupla. A operação de combinação foi a *partially-mapped crossover*(PMX). O processo seguiu até que 50 iterações fossem realizadas sem melhoria do melhor indivíduo. Após

as iterações sem melhoria, uma nova época é criada, esse parâmetro foi de 2500 nos testes, realizados com instâncias da literatura. Os resultados alcançados foram os valores ótimos ou de forma útil próximos. O autor sugere melhorar a proposta do algoritmo para redução de tempo computacional, e propõe testes para problemas mais robustos como uma extensão do trabalho.

Dorronsoro et al. (2007) utilizou uma abordagem híbrida para resolver instâncias robustas de *benchmark* do CVRP. No trabalho, implementa os três modelos clássicos de abordagem paralela em uma estrutura baseada em níveis. Uma população descentralizada foi dividida utilizando o modelo de ilhas, em que essas ilhas implementam topologia de vizinhança em anel unidirecional. No nível interno de cada ilha, uma implementação do modelo celular em grade é usada para a evolução das ilhas. Para o processo de busca local de cada ilha o trabalho utiliza o formato mestre-escravo. Testes foram realizados em plataforma em grade com 100 máquinas. A conclusão dos autores cita que para aquele momento, o algoritmo desenvolvido entregava as melhores soluções da literatura para muitas dos casos de testes utilizados.

Algumas estratégias, ainda, focam nas unidades de processamento gráfico (GPU's) como alternativa de *hardware*, como em Abdelatti e Sodhi (2020), que dividiu em *kernels* as etapas de um AG. O autor conseguiu obter resultados de alta qualidade em tempo de processamento computacional razoáveis, quando comparados com o mesmo sistema executado em CPU, para os problemas menores de *benchmarks* do CVRP.

Um exemplo de solução hibrida é apresentado por Ammi e Chikhi (2015). Neste trabalho, foi desenvolvido uma generalização do modelo de ilhas em formato distribuído em arquipélagos, com o uso simultâneo de AG e a meta-heurística Colônia de Formiga. Essa abordagem cooperativa foi aplicada visando solucionar problemas CVRP de larga escala. O resultado deste trabalho foi gerar novos pilares na literatura para alguns dos problemas testados, e gerar resultados competitivos nos demais.

3 DESENVOLVIMENTO

Este capítulo tem como objetivo apresentar a heurística desenvolvida, destacando a partir dos conceitos levantados no referencial teórico, os métodos e os parâmetros configuráveis associados no desenvolvimento de um AG visando a solução do problema, com o uso de mecanismo paralelizável.

Um código foi desenvolvido contendo a heurística proposta, apresentada a seguir. Todos os algoritmos foram construídos na linguagem C, com uso da biblioteca *pthread*. Esse algoritmo foi implementado para ser executado em ambiente Windows, por isso foram compilados com GCC na versão 4.8.1-3, com uso do MinGW para gerar código nativo do sistema. Esses, estão disponibilizados no Github [1], junto com detalhes de compilação e dos testes que serão apresentados no Capítulo 4.

3.1 HEURÍSTICA PROPOSTA

Com base na pesquisa realizada no referencial teórico, optamos por uma abordagem que se baseia no modelo de granularidade grossa, o qual é expresso por meio do conceito de ilhas. A ideia principal por trás dessa escolha é explorar o potencial dos múltiplos núcleos presentes em processadores de computadores e servidores contemporâneos (arquiteturas x86 e x64), através de subprocessos independentes alocados em *threads*. Observamos que, nesse contexto, abstraímos o escalonamento realizado pelo Sistema Operacional. Antes de citar mais detalhes, segue uma visão geral da heurística proposta na Figura 17.

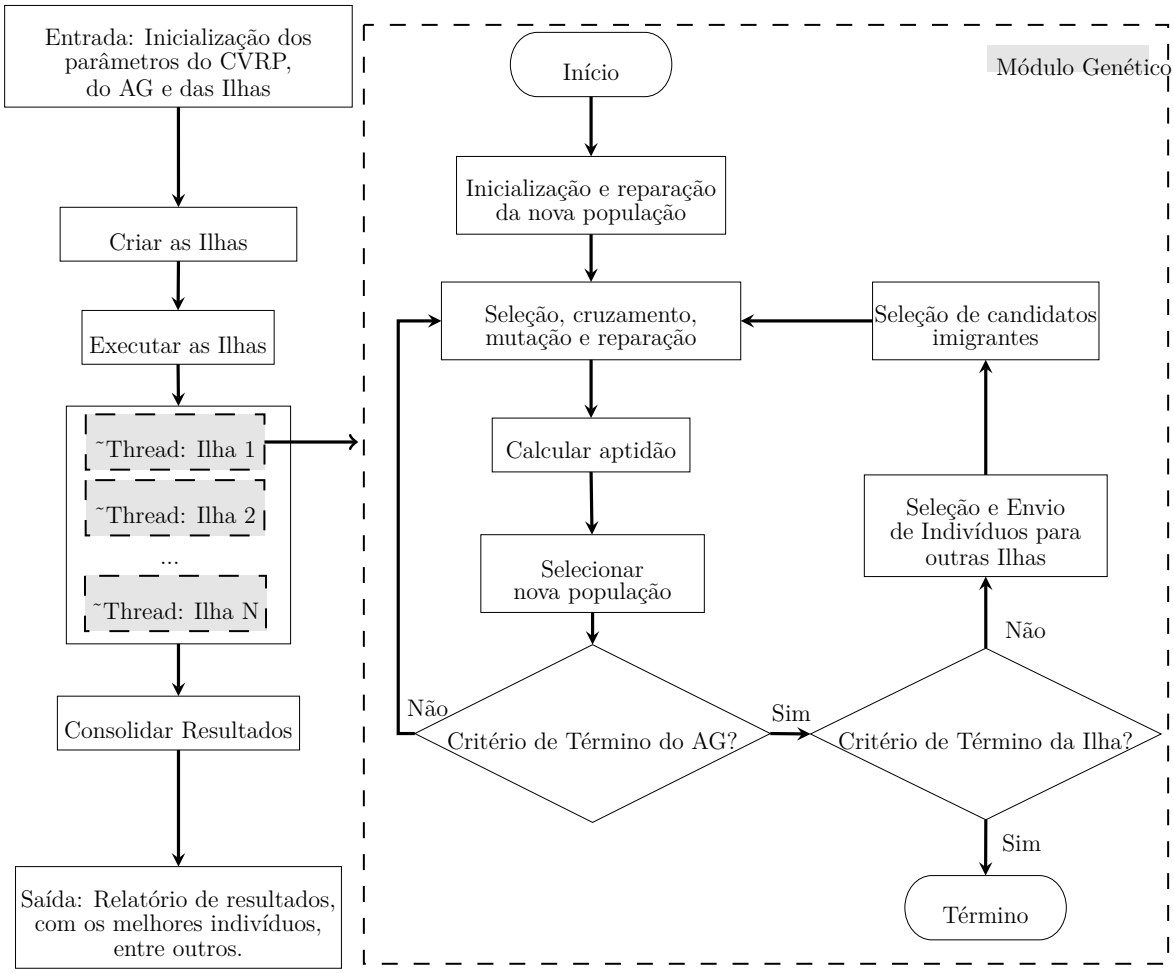
Neste trabalho, um conjunto referente aos parâmetros de um problema CVRP é esperado como entrada para o algoritmo. Uma instância principal (*thread*) constrói e executa instâncias de ilhas em *threads* subordinadas. Espera-se que esse conjunto de ilhas se desenvolva e troque informações genéticas utilizando memória compartilhada e controle de acesso. Após as execuções, definidas na entrada para cada ilha (módulos genéticos), a instância principal fica responsável por gerar os relatórios finais. Relatórios intermediários podem ser gerados pelas próprias ilhas, se necessário.

A topologia escolhida foi o formato em anel estático. Opcionalmente, um parâmetro determina se o formato de envio entre as ilhas é unidirecional ou bidimensional. Quando unidirecional, uma ilha i recebe de $i - 1$ e envia para $i + 1$. Quando bidirecional, tanto o envio quanto o recebimento são feitos para e de ambos os vizinhos.

Em relação às configurações do AG, para a **representação** do cromossomo, optou-se por usar a permutação sequencial dos n nós clientes adicionando divisores de rota, um modelo próximo do proposto por Alba e Dorronsoro (2006). Para a **inicialização** dos indivíduos,

[1] Disponível em: <https://github.com/rraidero/parallel-island-ag-cvrp>. Acesso em: 08 dez, 2023.

Figura 17 – Representação da Visão Geral da Heurística Proposta.



Fonte: Elaborado pelo autor (2023).

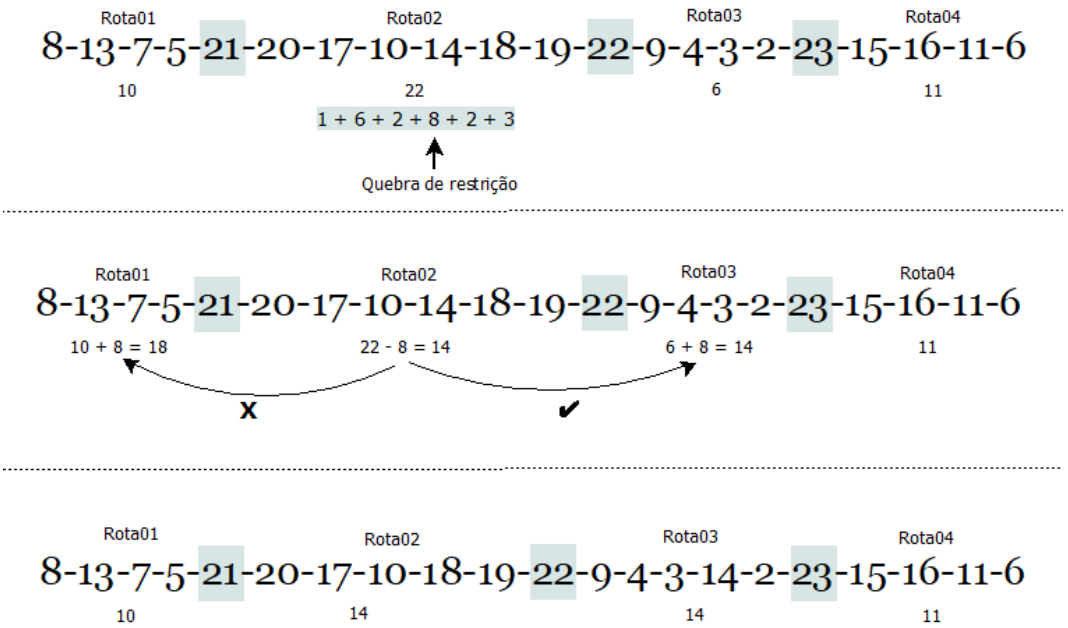
foi adotada o embaralhamento dos cromossomos. Indivíduos gerados sequencialmente são trocados de posição baseado em mecanismo pseudoaleatório. Na **seleção** de indivíduos, foi implementada uma abordagem análoga a apresentada por Abbasi et al. (2020). Um par de cromossomos é selecionado aleatoriamente. Em seguida, um valor dentro do intervalo $(0, 1)$ é escolhido de forma aleatória. Um parâmetro predefinido representa a porcentagem de preferência pelo cromossomo mais apto entre os dois selecionados. Após essa seleção, ambos os cromossomos retornam à população original, e esse processo é repetido quantas vezes forem necessárias para gerar o número adequado de indivíduos que avançarão para a próxima etapa. Para realizar a **combinação**, foi escolhido a abordagem *partially-mapped crossover* (PMX), como pode ser visto em 2.2.5. Foram selecionados 4 métodos de **mutação** para indivíduos representados por permutação, citados por Eiben e Smith (2015): (a) mutação por troca; (b) mutação por inserção; (c) mutação por embaralhamento; (d) mutação por inversão. Esses métodos de mutação podem ser vistos em 2.2.6.

Indivíduos inicializados, combinados e/ou sujeitos a mutações podem gerar soluções que violam a restrição de carga, dependendo do método de representação escolhido. A validação das soluções tornou-se crucial antes de considerá-las úteis. Para abordar esse desafio, optou-se por implementar uma operação de **reparação**, visando corrigir várias soluções inviáveis. Como operador de reparação, visando manter o possível de diversidade genética, foi escolhido a utilização de duas estratégias: reparação por realocação e reparação por destruição.

- Reparo por realocação: envolve a transferência de um indivíduo para outra rota. De maneira iterativa, quando ocorre uma violação de restrição ao exceder a capacidade de uma rota, a solução procura uma rota alternativa que possa acomodar esse indivíduo. Se uma rota alternativa for encontrada, o algoritmo insere o indivíduo nessa rota em uma posição aleatória e continua a busca por outras violações de restrições. A Figura 18 exemplifica um caso de aplicação da realocação, onde uma solução é gerada a partir de uma não válida.

- Reparo por destruição: Situações podem surgir em que a inserção de um nó em qualquer outra rota seja inviável devido a restrições quebradas. Nesse cenário, a abordagem consiste em realocar aleatoriamente o nó problemático e reiniciar o processo de reparação.

Figura 18 – Exemplo de reparação por realocação quando possível.



Fonte: Autoria própria.
Nota: Considerando que o limite de carga dos veículos são 15, é localizado na primeira parte que a Rota02 não atende tal requisito. Para solucionar, na segunda parte, é verificado se há outras rotas em que é possível alocá-la. Como é possível ver na solução gerada, ele inseriu na Rota03, após o nó 3. Isso gerou uma solução que atendeu tal restrição.

O algoritmo de reparação é usado para corrigir as rotas inválidas, antes de juntar as gerações, a corrente e seus filhos. A construção das novas gerações ocorrem partindo de três políticas: (a) um parâmetro que determina a quantidade máxima de indivíduos da nova geração a serem incorporados, (b) os novos indivíduos não podem ser iguais a algum outro presente na população legada, (c) a substituição ocorre sempre levando em consideração o elitismo, onde os piores indivíduos da geração anterior dão lugar aos novos mais aptos.

Já a política de troca de material genético entre as ilhas ocorre em duas fases distintas e assíncronas: (I) fase de envio e (II) fase de recebimento. No envio (I), um parâmetro determina quantos indivíduos são selecionados da elite e entregues nas ilhas que possuem conexão. Em (II), uma população candidata de imigrantes pode estar esperando em uma fila. Quando isso ocorre, sua incorporação acontece de nos padrões 'b' e 'c' da consolidação apresentada anteriormente, respeitando o critério de não repetição de material genético e com enfoque no elitismo. Essa troca ocorre sempre que uma quantidade parametrizada de gerações acontece, em cada ilha. A utilização de um modelo assíncrono não garante trocas equilibradas, porém, também não impede que uma série de ilhas fiquem aguardando outras.

As configurações e métodos implementados nesta etapa de desenvolvimento, levantados neste capítulo, segue resumida na Tabela 4 a seguir.

Tabela 4 – Resumo das configurações implementadas.

	Conceito	**Abordagem Escolhida**
Ilhas	Topologia	Anel (bidirecional ou unidirecional, parametrizado)
	Gerações	Assíncronas
	Política de Envio de Indivíduos	Elitismo (quantidade parametrizada)
	Política de Recebimento de Indivíduos	Elitismo (quantidade parametrizada) *sem incorporar indivíduos repetidos
Algoritmo Genético	Inicialização	Aleatória
	Seleção	Par sorteado com probabilidade de escolha do mais apto (parametrizado)
	Combinação	*Partially-Mapped Crossover (2-pontos)*
	Mutação	Troca, Inserção, Embaralhamento e Inversão (serial de probabilidade parametrizada)
	Reparação	Realocar em outra rota se possível (se não, destruir e reiniciar realocação)
	Novas Gerações	Elitismo (quantidade parametrizada) *sem incorporar indivíduos repetidos

Fonte: Elaborado pelo autor (2023).

4 EXPERIMENTOS

Este capítulo apresenta com mais detalhes o projeto de experimentos construído para testar o algoritmo desenvolvido. Além disso, apresentamos o conjunto de parâmetros associados aos testes. Os resultados desses testes são apresentados posteriormente.

4.1 PROJETO DE EXPERIMENTOS

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

As instâncias podem ser classificadas sob quatro perspectivas: (I) quantidade de clientes, (II) quantidade de veículos, (III) tipo de distribuição dos dados e (IV) e o *tigh*. O tipo de distribuição (III) se refere ao formato de distribuição (espalhamento) dos clientes no espaço geográfico, no geral é possível separá-los como aleatórios ou agrupados (*clustering*). O *tigh* (do inglês *tightness*, em português aperto ou rigidez) é a relação de carga total com o total de veículos. Os dois primeiros afetam diretamente no tamanho do problema. Os demais podem impactar nos resultados de forma diferente, uma vez que dados mais agrupados podem gerar inúmeros filhos (soluções) inválidos, principalmente levando em consideração uma margem mais aperta do *tigh*. Alguns casos de testes serão usados para verificar essa afirmativa.

Usando essa classificação, foram selecionados 24 instâncias de testes organizados em grupos, visando proporcionar um entendimento mais aprofundado e análises específicas dos resultados e a relação com os problemas. A abordagem adotada consiste em realizar 10 testes para cada caso, variando o número de ilhas, contemplando configurações de 1, 2, 4 e 6 ilhas. Esse procedimento totalizou 960 execuções, abrangendo todas as combinações de problemas, repetições e ilhas.

O primeiro grupo (G1), visou ancorar uma quantidade baixa de veículos e distribuição aleatória de clientes. O intuito de grupo é avaliar o desempenho com o crescimento acentuado no número de clientes. No G2 também foi utilizado uma padronização nos veículos, pares aproximados de clientes foram usados variando o formato de distribuição, a fim de verificar se pequenos agrupamentos influenciariam nos resultados.

Em G3, foi fixado a quantidade de clientes, variando principalmente o *tigh*. O principal objetivo desse grupo é partir de problemas aproximados observando o impacto dessa variação nos resultados. Em G4 temos situações em que os clientes são iguais, tanto em quantidade tanto em posicionamento no plano, com variação na quantidade de clientes e no *tigh*. Esse grupo simula um caso em que há uma tomada de decisão com relação ao

número e capacidade dos veículos, supondo uma demanda a ser realizada.

Por fim, G5 foca em problemas maiores, variando consideravelmente as características. O intuito é observar o comportamento da heurística para problemas mais difíceis, tanto no sentido da qualidade das soluções quanto no tempo de execução. A Tabela 5 lista todos os casos selecionados e suas características básicas. A coluna *tigh* refere-se ao conceito de *tightness* (aperto, em português), que é a divisão do somatório de todas as demandas com a quantidade de veículos (FISHER, 1994). Um problema é apertado quando se aproxima de 1, nesse caso todos os veículos irão ser carregados no limite. Quanto menor, mais flexibilidade terá, facilitando a concatenação de clientes.

Tabela 5 – Instâncias de testes.

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

Fonte: Elaborado pelo autor (2023).

4.2 CALIBRAÇÃO DE PARÂMETROS

A calibração dos parâmetros tem como objetivo selecionar um conjunto de parâmetros para a execução dos testes anteriormente citados. No entanto, uma análise detalhada da ampla gama de parâmetros não é o foco deste trabalho. Com o objetivo de restringir o

escopo da análise, optamos por realizar uma calibração que será aplicada uniformemente a todos os grupos de teste.

Testes preliminares foram realizados usando a instância *E-n76-k10* com uso de 4 ilhas. Foram avaliados variações na topologia unidirecional e bidirecional. A quantidade de gerações a serem executadas $[500, ..., 5000]$ ± 250. Na etapa de seleção, o mais apto $[50\%, ..., 90\%]$ $\pm 5\%$. Valores dos métodos de mutação [0.5%, ..., 10%] $\pm$0,5%. Quantidade de indivíduos filhos que poderão ser utilizados na nova geração variando de $[5\%, ..., 100\%]$ $\pm 5\%$. Intervalo de migração em gerações $[100, ..., 300]$ ± 50. E quantidade de imigrantes entre ilhas $[1, ..., 10]$ ± 1.

As configurações escolhidas do AG que irão nortear as execuções foram: tamanho da população 100 com 4500 gerações, seleção com 80% do mais apto entre indivíduos selecionados aleatoriamente, mutação serial de 7% e potencial de reaproveitamento de 10% dos filhos na geração seguinte. Os parâmetros das ilhas: topologia unidirecional, troca de material genético a cada 300 gerações, quantidade de indivíduos enviados e recebidos de 2. Quando é executada somente 1 ilha, os parâmetros de migração não são considerados.

Para avaliar a aptidão, seguimos a proposta da literatura. Os dados informados representam pontos em um plano 2D. É calculada através da função de distância Euclidiana $d(Euc) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Os trabalhos relacionados consideram somente custos inteiros nas rotas, assim, um arredondamento é realizado com uso da função *round* disponível pela biblioteca *math.h*. A matriz de custo para sair de i para j é armazenado já com custo inteiro de cada trajeto disponível, seguindo os trabalhos desenvolvidos com essas bases. O valor total da aptidão é obtido com a somatória das rotas de todos os veículos.

Um resumo dos parâmetros que foram escolhidos para a execução dos testes, que serão apresentados no Capítulo 5, segue na Tabela 6, a seguir.

Tabela 6 – Resumo dos Parâmetros Calibrados.

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

Fonte: Elaborado pelo autor (2023).

5 RESULTADOS

Neste capítulo, apresentamos e discutimos os resultados obtidos a partir dos testes realizados com a heurística proposta no Capítulo 3, utilizando o plano de testes levantado no Capítulo 4. Todos foram realizados em um ambiente de execução Windows 10, processador AMD Ryzen 5 5600 3,5 GHz x 6 núcleos e 12 threads, em uma máquina com 16 GB de memória RAM. Os resultados brutos foram compilados e podem ser consultados no Apêndice A.

Olhando para o conjunto de resultados de forma geral, a Tabela 7 sintetiza os resultados obtidos a partir da médias. É possível verificar que houve uma melhoria progressiva da média e do desvio padrão quando adicionado mais ilhas. Em relação ao tempo de execução com 1 e mais ilhas, ocorreu um aumento no tempo de execução de 33 milissegundos com a adição de 1 ilha, 53 milissegundos com 4 ilhas, e 229 milissegundos com 6 ilhas. Com isso, é constatado uma maior consistência do método em troca de tempo de processamento. Exemplificando essa relação de troca, comparando a execução de 1 e 6 ilhas, o método gerou uma diferença de 6,56% de distância em relação ao valor ótimo, uma melhora relativa de 49,58%, com aumento de 21,26% milissegundos no tempo de execução.

Tabela 7 – Média dos Resultados Obtidos Dos Testes

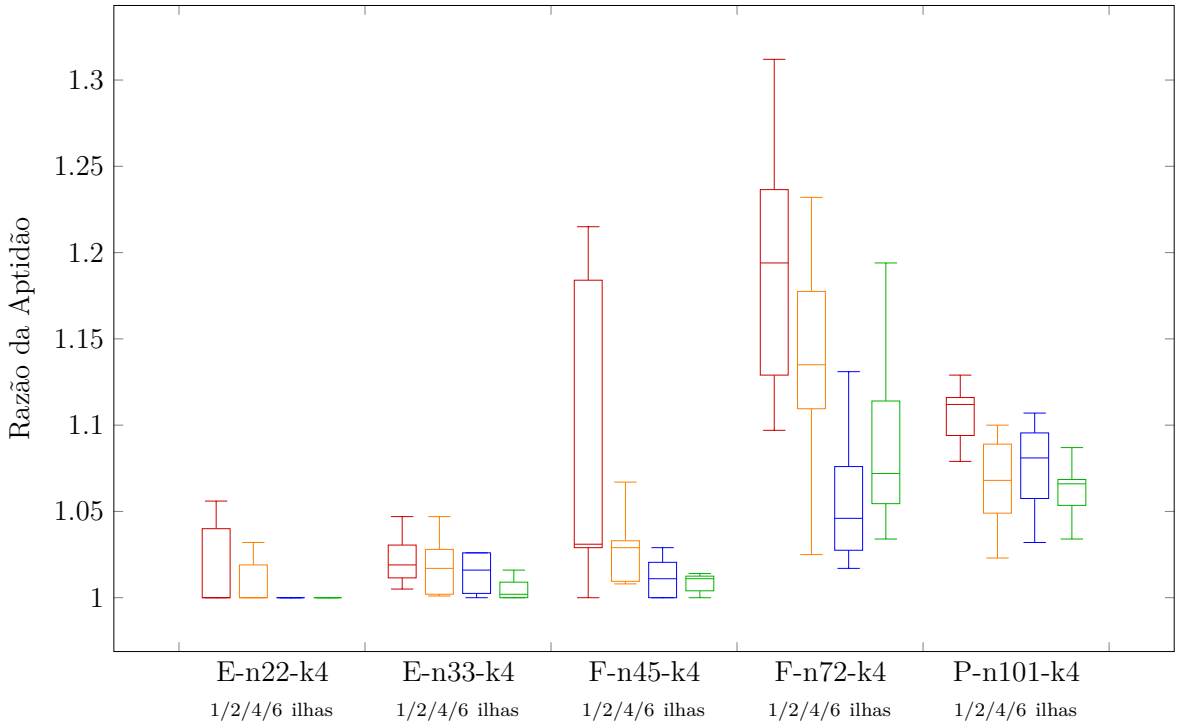
	1 Ilha	2 Ilhas	4 Ilhas	6 Ilhas
Distância da Média dos Resultados em Relação ao Ótimo Global (%)	19,79%	16,22%	14,40%	13,23%
Desvio Padrão dos Resultados	35,21	25,41	24,15	20,45
Tempo de Execução (segundos)	1,077	1,110	1,130	1,306

Fonte: Elaborado pelo autor (2023).

A Figura 19 demonstra o resumo gráfico dos testes do Grupo 1 em relação a aptidão. No eixo X, encontra-se as instâncias separadas por classes, onde cada uma contém o resumo com o crescimento de ilhas. No eixo Y, a proporção dos resultados em relação ao ótimo global, por exemplo, uma razão de 1.1 significa uma solução de 10% distante do ótimo global.

Como é possível observar sobre o G1, o algoritmo foi capaz de encontrar soluções ótimas ou próximas, com uma tendência de degradação na medida em que os problemas ficam maiores, como em *F-n72-k5* e *P-n101-k4*, que já não foi possível encontrá-los. O crescimento das ilhas na maioria dos casos representou um avanço no sentido do afunilamento do gráfico com uma menor dispersão. A maior parte dos quartis, o mínimo e o máximo se aproximam da razão ótima 1 com o aumento de ilhas. Além das observações globais, outro ponto que chamou atenção nesses resultados foi que a instância *F-n72-k5* gerou os piores resultados do Grupo, apesar de não ser o maior caso de teste. Esse item indicia fortemente o impacto do *tigh* no cenário.

Figura 19 – Resultados do Grupo 1.



Fonte: Elaborado pelo autor (2023).

No Grupo 2, não foi possível encontrar alguma evidência conclusiva sobre uma diferença de desempenho do algoritmo que relacione o formato de distribuição dos clientes em instâncias de tamanho médio com alguma flexibilidade. Em quase todos os testes foi possível obter uma solução igual ou aproximada do ótimo. A Tabela 8 demonstra os índices de melhoria obtidos a partir do uso de 1 ilha, com comparativo dos demais testes. É possível observar que a instância *A-n34-k5* obteve um bom desempenho com uso de 1 ilha, fazendo com que o uso de mais ilhas não apresentasse melhoria considerável, tanto na média quanto no desvio padrão. Além disso, a instância *B-n45-k5* teve um comportamento contrário, aumentando consideravelmente os índices. É possível ver com clareza na segunda coluna que o uso do modelo diminui a variância significativamente.

Tabela 8 – Taxas de melhoria do G2 em relação ao uso de 1 ilha

Nome	Tipo	Taxa de melhoria da média			Melhoria relativa do desvio padrão		
		2 ilhas	4 ilhas	6 ilhas	2 ilhas	4 ilhas	6 ilhas
B-n34-k5	Cluster	1,27%	1,67%	1,67%	77,37%	80,00%	84,41%
A-n34-k5	Aleatório	0,08%	0,66%	0,62%	-35,34%	9,39%	36,50%
B-n39-k5	Cluster	2,45%	3,16%	4,34%	30,76%	55,35%	76,66%
A-n39-k5	Aleatório	2,21%	3,82%	3,93%	53,47%	52,78%	68,90%
B-n45-k5	Cluster	8,55%	8,74%	9,49%	81,12%	62,38%	82,63%
P-n45-k5	Aleatório	3,03%	2,07%	3,67%	51,65%	50,27%	73,30%

Fonte: Elaborado pelo autor (2023).

As médias do Grupo 4 seguiram o padrão de melhoria com o acréscimo das ilhas observada nos demais conjuntos, conforme Tabela 9, que contém a média das execuções e sua distância

em relação ao ótimo. O acréscimo de veículos representou uma melhoria na distância em relação ao ótimo. Mais veículos carregam menos carga e possui um cromossomo maior, ocasionando maior dificuldade em realizar a busca confirmada pelos resultados. Além disso, *E-n76-k7* que é menos restritivo se aproximou mais da melhor solução.

Tabela 9 – Resultados do Grupo 4

		1 Ilha		2 Ilhas		4 Ilhas		6 Ilhas	
Instância	Ótimo	Média	Diferença	Média	Diferença	Média	Diferença	Média	Diferença
P-n76-k4	593	681,2	14,87%	673,5	13,58%	658,3	11,01%	638,6	7,69%
E-n76-k7	682	740,7	8,61%	720,4	5,63%	713,6	4,63%	709,2	3,99%
E-n76-k10	830	975,8	17,57%	962,8	16,00%	938,8	13,11%	928,5	11,87%
E-n76-k14	1021	1.171,00	14,69%	1.160,30	13,64%	1.122,60	9,95%	1.129,20	10,60%

Fonte: Elaborado pelo autor (2023).

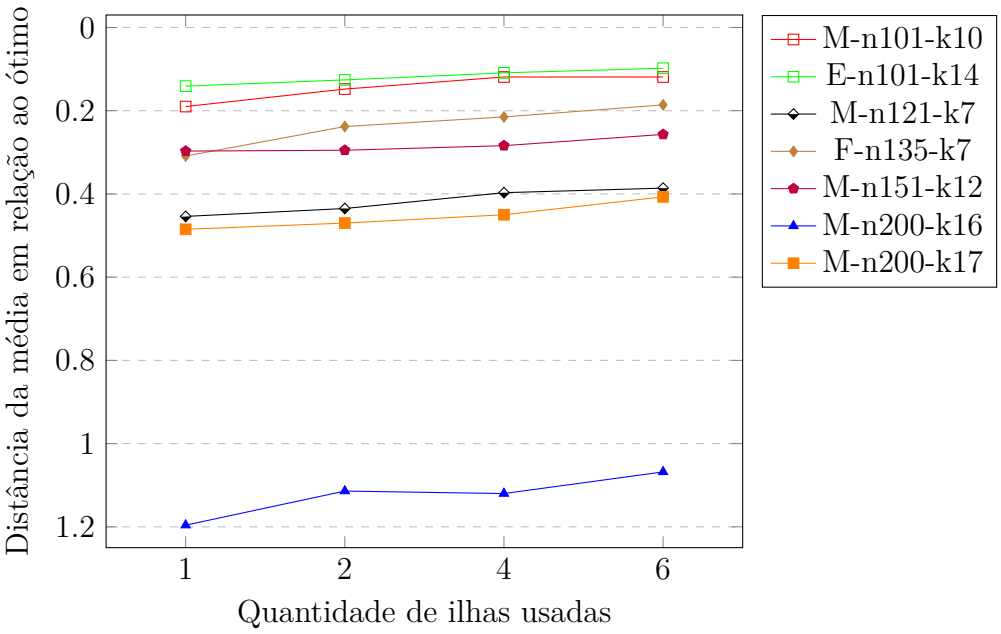
G3 e G5 corroboram com a ideia de que um dos desafios para o algoritmo é a resolução de problemas com *tigh* apertado. Os dados encontrados demonstram uma melhoria com o uso de ilhas, porém há uma clara queda de desempenho nos resultados em problemas com pouca margem. Entretanto, no tempo de processamento não houve alterações que representasse algum padrão, estando aproximadas.

A Figura 20 sintetiza os resultados das médias obtidas nas instâncias do G5. No eixo X, encontrasse a quantidade de ilhas, no Y, a distância da média em relação ao ótimo global. Dois resultados presentes chamam atenção. Primeiro o desempenho geral, onde há uma progressão dos resultados com o uso de mais ilhas, em todos os casos é possível ver uma curva de melhora. Como segunda observação, é possível perceber que houve uma diferença entre as instâncias com margem (*tigh*) e as sem margem. Um comparativo entre *M-n200-k16* e *M-n200-k17*, com margem respectivas de 1 e $0,94$, confirma uma diferença consistente na capacidade do método em melhorar as soluções com essas características. O mesmo *outlier* ocorre olhando para *M-n121-k7*, com resultados comparáveis a *M-n151-k12*, cujo problema é consideravelmente maior.

Considerando todos os grupos de testes, de uma forma geral, aumentar ilhas demonstrou uma tendência de melhora da média das soluções, redução no desvio padrão e maior alcance do melhor indivíduo, salvo exceções de prováveis artefatos. Esses resultados mostram a tendência do modelo de aprimorar o desempenho em troca do aumento do tempo de processamento e do consumo de recursos computacionais.

Em relação ao tempo de execução das instâncias, como já citado no referencial teórico, a relação de tempo com o aumento de ilhas é condicionado pelo *speedup* do algoritmo. Abstraindo como o Sistema Operacional realiza o escalonamento das *threads*, o principal fator de influência está na utilização dos mutexes ao acessar áreas de memória para a migração genética. Assim, a Tabela 21 demonstra as médias de tempo dos testes, expressos em segundos, além dos valores mínimos e máximos. Os resultados indicaram um aumento moderado de tempo com o acréscimo de ilhas, especialmente ao atingir 4 ilhas, que teve

Figura 20 – Gráfico da variação da média com o aumento do número de ilhas.



Fonte: Elaborado pelo autor (2023).

custo somente 4,63% maior que com o uso de 1 ilha, tratando-se da média. Instâncias menores naturalmente possuem uma diferença menor, assim como instâncias maiores aumentam.

Figura 21 – Tempo de Execução dos Testes em Segundos.

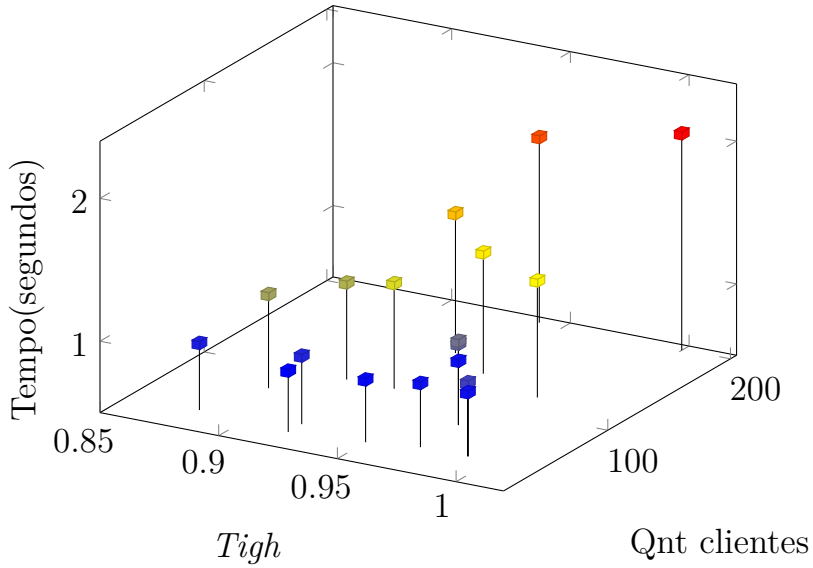
	1 Ilha	2 Ilhas	4 Ilhas	6 Ilhas
Média	1,08	1,11	1,13	1,31
Mínimo	0,89	0,92	0,9	1,07
Máximo	1,96	1,97	2,02	2,36

Fonte: Elaborado pelo autor (2023).

A Figura 22 mostra um gráfico em três dimensões, cujos eixos do plano inferior representam os principais fatores de impacto no tempo de execução das instâncias, a flexibilidade (*tigh*) do problema, e a quantidade de clientes. Como é possível observar, o fator preponderante é a quantidade de clientes, que ocasiona o aumento no tamanho real dos cromossomos e consequentemente demanda maior processamento, fruto de mais iterações dentro dos laços de repetição. Já a flexibilidade, não representou um maior custo de forma consistente, impactando somente na qualidade das soluções.

Selecionamos alguns trabalhos da literatura que utilizam a base de testes, mesmo que parcialmente, para fins de localização da solução frente a outros métodos. Focamos em soluções mais recentes que apresentavam em seus resultados a aptidão e o custo computacional. Mesmo que não utilizando o paralelismo, esses trabalhos ajudam a ilustrar o *trade-off* (relação de troca) da qualidade das soluções com o tempo obtido:

Figura 22 – Gráfico do Tempo de Execução.



Fonte: Elaborado pelo autor (2023).

- Faiz, Subiyanto e Arief (2018) usando os conjuntos A e B, realizou testes usando uma solução de pesquisa de vizinhança variável baseada em perturbação (*perturbation based variable neighborhood search*). O trabalho deles alcançou o ótimo global ou ficou muito próximo, considerando as instâncias intercedentes com o nosso. Porém, a média de tempo de execução obtido nos testes deles foram superiores a 1 minuto, um tempo maior.

- Sbai, Krichen e Limam (2020) conduziram experimentos usando os conjuntos A, B, P e E. Eles exploraram duas abordagens baseadas em Algoritmos Genéticos, sem paralelismo. Seus resultados de aptidão foram melhor que os resultados encontrados no nosso trabalho, porém, com um tempo de execução substancialmente maior. Como ilustração, na instância *P-n45-k5*, os autores identificaram a solução ótima global com uma média de tempo de 37,65 segundos, enquanto alcançamos uma distância de 1,49% em relação ao ótimo global com uso de 6 ilhas, e um tempo de 1,12 segundos, além de encontrarmos o ótimo em algumas das execuções.

- Kır, Yazgan e Tüncel (2017) usando busca tabu e pesquisa adaptativa de grande vizinhança (*adaptive large neighborhood search*). O conjunto A e algumas instâncias do grupo M foram testadas. Na instância *A-n34-k5*, os autores alcançaram o ótimo global em 2,98 segundos, enquanto em alguns dos nossos resultados foi possível encontrar esse valor com um tempo de 0,93 segundos (2 e 4 ilhas). Em *M-n151-k12*, os autores se aproximaram do ótimo global com um pouco mais que 10,37 horas, enquanto ficamos distante cerca de 26%, com tempo de 1,72 segundos. Já em *M-n200-k17*, quase 12 horas foram necessárias, enquanto ficamos cerca de 41% do ótimo global em 2,08 segundos. Não é possível afirmar se o nosso código conseguiria realizar esse afunilamento, porém,

esses resultados mostram o quão complexo e custoso pode ser esse processo.

Os estudos mencionados destacaram uma questão relevante relacionada ao plano de testes, que envolveu a falta de diferenciação entre os tamanhos dos problemas. Observou-se uma adaptação limitada aos parâmetros ajustados do algoritmo, com foco especial na quantidade de gerações. Essa limitação se refletiu claramente no tempo de execução dos testes, onde apresentamos uma amplitude de apenas $1,47$ segundos entre as instâncias.

Ao analisar os dados e compará-los com a literatura, torna-se evidente que as instâncias mais extremas da base de testes (*E-n76-k10*), parecem desequilibradas, com um excesso de gerações para instâncias menores e uma necessidade de mais gerações para uma avaliação adequada da capacidade de exploração em instâncias maiores. Essa observação sugere que a otimização dos parâmetros poderia resultar em soluções próximas com tempos de execução mais eficientes, pelo menos para um subconjunto de instâncias do problema.

6 CONCLUSÃO E TRABALHOS FUTUROS

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

As etapas realizadas foram: um levantamento bibliográfico dos assuntos pertinentes; a definição de uma proposta; a implementação e testes preliminares; o levantamento de uma base e um plano de testes a partir de *benchmark* da literatura; a calibração de um conjunto de parâmetros; a execução do plano; e a apresentação e discussão dos resultados.

Foram conduzidos testes para avaliar o método construído, variando a quantidade de ilhas (1, 2, 4 e 6), e mantendo constantes os parâmetros genéticos e migratórios. Os resultados destes testes revelaram uma melhoria dos resultados às custas de um aumento no custo de processamento ao comparar o uso de uma única ilha com múltiplas ilhas, permitindo um melhor aproveitamento do sistema multinúcleo. Notavelmente, em 87,5% das instâncias avaliadas, a melhor média foi alcançada com o uso de 6 ilhas, enquanto nos outros 12,5%, 4 ilhas foram mais eficazes. Para realizar isso, um aumento de custo médio de 4,84% (4 ilhas) e 21,09% (6 ilhas) do tempo em relação ao AG tradicional (1 ilha). Além disso, vale o destaque que o acréscimo de ilhas resultou em uma redução significativa no desvio padrão. Esses resultados indicam que o método paralelo não apenas melhorou a capacidade de gerar indivíduos mais aptos, mas também tornou-se mais consistente, robusto e eficiente quando aplicado no modelo, corroborando com a literatura acerca do assunto.

No geral, o modelo se demonstrou promissor enquanto alternativa para a construção de sistemas para a resolução de problemas roteamento de veículos, em especial o capacitado, bem como variantes, como o problema do caixeiro viajante. Esses sistemas, partindo de uma expansão horizontal de *hardware*, podem ser ferramentas úteis quando a otimização de problemas complexos carece de alternativas de tratamento eficazes.

Apesar de se demonstrar promissor, pode ser que as migrações criem condições favoráveis a uma exploração de um mesmo mínimo local em todas as ilhas. Assim, trabalhos futuros podem explorar o impacto na diversidade entre as ilhas no modelo. Também, parâmetros ou técnicas distintas podem ser usados nas ilhas, tanto no sentido de criar ambientes propícios para diversidade de formato de exploração, no sentido de adaptabilidade quando detecta-se uma dificuldade em realizar uma busca local, e, no sentido de explorar outras heurísticas. Outra sugestão, a arquitetura pode ser expandida para ambientes de computação distribuída, aumentando a capacidade computacional, visando solucionar problemas maiores, mais complexos ou técnicas de soluções mais custosas que justifiquem o tempo de troca em rede de soluções.

REFERÊNCIAS

ABBASI, M. et al. An efficient parallel genetic algorithm solution for vehicle routing problem in cloud implementation of the intelligent transportation systems. *Journal of Cloud Computing*, v. 9, p. 6, 12 2020. ISSN 2192-113X.

ABDELATTI, M. F.; SODHI, M. S. An improved gpu-accelerated heuristic technique applied to the capacitated vehicle routing problem. In: . New York, NY, USA: Association for Computing Machinery, 2020. (GECCO '20), p. 663–671. ISBN 9781450371285. Disponível em: <https://doi.org/10.1145/3377930.3390159>.

ADAMIDIS, P.; GREECE, T. Review of parallel genetic algorithms bibliography - version 1. 05 1995.

ALBA, E.; DORRONSORO, B. Computing nine new best-so-far solutions for capacitated vrp with a cellular genetic algorithm. *Information Processing Letters*, v. 98, n. 6, p. 225–230, 2006. ISSN 0020-0190. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0020019006000548>.

ALBA, E.; TROYA, J. M. A survey of parallel distributed genetic algorithms. *Complex.*, John Wiley & Sons, Inc., USA, v. 4, n. 4, p. 31–52, mar 1999. ISSN 1076-2787.

ALTABEEB, A. M.; MOHSEN, A. M.; GHALLAB, A. An improved hybrid firefly algorithm for capacitated vehicle routing problem. *Applied Soft Computing*, v. 84, p. 105728, 2019. ISSN 1568-4946. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1568494619305095>.

AMMI, M.; CHIKHI, S. *A Generalized Island Model Based on Parallel and Cooperating Metaheuristics for Effective Large Capacitated Vehicle Routing Problem Solving*. Faculty of Electrical Engineering and Computing, Univ. of Zagreb, 2015. 141 p. Disponível em: <http://dx.doi.org/10.2498/cit.1002465>.

AMOUS, M. et al. A variable neighborhood search algorithm for the capacitated vehicle routing problem. *Electronic Notes in Discrete Mathematics*, v. 58, p. 231–238, 2017. ISSN 1571-0653. 4th International Conference on Variable Neighborhood Search. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1571065317300665>.

AUGERAT, P. et al. *Computational results with a branch and cut code for the capacitated vehicle routing problem*. 1995.

BALLOU, R. H. *Gerenciamento da Cadeia de Suprimentos/Logística Empresarial*. 5ª edição. ed. [S.l.]: Bookman, 2006.

BATTARRA, M.; ERDOGAN, G.; VIGO, D. Exact algorithms for the clustered vehicle routing problem. *Operations Research*, v. 62, p. 58–71, 2 2014. ISSN 0030-364X.

BLASUM, U.; HOCHSTÄTTLER, W. Application of the branch and cut method to the vehicle routing problem. In: . [S.l.: s.n.], 2000.

BORCINOVA, Z. Two models of the capacitated vehicle routing problem. *Croatian Operational Research Review*, v. 8, p. 463–469, 12 2017.

BORCINOVA, Z. Solving the capacitated vehicle routing problem using a parallel micro genetic algorithm. *2018 IEEE Workshop on Complexity in Engineering (COMPENG)*, p. 1–6, 2018.

CANTÚ-PAZ, E. A survey of parallel genetic algorithms. *Calculateurs Paralleles, Reseaux et Systems Repartis*, v. 10, n. 2, p. 141–171, 1998.

CANTÚ-PAZ, E. Efficient and accurate parallel genetic algorithms. *Boston, Kluwer Academic Publishers*, 01 2000.

CANTÚ-PAZ, E.; GOLDBERG, D. E. Efficient parallel genetic algorithms: theory and practice. *Computer Methods in Applied Mechanics and Engineering*, v. 186, n. 2, p. 221–238, 2000. ISSN 0045-7825. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0045782599003850>.

CHIPPERFIELD, A.; FLEMING, P. Parallel genetic algorithms. *Parallel and Distributed Computing Handbook*, p. 1118–1143, 01 1996.

CHRISTOFIDES, N.; EILON, S. An algorithm for the vehicle-dispatching problem. *Journal of the Operational Research Society*, v. 20, p. 309–318, 1969. Disponível em: <https://api.semanticscholar.org/CorpusID:62174543>.

CHRISTOFIDES, N.; MINGOZZI, A.; TOTH, P. The vehicle routing problem. *Combinatorial Optimization. Wiley, Chichester*, p. 315– 338, 1979.

CHRISTOFIDES, N.; MINGOZZI, A.; TOTH, P. State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, v. 11, n. 2, p. 145–164, 1981. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/net.3230110207>.

CLARKE, G.; WRIGHT, J. W. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, v. 12, n. 4, p. 568–581, 1964. Disponível em: <https://doi.org/10.1287/opre.12.4.568>.

CNT. *Plano CNT de transporte e logística 2018.* [S.l.], 2018.

CRAINIC, T. G.; TOULOUSE, M. Parallel strategies for meta-heuristics. In: _____. *Handbook of Metaheuristics*. Boston, MA: Springer US, 2003. p. 475–513. ISBN 978-0-306-48056-0. Disponível em: <https://doi.org/10.1007/0-306-48056-5_17>.

CVRPLIB. *CVRPLIB - All Instances*. 2014. Disponível em: <http://vrp.galgos.inf.puc-rio.br/>.

DANTZIG, G. B.; RAMSER, J. H. The truck dispatching problem. *Management Science*, v. 6, n. 1, p. 80–91, 1959. Disponível em: <https://doi.org/10.1287/mnsc.6.1.80>.

DORRONSORO, B. et al. A grid-based hybrid cellular genetic algorithm for very large scale instances of the cvrp. In: *21st European Conference on Modelling and Simulation: Simulations in United Europe, ECMS 2007*. [S.l.: s.n.], 2007. p. 759–765. ISBN 9780955301827.

DREXL, M. Rich vehicle routing in theory and practice. *Logistics Research*, v. 5, p. 47–63, 8 2012. ISSN 1865-035X.

DUARTE, G.; LEMONGE, A.; GOLIATT, L. A dynamic migration policy to the island model. In: *2017 IEEE Congress on Evolutionary Computation (CEC)*. [S.l.: s.n.], 2017. p. 1135–1142.

EIBEN, A. E.; SMITH, J. E. *Introduction to Evolutionary Computing*. 2nd. ed. [S.l.]: Springer Publishing Company, Incorporated, 2015. ISBN 3662448734.

EKSIOGLU, B.; VURAL, A. V.; REISMAN, A. The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, v. 57, p. 1472–1483, 11 2009. ISSN 03608352.

FAIZ, A.; SUBIYANTO, S.; ARIEF, U. An efficient meta-heuristic algorithm for solving capacitated vehicle routing problem. *International Journal of Advances in Intelligent Informatics*, v. 4, n. 3, p. 212–225, 2018. ISSN 2548-3161. Disponível em: <https://ijain.org/index.php/IJAIN/article/view/244>.

FERNANDEZ, F.; TOMASSINI, M.; VANNESCHI, L. An empirical study of multipopulation genetic programming. *Genetic Programming and Evolvable Machines*, v. 4, n. 1, p. 21–51, mar. 2003. ISSN 1389-2576. Disponível em: <https://rdcu.be/c5oUz>.

FISHER, M. L. Optimal solution of vehicle routing problems using minimum k-trees. *Operations Research*, INFORMS, v. 42, n. 4, p. 626–642, 1994. ISSN 0030364X, 15265463. Disponível em: <http://www.jstor.org/stable/171617>.

FOX, B.; MCMAHON, M. Genetic operators for sequencing problems. In: RAWLINS, G. J. (Ed.). Elsevier, 1991, (Foundations of Genetic Algorithms, v. 1). p. 284–300. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780080506845500215>.

GLOVER, F.; KOCHENBERGER, G. A. (Ed.). *Handbook of Metaheuristics*. [S.l.]: Springer US, 2003. v. 57. ISBN 978-1-4020-7263-5.

GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. 1st. ed. USA: Addison-Wesley Longman Publishing Co., Inc., 1989. ISBN 0201157675.

GOLDBERG, D. E.; DEB, K. A comparative analysis of selection schemes used in genetic algorithms. In: . [S.l.: s.n.], 1991. p. 69–93.

GOLDBERG, D. E.; LINGLE, R. Alleles, loci and the traveling salesman problem. In: *Proceedings of the 1st International Conference on Genetic Algorithms*. USA: L. Erlbaum Associates Inc., 1985. p. 154–159. ISBN 0805804269.

GOODMAN; PUNCH; LIN, S.-C. Coarse-grain parallel genetic algorithms: categorization and new approach. In: *Parallel and Distributed Processing, IEEE Symposium on*. Los Alamitos, CA, USA: IEEE Computer Society, 1994. p. 28–37. Disponível em: <https://doi.ieeecomputersociety.org/10.1109/SPDP.1994.346184>.

HARADA, T.; ALBA, E. Parallel genetic algorithms: A useful survey. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 53, n. 4, aug 2020. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3400031>.

HAUPT, R. L.; HAUPT, S. E. *Practical genetic algorithms*. 2. ed. Nashville, TN: John Wiley & Sons, 2004.

HOTTUNG, A.; TIERNEY, K. Neural large neighborhood search for the capacitated vehicle routing problem. *arXiv*, arXiv, 2019. Disponível em: <https://arxiv.org/abs/1911.09539>.

KAZIMIPOUR, B.; LI, X.; QIN, A. K. A review of population initialization techniques for evolutionary algorithms. In: . [S.l.]: IEEE, 2014. p. 2585–2592. ISBN 978-1-4799-1488-3.

KIR, S.; YAZGAN, H. R.; TÜNCEL, E. A novel heuristic algorithm for capacitated vehicle routing problem. *J. Ind. Eng. Int.*, Springer Nature, v. 13, n. 3, p. 323–330, set. 2017.

KUMAR, V. S. G.; PANNEERSELVAM, R. A study of crossover operators for genetic algorithms to solve vrp and its variants and new sinusoidal motion crossover operator. *International Journal of Computational Intelligence Research*, Volume 13, p. Page 1717–1733, 06 2017.

LAHYANI, R.; KHEMAKHEM, M.; SEMET, F. Rich vehicle routing problems: From a taxonomy to a definition. *European Journal of Operational Research*, v. 241, n. 1, p. 1–14, 2015. ISSN 0377-2217. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0377221714006146>.

LAPORTE, G. et al. Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research*, v. 7, n. 4, p. 285–300, 2000. ISSN 0969-6016. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0969601600000034>.

LARDEUX, F.; GOËFFON, A. A dynamic island-based genetic algorithms framework. p. 156–165, 12 2010.

LARDEUX, F. et al. Migration policies in dynamic island models. *Natural Computing: An International Journal*, Kluwer Academic Publishers, USA, v. 18, n. 1, p. 163–179, mar 2019. ISSN 1567-7818. Disponível em: <https://doi.org/10.1007/s11047-017-9660-z>.

LARRAÑAGA, P. et al. Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artificial intelligence review: An international survey and tutorial journal*, Springer Netherlands, v. 13, n. 2, p. 129–170, abr. 1999. ISSN 0269-2821.

LEE, T. R.; UENG, J. A study of vehicle routing problems with load-balancing. *International Journal of Physical Distribution & Logistics Management*, v. 29, p. 646–657, 1999.

LIN, W.-Y. et al. Revisiting the design of adaptive migration schemes for multipopulation genetic algorithms. In: *2012 Conference on Technologies and Applications of Artificial Intelligence*. [S.l.: s.n.], 2012. p. 338–343.

LINDEN, R. *Algoritmos Geneticos*. 3ª edicao. ed. [S.l.]: Ciência Moderna, 2012. Original-date: 2012. ISBN 9788539901951.

LUCASIUS, C.; KATEMAN, G. Understanding and using genetic algorithms part 2. representation, configuration and hybridization. *Chemometrics and Intelligent Laboratory Systems*, v. 25, n. 2, p. 99–145, 1994. ISSN 0169-7439. Disponível em: <https://www.sciencedirect.com/science/article/pii/0169743994850380>.

LYSGAARD, J.; LETCHFORD, A. N.; EGLESE, R. W. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, v. 100, p. 423–445, 6 2004. ISSN 0025-5610.

MAMBRINI, A.; SUDHOLT, D. Design and analysis of adaptive migration intervals in parallel evolutionary algorithms. In: *Proceedings of the 2014 Annual Conference on*

Genetic and Evolutionary Computation. New York, NY, USA: Association for Computing Machinery, 2014. (GECCO '14), p. 1047–1054. ISBN 9781450326629. Disponível em: <https://doi.org/10.1145/2576768.2598347>.

MAN, K.; TANG, K.; KWONG, S. Genetic algorithms: concepts and applications [in engineering design]. *IEEE Transactions on Industrial Electronics*, v. 43, n. 5, p. 519–534, 1996.

MICHALEWICZ, Z. Genetic algorithms + data structures = evolution programs. In: *Springer Berlin Heidelberg*. [S.l.: s.n.], 1996.

MIRJALILI, S. Genetic algorithm. In: _____. *Evolutionary Algorithms and Neural Networks: Theory and Applications*. Cham: Springer International Publishing, 2019. p. 43–55. ISBN 978-3-319-93025-1. Disponível em: <https://doi.org/10.1007/978-3-319-93025-1_4>.

MITCHELL, M. *An Introduction to Genetic Algorithms*. The MIT Press, 1998. ISBN 9780262280013. Disponível em: <https://doi.org/10.7551/mitpress/3927.001.0001>.

PEARL, J. Heuristics: Intelligent search strategies for computer problem solving. 1 1984. Disponível em: <https://www.osti.gov/biblio/5127296>.

RALPHS, T. et al. On the capacitated vehicle routing problem. *Mathematical Programming*, v. 94, p. 343–359, 1 2003. ISSN 0025-5610.

REEVES, C. R.; ROWE, J. E. *Genetic Algorithms—Principles and Perspectives*. [S.l.]: Springer US, 2002. v. 20. Https://doi.org/10.1007/b101880. ISBN 978-1-4020-7240-6.

RUCIŃSKI, M.; IZZO, D.; BISCANI, F. *On the Impact of the Migration Topology on the Island Model*. 2010.

SASTRY, K.; GOLDBERG, D.; KENDALL, G. Genetic algorithms. In: _____. *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. Boston, MA: Springer US, 2005. p. 97–125. ISBN 978-0-387-28356-2. Disponível em: <https://doi.org/10.1007/0-387-28356-0_4>.

SBAI, I.; KRICHEN, S.; LIMAM, O. Two meta-heuristics for solving the capacitated vehicle routing problem: the case of the tunisian post office. *Operational Research*, Springer Science and Business Media LLC, v. 22, n. 1, p. 507–549, jan. 2020. Disponível em: <https://doi.org/10.1007/s12351-019-00543-8>.

SCHMIDT, B. et al. Chapter 2 - theoretical background. In: SCHMIDT, B. et al. (Ed.). *Parallel Programming*. Morgan Kaufmann, 2018. p. 21–45. ISBN 978-0-12-849890-3. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780128498903000022>.

SHUKLA, A.; PANDEY, H. M.; MEHROTRA, D. Comparative review of selection techniques in genetic algorithm. In: . [S.l.]: IEEE, 2015. p. 515–519. ISBN 978-1-4799-8433-6.

SKOLICKI, Z.; JONG, K. D. The influence of migration sizes and intervals on island models. In: *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*. New York, NY, USA: Association for Computing Machinery, 2005. (GECCO '05), p. 1295–1302. ISBN 1595930108. Disponível em: <https://doi.org/10.1145/1068009.1068219>.

TAN, S.-Y.; YEH, W.-C. The vehicle routing problem: State-of-the-art classification and review. *Applied Sciences*, v. 11, p. 10295, 11 2021. ISSN 2076-3417.

TANG, J. et al. Study of migration topology in island model parallel hybrid-ga for large scale quadratic assignment problems. *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference, 2004.*, v. 3, p. 2286–2291 Vol. 3, 2004. Disponível em: <https://api.semanticscholar.org/CorpusID:12185358>.

TOTH, P.; VIGO, D. *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics, 2002. Disponível em: <https://epubs.siam.org/doi/abs/10.1137/1.9780898718515>.

TOTH, P.; VIGO, D. *The Granular Tabu Search and Its Application to the Vehicle-Routing Problem*. Institute for Operations Research and the Management Sciences (INFORMS), 2003. 333–346 p. Disponível em: <http://dx.doi.org/10.1287/ijoc.15.4.333.24890>.

VRAJITORU, D. Large population or many generations for genetic algorithms? implications in information retrieval. In: . [S.l.: s.n.], 2000. p. 199–222.

WHITLEY, D. A genetic algorithm tutorial. *Statistics and Computing*, v. 4, 6 1994. ISSN 0960-3174.

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

μ: média aritmética; σ: desvio padrão; *Best*: melhor resultado; $t(s)$: tempo(segundos)