UNIVERSIDADE FEDERAL DO CEARÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO E ELÉTRICA

KILLDARY AGUIAR DE SANTANA

MÉTODO METAHEURÍSTICO MULTIOBJETIVO PARA O CÁLCULO DE ROTAS DE ROBÔS COLABORATIVOS COM RESTRIÇÕES ENERGÉTICAS

SOBRAL

2020

KILLDARY AGUIAR DE SANTANA

MÉTODO METAHEURÍSTICO MULTIOBJETIVO PARA O CÁLCULO DE ROTAS DE ROBÔS COLABORATIVOS COM RESTRIÇÕES ENERGÉTICAS

Dissertação apresentada ao Curso de do Programa de Pós-Graduação em Engenharia da Computação e Elétrica do da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia da Computação e Elétrica. Área de Concentração: Eletrônica de Potência

Orientador: Prof. Dr. Vandilberto Pereira Pinto

SOBRAL

2020

KILLDARY AGUIAR DE SANTANA

MÉTODO METAHEURÍSTICO MULTIOBJETIVO PARA O CÁLCULO DE ROTAS DE ROBÔS COLABORATIVOS COM RESTRIÇÕES ENERGÉTICAS

Dissertação apresentada ao Curso de do Programa de Pós-Graduação em Engenharia da Computação e Elétrica do da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia da Computação e Elétrica. Área de Concentração: Eletrônica de Potência

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Vandilberto Pereira Pinto (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Eber de Castro Diniz
Universidade Federal do Ceará (UFC)

Prof. Dr. Allberson Bruno de Oliveira Dantas
Universidade da Integração Internacional Lusofonia Afro-Brasileira (Unilab)

À minha família, por sua capacidade de acreditar em mim e investir em mim. Pai, sua presença e carinho significaram segurança e certeza de que não estou sozinho nessa caminhada. Mãe, seu cuidado e dedicação foi o que me permitiu crescer, dedico essa vitória em sua memória.

AGRADECIMENTOS

É chegado o momento de agradecer pelo apoio e o carinho que recebi ao longo deste período de mestrado.

Ao Pai Celestial, a mim Deus, por ter me dado o valor, força para lutar e conseguir este sonho, por me permitir acordar dia a dia e ver o céu, o sol, a chuva que ele fez para mim. O Senhor que me concedeu a benção da vida e toda as minhas conquistas, pois sem a ele nada seria possível. Eu agradeço e louvo o seu santo nome.

Agradeço ao meu pai Tarcivaldo e minha Mãe Socorro, apesar de não mais presente sei que ela ainda olha por nós e cuida de mim, que sempre me incentivaram nos estudos, me guiaram com seus conselhos e também pelo amor e carinho. Obrigada mãe por ter me dado a vida, ter me ensinado, me guiado e me preparado para as batalhas do cotidiano. Amo-te eternamente.

Aos meus irmãos Kellry, KLinger e Klivya pelo apoio e carinho.

Um obrigado aos meus amigos Doutorando em Engenharia Elétrica Darielson e Engenheiro de Automação João Lucas por me acompanharem e me guiarem quando dificuldades eram encontradas. A amizade de vocês foi muito importante na conclusão deste trabalho.

Ao meu orientador Prof. Dr. Vandilberto pela oportunidade e a honra de tê-lo como orientador. A sua paciência e ensinamentos foram fundamentais para a conclusão da pesquisa.

Este trabalho não seria possível sem o apoio e amor incondicional da minha companheira Sâmia. Muito obrigado pelas noites em que me acompanhou e revisou meu trabalho,pelo incetivo em cada noite em que o cansaço do trabalho pesava. A você dedico o meu agradecimento e meu amor. Amo-te.

Á Universidade Federal do Ceará (UFC) e seus docentes pelos ensinamentos durante essa jornada. Foi uma honra ser aluno dessa instituição.

Á empresa Grendene e meus líderes e chefes Maurício e Francivaldo pelo apoio durante o mestrado.

A todos vocês o meu mais sincero muito obrigado.

“Consagre ao Senhor tudo o que você faz e os seus planos serão bem-sucedidos”

: (Provérbios 16:3)

RESUMO

Um dos maiores desafios encontrados em robôs móveis se dá no planejamento de sua trajetória e tem recebido atenção por parte dos pesquisadores, tanto na indústria como no meio acadêmico. O grande motivador dos pesquisadores nesta área se deve ao desenvolvimento de solução que permitam maior autonomia para os robôs. A complexidade do problema de planejamento de trajetórias destes equipamentos tem motivado o desenvolvimento de diversos algoritmos. Isso advém da necessidade de integrar sua a navegação junto ao seu sensoriamento, a eficiência e o planejamento de rotas, como também a necessidade de poupar recursos importantes e a participação de múltiplos agentes. Grande parte das missões que devem atender a vários pontos são bastante complexas, pois além dos custo para sua conclusão elas também podem possuir pesos que determinam sua prioridade de atendimento. Essa complexidade é incrementada com a possibilidade de diversos robôs em diferentes localidades participarem da missão de forma colaborativa. O presente trabalho apresenta um modelo de cálculo para múltiplos robôs com restrições energéticas e mudança de base utilizando a combinação do problema de orientação de equipes e o problema das múltiplas mochilas. Para averiguar a eficiência da solução foram desenvolvidos dois algoritmos meta heurísticos, um algorítimo genético e um método que utiliza otimização por enxame de partículas. Os algoritmos foram testados sobre 20 instâncias autogeradas com diferentes números de robôs, capacidades de bateria e depósitos. Para cada instância foi executado 3 experimentos propostos para execução normal, com redução de robôs e mudança de bases.

Palavras-chave: Múltiplos robôs. Restrições energéticas. Planejamento de rotas. Algoritmos meta-heurísticos. Problema de orientação de equipe.

ABSTRACT

One of the biggest challenges encountered in mobile robots is planning their trajectory and has received attention from researchers, both in industry and academia. The great motivator of researchers in this area is the development of solutions that allow greater autonomy for robots. The complexity of the trajectory planning problem of these equipments has motivated the development of several algorithms. This comes from the need to integrate your navigation with your sensing, efficiency and route planning, as well as the need to save significant resources and the involvement of multiple agents. Most missions that must fulfill several points are quite complex, because in addition to the costs to complete them can also have weights that determine your priority of service. This complexity is increased by the possibility that several robots in different locations collaboratively participate in the mission. This paper presents a calculation model for multiple robots with energy constraints and base shifting using the combination of the team orientation problem and the multiple backpack problem. To verify the efficiency of the solution, two meta heuristic algorithms were developed, a genetic algorithm and a method that uses particle swarm optimization. The algorithms were tested on 20 self-generated instances with different robot numbers, battery capacities and deposits. For each instance 3 experiments were performed for normal execution, with reduction of robots and change of bases.

Keywords: Multiple Robots. Energy Restrictions. Route planning. Metaheuristic Algorithms. Team Orienteering Problem.

LISTA DE FIGURAS

Figura 1 – Rotas para multiplos robôs 18
Figura 2 – Problema da Mochila Multipla 19
Figura 3 – Grafo Simples 25
Figura 4 – Cruzamento de Ponto Único 38
Figura 5 – Cruzamento Múltiplos 39
Figura 6 – PMX 40
Figura 7 – OX 41
Figura 8 – CX 42
Figura 9 – WGWRGM 44
Figura 10 – SWGLM 45
Figura 11 – PSO Topologia 48
Figura 12 – Cromossomo AG Proposto 61
Figura 13 – Cruzamento AG Proposto 69
Figura 14 – Instância 1 70
Figura 15 – Convergência AG proposto 73
Figura 16 – Convergência OEP proposto 74
Figura 17 – Gráfico de dispersão dos dados experimento 1 para a instância I1 75
Figura 18 – Rotas geradas para o experimento 1 na instância $I18$ utilizando AG-Trad e AG-Prop 77
Figura 19 – Rotas geradas para o experimento 1 utilizando para a instância $I1$ AG-Trad, AG-Prop e OEP-Prop 77
Figura 20 – Gráfico de dispersão dos dados do experimento 2 para a Instância I1 79
Figura 21 – Rota Experimento 1 e 2 instância $I20$ 80
Figura 22 – Rotas geradas para o experimento 2 utilizando para a instância $I19$ AG-Trad, AG-Prop 80
Figura 23 – Rotas geradas para o experimento 2 utilizando para a instância $I6$ OEP-Prop e AG-Prop 81
Figura 24 – Rotas geradas para o experimento 2 na exucução da instância e utilizando AG-Trad, AG-Prop e OEP-Prop 81
Figura 25 – Gráfico de dispersão dos dados do experimento 2 para a instância I1 83

Figura 26 – Rotas geradas para o experimento 3 na instância $I5$ utilizando OEP-Prop e AG-Trad 83
Figura 27 – Rotas geradas para o experimento 3 na instância $I1$ utilizando OEP-Prop e AG-Trad 84

LISTA DE TABELAS

Tabela 1 – Variações PCV 26
Tabela 2 – Valores das Instâncias 71
Tabela 3 – Parametrização do Algoritmo genético 72
Tabela 4 – Parametrização do OEP 73
Tabela 5 – Execução do Experimento 1 76
Tabela 6 – Execução do Experimento 2 78
Tabela 7 – Execução do Experimento 3 82

LISTA DE ABREVIATURAS E SIGLAS

ACO	*Ant Colony Otimization*
AG	Algorítimo Genético
AGV	*Automated Guided Vehicle*
ARA*	*Anytime Repairing A**
CX	*Cycle Crossover*
FEVRPTW	*Fuzzy Electric Vehicle Routing Problem with Time Windows and Recharging Stations*
GRASP	*General Responsibility Assignment Software Patterns*
IFA	*In-Flight Awareness*
LNS	*Large Neighborhood Search*
MILP	*Mixed-Integer Linear Programming*
MKP	*Multiple Knapsack Problem*
OEP	Otimização por Enxame de Partículas
OP	*Orienteering Problem*
OX	*Order Crossover*
PCV	Problema do Caixeiro Viajante
PMM	Problema das Mochilas Múltiplas
PMX	*Partially Mapped Crossover*
POE	Problema de Orientação de Equipes
POEMRLMB	Problema de Orientação de Equipes com Múltiplas Restrições de Locomoção e Mudança de Base
PRP	*Pollution Routing Problem*
PRV	Problema de Roteamento de Veículos
PSO	*Particle Swarm Otmization*
SA	*Simulated Annealing*
SWGLM	*Swap Worst Gene Locally Mutation*
TOP	*Team Orienteering Problem*
VANT	Veículos Autônomos Não Tripulados
WGWRGM	*Worst Gene with Random Gene Mutation*
WLRGWRGM	*Worst Left and Right Gene with Random Gene Mutation*

LISTA DE SÍMBOLOS

N_c	Conjunto de *waypoints*
A	Conjunto de arestas
c	Variável que representa o custo de uma arestas
X	Matriz binária que indica a presença das arestas na rota
x	Variável binária que indica a presença de uma aresta na rota
S	Subconjunto de uma rota
T_{max}	Restrição de locomoção
p_i	Variável que representa um premio de um *waypoint* ou item i
w_i	Variável que representa um peso de um item i
N	Conjunto de Itens
$\tilde{N}$	Subconjunto de itens
M	Conjunto de mochilas
N_k	Conjunto de robôs
k	Variável que representa um robô
C_{kmax}	Restrição de locomoção para o robô k
α	Variável que representa o peso do custo
β	Variável que representa o peso da premiação
S_k	Subconjunto de *waypoints* de um agente k
u	Elemento de um sub-rota

SUMÁRIO

1 INTRODUÇÃO 17
1.1 Objetivos 20
1.1.1 Objetivo Geral 20
1.1.2 Objetivos Específicos 20
1.1.3 Justificativa 21
1.2 Organização do Trabalho 21
1.3 Publicações Originadas da Pesquisa 23
2 FUNDAMENTAÇÃO TEÓRICA 24
2.1 Problema do Caixeiro Viajante (PCV) 24
2.1.1 Formulação Matemática 25
2.1.2 Variações do PCV 26
2.2 Problema de Orientação (OP) 27
2.2.1 Formulação Matemática 27
2.3 Team Orienteering Problem (TOP) 28
2.3.1 Formulação Matemática 29
2.4 Problema da Mochila Múltipla (PMM) 31
2.4.1 Formulação Matemática 31
2.5 Meta Heurísticas 32
2.6 Algoritmos Genéticos (AG) 32
2.6.1 Cromossomo 33
2.6.2 Avaliação 34
2.6.3 Seleção 35
2.6.3.1 Seleção de Truncamento *35*
2.6.3.2 Seleção por Roleta *35*
2.6.3.3 Amostragem Universal Estocástica *36*
2.6.3.4 Seleção por Classificação *36*
2.6.3.5 Torneio *36*
2.6.3.6 Elitismo *37*
2.6.4 Cruzamento 37
2.6.4.1 Cruzamento de Ponto Único *37*

2.6.4.2 Cruzamento Múltiplo *38*
2.6.4.3 Cruzamento Uniforme *38*
2.6.4.4 Partially Mapped Crossover(PMX) *39*
2.6.4.5 Order Crossover Operator (OX) *40*
2.6.4.6 Cycle Crossover Operator(CX) *41*
2.6.5 Mutação 42
2.6.5.1 Mutação por Inserção *43*
2.6.5.2 Mutação por Inversão *43*
2.6.5.3 Mutação por Embaralhamento *43*
2.6.5.4 Mutação Swap *43*
2.6.5.5 Mutação Bit Flip *43*
2.6.5.6 Mutação Uniforme *44*
2.6.5.7 Mutação Creep *44*
2.6.5.8 Worst Gene with Random Gene Mutation (WGWRGM) *44*
2.6.5.9 Worst Left and Right Gene with Random Gene Mutation (WLRGWRGM) . . *45*
2.6.5.10 Swap Worst Gene Locally Mutation (SWGLM) *45*
2.7 Otimização por Enxame de Partículas (OEP) 46
3 REVISÃO BIBLIOGRÁFICA 49
3.1 Cálculo de rotas para robôs 49
3.2 Problema de Roteirização de Veículos 55
4 PROBLEMA DE ORIENTAÇÃO DE EQUIPES COM MÚLTIPLAS RESTRIÇÕES DE LOCOMOÇÃO E MUDANÇA DE BASE (POEMRLMB) 58
4.1 Algoritmo Genético Proposto 59
4.1.1 Estrutura do Cromossomo 60
4.1.2 Inicialização da População 60
4.1.3 Fitness 61
4.2 Cruzamento 63
4.2.1 Mutação 63
4.3 Otimização por Enxame de Partículas 65
4.3.1 Estrutura da Partícula 65
4.3.2 Inicialização do Enxame 66
4.3.3 Fitness 66

4.3.4 ***Velocidade da Partícula*** 66
5 **AMBIENTE DE TESTES** 70
5.1 **Instâncias** 70
5.1.1 ***Parametrização do AG proposto*** 72
5.1.2 ***Parametrização do OEP proposto*** 73
6 **RESULTADOS E DISCUSSÕES** 75
6.1 **Resultados do Experimento 1** 75
6.2 **Resultados do Experimento 2** 78
6.3 **Resultados do Experimento 3** 82
7 **CONCLUSÕES E TRABALHOS FUTUROS** 85
7.1 **Trabalhos Futuros** 86
REFERÊNCIAS 87
APÊNDICES 94
APÊNDICE A – Código do Projeto 94
ANEXOS 94

1 INTRODUÇÃO

Com o avanço da tecnologia os robôs se tornaram mais eficazes na execução de diversas tarefas, podendo executar tarefas que antes eram consideradas impossíveis para uma máquina, deixando de ser um componente caro e acessível apenas para grandes indústrias e passando a atuar nas mais diversas aplicações que antes eram executadas por humanos, impulsionando assim o desenvolvimento de robôs autônomos.

A finalidade da pesquisa de robôs móveis com autonomia é construir máquinas para realizar tarefas com precisão e capazes de tomar decisões adequadas frente a uma situação inesperada. As pesquisas e fabricação de robôs com comportamentos autônomos são justificáveis quando é pensado na utilização destas máquinas para realizar tarefas em ambientes hostis, ou seja, nocivos ao ser humano. Por exemplo, a realização de trabalhos em regiões afetadas por radiações químicas ou nucleares, poluídas por gases venenosos, em fundo de oceanos e até mesmo em explorações em outros planetas (ABE *et al.*, 2006). Além de ambientes hostis podem ser utilizados no auxílio de deficientes, na construção civil, na medicina, na domótica e no comércio (ARAI, 2011).

Com o início da industria 4.0 aumentou a demanda da aplicação de robôs na linha de produção, incluindo a utilização de veículos auto guiados, também conhecidos como *Automated Guided Vehicle* (AGV), onde sua antiga navegação através da detecção indutiva de fios embutidos no piso está sendo substituída pelo padrão de mercado utilizando uma navegação inercial baseada em acerto de contas preciso, por meio de giroscópios e odometria de roda, exigindo um melhor planejamento de rotas (WU *et al.*, 2020). A grande utilização de grupos de AGVs na industria é justificado pelo abastecimento de matéria prima nas linhas de produção como o recolhimento de peças e o seu transporte para o depósito, porém um grupo de robôs podem apresentar complexidades na sincronização de rotas e ao poupar suas baterias.

Um dos maiores desafios encontrados em robôs móveis se dá no planejamento de sua trajetória e tem recebido grande atenção por parte dos pesquisadores, tanto na indústria como no meio acadêmico, pois seu desenvolvimento está diretamente relacionado com a maior autonomia dos robôs, A complexidade do problema de planejamento do movimento tem motivado o desenvolvimento dos mais diversos algoritmos.

As rotas de robôs possuem os mais diversos formatos para diversas finalidades, a Figura 1 mostra 2 exemplos de rotas para robôs.

Desse modo, a complexidade advém da necessidade de integrar a navegação do

Figura 1 – Rotas para multiplos robôs

Fonte: (ACKERMAN, 2015; LEE *et al.*, 2019)

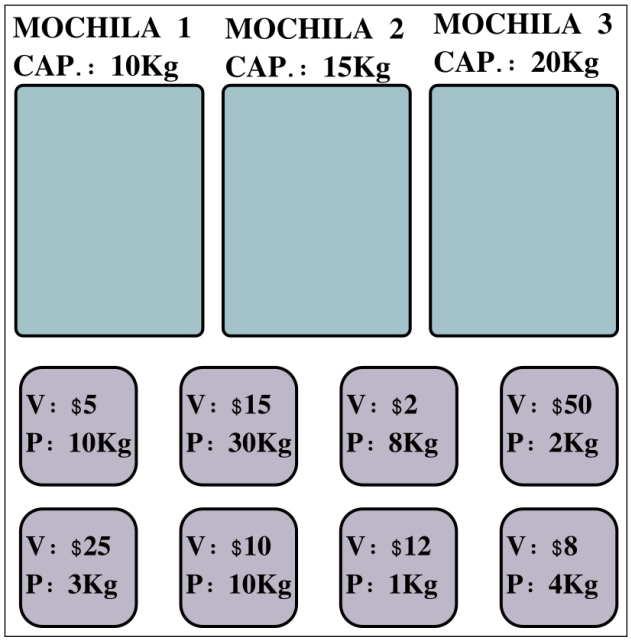
robô com o sensoriamento, a eficiência energética e o planejamento de rotas, como também a necessidade de poupar recursos importantes e preservar suas partes mecânicas. Grande parte das missões dos robôs que devem atender a vários pontos possuem custos para sua conclusão e pesos que determinam sua prioridade de atendimento. As missões realizadas por robôs ainda podem ser múltiplo agentes, no qual os agentes estarão em diferentes localidades e participarão das missões de forma colaborativa, o que acaba por aumentar a complexidade do problema (MURANO *et al.*, 2015).

Uma reconfiguração aplicável ao cálculo de rotas para robôs colaborativos com restrições energéticas encaixa-se no Problema de Roteamento de Veículos (PRV) baseado na coleta de prêmios com restrições de locomoção, encaixa-se mais precisamente no Problema de Orientação de Equipes (POE), mais conhecido como *Team Orienteering Problem* (TOP) citado pela primeira vez em (BUTT; CAVALIER, 1994). O problema consiste em uma coleção de pontos, cada um com uma premiação, que devem ser visitados por múltiplos agentes idênticos com a mesma restrição de locomoção. O objetivo é recolher a maior premiação possível sem visitar um ponto duas vezes e sem ultrapassar a restrição de locomoção. Este problema tem recebido grande visibilidade em aplicações de logística e no cálculo de rotas para robôs.

O TOP e o cálculo para rotas de múltiplos robôs possuem características semelhantes, ambos possuem mais de um agente para a execução das missões, os objetivos possuem pontuações para definir a prioridade de atendimento e os agentes possuem limitações de locomoção. Apesar das semelhanças, o TOP ainda carece de uma característica essencial para o cálculo de rotas para robôs, que é a capacidade de aproveitamento de restrição por agente. No TOP todos os agentes possuem a mesma restrição, o que acaba não atendendo problemas reais de cálculo de rotas para múltiplos robôs, já que um conjunto de robôs podem possuir restrições diferentes, como capacidade de bateria únicas para cada robô.

O Problema das Mochilas Múltiplas (PMM) consiste em um conjunto de item cada um com um valor e um peso e um conjunto de mochilas cada uma com uma capacidade, o objetivo é coletar os itens e colocá-los nas mochilas de modo que seja recolhido o maior valor possível sem exceder a capacidade das mochilas (KELLERER *et al.*, 2004). A Figura 2 mostra uma representação gráfica do PMM, no qual é composto por 3 mochilas com diferentes capacidades e uma coleção de itens que deverão ser colocados nas mochilas.

Figura 2 – Problema da Mochila Multipla



Fonte: Do Autor

Atendendo a restrição do planejamento de rotas para múltiplos agentes com restrições energéticas, este trabalho irá fornecer duas soluções para o cálculo de rotas para múltiplos robôs com restrições energéticas, a primeira solução utiliza um algoritmo genético baseado na solução de (Bederina; Hifi, 2017) e será referenciado neste trabalho como AG-Trad e o AG proposto como AG-Prop, e um algoritmo de Otimização por Enxame de Partículas (OEP), onde será referenciado como OEP-Prop. Os métodos propostos diferem de (Bederina; Hifi, 2017) pela combinação do PMM ao TOP para realizar a inserção da restrição de custos distintos para cada robô, no qual será referido como problema de orientação de equipes com múltiplas restrições (POEMR), de modo a limitar cada agente a um custo específico e adicionar a capacidade de mudança de depósitos de origem e destino, tornando a solução mais próxima da realidade. A utilização do PMM é feita para dar a capacidade de restrição única para cada robô. Trata-se de um problema de elevada complexidade em termos de otimização e requer a utilização de uma metodologia heurística apropriada à resolução de problemas NP-completos. A função objetivo

será modificada para uma função multi objetivo proposta por (CANDIDO, 2015) que realiza o cálculo de pontos de aptidão através da atribuição de pesos para o objetivo de custo e premiação, tornando a solução mais adaptável a diversas aplicações.

A comprovação da solução proposta foi realizada em um ambiente próprio de testes possuindo mapas simétricos de quarenta a oitenta pontos a serem visitados, cada ponto possui uma premiação. Os testes foram executados com diferentes números de robôs e diferentes custos aplicados.

O cálculo de rotas para robôs não é um campo de pesquisa novo, porém diversos conceitos surgiram para os mais variados planejamento de trajetórias como planejamento de rotas em um mapa para robôs móveis(ACKERMAN, 2015) ou uma linha de montagem para o encaixe de peças para manipuladores robóticos (LEE *et al.*, 2019).

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo geral é modelar um algoritmo para o cálculo de rotas para robôs colaborativos com restrições energéticas utilizando como modelo uma nova variante do problema de orientação de equipes que adiciona características do problema da mochila múltipla, no qual presentemente será denominada como problema de orientação de equipes com múltiplas restrições. Para atingir esse objetivo será desenvolvido uma meta-heurística.

1.1.2 Objetivos Específicos

- Desenvolver o modelo matemático do problema do cálculo de rotas para múltiplos robôs colaborativos com restrições energéticas;
- Desenvolver algoritmo genético na linguagem Python que apresente resultados ótimos ou próximos do ótimo.
- Comparar os resultados obtidos no cálculo de rotas entre o AG-Trad, o AG-Prop e o OEP-Prop.

1.1.3 Justificativa

Para que um robô seja cada vez mais autônomo e possua uma maior liberdade na execução de suas tarefas é necessário que ele não seja limitado a cabos para fornecimento de energia recorrendo assim as baterias. Contudo, com baterias que possuem uma carga energética maior ela também possuirá um maior peso ou um preço mais elevado (GROUP, 2017), desta forma é necessário encontrar um equilíbrio entre peso e capacidade energético da bateria a ser utilizada o que afetaria a mobilidade do robô. Sendo assim, é necessário otimizar a rota do robô para possuir um melhor aproveitamento.

Um grupo de robôs idênticos, mesmo possuindo baterias com especificações idênticas, podem apresentar diferentes capacidades energéticas, devido ao uso continuo de um único agente em comparação aos outros, ou idade devido a aquisição de robs posteriormente ou a mudança de baterias.

A importância do cálculo de rotas para múltiplos robôs energéticos, se justifica pela possibilidade de aumentar a autonomia dos robôs através de um melhor aproveitamento das limitações das suas baterias. Um método de cálculo de rotas eficiente pode ser aplicado em diversas áreas críticas como na industria 4.0 no uso de AGV ou em missões com Veículos Autônomos Não Tripulados (VANT).

1.2 Organização do Trabalho

O trabalho encontra-se estruturado em 6 capítulos, incluindo este “INTRODUÇÃO”, que apresenta uma visão geral das partes mais importantes da pesquisa, como a problemática, justificativa, objetivos e materiais e métodos.

O capitulo 2, "FUNDAMENTAÇÃO TEÓRICA", apresenta os principais conceitos utilizados para o completo entendimento da pesquisa, no qual envolve modelos matemáticos e métodos meta-heurísticos e as principais técnicas aplicadas em pesquisas relacionadas ao tema.

O capitulo 3, "REVISÃO BIBLIOGRÁFICA", apresenta as principais pesquisas que envolvem o tema.

O capítulo 4, "PROBLEMA DE ORIENTAÇÃO DE EQUIPES COM MÚLTIPLAS RESTRIÇÕES DE LOCOMOÇÃO E MUDANÇA DE BASE", é apresentado o modelo desenvolvido nesta pesquisa para o cálculo da trajetória dos robôs. Neste capítulo é apresentado a função multiobjetivo junto com as restrições necessárias para atender ao objetivo.

O capítulo 5, "AMBIENTE DE TESTES", mostra como foram realizados os testes. Neste capítulo são descritos as instâncias autogeradas para validação, assim como, os experimentos propostos para a execução de cada método e instância. O capítulo 5, "RESULTADOS E DISCUSSÕES", mostra o desenvolvimento final deste trabalho, onde serão observados os resultados e discutidos a eficiências dos métodos testados.

No capítulo 7, "CONCLUSÃO", contém alguns pontos de vista e observações de acordo com o trabalho desenvolvido. São apresentadas algumas sugestões de melhorias como trabalho futuro

1.3 Publicações Originadas da Pesquisa

A presente pesquisa gerou as seguintes publicações:

- SANTANA, K. A.; Pinto, V. P.; SOUZA, D. A; **Método heurístico multiobjetivo para o cálculo de rotas de robôs colaborativos com restrições energéticas**. In: Conferência Brasileira de Dinâmica, Controle e Aplicações (DINCON), 2019, São Carlos-SP. Anais do Evento, 2019.
- SANTANA, K. A.; Pinto, V. P.; SOUZA, D. A; **New GA Applied Route Calculation for Multiple Robots with Energy Restrictions**. In: 1st International Conference on Applied Technologies (ICAT), 2019, Quito - Equador. Anais do evento, 2019.
- Santana K.A., Pinto V.P., Souza D.A. (2020) Multi-robots Trajectory Planning Using a Novel GA. In: Rocha Á., Ferrás C., Montenegro Marin C., Medina García V. (eds) Information Technology and Systems. ICITS 2020. Advances in Intelligent Systems and Computing, vol 1137. Springer, Cham

Além das publicações apresentadas acima esta pesquisa também foi aceita e será apresentada no *International Conference on Information Technology Systems* (ICITS) em Bogotá-Colombia, nos dias 5 e 7 de fevereiro de 2020.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta alguns conceitos sobre o problema do caixeiro viajante, o *Orieentering Problem* , o *Team Orieentering Problem*, o *Problema da Mochila Múltipla*, o cálculo de rotas para múltiplos robôs e Algoritmos Genéticos.

2.1 Problema do Caixeiro Viajante (PCV)

O Problema do Caixeiro Viajante (PCV) tem sido muito utilizado no experimento de diversos métodos de otimização por ser, principalmente, um problema de fácil descrição e compreensão, mas de grande dificuldade de solução, uma vez que é NP-completo (DEINEKO; TISKIN, 2006). O PCV determina que um vendedor tem N cidades onde todas deverão ser visitadas, sem que haja repetição na visita e voltar para a cidade de partida de modo que o custo da viagem seja mínima. O PCV pode ser classificado em simétrico, a distância de uma cidade para outra não altera com a mudança de sentidos, e assimétrico, a distâncias entre cidades pode alterar com a mudança de sentidos. O PCV apesar de ser um problema muito conhecido tem sua origem incerta, sendo que problemas relacionados ao PCV foram citados em 1800 desenvolvidos por dois matemáticos: o escocês William Rowan Hamilton e o britânico Thomas Penyngton Kerkman. A forma geral do PCV parece ter sido utilizada pela primeira vez em Harvard e Viena em 1930. O problema foi finalmente conhecido globalmente em 1950 através dos estudos de Hassler Whitney e Merril Flood em Princeton(APPLEGATE *et al.*, 2007).

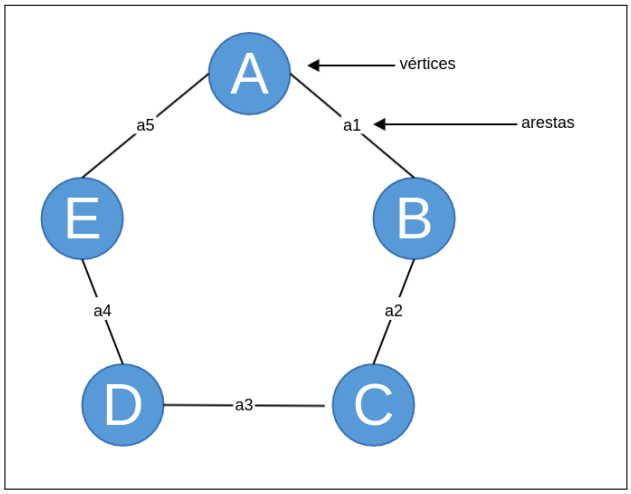
Diversas metodologias foram propostas para a resolução do PCV dentre elas métodos meta-heurísticos como (GOLDBARG *et al.*, 2008) que utiliza a otimização por enxame de partículas para solução do PCV, em (DORIGO, 1992) foi o primeiro registro da utilização da otimização por colônia de formigas. Já em (SIQUEIRA *et al.*, 2008) é utilizado redes neurais recorrentes. Também é possível a utilização de metodologias determinísticas como em (CHVÁTAL *et al.*, 1954) que foi umas das primeiras abordagens determinísticas na solução do PCV, neste método é utilizado programação linear (PL) para resolver a formulação inteira adicionando desigualdades lineares adequadamente escolhida para a lista de restrições. Em (HELD; KARP, 1962) foi apresentado uma formulação de programação dinâmica para uma solução exata do PCV, no entanto, possui uma complexidade de espaço muito alta, o que o torna muito ineficiente para valores mais altos de N[9].

2.1.1 Formulação Matemática

Existem diversas formulações matemáticas para a representação do PCV segundo (GOLDBARG; LUNA, 2000), pela sua representação simplificada será exibida a formulação de Dantzig-Fulkerson-Johnson. Para o teor desta pesquisa será exibida a formulação simétrica do problema.

Seja um grafo $G(N_C, A)$ onde N_c representa o conjunto de todas as cidades e A o conjunto de todas as arestas. Seja uma matriz simétrica com custos ou distâncias mínimas entre os nós da rede considerando que $c_{ij} = \forall i \in N_c$. A variável x_{ij} é uma variável binária que indica se um arco de i a j faz parte da rota. Uma representação de um grafo simples poder ser visualizado na Figura

Figura 3 – Grafo Simples



Fonte: Do Autor

Desta forma a formulação matemática é definida como (GOLDBARG; LUNA, 2000):

$$min \sum_{i=1}^{N_c} \sum_{j=1, i \neq j}^{N_c} c_{ij} x_{ij} \tag{2.1}$$

$$min \sum_{i=1}^{N_c} = 1 \qquad i = 2, 3, \ldots, N \tag{2.2}$$

$$min \sum_{j=1}^{N_c} = 1 \qquad j = 2, 3, \ldots, N \tag{2.3}$$

$$min \sum_{j=1}^{N_c} x_{ij} \leq |S| - 1 \tag{2.4}$$

$$X = x_{ij} \in S \qquad \forall |S| \subset N_c, S \neq 0 \tag{2.5}$$

$$X = x_{ij} \in \{1,0\} \tag{2.6}$$

A equação (2.1) tem como objetivo minimizar o custo da rota. Os dois primeiros grupos de restrições (2.2) e (2.3) garantem que exatamente um arco (i, j) tem origem cada nó i da rota e exatamente um arco(i, j) é direcionado para um nó j da rota. A penúltima restrição (2.6) contém um subconjunto S que pode ser qualquer conjunto de restrições que impeça a formação de sub-rotas. Estas restrições são chamadas restrições de quebra de sub-rotas.

2.1.2 Variações do PCV

O PCV é largamente utilizado em diversas aplicações do mundo real, sendo assim ao longo dos anos surgiram diversas variações do PCV, como é possível notar por meio das tabelas abaixo com algumas das suas variações:

Tabela 1 – Variações PCV

Pcvs Relatados	Na Literatura
PCV Assimétrico	PVC Simétrico
PCV com Coleta	PCV com Janelas de Tempo
PCV Múltiplo	PCV Seletivo
PCV com Passageiros	PCV estocástico
PCV com Seleção de Hotéis	Problema do transporte Solidário
PCV com Passageiros e Cotas	PCV Biobjetivo
PCV Seletivo Euclidiano	PCV com Gargalo

Fonte: (CHAVES; EXATA, 2003; GUTIN; PUNNEN, 2002)

Devido ao grande número de variações do PCV este trabalho focará nas variações mais importantes para a contextualização desta pesquisa que, no caso, é a o caixeiro viajante seletivo, porém existem muitas referências ao outro nome que o problema possui, que é o problema de orientação ou *orienteering problem*(OP) em inglês, e o problema do caixeiro viajante múltiplo.

2.2 Problema de Orientação (OP)

O problema de orientação,*Orienteering Problem* (OP) foi citado pela primeira vez em (GOLDEN *et al.*, 1987) e é definido como um conjunto de pontos onde cada ponto possui um prêmio e um agente com uma restrição de locomoção igual à T_{max}. O objetivo é realizar a maior coleta de pontos possíveis sem ultrapassar o custo máximo estabelecido. Desde sua criação diversas variações surgiram como o *Team Orientering Problem*, *Team Orienteering Problem with Time Windows*(TOPTW) e o *Time Dependent Orienteering Problem*(TDOP).

Este problema pode ser definido da seguinte forma: dado um conjunto de N_c vértices em que cada um tem uma recompensa não negativa P_i associada. O custo de locomoção c_{ij} entre o vértice i e o vértice j é conhecido para todos os pares de vértices. Em princípio pode não serem visitados todos os vértices, porque existe um custo de locomoção máximo T_{max}. O objetivo do OP é encontrar um caminho, limitado pelo T_{max} que visite alguns vértices, de tal forma que a recompensa seja maximizada. Assume-se que cada vértice só pode ser visitado uma vez, que o ponto inicial do caminho é o vértice 1, e que o vértice N é o ponto terminal do caminho.

2.2.1 Formulação Matemática

Segundo (KARA *et al.*, 2016) o OP também pode ser definido por um grafo $G = (N_c, A)$, onde $N_c = \{0, 1, 2, ..., n\}$ é o conjunto de todos os vértices, 0 é o depósito ou base e n é o último vértice presente no conjunto, e A é o conjunto de todas as arestas, onde $A = \{(i, j) : ij \in N, i \neq j\}$, no qual i é aresta de partida e j é a aresta de chegada. Cada aresta possui um custo de locomoção, esse custo pode ser simétrico, assimétrico, euclidiano, etc. Para cada vértice i é atribuído um prêmio p_i Todos os vértices possuem um premiação p atribuída. Cada arco a_{ij} possui um custo c_{ij}. O OP consiste em encontrar uma rota hamiltoniana de um vértice de origem e de destino coletando o maior prêmio possível sem exceder seu custo máximo de locomoção T_{max}. Cada arco é associado a uma variável binária x_{ij} que indica se o arco faz parte da solução.

Sua formulação matemática é definida como:

$$max \sum_{i=0}^{N_c-1} \sum_{j=1, i\neq j}^{N_c} p_i x_{ij} \tag{2.7}$$

$$\sum_{i=1}^{N_c} x_{0i} = 1 \tag{2.8}$$

$$\sum_{i=1}^{N_c-1} x_{in} = 1 \tag{2.9}$$

$$\sum_{i=0}^{N_c} x_{ij} \leq 1 \qquad , \quad j = 1,2,...,n-1 \tag{2.10}$$

$$\sum_{j=0}^{N_c} x_{ij} \leq 1 \qquad , \quad i = 1,2,...,n-1 \tag{2.11}$$

$$\sum_{i=0}^{N_c} x_{ij} = \sum_{i=1}^{N_c-1} x_{ji} \qquad , \quad j = 1,2,...,n-1 \tag{2.12}$$

$$\sum_{i=0}^{N_c-1} \sum_{j=1}^{N_c} c_{ij} x_{ij} \leq T_{max} \tag{2.13}$$

$$+ \textit{Regras para eliminao de Subrotas} \tag{2.14}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in A \tag{2.15}$$

Nesta formulação a equação 2.7 tem como objetivo maximizar os prêmios coletados da rota. As equações 2.8 e 2.9 são utilizadas para definir os depósitos de partida e destino da rota, já as restrições 2.10 e 2.11 asseguram que cada ponto possa ser visitado apenas uma vez. A restrição 2.12 é uma restrição de conservação de fluxo, a restrição 2.13 assegura que o tempo da rota seja menor ou igual ao valor máximo determinado por T_{max}. A restrição 2.14 é uma restrição implícita da forma de eliminação de sub-rotas formadas. Essas restrições devem garantir que a solução não contenha sub-rotas ilegais. As restrições de integralidade são fornecidas em 2.15.

2.3 Team Orienteering Problem (TOP)

O TOP, em português é conhecido com POE, é um problema de otimização combinatória que possui complexidade NP-difícil e foi estudado primeiramente por (CHAO *et al.*,

1996). O TOP é definido como um conjunto de pontos onde cada ponto possui um prêmio e um conjunto de agentes idênticos, onde todos possuem restrições de locomoção iguais à T_{max}, o objetivo é que todos os agentes possuam a mesma origem e destino e realizem a maior coleta de pontos possíveis sem ultrapassar o custo de locomoção máximo estabelecido e sem que um ponto seja visitado mais de uma vez. O TOP é uma variação do OP com múltiplos agentes.

2.3.1 Formulação Matemática

Segundo (DANG *et al.*, 2013) o TOP pode ser definido por um grafo $G = (N_c, A)$, onde $N_c = \{0, 1, 2, ..., N_c\}$ é o conjunto de todos os vértices e depósitos, 0 é o depósito ou base e n é o último vértice presente no conjunto, e A é o conjunto de todas as arestas, onde $A = \{(i, j) : i, j \in N_c, i \neq j\}$, no qual i é o vértice de partida e j é o vértice de chegada. Para cada vértice é atribuído um prêmio p_i, todos os vértices possem prêmios exceto os depósitos de partida e chegada. A variável N_k indica o total de agentes. A variável binária x_{ijk} indica se a aresta entre i e j foi visitada por um agente k. A variável binária y_{ik} indica se um vértice i foi visitado por um agente k, no qual 1 indica que o vértice foi visitado e 0 não.

Sua formulação matemática, segundo (DANG *et al.*, 2013), pode ser definida como:

$$max \sum_{i=1}^{N_c-1} \sum_{k=1}^{N_k} p_i x_{ik} \tag{2.16}$$

$$\sum_{k=1}^{N_k} \sum_{j=2}^{N_c} x_{1jk} = N_k \tag{2.17}$$

$$\sum_{k=1}^{N_k} \sum_{i=1}^{N_c-1} x_{j1k} = N_k \tag{2.18}$$

$$\sum_{i=1}^{N_c-1} x_{ni} = 0 \tag{2.19}$$

$$\sum_{i=1}^{N_c} x_{ij} \leq 1 \qquad j = 2,3,\ldots,N_c \tag{2.20}$$

$$\sum_{j=1}^{N_c} x_{ij} \leq 1 \qquad i = 2,3,\ldots,N_c \tag{2.21}$$

$$\sum_{i=1}^{N_c} \sum_{j=1}^{N_c} c_{ij} x_{ijk} \leq T_{max} \qquad k = \{1,2,...,N_k\} \tag{2.22}$$

$$1 \leq u_{ik} \leq N_c \qquad \forall i \in T_k \tag{2.23}$$

$$i_i - u_j + 1 \leq (1 + -x_{ijk})N_c \qquad 2 \leq i \neq j \leq N_c \tag{2.24}$$

Nesta formulação a equação (2.16) tem como objetivo maximizar os prêmios coletados da rota. As restrições (2.17) e (2.18) garantem que o mesmo número de agentes que saiam do depósito de saída é o mesmo número que chegue ao depósito de destino. A equação (2.19) garante que nenhum ponto seja visitado quando chegar ao depósito final. As restrições (2.20) e (2.21) restringe que cada vértice pode ser visitado apenas uma vez. A restrição (2.22) assegura que as rotas realizadas pelos agentes não ultrapassem o custo máximo estabelecido por T_{max}. As restrições (2.23) e (2.24) tem como objetivo eliminar sub-rotas.

2.4 Problema da Mochila Múltipla (PMM)

O PMM, ou em inglês *Multiple Knapsack Problem* (MKP), consistem em um conjunto de N_i itens, no qual cada item possui um valor e um peso, e um conjunto N_m de mochilas cada uma com uma capacidade. O objetivo é recolher os itens e colocá-los nas mochilas de modo que a soma dos valores dos itens devem ser a maior possível, sem exceder a capacidade de cada mochila.

2.4.1 Formulação Matemática

Segundo (KELLERER *et al.*, 2004) o MKP pode ser definido como um conjunto de itens $N = 1,...,N$ com prêmios p_i e com pesos w_i, onde $i = 1,...,N$ e um conjunto de mochilas $M = 1,...,M$ com capacidades $C_j = 1,...M$. Se um sub-conjunto $\tilde{N} \subseteq N$ é viável se os itens de $\tilde{N}$ podem ser colocados no conjunto de mochilas sem ultrapassar seus pesos, ou seja, se $\tilde{N}$ pode ser dividido em M conjuntos aplicados em N_i, de tal modo que $w(N_i) \leq c_i$ para $i = 1,...,M$. O objetivo é selecionar um subconjunto $\tilde{N}$, de tal modo que os prêmios de $\tilde{N}$ sejam maximizados.

Sua formulação matemática, segundo (KELLERER *et al.*, 2004):

$$max \sum_{i=1}^{M} \sum_{j=1}^{N} p_j x_{ij} \tag{2.25}$$

$$\sum_{j=1}^{N} w_j x_{ij} \leq C_i \qquad i = 1,...,M, \tag{2.26}$$

$$\sum_{i=1}^{M} x_{ij} \leq 1, j = 1,...,N \tag{2.27}$$

$$x_{ij} \in 0,1, \qquad i = 1,...,M, \quad j = 1,...,N \tag{2.28}$$

Na formulação acima a variável x_{ij} é uma variável binária que assume o valor 1 quando o item j estiver na mochila i e 0 caso contrário. Nesta formulação a equação 2.25 tem como objetivo coletar ao máximo os prêmio de todas as mochilhas. A restrição 2.26 garante que os itens coletados para uma mochila não ultrapassarão sua capacidade máxima. A restrição 2.27 garante que nenhum item possa ser colocado em mais de uma mochila.

2.5 Meta Heurísticas

Problemas de otimização consistem em achar a melhor combinação dentre um conjunto de variáveis para maximizar ou minimizar uma função, geralmente chamada de função objetivo ou função custo. Esses problemas podem ser divididos em três categorias: aqueles cujas variáveis assumem valores reais (ou contínuos), aqueles cujas variáveis assumem valores discretos (ou inteiros) e aqueles em que há variáveis inteiras e contínuas, classificados, respectivamente, como problemas de Otimização Contínua, Otimização Combinatória ou Discreta, e Otimização Mista. Para esta pesquisa é considerada apenas o problema de otimização combinatória(BECCENERI, 2008).

As meta-heurísticas são procedimentos desenvolvidos com o intuito de procurar uma boa solução, eventualmente ótima, consistindo na aplicação de uma heurística subordinada a cada passo, a qual deve ser modelada de forma específica a cada problema de otimização combinatória.

As meta-heurísticas possui como uma característica fundamental a sua capacidade de escapar de ótimos locais, assim, ampliando o espaço de busca de soluções. Outra característica peculiar das meta-heurísticas é a sua estrutura flexível e de propósito geral, a qual permite ser aplicável a diferentes tipos de problemas com relativamente poucas modificações em sua adaptação. Uma heurística pode ser considerada como a instanciação de uma meta-heurística em um problema específico de otimização (PORTO, 2010). A seguir, alguns exemplos de meta-heurísticas bastante utilizadas na resolução de problemas de otimização combinatória: Algoritmos Genéticos, Otimização por Enxame de Partículas, Colônia de Formigas, *simulated annealing*, *General Responsibility Assignment Software Patterns* (GRASP), Busca Tabu, Busca Harmônica e Colônia de Abelhas.

As próximas seções apresentam duas das técnicas mais utilizadas na resolução de problemas combinatórios os Algoritmos Genéticos e a Otimização por Enxame de Partículas.

2.6 Algoritmos Genéticos (AG)

Os Algorítimo Genético (AG) são técnicas de otimização inspiradas no princípio da sobrevivência e reprodução dos indivíduos mais aptos, proposto por Charles Darwin, (COPPIN, 2004; GOLDBERG, 1989).

O desenvolvimento de simulações computacionais de sistemas genéticos teve início

nos anos 50 e 60 através de muitos biólogos, mas foi John Holland que começou a desenvolver as primeiras pesquisas no tema. Em 1975, Holland publicou "*Adaptation in Natural and Artificial Systems*", ponto inicial dos AGs. David E. Goldberg, aluno de Holland, nos anos 80 obteve seu primeiro sucesso em aplicação industrial com AGs. Desde então os AGs são utilizados para solucionar problemas de otimização e aprendizado de máquinas(POZO *et al.*, 2005).

De acordo com (KRAMER, 2017) o algoritmo genético clássico é baseado em um conjunto de soluções candidatas que representam uma solução para o problema de otimização que se deseja resolver. Uma solução é uma possível candidata a um ótimo problema de otimização. Sua representação desempenha um papel importante, pois determina a escolha dos operadores genéticos. As representações são normalmente listas de valores e, geralmente, são baseadas em conjuntos de símbolos. Se eles são contínuos, são chamados vetores, se consistem em bits, são chamados de cadeias de bits. No caso de problemas combinatórios, as soluções geralmente consistem em símbolos que aparecem em uma lista. Os operadores genéticos produzem novas soluções na representação escolhida e permitem a caminhada no espaço da solução. A codificação da solução como representação, que está sujeita ao processo evolutivo, é chamada genótipo ou cromossomo. O algoritmo 1 mostra o pseudocódigo do algoritmo genético básico, que pode servir como modelo para muitas abordagens relacionadas. No início, um conjunto de soluções, denotado como população, é inicializado. Essa inicialização é recomendada cobrir aleatoriamente todo o espaço da solução ou modelar e incorporar conhecimentos especializados. A representação determina o processo de inicialização. Para representações de cadeia de bits, uma combinação aleatória de zeros e uns é razoável, por exemplo o cromossomo aleatório inicial 1001001001 como uma sequência de bits típica de comprimento 10. O principal ciclo geracional do algoritmo genético gera novas soluções candidatas à prole com cruzamento e mutação até que a população esteja completa.

O funcionamento básico de um AG pode ser visualizado no algoritmo 1.

2.6.1 Cromossomo

A representação do cromossômica é o primeiro passo para a implementação de um AG. Consiste na representação de uma instância do problema a ser resolvido. Em boa parte dos problemas de engenharia uma solução pode ser dada por um vetor real que especifica as dimensões dos parâmetros importantes do problema. A estrutura de uma solução poderá depender diretamente do problema abordado. Cada método de busca também possui características que

Algoritmo 1: Algoritmo Genético

```
início
    iniciar população;
    Enquanto teste de parada não for atingido para cada interação faça
        Enquanto população não estiver completa para cada membro faça
            avaliação;
            seleção;
            cruzamento;
            mutação;
        fim
    fim
fim
```

ajudam na especificação do tipo de representação a ser empregada(CASTRO; ZUBEN, 2015).

Os cromossomos podem possuir uma representação binária, codificação real, permutações e máquinas de estado finito.

2.6.2 Avaliação

No processo evolutivo a aptidão é o ponto principal da teoria de Darwin, o mesmo se aplica aos AGs. Nos AGs a aptidão de um indivíduo é dada através do conceito de função de aptidão, ou normalmente conhecido como *fitness*. O *fitness* mede quão próximo um indivíduo está da solução desejada ou quão boa é esta solução. O fitness de um indivíduo é calculado com base no problema proposto. É essencial que esta função seja muito representativa e diferencie na proporção correta as más soluções das boas. Se houver pouca precisão na avaliação, uma ótima solução pode ser posta de lado durante a execução do algoritmo, além de gastar mais tempo explorando soluções pouco promissoras.

Segundo(KRAMER, 2017) a aptidão de soluções pode ser guiada adicionando restrições de modo a desobedecê-las pode adicionar penalizações ao valor de aptidão, ou adicionando vários objetivos que precisam ser otimizados ao mesmo tempo, os valores da função de adequação de cada objetivo único podem ser agregadas.

2.6.3 Seleção

A operação de seleção permite a convergência em direção a soluções ideais, seguindo a teoria da evolução é necessário selecionar os melhores indivíduos em uma geração para serem os pais da próxima geração. A prole de uma geração sempre é gerada com indivíduos em excesso para selecionar os melhores indivíduos. Esse processo de seleção é baseado nos valores de aptidão na população. No caso de problemas de minimização, são preferidos baixos valores de adequação e vice-versa no caso de problemas de maximização. Problemas de minimização podem ser facilmente transformados em problemas de maximização com negação. Obviamente, isso também funciona para transformar problemas de maximização em problemas de minimização. A escolha de uma metodologia de seleção não é trivial, já que uma escolha errada pode vir a acarretar a convergência prematura da população.

Na literatura existem diversas técnicas de seleção. Entre elas as mais utilizadas segundo (CHUDASAMA *et al.*, 2011; YADAV; SOHAL, 2017; JEBARI, 2013) são a seleção de truncamento, roleta, amostragem estocástica universal, torneio, elitismo e seleção de Boltzmann.

2.6.3.1 Seleção de Truncamento

A seleção de truncamento é a técnica de seleção mais simples. Como o nome indica, ele usa truncamento para seleção. Nesta técnica, os indivíduos são classificados primeiro de acordo com sua aptidão. E os indivíduos acima do valor limite são selecionados por fazer parte do conjunto de acasalamentos e o restante dos indivíduos é truncado. Mas pode ser que o número de indivíduos selecionados não seja igual ao tamanho da população. Para manter o tamanho da população, são criadas cópias dos indivíduos selecionados. Por exemplo, se o tamanho da população for 80, podemos selecionar 25% dos indivíduos mais aptos dessa população, ou seja, 20 indivíduos serão selecionados, mas o tamanho da população é 80. Agora, para manter o tamanho da população, precisamos copiar quatro vezes os indivíduos mais aptos.

2.6.3.2 Seleção por Roleta

A seleção da roleta é a técnica de seleção proporcional ao *fitness* mais comum e mais simples. Cada indivíduo da população recebe uma seção de uma roda de roleta imaginária, proporcional à sua aptidão. O candidato mais apto tem a maior seção da roda e o candidato mais fraco tem o menor. A roda é girada n número de vezes, onde n é o tamanho da população e toda

vez que o indivíduo associado à seção vencedora é selecionado. A característica desse método de seleção é o fato de atribuir a cada indivíduo x da população atual uma probabilidade $p(x)$ de ser selecionada, proporcional à sua aptidão $f(x)$. Onde n indica o tamanho da população.

2.6.3.3 Amostragem Universal Estocástica

A Amostragem Universal Estocástica é semelhante à Seleção por Roleta, mas, diferentemente da Seleção por Roleta nesta técnica, existem n números de ponteiros, em que n representa o número de indivíduos a serem selecionados. Esses n ponteiros estão igualmente distantes. A distância entre dois ponteiros é $1/n$. A seleção é feita gerando um número aleatório apenas uma vez. O primeiro ponteiro do pente, como a montagem dos ponteiros equidistantes, move-se para o número aleatório gerado e o restante dos ponteiros move-se de acordo. Dessa maneira n, o número necessário de indivíduos é selecionado de uma só vez.

2.6.3.4 Seleção por Classificação

A seleção de classificação é semelhante à seleção de roleta, exceto pelo fato de que a probabilidade de seleção nesta técnica é proporcional ao condicionamento relativo em vez de condicionamento absoluto. Nesta técnica, a probabilidade de seleção de um indivíduo é mais em função da classificação do que da aptidão. Não faz diferença se o candidato mais apto é dez vezes mais apto que o próximo ou 0,1% mais apto. Em todos os casos, as probabilidades de seleção permanecem as mesmas, tudo o que importa é a classificação em relação a outros indivíduos. A seleção de classificação é um processo de duas etapas. A Seleção por Classificação primeiro classifica a população e então atribui a cada indivíduo um valor de adequação determinado pela sua classificação. O pior terá adequação igual a 1, o segundo pior 2 etc. de forma que o melhor terá adequação igual a N (número de indivíduos na população)

2.6.3.5 Torneio

Na seleção de torneios, cada indivíduo da população é emparelhado aleatoriamente com outro. Os valores de aptidão de cada par são comparados. O indivíduo mais apto do par passa para a próxima rodada, enquanto o outro é desqualificado. Isso continua até que haja um número de vencedores igual ao número desejado de pais. Então este último grupo de vencedores é emparelhado como pais de novos indivíduos. Nesta técnica de seleção, a seleção é

controlada pelo tamanho do torneio. Se cada vez que o tamanho do torneio for igual ao tamanho da população, o vencedor será sempre o mesmo.

2.6.3.6 Elitismo

A ideia do elitismo é preservar o melhor indivíduo de uma população. Devido ao operador de *crossover* e mutação os melhores indivíduos são perdidos. O objetivo do elitismo é não apenas preservar o melhor indivíduo, mas ao mesmo tempo permitir que ele participe da reprodução, a fim de propagar as melhores características para as próximas gerações.

2.6.4 Cruzamento

Se acordo com (PAVAI; GEETHA, 2016), desde o momento em que os AG foram inventados por Holland em 1992, os operadores de cruzamento, ou *crossover*, são significativos como sua força exploratória devido à sua capacidade de explorar soluções em uma área mais ampla do espaço de pesquisa. O cruzamento pode ser considerado idêntico à reprodução na evolução natural que mantém o processo de evolução. *Crossover* é a operação responsável pela criação de filhos, pela qual o AG pode explorar melhor o espaço de pesquisa e encontrar a solução globalmente ideal para o problema em consideração. A operação de cruzamento considera dois ou mais indivíduos da população como pais para formar um ou mais filhos, escolhendo genes de um dos pais escolhidos ou de uma combinação de ambos os pais. Diz-se que o operador de *crossover* é o operador mais importante no caso de AGs devido à hipótese básica. Essa hipótese sugere que certos elementos fundamentais importantes são responsáveis pela tarefa de produzir indivíduos altamente aptos na população. A operação de *crossover* é a operação mais adequada, na qual os indivíduos da população podem misturar e combinar os blocos de construção importantes (conjunto de genes) para formar indivíduos melhores.

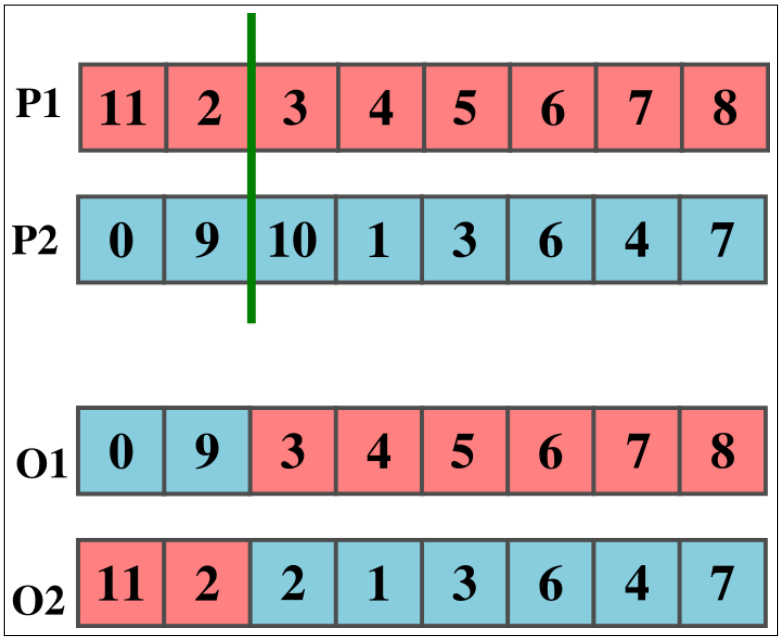
Os principais métodos de cruzamento na literatura, segundo (LIM *et al.*, 2017; HUSSAIN *et al.*, 2017) são o Cruzamento de Ponto Único, Cruzamento Múltiplo, Uniforme, *Partially Mapped Crossover* (PMX), *Order Crossover* (OX) e *Cycle Crossover* (CX).

2.6.4.1 Cruzamento de Ponto Único

Segundo (PAVAI; GEETHA, 2016; Bala; Sharma, 2015) o Cruzamento de Ponto Único é o método mais simples e frequentemente usado para o AG. Quando o cruzamento

é aplicado, os dois pais selecionados são divididos em um ponto de cruzamento escolhido aleatoriamente. Um único ponto de cruzamento na sequência de cromossomos de ambos os pais é selecionado. Todos os dados além desse ponto na sequência individual são trocados entre os dois indivíduos.

Figura 4 – Cruzamento de Ponto Único



Fonte: elaborado pelo autor (2019).

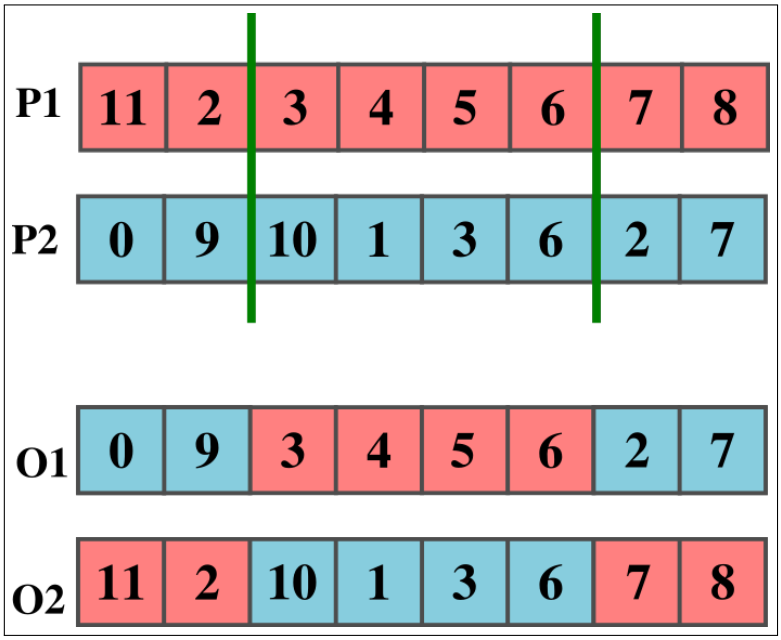
2.6.4.2 Cruzamento Múltiplo

Segundo (NERY, 2017; PAVAI; GEETHA, 2016) esse cruzamento aplica múltiplos pontos de cortes definidos aleatoriamente. As características genéticas contidas entre os pontos de corte são trocadas alternadamente entres os cromossomos progenitores.

2.6.4.3 Cruzamento Uniforme

Segundo (NERY, 2017; PAVAI; GEETHA, 2016) o Cruzamento Uniforme é semelhante ao cruzamento de Múltiplos Pontos, porém o número de pontos de cortes não é especificado a priori. Utiliza-se uma estrutura de máscara com o mesmo comprimento que os cromossomos e sorteia-se os genes que cada cromossomo progenitor fornecerá ao primeiro filho e o outro filho é gerado pelo complemento da máscara.

Figura 5 – Cruzamento Múltiplos



Fonte: elaborado pelo autor (2019).

2.6.4.4 Partially Mapped Crossover(PMX)

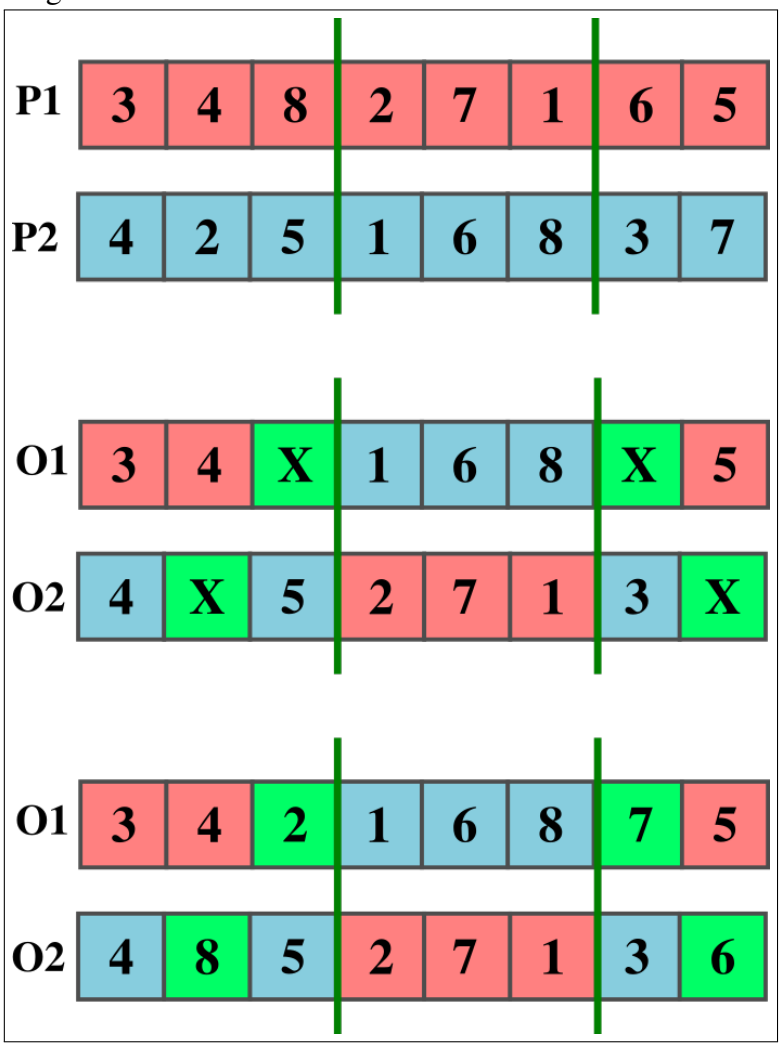
O PMX foi proposto por (GOLDBERG *et al.*, 1987). Depois de escolher dois pontos de corte aleatórios nos pais para criar filhos, a parte entre os pontos de corte, a sequência de um dos pais é mapeada na sequência do outro pai e as informações restantes são trocadas.

Para teor de demonstração considere, por exemplo, que dois pais percorrem aleatoriamente um ponto de corte entre a 3º e a 4º posição do vetor e outro ponto de corte entre a 6º e a 7º posição do vetor como mostra a figura 6.

As seções de mapeamento estão entre os pontos de corte. Neste exemplo, os sistemas de mapeamento são $2 \leftrightarrow 1$, $7 \leftrightarrow 6$ e $1 \leftrightarrow 8$. Em seguida as duas seções de mapeamento são copiadas entre si gerando dois novos descendentes, ou *offsprings*. Os espaços vazios são preenchidos com os dados dos pais, desde que não haja conflito.

Agora para o preenchimento dos espaços vazios verifica-se o primeiro descendente e o valor que 8 deveria ocupar a primeira posição vazia, que vem do primeiro pai, porém o 8 já existe na prole, então consultando o mapeamento nota-se que o valor fornecido é 1 e novamente esse valor já existe na prole, então a operação de consulta do mapeamento é feita novamente e chega-se a um valor não presente o 2 que irá ocupar a primeira posição vazia. A segunda posição vazia deveria ser ocupada pelo valor 6, porém o valor já existe, repete-se a operação de consulta do mapeamento e o valor retornado é igual a 7 que ocupa o espaço vazio, desta forma o primeiro *offspring* é terminado. A operação é análoga para o *offspring 2*.

Figura 6 – PMX



Fonte: elaborado pelo autor (2016).

2.6.4.5 Order Crossover Operator (OX)

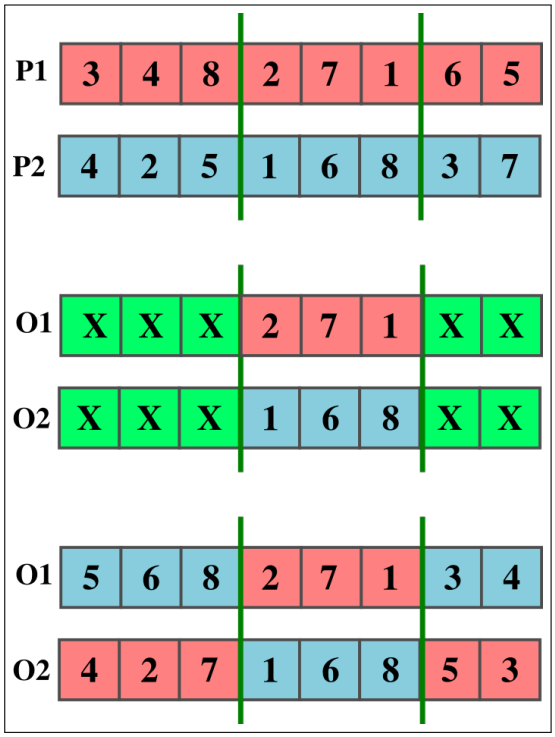
O cruzamento de ordens (OX) foi proposto por (DAVIS, 1985). Ele cria a descendência escolhendo uma *subtour* de um pai e preservando a ordem relativa de itens do outro pai.

Para teor de demonstração considere, por exemplo, que dois pais percorrem aleatoriamente um ponto de corte entre a 3^o e a 4^o posição do vetor e outro ponto de corte entre a 6^o e a 7^o posição do vetor como mostra a figura 7.

A prole é produzida da seguinte maneira. Primeiro, os itens entre os cortes dos pais são copiados da mesma maneira para a prole. Depois disso, a partir do segundo ponto de corte de um pai, os itens do outro pai são copiados na mesma ordem, omitindo os itens existentes. A sequência dos itens no segundo pai a partir do segundo ponto de corte é $3 \rightarrow 7 \rightarrow 4 \rightarrow 2 \rightarrow 5 \rightarrow 1 \rightarrow 6 \rightarrow 8$. Após a remoção dos bits $2, 7$ e 1, que já estão na primeira prole, a nova sequência é

$3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 8$. Essa sequência é colocada na primeira prole a partir do segundo ponto de corte. A operação é análoga para o *offspring 2*.

Figura 7 – OX



Fonte: elaborado pelo autor (2016).

2.6.4.6 Cycle Crossover Operator(CX)

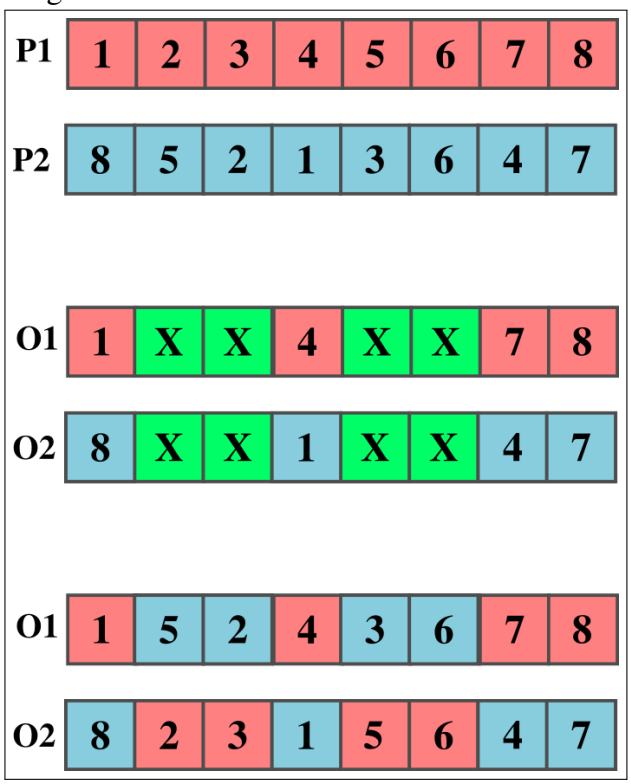
O operador de CX foi proposto pela primeira vez por (OLIVER *et al.*, 1987). O CX identifica um número dos chamados ciclos entre dois cromossomos pais. Em seguida, para formar o Filho 1, o ciclo 1 é copiado do pai 1, o ciclo 2 do pai 2, o ciclo 3 do pai 1 e assim por diante. Um ciclo é formado obtendo 1 valor do pai 1 e este valor será utilizado como índice para resgatar o próximo valor no pai 2, com o valor obtido do pai 2 utiliza-se como índice para resgatar o próximo valor do pai 1, esse processo é repetido até que se chegue no primeiro valor resgatado do pai 1, fechando um ciclo**.**

Para teor de demonstração considere, por exemplo, que dois pais executam o CX como mostra na figura 10.

O CX é iniciado obtendo o primeiro valor valor 1 no pai 1, $P1$, agora para obter o próximo valor é necessário obter o número no pai 2, $P2$, que está no posição 1, o valor obtido é 8, a operação é repetida para resgatar o valor que está na posição 8 de $P1$. O primeiro ciclo de $P1$ é $1 \rightarrow 8 \rightarrow 7 \rightarrow 4 \rightarrow 1$. Em seguida é necessário preencher o *offspring* 1 com os valores do

ciclo nas suas posições correspondentes de com $P1$. As posições vazias do *offspring* 1 deverão ser preenchidas com os valores presentes nas mesmas posições do $P2$. A operação é análoga para o *offspring 2*.

Figura 8 – CX



Fonte: elaborado pelo autor (2016).

2.6.5 Mutação

Segundo (KRAMER, 2017), um dos problemas comuns com algoritmos genéticos é a convergência prematura, diretamente relacionada à perda de diversidade. Atingir a diversidade populacional é uma meta desejada, pois o espaço de busca se torna melhor (diverso) de acordo e também evita uma solução sub-ótima. A mutação é considerada um mecanismo importante para manter a diversidade. Os operadores de mutação alteram uma solução, perturbando-os. A mutação é baseada em mudanças aleatórias. A força desse distúrbio é chamada taxa de mutação. Em espaços de solução contínuos, a taxa de mutação também é conhecida como tamanho da etapa.

O operador de mutação depende da forma de representação do cromossomo. Existem diversas formas de mutação na literatura, porém será apresentado neste trabalho operadores de mutação de permutação. As mutações mais utilizadas são mutação de inserção, inversão,

embaralhamento, *swap*, *bit flip*, uniforme, *creep*, *Worst Gene with Random Gene Mutation* (WGWRGM), *Worst Left and Right Gene with Random Gene Mutation* (WLRGWRGM) e *Swap Worst Gene Locally Mutation* (SWGLM).

2.6.5.1 Mutação por Inserção

A mutação por inserção é usada na codificação de Permutação. Primeiro, é escolhido aleatoriamente dois valores escolha dois valores de alelo aleatoriamente. Então o alelo na primeira posição é inserido na segunda posição e logo depois removido da posição anterior (SIVANANDAM; DEEPA, 2007).

2.6.5.2 Mutação por Inversão

A mutação por inversão é usada na codificação de Permutação. A mutação consiste na escolha de duas posições e a informação entre elas é invertida (SIVANANDAM; DEEPA, 2007).

2.6.5.3 Mutação por Embaralhamento

A mutação por embaralhamento, ou *scramble*, consiste na escolha aleatória de dois alelos, e reorganizar aleatoriamente todos os alelos que estão entre os alelos escolhidos (SONI; KUMAR, 2014).

2.6.5.4 Mutação Swap

A mutação *Swap*, ou troca, consiste na escolha aleatória de dois alelos e na troca de suas posições.(SONI; KUMAR, 2014)

2.6.5.5 Mutação Bit Flip

A mutação *bit flip* é utilizada em cromossomos com a representação binária. Esta mutação consiste na escolha de um alelo em um cromossomo binário e inverter seu valor de 0 para 1 ou 1 para 0.(SONI; KUMAR, 2014)

2.6.5.6 Mutação Uniforme

A mutação uniforme altera o valor do gene escolhido com um valor aleatório uniforme selecionado entre o limite superior e inferior especificado pelo usuário. É usado no caso de representação real e inteira. (SONI; KUMAR, 2014)

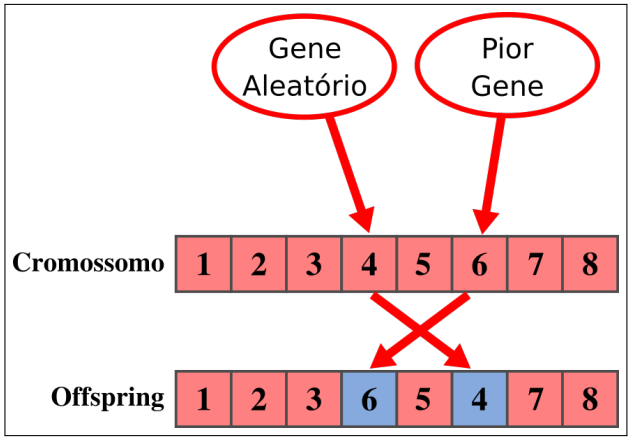
2.6.5.7 Mutação Creep

Na mutação *creep*, um gene aleatório é selecionado e seu valor é alterado com um valor aleatório entre o limite inferior e o superior. É usado em caso de representação real (SONI; KUMAR, 2014).

2.6.5.8 Worst Gene with Random Gene Mutation (WGWRGM)

O WGWRGM consistem em buscar o pior gene no cromossomo. O pior gene varia de acordo com a definição do pior para cada problema e cada método. O pior gene para um PCV é a cidade com o maior custo, enquanto para um problema da mochila o pior gene é o item com o menor valor $prmio/peso$. Depois de identificar o pior gene é realizada a seleção aleatória de outro gene e então os genes são trocados, semelhante a mutação de *swap* (HASSANAT *et al.*, 2016).

Figura 9 – WGWRGM



Fonte: (HASSANAT *et al.*, 2016)

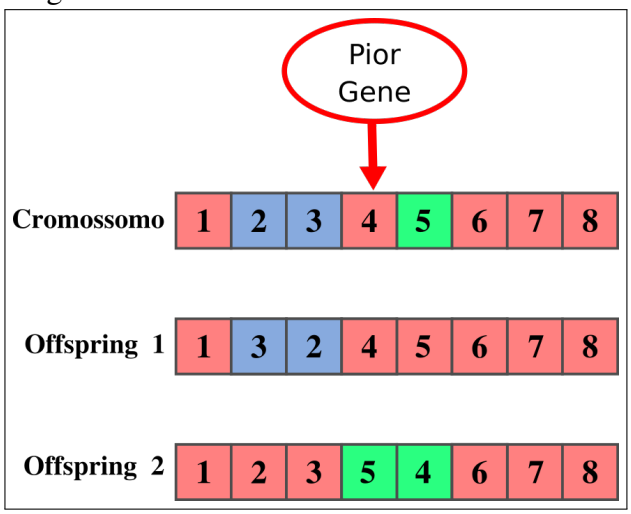
2.6.5.9 Worst Left and Right Gene with Random Gene Mutation (WLRGWRGM)

Esse método também é semelhante ao WGWRGM, mas a diferença é que o pior gene é aquele com a distância total máxima entre esse gene e os dois vizinhos - os da esquerda e os da direita. Considerando que as duas distâncias (esquerda e direita) podem ser mais informativo do que considerar apenas uma distância da esquerda ou da direita.(HASSANAT *et al.*, 2016)

2.6.5.10 Swap Worst Gene Locally Mutation (SWGLM)

Essa mutação se baseia em encontrar o pior gene usando WLRGWRGM e, em seguida, troca genes relacionados localmente, ou os vizinhos esquerdos são trocados ou o pior gene é trocado pelo vizinho direito. A melhor prole resultante decide quais genes serão trocados.

Figura 10 – SWGLM



Fonte: (HASSANAT *et al.*, 2016)

O WGWRGM consistem em buscar o pior gene no cromossomo. O pior gene varia de acordo com a definição do pior para cada problema e cada método. O pior gene para um PCV é a cidade com o maior custo, enquanto para um problema da mochila o pior gene é o item com o menor valor $prmio/peso$. Depois de identificar o pior gene é realizada a seleção aleatória de outro gene e então os genes são trocados, semelhante a mutação de *swap*. (HASSANAT *et al.*, 2016).

2.7 Otimização por Enxame de Partículas (OEP)

Bandos de pássaros, cardumes de peixes e manadas de animais constituem exemplos representativos de sistemas naturais em que comportamentos agregados são encontrados, produzindo movimentos sincronizados impressionantes, livres de colisões. Nesses sistemas, o comportamento de cada membro do grupo é baseado em respostas inerentes simples, embora seu resultado seja bastante complexo do ponto de vista macroscópico. (PARSOPOULOS; VRAHATIS, 2010) Esse comportamento de bando que inspirou o desenvolvimento da OEP, ou *Particle Swarm Otmization* (PSO). O PSO é um método estocástico de otimização baseado em população proposto por (KENNEDY, 2010) para tratar problemas no domínio contínuo.

De acordo com (PARSOPOULOS; VRAHATIS, 2010) o algoritmo de PSO é semelhante ao AG porém o PSO é composto por uma população, ou enxame, de partículas, onde cada partícula é uma representação de uma solução no espaço de busca de dimensão d. Para que o enxame sempre se aproxime do objetivo utiliza-se o *fitness* $(f(p))$, como no AG essa função irá avaliar o desempenho das partículas. Em um bando cada membro faz uso da sua própria experiência ou do bando para a busca de recursos, no PSO a experiência de cada indivíduo é representado pela variável *pBest*, já a experiência do enxame é chamada *gBest*.

O deslocamento das partículas no espaço de busca multidimensional é ajustado de acordo com sua experiência e com a de seus vizinhos a cada iteração do algoritmo. Onde $x_k(t)$ é a posição da partícula k no espaço de busca, na iteração t. A posição da partícula é atualizada por meio da adição da velocidade à sua posição atual. A partícula p_k irá se mover em uma determinada direção em função da posição atual e de uma velocidade $v_k(t+1)$. A velocidade da partícula é definida pelo seu melhor desempenho, *pBest*, e do melhor desempenho do bando, *gBest*. A velocidade de uma partícula é dada por (SERAPIAO, 2009):

$$v_k(t+1) = v_k(t) + \varphi_1 * (pBest - x_k(t)) + \varphi_2 * (gBest - x_k(t)) \tag{2.29}$$

onde: φ_1 e φ_2 são constantes definidas por (KENNEDY, 2010) para coeficientes "cognitivos"e "sociais".

Com a velocidade da partícula calculada a posição da partícula deverá ser alterada em função da posição atual e da nova velocidade:

$$x_k(t+1) = x_k + v_k(t_1) \tag{2.30}$$

O PSO define uma restrição de movimentação da partícula para que ela não ultrapasse o espaço de busca, tal restrição é definida como v_{max} para cada dimensão d no espaço de busca:

Algoritmo 2: Restrições PSO

```
if v_t > v_max then
    v_t = v_max
else
    if v_t < -v_max then
        v_t = -v_max
    end
end
```

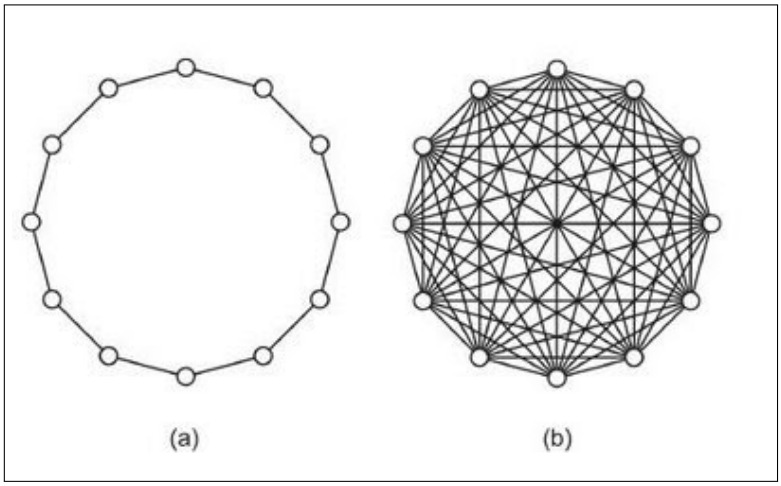
O funcionamento básico de um PSO pode ser visualizado no algoritmo :

Algoritmo 3: PSO

```
Iniciar enxame aleatório p;
Iniciar velocidades aleatória v;
while t < numeroTotalInteracoes do
    foreach particula no enxame do
        determinar pBest;
        determinar gBest;
        calcular velocidade v_t;
        calcula nova posicao de x_k;
    end
    t = t + 1;
end
```

Outro componente importante que influencia no desempenho do algorítimo PSO é a estrutura ou topologia da comunicação das partículas. Ela rege como as partículas do enxame trocam informações e se comunicam. Logo, a escolha da topologia influencia na avaliação da velocidade das partículas. A depender de como as partículas se comunicam entre si e do problema a ser tratado, a busca pela solução ótima pode priorizar tanto a velocidade de convergência, a qualidade da solução ou ambas. As principais topologias utilizadas como mecanismos de comunicação entre as partículas são: a topologia global e a topologia local. A figura abaixo apresenta a estrutura de tais topologias.

Figura 11 – PSO Topologia



Fonte: (PACHECO, 2016)

3 REVISÃO BIBLIOGRÁFICA

Este capítulo descreve alguns trabalhos da literatura mais relevantes para a contextualização do problema proposto nesta dissertação. Os trabalhos apresentados foram obtidos nos repositórios Researchgate, Elsevietr, IEEE, Springer e Google Sholar.

A seguintes seções serão categorizadas através dos temas de busca e serão apresentadas as palavras-chaves e o range de datas utilizadas na pesquisa.

3.1 Cálculo de rotas para robôs

Os trabalhos apresentados nesta seção foram obtidos através das palavras *Path Planning*, *Robots*, *Restrictions* e *Meta-heuristic*. O foram pesquisados artigos entre os anos de 2009 à 2019.

Em (BERG *et al.*, 2009) é pesquisado o cálculo de rotas para múltiplos robôs com obstáculos de modo que os robô evitem colisões nos obstáculos e entre eles mesmos. Nesta pesquisa o autor utiliza um algoritmo para desacoplar um planejamento de rotas em subproblemas, de modo que estes subproblemas possam ser resolvidos de forma sequencial por um planejador externo. Dado um planejador de caminho externo para espaços de configuração geral, o algoritmo encontra uma sequência de execução que minimiza a dimensão do subproblema de maior dimensão em todas as sequências de execução possíveis. Nesta proposta o algoritmo só estará completo caso o planejador externo também estiver, o algoritmo pode dissociar e resolver problemas de planejamento de caminhos com muitos robôs, mesmo com planejadores externos incompletos. O autor demonstra a eficiência do método através de cenários que envolvem de 16 a 65 robôs, nos quais os problemas de planejamento possuem tamanho de dimensões que variam entre 32 a 130.

Na proposta de (SILVEIRA *et al.*, 2012) é descrito um método de planejamento de rotas para vários robôs em ambientes desconhecidos utilizando dois algoritmos o D* e o algoritmo de Colonização do Espaço. A principal característica apontada pelo autor do algoritmo de Colonização Espacial é a troca de preferência por espaços, para evitar o risco de colisão, dada a incerteza e a escalabilidade de obstáculos móveis e fixos. A proposta do autor foca em ambientes lotados, a dificuldade de ter uma medida confiável da proximidade entre o robô e os obstáculos devem ser endereçados. O autor ressalta que alguns métodos que preferem ambientes livres, como campos em potencial, não garantem a localização dos caminhos (caem no mínimo

local), diferentemente do Espaço D*, que herda essa característica de D*. Nessa abordagem é adequada para uso em ambientes reais, concentrando-se na criação de caminhos através de ambientes com mais espaço livre.

A pesquisa de (PANDA *et al.*, 2015) se concentra no planejamento de trajetórias para robôs móveis em um ambiente estático desconhecido e composto. Segundo o autor o planejamento do caminho normalmente é feito de maneira offline, levando o conhecimento existente sobre o ambiente. Nesta pesquisa o melhor caminho é definido como sendo o caminho com custo mínimo que é constituído pelo caminho mais curto e sem colisões. Para realizar o cálculo desta trajetória é proposta uma solução que hibridiza duas meta-heurísticas o algoritmo genético e o sistema imunológico artificial, de modo que a combinação das técnicas supere as desvantagem apresentada por cada uma delas.

A pesquisa de (TANG *et al.*, 2016) propõe uma nova metodologia de cálculo de rotas de robôs utilizando um algoritmo de enxame de partículas híbridas, no qual faz a combinação com um algoritmo de evolução diferencial. O autor evita a estagnação das partículas aplicando rankings nas mesmas para uma melhor caminhada no espaço amostral.

O trabalho de (S. *et al.*, 2017) propõe o cálculo de rotas para um enxame de robôs inspirado no problema dos múltiplos caixeiros viajantes e considera para o planejamento de rota uma navegação autônoma para a coleta de dados que auxilie na navegação. Os testes foram aplicados em enxame com 8 robôs e com mapas que variam entre 20 e 40 *waypoints*.

Em (Araki *et al.*, 2017) foi criado um planejamento de rotas para robôs híbridos autônomos que tenham a capacidade de voar e se locomover por terra, os robôs voadores e terrestres que enfrentam dois conjuntos diferentes de desafios - os robôs voadores, particularmente os quadrirrotores, têm autonomia limitada da bateria, mas são rápidos e manobráveis, enquanto os robôs terrestres autônomos são energeticamente eficientes, porém limitados as ruas e ao transito. A proposta desta pesquisa apresenta um sistema de trajetória para a coordenação e controle de robôs híbridos em um ambiente urbano. Dois algoritmos de planejamento de caminhos com vários robôs são propostos, o *Safe Interval Path Planning* (SIPP) e um algoritmo baseado em programa linear inteiro ideal, ou *optimal integer linear program-based algorithm*. Uma pequena frota de quadriciclos de rodas com um design inovador foi construído. E, finalmente, um sistema para controlar esses quadrirrotores de rodas ao longo de suas trajetórias desejadas foi implementado. A demonstração final foi realizada por meio de quadriciclos viajando por via aérea e terrestre em uma cidade em miniatura, os resultados mostraram que esses algoritmos

podem de fato planejar caminhos livres de colisão para veículos que voam e dirigem.

Em (ALMEIDA *et al.*, 2017) é apresentado um método para planejamento de rota *offline* para um enxame de robôs que utiliza algoritmo genético e é inspirado no problema do caixeiro viajante múltiplo, ou *Multiple Travelling Salesmen Problem* (MTSP). O objetivo do enxame é navegar de forma autônoma em um ambiente semi desconhecido, coletar um determinado número de alvos que são espalhados de forma aleatória e alcançar um ponto pré-determinado de chegada. O planejamento de rotas desenvolvido neste pelo autor distribui sub-conjuntos de alvos a serem coletados por cada um dos agentes, de tal maneira que a distância total percorrida por todos os agentes seja minimizada e que nenhum alvo deixe de ser atribuído a um agente. De posse da rota a ser seguida, contendo os alvos e a ordem em que estes devem ser coletados, inicia-se a navegação autônoma dos agentes no ambiente. Durante esta navegação, o agente deve desviar-se de obstáculos estáticos e dinâmicos enquanto persiste em seguir de forma satisfatória a rota a ele determinada.

O trabalho de (PETRINIĆ *et al.*, 2017) trata-se de encontrar um perfil de velocidade no tempo ideal ao longo do caminho predefinido para formações estáticas de robôs móveis, a fim de percorrer o caminho no menor tempo possível e satisfazer, para cada robô móvel, na formação, velocidade, aceleração e restrições. O planejamento de velocidade ideal no tempo é conseguido usando o chamado controle bang-bang, onde as acelerações mínima e máxima da formação estão alternadas.

Na pesquisa realizada por (HAMEED, 2017) propõe um algoritmo de planejamento de rotas baseado no caminho de Dubins 2D para a construção de uma trajetória contínua para orientação de robôs de marcação de linhas em estádios de futebol. O algoritmo começa com quatro pontos que representam os antos do campo de futebol e gera um conjunto de *waypoints* de referência representando várias partes do layout do campo, como linhas de meio-campo, linhas laterais, linhas de fundo, grande área, meia lua, pequena área e tiro penal. Uma trajetória completa, contínua e suave é gerada ao conectar esses *waypoints* usando o caminho de Dubins 2D de forma a garantir que o caminho gerado leve em consideração as restrições dinâmicas do veículo (como curvatura e velocidade máximas), mantenha o veículo em a uma distância segura dos obstáculos e não prejudique a grama do campo. A eficiência do algoritmo é testada usando simulações e ambientes reais. Os resultados alcançados pelo autor mostraram que o algoritmo é capaz de planejar com segurança um caminho seguro em tempo real, capaz de comandar o robô de marcação de linha com alta precisão e sem a necessidade de orientação humana.

Em (MORAIS *et al.*, 2017) é desenvolvido um sistema de cálculo de rotas para evitar colisões, no qual utiliza da detecção de obstáculos e estimação de pose para assegurar a autonomia e a segurança do robô. A detecção de obstáculo é feita através de sensores RGB-Ds. A solução anti-colisão proposta faz uso do método de Campo Potencial Artificial, que é capaz de fazer um robô móvel passar em espaços estreitos minimizando as colisões. A validação do método foi feita em uma plataforma robótica "Turtlebot 2"executando 4 *experimentos evitar obstáculos, passagem* entre barreiras, passagem por baixo de obstáculos e contornando obstáculos.

Em (SERRANTOLA *et al.*, 2017) é proposto um método para planejamento de rota no espaço das juntas, com suporte a desvio de obstáculos estáticos, para um manipulador espacial de base livre flutuante com dois braços de forma a minimizar o movimento da base livre flutuante. Para isso, utilizou-se o conceito de Manipulador Dinamicamente Equivalente para modelar o manipulador de base livre flutuante como um manipulador de base fixa sub-atuado. O planejamento de movimento cinemático foi feito utilizando o algoritmo RRT*.

No trabalho de (LIU; BUCKNALL, 2018) é focado no planejamento de rotas para robôs autônomos de superfícies marinhas, ou *unmanned surface vehicles* (USVs). O autor propõe que as missões sejam iniciadas com a organização de tarefas a serem executadas e depois traça uma trajetória otimizada para atendê-las. Para realizar essas missões de maneira mais eficiente, é proposto um algoritmo híbrido eficiente de alocação de múltiplas tarefas e planejamento de caminhos. Em termos de alocação de múltiplas tarefas, um novo algoritmo baseado em um mapa auto-organizado, ou *self-organising map* (SOM) foi desenvolvido. A principal contribuição da pesquisa é a adaptação de um campo de força repulsivo artificial adaptativo que foi construído e integrado ao SOM para obter a capacidade de evitar colisões. O novo algoritmo é capaz de gerar rápida e efetivamente uma sequência para executar várias tarefas em um ambiente marítimo desordenado, envolvendo numerosos obstáculos. Após gerar uma sequência otimizada de execução de tarefas, um algoritmo de planejamento de caminho baseado no algoritmo de marcha rápida quadrática, ou *fast marching square*(FMS) proposto por (LIU; BUCKNALL, 2015), é utilizado para calcular as trajetórias. Devido à introdução de um parâmetro de segurança, o FMS pode ajustar adaptativamente a influência dimensional de um obstáculo e, consequentemente, gerar os caminhos para garantir a segurança do USV. Os algoritmos foram verificados e avaliados através de várias simulações e comprovando seu funcionamento eficiente em ambientes marítimos simulados e práticos.

No trabalho desenvolvido por (XU *et al.*, 2018) propõe dois algoritmos para o

planejamento de trajetórias ótimas para robôs móveis. O primeiro algoritmo é utilizado para calcular a trajetória e para evitar obstáculos utilizando o método de campo potencial artificial melhorado. Em seguida é proposto um algoritmo I-RRT* para planejamento do movimento, que combina o ambiente com as restrições de obstáculos, as restrições de veículos e as restrições cinemáticas.

A pesquisa realizada por (Sudhakara *et al.*, 2018) é focada no planejamento de trajetória para robôs móveis com rodas. Neste trabalho foi utilizado um campo potencial artificial aprimorado, no qual gera trajetórias que garantem a eficácia e a continuidade da trajetória. Diferente de outras pesquisas que envolvem campo potencial, o autor não foca na força de atração, mas sim na força repulsiva para desviar de obstáculos. O potencial repulsivo é construído pela função repulsiva para discretizar o contorno de um obstáculo de forma arbitrária com pontos. Isso descreve o espaço de trabalho do robô móvel com rodas com mais precisão. A confiabilidade é comprovada na maioria dos casos. Os resultados da simulação realizados confirmam a viabilidade do algoritmo proposto de campo potencial artificial aprimorado de que ele pode ser utilizado de maneira eficaz no planejamento de trajetória de robôs móveis com rodas e pode ser aplicado em cenários em tempo real.

Na pesquisa de (DEWANGAN *et al.*, 2019) é proposto um método de priorização de planejamento de trajetória para vários robôs, que busca o comprimento das rotas otimizadas através da técnica de força bruta. Nesta pesquisa o objetivo principal é produzir o menor comprimento de caminho para todos os robôs. Para isso, são propostas prioridades de todas as combinações de robôs para minimizar o tempo de computação. A eficiência é medida através de vários testes em ambientes, usando vários parâmetros de avaliação. Esse método proposto fornece o resultado ideal, pois apenas uma atribuição de prioridade para um único agente tem a possibilidade de levar ao pior resultado.

Na proposta de (PAPACHRISTOS *et al.*, 2019) é apresentada uma estratégia de planejamento de caminhos com reconhecimento de incerteza para alcançar a exploração robótica aérea autônoma de ambientes desconhecidos, garantindo a consistência do mapeamento em movimento. O planejador segue um paradigma de objetivos hierarquicamente otimizados, que são executados de maneira a retroceder no horizonte.

A pesquisa realizada por (Li, 2019) trabalha com planejamento de rotas para veículos subaquáticos autônomos, ou *Autonomous Underwater Vehicles* (AUVs). Os AUVs devem visitar vários nós com um caminho projetado. Nesta pesquisa propõe um novo método para projetar

caminhos 3D suaves para AUVs visitarem vários destinos. Nesta pesquisa o autor divide o método para resolver dois problemas, o problema de atribuição de destino e o problema de design de caminho. O problema de atribuição de destino foi modelado como um problema dos múltiplos caixeiros viajantes e resolvido através um AG. Para o design do caminho o autor descreve a complexidade no design devido continuidade das trajetórias planejadas no espaço 3D (ou seja, oceano) que é uma restrição importante. Para satisfazer essa restrição o autor utiliza a curva de Bezier para suavizar curvas bruscas e dar continuidade a rota. O cálculo da curva de Bezier 3D proposto nesta pesquisa faz com que os AUVs passem por vários alvos sem mudança de direção satisfazendo as restrições de menor rota.

Em (Hou *et al.*, 2019) é proposto um modelo de cálculo de rotas para um peixe robótico utilizando o algoritmo da colônia de formigas para o cálculo da rota. O autor ainda realiza uma modificação na rota final para trocar curvas acentuadas por pequenos arcos para permitir que o peixe robótico se mova normalmente.

A pesquisa de (FERNANDES *et al.*, 2019) é focada no planejamento de rotas para robôs de inspeção que não usam bateria, mas sim um cabo("cordão umbilical") que fornece energia e que possui um comprimento limitado. Para o cálculo de rota é proposto um algoritmo da colônia de formigas combinado com a técnica conhecida como algoritmo cultural em um robô com a limitação de um cordão umbilical em um curto espaço de tempo.

O trabalho de (NAGARAJAN *et al.*, 2019) aborda a problemática da inspeção das linhas de transmissão usando robôs suspensos, e propõe um algoritmo de roteamento para identificar onde a estação de controle precisa ser implantada ao usar o robô suspenso na inspeção da linha de transmissão. O movimento e a localização entre o robô e a equipe de terra são coordenados. A transmissão de dados bidirecional entre os dois lados deve ser garantida, de modo que a comunicação em tempo real seja garantida para que o robô possa ser controlado, obstáculos de grandes dimensões possam ser eliminados de forma eficiente e que os dados da condição de saúde da linha de transmissão sejam obtidos. Para realizar a validação do seu método o autor utiliza um estudo baseado em um segmento das linhas de transmissões no estado do Missouri, nos Estados Unidos, e sua solução é implementada para validar a eficácia do algoritmo proposto. Os resultados apresentaram um bom desempenho dos aspectos econômicos e ambientais com uma melhora significante em comparação com o modo de inspeção tradicional, empregando helicópteros e homens na manutenção das linhas.

3.2 Problema de Roteirização de Veículos

Os trabalhos apresentados nesta seção foram obtidos através das palavras *Vehicle Routing Problem*, *Restrictions*, *Vehicle* e *Meta-heuristic*. O foram pesquisados artigos entre os anos de 2009 à 2019.

Em (Yu *et al.*, 2014) é proposto uma nova extensão do OP, chamado Problema de Orientação Correlacionada ou *Correlated Orienteering Problem* (COP). Neste artigo as rotas são criadas em um ambiente com correlações espaciais, onde os passeios são restritos a um orçamento fixo de duração ou tempo. A principal característica do COP é uma função de utilidade quadrática que captura correlações espaciais entre pontos de interesse próximos um do outro. O autor utiliza programação quadrática inteira mista, *mixed integer quadratic programming* (MIQP), para o cálculo de rota.

O trabalho desenvolvido por (BRAGA *et al.*, 2014) estuda o uso de uma versão modificada do algoritmo de busca *Anytime Repairing A** (ARA*), para o planejamento de rotas em redes viárias reais, que faz uso de um sistema de monitoramento de trânsito. Para o estudo foram implementados os algoritmos *anytime ARA** e ITS-ARA* (uma versão de ARA* para ITS), além dos algoritmos de Dijkstra e A*. Os cenários utilizados foram os mapas das cidades de Manaus e do Rio de Janeiro.

O cálculo de rotas não está focado apenas na atribuição de destino, mas também para cálculo de rotas emergenciais em caso de falhas como mostrado por (ARANTES *et al.*, 2016), no qual utiliza técnicas de computação evolutiva como AGs e AGs Multi-Populacional(AGMP), além de uma heurística gulosa e um modelo de programação linear mista no tratamento de falhas críticas juntamente com o conceito de *In-Flight Awareness* (IFA).

Diversas pesquisas focam no melhoramento de rotas para economia de recursos, porém no trabalho realizado por (Rabbani *et al.*, 2017) é focado na melhora da rota para uma menor emissão de gases nocivos de veículos, este problema é chamado de problema de roteamento de poluição ou *Pollution Routing Problem* (PRP). O objetivo abordado deste problema é minimizar o custo das mudanças de carga, velocidade e pagamento aos motoristas, considerando a poluição emitida pelo veículo. O autor apresenta dois algoritmos meta-heurísticos, o algoritmo genético (AG) e o *Simulated Annealing* (SA), que são selecionados para resolver o problema apresentado, juntamente com um algoritmo meta-heurístico híbrido.

O trabalho de (YAN, 2018) tem como foco a solução do problema de roteamento de veículos autônomos considerando o conforto do piloto, pois sua satisfação está ligada diretamente

com a qualidade da rota executada, para isso o autor usa um algoritmo de *Ant Colony Otimization* (ACO). O autor primeiro considera o melhor tempo conseguido por um piloto humano e considera este tempo como base para melhorar a qualidade do voo autônomo, de modo a considerar de forma abrangente o custo autônomo e a satisfação do piloto. Em seguida, um algoritmo ACO paralelo é usado para otimização global e aplicado *mapreduced*, para melhorar o desempenho do algoritmo e obter a solução global ideal. Por fim, é conduzido um experimento de contraste e os resultados mostram que o algoritmo proposto pode efetivamente reduzir o tempo de voo do piloto e ajudar a melhorar a qualidade do serviço.

Novas abordagens evolutivas vêm sendo propostas no planejamento de rotas, como em (Orozco-Rosas *et al.*, 2019) no qual propõe um algoritmo de planejamento de caminho híbrido baseado no campo de potencial da membrana pseudo-bacteriano (MemPBPF). Algoritmos inspirados em membranas podem atingir um comportamento evolutivo baseado em processos bioquímicos para encontrar os melhores parâmetros para gerar um caminho viável e seguro.

O trabalho de (ZHANG *et al.*, 2019) trata do problema de roteamento difuso para carros elétricos e estações de recarga com janelas de tempo, ou *Fuzzy Electric Vehicle Routing Problem with Time Windows and Recharging Stations* (FEVRPTW). Para resolver este problema o autor utiliza um modelo de geração de números *fuzzy* para indicar as incertezas do carregamento da bateria do carro, estas incertezas envolvem tempo de recarga, tempo de consumo e tempo de viagem, inclusive o autor considera a recarga parcial da bateria como uma incerteza. Para verificar a eficiência do modelo foram desenvolvidos três algoritmos o *Adaptive Large Neighborhood Search*, um de seleção *fuzzy* e um de *variable neighborhood descent*.

Em (WANG *et al.*, 2019) para resolver o cálculo de trajetória de um grupo de veículos com carga que precisam visitar um grupo de clientes e minimizar o custo total do transporte, é proposta uma abordagem de computação paralela baseada nas operações do DNA. Com base no modelo de computação bio-heurística e manipulações moleculares de DNA, algoritmos de biocomputação paralela para resolver problemas de roteamento de veículos capacitados são propostos neste artigo. O autor dedica-se a utilizar apropriadamente diferentes cadeias biológicas para representar os vértices, as arestas, os pesos e adotamos operações biológicas apropriadas para pesquisar as soluções do problema com complexidade de tempo $O(n^2)$. Para aumentar o escopo de aplicação da biocomputação, foi reduzido a complexidade computacional e verificado a praticabilidade de algoritmos paralelos de DNA por meio de simulações.

A pesquisa feita por (ALTABEEB *et al.*, 2019) utiliza do algoritmo meta-heurístico

do vaga-lume para calcular a rotas de veículos com carga. Este método apresenta uma desvantagem de ficar preso facilmente em soluções ótimas locais, porém o autor propõe a modificação deste método aplicando dois tipos de buscas locais e a aplicação de operadores genéticos para aumentar a qualidade dos resultados. Para realizar a validação do método o autor executa 82 instâncias de benchmark. A pesquisa demonstrou uma taxa de convergência rápida e resultados promissores.

No trabalho proposto por (ZUO *et al.*, 2019) é estudado o problema de roteamento de veículos elétricos com janelas de tempo, onde o autor considera novos recursos de veículos elétricos, como capacidade de bateria limitada, falta de infra-estruturas e um longo período para o carregamento. Neste estudo foram apresentadas novas formulações técnicas para a seleção de rotas considerando as estações de carregamento, o que reduz a complexidade da formulação sem o uso de nós ou arcos fictícios duplicados. O autor também desenvolveu um novo método de linearização que emprega um conjunto de linhas secantes para substituir a função de carregamento não linear côncavo com restrições lineares. Este método define o tempo de carregamento como uma variável contínua e diminuindo o número de variáveis presentes nas formulação existente na literatura. Um modelo de programação linear com número inteiro misto, ou *Mixed-Integer Linear Programming* (MILP), foi desenvolvido para o problema e experimentos computacionais nas instâncias de (SOLOMON, 2008) foram realizados para validar o modelo proposto.

As pesquisas focadas na otimização de rotas, em sua grande maioria, focam-se na minimização dos custos de transporte. No entanto, em ambientes práticos, muitas empresas também prestam atenção em como a carga de trabalho é distribuída entre seus veículos. Neste contexto a pesquisa conduzida por (LEHUéDé *et al.*, 2020) propõe modelar a equidade do objetivo de menor rota e balanceamento de cargas entre vários veículos. Para realizar este balanceamento é proposto uma abordagem lexicográfica de minimax, ou *lexicographic minimax approach*, enraizada na teoria da escolha social. Para realizar a validação do modelo proposto é realizada uma comparação com uma heurística baseada na estrutura de pesquisa local multidirecional, além da comparação com diversas abordagens *Large Neighborhood Search* (LNS).

A pesquisa realizada por (ESHTEHADI *et al.*, 2020) estuda o problema de roteamento de veículos com compartimentos múltiplos com um único depósito dentro de uma janela de tempo para atender os clientes. A proposta desta pesquisa utiliza um algoritmo *Adaptive Large Neighborhood Search*. O ambiente de testes foram as caixas postais da cidade de Londres que selecionadas aleatoriamente formaram mapas que continham de 10 a 200 caixas postais.

4 PROBLEMA DE ORIENTAÇÃO DE EQUIPES COM MÚLTIPLAS RESTRIÇÕES DE LOCOMOÇÃO E MUDANÇA DE BASE (POEMRLMB)

O Problema de Orientação de Equipes com Múltiplas Restrições de Locomoção e Mudança de Base (POEMRLMB) é a proposta deste trabalho que consiste em uma variação do POE combinado ao PMM, com esta combinação é possível dar a cada agente do POE um custo único de locomoção o que garante um melhor aproveitamento do problema na área de cálculo de rotas para robôs, também é adicionado ao problema a capacidade de mudança de depósitos de cada agente.

Considere um grafo $G(N,A)$, onde N representa o conjunto $(|N| = N_c)$, no qual N_c é o conjunto de todos os *waypoints*, e A o conjunto de arestas. Seja, uma matriz simétrica com custos ou distâncias mínimas entre os nós da rede considerando ainda que $c_{ij} = \forall i \in N_c$. A solução pode utilizar N_k robôs, no qual cada robô possui um custo máximo C_{kmax}. Os *waypoints* x_i que compõe o conjunto N_c possuem um prêmio p_i. O POEMRLMB utiliza variáveis binárias x_{ik} para indicar que um *waypoint* x_i foi visitado por um robô k, no qual 1 indica que o *waypoint* foi visitado e 0 não. A variável S_k é um conjunto de *waypoints* pertencentes a uma sub-rota do robô k. A matriz $X[x_{ijk}]$ é compostas por variáveis binárias que indicam se um arco x_{ij} faz parte da rota realizada pelo robô k.

Desta forma, a formulação de matemática pode ser definida como (KELLERER *et al.*, 2004; CANDIDO, 2015) :

$$min\left(\alpha\sum_{i=1}^{N_c}\sum_{j=1,i\neq j}^{N_c}\sum_{k=1}^{N_k} c_{ij}x_{ijk} - \beta\sum_{i=1}^{N_c}\sum_{k=1}^{N_k} p_i x_{ik}\right) \tag{4.1}$$

Sujeito à:

$$\sum_{i=1}^{N_c-1} x_{ni} = 0 \tag{4.2}$$

$$\sum_{i=1}^{N_c} x_{ij} \leq 1 \qquad j = 2,3,\ldots,N_c \tag{4.3}$$

$$\sum_{j=1}^{N_c} x_{ij} \leq 1 \qquad i = 2,3,\ldots,N_c \tag{4.4}$$

$$\sum_{i=1}^{N_c}\sum_{j=1}^{N_c}\sum_{k=1}^{N_k} c_{ij}x_{ijk} \leq C_{k_{max}} \tag{4.5}$$

$$1 \leq u_{ik} \leq N_c \qquad \forall i \in S_k \tag{4.6}$$

$$i_i - u_j + 1 \leq (1 + -x_{ijk})N_c \qquad 2 \leq i \neq j \leq N_c \tag{4.7}$$

Neste problema a função objetivo (4.1) trata-se de uma função multi-objetivo assim como em (Bederina; Hifi, 2017), na qual procura maximizar a coleta de prêmios e minimizar o custo de todos os agentes. Essa proposta utiliza uma versão modificada da função objetiva proposta por (CANDIDO, 2015). A função objetivo é formado por duas equações a primeira equação visa minimizar a rota e é subtraída pela segunda equação que visa maximizar a coleta de prêmios.A função possui duas variáveis α e β, nas quais possuem como funções primárias inalteradas como pesos para mínima rota e maximização de prêmios. A função de (CANDIDO, 2015) é uma função mono agente, então ela foi modificada pra ser múltiplo agente adicionando variável k. A restrição (4.2) impõe que nenhum ponto seja visitado após chegar ao ponto final, as equações (4.3) e (4.4) restringem que cada vértice pode ser visitado no máximo uma única vez, pois ainda existe a possibilidade que um vértice não pertença a nenhuma rota, a equação (4.5) é uma versão modificada da restrição de peso do PMM (KELLERER *et al.*, 2004), porém nesta versão no lugar de pesos são colocadas as distâncias $c_i j$ e os objetos são as arestas percorridas $x_i jk$, assim a soma dos custos percorridos por cada robô k poderá ultrapassar $C_{k_{max}}$, sub-rotas são impedidas de serem geradas pelas as equações (4.6) e (4.7). O resultado desta otimização será um conjunto de vértices ordenados a serem visitados pelos agentes.

4.1 Algoritmo Genético Proposto

Para solucionar o método proposto na seção anterior foi desenvolvido um AG. A função *fitness* utilizada foi a função objetivo 4.1. As subseções a seguir dedicam-se a descrever o AG proposto.

4.1.1 Estrutura do Cromossomo

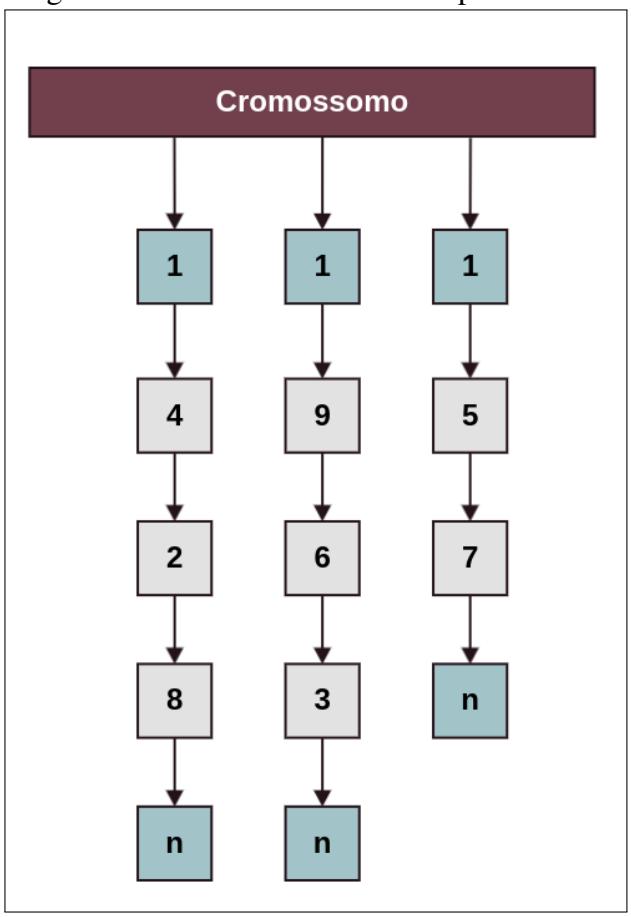
Como descrito em 2.6.1 o cromossomo é a representação de uma solução do problema a ser resolvido pelo AG. A proposta deste trabalho visa encontrar a menor rota para vários robôs em um conjunto de *waypoints*. A melhor representação para este problema é uma codificação de permutação.

Este trabalho utilizou a representação cromossômica proposta por (Bederina; Hifi, 2017). Nesta representação cada cromossomo é uma representação completa de uma solução. O cromossomo é um vetor composto por n genes, onde n é o número de agentes utilizado no POE e cada gene representa uma rota de um agente. Na proposta de (Bederina; Hifi, 2017) todos os genes iniciam a rota na mesma base e os genes finalizam na mesma base, porém a representação original não atenderia ao objetivo deste trabalho, pois a base de início e de finalização podem ser diferentes. Desta forma, para atender o modelo proposto por esta pesquisa, a representação cromossômica de (Bederina; Hifi, 2017) foi alterada para que cada agente possa iniciar e finalizar em uma base diferente. A representação cromossômica pode ser vista na Figura 12. O processo de gerar novos cromossomos são realizados em 3 etapas na inicialização da população, no cruzamento e na mutação. Além do cromossomo o AG-Prop possui um vetor de restrições que armazena a restrição de locomoção que cada agente possui, cada gene possui uma restrição de locomoção na mesma posição no vetor de restrições.

4.1.2 Inicialização da População

A inicialização da população é feita utilizando um algoritmo do vizinho mais próximo randômico. O algoritmo recebe uma lista de depósitos de início, lista de depósitos de fim, o número de agentes, uma lista de custos com a quantidade igual a de agentes e uma lista de *waypoint* disponíveis. O processo inicia com a seleção aleatório de um *waypoint i* não utilizado da lista de *waypoint* disponíveis, para o gene k. Depois que um *waypoint* é inserido em um gene o mesmo é removido da lista de disponibilidade. Em seguida é selecionado o *waypoint* mais próximo do último inserido. O processo se repete até que o custo do gene seja igual ou próximo do custo máximo do respectivo robô, C_{kmax}, ou quando não há mais *waypoints* disponíveis. Caso o custo máximo seja ultrapassado com a inserção do último *waypoint*, então a operação será desfeita e é iniciado o processo do próximo gene. Neste momento a obtenção de prêmios é desprezada, sendo assim o cálculo do *fitness* não é utilizado. A inicialização da população pode

Figura 12 – Cromossomo AG Proposto



Fonte: (Bederina; Hifi, 2017)

ser vista no algoritmo 4.

O processo é finalizado quando a quantidade de cromossomos, ou indivíduos, desejados da população é gerado.

4.1.3 Fitness

Com a população iniciada é dado o início a medição do seu *fitness*. O cromossomo é composto por vários genes e cada gene possui um valor de *fitness* único, já que o *fitness* é a avaliação de uma rota. O *fitness* de um cromossomo consiste da soma do *fitness* dos seus genes. O cálculo de *fitness* é o proposto nesta pesquisa e mostrado na equação 4.1 no qual é determinado um peso para o prêmio e um peso para o custo. O processo de medição pode ser visto no algoritmo 5.

Algoritmo 4: Vizinho mais próximo randômico

```
entrada: ListaCustos, ListaDepositoInicio, ListaDepositoFim, QuantidadeAgentes,
  VerticesDisponiveis;
saída: Cromossomo;
Início;
foreach A ∈ QuantidadeAgentes do
    Gene ← New;
    Inicio ← ListaDepositoInicio(A);
    Fim ← ListaDepositoFim(A);
    verticeI ← SelecaoAleatoria(VerticesDisponiveis);
    TmpGene ← Juntar(Inicio,Gene,Fim);
    CustoGene ← MedeCusto(TmpGene);
    N ← VerticesDisponiveis ;
    C ← ListaCusto(A);
    while CustoGene < C e Tamanho(N) > 0 do
        Remove(verticeI, VerticesDisponiveis);
        Insere(verticeI,Gene);
        TmpGene ← Juntar(Inicio,Gene,Fim);
        verticeI ← VizinhoProximo(verticeI);
        CustoGene ← MedeCusto(TmpGene);
    end
    Gene ← TmpGene;
    Insere(Gene, Cromossomo);
end
Finalização
```

Algoritmo 5: Fitness

```
entrada: Cromossomo, CustosMapa, Premios, PesoCusto, PesoPremio;
saída: Fitness;
Início;
Fitness ← 0;
foreach g ∈ Cromossomo do
    P ← Premios(g)*PesoPremio;
    C ← CustosMapa(g)*PesoCusto;
    FitnessGene ← C - P;
    Fitness ← Fitness + FitnessGene;
end
Finalização;
```

4.2 Cruzamento

Com a população com o *fitness* calculado é iniciada a seleção dos cromossomos para realizar o cruzamento. Com os métodos de seleção apresentados na seção 2.6.3 foi optado o método de torneio.

Uma caraterística importante no cromossomo apresentado é que seus genes podem possuir quantidades diferentes de *waypoints*, isso acontece devido aos custos entre o *waypoints* serem diferentes e as restrições de locomoção dos agentes também serem distintas, o que pode acarretar em um erro durante o *crossover*, por exemplo um cruzamento de 2 pontos entre genes pode ser maior que um dos genes selecionado. Desta forma esta pesquisa propõe um novo método de cruzamento. O processo inicia selecionando um gene aleatório de cada pai. Cada gene selecionado é removido os depósitos de início e de finalização. Logo após, para cada gene é gerado 2 pontos de recorte, esses pontos de recorte poderão ter tamanhos diferentes. Os recortes dos genes são removidos dos agentes. Por fim será inserido o recorte do Gene1 no Gene2 e vice-versa, a posição da inserção é realizada com base no tamanho resultante do Gene após o recorte, e os depósitos são novamente inseridos. Durante a troca dos recortes é realizado uma remoção dos *waypoints* presentes em outros genes para evitar duplicidade de visitas. Após as trocas serem realizadas os depósitos são novamente inseridos. Os filhos são gerados recebendo os genes não modificados do pai e o novo gene gerado pela operação. A operação de cruzamento pode ser vista no algoritmo 6. A Figura 13 apresenta de forma gráfica a operação de cruzamento, onde representam os três estágios para o cruzamento a seleção de genes, a troca de vértices entre os genes e a devolução dos novos genes gerados.

4.2.1 Mutação

Com o cruzamento finalizado é iniciado o processo de mutação da população. Uma parte da população sofrerá uma mutação para garantir a heterogeneidade dos indivíduos, para que a mutação seja aplicada em um AG é necessário utilizar um parâmetro de probabilidade que selecione os indivíduos aleatoriamente, este parâmetro será chamado de $PropMut$. A mutação de um cromossomo é realizada em todos os genes e o processo realizado nesta pesquisa consiste em 3 operações: uma operação de inserção, uma de deleção e a de mutação SWGLM que foi explicado na seção 2.6.5.10.

As operações de inserção e remoção são realizadas para garantir uma melhor di-

Algoritmo 6: Cruzamento

```
entrada: Pai1, Pai2, Start, End;
saída: Filho1, Filho2;
Início;
indiceGene ← SelecaoAleatoria(Pai1);
Gene1, Gene2 ← Pai1(indiceGene), Pai2(indiceGene);
Gene1, Gene2 ← RemoverDepositos(Gene1, Gene2);
Slice1, Slice2 ← Recorte(Gene1), Recorte(Gene2);
pos1, pos2 ← SelecaoAleatoria(Gene1,Gene2);
Gene1 ← Remove(Gene1, Slice1);
Gene2 ← Remove(Gene2, Slice2);
Gene1 ← Junção(Gene1, Slice2, pos1);
Gene2 ← Junção(Gene2, Slice1, pos2);
Gene1, Gene2 ← InserirDepostios(Gene1, Gene2, Start, End, indiceGene);
foreach g ∈ Indices(Pai1) do
    if g = indiceGene then
        Inserir(Gege1, Filho1);
        Inserir(Gege2, Filho2);
    else
        Inserir(RetornaGene(Pai1, g), Filho1);
        Inserir(RetornaGene(Pai2, g), Filho2);
    end
end
RemoverElementosRepetidos(Filho1);
RemoverElementosRepetidos(Filho2);
Finalização;
```

versidade da população inserindo novos *waypoints* nos cromossomos. A operação de inserção seleciona aleatoriamente um *waypoint* não presente no cromossomo, em seguida, é realizada uma permutação em cada posição do gene e medido o seu *fitness*, a posição que apresentar o melhor *fitness* será a posição final do novo *waypoint* no gene. A operação de deleção ocorre apenas se o custo do gene ultrapassar o custo máximo especificado para aquele gene e quando inciar será removido um *waypoint* e em seguida será calculado o seu *fitness*, o *waypoint* que possuir o menor *fitness* será removido do gene. As mutações de inserção e deleção podem ser visualizadas no algoritmo 7.

A mutação SWGLM foi selecionada com base no estudo realizado por (HASSANAT *et al.*, 2016), neste estudo o autor fez a comparação entre diversos métodos de mutação em um AG para resolver o PCV, os resultados demonstraram que o SWGLM apresentou a melhor performance, mesmo as pesquisas de (HASSANAT *et al.*, 2016) e este trabalho serem distintas as duas tratam de problemas de permutação, justificando sua escolha.

Algoritmo 7: Mutação inserção e Deleção

```
entrada: Cromossomo, CustoMaximo, ProbMut,VerticesDisponiveis;
saída: Cromossomo;
Início;
prob ← GerarValorRandomico(0,1);
if prob >= ProbMut then
    foreach g ∈ Indices(Cromossomo) do
        gene ← Cromossomo[g];
        c ← MedeCusto(gene);
        if gene < CustoMaximo[g] then
            v ←;
            SelecaoAleatoria(VerticesDisponiveis);
            pos ← PosicaoMelhorFitness(gene, v);
            novogene ← Insere(v,pos);
            if c < MedeCusto(novogene) then
                gene ← novogene
            end
        else
            v ← SelecaoAleatoria(geneg);
            pos ← PosRemoveMenorFitness(gene, v);
            gene ← Remover(v,pos);
        end
        Cromossomo[g] ← gene;
    end
end
Finalização;
```

4.3 Otimização por Enxame de Partículas

Para validação do POEMRLMB foi desenvolvido um algoritmo de Otimização por Enxame de Partículas (OEP), referido como OEP-Prop. A função *fitness* utilizada foi a função objetivo 4.1.1. As subseções a seguir dedicam-se a descrever o OEP-Prop.

4.3.1 Estrutura da Partícula

A estrutura escolhida para a representação da partícula foi a mesma utilizada no AG proposto que foi explicado na seção 4.1.1, sendo sua posição uma rota válida para todos os agentes dentro das restrições energéticas impostas.

4.3.2 Inicialização do Enxame

Para a inicialização do enxame o método utilizado foi o algoritmo do vizinho mais próximo randômico, o mesmo método utilizado no AG proposto e que pode ser visualizado no algoritmo 4.

4.3.3 Fitness

Para o cálculo do *fitness* é utilizado o proposto na equação 4.1 e implementado no AG proposto na seção 4.1.3.

Após realizar o cálculo do fitness de todas as partículas é selecionada a melhor partícula do enxame antes de iniciar as interações, essa partícula será o *pBest* e o *gBest*.

4.3.4 Velocidade da Partícula

Normalmente a velocidade de uma partícula pode ser calculada utilizando a fórmula 2.29, apresentada na seção 2.7, e sua posição é alterada de acordo com a equação 2.30. Contudo, na estrutura de representação da partícula torna-se inviável a utilização destes cálculos para a velocidade e mudança de posição.

O POE proposto altera a posição da partícula realizando a combinação delas com o *pBest* e o *gBest*. Para realizar a combinação delas é utilizado o método de cruzamento PMX mostrado na seção 2.6.4.4, no qual pode ser visualizado no algoritmo 8.

A aplicação do cruzamento PMX se diferencia do método convencional, pois uma partícula pode ser composta por várias rotas de robôs, então é necessário executar a combinação de uma uma rota de uma partícula com seu respectivo par da segunda partícula. Com as novas partículas retornadas pela combinação, é necessário realizar uma nova operação em cada partícula que será a remoção de elementos repetidos, essa operação é necessária para evitar que um *waypoint* que já exista em uma rota dentro da partícula seja reinserido em outra rota, o que acaba por infligir uma das restrições impostas na formulação matemática, a qual a ordem de remoção dos *waypoints* repetidos é feito de forma aleatória. Por fim é realizado uma comparação entre o *fitness* das partículas geradas no PMX com o antigo *fitness* e é retornado a partícula com a melhor performance.

Após a realização da combinação entre a partícula, o *pBest* e o *gBest* será realizado a operação de movimentação da partícula no espaço amostral. Normalmente, esta atividade é

Algoritmo 8: PMX POE

```
entrada: , Particula2;
saida: NovaParticula;
Início;
TmpParticula1 ← Nova;
TmpParticula2 ← Nova;
foreach ind ∈ Indexes(Particula1) do
    RotaParticula1 ← Particula1(ind);
    RotaParticula2 ← Particula2(ind);
    NovaRota1, NovaRota2 ← PMX(RotaParticula1, RotaParticula2);
    TmpParticula1(ind) ← NovaRota1;
    TmpParticula2(ind) ← NovaRota2;
end
RemoverElementosRepetidos(TmpParticula1);
RemoverElementosRepetidos(TmpParticula2);
NovaParticula ← MelhoParticula(TmpParticula1,TmpParticula2);
Finalização;
```

realizada com a mudança de velocidade da partícula, porém no OEP desenvolvido a mudança de posição da partícula será realizada utilizando três operações diferentes: a inserção, a recombinação e a deleção. Diferentemente do AG proposto nesta pesquisa, no qual é utilizado apenas uma dessas três operações em um indivíduo, no OEP serão executadas as três operações na ordem que foram apresentadas e serão executadas α vezes, isso se justifica devido a operação de combinação utilizando o PMX, pois novos segmentos foram inseridos nas rotas e é possível que algumas rotas tenham aumentado seu custo de locomoção ultrapassando sua restrição ou então tenham diminuído devido a um melhor arranjo na rota.

Desta forma, realizando a inserção de um novo elemento na rota pode-se aproveitar o custo que não esteja totalmente em uso ou remover elementos com o pior *fitness* para a rota. A operação de recombinação auxilia a organizar os elementos da rota que foram modificados pelo PMX. O elemento de recombinação da rota é uma operação de SWAP, como mostrado na seção 2.6.5.4, que será executada um total de β vezes para cada rota. A variável β é um parâmetro que especifica quantas vezes a operação de SWAP deverá ser executada . As operações de inserção e deleção são as mesmas utilizadas no AG proposto e foram exibidas no algoritmo 7. A operação de movimentação pode ser vista no algoritmo 9.

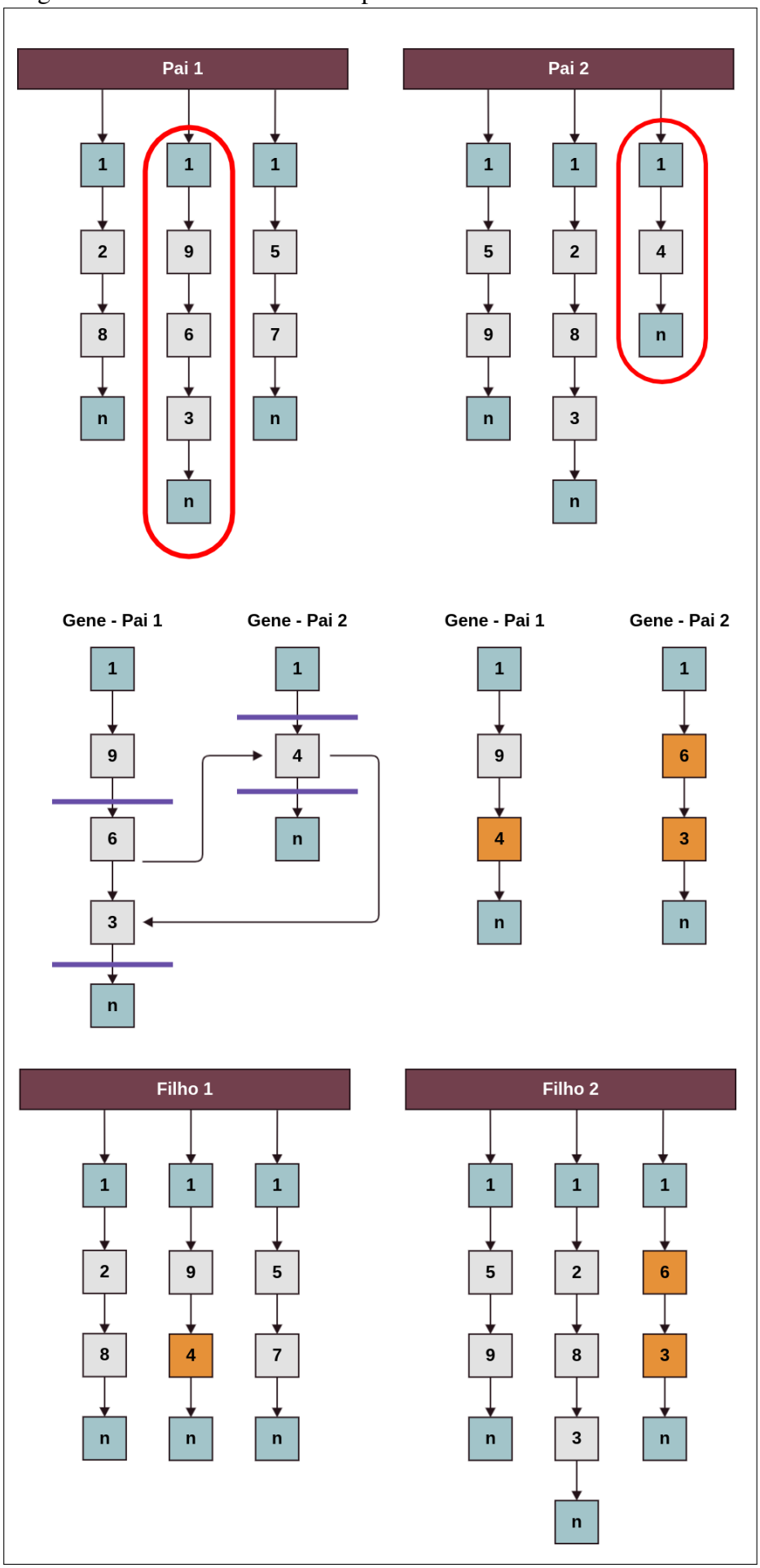
Logo após movimentar a partícula é realizada uma comparação entre o *fitness* da nova posição e a posição antiga, caso a posição mais recente possua um melhor *fitness*, a nova assumirá. O último passo é comparar se a partícula recente possui um *fitness* melhor que o

Algoritmo 9: Movimentação da Partícula

```
entrada: Particle;
saida: ParticleNewPosition;
Inicio;
ParticleNewPosition ← New;
foreach ind ∈ Indexes(Particle) do
    Route ← Particle(ind);
    Route ← Insert(Route);
    foreach i ∈ Range(5) do
        Route ← SWAP(Route);
    end
    Route ← Remove(Route);
    ParticleNewPosition(ind) ← Route;
end
Finishing;
```

gBest, caso possua, ela tomará a posição do *gBest* e tanto a operação quanto o processo iniciam novamente até completar o número de interações.

Figura 13 – Cruzamento AG Proposto



Fonte: Do Autor

5 AMBIENTE DE TESTES

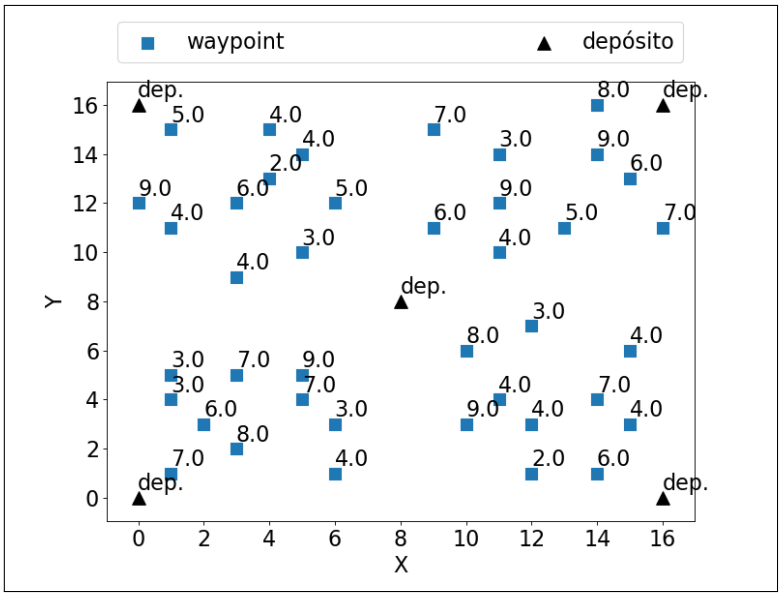
O presente capítulo expõe alguns detalhes do ambiente de testes utilizado para realizar a validação do modelo proposto. Também serão descritos os experimentos executados e as instâncias utilizadas para testes.

5.1 Instâncias

A seguinte seção apresenta as instâncias utilizadas para a validação das soluções propostas. No total foram utilizadas 20 instâncias auto-geradas, simétricas, onde as distâncias entre os *waypoints* são as mesmas, não importando o sentido, com diferente número de robôs e com diferentes tempos de baterias para cada robô. Cada *waypoint* possui uma pontuação, a pontuação de todos os *waypoints* em qualquer cenário varia entre 2 e 9. O custo de locomoção entre cada *waypoint* é calculado utilizando a distância euclidiana. Para o experimento é assumido que o tempo de bateria disponível para cada robô será dado em minutos e que a velocidade de locomoção dos robôs é constante. Diante disso, a distância entre os *waypoints* também é representada em minutos.

Para um melhor entendimento das instâncias utilizadas neste trabalhos são apresentadas na Tabela 2 e será realizado um detalhamento da instância 1, no qual o mapa completo poderá ser visualizado na figura 14.

Figura 14 – Instância 1



Fonte: elaborado pelo autor (2019)

Tabela 2 – Valores das Instâncias

Inst.	Robôs	Waypoints	Prem.	Tempo Bateria Robôs(min)
I_1	4	40	218	20, 23, 25, 30
I_2	4	45	222	23, 25, 27, 30
I_3	4	50	241	23, 25, 27, 30
I_4	4	60	313	20, 23, 25, 30
I_5	4	70	387	20, 23, 25, 30
I_6	4	80	436	20, 23, 25, 30
I_7	5	40	206	18, 22, 24, 25, 28
I_8	5	50	236	19, 23, 25, 26, 29
I_9	5	60	309	20, 22, 25, 27, 30
I_{10}	5	70	383	21, 23, 26, 27, 30
I_{11}	5	80	441	22, 24, 26, 28, 32
I_{12}	6	50	239	18, 20, 22, 24, 25, 27
I_{13}	6	60	310	19, 22, 23, 24, 25, 28
I_{14}	6	70	379	20, 22, 24, 25, 27, 30
I_{15}	6	80	440	21, 23, 24, 26, 29, 30
I_{16}	7	60	310	15, 20, 23, 25. 25, 27, 30
I_{17}	7	70	377	19, 20, 21, 23, 24 ,25, 29
I_{18}	7	80	443	19, 21, 23, 24, 25 ,26, 30
I_{19}	8	70	388	15, 20, 23, 25, 25, 27, 30, 30
I_{20}	8	80	428	15, 20, 23, 25, 25, 27, 30, 30

Fonte: elaborado pelo autor (2019).

A instância $I1$ é composta por 40 *waypoints* e 5 depósitos, no qual os robôs poderão ter sua origem e retorno definidos nos depósitos. Os depósitos não possuem pontuação e estarão disponíveis para o robô apenas se forem definidos como origem ou destino. A instância $I1$ possui uma premiação total de 218 pontos.

Os testes serão executados com base em 3 experimentos:

- **Experimento 1**: Execução do cálculo de rotas utilizando todos os robôs, com um único depósito para saída e chegada e que todos os robôs mantém seu nível de bateria;
- **Experimento 2**: Execução do cálculo de rotas utilizando um número reduzido de robôs, onde será reduzido o robô com o menor nível de bateria, com um único depósito para saída e chegada e que os robôs utilizados mantém seu nível de bateria;
- **Experimento 3**: Execução do cálculo de rotas utilizando todos os robôs, no qual foi modelado com a mudança de depósito destino por robô e que todos os robôs mantém seu nível de bateria;

Na execução dos testes é importante ressaltar o estado da bateria na execução dos experimentos, os métodos Ag-prop e OEP-Prop manterão o nível de bateria dos robôs, isso devido a capacidade fornecida pela modelagem do POEMRLMB, já o AG-Trad não possui a capacidade de preservar múltiplas restrições isso se deve pela modelagem utilizando o TOP inalterado. Para que os robôs que tiverem suas rotas calculadas pelo AG-Trad consigam sair e

voltar para a base será usado como restrição para todos os robôs o menor nível de bateria, pois caso utilize o maior nível de bateria robôs que não possuírem o mesmo nível não retornarão para a base.

O AG-Trad não será executado no experimento 3, este experimento impõe a mudança de base de destino para todos os robôs e como o AG-Trad foi modelado com base no TOP não será possível executar este experimento.

Os testes foram executados em um computador com um processador Intel i5-7600, 8GB de RAM e sistema operacional Ubuntu 16.04. O AG e o OEP propostos foram desenvolvidos na linguagem Python em sua versão 3.6 com auxílio da biblioteca matemática Numpy 1.14.5. e os gráficos gerados utilizando a biblioteca Matplotlib.

5.1.1 Parametrização do AG proposto

A parametrização de um AG é uma das etapas mais importantes para sua perfeita execução, na qual a utilização de parâmetros nos valores máximos acaba por gerar uma convergência prematura. Para a parametrização do AG proposto foi utilizada uma parametrização randômica aliada a uma busca exaustiva.

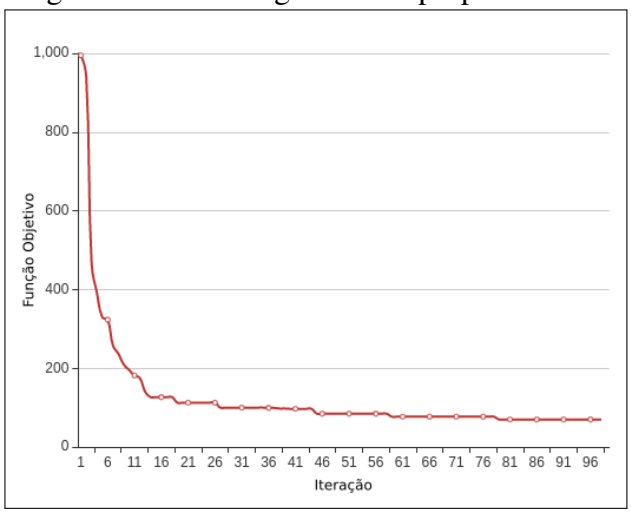
Primeiramente, foi realizada a geração de um conjunto de parâmetros aleatórios e em seguida, foram executados testes com a mudança de cada parâmetro de um valor mínimo até um valor máximo. A parametrização foi testada utilizando um mapa com 4 robôs e 40 pontos a serem visitados. A cada mudança de um parâmetro o AG foi executado 10 vezes. A Tabela 3 apresenta os dados propostos juntamente com o resultado alcançado. A parametrização ideal foi escolhida com bases nos resultados obtidos, onde foram escolhidos os parâmetros que obtiveram os valores com o menor desvio padrão e por fim, foi realizada uma execução entre o parâmetro aleatório e o parâmetro ideal. A curva de convergência com os dados especificados como ideal pode ser visto na Figura 15.

Tabela 3 – Parametrização do Algoritmo genético

Parâmetros	Aleatório	Range	Ideal
Tamanho População	300	[50, 100, ..., 500]	100
Prob. Cruzamento	.2	[.1, .2, ... , .9]	0.6
Prob. Mutação	.4	[.1, .2, ... , .9]	0.8
Ger. Máxima	35	[5, 10, ... , 50]	25

Fonte: elaborado pelo autor (2019).

Figura 15 – Convergência AG proposto



Fonte: elaborado pelo autor (2019)

5.1.2 Parametrização do OEP proposto

A parametrização de um OEP é tão importante quanto no AG, pois exceder os valores dos parâmetros pode acarretar uma convergência prematura. Assim como na parametrização do AG proposto o OEP deste trabalho utilizou uma parametrização randômica aliada a uma busca exaustiva.

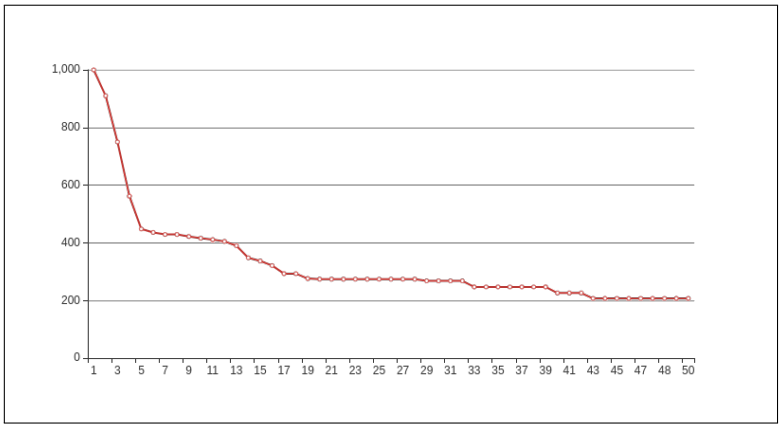
Primeiramente, foi realizada a geração de um conjunto de parâmetros aleatórios e em seguida foram executados testes com a mudança de cada parâmetro de um valor mínimo até um valor máximo. A parametrização foi testada utilizando um mapa com 4 robôs e 40 pontos a serem visitados. A cada mudança de um parâmetro do OEP foram executados 10 vezes. A Tabela 4 apresenta os dados propostos juntamente com o resultado alcançado. A parametrização ideal foi escolhida com bases nos resultados obtidos, onde foram escolhidos os parâmetros que obtiveram os valores com o menor desvio padrão e por fim, foi realizada uma comparação entre o parâmetro aleatório e o parâmetro ideal. A curva de convergência com os dados especificados como ideal pode ser visto na Figura 16.

Tabela 4 – Parametrização do OEP

Parâmetros	Aleatório	Range	Ideal
Tamanho do Enxame	300	[25, 50, ..., 500]	100
Número Interações	400	[50, 100, ..., 1000]	250
α	8	[.1, .2, ... , .9]	3
β	2	[.1, .2, ... , .9]	5

Fonte: elaborado pelo autor (2019).

Figura 16 – Convergência OEP proposto



Fonte: elaborado pelo autor (2019)

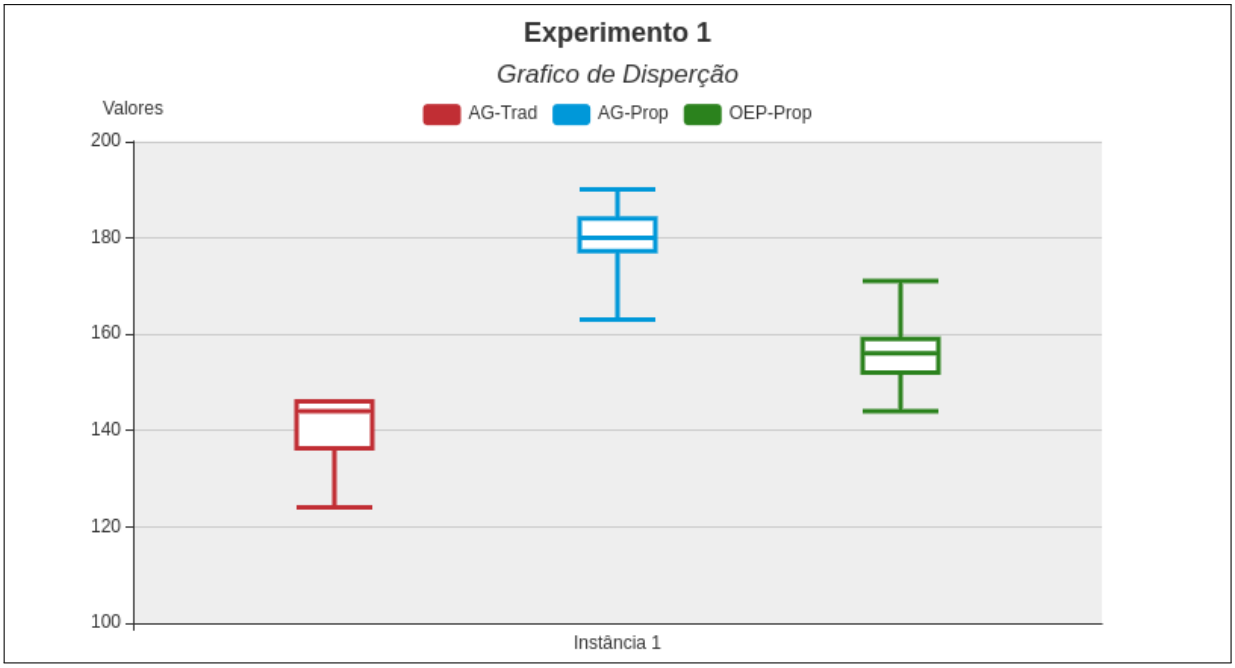
6 RESULTADOS E DISCUSSÕES

Este capítulo apresentará os resultados obtidos das execuções de três métodos para o cálculo de rotas de múltiplos robôs: a proposta de (Bederina; Hifi, 2017) que implementa um AG para a resolução do TOP tradicional(AG-Trad), um AG implementado proposto que resolve o POEMRLMB(AG-Prop), e uma otimização por enxame de partículas proposto que resolve o POEMRLMB(OEP-Prop). Cada método foi executado 30 vezes para os 3 experimentos descritos na seção 5.1. Os resultados são a média de coleta de prêmios realizadas por cada método e suas execuções. A dispersão de dados arrecadados pelos métodos são exibidas através de um gráfico de caixa. Os resultados serão descritos para cada experimento executado.

6.1 Resultados do Experimento 1

O experimento 1 consiste na execução do cálculo de rotas para cada instância descrita na seção 5.1. Os resultados podem ser visualizados na Tabela 5. Os resultados apresentam o recolhimento de prêmios de cada método. Para cada método é apresentado o menor prêmio, o maior prêmio e a média de todos os prêmios recolhidos. A Figura 17 mostra a variação de prêmios recolhidos da execução da instância $I1$ através de um diagrama de caixas(*Box Plot*). As rotas geradas para a instância $I1$ podem ser visualizadas na Figura 19.

Figura 17 – Gráfico de dispersão dos dados experimento 1 para a instância I1



Fonte: elaborado pelo autor (2019).

Tabela 5 – Execução do Experimento 1

Instâncias	AG-Trad			Ag-Prop			OEP-Prop		
	Min.	Max	Med. |	Min.	Max	Med. |	Min.	Max	Med.
I_1	124	146	140,86	163	190	180,23	144	171	156,23
I_2	155	185	170,16	196	216	205,36	157	179	166,33
I_3	165	188	176,3	198	214	207,5	159	185	171,13
I_4	209	237	223,4	243	275	254,8	186	208	197,36
I_5	245	273	261,1	280	312	299,3	211	241	223,26
I_6	254	305	276,1	295	349	324,7	221	246	231,43
I_7	112	125	120,5	180	202	193,6	163	185	173,2
I_8	129	157	145,7	193	214	202,4	176	193	183,4
I_9	177	220	202,9	254	279	265,3	223	244	235
I_{10}	232	255	242,4	293	320	306	264	290	275,2
I_{11}	253	284	266,4	323	355	340,8	297	330	311,06
I_{12}	139	168	155,7	197	225	210,8	179	201	189,5
I_{13}	178	218	204,6	262	284	270,9	229	257	244,93
I_{14}	236	274	257,1	325	350	337,4	282	309	296,83
I_{15}	272	301	288,5	256	382	365,8	325	357	338
I_{16}	200	218	207,2	263	288	274,7	231	261	248,13
I_{17}	252	284	270,8	313	354	336,4	285	311	299,8
I_{18}	248	288	272,2	378	402	388,4	338	368	348,46
I_{19}	178	196	188,2	311	343	329,2	278	314	291,26
I_{20}	284	202	193,9	355	387	370,5	312	340	324,46

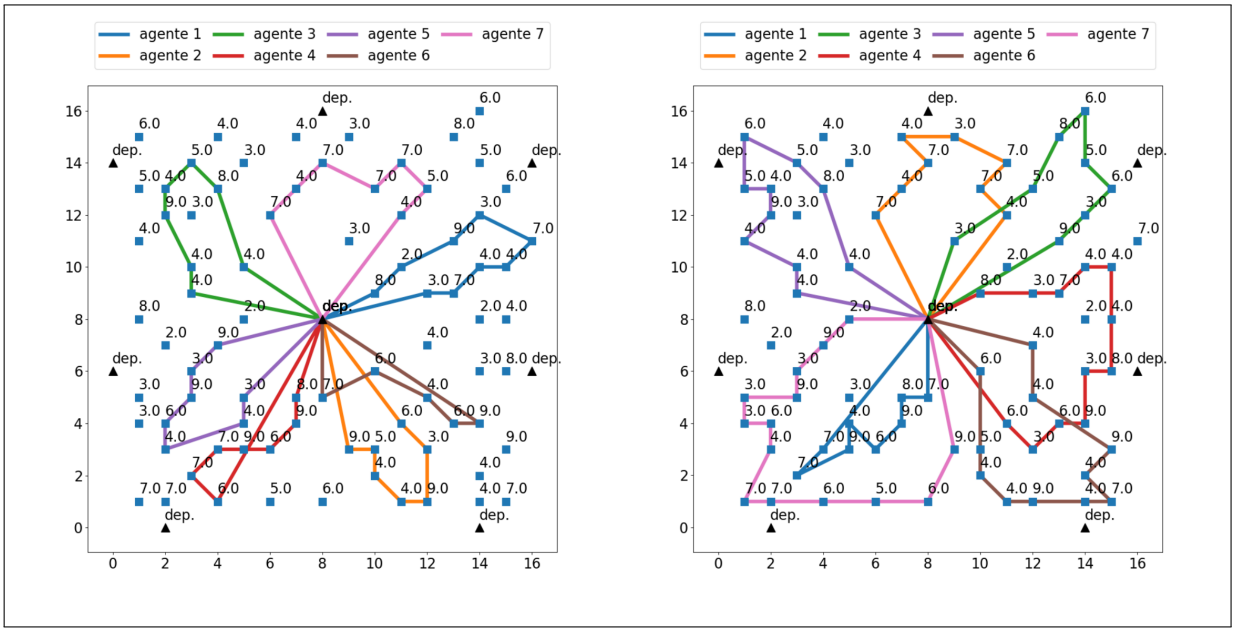
Fonte: Elaborado pelo autor (2019).

Os resultados exibidos pela Tabela 5 é possível comparar as execuções dos três métodos para o cálculo das rotas, é possível notar que o AG-Prop obteve a melhor pontuação em todos as instâncias.

Como explicado na seção 5.1 o AG-Trad foi modelado com base no TOP tradicional onde todos os agentes possuem o mesmo custo de locomoção, sendo assim para o robô com a menor capacidade energética retorne para sua base o AG-Trad deverá colocar a restrição energética dos agente igual ao do robô com o menor nível de bateria.

Ao realizar uma comparação entre os resultados obtidos entre o método AG-Prop e o AG-Trad é observado que o AG-Prop obteve uma melhor pontuação, a melhora é significativa pelo uso total da capacidade de locomoção de cada robô, é possível notar que a pontuação coletada no modelo proposto em certos cenários foi capaz de ultrapassar em 29,9% ao modelo tradicional, observado na instância $I18$, as rotas geradas para esta instância podem ser vistas na Figura 18. A instância $I20$ obteve uma diferença de 91,7% isso é justificado pelo baixo nível de bateria apresentado pelo robô 1 desta instância, em cenários reais robôs com baixo nível de energia podem ser removidos do cálculo de rota para o cumprimento de uma missão, na próxima seção será realizado os testes com a remoção do robô com menor custo e esta particularidade será abordada novamente.

Figura 18 – Rotas geradas para o experimento 1 na instância *I*18 utilizando AG-Trad e AG-Prop



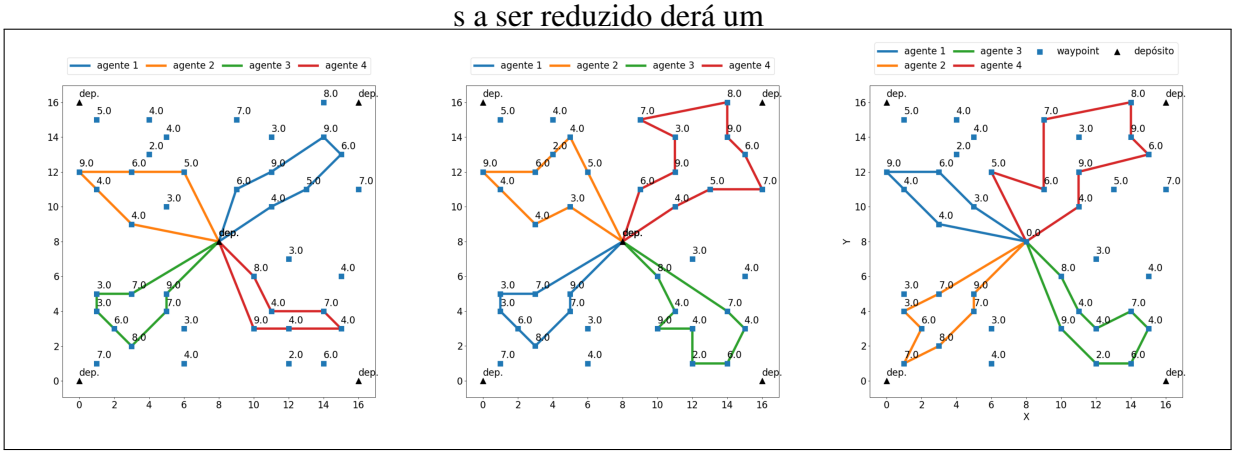
Fonte: elaborado pelo autor (2019).

Uma comparação também é realizada entre o AG-Prop e o OEP-Prop. Apesar de ambos os métodos implementarem o POEMRLMB o AG-Prop ainda obteve uma melhor pontuação comparado ao OEP-Prop, é possível notar uma melhora no recolhimento de prêmios em até 28,72% como pode ser visualizado na instância *I*6.

Na Figura 17 é possível observar o AG-Prop obteve o melhor resultado. A dispersão de dados mostrou uma coleta de dados superior as demais métodos, o gráfico mostra que o OEP-Prop pode ultrapassar os resultados do AG-Prop, porém somente considerando o pior resultado do AG-Prop em comparação do melhor resultado do OEP-Prop, uma probabilidade de 0,0011%.

Figura 19 – Rotas geradas para o experimento 1 utilizando para a instância *I*1 AG-Trad, AG-Prop e OEP-Prop

s a ser reduzido derá um



Fonte: elaborado pelo autor (2019).

6.2 Resultados do Experimento 2

O experimento 2 consiste na execução do cálculo de rotas para cada instância descrita na seção 5.1, porém o número de robôs será reduzido. Este experimento removerá apenas um único robô e será o que possuir o menor custo de locomoção. O motivo da remoção de um robô do cálculo é para que o AG-Trad possa aumentar a sua restrição de locomoção de seus robôs. Os resultados podem ser visualizados na Tabela 6. Os resultados apresentam o recolhimento de prêmios de cada método. Para cada método é apresentado o menor prêmio, o maior prêmio e a média de todos os prêmios recolhidos. A Figura 20 mostra a variação de prêmios recolhidos da execução da instância $I1$ através de um *box plot*. As rotas geradas para a instância $I1$ podem ser visualizadas na Figura 24.

Tabela 6 – Execução do Experimento 2

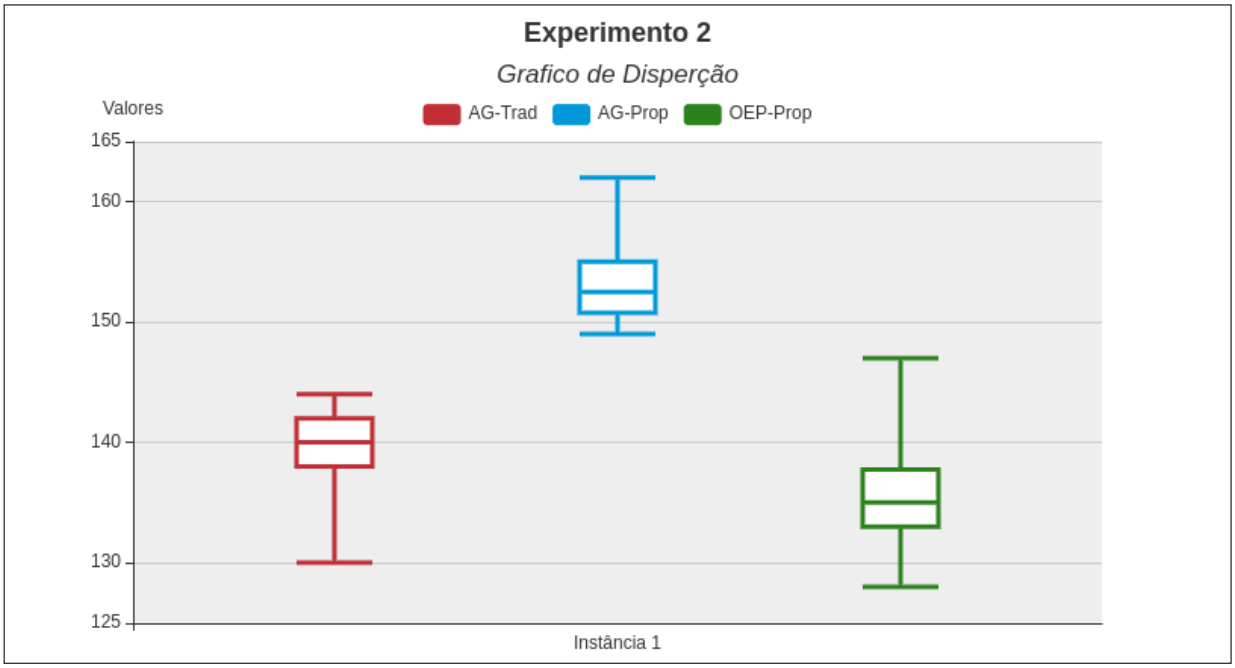
Instâncias	AG-Trad			Ag-Prop			OEP-Prop		
	Min.	Max	Med. |	Min.	Max	Med. |	Min.	Max	Med.
I_1	130	144	139,36	149	162	153,13	128	147	135,66
I_2	150	162	157,2	166	184	175,26	138	163	148,4
I_3	152	167	164,5	171	182	177,23	141	165	150,5
I_4	178	206	197,01	223	275	210,1	164	188	175,36
I_5	213	245	228,6	245	280	262,53	186	217	197,93
I_6	238	274	253,6	247	288	273,4	192	222	205,06
I_7	138	168	157,5	178	193	186,13	155	173	164,76
I_8	147	172	162,8	184	204	191,9	163	180	171,4
I_9	183	205	197,06	132	260	246,4	201	235	216,7
I_{10}	220	243	230,2	272	297	284,6	240	262	249,1
I_{11}	238	272	253,2	289	332	312,4	267	287	276,4
I_{12}	156	182	170,9	186	211	197,1	155	184	172,3
I_{13}	208	237	225,2	236	273	255,6	223	240	231,33
I_{14}	238	269	255,8	296	328	313,5	259	301	276,6
I_{15}	264	308	282,4	324	364	344,26	299	324	309,23
I_{16}	187	219	205,8	255	286	267,2	227	248	336,6
I_{17}	233	291	258,4	306	330	318,4	261	293	279,23
I_{18}	255	293	276,2	341	380	359,4	315	348	327,63
I_{19}	218	248	227,2	306	328	317,2	267	295	281,33
I_{20}	264	292	275,09	345	273	358,1	298	328	313,06

Fonte: Elaborado pelo autor (2019).

Os resultados exibidos pela Tabela 6 é possível comparar as execuções dos três métodos para o cálculo das rotas com redução de robôs, é possível notar que o AG-Prop obteve a melhor pontuação em todos as instâncias, mesmo com a redução de agentes.

Com a redução de agentes o AG-Trad obteve uma melhora no recolhimento de prêmios, no qual observando os resultados da instância $I20$ do experimento 1 e comparando ao experimento 2 é possível notar que a média de coleta de prêmios para a instância conseguiu

Figura 20 – Gráfico de dispersão dos dados do experimento 2 para a Instância I1

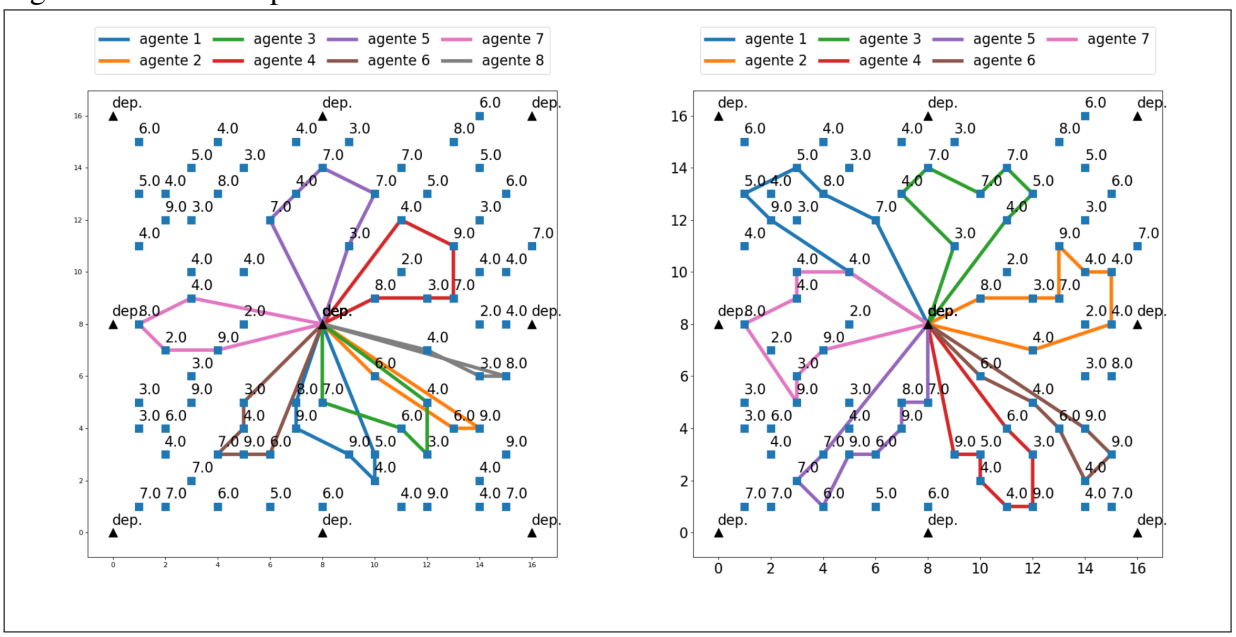


Fonte: elaborado pelo autor (2019).

arrecadar 42,24% a mais de premiações. As rotas geradas para os dois experimentos pode ser visto na Figura 21. A melhora do experimento 2 em comparação ao experimento 1 do método AG-Trad é justificável pela capacidade do TOP, utilizado no método AG-Trad, que aceita apenas um custo de locomoção para todos os robôs. Para que todos os robôs sejam utilizados no cálculo da rota do TOP é necessário utilizar como restrição de bateria o menor valor de bateria entre os robôs. Utilizando o menor valor de bateria, que é igual a 15 minutos, o TOP acabou por ignorar o nível de bateria dos demais robôs, como na instância $I20$ os robôs com os maiores níveis de bateria, que são iguais a 30 minutos, puderam utilizar apenas metade da sua capacidade de locomoção. Com a remoção do robô com o menor nível de bateria o TOP utiliza o segundo menor valor, que na instância $I20$ será igual a 20 minutos, considerando este valor será o nível de bateria para todos os robôs a caminhada de cada robô aumentará em 5 minutos, como se tem 7 robôs restantes a caminhada total aumenta em 35 minutos,o que resulta em um valor maior de bateria do primeiro robô. A redução de robôs também permitiu que os resultados do experimento 2 chegassem a ser quase iguais ao do experimento 1 para o AG-Trad. Contudo, mesmo com a melhora no resultado do AG-Trad o Ag-Prop ainda obteve o melhor recolhimento de premiações, onde em determinadas instâncias o recolhimento de premiações pode chegar a 39,2%, observado na instância $I19$, as rotas geradas do AG-Trad e o AG-Pro para a instância $I19$ podem ser visualizadas na Figura 22.

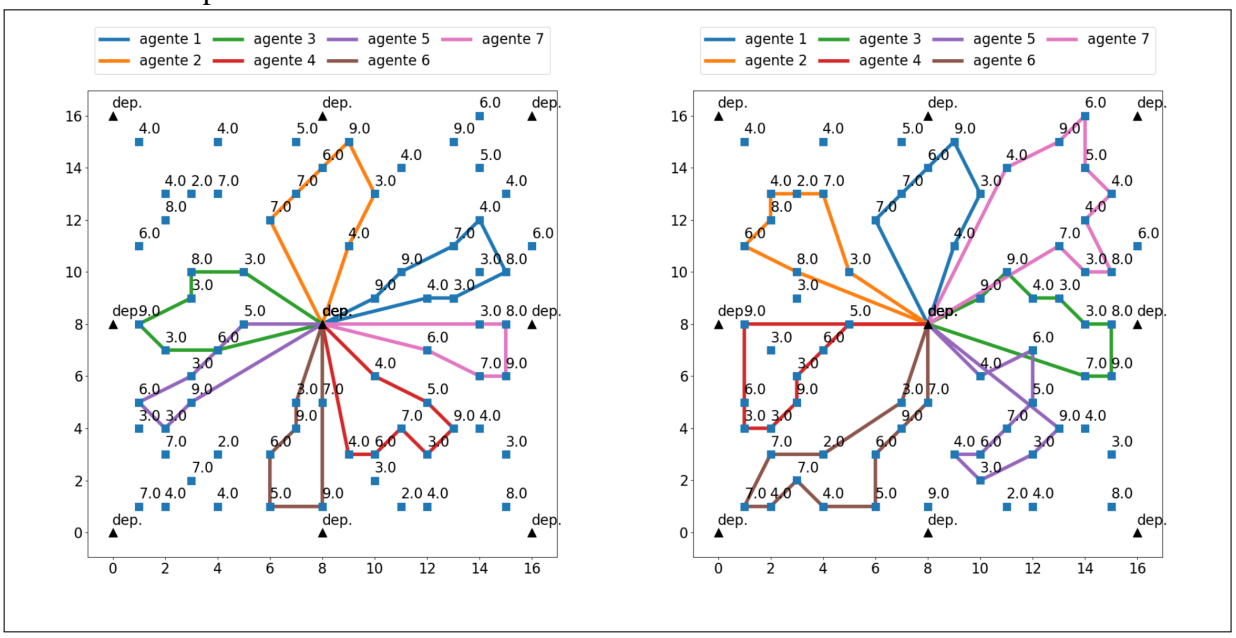
O método utilizando o OEP-Prop obteve uma melhora na coleta de prêmios em

Figura 21 – Rota Experimento 1 e 2 instância *I*20



Fonte: elaborado pelo autor (2019).

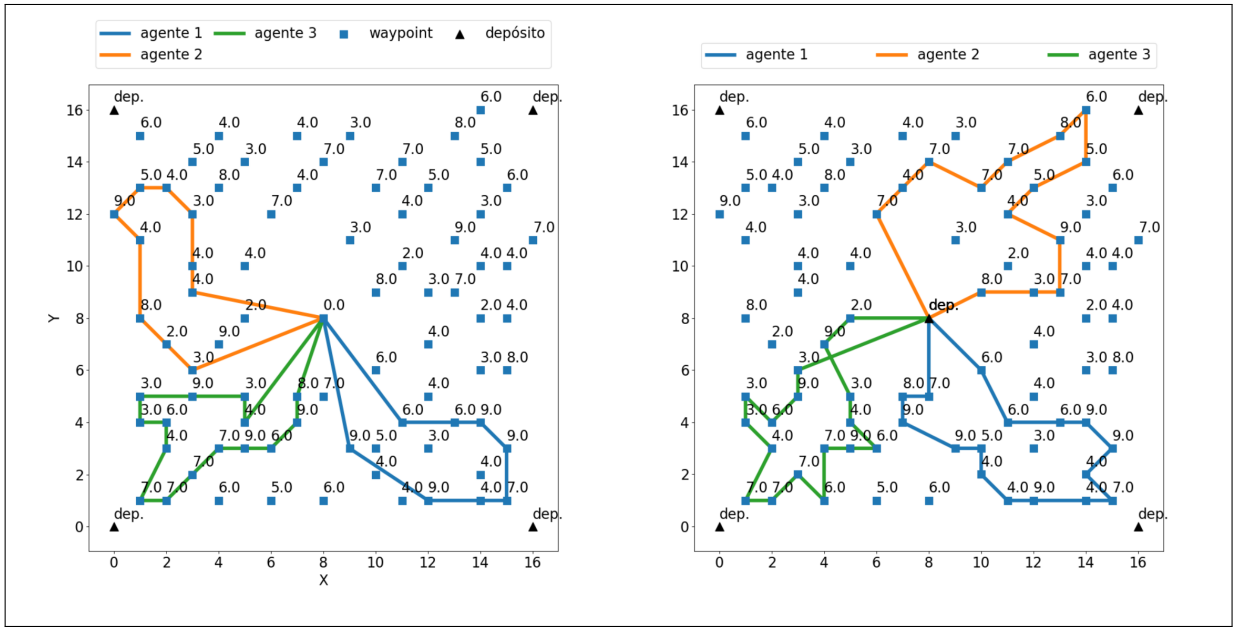
Figura 22 – Rotas geradas para o experimento 2 utilizando para a instância *I*19 AG-Trad, AG-Prop



Fonte: elaborado pelo autor (2019).

determinadas instâncias, *I*13 e *I*16, no qual em alguns casos a coleta de pontos é próxima a sua primeira execução, o experimento 1 comparado ao experimento 2, isso se deve ao aumento de locomoção dos robôs restantes, porém o AG-Prop ainda obteve um melhor desempenho comparados ao OEP-Prop, onde em certos cenários conseguiu ultrapassar os resultados obtidos pelo OEP-Prop em até 40,3% na coleta de prêmios, como pode ser observado na instância *I*6, as rotas geradas do OEP-Pro e o AG-Pro para a instância *I*6 podem ser visualizadas na Figura 23.

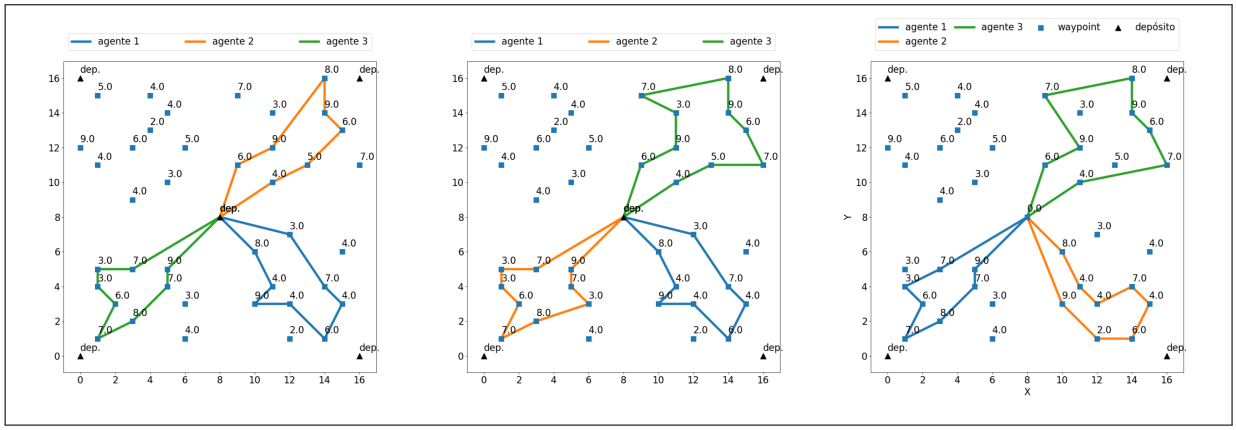
Figura 23 – Rotas geradas para o experimento 2 utilizando para a instância *I*6 OEP-Prop e AG-Prop



Fonte: elaborado pelo autor (2019).

Na Figura 20 é possível observar que mesmo nos melhores casos para o AG-Trad e o OEP-Prop não se equiparam ao AG-Prop no recolhimento de prêmios para a instância *I*1.

Figura 24 – Rotas geradas para o experimento 2 na exucução da instância e utilizando AG-Trad, AG-Prop e OEP-Prop



Fonte: elaborado pelo autor (2019).

6.3 Resultados do Experimento 3

O experimento 3 consiste na execução do cálculo de rotas para cada instância descrita na seção 5.1, porém a base de destino de todos os robôs será diferente. Como descrito na seção 5.1, neste experimento não será utilizado o método AG-Trad, como esse método é baseado no TOP tradicional onde não é possível definir depósitos de saída e de destino diferentes capa cada robô, os robôs tem seus depósitos de saída e sua chegada o mesmo para todos. Desta forma, no experimento 3 serão executados apenas os métodos AG-Prop e o OEP-Prop. Para cada método é apresentado o menor prêmio, o maior prêmio e a média de todos os prêmios recolhidos. A Figura 25 mostra a variação de prêmios recolhidos da execução da instância $I1$ através de um diagrama de caixas. As rotas geradas para a instância $I1$ podem ser visualizadas na Figura 24.

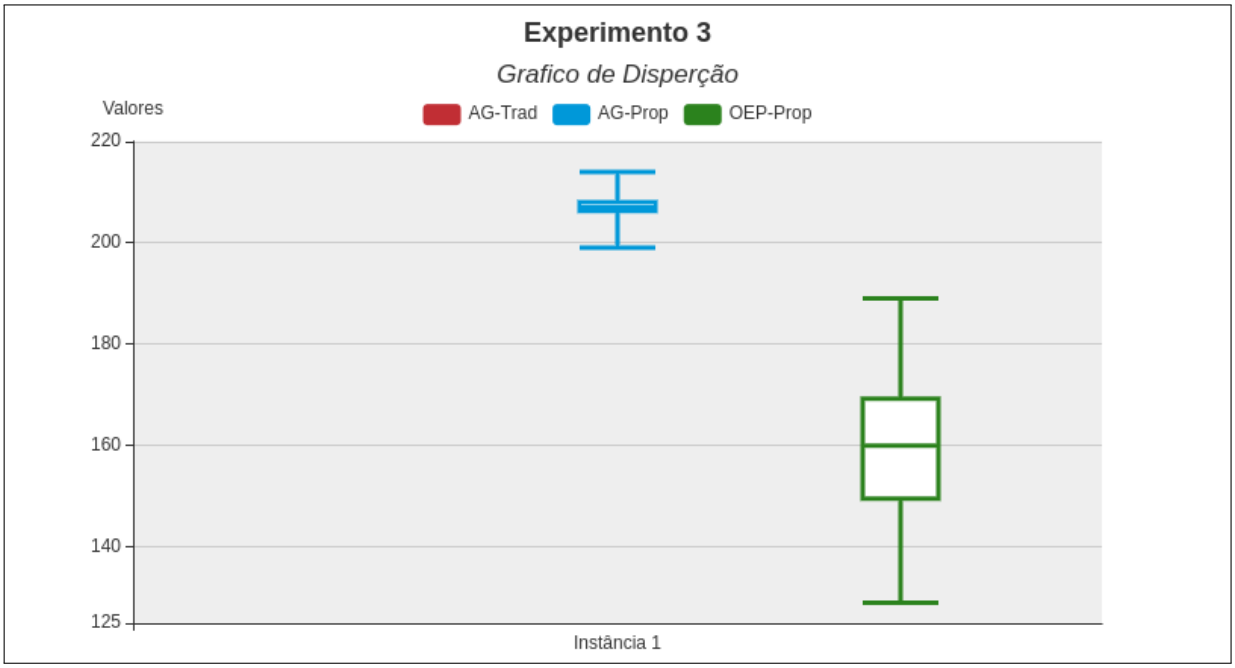
Tabela 7 – Execução do Experimento 3

Instâncias	AG-Trad			Ag-Prop			OEP-Prop		
	Min.	Max	Med.	Min.	Max	Med.	Min.	Max	Med.
I_1	–	–	–	199	214	207,5	128	189	160,16
I_2	–	–	–	190	215	208,2	127	183	161,13
I_3	–	–	–	204	226	223,4	136	186	161,03
I_4	–	–	–	240	289	265,36	169	216	194,2
I_5	–	–	–	300	350	329,5	180	251	219,06
I_6	–	–	–	309	379	347,7	213	267	233,03
I_7	–	–	–	186	203	199,3	150	194	178,13
I_8	–	–	–	213	233	224,4	174	212	191,6
I_9	–	–	–	263	304	283,7	205	251	236
I_{10}	–	–	–	299	355	334,7	248	312	275,16
I_{11}	–	–	–	367	392	379,2	289	344	315,96
I_{12}	–	–	–	213	237	228,1	166	197	181,8
I_{13}	–	–	–	270	308	294,2	217	265	239,33
I_{14}	–	–	–	342	380	360	268	319	288,8
I_{15}	–	–	–	373	426	396,6	295	371	330,73
I_{16}	–	–	–	288	303	294,6	208	269	244,6
I_{17}	–	–	–	341	374	361,1	265	322	287,53
I_{18}	–	–	–	382	425	401,1	288	364	331,96
I_{19}	–	–	–	348	376	363,7	246	309	276,16
I_{20}	–	–	–	379	423	402,1	268	347	305,6

Fonte: Elaborado pelo autor (2019).

O OEP-Prop obteve uma queda de performance no experimento 3 comparada a sua execução no experimento 1, este método não demonstrou uma boa performance com a mudança de depósitos de destino para cada robô, isso se deve a forma de cruzamento escolhida, o OEP-Prop só insere novos elementos na mutação diferente do AG-Prop que durante o próprio cruzamento o tamanho das rotas podem ser alterados. Realizando a comparação entre os experimentos 3 do OEP-Prop e o AG-Prop é possível notar que o AG-Prop conseguiu alcançar ótimos resultados

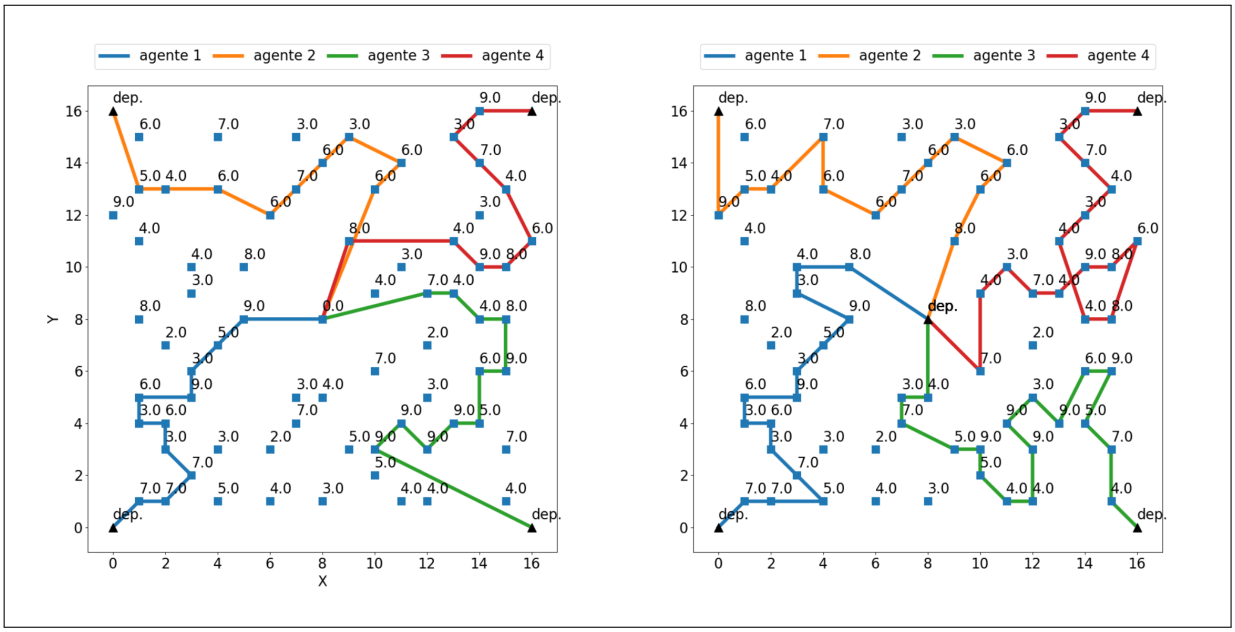
Figura 25 – Gráfico de dispersão dos dados do experimento 2 para a instância I1



Fonte: elaborado pelo autor (2019).

superando a coleta de prêmio em até 50,4%, como pode ser visto na instância *I*5, a rota gerada na instância *I*5 pode ser visualizada na Figura 27. Além da melhora de performance entre as soluções o resultado obtido no experimento 3 do AG-Prop superou os resultados obtidos por ele mesmo na execução do experimento 1, no qual uma melhora de até 13% na coleta de prêmios, isso demonstra que o método proposto possui uma boa generalização na solução de problemas com diferente depósitos.

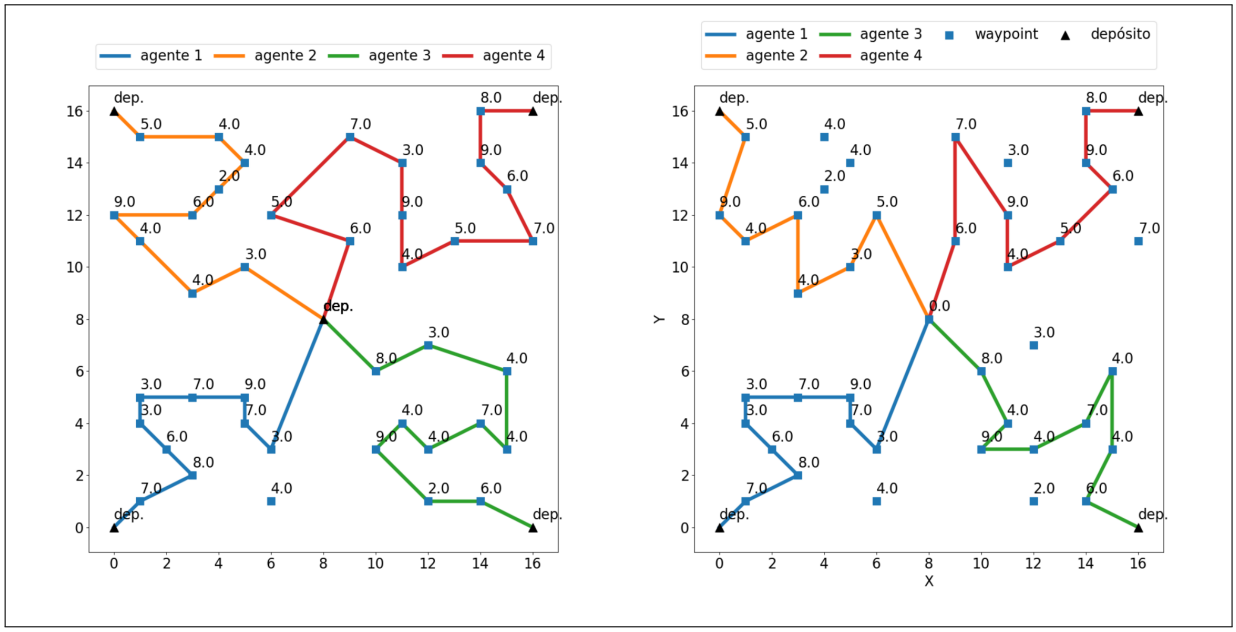
Figura 26 – Rotas geradas para o experimento 3 na instância *I*5 utilizando OEP-Prop e AG-Trad



Fonte: elaborado pelo autor (2019).

A Figura 25 mostra a variação de prêmios recolhidos, é possível observar que o OEP-Prop mesmo no melhor cenário de recolhimento de prêmios não é possível superar o AG-Prop, demonstrando a eficácia do AG-Prop com a mudança de depósitos. O gráfico também mostra que neste experimento os dados do AG-Prop tendem a sofrer menos dispersão de dados, onde os valores tende a ficarem próximos da mediana.

Figura 27 – Rotas geradas para o experimento 3 na instância *I*1 utilizando OEP-Prop e AG-Trad



Fonte: elaborado pelo autor (2019).

7 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi discutido o cálculo de rotas para múltiplos robôs colaborativos com restrições energéticas. Este problema foi modelado através do *Team Orieentering Problem* combinado a Problema das Múltiplas Mochilas. Tendo em vista este modelo foi desenvolvido duas meta-heurísticas para a sua resolução um algoritmo genético e um algoritmo de otimização por enxame de partículas. Os métodos ainda foram comparados ao algoritmo genético desenvolvido por (Bederina; Hifi, 2017) para a resolução do TOP tradicional. Para a validação dos métodos foram propostas 20 instâncias com diferentes variações de robôs, números de *waypoints* e restrições para cada robô. Para cada instância foram executados 3 experimentos com sua execução com o set completo de robôs, a redução de robôs e a mudança de depósitos.

Os modelos propostos apresentam bons resultados no recolhimento de prêmios em comparação ao modelo de (Bederina; Hifi, 2017) em todas as instâncias e experimentos testados, com exceção das instâncias com 4 robôs para o método OEP-Prop, no qual o modelo de (Bederina; Hifi, 2017) apresentou um melhor recolhimento de prêmios. O método OEP-Prop apresentou uma queda de performance no experimento 3 com a mudança de depósitos, demonstrando que este método é menos generalista em comparação ao método AG-Prop.

Em relação aos resultados é notório afirmar que o AG-Prop não apenas satisfaz os objetivos propostos, mas também apresenta uma performance superior em todos os experimentos em comparação aos os dois outros métodos executados, e sua capacidade de generalização o torna ideal para aplicações reais. A capacidade de inserção de *waypoints* realizada pelo cruzamento de *slice* foi um dos grandes responsáveis pela melhor performance do método, isso se deve a a capacidade de inserir novos *waypoints* e alterar o tamanho da rota original apenas realizando a combinação genética.

Embora os resultados sejam satisfatórios a natureza simétrica dos testes faz seu uso limitado a ambientes sem volatilidade de custos entre os *waypoints* a serem visitados, no qual o seu uso para cálculo de rotas para VANT não é o ideal, já que apresentem um ambiente mais dinâmico devido a força, velocidade e direção dos ventos, porém seu uso pode ser aplicado em ambientes industriais no cálculo de rotas para AGV, estendendo seu uso a indústria 4.0.

7.1 Trabalhos Futuros

Como trabalho futuro é necessário realizar o desempenho do método em ambientes assimétricos para uma melhor extensão do seu uso em ambientes voláteis.

A natureza múltiplo depósito da proposta o faz ideal para ser testado juntamente com sistemas de diagnósticos de falhas em baterias, já que o custo de locomoção pode ser reduzido rapidamente, necessitando um recálculo da rota para um melhor aproveitamento da missão.

Outro trabalho futuro visa em atribuir custos as visitas dos *waypoints* para simular o custo da execução de tarefas nos *waypoints*. Finalmente seria importante estender seu uso futuro a mapas com janelas de tempo para a visita dos *waypoints*.

Como comentado na seção anterior, a proposta deste trabalho pode ser aplicada na indústria, mais precisamente no cálculo de rotas para AGV's. Como trabalho futuro é proposto que o cálculo das rotas de recolhimentos de produtos montados em uma linha de produção de uma fábrica seja recolhida por um grupo de AGV's que deverão levar as peças da linha para os depósitos de armazenamento. Para que os AGV's possam receber as rotas geradas será feita a adaptação dos equipamentos com computadores Linux embarcado para o controle de navegação, desta forma o código desenvolvido em Python poderá ser adaptado exigindo o mínimo de alterações.

REFERÊNCIAS

ABE, J. M.; FILHO, J. I. da S.; TORRES, C. R. Robô móvel autônomo emmy: uma aplicação eficiente da lógica paraconsistente anotada. **Seleçao Documental: Inteligência Artificial e novas Tecnologias**, Universidade Santa Cecília, n. 3, p. 19–26, 2006.

ACKERMAN, E. **Robots With Smooth Moves Are Up to 40% More Efficient**. 2015. Disponível em: <https://spectrum.ieee.org/automaton/robotics/industrial-robots/robots-with-smooth-moves-are-more-efficient>. Acesso em: 28 set. 2019.

ALMEIDA, J. P. L. S. de; ARRUDA, L. V. R. de; NEVES-JR, F. Planejamento de rota por meio de algoritmo genético para um enxame de rob^os. 2017.

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: <http://www.sciencedirect.com/science/article/pii/S1568494619305095>.

APPLEGATE, D. L.; BIXBY, R. E.; CHVATAL, V.; COOK, W. J. **The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics)**. Princeton, NJ, USA: Princeton University Press, 2007. ISBN 0691129932, 9780691129938.

ARAI, T. Advanced robotics and mechatronics and their applications in construction automation. In: . [S.l.: s.n.], 2011.

Araki, B.; Strang, J.; Pohorecky, S.; Qiu, C.; Naegeli, T.; Rus, D. Multi-robot path planning for a swarm of robots that can both fly and drive. In: **2017 IEEE International Conference on Robotics and Automation (ICRA)**. [S.l.: s.n.], 2017. p. 5575–5582.

ARANTES, J. d. S. *et al.* Planejamento de rota para vants em caso de situação crítica: Uma abordagem baseada em segurança. 2016.

Bala, A.; Sharma, A. K. A comparative study of modified crossover operators. In: **2015 Third International Conference on Image Information Processing (ICIIP)**. [S.l.: s.n.], 2015. p. 281–284.

BECCENERI, J. C. Meta-heurísticas e otimização combinatória: Aplicações em problemas ambientais. 2008.

Bederina, H.; Hifi, M. A hybrid multi-objective evolutionary algorithm for the team orienteering problem. In: **2017 4th International Conference on Control, Decision and Information Technologies (CoDIT)**. [S.l.: s.n.], 2017. p. 0898–0903.

BERG, J. van den; SNOEYINK, J.; LIN, M.; MANOCHA, D. Centralized path planning for multiple robots: Optimal decoupling into sequential plans. In: . [S.l.: s.n.], 2009.

BRAGA, M. d. L.; SANTOS, A. d. J.; PEDROZA, A. C. P.; COSTA, L. H. M. K. Planejamento de rotas com algoritmos anytime em redes veiculares na plataforma raspberry pi. 2014.

BUTT, S. E.; CAVALIER, T. M. A heuristic for the multiple tour maximum collection problem. **Computers Operations Research**, v. 21, n. 1, p. 101 – 111, 1994. ISSN 0305-0548. Disponível em: <http://www.sciencedirect.com/science/article/pii/0305054894900655>.

CANDIDO, A. d. S. Sistema de gerenciamento do voo de quadrirotores tolerante a falhas. 01 2015.

CASTRO, I.-P. L. de; ZUBEN, V. **Representação e Operadores Evolutivos**. 2015. Disponível em: <ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/ia707_02/topico14_02.pdf>. Acesso em: 05 nov. 2019.

CHAO, I.-M.; GOLDEN, B. L.; WASIL, E. A. The team orienteering problem. **European Journal of Operational Research**, v. 88, n. 3, p. 464 – 474, 1996. ISSN 0377-2217. Disponível em: <http://www.sciencedirect.com/science/article/pii/0377221794002894>.

CHAVES, A. A.; EXATA, M. Heurística para resolução do problema do caixeiro viajante com coleta de prêmios. **Universidade Federal de Ouro Preto Instituto de Ciências Exatas e Biológicas Departamento de Computação. Ouro Preto, MG**, 2003.

CHUDASAMA, C.; SHAH, S.; PANCHAL, M. Comparison of parents selection methods of genetic algorithm for tsp. **Proc. of the Int. Conf. on Computer Communication and Networks CSI-COMNET-2011**, v. 1, p. 102–105, 01 2011.

CHVÁTAL, V.; COOK, W. J.; DANTZIG, G. B.; FULKERSON, D. R.; JOHNSON, S. M. Solution of a large-scale traveling-salesman problem. In: **50 Years of Integer Programming**. [S.l.: s.n.], 1954.

COPPIN, B. **Artificial Intelligence Illuminated**. 1st ed. ed. [S.l.]: Jones and Bartlett Publishers, 2004. ISBN 0763732303,9780763732301.

DANG, D.-C.; EL-HAJJ, R.; MOUKRIM, A. A branch-and-cut algorithm for solving the team orienteering problem. In: GOMES, C.; SELLMANN, M. (Ed.). **Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 332–339. ISBN 978-3-642-38171-3.

DAVIS, L. Applying adaptive algorithms to epistatic domains. In: **IJCAI**. [S.l.: s.n.], 1985. v. 85, p. 162–164.

DEINEKO, V.; TISKIN, A. One-sided monge tsp is np-hard. In: GAVRILOVA, M.; GERVASI, O.; KUMAR, V.; TAN, C. J. K.; TANIAR, D.; LAGANÁ, A.; MUN, Y.; CHOO, H. (Ed.). **Computational Science and Its Applications - ICCSA 2006**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 793–801. ISBN 978-3-540-34076-8.

DEWANGAN, R. K.; SHUKLA, A.; GODFREY, W. W. Path planning for multiple mobile robots by priority assignment. In: VERMA, N. K.; GHOSH, A. K. (Ed.). **Computational Intelligence: Theories, Applications and Future Directions - Volume II**. Singapore: Springer Singapore, 2019. p. 161–172. ISBN 978-981-13-1135-2.

DORIGO, M. Optimization, learning and natural algorithms. ph.d. thesis. In: ____. [S.l.: s.n.], 1992.

ESHTEHADI, R.; DEMIR, E.; HUANG, Y. Solving the vehicle routing problem with multi-compartment vehicles for city logistics. **Computers Operations Research**, v. 115, p. 104859, 2020. ISSN 0305-0548. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0305054819303016>.

FERNANDES, R.; NETO, T.; COELHO, L.; COELHO, S. Planejamento de rotas para robÔs de inspeÇÃo usando um algoritmo hÍbrido de colÔnia de formigas e algoritmo cultural. 12 2019.

GOLDBARG, E.; GOLDBARG, M.; SOUZA, G. Particle swarm optimization algorithm for the traveling salesman problem. In: ____. [S.l.: s.n.], 2008. ISBN 978-953-7619-10-7.

GOLDBARG, M.; LUNA, H. **Otimização combinatória e programação linear: modelos e algoritmos**. ELSEVIER EDITORA, 2000. ISBN 9788535205411. Disponível em: <https://books.google.com.br/books?id=QJuqtAEACAAJ>.

GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization, and Machine Learning**. 1. ed. [S.l.]: Addison-Wesley Professional, 1989. ISBN 0201157675,9780201157673.

GOLDBERG, D. E. *et al.* Alleles, loci, and the traveling salesman problem. In: LAWRENCE ERLBAUM, HILLSDALE, NJ. **Proceedings of an international conference on genetic algorithms and their applications**. [S.l.], 1987. v. 154, p. 154–159.

GOLDEN, B. L.; LEVY, L.; VOHRA, R. The orienteering problem. **Naval Research Logistics (NRL)**, v. 34, n. 3, p. 307–318, 1987. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/1520-6750%28198706%2934%3A3%3C307%3A%3AAID-NAV3220340302%3E3.0.CO%3B2-D>.

GROUP, B. U. **BU-1006: Cost of Mobile and Renewable Power**. 2017. Disponível em: <https://batteryuniversity.com/index.php/learn/article/bu_1006_cost_of_mobile_power>. Acesso em: 29 nov. 2019.

GUTIN, G.; PUNNEN, A. The traveling salesman problem and its variations. v. 12, 01 2002.

HAMEED, I. A. Path planning for line marking robots using 2d dubins' path. In: HASSANIEN, A. E.; SHAALAN, K.; GABER, T.; AZAR, A. T.; TOLBA, M. F. (Ed.). **Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016**. Cham: Springer International Publishing, 2017. p. 900–910. ISBN 978-3-319-48308-5.

HASSANAT, A.; ALKAFAWEEN, E.; ALNAWAISEH, N.; ABBADI, M.; ALKASASSBEH, M.; ALHASANAT, M. Enhancing genetic algorithms using multi mutations: Experimental results on the travelling salesman problem. **International Journal of Computer Science and Information Security**, v. 14, p. 785–801, 09 2016.

HELD, M.; KARP, R. M. A dynamic programming approach to sequencing problems. **Journal of the Society for Industrial and Applied Mathematics**, v. 10, n. 1, p. 196–210, 1962. Disponível em: <https://doi.org/10.1137/0110015>.

Hou, N.; Wang, H.; Yu, M.; Chen, L.; Cao, Z.; Zheng, J.; Man, Z. Robotic fish path planning in complex environment. In: **2019 Chinese Control Conference (CCC)**. [S.l.: s.n.], 2019. p. 4519–4524.

HUSSAIN, A.; MUHAMMAD, Y. S.; SAJID, M. N.; HUSSAIN, I.; SHOUKRY, A. M.; GANI, S. Genetic algorithm for traveling salesman problem with modified cycle crossover operator. v. 2017, p. 7, 2017. Disponível em: <10.1155/2017/7430125>.

JEBARI, K. Selection methods for genetic algorithms. **International Journal of Emerging Sciences**, v. 3, p. 333–344, 12 2013.

KARA, I.; BICAKCI, P. S.; DERYA, T. New formulations for the orienteering problem. **Procedia Economics and Finance**, v. 39, p. 849 – 854, 2016. ISSN 2212-5671. 3rd GLOBAL CONFERENCE on BUSINESS, ECONOMICS, MANAGEMENT and TOURISM. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2212567116302520>.

KELLERER, H.; PFERSCHY, U.; PISINGER, D. Multiple knapsack problems. In: ____. **Knapsack Problems**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 285–316. ISBN 978-3-540-24777-7. Disponível em: <https://doi.org/10.1007/978-3-540-24777-7_10>.

KENNEDY, J. Particle swarm optimization. In: ____. **Encyclopedia of Machine Learning**. Boston, MA: Springer US, 2010. p. 760–766. ISBN 978-0-387-30164-8. Disponível em: <https://doi.org/10.1007/978-0-387-30164-8_630>.

KRAMER, O. **Genetic Algorithm Essentials**. Springer International Publishing, 2017. (Studies in Computational Intelligence). ISBN 9783319521565. Disponível em: <https://books.google.com.br/books?id=NxLcDQAAQBAJ>.

LEE, M.-T.; CHEN, B.-Y.; LU, W.-C. Failure-robot path complementation for robot swarm mission planning. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 9, n. 18, p. 3756, 2019.

LEHUéDé, F.; PéTON, O.; TRICOIRE, F. A lexicographic minimax approach to the vehicle routing problem with route balancing. **European Journal of Operational Research**, v. 282, n. 1, p. 129 – 147, 2020. ISSN 0377-2217. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0377221719307489>.

Li, Q. 3d cubic bezier curves for multi-target path planning for autonomous underwater vehicles. In: **OCEANS 2019 - Marseille**. [S.l.: s.n.], 2019. p. 1–5.

LIM, S. M.; SULTAN, A. B. M.; SULAIMAN, M. N.; MUSTAPHA, A.; LEONG, K. Crossover and mutation operators of genetic algorithms. **International journal of machine learning and computing**, v. 7, n. 1, p. 9–12, 2017.

LIU, Y.; BUCKNALL, R. Path planning algorithm for unmanned surface vehicle formations in a practical maritime environment. **Ocean Engineering**, v. 97, p. 126 – 144, 2015. ISSN 0029-8018. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0029801815000165>.

LIU, Y.; BUCKNALL, R. Efficient multi-task allocation and path planning for unmanned surface vehicle in support of ocean operations. **Neurocomputing**, v. 275, p. 1550 – 1566, 2018. ISSN 0925-2312. Disponível em: <http://www.sciencedirect.com/science/article/pii/S092523121731617X>.

MORAIS, C.; NASCIMENTO, T.; BRITO, A.; BASSO, G. A 3d anti-collision system based on artificial potential field method for a mobile robot. In: . [S.l.: s.n.], 2017. p. 308–313.

MURANO, A.; PERELLI, G.; RUBIN, S. Multi-agent path planning in known dynamic environments. In: CHEN, Q.; TORRONI, P.; VILLATA, S.; HSU, J.; OMICINI, A. (Ed.). **PRIMA 2015: Principles and Practice of Multi-Agent Systems**. Cham: Springer International Publishing, 2015. p. 218–231. ISBN 978-3-319-25524-8.

NAGARAJAN, B.; LI, Y.; SUN, Z.; QIN, R. A routing algorithm for inspecting grid transmission system using suspended robot: Enhancing cost-effective and energy efficient infrastructure

maintenance. **Journal of Cleaner Production**, v. 219, p. 622 – 638, 2019. ISSN 0959-6526. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0959652619304779>.

NERY, S. W. L. Análise de operadores de cruzamento genético aplicados ao problema do caixeiro viajante. 01 2017.

OLIVER, I.; SMITH, D.; HOLLAND, J. R. Study of permutation crossover operators on the traveling salesman problem. In: HILLSDALE, NJ: L. ERLHAUM ASSOCIATES, 1987. **Genetic algorithms and their applications: proceedings of the second International Conference on Genetic Algorithms: July 28-31, 1987 at the Massachusetts Institute of Technology, Cambridge, MA**. [S.l.], 1987.

Orozco-Rosas, U.; Picos, K.; Montiel, O. Hybrid path planning algorithm based on membrane pseudo-bacterial potential field for autonomous mobile robots. **IEEE Access**, v. 7, p. 156787–156803, 2019.

PACHECO, A. **Otimização por enxame de partículas - PSO**. 2016. Disponível em: <http://computacaointeligente.com.br/algoritmos/otimizacao-por-enxame-de-particulas/>. Acesso em: 28 set. 2018.

PANDA, M. R.; PRIYADARSHINI, R.; PRADHAN, S. An optimal path planning for multiple mobile robots using ais and ga: A hybrid approach. In: PRASATH, R.; VUPPALA, A. K.; KATHIRVALAVAKUMAR, T. (Ed.). **Mining Intelligence and Knowledge Exploration**. Cham: Springer International Publishing, 2015. p. 334–346. ISBN 978-3-319-26832-3.

PAPACHRISTOS, C.; MASCARICH, F.; KHATTAK, S.; DANG, T.; ALEXIS, K. Localization uncertainty-aware autonomous exploration and mapping with aerial robots using receding horizon path-planning. **Autonomous Robots**, v. 43, n. 8, p. 2131–2161, Dec 2019. ISSN 1573-7527. Disponível em: <https://doi.org/10.1007/s10514-019-09864-1>.

PARSOPOULOS, K.; VRAHATIS, M. **Particle Swarm Optimization and Intelligence: Advances and Applications**. [S.l.: s.n.], 2010. ISBN [ISBN-10: 1615206663] [ISBN-13: 978-1-61520-666-7].

PAVAI, G.; GEETHA, T. V. A survey on crossover operators. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 49, n. 4, p. 72:1–72:43, dez. 2016. ISSN 0360-0300. Disponível em: <http://doi.acm.org/10.1145/3009966>.

PETRINIĆ, T.; BREZAK, M.; PETROVIĆ, I. Time-optimal velocity planning along predefined path for static formations of mobile robots. **International Journal of Control, Automation and Systems**, v. 15, n. 1, p. 293–302, Feb 2017. ISSN 2005-4092. Disponível em: <https://doi.org/10.1007/s12555-015-0192-y>.

PORTO, F. R. Abordagem comparativa de técnicas metaheurísticas na elaboração automática de quadros horários, com enfoque nas técnicas: Algoritmos genéticos e algoritmo de seleção clonal. 2010.

POZO, A.; ISHIDA, C.; SPINOSA, E.; RODRIGUES, E. M. Computação evolutiva. 2005.

Rabbani, M.; Tahaei, Z.; Farrokhi-Asl, H.; Saravi, N. A. Using meta-heuristic algorithms and hybrid of them to solve multi compartment vehicle routing problem. In: **2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)**. [S.l.: s.n.], 2017. p. 1022–1026. ISSN 2157-362X.

S., A. J. P. L.; R., A. L. V.; F., N. Planejamento de rota por meio de algoritmo genÉtico para um enxame de robÔs. **Simpósio Brasileiro de Automação Inteligente**, Oct 2017.

SERAPIAO, A. B. d. S. Fundamentos de otimizaÃ§Ãpor inteligÃde enxames: uma visÃgeral. **Sba: Controle AutomaÃ§ÃSociedade Brasileira de Automatica**, scielo, v. 20, p. 271 – 304, 09 2009. ISSN 0103-1759. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-17592009000300002&nrm=iso>.

SERRANTOLA, W. G.; BUENO, J. N.; GRASSI, V. Planejamento de rota de um manipulador espacial planar de base livre flutuante com dois bra ¸cos utilizando rrt*. 10 2017.

SILVEIRA, L.; MAFFEI, R.; BOTELHO, S.; DREWS-JR, P.; DE, A.; BICHO, A.; FILHO, N. D. Space d*: A path-planning algorithm for multiple robots in unknown environments. **Journal of the Brazilian Computer Society**, v. 18, p. 363–373, 11 2012.

SIQUEIRA, P.; SCHEER, S.; STEINER, M. A recurrent neural network to traveling salesman problem. In: ____. [S.l.: s.n.], 2008. ISBN 978-953-7619-10-7.

SIVANANDAM, S. N.; DEEPA, S. N. **Introduction to Genetic Algorithms**. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2007. ISBN 354073189X, 9783540731894.

SOLOMON, M. M. **VRPTW Benchmark Problems**. 2008. Disponível em: <https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/>. Acesso em: 01 dez. 2019.

SONI, N.; KUMAR, T. Study of various mutation operators in genetic algorithms. In: . [S.l.: s.n.], 2014.

Sudhakara, P.; Ganapathy, V.; Sundaran, K. Route planning of a wheeled mobile robot (wmr) using enhanced artificial potential field (e-apf) method. In: **2018 International Conference on Communication, Computing and Internet of Things (IC3IoT)**. [S.l.: s.n.], 2018. p. 12–15. ISSN null.

TANG, B.; ZHU, Z.; LUO, J. Hybridizing particle swarm optimization and differential evolution for the mobile robot global path planning. **International Journal of Advanced Robotic Systems**, v. 13, n. 3, p. 86, 2016. Disponível em: <https://doi.org/10.5772/63812>.

WANG, Z.; REN, X.; JI, Z.; HUANG, W.; WU, T. A novel bio-heuristic computing algorithm to solve the capacitated vehicle routing problem based on adleman–lipton model. **Biosystems**, v. 184, p. 103997, 2019. ISSN 0303-2647. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0303264719301601>.

WU, X.; SUN, C.; ZOU, T.; LI, L.; WANG, L.; LIU, H. Svm-based image partitioning for vision recognition of agv guide paths under complex illumination conditions. **Robotics and Computer-Integrated Manufacturing**, v. 61, p. 101856, 2020. ISSN 0736-5845. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0736584518305647>.

XU, X.; YANG, Y.; PAN, S. Motion planning for mobile robots. In: RóKA, R. (Ed.). **Advanced Path Planning for Mobile Entities**. Rijeka: IntechOpen, 2018. cap. 8. Disponível em: <https://doi.org/10.5772/intechopen.76895>.

YADAV, S.; SOHAL, A. Study of the various selection techniques in genetic algorithms. 07 2017.

YAN, F. Autonomous vehicle routing problem solution based on artificial potential field with parallel ant colony optimization (aco) algorithm. **Pattern Recognition Letters**, v. 116, p. 195 – 199, 2018. ISSN 0167-8655. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167865518308304>.

Yu, J.; Schwager, M.; Rus, D. Correlated orienteering problem and its application to informative path planning for persistent monitoring tasks. In: **2014 IEEE/RSJ International Conference on Intelligent Robots and Systems**. [S.l.: s.n.], 2014. p. 342–349.

ZHANG, S.; CHEN, M.; ZHANG, W.; ZHUANG, X. Fuzzy optimization model for electric vehicle routing problem with time windows and recharging stations. **Expert Systems with Applications**, p. 113123, 2019. ISSN 0957-4174. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0957417419308401>.

ZUO, X.; XIAO, Y.; YOU, M.; KAKU, I.; XU, Y. A new formulation of the electric vehicle routing problem with time windows considering concave nonlinear charging function. **Journal of Cleaner Production**, v. 236, p. 117687, 2019. ISSN 0959-6526. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0959652619325375>.

APÊNDICE A – CÓDIGO DO PROJETO

O código desenvolvido nesta pesquisa está disponível em um repositório online, o GitHub.

https://github.com/killdary/genetic_algorithm_route_calculation/settings