UNIVERSIDADE ESTADUAL DO CEARÁ

CENTRO DE CIÊNCIAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MESTRADO ACADÊMICO EM CIÊNCIA DA COMPUTAÇÃO

LEONARDO FERREIRA DA COSTA

ALGORITMOS PARA OTIMIZAÇÃO DE COBERTURA DE ALVOS POR VEÍCULOS AÉREOS NÃO TRIPULADOS

FORTALEZA – CEARÁ

2020

LEONARDO FERREIRA DA COSTA

ALGORITMOS PARA OTIMIZAÇÃO DE COBERTURA DE ALVOS POR VEÍCULOS AÉREOS NÃO TRIPULADOS

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Orientador: Prof. Dr. Leonardo Sampaio Rocha

Co-Orientador: Prof. Dr. Gustavo Augusto Lima de Campos

FORTALEZA – CEARÁ

2020

Dados Internacionais de Catalogação na Publicação
Universidade Estadual do Ceará
Sistema de Bibliotecas

Costa, Leonardo Ferreira da.
Algoritmos para otimização de cobertura de alvos por veículos aéreos não tripulados [recurso eletrônico] / Leonardo Ferreira da Costa. – 2020.
99 f. : il.

Trabalho de Conclusão de Curso (Mestrado acadêmico) – Universidade Estadual do Ceará, Centro de Ciências e Tecnologia, Curso de Mestrado Acadêmico em Ciência da Computação, Fortaleza, 2020.

Orientação: Prof. Dr. LEONARDO SAMPAIO ROCHA.

1. Veículos aéreos não tripulados. 2. Posicionamento de UAVs. 3. Algoritmos de Agrupamento. 4. Algoritmos de Otimização. 5. Teoria dos Grafos. I. Título.

GOVERNO DO ESTADO DO CEARÁ
SECRETARIA DE CIÊNCIA, TECNOLOGIA E EDUCAÇÃO SUPERIOR
UNIVERSIDADE ESTADUAL DO CEARÁ
MESTRADO ACADÊMICO EM CIÊNCIA DA COMPUTAÇÃO - MACC

ATA DE DEFESA PÚBLICA DE DISSERTAÇÃO DE MESTRADO

Aos quatro dias de fevereiro de dois mil e vinte, no(a) miniauditório do MACC, realizou-se a sessão pública de defesa da dissertação de LEONARDO FERREIRA DA COSTA, aluno(a) regularmente matriculado(a) no curso MESTRADO ACADÊMICO EM CIÊNCIA DA COMPUTAÇÃO - MACC, Intitulada: **Algoritmos para Otimização de Cobertura de Alvos por Veículos Aéreos não Tripulados**. A Banca Examinadora reuniu-se no horário de 14:00h às 14 horas, sendo constituída por: Prof. Pós-Dr. LEONARDO SAMPAIO ROCHA (Orientador e Presidente da Banca/UECE), Prof. Dr. GUSTAVO AUGUSTO LIMA DE CAMPOS (Coorientador/UECE), Prof. Dr. GERARDO VALDISIO RODRIGUES VIANA (UECE) e Prof. Dr. PAULO ANTONIO LEAL REGO (UFC). Inicialmente o(a) mestrando(a) expôs seu trabalho e a seguir foi submetido(a) à arguição pelos membros da Banca, dispondo cada membro de tempo para tal. Finalmente a Banca reuniu-se em separado e concluiu por considerar o(a) mestrando(a) APROVADO, por sua dissertação e sua defesa pública. Eu, Prof. Pós-Dr. LEONARDO SAMPAIO ROCHA, orientador(a) e presidente da banca, lavrei a presente ata que será assinada por mim e os demais membros. Fortaleza, 04 de Fevereiro de 2020.

Prof. Pós-Dr. LEONARDO SAMPAIO ROCHA
(Orientador e Presidente da Banca/UECE)

Prof. Dr. GUSTAVO AUGUSTO LIMA DE CAMPOS
(Coorientador/UECE)

Prof. Dr. GERARDO VALDISIO RODRIGUES VIANA
(UECE)

Prof. Dr. PAULO ANTONIO LEAL REGO
(UFC)

Aos meus pais e aos meus colegas, pelo incentivo e pelo apoio constantes.

AGRADECIMENTOS

Primeiramente a Deus que permitiu que tudo isso acontecesse, ao longo de minha vida, e não somente nestes anos como universitária, mas que em todos os momentos é o maior mestre que alguém pode conhecer.

A minha família, pelo amor, incentivo e apoio incondicional.

A esta universidade, seu corpo docente, direção e administração que oportunizaram a janela que hoje vislumbro um horizonte superior, eivado pela acendrada confiança no mérito e ética aqui presentes.

Agradeço a todos os professores por me proporcionar o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional, por tanto que se dedicaram a mim, não somente por terem me ensinado, mas por terem me feito aprender. a palavra mestre, nunca fará justiça aos professores dedicados aos quais sem nominar terão os meus eternos agradecimentos.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

"Não se pode aprender nada de uma lição que não venha acompanhada da dor. Já que não se pode conseguir nada sem um sacrifício."

(Hiromu Arakawa)

RESUMO

Veículos aéreos não tripulados, ou UAVs, são cada vez mais populares para uma variedade de aplicações, uma das quais é o rastreamento de alvos em uma região de interesse específica. Nesse contexto, os UAVs devem ser posicionados de modo que o maior número de alvos na região seja coberto usando o menor número possível de UAVs. Além disso, a conectividade entre os UAVs e a estação base deve ser garantida. Os métodos de agrupamento de alvos são uma possibilidade para o posicionamento eficiente de UAVs, mas não garantem a conectividade da rede de comunicação entre UAVs e estação base. Para garantir a conectividade, pode-se visualizar toda a estrutura da rede de comunicação desses UAVs como um gráfico, onde os conceitos da teoria de grafos podem ser aplicados para formular restrições para os algoritmos de otimização. Este trabalho propõe a criação de algoritmos de alocação de drones em uma região de interesse, cobrindo o maior número possível de alvos naquela região, com auxilio de métodos de agrupamento e de teoria dos grafos. Foram dois algoritmos propostos, um específico para alvos fixos e outro para alvos móveis. Para os cenários considerados, os resultados foram bastante promissores, onde as taxas de cobertura de alvos superaram 60%, mesmo nos piores casos.

Palavras-chave: Veículos aéreos não tripulados. Posicionamento de UAVs. Algoritmos de Agrupamento. Algoritmos de Otimização. Teoria dos Grafos

ABSTRACT

Unmanned aerial vehicles, or UAVs, are increasingly popular for a variety of applications, one of which is target tracking in a specific region of interest. In this context, UAVs should be positioned so that the largest number of targets in the region is covered using as few UAVs as possible. In addition, connectivity between UAVs and the base station must be ensured. Target grouping methods are a possibility for efficient positioning of UAVs, but do not guarantee connectivity of the communication network between UAVs and base station. To ensure connectivity, you can view the entire communication network structure of these UAVs as a graph, where graph theory concepts can be applied to formulate constraints for optimization algorithms. This work proposes the creation of drone allocation algorithms in a region of interest, covering as many targets as possible in that region, with the aid of clustering methods and graph theory. There were two proposed algorithms, one specific for fixed targets and one for moving targets. For the scenarios considered, the results were very promising, where target coverage rates exceeded 60%, even in the worst cases.

Keywords: Unmanned aerial vehicles. UAVs Deployment. Clustering Algorithms. Optimzation Algorithms. Graph Theory

LISTA DE ILUSTRAÇÕES

Figura 1 – As sete pontes de Konigsberg. 21
Figura 2 – Representação das sete pontes de Konigsberg como um grafo. 22
Figura 3 – Representação de um grafo G. 23
Figura 4 – Possibilidades de relacionamentos uv, unidirecional e bidirecional. . . . 23
Figura 5 – Grafo G e subgrafo G'. 24
Figura 6 – Grafo planar (A) e grafo não planar (B). 25
Figura 7 – Representação de uma rede de fluxo F. 26
Figura 8 – Construindo quatro grupos a partir de dados não rotulados. 28
Figura 9 – Exemplo de conjunto de dados para agrupar. 31
Figura 10 – Gráfico para análise do método do cotovelo usando o parâmetro "distorção". 31
Figura 11 – Gráfico para análise do método do cotovelo usando o parâmetro "inercia". 32
Figura 12 – Aplicação do *Kneedle algorithm* para detecção de joelho/cotovelo em curva. 33
Figura 13 – Construindo quatro grupos a partir de dados não rotulados. 36
Figura 14 – Exemplo de frentes de Pareto. 40
Figura 15 – Exemplos de polígonos. 41
Figura 16 – Algumas denominações de polígonos de acordo com o número de lados. 42
Figura 17 – Alguns exemplos de diferentes tipos de polígonos. 42
Figura 18 – Exemplo de centroide de um polígono. 43
Figura 19 – Estrutura de um agente. 44
Figura 20 – UAV para captura de imagens. 46
Figura 21 – Boeing Insitu ScanEagle. 47
Figura 22 – Boeing Insitu Integrator. 48
Figura 23 – Frentes de Pareto para diferentes tamanhos de N e P. 52
Figura 24 – Soma das altitudes de UAVs implantados. 52
Figura 25 – Número médio de alvos cobertos pelos UAVs implantados para \|P\| = 108, 147, 300... 53
Figura 26 – Tempo computacional. 53
Figura 27 – Custo de implantação das soluções ótimas. 54

Figura 28 – Custo da conectividade dos UAVs. 54
Figura 29 – Ilustração do *street graph* S. 56
Figura 30 – Exemplo visual das fases do Algoritmo 7: fase do gráfico de decisão (esquerda), fase de formação de *cluster* (centro) e fase de otimização da cobertura do drone (direita) 62
Figura 31 – Resultados para o Algoritmo 7 (esquerda) e para o Algoritmo 8 (direita). No centro tem-se o o primeiro gráfico de decisão de iteração que contribui (junto com outras 5 iterações) para gerar a figura no lado direito. 63
Figura 32 – UAV genérica para observação de alvos. 68
Figura 33 – Grafo de comunicação C. 69
Figura 34 – Alvos espalhados pela região genérica R. 70
Figura 35 – Alocação de drones e cobertura de alvos sobre a região R. 71
Figura 36 – Representação esquemática da solução padrão utilizada nos algoritmos propostos. 71
Figura 37 – Visualização de uma instancia gerada no Netlogo de uma região R de dimensões $400x400$, com 200 alvos. 82
Figura 38 – Resultados para os algoritmos propostos aplicados em uma região de $10000\ m^2$ e com a estação base posicionada na origem (0,0). 85
Figura 39 – Resultados para os algoritmos propostos aplicados em uma região de $10000\ m^2$ e com a estação base posicionada no meio. 86
Figura 40 – Resultados para os algoritmos propostos aplicados em uma região de $10000\ m^2$ e com a estação base posicionada aleatoriamente. 86
Figura 41 – Resultados para os algoritmos propostos aplicados em uma região de $250000\ m^2$ e com a estação base posicionada na origem (0,0). 86
Figura 42 – Resultados para os algoritmos propostos aplicados em uma região de $250000\ m^2$ e com a estação base posicionada no meio. 87
Figura 43 – Resultados para os algoritmos propostos aplicados em uma região de $250000\ m^2$ e com a estação base posicionada aleatoriamente. 87
Figura 44 – Resultados para os algoritmos propostos aplicados em uma região de $1000000\ m^2$ e com a estação base posicionada na origem (0,0). 87
Figura 45 – Resultados para os algoritmos propostos aplicados em uma região de $1000000\ m^2$ e com a estação base posicionada no meio. 88

Figura 46 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 e com a estação base posicionada aleatoriamente. 88
Figura 47 – Quantidade média de HUBs utilizadas nos testes em regiões de 10000 m^2. 89
Figura 48 – Quantidade média de HUBs utilizadas nos testes em regiões de 250000 m^2. 90
Figura 49 – Quantidade média de HUBs utilizadas nos testes em regiões de 1000000 m^2. 90
Figura 50 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 e com a estação base posicionada na origem (0,0), fixando a quantidade de UAVs máxima. 90
Figura 51 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 e com a estação base posicionada no meio, fixando a quantidade de UAVs máxima. 91
Figura 52 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 e com a estação base posicionada aleatoriamente, fixando a quantidade de UAVs máxima. 91

LISTA DE TABELAS

Tabela 1 – Complexidades dos métodos da literatura utilizados nos algoritmos propostos. 77
Tabela 2 – Os tipos de cenários para a geração de instâncias. 81
Tabela 3 – Complexidades dos algoritmos propostos. 88
Tabela 4 – Resultados do algoritmo de alvos móveis com relação a base de dados externa. 92
Tabela 5 – Resultados do algoritmo de alvos móveis com relação a base de dados externa, limitando a quantidade de UAVs máxima. 92

LISTA DE ALGORITMOS

Algoritmo 1 – DFS. 27
Algoritmo 2 – K-MEANS. 29
Algoritmo 3 – *Kneedle algorithm*. 35
Algoritmo 4 – DBSCAN 37
Algoritmo 5 – *Trajectory clustering algorithm* 39

Algoritmo 6 – BBFWA 60
Algoritmo 7 – *Max-capacity clustering algorithm*. 61
Algoritmo 8 – *Iterative clustering algorithm for 5G UAV-BS placement*. 63

Algoritmo 9 – MÉTODO PARA VERIFICAR MELHOR VALOR DE k PARA O K-MEANS 73
Algoritmo 10 – CONSTRUÇÃO DO GRAFO DE COMUNICAÇÃO C 73
Algoritmo 11 – VERIFICAR COMPONENTES CONEXAS DO GRAFO C 74
Algoritmo 12 – ALGORITMO PARA POSICIONAR HUBS PARA COMUNICAÇÃO ENTRE UAVS 75
Algoritmo 13 – ALGORITMO PARA ALVOS ESTÁTICOS 76
Algoritmo 14 – ALGORITMO PARA ALVOS MÓVEIS 77

LISTA DE ABREVIATURAS E SIGLAS

BBFWA	Bare Bones Fireworks Algorithm
DBSCAN	Density Based Clustering Algorithm
DFS	Depth-First Search
IoT	Internet of Things
MDP	Markov Decision Process
UAV	Unmanned Aerial Vehicle
UAV-BS	Unmanned Aerial Vehicle Base Station
VoI	Valor da Informação
VTOL	Vertical Take-Off and Landing

SUMÁRIO

1 INTRODUÇÃO 17
1.1 MOTIVAÇÃO 19
1.1.1 Objetivos 19
1.1.1.1 Geral 19
1.1.1.2 Específicos 20
1.2 ORGANIZAÇÃO DO DOCUMENTO 20
2 FUNDAMENTAÇÃO TEÓRICA 21
2.1 TEORIA DOS GRAFOS 21
2.1.1 Definições 22
2.1.2 Rede de fluxo 25
2.1.3 *Depth-First Search* 26
2.2 ALGORITMOS DE AGRUPAMENTO 27
2.2.1 K-Means 28
2.2.1.1 Método do cotovelo 30
2.2.1.2 Método de detecção de joelhos 31
2.2.2 DBSCAN 36
2.2.3 *Trajectory clustering algorithm* 37
2.3 EFICIÊNCIA DE PARETO 39
2.4 POLÍGONOS 41
2.4.1 Centroide de um polígono 42
2.5 AGENTES INTELIGENTES 43
2.6 VEÍCULOS AÉREOS NÃO TRIPULADOS 45
2.6.1 Tipos de UAVs 46
2.6.2 Exemplos de aplicações de UAVs 47
3 TRABALHOS RELACIONADOS 49
3.1 OPTIMIZATION OF MOBILE SENSOR COVERAGE WITH UAVS . . . 49
3.2 AN ALGORITHM OF EFFICIENT PROACTIVE PLACEMENT OF AUTONOMOUSDRONES FOR MAXIMUM COVERAGE IN CELLULAR NETWORKS 55
3.3 EFFICIENT DRONE PLACEMENT FOR WIRELESS SENSOR NETWORKS COVERAGE BY BARE BONES FIREWORKS ALGORITHM 58

3.4 PLACEMENT OF 5G DRONE BASE STATIONS BY DATAFIELD CLUSTERING 61
3.5 OUTROS TRABALHOS RELACIONADOS 63
4 ABORDAGEM PROPOSTA 67
4.1 MODELAGEM DO PROBLEMA 67
4.2 ALGORITMO PARA ALVOS FIXOS 71
4.3 ALGORITMO PARA ALVOS MÓVEIS 74
4.4 ESTUDO DE COMPLEXIDADE DOS ALGORITMOS PROPOSTOS . . . 76
5 METODOLOGIA PARA OS EXPERIMENTOS 80
5.1 IMPLEMENTAÇÃO DOS ALGORITMOS PROPOSTOS 80
5.2 GERAÇÃO DE INSTANCIAS 80
5.3 BASE DE DADOS EXTERNA 82
5.4 EXPERIMENTOS 83
6 RESULTADOS 85
6.1 RESULTADOS DOS TESTES DOS ALGORITMOS PROPOSTOS UTILIZANDO AS INSTÂNCIAS GERADAS 85
6.2 RESULTADOS DO ALGORITMO PARA ALVOS MÓVEIS UTILIZANDO A BASE DE DADOS EXTERNA 91
7 CONCLUSÃO 93
7.1 CONTRIBUIÇÕES DO TRABALHO 93
7.2 LIMITAÇÕES 93
7.3 TRABALHOS FUTUROS 94
REFERÊNCIAS 95

1 INTRODUÇÃO

O monitoramento de objetos ou seres vivos em determinadas regiões pode ser um desafio. As condições adversas de um determinada região podem fazer que não seja viável a implantação de estruturas fixas de monitoramento, principalmente pelo alto custo e pela dificuldade de interferência humana no determinado local. Assim, veículos aéreos não tripulados ou UAVs, ou simplesmente drones, são uma solução encontrada para este problema de cobertura de alvos.

O problema de cobertura de alvos, no contexto das UAVs, é um denominação genérica para todo tipo de problema em que se deseja posicionar uma determinada quantidade de drones em uma região para que eles possam monitorar determinados alvos presentes nesta região. Os alvos podem ser objetos ou seres vivos. O monitoramento de alvos pode ser a coleta de informações visuais do alvo através de um câmera acoplada ao drone, como considerado nos trabalhos de Saeed *et al.* (2017), Khan *et al.* (2017), Caillouet, Giroire e Razafindralambo (2018). Além do monitoramento visual do alvo, alguns trabalhos consideram a cobertura de alvos para outros fins, como coleta de informações de sensores espalhados ao longo de uma região, como nos trabalhos de Xu *et al.* (2016), Tuba *et al.* (2018).

Este problema é normalmente referenciado na literatura como posicionamento eficiente de drones, mas não se resume apenas em posicionar os drones com relação aos alvos, a rede de comunicação entre UAVs, e entre UAVs e estação(ões) base, também é considerada. No trabalho de Huang e Savkin (2018) foi proposto um modelo de otimização que maximiza a cobertura dos usuários e minimiza o custo de comunicação entre drones, onde foram propostos duas visualizações da situação a partir de grafos, um com relação a comunicação entre drones e estações bases, e outro que eles chamam de *street graph*, que considera os cruzamentos das ruas da cidade com vértices e possíveis pontos de alocação de drones.

Outro ponto a considerar é o tipo de cenário para o posicionamento de drones. Algumas pesquisas com ponto de vista prático, que consideram cenários realistas para a implantação das UAVs, focam na alocação distribuída de UAVs, como nos trabalhos de Saeed *et al.* (2017), Chauhan, Unnikrishnan e Figliozzi (2019), que seria fazer o posicionamento dos drones de forma que eles povoem a região e façam a cobertura da maior quantidade de terreno ou alvos possível, dado o cenário apresentado. Em contrapartida, as pesquisas que focam na teoria, que não consideram cenários realistas, se concentram na implantação ideal, ou seja, apresentam métodos para encontrar o posicionamento ótimo de drones, como mostrado nos trabalhos de

Zorbas *et al.* (2016), Strumberger *et al.* (2017).

Este trabalho tem o objetivo de propor algoritmos para calcular a quantidade de UAVs necessárias para o monitoramento em tempo integral de todos os alvos de um região. Além disso, os algoritmos devem garantir que a rede de comunicação entre drones e estação base seja conexa ao posicionar as UAVs de forma automatizada.

Neste trabalho a região considerada é genérica idealmente, porém não são consideradas grandes estruturas que podem prejudicar a movimentação das UAVs. Assim a região de interesse, a nível de contextualização, representariam grandes regiões rurais e esparsas. Dessa forma, os alvos poderiam ser animais ou maquinas agrícolas a serem observados pelos drones, por exemplo.

Os alvos a serem observados também são genéricos, apenas classificados com estáticos ou móveis, podendo representar objetos, veículos, pessoas, animais, e entre outros. As UAVs consideradas neste trabalho também são genéricas, porém podem ser definidas como dispositivos com o objetivo de observação, como os drones considerados pelo trabalho de Caillouet, Giroire e Razafindralambo (2018).

Inspirado em Huang e Savkin (2018), neste trabalho pretende-se visualizar toda a estrutura da rede de comunicação das UAVs como um grafo conexo, onde os drones seriam os vértices, e as arestas seriam formadas entre dois drones, ou entre drone e estação base, que estejam dentre do alcance de comunicação deles, a fim desta visualização facilitar na criação de mecanismos para garantir a conectividade da rede de comunicação.

Um mecanismo importante para auxiliar os algoritmos no posicionamento dos UAVs, são os algoritmos de agrupamento. Eles, no contexto deste trabalho, servem para indicar *clusters* (grupos) de alvos na região de observação, e indicar pontos candidatos a posicionamento de drones. Em Zorbas *et al.* (2016), em uma das abordagens propostas, foi utilizado o clássico algoritmo de agrupamento K-Means para agrupar alvos estáticos e posicionar os drones sobre os centros dos grupos.

Já o trabalho de Iellamo, Lehtomaki e Khan (2017) considera alvos móveis para agrupar e posicionar os drones sobre os grupos. Entretanto, os autores observaram que os algoritmos clássicos de agrupamento não são eficientes para alvos móveis, assim eles adaptaram um método proposto em Zhao *et al.* (2015), que, ao invés de agrupar os alvos, ele agrupa os pontos de trajetória dos alvos obtidos de um determinado período de observação e identifica pontos na região onde são mais suscetíveis a presença de vários alvos próximos. Esse pontos são chamados de *hotspots*, e sobre eles que os drones seriam posicionados.

Neste trabalho foram propostos dois algoritmos, um aplicado para situações com alvos fixos, e outro para alvos móveis. Em ambos os algoritmos, foram utilizados métodos de agrupamento para identificação de agrupamentos de alvos, no intuito de posicionar 1 UAV sobre esses grupos, portanto o grupo deve ser totalmente observável pelo drone. Além disso, eles foram combinados com a estratégia com grafos para verificar se a rede de comunicação entre drones e estação base é conexa, e fazer os devidos ajustes para garantir que a quantidade de drones encontradas pelos algoritmos, além de garantir a cobertura em tempo integral de todos os alvos, também mantenha a rede de comunicação conectada.

1.1 MOTIVAÇÃO

Ter a cobertura dos mais diversos tipo de regiões, tanto para monitoramento quanto para comunicação, é um necessidade que se tem na atualidade, e utilizar UAVs para essa tarefa é mais viável do que utilização de estruturas fixas no solo. A implantação de UAVs independe da situação em que a região de interesse se encontra, possibilitando a cobertura em regiões de difícil acesso via solo e áreas muito esparsas com pouca infraestrutura.

No entanto, além de cobrir toda região de interesse, deve-se prezar pela otimalidade da implantação, pois deseja-se também que os custo da operação sejam os mínimos possíveis. Assim, qualquer que seja o modelo de cobertura encontrado, tem-se que abranger a maior quantidade de área possível com a menor quantidade de UAVs. Além disso, deve-se garantir que a rede de comunicação entre os drones e a estação base seja conexa, para que as informações obtidas no monitoramento dos alvos sejam difundidas e cheguem com sucesso na estação base.

1.1.1 Objetivos

Neste tópico são apresentados os objetivos deste trabalho, onde em objetivo geral é apresentado a ideia central desenvolvida, e nos específicos, são apresentados os itens necessários para se alcançar o objetivo geral.

1.1.1.1 Geral

Propor, implementar e validar uma abordagem para determinar implantações de drones para coberturas de regiões de interesse, de forma que se encontre a melhor cobertura da região utilizando a menor quantidade de UAVs possíveis.

1.1.1.2 Específicos

a) Revisar a bibliografia sobre métodos de alocação de UAVs;
b) Identificar melhores métodos para agrupamento para calcular *clusters* de alvos, de forma que cada um seja totalmente observável por apenas 1 UAV;
c) Modelar o problema e gerar restrições que garantem a conectividade da rede de comunicação entre UAVs;
d) Propor algoritmos para determinar melhores implantações de UAVs e garantir a conectividade da rede de comunicação dos drones;
e) Validar as soluções encontradas em cenários realistas.

1.2 ORGANIZAÇÃO DO DOCUMENTO

Este documento está organizado da seguinte forma:

- No Capítulo 2 são apresentados os principais conceitos para contextualização com o problema abordado neste trabalho;
- No Capítulo 3 são relatados trabalhos relacionados com o presente estudo, que apresentam pesquisas que inspiraram e contribuíram para o desenvolvimento deste trabalho, apresentando diferentes estratégias para posicionamento de UAVs;
- No Capítulo 4 é definido o problema abordado neste trabalho, a modelagem dele, e os dois algoritmos propostos;
- No Capítulo 5 são detalhados os procedimentos aplicados para realização dos testes de avaliação dos algoritmos propostos;
- No Capítulo 6 são apresentados os resultados dos experimentos com os algoritmos propostos;
- E, no Capítulo 7, tem-se as conclusões deste trabalho, as contribuições, as limitações e os trabalhos futuros.

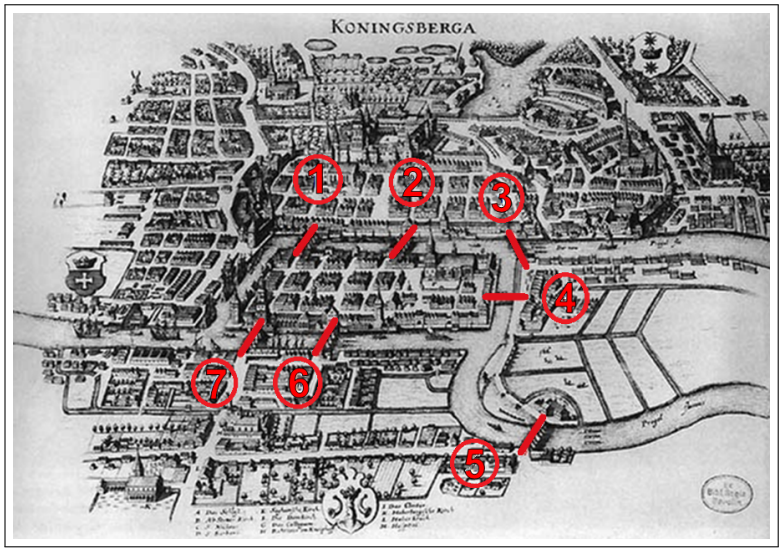
2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os principais conceitos utilizados neste trabalho, são eles: conceitos básicos de teoria dos grafos e alguns algoritmos em grafos; algoritmos de agrupamentos; definição de eficiência de Pareto; alguns conceitos sobre polígonos; introdução sobre agentes inteligentes; e algumas informações sobre veículos aéreos não tripulados ou simplesmente UAVs.

2.1 TEORIA DOS GRAFOS

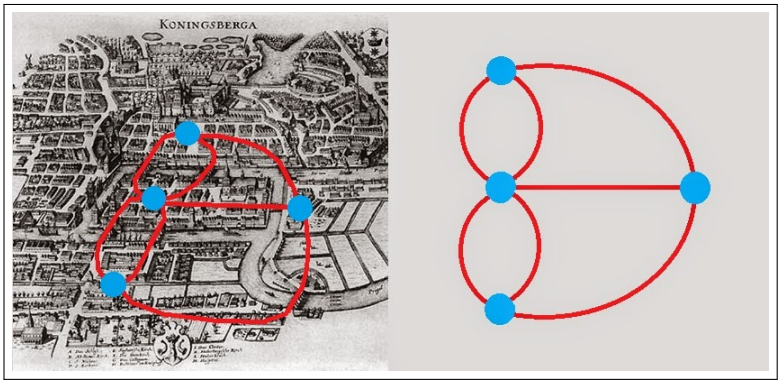
Grafos são estruturas utilizadas para modelar as relações de emparelhamento entre objetos. É uma representação simbólica de uma rede e de sua respectiva conectividade. Implica em uma abstração da realidade para que ela possa ser simplificada como um conjunto de nós associados entre si (GOLDBARG; GOLDBARG, 2012). As origens da teoria dos grafos podem ser atribuídas a Leonhard Euler, que concebeu em 1735 um problema que veio a ser conhecido como as "Sete Pontes de Konigsberg" (Figura 1) (EULER, 1741; HARJU, 2014). Nesse problema, alguém tinha que atravessar todas as pontes apenas uma vez e em uma sequência contínua, uma situação que o Euler provou não ter solução, representando-o como um conjunto de nós e relacionamentos (chamados de arestas), ou seja, um grafo (Figura 2). Isso levou a fundamentação da teoria dos grafos e suas melhorias subsequentes. Foi enriquecido nas últimas décadas por influências crescentes de estudos de redes sociais e complexas.

Figura 1 – As sete pontes de Konigsberg.



Fonte: Brito (2015).

Figura 2 – Representação das sete pontes de Konigsberg como um grafo.



Fonte: Brito (2015).

No contexto dos transportes terrestres, a maioria dos grafos formado pelas redes de transporte tem uma fundamentação espacial evidente, como as redes de rodovias, de trânsito e ferroviárias, que tendem a ser mais definidas pelos suas arestas do que pelos seus nós. Isso não é necessariamente o caso de todas as redes de transporte. Por exemplo, as redes marítima e aérea tendem a ser mais definidas por seus nós do que por suas arestas, já que as conexões entre os nós nesses casos, muitas vezes, não estão claramente definidas.

Um rede de telecomunicações também pode ser representada como um grafo, onde os telefones celulares e antenas podem ser representados como nós, enquanto as arestas podem ser chamadas telefônicas individuais. Os servidores, o núcleo da internet, também podem ser representados como nós dentro de um grafo, enquanto a infraestrutura física entre eles, ou seja, cabos de fibra ótica, podem atuar como arestas. Consequentemente, todas as redes de transporte podem ser representadas por grafos.

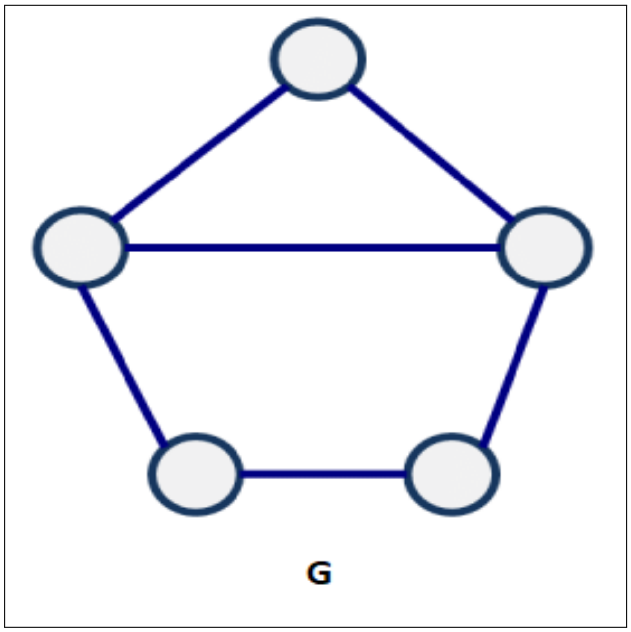
2.1.1 Definições

Formalmente, um grafo G é definido por um conjunto de vértices (nós) V que são conectados pelas arestas do conjunto A, portanto, $G = \{V, A\}$. A Figura 3 mostra a representação gráfica de um grafo. Um vértice u, onde $u \in V$, é um ponto terminal ou de interseção de G, sendo uma abstração de um local como uma cidade, uma divisão administrativa, um cruzamento de estradas ou um terminal de transporte (estações, terminais, portos e aeroportos) (GOLDBARG; GOLDBARG, 2012).

Dados dois vértices u e v, onde $\{u, v\} \in V$, uma aresta uv, onde $uv \in A$, diz-se que existe um relacionamento entre u e v, que pode ser uma representação de uma estrada entre duas cidades. A representação de uma aresta é comumente feita por setas, caso o relacionamento seja

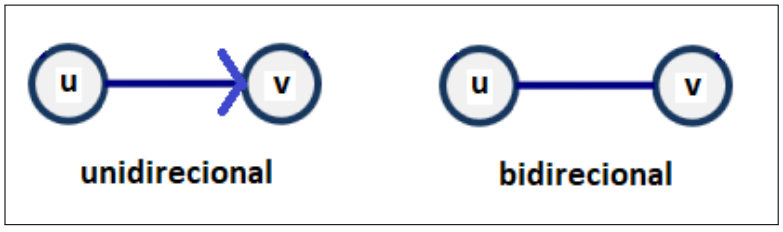
direcionado, isto é, em um mesma uníca direção, ou seja, se $uv \in A$ não necessariamente implica em $vu \in A$. Caso o relacionamento entre u e v seja bidirecional, $\{uv, vu\} \in A$, a representação é apenas uma linha simples entre os dois vértices (BONDY; MURTY, 2008; CORMEN *et al.*, 2009; GOLDBARG; GOLDBARG, 2012). A Figura 4 ilustra os dois tipos de representação de arestas.

Figura 3 – Representação de um grafo G.



Fonte: Elaborada pelo autor

Figura 4 – Possibilidades de relacionamentos uv, unidirecional e bidirecional.



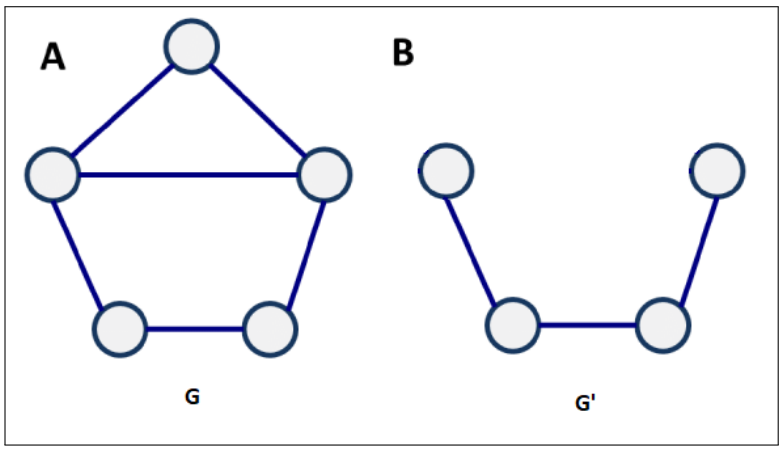
Fonte: Elaborada pelo autor

Um grafo pode ser não direcionado, o que significa que não há distinção entre os dois vértices associados a cada aresta, ou suas arestas podem ser direcionadas de um vértice para outro. Chama-se um grafo direcionado de dígrafo (GOLDBARG; GOLDBARG, 2012).

Existe também o conceito de subgrafo, onde um subgrafo $G' = \{V', A'\}$ é formado por uma determinada quantidade de vértices e arestas de $G = \{V, A\}$, como mostrado pela Figura 5 (BONDY; MURTY, 2008). A menos que o sistema de transporte global seja considerado em

seu todo, qualquer rede de transporte é, em teoria, um subgrafo de outro. Por exemplo, a rede de transporte rodoviário de uma cidade é um subgrafo de uma rede de transporte regional, que é em si um subgrafo de uma rede de transporte nacional.

Figura 5 – Grafo G e subgrafo G'.



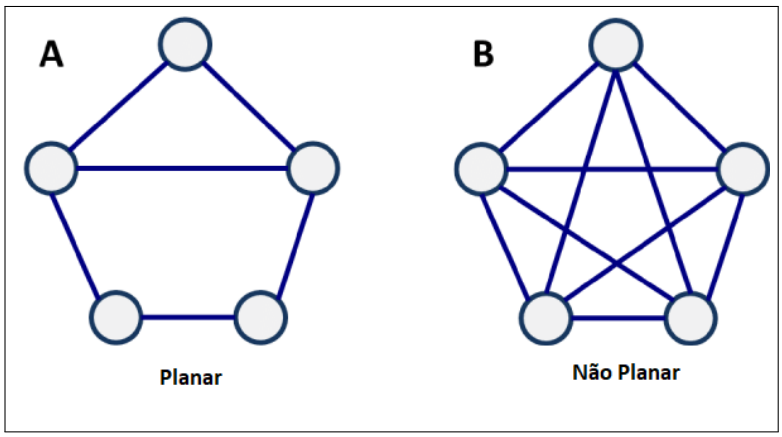
Fonte: Elaborada pelo autor

Um grafo pode ter uma representação onde todas as interseções de duas arestas é um vértice, é o que se chama de grafo planar (Figura 6A). Como esse grafo está localizado dentro de um plano, sua topologia é bidimensional. Esse é tipicamente o caso das redes elétricas, rodoviárias e ferroviárias, embora grande cuidado deva ser inferido à definição de nós (terminais, armazéns, cidades). O caso contrário seria o grafo não planar, onde não há vértices na intersecção de pelo menos duas arestas (Figura 6B) . Redes que podem ser consideradas de forma planar, como estradas, podem ser representadas como redes não planares. Isto implica uma terceira dimensão na topologia do grafo, uma vez que existe a possibilidade de um movimento "passar por cima" de outro movimento, como para o transporte aéreo e marítimo, ou um viaduto para uma estrada. Um grafo não planar tem potencialmente muito mais conexões do que um grafo planar (GOLDBARG; GOLDBARG, 2012).

Grafos costumam representar a ideia de fluxo de pessoas, carros, informações e entre outros. Assim, na teoria dos grafos também existe o conceito de percorrimento, que pode ser dos seguintes tipos:

- Passeio: É um sequencia alternante entre vértices e arestas de um grafo que começa e termina em vértices. Exemplo: Passeio $P = \{V_1, A_1, V_2, A_2, V_3\}$, onde $\{V_1, V_2, V_3\}$ são vértices e $\{A_1, A_2\}$ são arestas;
- Caminho: É um passeio cujo todos os vértices são distintos;

Figura 6 – Grafo planar (A) e grafo não planar (B).



Fonte: Elaborada pelo autor

- Trilha: É um passeio cujo todos as arestas são distintas;
- Circuito: É um caminho cujo o vértice inicial e o final são os mesmos;
- Ciclo: É um caminho cujo o vértice inicial e o final são os mesmos, e nenhuma aresta foi utilizada mais de uma vez. Um circuito pode ser considerado um ciclo se todas as arestas forem percorridas na mesma direção.

Um grafo simples não direcionado, no qual cada par de vértices distintos é conectado por uma aresta única, é chamado de grafo completo. Um dígrafo completo é um grafo direcionado no qual cada par de vértices distintos é conectado por um par de arestas únicas (uma em cada direção). Um grafo é descrito como conexo se existir um caminho entre quaisquer par de vértices (GOLDBARG; GOLDBARG, 2012).

2.1.2 Rede de fluxo

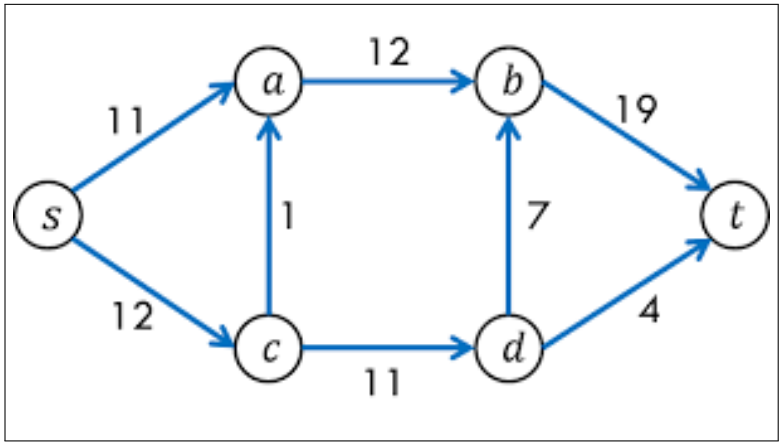
Uma rede de fluxo, ou rede de transporte, é um grafo direcionado onde cada aresta tem uma capacidade e pode receber um fluxo. Em um grafo G, a taxa de fluxo que passa em uma determinada aresta uv, onde $uv \in G$, tem que ser menor ou igual a capacidade que foi atribuída para essa aresta (CORMEN *et al.*, 2009).

Essas redes de fluxo são normalmente usadas para modelar problemas envolvendo o transporte de itens entre locais, usando um esquema de rotas com capacidade limitada. Exemplos incluem modelagem de tráfego em uma rede de estradas, fluido em uma rede de tubulações e fluxo de eletricidade em uma rede de componentes de circuito.

Formalmente, uma rede de fluxo F é definida como uma tupla $F = \{G, c, s, t\}$, onde: $G = \{V, A\}$ é um grafo direcionado; $c : E \rightarrow R_0^+$ é um mapeamento das arestas para os reais

não-negativos e $c(a)$ é chamado de capacidade da aresta a, onde $a \in A$; $s,t \in G$ são arestas especiais que representam, respectivamente, o inicio (*source*) e o fim (*sink*) da rede de fluxo. A Figura 7 apresenta uma visualização da rede de fluxo F descrita (BONDY; MURTY, 2008; CORMEN *et al.*, 2009).

Figura 7 – Representação de uma rede de fluxo F.



Fonte: Park (2015)

Um fluxo permitido em F é definido como: Seja $f(a)$, o fluxo que passa na aresta a, $a \in A$, e $c(a)$, a capacidade da aresta a, tem-se as seguintes restrições:

$$0 \leq f(a) \leq c(a) \tag{2.1}$$

Para todas as arestas a em A:

$$\sum_{a^+=u} f(a) = \sum_{a^-=u} f(a) \tag{2.2}$$

Para cada vértice u, $u \in V$ e $u \neq s,t$, onde a^+ e a^- denotam o vértice inicial e final da aresta a, respectivamente.

2.1.3 *Depth-First Search*

Busca em profundidade (*Depth-First Search* ou DFS) é um algoritmo de busca ou percorrer um grafo. É um algoritmo recursivo que usa a ideia de *backtracking*. Envolve buscas exaustivas de todos os nós indo para a frente visitando todos os vértices do grafo em um único

caminho, se possível, caso contrário retorna para algum no que ainda tenha algum caminho para algum outro vértice não visitado (CORMEN *et al.*, 2009; GOLDBARG; GOLDBARG, 2012).

O algoritmo clássico do DFS é implementado utilizando pilhas, e a ideia básica dele é o que se segue nos passos abaixo:

Algoritmo 1: DFS.

Entrada: G ← Grafo a ser percorrida na busca.

1 Escolha um nó inicial e marque como visitado.
2 Empurre todos os seus nós adjacentes para uma pilha.
3 **repita**
4 Remova o nó do topo da pilha para selecionar o próximo nó para visitar.
5 Marque este nó como visitado.
6 Empurrar todos os seus nós adjacentes não visitados na pilha.
7 **até** *a pilha estiver vazia*;
8 **retorna** *Ordem de visitação dos vértices.*

O DFS possui complexidade no pior caso de $O(V+A)$, onde dado um grafo $G = (V+A)$, onde V é o número de vértices de G e A é número de aresta de G.

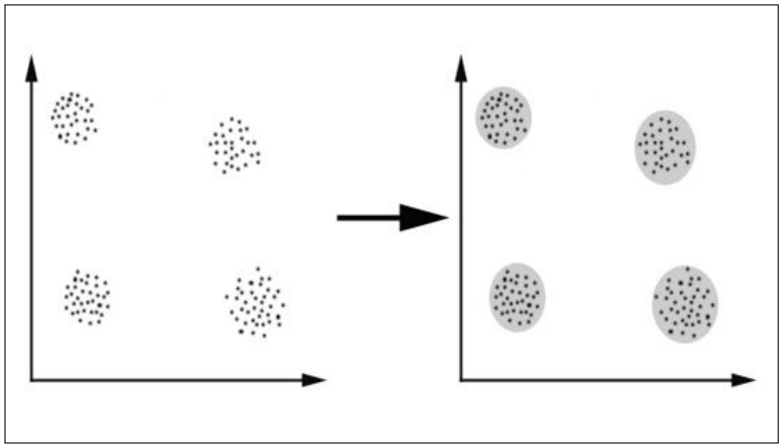
Este algoritmo tem algumas utilidades extras, como a identificação de componentes conexas de um grafo. Por exemplo, se um grafo G é desconexo e possuir mais de uma componente conexa, isto é, existe um ou mais vértices que completamente desconectados entre si em G, então varias DFS forem executadas partindo de todos os vértices de G (um de cada vez por execução), as listas que registram os vértices vistados em cada execução do algoritmo vão conter elementos diferentes em algumas das execuções. Se houver, duas listas onde não há interseção dos elementos entre elas, significa que G possui duas componentes conexas.

2.2 ALGORITMOS DE AGRUPAMENTO

Algoritmos de agrupamento ou clusterização (*clustering algorithms*) agrupam objetos de dados com base apenas nas informações encontradas nos dados que descrevem os objetos e seus relacionamentos (AGGARWAL, 2014). O objetivo é que os objetos dentro de um grupo (*cluster*) sejam similares, ou relacionados, uns aos outros e diferentes, ou não relacionados a objetos em outros grupos. Quanto maior a similaridade (ou homogeneidade) dentro de um grupo

e quanto maior a diferença entre os grupos, melhor ou mais distinto é o agrupamento. A Figura 8 mostra um exemplo de agrupamento de dados.

Figura 8 – Construindo quatro grupos a partir de dados não rotulados.



Fonte: Naik (2010).

Com o surgimento de muitos algoritmos de agrupamento de dados nos últimos anos e seu uso extensivo em ampla variedade de aplicativos, incluindo processamento de imagens, biologia computacional, comunicação móvel, medicina e economia, levou à popularidade desses algoritmos. O principal problema com os algoritmos de agrupamento de dados é que ele não pode ser padronizado. O algoritmo desenvolvido pode dar o melhor resultado com um tipo de conjunto de dados, mas pode falhar ou dar um resultado ruim com o conjunto de dados de outros tipos. Embora tenha havido muitas tentativas de padronizar os algoritmos que podem ter um bom desempenho em todos os casos de cenários, mas nenhuma grande conquista foi alcançada (JAIN, 2010).

Muitos algoritmos de agrupamento foram propostos até então, no entanto, cada algoritmo tem seus próprios méritos e deméritos e não pode funcionar para todas as situações reais. A seguir são apresentados alguns destes algoritmos:

2.2.1 K-Means

O K-Means é um dos mais simples algoritmos de aprendizado não supervisionados que resolvem o conhecido problema de agrupamento. O algoritmo foi proposto inicialmente em Steinhaus (1956), porém apenas foi melhor difundido e nomeado como K-Means em MacQueen *et al.* (1967).

O procedimento segue uma maneira simples e fácil de classificar um determinado

conjunto de dados por meio de um determinado número k de *clusters*. No K-Means, cada *cluster* é representado pelo seu centro (isto é, centróide), que corresponde à média dos pontos atribuídos ao *cluster*.

O algoritmo padrão é o proposto em Wong e Hartigan (1979), que define a variação total dentro do grupo como a soma das distâncias quadradas euclidianas entre os itens e o centroide correspondente:

$$W(C_k) = \sum_{x_i \in C_k} (x_i - \mu_k)^2 \tag{2.3}$$

onde x_i representa um ponto dentre os dados pertencentes ao grupo C_k, e μ_k é o valor médio dos pontos atribuídos ao grupo C_k.

Cada valor (x_i) é atribuído a um determinado grupo, de tal forma que a soma da distância dos quadrados (SS) do valor aos centros atribuídos do grupo μ_k seja mínimo. Define-se a variação total dentro do *cluster* como segue:

$$\sum_{k=1}^{k} W(C_k) = \sum_{k=1}^{k} \sum_{x_i \in C_k} (x_i - \mu_k)^2 \tag{2.4}$$

O K-Means procede na seguinte forma: Primeiro é escolhido k centroides iniciais, onde k é um parâmetro especificado pelo usuário, ou seja, o número de *clusters* desejados; Cada ponto é então atribuído ao centroide mais próximo, e cada coleção de pontos atribuídos a um centroide é um *cluster*; O centroide de cada *cluster* é atualizado com base nos pontos atribuídos ao *cluster*; Repete-se a atribuição e são atualizadas as etapas até que nenhum ponto altere os *clusters*, ou equivalentemente, até que os centroides permaneçam os mesmos. Formalmente, o Algoritmo 5 apresenta a versão básica do K-Means descrita anteriormente.

Algoritmo 2: K-MEANS.

Entrada: P ← Base de dados com os pontos a serem agrupados.

1 Selecione k pontos de P como centroides iniciais.

2 **repita**

3 Formar os k *clusters* atribuindo cada ponto ao seu centroide mais próximo.

4 Recalcular o centroide de cada *cluster*.

5 **até** *os centroides não mudarem*;

A complexidade do pior caso para esta versão do K-Means, onde os valores de k e d (dimensão do espaço euclidiano) são definidos, é $O(ndki)$, onde n é o número de elementos a serem agrupados e i é o número de interações até o K-Means convergir, isto é, os centroides pararem de mudar (WONG; HARTIGAN, 1979; INABA; KATOH; IMAI, 1994).

2.2.1.1 Método do cotovelo

Um grande problema quanto ao uso do K-Means é a definição do valor ideal de k para determinar os agrupamentos da melhor forma possível. Um método bastante conhecido e utilizado para auxiliar na definição do valor de k é o método do cotovelo.

O método do cotovelo (*elbow method*) é um método heurístico de interpretação e validação de consistência na análise de *cluster* projetada para ajudar a encontrar o número apropriado de *clusters* em um conjunto de dados (KETCHEN; SHOOK, 1996). O cotovelo é definido como um ponto em que indica o melhor *trade-off* de um curva de função. A ideia para esse método surgiu em Thorndike (1953).

Este método consistem em verificar a evolução dos valores k testados em detrimento a algum parâmetro escolhido em um gráfico, e verificar em que ponto a função começa a ter um comportamento próximo de ser linear. Esse ponto é chamado de "cotovelo"da função, e o valor de k corresponde a ele é o mais indicado a ser utilizado no K-Means.

Diversos tipos de parâmetros são utilizados para verificar o melhor valor de k usando o método do cotovelo. A seguir, dois desses parâmetros são apresentados:

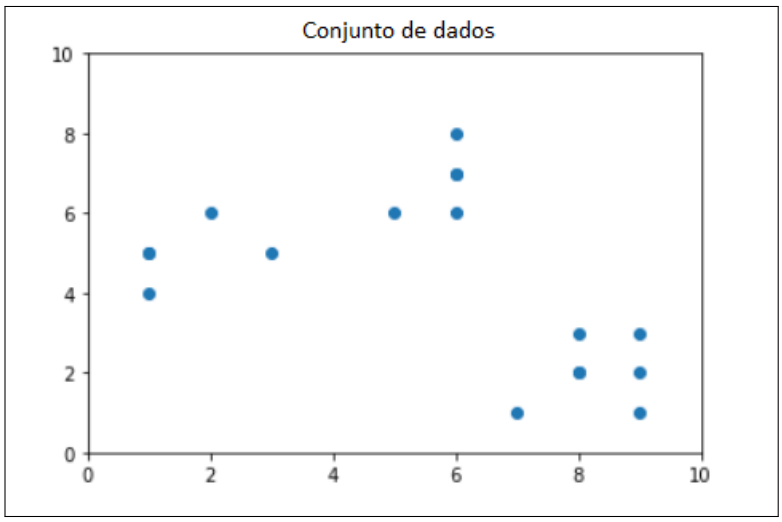
- Distorção: É calculada como a média das distâncias ao quadrado dos centros de *cluster* dos respectivos *clusters*. Normalmente, a métrica de distância euclidiana é usada;
- Inércia: É a soma das distâncias quadradas das amostras ao centro de aglomerado mais próximo.

Para exemplificar o método do cotovelo, considere a Figura 9, que apresenta um conjunto de dados onde pode ser aplicar o K-Means para identificação de agrupamentos desses dados.

Para definir o melhor k para agrupar os dados da Figura 9, são utilizados gráficos que mostram as variações dos parâmetros de distorção e inércia, considerando valores de $k = 1, 2, 3, ..., 9$. As Figuras 10 – 11 apresentam os gráficos para análise do método do cotovelo, para os respectivos parâmetros.

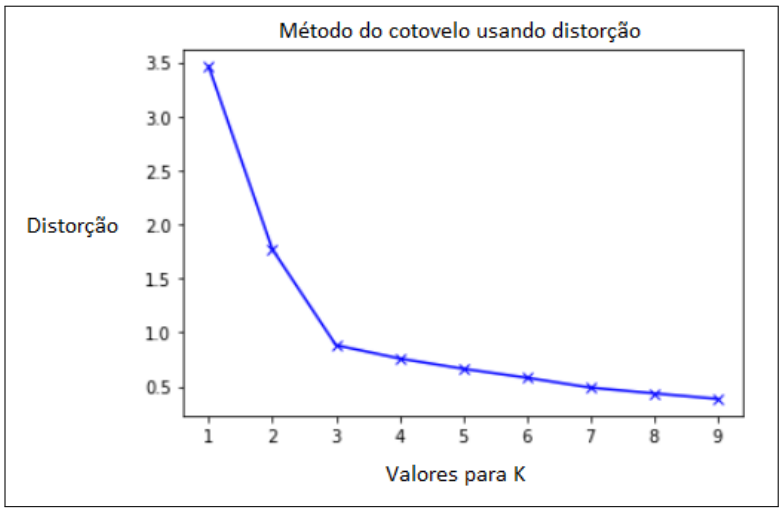
Em ambas as figuras, é perceptível que em $k = 3$ o comportamento da função tornasse

Figura 9 – Exemplo de conjunto de dados para agrupar.



Fonte: Gupta (2019).

Figura 10 – Gráfico para análise do método do cotovelo usando o parâmetro "distorção".



Fonte: Gupta (2019).

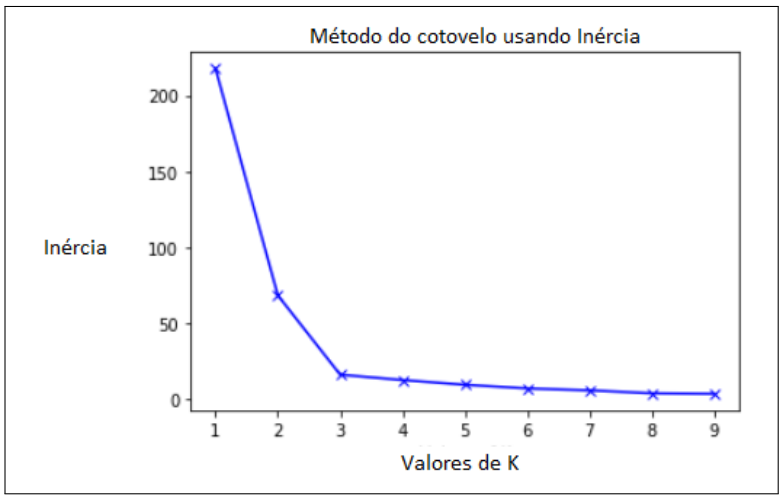
próximo de linear. Assim, este valor de k é o que seria escolhido para aplicar o K-Means no conjunto de dados da Figura 9.

2.2.1.2 Método de detecção de joelhos

A grande desvantagem do uso do método do cotovelo, é que ele resume-se em uma análise visual de um gráfico, e a decisão da escolha do melhor valor de k para o K-Means é manual.

Uma forma de tornar esse processo automático é utilizando o método de detecção de

Figura 11 – Gráfico para análise do método do cotovelo usando o parâmetro "inercia".



Fonte: Gupta (2019).

joelhos (*Knee detection*), proposto por Satopaa *et al.* (2011). Joelho é outra denominação para o que é definido como cotovelo, que é o ponto de uma determinada função que indica o melhor *trade-off* em uma curva.

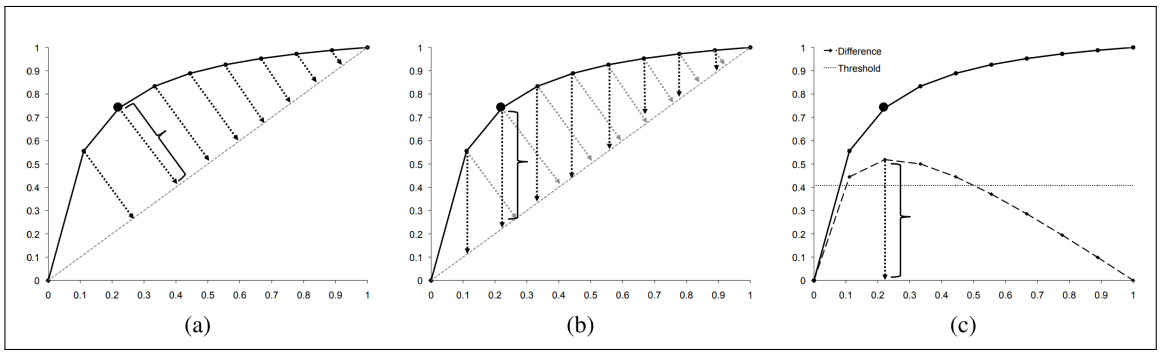
Para qualquer função contínua f existe uma forma fechada padrão $K_f(x)$ que define a curvatura de f em qualquer ponto como uma função de sua primeira e segunda derivada, definida pela equação a seguir:

$$K_f(x) = \frac{f''(x)}{(1+f'(x)^2)^{1.5}} \tag{2.5}$$

Os autores de Satopaa *et al.* (2011) propuseram um algoritmo que permite calcular o ponto de cotovelo ideal de acordo com a equação acima. Este algoritmo é chamado de *Kneedle algorithm*, fazendo alusão a expressão "procurando uma agulha no palheiro". Os joelhos ou cotovelos ocorrem quando a curva da função se torna mais "plana"indicando um decrescimento na curvatura.

A Figura 12 apresenta um exemplo de aplicação do *Kneedle algorithm*. A parte (a) da figura descreve os dados suavizados e normalizados, com barras tracejadas indicando a distância perpendicular de $y = x$ com a distância máxima indicada. A parte (b) mostra os mesmos dados, mas desta vez as barras tracejadas são giradas 45^o. A magnitude dessas barras corresponde aos valores de diferença usados no algoritmo. E, a parte (c) mostra o gráfico desses valores de diferença e os valores limite correspondentes (com $S = 1$). O joelho é encontrado em $x = 0.22$ e é detectado após receber o ponto $x = 0.55$.

Figura 12 – Aplicação do *Kneedle algorithm* para detecção de joelho/cotovelo em curva.



Fonte: Satopaa *et al.* (2011).

Primeiro, é aplicado um método de *smoothing spline* suavizar a curva formada pelo pontos a serem analisados. Esse método gera uma versão estimada da função formada pelos pontos de forma que ela tenha uma curva mais "suave".

A função estimada $\hat{f}(x)$ é obtida a partir do conjunto de observações y_i do alvo $f(x_i)$, a fim de equilibrar uma medida de qualidade de ajuste de $\hat{f}(x_i)$ para y_i com uma medida derivada da suavidade de $\hat{f}(x)$.

Seja $\{x_i, Y_i : i = 1, ..., n\}$ o conjunto de observações da função, modeladas pela expressão $Y_i = f(x_i) + \varepsilon_i$, onde ε_i representa variáveis independentes com variação aleatória media zero (Normalmente assume-se que tenha variância constante). A versão suavizada de $f(x_i)$ é dada por (GREEN; SILVERMAN, 1993):

$$\hat{f}(x) = \sum_{i=1}^{n} \{Y_i - \hat{f}(x_i)\}^2 + \lambda \int \hat{f}''(x)^2 \, dx. \tag{2.6}$$

onde $\lambda \geq 0$ é o parâmetro de suavização, controlar o *trade-off* entre fidelidade aos dados e irregularidade da estimativa da função.

Seja D_s o conjunto finito de valores x e y que definem uma curva suave, isto é, uma que foi ajustada pelo *smoothing spline*, onde:

$$D_s = \{(x_{s_i}, y_{s_i}) \in \mathbb{R} | x_{s_i}, y_{s_i} \geq 0\} \tag{2.7}$$

A seguir, os pontos em D_s são normalizados da seguinte forma:

$$D_{sn} = \{(x_{sn}, y_{sn})\} \tag{2.8}$$

onde:

$$x_{sn_i} = (x_{s_i} - min\{x_s\})/(max\{x_s\} - min\{x_s\}) \tag{2.9}$$

$$y_{sn_i} = (y_{s_i} - min\{y_s\})/(max\{y_s\} - min\{y_s\}) \tag{2.10}$$

Seja D_d o conjunto de diferenças entre os valores de x e y, isto é, os o conjunto de pontos $(x, y-x)$ ilustrado na Figura 12(b). O objetivo é descobrir quando a curva de diferença muda de horizontal para diminuindo acentuadamente, pois isso indica a presença de um joelho no conjunto de dados original. Os valores reais dos pontos de diferença são irrelevantes, o que importa é apenas observar as tendências da curva da diferença, como mostra a Figura 12(c). Assim D_d é definido:

$$D_d = \{(x_{d_i}, y_{d_i}))\} \tag{2.11}$$

onde:

$$x_{d_i} = x_{sn_i} \tag{2.12}$$

$$y_{d_i} = y_{sn_i} - x_{sn_i} \tag{2.13}$$

Para encontrar os pontos do joelho na curva normalizada, por exemplo, os lugares onde a curva se achata, é calculado os máximos locais da curva da diferença. Esses pontos indicam os casos em que a taxa de aumento de y começa a diminuir. Cada um desses pontos máximos locais, pertencentes ao conjunto D_{lmx}, é um ponto de joelho candidato na curva de dados original:

$$D_{lmx} = \{(x_{lmx_i}, y_{lmx_i}))\} \tag{2.14}$$

onde:

$$x_{d_i} = x_{d_i} \tag{2.15}$$

$$y_{d_i} = y_{d_i} | y_{d_{i-1}} < y_{d_i}, y_{d_{i+1}} < y_{d_i} \tag{2.16}$$

Para cada máximo local (x_{lmx_i}, y_{lmx_i}) na curva da diferença, é definido um valor limite exclusivo, T_{lmx_i}, que se baseia na diferença média entre valores x consecutivos e um parâmetro de sensibilidade, S. O parâmetro de sensibilidade permite a agressividade deseja que o *Kneedle algorithm* tenha ao detectar joelhos. Valores menores para S detectam joelhos mais rapidamente, enquanto valores maiores são mais conservadores. Simplificando, S é uma medida de quantos pontos "planos" é esperado ver na curva de dados não modificados antes de declarar como um joelho. Na Figura 12(c), a linha do limiar é plotada com $S = 1$.

$$T_{lmx_i} = y_{lmx_i} - S\frac{\sum_{i=1}^{n-1}(x_{sn_{i+1}} - x_{sn_i})}{n-1} \tag{2.17}$$

Se qualquer valor de diferença (x_{d_j}, y_{d_j}), em que $j > i$, cai abaixo do limite $y = T_{lmx_i}$ para (x_{lmx_i}, y_{lmx_i}) antes que o próximo máximo local na curva de diferença seja atingido, o *Kneedle algorithm* declara um joelho no valor x de máximo local correspondente $x = x_{lmx_i}$. Se os valores da diferença atingirem um mínimo local e começarem a aumentar antes que $y = T_{lmx_i}$ seja atingido, é redefinido o valor limite para 0 e aguarda-se até que outro máximo local seja atingido.

O *Kneedle algorithm* é resumido no Algoritmo 3.

Algoritmo 3: *Kneedle algorithm*.

Entrada: X ← lista de valores dos pontos no eixo X; Y ← lista de valores dos pontos no eixo Y; S ← valor da sensibilidade (normalmente definido como $S = 1$).

1 Suavizar a curva formada pelos pontos (X_0, Y_0), (X_1, Y_1), ...,(X_n, Y_n) e armazenar em D_s.
2 Normalizar os pontos em D_s para D_{sn} de acordo com as equações (2.7) – (2.9).
3 Encontrar o conjunto de diferenças de D_{sn} e armazenar em D_d de acordo com as equações (2.10) – (2.12).
4 Calcular os máximos locais de D_d e armazenar em D_{lmx} de acordo com as equações (2.13) – (2.15).
5 Calcular o valor de T_{lmx} para cada ponto em D_{lmx} de acordo com a equação (2.16).
6 **retorna** *O valor x declarado como máximo local correspondente a* $x = x_{lmx_i}$.

A complexidade do *Kneedle algorithm* é definida pela etapa de suavização da curva,

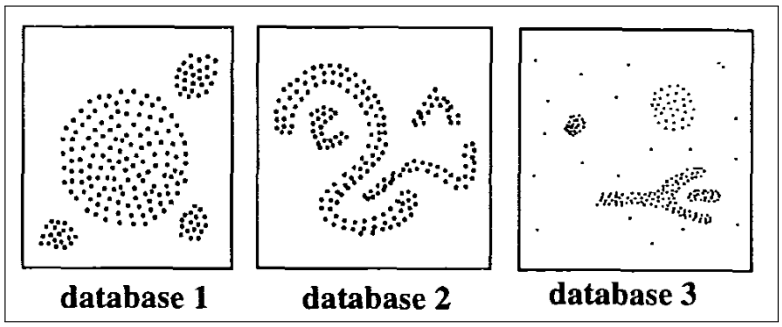
feito pelo método de *smoothing spline*, que tem complexidade $O(n^3)$, onde n é número de pontos a serem ajustados (MA; HUANG; ZHANG, 2015). As outras etapas, como não vão além de somatórios ou consultas de complexidade $O(n)$, são desprezíveis.

2.2.2 DBSCAN

Density-Based Spatial Clustering and Application with Noise ou DBSCAN é um algoritmo de agrupamento baseado em densidade, proposto por Ester *et al.* (1996), que pode ser usado para identificar grupos de qualquer formato em um conjunto de dados contendo ruído e valores discrepantes.

A ideia básica por trás da abordagem de agrupamento baseada em densidade é derivada de um método de agrupamento intuitivo humano. Por exemplo, observando a Figura 13, pode-se identificar facilmente quatro grupos junto com vários pontos de ruído, devido às diferenças na densidade de pontos.

Figura 13 – Construindo quatro grupos a partir de dados não rotulados.



Fonte: Ester *et al.* (1996)

Os grupos são regiões densas no espaço de dados, separadas por regiões de menor densidade de pontos. O algoritmo DBSCAN baseia-se nesta noção intuitiva de grupos e ruido. A ideia é que, para cada ponto de um grupo, a vizinhança de um determinado raio deve conter pelo menos um número mínimo de pontos.

O objetivo é identificar regiões densas, que podem ser medidas pelo número de objetos próximos a um determinado ponto. Dois parâmetros importantes são necessários para o DBSCAN: uma constante r e pontos mínimos $MinPts$. O parâmetro r define o raio da vizinhança em torno de um ponto p. É chamado de "vizinhança de p". O parâmetro $MinPts$ é o número mínimo de vizinhos dentro do raio r.

Qualquer ponto p no conjunto de dados, com uma contagem de vizinhos maior ou

igual a *MinPts*, é marcado como um ponto central. Dizemos que p é o ponto de fronteira, se o número de seus vizinhos for menor que *MinPts*, mas pertence à vizinhança de algum ponto central q. Finalmente, se um ponto não é nem um núcleo nem um ponto de fronteira, então ele é chamado de ponto de ruído.

Formalmente o DBSCAN pode ser descrito nos passos do Algoritmo 4.

Algoritmo 4: DBSCAN

```
Entrada: B ← Base de dados com os pontos a serem agrupados; r ← raio da
         vizinhança; MinPts ← Número mínimo de pontos para definir um cluster
para cada ponto p não visitado dentro da base de dados B faça
    Marcar p como visitado.
    Verificar os pontos com distancia máxima r de p.
    se a quantidade pontos encontrada for menor que MinPts então
        Marque p como ruído.
    fim
    senão
        Criar um novo cluster C e adicione p a ele.
        para cada ponto q com distancia máxima r de p faça
            se q é um ponto não visitado ainda então
                Marcar q como visitado.
                Verificar os pontos com distancia máxima r de q.
                se a quantidade pontos encontrada for maior ou igual a MinPts então
                    Adicionar os pontos encontrados a mesma lista dos pontos com
                     distancia máxima r de p.
                fim
            fim
            se q não é membro de nenhum cluster ainda então
                Adicione q ao cluster C.
            fim
        fim
    fim
fim
```

A complexidade do DBSCAN é $O(n^2)$ no pior caso, onde n é a quantidade de pontos a serem agrupados.

2.2.3 *Trajectory clustering algorithm*

O *trajectory clustering algorithm* é um algoritmo proposto por Zhao *et al.* (2015), para identificação de agrupamentos de alvos móveis. Os autores prepuseram esse algoritmo para identificar aglomerados de pessoas através do estudo de mapas de calor de uma cidade ao longo do dia, para que os centros identificados desses grupos se tornem *hotspots* (pontos estratégicos)

para posicionamento de táxis.

Suponha $P = \{p_1, p_2, ..., p_n\}$ é um conjunto contendo os n pontos de trajetória possíveis para posicionar um táxi, cada ponto é considerado uma partícula com massa e existe um campo virtual em torno dele. Qualquer ponto de trajetória nesse campo receberá interação mútua de outros pontos. Desse modo, um campo de dados de trajetória se forma nesse espaço de trajetória. O valor potencial de p_i (ϕp_i) é representado como:

$$\phi(p_i) = \sum_{j=1}^{n} (m_j e^{-(\frac{d_{ij}}{\sigma})^k}) \tag{2.18}$$

onde m_j é massa do ponto de trajetória p_j (sendo $j = 1, ..., n$); d_{ij} a distancia entre os pontos p_i e p_j; $\sigma \in (0, +\infty)$ é o alcance de interação entre os pontos; e o $k \in N$ é o índice de distancia. No trabalho original, os autores fixaram o valor de k igual a 2.

Para encontrar o valor σ, é necessário atribuir vários valores a ele para que se possa calcular o valor ótimo, que é obtido quando a entropia potencial atinge o mínimo (LI; DU, 2007). A entropia pode ser calculada pela seguinte equação (Versão adaptada por Iellamo, Lehtomaki e Khan (2017) de Shannon (1948)):

$$H = -\sum_{i=1}^{n} \frac{\phi(p_i)}{Z} \log(\frac{\phi(p_i)}{Z}) \tag{2.19}$$

onde $Z = \sum_{i=1}^{n} \phi(p_i)$ e $0 \leq H \leq \log(n)$.

O grafo de decisão que determina o posicionamento dos centros dos *clusters* precisa de duas informações: o valor da densidade local ρ_i e da distancia local δ_i dos pontos de maior densidade. Os pontos com ambos maiores ρ_i e δ_i são considerados os centros dos *clusters*. ρ_i é calculado pela equação:

$$\rho_i = \sum_{j} X(d_{ij} - d_c) \tag{2.20}$$

onde $X(x) = 1$, se $x < 0$, caso contrário $X(x) = 0$; d_c é a distancia de corte, ou seja, maior distancia considerada para que dois pontos estejam no mesmo *cluster*; d_{ij} é a distancia entre os pontos p_i e p_j.

Já o δ_i é a distancia mínima entre p_i de qualquer outro ponto com densidade superior a ele. δ_i é calculado pela equação:

$$\delta_i = min_{j:\rho_j > \rho_i}(d_{ij}) \tag{2.21}$$

O *trajectory clustering algorithm* é descrito nos passos a apresentados no Algoritmo 5.

Algoritmo 5: *Trajectory clustering algorithm*

Entrada: P ← Base de dados com os pontos de trajetória.
Selecione aleatoriamente vários valores para σ.
para *cada* σ **faça**
    Calcular o valor potencial correspondente de acordo com a equação (2.18).
fim
Calcule o valor ideal para o σ do fator de impacto.
para *cada ponto de trajetória* **faça**
    Calcule o valor potencial com a equação (2.18), baseado no valor ótimo de σ encontrado. O valor da massa de cada ponto é fixado em 1.
fim
Selecione os centros dos *clusters* (Pontos com valores potenciais maiores que a média + desvio padrão).
Identifique os pontos de ruído. Como os pontos de ruído geralmente se espalham no campo de dados e recebem uma interação mútua fraca, eles têm valores de potencial menores que a média + devio padrão.
para *cada ponto em P* **faça**
    Atribuir ao mesmo *cluster* o vizinho dele mais próximo com o valor potencial mais alto.
fim

A complexidade do *trajectory clustering algorithm* é $O(n^2)$, considerando n a quantidade de pontos de tragetória dados como entrada.

2.3 EFICIÊNCIA DE PARETO

A eficiência de Pareto, ou ótimo de Pareto, é uma noção importante na economia neoclássica, com amplas aplicações na teoria dos jogos, engenharia e ciências sociais. Dado um conjunto de alocações alternativas e um conjunto de indivíduos, um movimento de uma alocação para outra que pode melhorar pelo menos um indivíduo, sem prejudicar qualquer outro indivíduo, é chamado de melhoria de Pareto ou otimização de Pareto. Uma alocação de recursos é Pareto eficiente ou Pareto ideal quando nenhuma melhoria de Pareto adicional puder ser feita (BARR,

2012).

O termo tem o nome de Vilfredo Pareto, economista italiano que usou o conceito em seus estudos sobre eficiência econômica e distribuição de renda (PARETO, 1964).

O conjunto de todas as alocações eficientes de Pareto é chamado frente de Pareto, fronteira de Pareto ou conjunto de Pareto. A Figura 14 apresenta um exemplo de um conjunto de soluções, onde as destacadas em vermelho são as soluções Pareto eficientes, e as cinzas são aquelas Pareto não eficientes.

Figura 14 – Exemplo de frentes de Pareto.

Fonte: Autor desconhecido.

A eficiência de Pareto é importante porque fornece um norma fraca, mas amplamente aceita, para comparar resultados econômicos. É considerado uma norma fraca, porque pode haver muitas situações eficientes e o teste de Pareto não determina qual solução escolher.

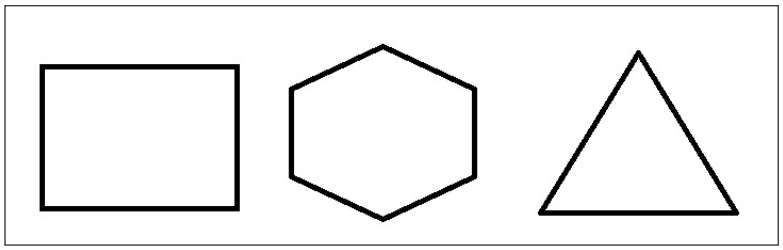
Para exemplificar a eficiência de Pareto, considere: Em uma venda de um carro usado, o vendedor está anunciando um carro que avaliado em R$ 30,000, enquanto o comprador está disposto a pagar até R$ 45,000 por este carro. Um acordo em que o carro seja vendido por R$ 37.500 seria Pareto eficiente, porque o vendedor e o comprador estão em melhor situação de troca justa no comércio. Nesse caso, o vendedor estaria lucrando R$ 7,500, e o comprador estaria economizando R$ 7,500 em relação ao valor que ele pretendia gastar. No entanto, qualquer

preço entre R$ 30,000 e R$ 45,000 é Pareto eficiente, porque o vendedor não teria prejuízo na venda do veículo, e o comprador não pagaria mais do que ele pretendia pelo automóvel.

2.4 POLÍGONOS

Um polígono é qualquer forma bidimensional formada por linhas retas. Triângulos, quadriláteros, pentágonos e hexágonos são exemplos de polígonos. O nome informa quantos lados a forma possui. Por exemplo, um triângulo tem três lados e um quadrilátero tem quatro lados. Portanto, qualquer forma que possa ser desenhada conectando três linhas retas é chamada de triângulo, e qualquer forma que possa ser desenhada conectando quatro linhas retas é denominada quadrilateral (PINHO; BATISTA; CARVALHO, 2010). A Figura 15 apresenta alguns exemplos de polígonos.

Figura 15 – Exemplos de polígonos.



Fonte: Elaborada pelo autor.

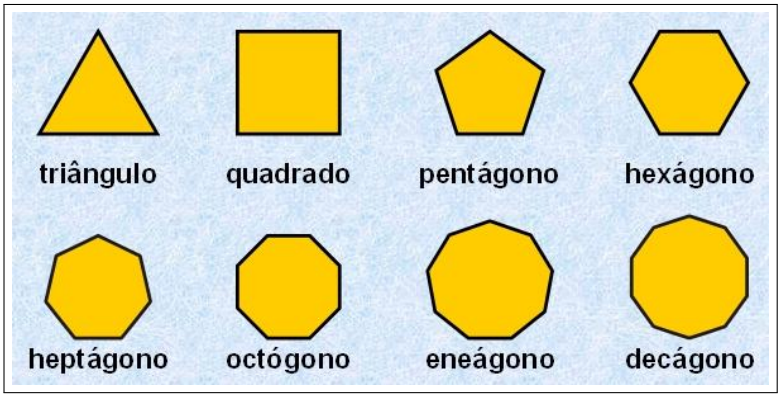
Os segmentos de um circuito poligonal são chamados de arestas ou lados, e os pontos onde as duas arestas se encontram são os vértices do polígono.

Os polígonos são classificados de forma primária através do número de lados que eles possuem. A denominação de um polígono para um valor desconhecido de lados n é de n-gon. Alguns dessas classificações estão ilustradas pela Figura 16.

Os polígonos também possuem outros tipos de classificação, como mostrado abaixo:

- Regular: Um polígono com todos os lados e ângulos internos iguais;
- Irregular: Cada lado pode ter um comprimento diferente, cada ângulo pode ser uma medida diferente. É o oposto de um polígono regular;
- Convexo: Todos os ângulos internos são inferiores a 180^{o} e todos os vértices apontam para fora do interior. Polígonos regulares são sempre convexos;
- Concavo: Um ou mais ângulos interiores superiores a 180^{o}. Alguns vértices empurram 'para dentro' em direção ao interior do polígono. É o oposto de convexo;

Figura 16 – Algumas denominações de polígonos de acordo com o número de lados.

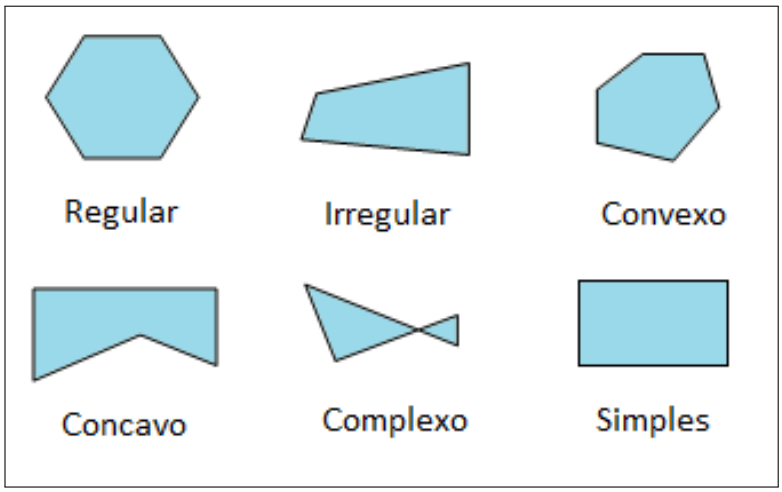


Fonte: Melo (2014).

- Complexo: Também conhecido como polígono com auto-interseção ou cruzado. É um polígono em que um ou mais lados cruzam de volta sobre o outro lado, criando vários polígonos menores. A maioria das propriedades e teoremas relativos a polígonos não se aplica a essa forma;
- Simples: É oposto do complexo, onde é um polígono sem auto-interseção.

A Figura 17 ilustra os tipos de polígonos descritos acima.

Figura 17 – Alguns exemplos de diferentes tipos de polígonos.



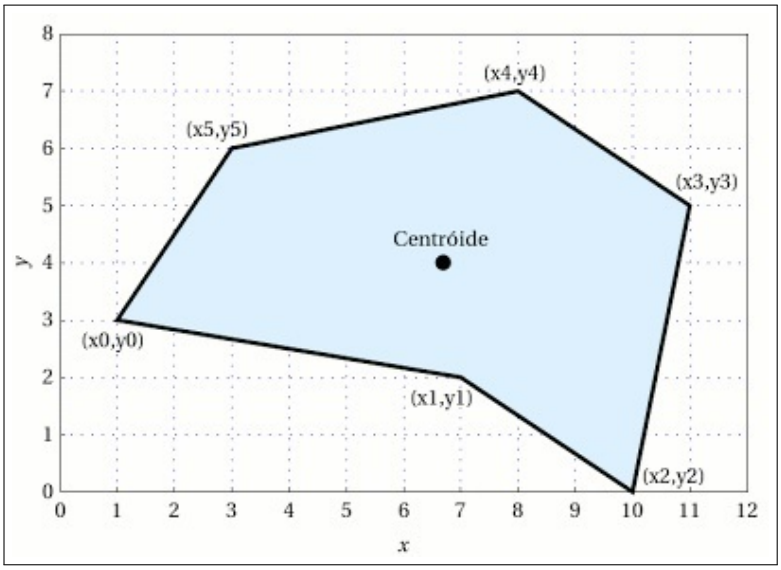
Fonte: Elaborada pelo autor.

2.4.1 Centroide de um polígono

O centroide ou centro geométrico de uma figura plana é a posição que corresponde a média aritmética de todos os pontos da figura. O termo baricentro é comumente utilizado como sinônimo de centroide de um polígono. A Figura 18 apresenta um exemplo de centroide de um

polígono.

Figura 18 – Exemplo de centroide de um polígono.



Fonte: MADEIRA (2009).

O centroide de um polígono, definido pelo ponto (C_x,C_y), pode ser calculado através das equações (BOURKE, 1988):

$$C_x = \frac{1}{6A}\sum_{i=0}^{n-1}(x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \tag{2.22}$$

$$C_y = \frac{1}{6A}\sum_{i=0}^{n-1}(y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \tag{2.23}$$

onde A é área do polígono, descrita pela fórmula de Shoelace (BRADEN, 1986), dada pela equação:

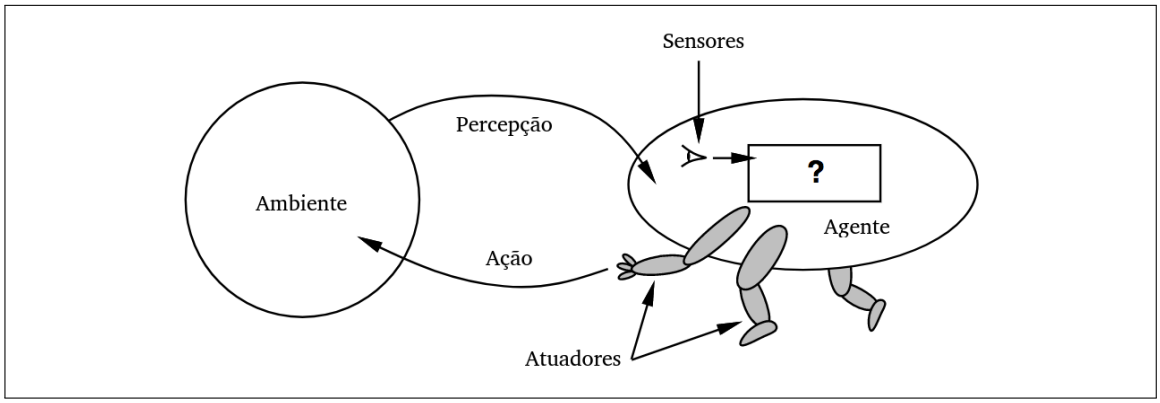
$$A = \frac{1}{2}\sum_{i=0}^{n-1}(x_i y_{i+1} - x_{i+1} y_i) \tag{2.24}$$

2.5 AGENTES INTELIGENTES

Um agente inteligente é um programa que pode tomar decisões ou executar um serviço com base em seu ambiente, entrada do usuário e experiências. Esses programas realizam uma ação com o melhor resultado depois de considerar as percepções passadas e atuais (entradas perceptivas do agente em uma determinada instância) (RUSSELL; NORVIG, 2016).

Um sistema de inteligência artificial é composto por um agente e seu ambiente. Os agentes atuam em seu ambiente. O ambiente pode conter outros agentes. Um agente é qualquer coisa que possa ser vista como algo que percebe seu ambiente através de sensores e age sobre esse ambiente através de atuadores (RUSSELL; NORVIG, 2016). A Figura 19 representa a estrutura padrão de um agente.

Figura 19 – Estrutura de um agente.



Fonte: Basa (2018)

Um agente é uma composição de uma arquitetura e o programa do agente, onde arquitetura é a máquina na qual o agente executa. É um dispositivo com sensores e atuadores, por exemplo: um carro robótico, uma câmera, um computador. O programa do agente é uma implementação de uma função do agente. Uma função de agente é um mapa da sequência de percepção (histórico de tudo que um agente percebeu até a data) para uma ação. A seguir tem-se alguns exemplos de agentes:

- Um agente de software possui entradas de teclado, informações de arquivo, pacotes de rede recebidos que atuam como sensores e *displays* na tela, arquivos, pacotes de rede enviados atuando como atuadores.
- Um agente humano possui olhos, ouvidos e outros órgãos que atuam como sensores e mãos, pernas, boca e outras partes do corpo atuando como atuadores.
- Um agente robótico possui câmeras e rastreadores infravermelhos que atuam como sensores e vários motores atuando como atuadores.

Existem vários tipos de agentes que variam de acordo com o alcance de suas capacidades e níveis de inteligência. São alguns deles:

- Agentes reflexivos simples: Esses agentes funcionam em um estado atual, ignorando a história passada. As respostas baseiam-se na regra de ação-condição-evento (regra ECA)

na qual um usuário inicia um evento e o agente se refere a uma lista de regras predefinidas e resultados pré-programados.

- Agentes reflexivos baseados em modelos: esses agentes escolhem uma ação da mesma forma que um agente reflexivo, mas eles têm uma visão mais abrangente do ambiente. Um modelo do mundo é programado no sistema interno que incorpora a história do agente.
- Agentes reflexivos baseados em metas: Esses agentes expandem o armazenamento de informações de agentes baseados em modelo, incluindo também informações sobre metas ou informações sobre situações desejáveis.
- Agentes reflexivos baseados em utilitários: Esses agentes são semelhantes aos agentes baseados em metas, mas fornecem uma medida de utilidade extra que classifica cada cenário possível em seu resultado desejado e escolhe a ação que maximiza o resultado. Exemplos de critérios de classificação podem ser a probabilidade de sucesso ou os recursos necessários.
- Agentes de aprendizagem: Esses agentes têm a capacidade de melhorar gradualmente e tornar-se mais informados sobre um ambiente ao longo do tempo por meio de um elemento adicional de aprendizado. O elemento de aprendizagem usará *feedback* para determinar como os elementos de desempenho devem ser alterados para melhorar gradualmente.

2.6 VEÍCULOS AÉREOS NÃO TRIPULADOS

Um veículo aéreo não tripulado (Unmanned Aerial Vehicle ou UAV) é uma aeronave que não transporta nenhum piloto humano ou passageiros. Os UAVs - às vezes chamados de *drones* - podem ser total ou parcialmente autônomos, mas são mais frequentemente controlados remotamente por um ser humano. Podem ser descartáveis ou recuperáveis, e podem transportar uma grande variedade de cargas, dependendo dos tipos, funcionalidade, características operacionais e objetivos da missão (VALAVANIS; VACHTSEVANOS, 2015).

Antigamente, os UAVs eram mais frequentemente associados aos militares, onde inicialmente eram utilizados para recolhimento de informações e, de forma controversa, como plataformas de armas. Os drones agora também são usados em uma ampla gama de funções civis, desde busca e salvamento, vigilância, monitoramento de tráfego, monitoramento meteorológico e combate a incêndios, até drones pessoais e fotografia comercial baseada em drones, bem como videografia, agricultura e até mesmo serviços de entrega (VALAVANIS; VACHTSEVANOS, 2015). A Figura 20 exemplifica visualmente um drone pessoal para fotografia.

Figura 20 – UAV para captura de imagens.

Fonte: McGoldrick, Shivaram e Huggard (2016)

2.6.1 Tipos de UAVs

As plataformas de drones têm dois tipos principais: rotor, incluindo rotor único ou multi-rotor (como tricopters, quadcopters, hexacopters e octocoptors), ou de asa fixa, que incluem os drones VTOL (*Vertical Take-Off and Landing*) que não exigem pistas, já que podem decolar e aterrizar verticalmente. Os drones podem ser categorizados como de uso pessoal ou comercial (VALAVANIS; VACHTSEVANOS, 2015; OTTO *et al.*, 2018).

Os drones podem ser equipados com vários sensores, incluindo sensores de distância, sensores de tempo de voo, sensores químicos e sensores de estabilização e orientação, entre outros. Os sensores visuais oferecem dados estáticos ou de vídeo, com sensores RGB coletando comprimentos de onda visuais vermelhos, verdes e azuis padrão e sensores multi espectrais coletando comprimentos de onda visíveis e não visíveis, como infravermelho e ultravioleta. Acelerômetros, giroscópios, magnetômetros, barômetros e GPS também são componentes comuns dos drones (VALAVANIS; VACHTSEVANOS, 2015; OTTO *et al.*, 2018).

Por exemplo, sensores térmicos podem ser integrados em aplicações de vigilância ou segurança, como monitoramento de gado ou detecção de assinatura de calor. Os sensores hiper espectrais podem ajudar a identificar minerais e vegetação e são ideais para uso na saúde da cultura, qualidade da água e composição da superfície.

Muitos drones pessoais já estão disponíveis para uso do consumidor, oferecendo recursos de câmera fotográfica ou de captura de vídeo em alta definição, ou simplesmente

voando de forma aleatória. Esses drones geralmente pesam de menos de meio quilo a 10 quilos (VALAVANIS; VACHTSEVANOS, 2015; OTTO *et al.*, 2018).

Drones mais fortes e mais capazes também estão disponíveis para uso em ambientes comerciais. Por exemplo, a Insitu[1], uma empresa da Boeing, oferece o drone ScanEagle (Figura 21), que tem uma envergadura de três metros e pesa 35 quilogramas, e é usado para reconhecimento, geralmente de pontos de pesca em alto mar. A empresa também constrói o Integrator (Figura 22), uma aeronave que pesa cerca de 37 quilogramas e possui uma envergadura de seis metros, utilizado de forma similar ao ScanEagle. Esses drones possuem alguns dispositivos interessantes inclusos tais como sensores eletro-óticos, imageadores infravermelhos de onda média, marcadores infravermelhos e telêmetros a laser (VALAVANIS; VACHTSEVANOS, 2015; OTTO *et al.*, 2018).

Figura 21 – Boeing Insitu ScanEagle.

Fonte: Insitu (2018)

2.6.2 Exemplos de aplicações de UAVs

O uso de drones fora do exército cresceu tremendamente na última década. Além de aplicações de vigilância e entrega, os UAVs são usados em jornalismo, busca e salvamento, resposta a desastres, proteção de ativos, monitoramento da vida selvagem, combate a incêndios, retransmissão de comunicação, saúde e agricultura (VALAVANIS; VACHTSEVANOS, 2015; OTTO *et al.*, 2018).

[1] https://www.insitu.com/

Figura 22 – Boeing Insitu Integrator.

Fonte: Insitu (2018)

A integração da tecnologia de drones e internet das coisas (Internet of Things ou IoT) criou vários casos de uso corporativo. Os drones que trabalham com redes de sensores IoT no solo podem ajudar as empresas agrícolas a monitorar terras e plantações; as empresas de energia a pesquisarem linhas de energia e verificar se os equipamentos estão operando normalmente ou não; e as empresas de seguro a monitorar propriedades para possíveis reivindicações (VALAVANIS; VACHTSEVANOS, 2015; OTTO *et al.*, 2018).

3 TRABALHOS RELACIONADOS

Neste capítulo são apresentados alguns trabalhos relacionados com a este trabalho. São pesquisas que inspiraram e contribuiram para o desenvolvimento deste trabalho, apresentando diferentes estratégias para posicionamento de UAVs.

3.1 OPTIMIZATION OF MOBILE SENSOR COVERAGE WITH UAVS

Quanto ao problema de implantação de drones autônomos tem-se o trabalho de Caillouet, Giroire e Razafindralambo (2018). Eles consideram a minimização do custo de implantação e das altitudes dos UAVs para garantir uma boa qualidade de comunicação. Eles também consideraram a conectividade entre os drones e uma estação base para coletar e enviar informações aos alvos.

Nesse trabalho foi constatado que minimização do custo e da altura são objetivos antagônicos, pois ao reduzir a altitude dos drones também há a redução da área de cobertura de cada um deles, logo é necessário mais unidades autônomas, o que consequentemente iria aumentar o custo de implantação. Eles propuseram uma solução ótima de *trade-off* justo para este problema e também avaliaram o custo de adicionar conectividade à implantação de UAVs.

O objetivo é implantar UAVs e escolher suas respectivas posições e altitudes no conjunto de posições possíveis de forma que todos alvos móveis sejam associados a pelo menos um drone; e que todos os drones consigam se comunicar entre si e com a estação base. Foi associado um custo para implantação do drone em uma determinada posição, que representa os gastos de energia e despesas de monitoramento.

Antes de apresentar o modelo de otimização de Caillouet, Giroire e Razafindralambo (2018), seguem algumas definições:

1. P é o conjunto de posições disponíveis para alocação de drones;
2. U é o conjunto de UAVs disponíveis;
3. N é o conjunto de alvos a serem monitorados;
4. h_u é a altitude do drone u, $u \in U$;
5. r_u^h é o raio de observação do drone u, $u \in U$, em função da respectiva altura h_u;
6. c_u é o custo de implantação do drone u, $u \in U$;
7. z_u^p é uma variável binária que determina se o drone u, $u \in U$, foi alocado para a posição p, $p \in P$;
8. X_n^u é uma variável binária que determina se o drone u, $u \in U$, cobre o alvo n, $n \in N$;

9. b representa a estação base.

O modelo de otimização linear é o que segue:

$$min(\sum_{p\in P}\sum_{u\in U} c_u + \max_{p\in P,u\in U} h_u(p) * z_p^u) \tag{3.1}$$

onde:

$$\sum_{p\in P} z_u^p \leq 1, \forall u \in U \tag{3.2}$$

$$\sum_{u\in U} X_n^u \geq 1, \forall n \in N \tag{3.3}$$

$$X_n^u \leq \sum_{p\in P} z_u^p * r_u^h / d_{un}, \forall u \in U, n \in N \tag{3.4}$$

$$z_u^p, X_n^u \in \{0,1\} \tag{3.5}$$

Para garantir a conectividade entre os drones, tem-se as seguintes restrições:

$$\sum_{v\in U, v\neq u} f_{uv} - \sum_{v\in U, v\neq u} f_{vu} = \begin{cases} \sum_{v\in U}\sum_{v\in U} P^{z_p^u}, u = b \\ -\sum P^{z_p^u}, u \neq b \end{cases} \tag{3.6}$$

$$f_{uv} \leq \sum_{p\in P} z_p^u * R_u / D_{uv} * |U|, \forall u, v \in U \tag{3.7}$$

$$f_{uv} \leq \sum_{p\in P} z_p^v * R_u / D_{uv} * |U|, \forall u, v \in U \tag{3.8}$$

$$f_{uv} \in R \tag{3.9}$$

Assim, o programa multiobjetivo é definido nas seguintes equações:

$$min \sum_{p \in P} \sum_{u \in U} c_u * z_p^u \tag{3.10}$$

$$min \max_{u \in U, p \in P} h_u(p) * z_p^u \tag{3.11}$$

$$s.t. (3.2) - (3.9) \tag{3.12}$$

Para o experimentos, as instancias de teste foram definidas em um região quadrada $100 * 100$, onde os alvos foram distribuídos em coordenadas randômicas na superfície. Para cada coordenada (x, y) no mapa foram atribuídas três alturas possíveis para os drones, sendo 10 metros, 25 metros e 40 metros, respectivamente. A estação base foi posicionada na posição (0.0, 0.0, 0.0). As UAVs possuem ângulo de visualização de 60º e alcance de comunicação de 30 metros.

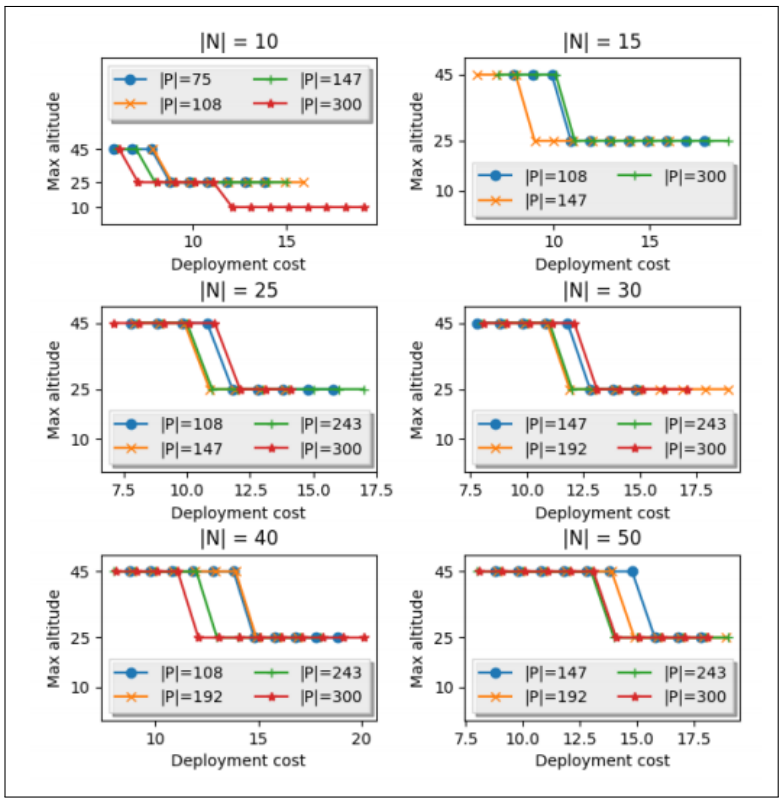
Para encontrar a solução foi utilizado o conceito de frente de Pareto, a partir da análise do trade-off entre custo de implantação e altitude, para obter uma solução ótima justa. A fim de gerar soluções ótimas de Pareto do problema cobertura de alvos com conectividade, os autores usaram o método $\varepsilon - constraint$ que transforma o problema biobjetivo em uma sequencia de problemas parametrizados de objetivo único, de tal forma que o de cada problema de objetivo único corresponde a uma solução ótima de Pareto. Assim, eles formularam e resolveram os problemas de otimização monoobjetiva da seguinte maneira:

$$\begin{cases} min f^i(x) \\ s.t. f^j(x) \leq \varepsilon^j, \forall j \neq i \end{cases} \tag{3.13}$$

Os ε^i são escolhidos de tal forma que $\overline{f^i} \leq \varepsilon^i$, onde f^i corresponde ao valor ótimo do problema mono-objetivo minimizando apenas o objetivo f^i. A Figura 23 mostra os resultados das frentes de Pareto.

Para analisar a solução ótima justa, os autores verificaram a soma das altitudes dos drones alocados, em busca da menor altura máxima (Figura 24), e média do número de alvos

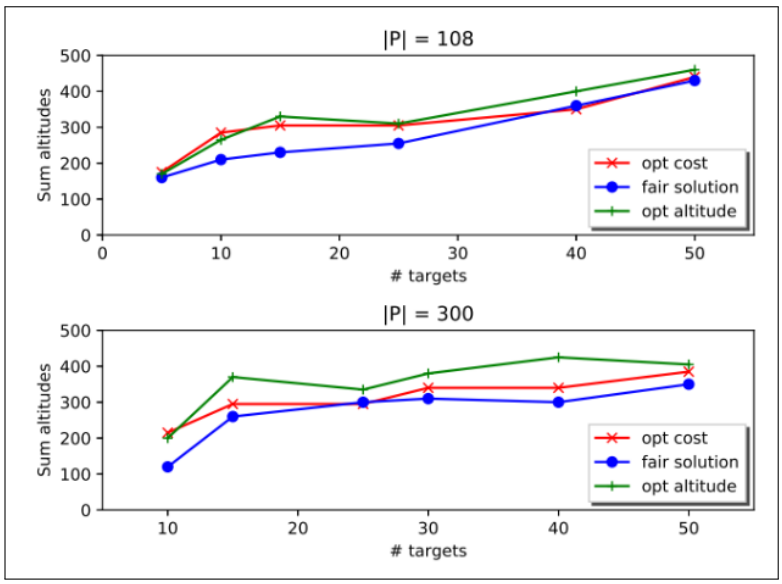
Figura 23 – Frentes de Pareto para diferentes tamanhos de N e P.



Fonte: Caillouet, Giroire e Razafindralambo (2018)

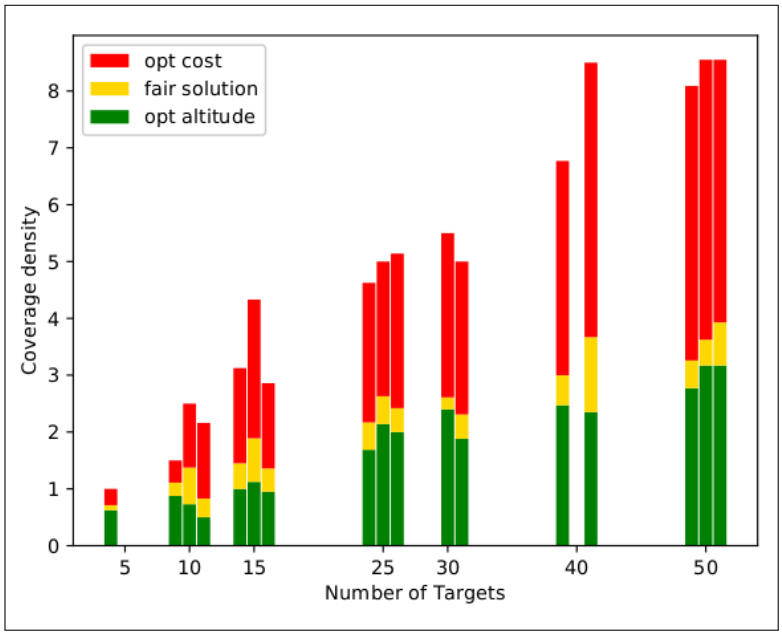
cobertos (Figura 25). Foram encontradas baixas altitudes globais para as UAVs e coberturas balanceadas entre os drones.

Figura 24 – Soma das altitudes de UAVs implantados.



Fonte: Caillouet, Giroire e Razafindralambo (2018)

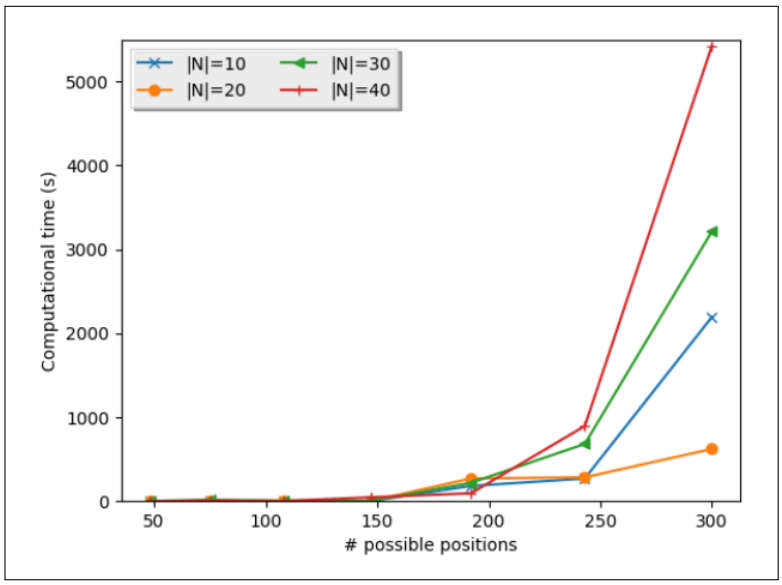
Figura 25 – Número médio de alvos cobertos pelos UAVs implantados para |P| = 108, 147, 300...



Fonte: Caillouet, Giroire e Razafindralambo (2018)

A fim de verificar a eficiência do modelo proposto foram analisados o tempo computacional gasto do programa (Figura 26), o custo de implantação das soluções ótimas (Figura 27), e o custo da conectividade entre os drones (Figura 28).

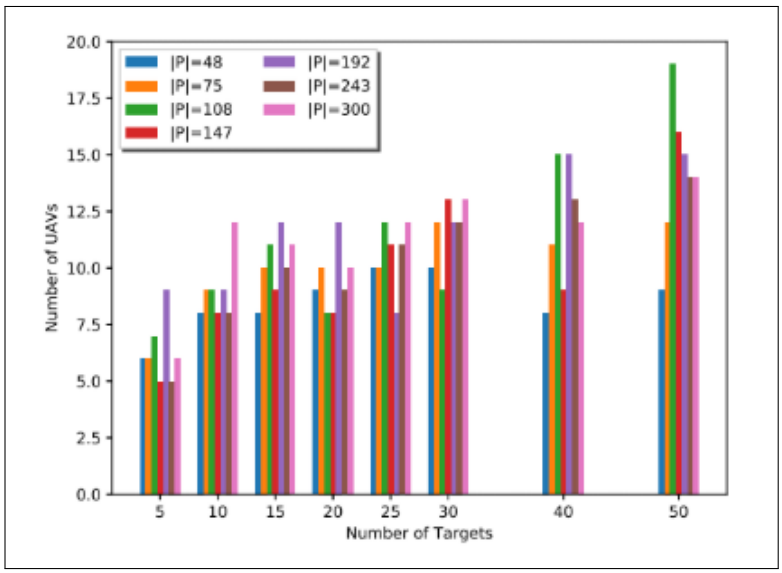
Figura 26 – Tempo computacional.



Fonte: Caillouet, Giroire e Razafindralambo (2018)

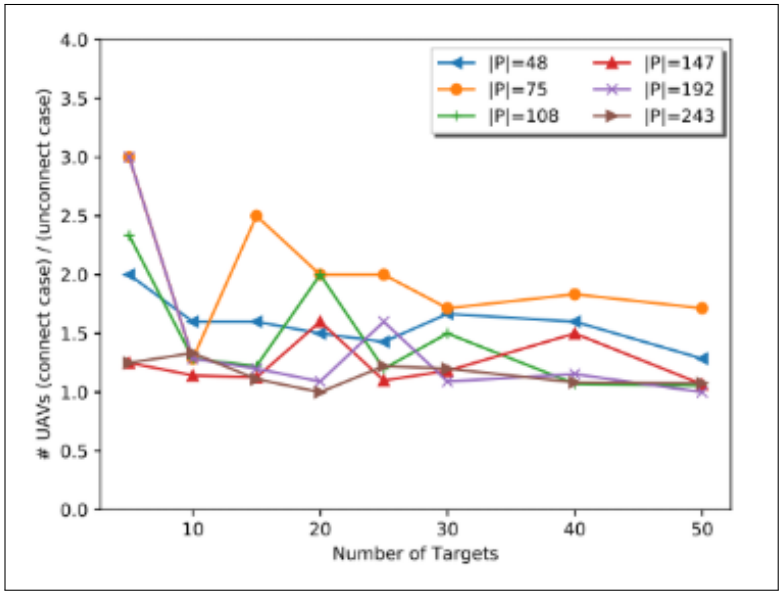
O tempo computacional foi razoável para até 200 posições possíveis para UAVs. Os

Figura 27 – Custo de implantação das soluções ótimas.



Fonte: Caillouet, Giroire e Razafindralambo (2018)

Figura 28 – Custo da conectividade dos UAVs.



Fonte: Caillouet, Giroire e Razafindralambo (2018)

autores verificaram que o custo de implantação aumenta linearmente de acordo com o número de alvos, mas se o número de posições possíveis P for pequeno, esse crescimento é quase constante. E quanto a custo de conectividade, foi verificado a influência das restrições de conectividade, com ou sem elas. Os autores então observaram que o custo é baixo (e até nulo) para grandes cenários em termos de número de alvos e possíveis pontos.

3.2 AN ALGORITHM OF EFFICIENT PROACTIVE PLACEMENT OF AUTONOMOUS-DRONES FOR MAXIMUM COVERAGE IN CELLULAR NETWORKS

Quanto ao uso de teoria dos grafos para o problema de implantação de drones autônomos, o trabalho de Huang e Savkin (2018) é outro que serviu de base para este trabalho. Eles tem como motivação atender aparelhos de usuários móveis em redes de *smartphones*. Eles propuseram um modelo de otimização que maximiza a cobertura dos usuários e minimiza o custo de comunicação entre drones, onde foram propostos duas visualizações da situação a partir de grafos, um com relação a comunicação entre drones e estações bases, e outro que eles chamam de *street graph*, que considera os cruzamentos das ruas da cidade com vértices e possíveis pontos de alocação de drones.

No trabalho é considerado um sistema de transmissão constituído de n drones autônomos, denominados de $i = 1, 2, ..., n$, e k estações bases fixas, nomeadas de $j = 1, 2, ..., k$. Os drones e as estações base devem se comunicar entre si respeitando um grafo de comunicação C. Esse grafo possui $n + k$ vértices, $v_1^r, v_2^r, ..., v_n^r, v_1^b, v_2^b, ..., v_n^b$, que equivalem aos drones e as estações base, respectivamente. Se existir uma aresta entre dois drones v_i^r e v_j^r, é possível haver comunicação entre eles, assim como se existir uma aresta entre v_i^b e v_j^b, o drone e a estação base podem se comunicar. Não há necessidade de duas estações bases se comunicarem no modelo proposto neste trabalho.

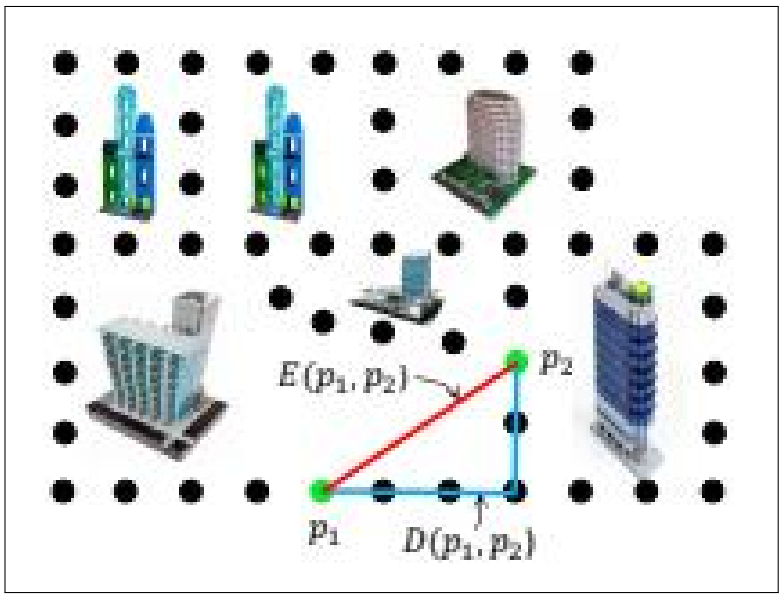
O ambiente urbano é modelado também como um grafo, chamado de *street graph* S, que contém M vértices $V_1, V_2, ..., V_M$, os quais representam as interseções entre duas ou mais ruas. S então é um conjunto de p pontos ao longo do plano, que podem conter drones e estações bases.

É definido que todos os drones estão na mesma altitude em relação ao plano horizontal. Seja $p1$ e $p2$ dois pontos, onde $p1, p2 \in S$, existe uma distancia $D(p1, p2)$ entre eles, que corresponde ao tamanho do menor caminho entre esses dois pontos dentro do grafo S. A figura 29 ilustra o grafo S e o conceito de distancia entre os dois pontos $p1$ e $p2$. $E(p1, p2)$ seria a distância euclidiana entre esses dois pontos, portanto $D(p1, p2) \geq E(p1, p2)$.

Alguns dos drones e estações bases devem estar perto o suficiente para respeitar o grafo de comunicação C, assim alguns restrições precisam ser satisfeitas, que são:

$$D(P_i, P_j) \leq R_1 \tag{3.14}$$

Figura 29 – Ilustração do *street graph* S.



Fonte: Huang e Savkin (2018)

se houver uma aresta entre os UAVs v_i^r e v_j^r em C, onde P_i e P_j são pontos onde foram posicionados drones autonomos, e $R1$ uma dada constante que é o alcance de comunicação entre os drones, com $R1 > 0$. E a outra restrição:

$$D(P_i, Q_j) \leq R_2 \tag{3.15}$$

se houver uma aresta entre o UAV v_i^r e a estação base v_j^b em C, onde P_i é um ponto onde foi posicionado um drone autônomo, e Q_j um ponto onde foi posicionada uma estação base. E $R2$ é o alcance de comunicação entre o drone e a estação base, com $R2 > 0$.

Existe também uma função $p(p)$, com $p(p) \geq 0$, que é continua e definida por todos os pontos em S, ela então é chamada de função densidade e descreve a concentração de aparelhos dos usuários em um determinado ponto p, em uma dado instante de tempo.

Seja o conjunto $C(P_1, ..., P_n) \subset C$ o comjunto de todos os pontos p, $p \in S$, cuja a distancia $D(p, P_i) \leq R$ para algum i. A qualidade da cobertura pelos drones colocados nos pontos $P_1, P_2, ..., P_n$ é descrito pela seguinte função:

$$\alpha(P_1, ..., P_n) := \int_{p \in C(P_1, ..., P_n)} p(p) dp \tag{3.16}$$

Além disso, se $c > 0$ e $d > 1$ forem dadas constantes, e seja ε o conjunto de pares (i, j) tal que os drones i e j estão conectados por uma aresta em C. O custo de energia das

comunicações entre i e j localizados nos pontos P_i e P_j são descritos pelo valor $D(P_i,P_j)^d$. Combinando isto com (3.14), obtém-se o seguinte problema de otimização:

$$\beta(P_1,...,P_n) := \int_{p\in C(P_1,...,P_n)} p(p)dp - c\sum_{(i,j)\in\varepsilon} D(P_i,P_j)^d \to max \tag{3.17}$$

Seja p algum ponto de S. Este ponto pode se mover continuamente ao longo de alguma aresta de S em alguma direção. Se p é um ponto interior de alguma aresta (V,W) de S conectando os vértices V e W, então p pode se mover continuamente em duas direções, seja para V ou para W. Se p for um vértice do grafo S com m bordas partindo deste vértice, então p pode se mover continuamente em m direções ao longo de qualquer uma dessas m arestas. Com isso e o problema de otimização descrito anteriormente, os autores propuseram o seguinte algoritmo de otimização:

1. Passo A1: Começa-se com alguns pontos iniciais $(P_1,P_2...,P_n)$ satisfazendo as restrições (3.14) e (3.15). Seja $N = sn+i$, onde $s = 0,1,2,...$ e $i = 1,2,...n$. Em cada passo N, move-se continuamente o ponto P_i da forma descrita no passo a seguir;
2. Passo A2: Se P_i é um ponto interno de alguma aresta que conecta os vértices V e W de S, e:

$$\sum_{p\in Pv(P_i)} p(p) + cd\sum_{j\in\varepsilon v(P_i)} D(P_i,P_j)^{d-1} \geq \sum_{p\in Pw(P_i)} p(p) + cd\sum_{j\in\varepsilon w(P_i)} D(P_i,P_j)^{d-1} \tag{3.18}$$

   então P_i está continuamente se movendo em direção a V até atingir o vértice V, ou atingir um ponto além do qual pelo menos uma das condições (3.18), (3.14), (3.15) deixam de ser respeitadas.
3. Passo A3: Se P_i for um vértice de S com m arestas partindo deste vértice rotulado $1,...,m$ e:

$$\sum_{p\in P_h(P_i)} p(p) + cd\sum_{j\in\varepsilon_h(P_i)} D(P_i,P_j)^{d-1} \geq \sum_{p\in\tilde{P}_h(P_i)} p(p) + cd\sum_{j\in\tilde{\varepsilon}_h(P_i)} D(P_i,P_j)^{d-1} \tag{3.19}$$

   para alguns h, então P_i está continuamente se movendo ao longo da aresta h até alcançar um outro vértice de S, ou atingir um ponto além do qual pelo menos uma das condições (3.19), (3.14), (3.15) deixam de ser respeitadas.

O algoritmo proposto por eles atinge um ótimo local em um número finito de etapas. E eles realizaram simulações abrangentes com base em um conjunto de dados real para demonstrar a eficácia do método proposto.

3.3 EFFICIENT DRONE PLACEMENT FOR WIRELESS SENSOR NETWORKS COVERAGE BY BARE BONES FIREWORKS ALGORITHM

Como exemplo de aplicação de cobertura utilizando drones não limitada uma ambiente específico, tem-se o trabalho de Tuba *et al.* (2018), que tem como objetivo posicionar drones para servirem de estações base de uma rede de sensores sem fio, de forma que deseja-se cobrir a maior quantidade de sensores com a menor quantidade de drones possível.

Para calcular o posicionamento dos drones, os autores adaptaram um algoritmo chamado *Bare bones fireworks algorithm* (BBFWA), proposto por Li e Tan (2018). Este algoritmo é uma meta heurística que faz alusão aos fogos de artifício, onde a melhor solução encontrada no ato da "explosão"seria onde estaria de fato o projétil, e as outras soluções seriam as faíscas ao redor dele.

Neste trabalho, a área de interesse, isto é, a região onde os sensores sem fio são implantados, é um terreno bidimensional retangular. Seja x_{max} o comprimento da área de monitoramento e y_{max} a largura correspondente. As posições dos drones são definidas no espaço tridimensional por suas coordenadas (x, y, z), onde a terceira dimensão representa a altura de voo da UAV. É definido que cada drone tem a mesma altura de voo máxima h_{max}. Conjunto de UAVs usados como estações base são denotados por U, enquanto S representa o conjunto de senadores sem fio. Ambos, sensores e drones são estáticos e as posições dos sensores são conhecidas.

A área de cobertura do UAV no solo pode ser definida pela seguinte equação:

$$D_{cov} = \pi r^2 = \pi(R^2 - h^2) \tag{3.20}$$

onde R representa o raio de detecção do drone, h é a altura do drone e r é o raio de cobertura do drone no solo.

A comunicação entre um sensor e o drone existe se a intensidade do sinal de rádio recebido (RSS) for maior que algum limite. Se o RSS_{min} representar a intensidade mínima do sinal de rádio recebido necessário para estabelecer a comunicação quando a distância entre o emissor e o receptor for de um metro, então o RSS (em dB) é definido pela seguinte equação:

$$RSS = RSS_{min} - 10\beta \log(d) + e \tag{3.21}$$

onde β é o coeficiente de perda de trajetória, d representa a distância, em metros,

entre o transmissor e o receptor, ou seja, entre o sensor e o drone e representa uma variável aleatória gaussiana de média zero onde o desvio padrão é variação estatística no *RSS* devido ao sombreamento. Como os autores consideraram apenas o modelo de cobertura, não a qualidade dos dados recebidos, foi considerou que o sensor é coberto se a distância for menor que d_{max}, a distância necessária para receber a intensidade mínima do sinal de rádio.

A distância entre o drone u posicionado em (x_u, y_u, z_u) e os sensores s_i com coordenadas (X_{s_i}, Y_{s_i}) é definida como a distância euclidiana:

$$D_{s_i}^{x_u,y_u,z_u} = \sqrt{(X_{t_i} - x_u)^2 + (Y_{t_i} - y_u)^2 + z_u^2} \tag{3.22}$$

A seguinte equação formaliza a determinação se o sensor $s_i \in S$ é coberto pelo drone $u \in U$:

$$Y_{s_i}^u = \begin{cases} 1, \ se \ D_{s_i}^{x_u,y_u,z_u} \leq d_{max} \\ 0, \ caso \ contrario \end{cases} \tag{3.23}$$

O objetivo principal é monitorar o número máximo de sensores com um número mínimo de UAVs e consumo mínimo de combustível (dependente da altura do drone). Para isso, o método proposto é testado com diferentes números de drones e a função objetivo foi reduzida para garantir apenas cobertura máxima com número fixo de UAVs. A função objetivo pode ser definida como:

$$max \sum_{s_i \in S} \sum_{u \in U} Y_{s_i}^u + \sum_{u \in U} \frac{1}{z_u} \tag{3.24}$$

onde os valores para Y_{s_i} são determinados pela Equação 3.24 e z_u representa a altura do drone u. Se um sensor é coberto por mais de um drone, ele é simplesmente considerado coberto.

Com a formalização matemática apresentada, pode-se seguir para o algoritmo proposto. No BBFWA, a solução inicial foi gerada aleatoriamente e considerada a melhor. Em cada geração, novas soluções (faíscas) são geradas no hiper-retângulo em torno da melhor solução da geração anterior. O tamanho do hiper-retângulo depende do tamanho do espaço de pesquisa no início. Nas últimas iterações, é controlado por dois parâmetros, C_a e C_r. Quando a nova melhor solução é encontrada, o tamanho do espaço em torno da melhor solução é aumentado pelo fator

$C_a > 1$, que é o parâmetro do algoritmo. Por outro lado, se a melhor solução permanece a mesma em uma geração, o tamanho do espaço de busca é reduzido pelo fator $C_r < 1$. A lógica por trás desse comportamento é que se nenhuma solução melhor for encontrada, então a área promissora é encontrada e ele precisa ser mais explorado gerando soluções em um espaço menor em torno dessa solução. Ao contrário, se a melhor solução foi encontrada, o espaço de pesquisa ainda não foi explorado de forma satisfatória.

A seguir tem-se o pseudocódigo do algoritmo proposto:

Algoritmo 6: BBFWA

Entrada: Vetor $x \sim U(Lb, Ub)$

Saída: Avaliar $f(x)$ pela equação 3.24

```
A = Ub − Lb
repita
    para i = 1 até N faça
        Vetor s_{p_i} ~ U(x − A, x + A)
        Aplicar o operador de mapeamento em s_{p_i}
        Avaliar f(s_{p_i}) pela equação 3.24
    fim
    se min_{i=1,2,...,n}(f(s_{p_i}) < f(x)) então
        x = argmin(f(s_{p_i}))
        A = C_a A
    fim
    senão
        A = C_r A
    fim
até Valor máximo de interação for atingido;
retorna x
```

No Algoritmo 16, Lb e Ub são os limites inferior e superior do espaço de pesquisa, respectivamente. O vetor x salva a melhor solução enquanto soluções s_{p_i}, $i = 1, 2, ..., n$ são geradas em torno de x em um hiper-retângulo com tamanho lateral $2A$. O valor inicial do parâmetro A é definido como $Ub - Lb$.

O método proposto foi testado em vários cenários e foi demonstrado que é muito eficiente em lidar com o problema de cobertura de redes de sensores sem fio utilizando drones estáticos.

3.4 PLACEMENT OF 5G DRONE BASE STATIONS BY DATAFIELD CLUSTERING

No trabalho de Iellamo, Lehtomaki e Khan (2017) foram propostos dois algoritmos de agrupamento para identificar grupos de usuários móveis a fim de posicionar estações bases de UAVs (Unmanned Aerial Vehicle Base Station ou UAV-BS) para fornecimento de rede 5G.

Os grupos são formados a partir de dados de espaço-tempo sobre o excesso de demanda para rede 5G, isto é, posições onde estações bases já posicionadas não são suficientes para fornecer internet para todos os usuários no determinado período de tempo.

Para resolver esse problema, os autores adaptaram a solução de Zhao *et al.* (2015), o qual propôs um método que analisa as variações de posicionamento de alvos ao longo do tempo em uma determinada região, e identificar regiões que ao longo do tempo tem mais chances de existirem aglomerações alvos. O algoritmo proposto por Zhao *et al.* (2015), o *trajectory clustering algorithm*, foi inicialmente aplicado para identificação de *hotspots* para táxis, isto é, pontos onde a maiores chances de existirem clientes para taxistas.

Baseado no *trajectory clustering algorithm*, os autores propuseram dois métodos para identificação das regiões que tem maiores chances de existirem maiores demandas por rede 5G. As definições de variáveis e equações para os algoritmos são aproveitadas do *trajectory clustering algorithm*, e já foram definidas neste trabalhos na Subseção 2.2.3.

O primeiro método proposto é o *max-capacity clustering algorithm* que é capaz de detectar *clusters* de qualquer tamanho e formato e, portanto, maximiza o aumento de capacidade que o operador de rede móvel pode obter se esses *clusters* forem devidamente atendidos. As principais etapas deste algoritmo são detalhados no Algoritmo 7.

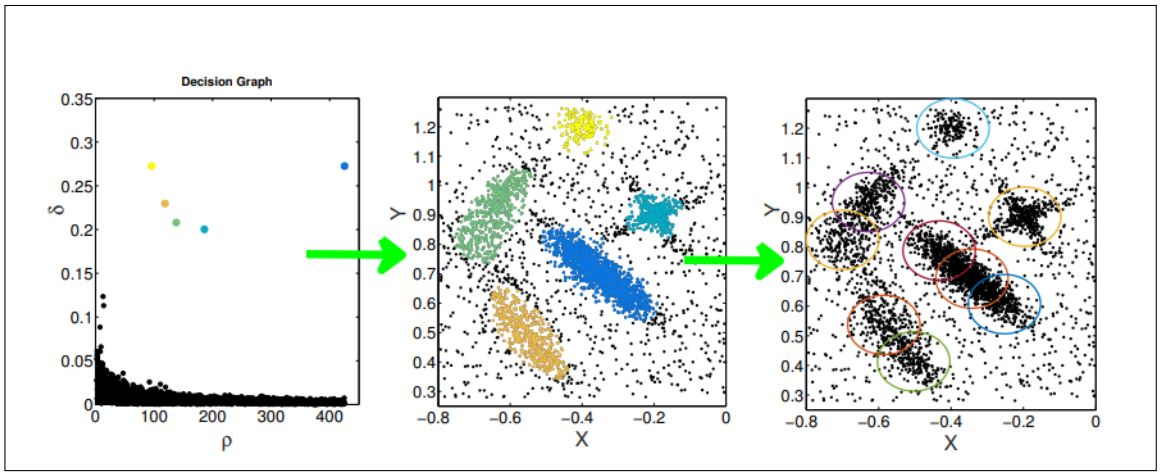
Algoritmo 7: *Max-capacity clustering algorithm*.

Entrada: t $\leftarrow$ tamanho do período de tempo; $d_c \leftarrow$ distancia máxima de corte.

1 **para** *cada período de tempo t* **faça**
2 Obtenha o fator de impacto ideal σ^* que minimiza o potencial de entropia $H(\sigma)$.
3 Calcule o valor potencial ϕ para cada ponto de demanda de dados de acordo com a Eq.(2.5).
4 Calcule $\delta_i(t)$ para cada ponto de demanda i (Eq.(2.8)).
5 Obter os valores limite de δ_{th}, ρ_{th}, ϕ_{th}^{sup} e ϕ_{th}^{inf}.
6 Identificar os centros dos *clusters*, isto é, todos os pontos i cujo $\delta_i(t) < \phi_{th}^{inf}(t)$.
7 Formar os *clusters* atribuindo cada ponto restante para o centro mais próximo que tenha o maior valor potencial.
8 Cubra os clusters formados colocando e operando devidamente um conjunto de UAV-BSs.
9 **fim**

Os valores limites δ_{th}, ρ_{th}, ϕ_{th}^{sup} e ϕ_{th}^{inf} são introduzidos pelos autores deste trabalho e não foram definidos por Zhao *et al.* (2015). Esses valores podem ser calculados a partir do método do cotovelo. A Figura 30 ilustra os passos dos Algoritmo 7.

Figura 30 – Exemplo visual das fases do Algoritmo 7: fase do gráfico de decisão (esquerda), fase de formação de *cluster* (centro) e fase de otimização da cobertura do drone (direita)



Fonte: Iellamo, Lehtomaki e Khan (2017).

Os autores queriam um método que indique exatamente onde as UAV-BSs devem ser posicionadas em vez das áreas a serem atendidas, para maximizar o aumento da capacidade. Assim, propuseram um segundo algoritmo que é uma versão interativa do Algoritmo 7, que gera *clusters* cuja extensão do centro é proporcional à cobertura máxima do UAV-BS e que o centro é indicativo a partir do ponto em que o UAV-BS deve ser colocado, levando em consideração o número de UAV-BS disponíveis. O *Iterative clustering algorithm for 5G UAV-BS placement* é apresentado no Algoritmo 8.

Para fazer experimentos e avaliar os algoritmos propostos, os autores utilizaram a base de dados GeoLife (ZHENG *et al.*, 2010), que contém dados de trajetória obtidos de GPS de 182 usuários, armazenados ao longo de uma coleta de cerca de 5 anos (Abril de 2007 a Agosto de 2012). A Figura 31 ilustra os resultados obtidos pelos testes. Os parâmetros para o Algoritmo 8 foram: $N_u = 30$, $d_c = 500m, d_{min} = 1km$.

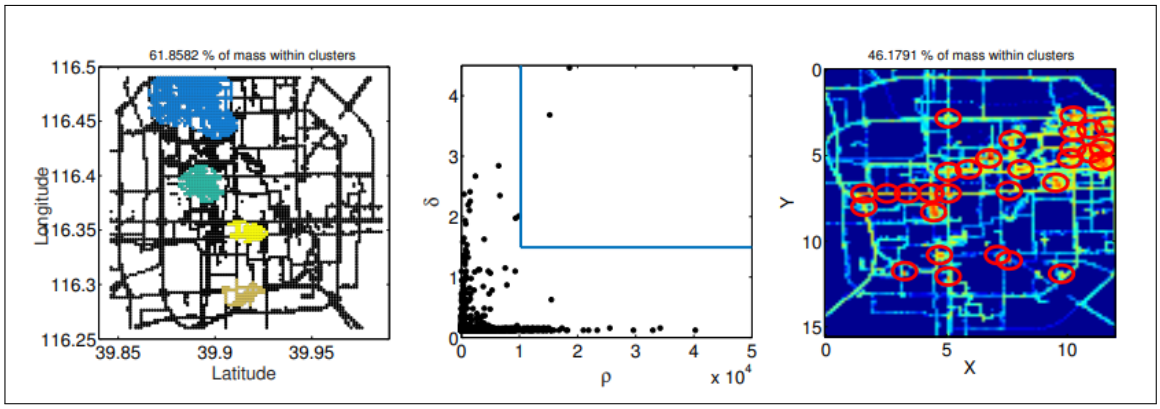
O Algoritmo 7 obteve aparentemente uma cobertura melhor do que o Algoritmo 8. Isso ocorre porque os *clusters* gerados pelo Algoritmo 8 garantem que a UAV-BS vai garantir a cobertura total da região, logo eles são menores do que os encontrados pelo Algoritmo 7, onde não há essa garantia e os *clusters* garantem apenas que haverá cobertura de pontos onde há

Algoritmo 8: *Iterative clustering algorithm for 5G UAV-BS placement*.

Entrada: $t \leftarrow$ tamanho do período de tempo; $d_c \leftarrow$ distancia máxima de corte; $d_min \leftarrow$ distancia mínima entre duas UAV-BSs; $N_u \leftarrow$ número de UAV-BSs disponíveis.

1 **para** *cada período de tempo t* **faça**
2 **repita**
3 Calcule o valor potencial ϕ para cada ponto de demanda de dados de acordo com a Eq.(2.5).
4 Calcule $\delta_i(t)$ para cada ponto de demanda i (Eq.(2.8)).
5 Escolher o valor limite de δ_{th} que seja maior que d_{min}, isto é, $\delta_{th} = min_{\delta_{elbow} > d_{min}}\{\delta_{elbow}\}$.
6 Obter os valores limite de ρ_{th}, ϕ_{th}^{sup} com o método do cotovelo.
7 Atribua cada ponto de demanda ao centro de *cluster* emitido mais próximo dentro de seu intervalo de cobertura.
8 Ordene os centros de *cluster* emitidos diminuindo a densidade δ_i e atribua um UAV-BS aos centros caracterizados por densidades mais altas primeiro.
9 Limpe todos os pontos de demanda que foram atribuídos a um centro de cluster (ou seja, atendidos por um UAV-BS) e atualize o número de UAV-BSs disponíveis.
10 **até** *ainda tiver excesso de demanda E ainda houver UAV-BSs disponíveis*;
11 **fim**

Figura 31 – Resultados para o Algoritmo 7 (esquerda) e para o Algoritmo 8 (direita). No centro tem-se o o primeiro gráfico de decisão de iteração que contribui (junto com outras 5 iterações) para gerar a figura no lado direito.



Fonte: Iellamo, Lehtomaki e Khan (2017).

excesso de demanda.

3.5 OUTROS TRABALHOS RELACIONADOS

Um trabalho relacionado a implantação de drones é o trabalho de Caillouet e Razafindralambo (2017), que compartilha alguns autores de Caillouet, Giroire e Razafindralambo (2018),

onde é proposto a implantação de um conjunto conectado de UAVs monitorando continuamente sensores móveis e enviando informações para uma estação base. Eles apresentam um modelo de otimização eficaz reduzindo o número de variáveis do problema e resolvendo usando a geração de colunas.

Os resultados mostram que o modelo proposto é tratável para grandes topologias com várias centenas de possíveis localizações 3D para a implantação de UAVs e fornece soluções inteiras com as colunas geradas muito próximas do ótimo. Além disso, as mudanças de implantação entre os tempos permanecem baixas em termos de número de drones e custo, para manter a conectividade e minimizar o atraso na coleta de dados para a estação base.

Ainda sobre implantação de drones, o trabalho de Amorosi *et al.* (2018) apresenta um abordagem voltada para resolver o problema de fornecimento de conexão 5G em zonas rurais utilizando estações bases carregadas pelas UAVs. Os autores tem como objetivo provar uma implantação eficiente do ponto de vista de consumo de energia das UAVs, de forma que o posicionamento garanta o fornecimento de 5G para toda a região de interesse e que também facilite a recarga estratégica de drones em estações de painéis solares.

Os autores então formularam o problema de otimização RURALPLAN, uma variante do problema de fluxo definido para um grafo que representa os deslocamentos dos UAVs. Os resultados em cima de cenários realísticos mostraram que a RURALPLAN é capaz de apresentar uma solução garantindo cobertura, mas não necessariamente considerando o gerenciamento de energia dos UAVs.

Em um contexto parecido com o trabalho de Amorosi *et al.* (2018), o trabalho de Li (2018) utiliza os UAVs como estações base para manter redes de celulares em regiões que perderam cobertura devido ao acontecimento de determinados incidentes, como desastres naturais.

A fim de encontrar o posicionamento ideal para cobrir a maior quantidade de usuários possível dentro da região afetada por algum desastre, sem ter o conhecimento prévio do posicionamento dos usuários (alvos), foi proposto o algoritmo *sweep and search*, que consiste em uma combinação de três etapas:

1. Decomposição poligonal da região: O algoritmo identifica região como um polígono convexo e calcula uma direção de varredura (*sweep*) ideal. A partir dessa varredura, decompõe-se a área em sub regiões, onde cada uma deverá ser coberta por uma UAV.
2. Controle de cobertura: Cada UAV deve cobrir a sub região que lhe foi atribuída seguindo um padrão em "zigzag", procurando sempre por aglomerados de usuários.

3. Prevenção de colisões: Define uma série de restrições, tais como distância mínima entre drones.

As simulações apresentadas mostram que o algoritmo proposto supera o algoritmo de pesquisa aleatória e o algoritmo de pesquisa atraente em relação ao número máximo de usuários cobertos sob a implantação de UAVs estações base.

Com relação ao monitoramento de alvos em uma determinada região, tem-se o trabalho de Xu *et al.* (2016). Os autores definiram como alvo animais em geral, ou seja, trabalham com alvos móveis. Neste trabalho foi proposto uma aplicação de monitoramento de animais utilizando redes de sensores sem fio e UAVs, no intuito de detectar locais de espécies ameaçadas em áreas de vida selvagem de grande escala e monitorar o movimento de animais sem dispositivos anexados aos animais.

Nessa aplicação, os sensores implantados em toda a área de observação são responsáveis pela coleta de informações sobre os animais. O UAV voa acima da área de observação e coleta as informações dos sensores. Para obter as informações de forma eficiente, os autores propuseram uma abordagem de planejamento de caminho para o UAV com base em um modelo de processo de decisão de Markov (*Markov Decision Process* ou MDP). O UAV recebe uma certa recompensa de uma área se alguns animais forem detectados naquele local.

Os autores implementaram o MDP usando o Q-learning (Técnica de aprendizagem de reforço) de tal forma que o UAV prefere ir para as áreas onde os animais são detectados antes. Enquanto isso, o UAV explora outras áreas para cobrir toda a rede e detecta mudanças nas posições dos animais. Assim definiram um modelo matemático subjacente ao problema de monitoramento de animais em termos do valor da informação (VoI) e recompensas. Propuseram um modelo de rede incluindo clusters de nós sensores e um único UAV que atua como um coletor móvel e visita os clusters. Em seguida, uma abordagem de planejamento de caminho baseada em MDP é projetada para maximizar o VoI e reduzir os atrasos de mensagens.

A eficácia da abordagem proposta é avaliada usando dois conjuntos de dados de movimento reais de zebras e leopardo. Os resultados da simulação mostram que a abordagem apresentada supera as heurísticas gananciosas e aleatórias, bem como o planejamento do caminho com base na solução do problema do vendedor ambulante.

Por fim, quanto ao posicionamento de drones sobre agrupamentos de alvos, o trabalho de Strumberger *et al.* (2017) apresenta a implementação do método *elephant herding optimization algorithm*, proposto por Wang, Deb e Coelho (2015), para resolver o problema de monitoramento de alvos estáticos por drones, com o objetivo de reduzir ao mínimo possível a quantidade UAVs

necessárias.

Para os testes, os autores utilizaram dois tipos de instancias: uma com 30 alvos distribuídos de forma uniforme, e outra com 30 alvos agrupados. Os resultados para a instancia com alvos agrupados foram melhores, necessitando de menos drones para cobertura completa dos alvos.

4 ABORDAGEM PROPOSTA

Neste capítulo é apresentado a definição do problema abordado neste trabalho e a modelagem dele como um problema de otimização, a fim de definir as restrições para validar as soluções dos métodos propostos neste trabalho. Também são apresentados os métodos propostos neste trabalho para encontrar soluções viáveis para o problema. São apresentadas duas abordagens: uma para ambientes estáticos (alvos fixos) e outra para ambientes dinâmicos (alvos móveis). Ao final deste capítulo, é feito estudo da complexidade dos algoritmos propostos.

4.1 MODELAGEM DO PROBLEMA

Em resumo, o problema abordado neste trabalho é o seguinte: Dado um conjunto de alvos distribuídos ao longo de uma região de interesse, tem-se que cobrir todos os alvos utilizando a menor quantidade de UAVs possível. O problema não considera o ambiente em que a região a ser observada se encontra, a solução deve ser genérica a qualquer tipo de região, a medida do possível.

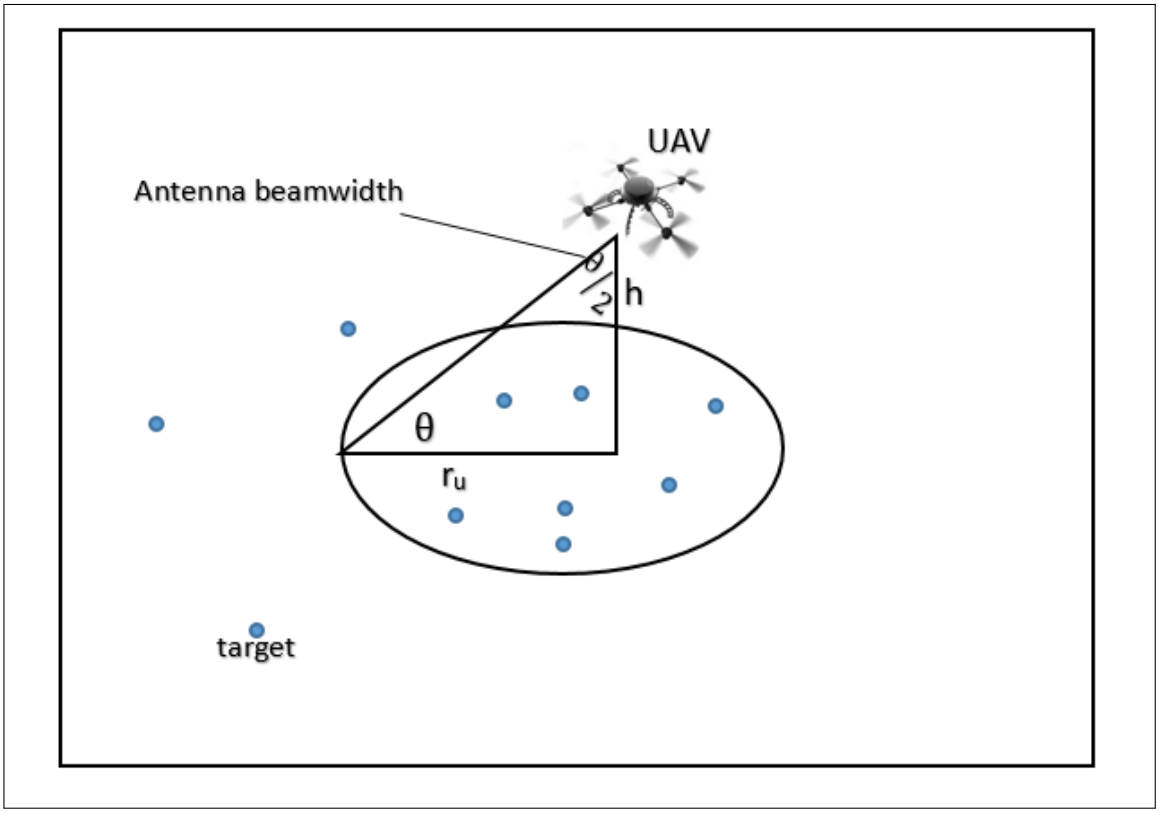
Para elaborar o modelo para resolução do problema de alocação de UAVs, neste trabalho é proposto uma abordagem, inspirada no método relatado por Huang e Savkin (2018), que utiliza teoria dos grafos, separando o problema de alocação em dois contextos, uma para comunicação entre drones, e outro para posicionamento de drones sobre uma região de interesse.

A região considerada neste trabalho é preferencialmente genérica, mas elas não são consideradas estruturas amplas que podem impedir o movimento das UAVs. Portanto, a nível de contextualização, a região de interesse representa áreas rurais grandes e esparsas. Com isso, por exemplo, os alvos podem ser animais ou máquinas agrícolas a serem monitoradas por drones.

Além disso, os alvos a serem observados são genéricos, categorizados apenas como estáticos ou móveis, e podem representar objetos, veículos, pessoas, animais, etc. Os UAVs considerados neste trabalho também são genéricos, mas podem ser descritos como dispositivos de observação, como os drones que o trabalho de Caillouet, Giroire e Razafindralambo (2018) considera. Esses drones são quadricópteros de baixo custo e possuem a câmera para observação na região frontal com a largura do feixe dita pelo angulo θ. O esquema de observação de uma UAV é mostrado na Figura 32.

O raio de observação (cobertura) r_u de um determinado drone u é descrito pela

Figura 32 – UAV genérica para observação de alvos.



Fonte: Caillouet, Giroire e Razafindralambo (2018).

seguinte equação:

$$r_u = h_u * tg(\theta/2) \tag{4.1}$$

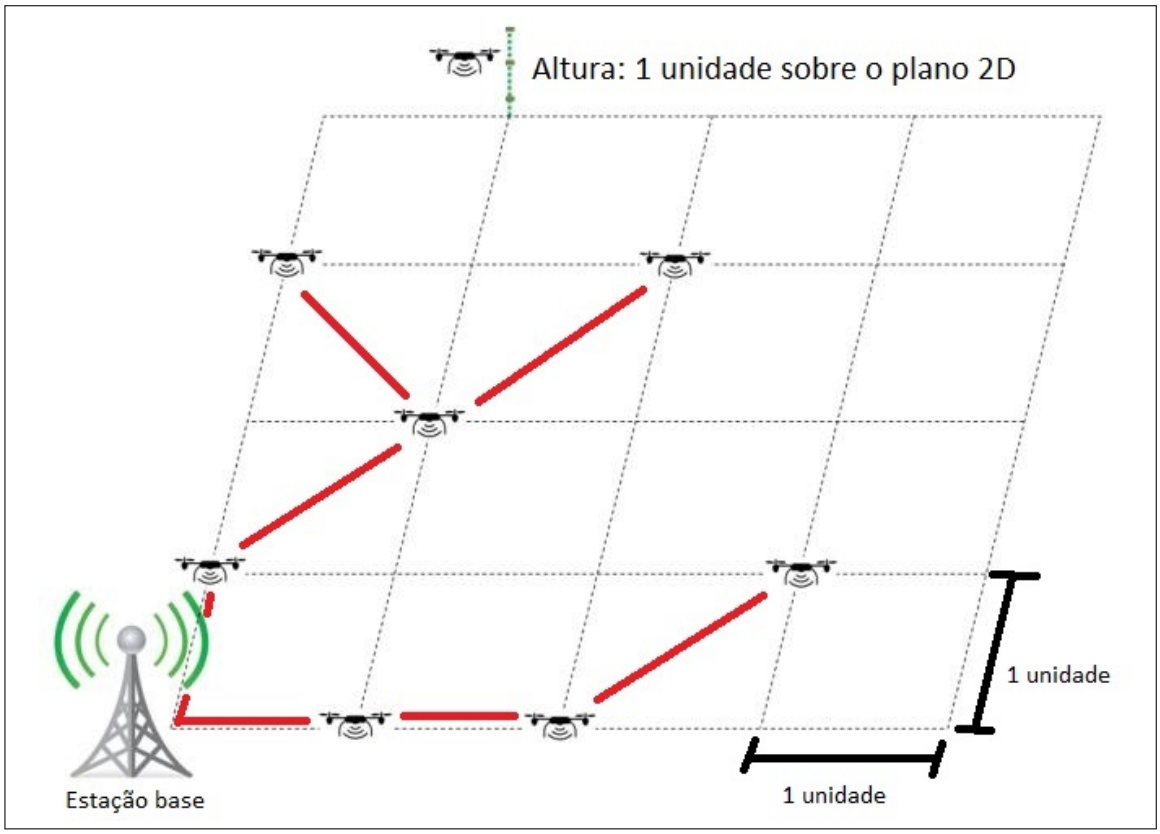
onde h_u é a altura em que o drone u esta com relação ao solo; E θ é o angulo entre a semi-reta do raio r_u com a hipotenusa do triangulo retângulo formado por r_u e h_u.

Considera-se que há apenas uma estação base e n drones à serem distribuídos pelo espaço. O grafo de comunicação C possui então $n+1$ vértices, considerando a quantidade de drones mais a estação base. Se um drone ou estação base u, onde $u \in C$, está dentro do alcance de comunicação do drone v, onde $v \in C$, então existe uma aresta vu entre eles em C. Será considerado que todos os drones e estação base possuem igual alcance de comunicação logo se existe vu em C, também há uv, assim C é um grafo bidirecional. Entretanto como a estação base b, onde $b \in C$, apenas recebe dados, logo o efeito bidirecional das arestas que chegam em b são meramente ilustrativos para a definição do problema.

A Figura 33 ilustra como seria o grafo de comunicação C de 7 UAVs e 1 estação base, considerando que a alcance de comunicação é de $\sqrt{2}$ unidade. As linhas vermelhas destacam as

arestas bidirecionais de C.

Figura 33 – Grafo de comunicação C.



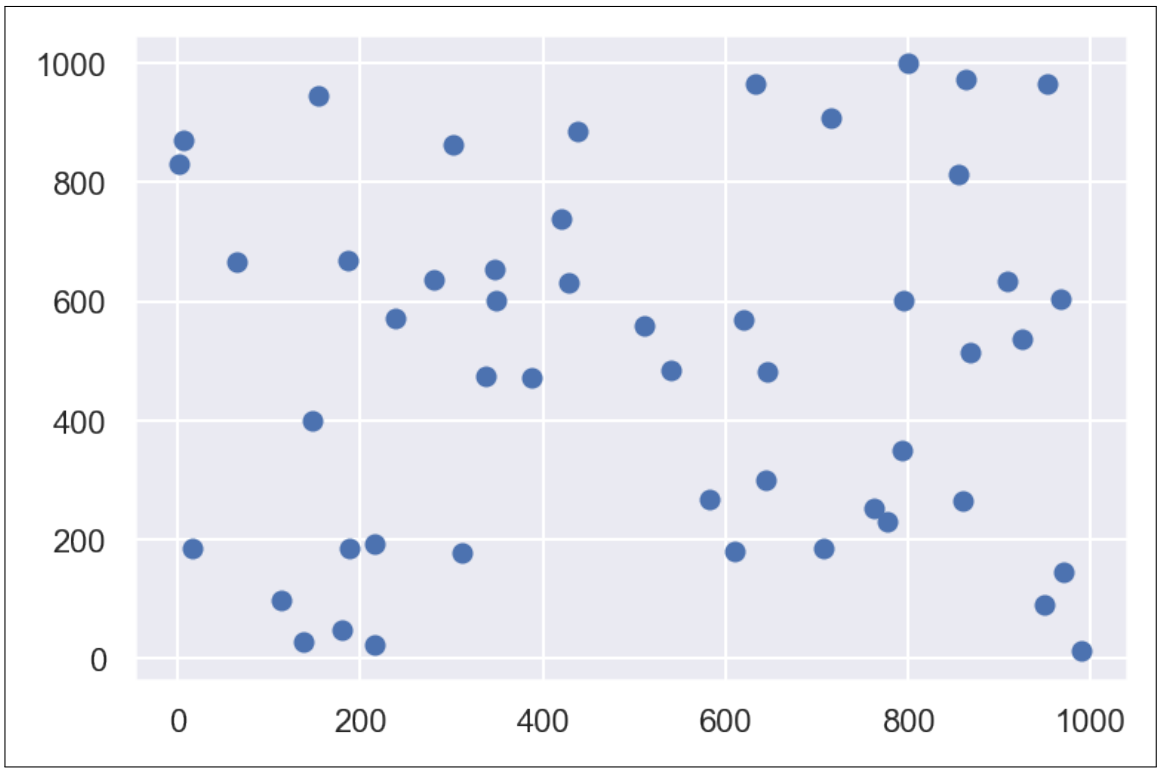
Fonte: Elaborada pelo autor.

Seja R a região de interesse onde são posicionados os alvos a, $a \in R$, os quais deseja-se que sejam observados pelos drones. A região R é representado por um plano quadrado. A Figura 34 ilustra um exemplo de região R com dimensões $1000x1000$ unidades com 50 alvos espalhados aleatoriamente pela região.

Os alvos presentes em R serão agrupados de forma que cada grupo seja formado por alvos que estão no máximo a r_u de distancia do UAV u, onde r_u é o raio de cobertura do drone u. Os grupos são formados considerando que os drones serão posicionados inicialmente posicionado sobre o centro do agrupamento de alvos aos quais eles foram associados, respectivamente.

Uma sub região c, onde $c \in R$, delimita um *cluster* de alvos encontrado em R. Os drones alocados devem, obrigatoriamente, sempre manter a orientação sobre as sub regiões c, $r \in R$, as quais foram designados. Cada região c pode possuir objetos que se desejam observar, cabe ao drone designado para ao *cluster* a responsabilidade de observar esse alvo, e caso algum deles saia da alcance do drone, deve imediadamente ser emitido um comunicado a partir do drone que perdeu o alvo de vista para os demais drones, e consequentemente para a estação base.

Figura 34 – Alvos espalhados pela região genérica *R*.



Fonte: Elaborada pelo autor.

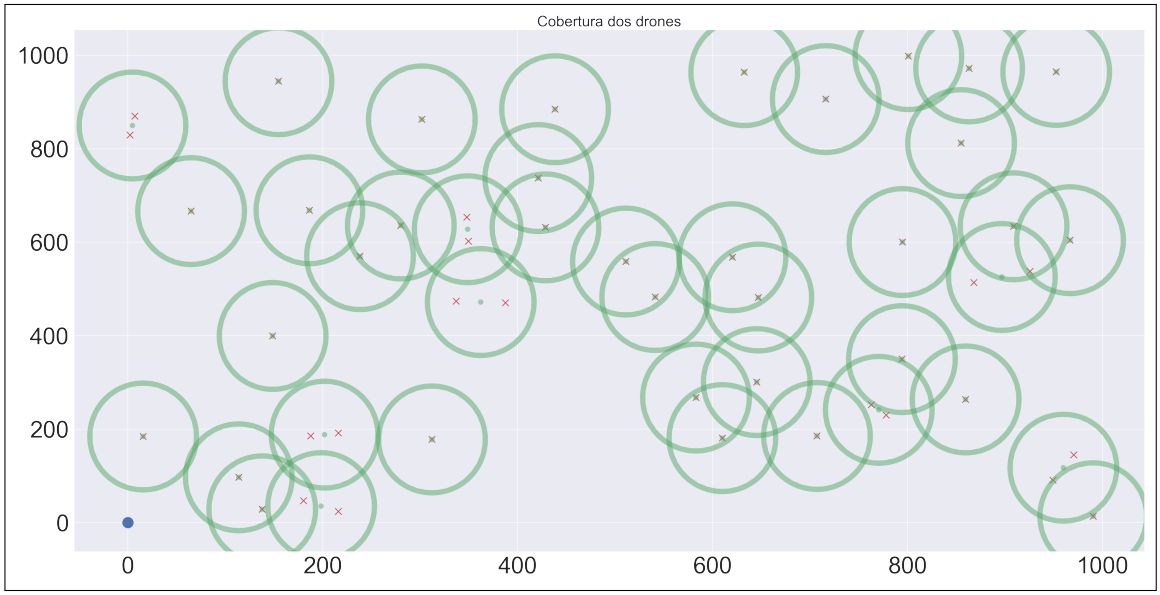
Na Figura 35 é ilustrado uma situação em que os drones foram posicionados sobre a região de interesse de forma que haja cobertura total dos alvos. Os pontos vermelhos com forma de "x"representam os alvos; o pontos verdes representam os UAVs posicionados na região, e os círculos verdes nos quais esses pontos ficam no centro deles indicam a área de cobertura do respectivo UAV; e o ponto azul representa a estação base, que neste exemplo foi posicionada na origem (0,0).

A seguir são apresentados os métodos propostos neste trabalho para encontrar soluções viáveis para o problema de alocação de UAVs. São apresentadas duas abordagens: uma para ambientes estáticos (alvos fixos) e outra para ambientes dinâmicos (alvos móveis).

Ambos os algoritmos seguem a seguinte lógica:

1. Determinam a quantidade de *clusters* de alvos presentes na região *R*, de forma que seja totalmente observável por apenas 1 drone (Tamanho do *cluster* igual à área determinada pelo raio *r* de observação da UAV);
2. Posicionam 1 drone sobre os centros de cada cluster;
3. Geram o grafo de comunicação *C* e verificam se ele é conexo. Se for, retornam a quantidade de drones encontrada, caso contrário, mais drones são posicionados sobre a região *R* até que o grafo *C* seja conexo.

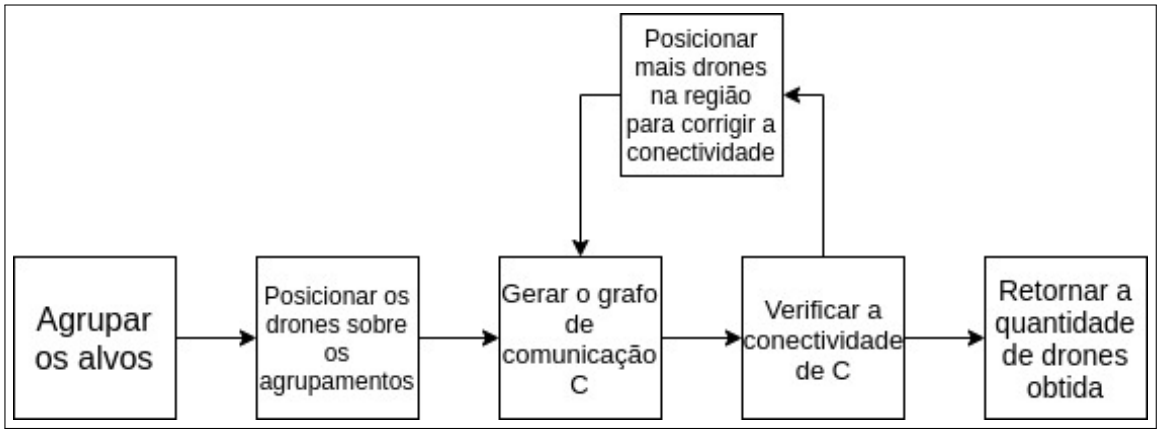
Figura 35 – Alocação de drones e cobertura de alvos sobre a região *R*.



Fonte: Elaborada pelo autor.

A Figura 36 apresenta o esquema padrão utilizado pelos algoritmos propostos citado acima.

Figura 36 – Representação esquemática da solução padrão utilizada nos algoritmos propostos.



Fonte: Elaborada pelo autor.

4.2 ALGORITMO PARA ALVOS FIXOS

O primeiro método apresenta a situação mais fácil de lidar, que seria com alvos fixos. O procedimento resume-se em agrupar esses alvos de forma que cada *cluster* seja observado por apenas um drone. A ideia por trás de alvos fixos, por exemplo, pode ser a representação

de estruturas ou objetos espalhados ao longo de uma planície, que seria interessante que sejam monitorados por drones.

O algoritmo recebe como entrada a base de dados as coordenadas das posições dos alvos presentes na região *R*, e a quantidade máxima de UAVs de observação disponíveis para posicionar.

Como não se sabe a quantidade de *clusters* a princípio, foi escolhido o algoritmo DBSCAN para fazer o agrupamento inicial dos alvos e calcular a quantidade de *clusters*. As delimitações de um *cluster* são definidas pela área de cobertura de um drone (referentes ao raio de observação do drone), e o valor mínimo de alvos para se ter em um agrupamento é igual a 1.

Com uma quantidade de *clusters* encontrada pelo DBSCAN, o próximo passo seria determinar se o posicionamento dos UAVs sobre os *clusters* forma um grafo conexo ou não. Para isso é interessante definir os centros dos agrupamentos, para representar a coordenada (X,Y) na qual o drone de fato será posicionado. A estrutura do DBSCAN não permite determinar os centros dos agrupamentos, ele retorna apenas uma quantidade de *clusters* que permite a agrupar de forma coerente todos os alvos dentro da região *R*. Esse valor pode ser entrada do algoritmo K-Means, que possui mecanismo de determinação dos centros dos *clusters*, mas necessita que seja dado como entrada a quantidade de agrupamentos a serem encontrados pelo algoritmo.

O próximo passo é identificar se o número de grupos encontrados pelo DBSCAN é de fato o melhor valor para *k*, ou se há outro valor que indique uma quantidade de *clusters* mais adequada para agrupar os alvos. Para isso, é aplicado o método do cotovelo. O valor inicial de *k* é exatamente o número de agrupamentos encontrados pelo DBSCAN, pois os parâmetros de entrada garantem que cada *cluster* encontrado seja totalmente observável por 1 drone, portanto um valor menor de *k* poderia não ter essa garantia.

Um valor maior para *k*, caso seja encontrado pela análise dos resultados do método do cotovelo, seria para corrigir possíveis casos em que os alvos foram classificados como ruídos na execução do DBSCAN, e não foram atribuídos a um *cluster*. Como a ideia é atribuir um 1 UAV por *cluster*, o valor máximo de *k* é igual a quantidade máxima de UAVs disponíveis. Caso o valor de *clusters* encontrados pelo DBSCAN for maior que o valor da quantidade máxima de UAVs, o valor de *k* para o K-Means passa ser igual a quantidade máxima de drones dado como entrada pelo algoritmo. Idealmente, o valor máximo de *k* seria igual a quantidade de alvos na região *R*, pois, no pior caso, pode ser que para haver a maior cobertura possível tem que ser necessário atribuir 1 UAV por alvo. O Algoritmo 9 resume os passos descritos acima com relação a encontrar o melhor valor de *k*. Foi utilizado o parâmetro "distorção", descrito na Subseção

2.2.1.1, para calcular o melhor k.

Algoritmo 9: MÉTODO PARA VERIFICAR MELHOR VALOR DE k PARA O K-MEANS

Entrada: P ← Base de dados com o posicionamento dos alvos; n ← número de *clusters* encontrados pelo DBSCAN (valor mínimo de k); MaxUAV ← quantidade máxima de UAVs disponíveis para posicionar (valor máximo de k)

distorções ← lista para armazenar os valores de distorção encontrado em cada execução de teste variando o valor de k.
para $k = n$ *até* $MaxUAV$ **faça**
    Armazenar em distorções resultado da distorção encontrada no K-Means(k).
fim
K = Kneedle_algorithm($k = \{n, n+1, ..., MaxUAV\}$, distorções)
retorna *Valor de K*.

Com o melhor k definido, utiliza-se o K-Means para encontrar novos agrupamentos de alvos em R, e determinar as coordenadas (X,Y) dos centros dos *clusters* para posicionar sobre ela a respectiva UAV responsável pelo monitoramento deste aglomerado.

Posicionadas as UAVs, o próximo passo é determinar as componentes conexas do grafo C de comunicação, formado a partir do Algoritmo 10.

Algoritmo 10: CONSTRUÇÃO DO GRAFO DE COMUNICAÇÃO C

Entrada: P ← base de dados com as coordenadas dos drones + estação base; d ← alcance de comunicação das UAVs

C ← Grafo de comunicação.
para *cada drone posicionado* **faça**
    Adicioná-lo como vértice de C.
fim
para *cada vértice u de C* **faça**
    para *cada vértice v de C* **faça**
        se $u \neq v$ **então**
            se *a distancia euclidiana entre u e v* $\leq d$ **então**
                Adicionar a aresta uv a C.
            fim
        fim
    fim
fim
retorna *a quantidade listas de vértices diferentes encontradas.*

Supondo a estação base b esteja posicionada, e que todas as n UAVs e b sejam vértices de C, são estabelecidas arestas entre os vértices C como descrito anteriormente neste capítulo. As componentes conexas são definidas através da aplicação de busca em profundidade (DFS) em C, como descrito no Algoritmo 11 a seguir:

Algoritmo 11: VERIFICAR COMPONENTES CONEXAS DO GRAFO *C*

Entrada: C ← Grafo de comunicação entre drones e estação base.

1 **para** *cada vértice no grafo C* **faça**

2 | Executar uma DFS partindo deste vértice, armazenando em um lista, todos os vértices armazenados na busca.

3 **fim**

4 **retorna** *a quantidade listas de vértices diferentes encontradas.*

Caso exista apenas uma componente conexa, o algoritmo segue para o próximo passo, caso contrário são calculados os centroides dos polígonos formados por cada componente conexa de *C*, e de em arranjos de 2 a 2, são posicionados um drones para servir de HUB[1] para comunicação entre os outros drones na coordenada correspondente ao ponto médio entre 2 dos centroides. Caso haja sobreposição de UAVs não é incrementado o valor de *n* (considera-se que há apenas um drone no respectivo ponto). E, caso a componente conexa possua menos de 3 vértices, isto é, não forme um polígono: Se tiver apenas 1 vértice, ele será considerado o centroide da componente conexa; E, se tiver 2 vértices, o ponto médio entre esses dois será considerado o centroide da componente conexa.

O algoritmo 12 resume o procedimento de determinação da quantidade e das posições dos HUBs para as UAVs.

A seguir repete-se o processo de construção do grafo *C* e em diante, até que haja apenas uma componente conexa.

Ao fim da execução, o algoritmo retorna a quantidade de drones final necessária para fazer a cobertura de todos os alvos em *R*, e garantir a conectividade de *C*. O Algoritmo 13 resume os passos descritos do método para alvos estáticos.

4.3 ALGORITMO PARA ALVOS MÓVEIS

Este segundo método representa uma adaptação do primeiro para alvos móveis, que já é uma situação um pouco mais complexa, visto que a abordagem de agrupamento pode não ser tão efetiva, visto que os *clusters* de alvos podem não se manter ao longo do tempo. Os alvos, por exemplo, podem ser animais de uma fazendas, que deseja-se que sejam monitorados para verificar se houve fuga ou furto deles.

A ideia desse algoritmo é inspirada na abordagem para detecção de *hotspots* para

[1] O HUB é um dispositivo que possui a função de conectar os computadores de uma rede local. Sua forma de trabalho é a mais simples se comparado ao *switch* e ao roteador: o HUB recebe dados vindos de um computador e os transmite às outras máquinas.

Algoritmo 12: ALGORITMO PARA POSICIONAR HUBS PARA COMUNICAÇÃO ENTRE UAVS

Entrada: Componentes_conexas ← lista com as listas de vértices presentes em cada componente conexa; P ← Lista com as posições das UAVs na região *R*.

```
Criar lista de centroides.
para cada componente conexa faça
    se a componente conexa tiver apenas 1 vértice então
        Adicionar as coordenadas desse vértice a lista de centroides.
    fim
    se a componente conexa tiver apenas 2 vértices então
        Calcular o ponto médio entre os dois vértices.
        Adicionar as coordenadas do ponto médio a lista de centroides.
    fim
    se a componente conexa tiver 3 ou mais vértices então
        Calcular o centroide do polígono formado pelos vértices da componente conexa.
        Adicionar o centroide a lista de centroides.
    fim
fim
para cada centroide da lista faça
    Calcular o ponto médio com relação a todos os outros centroides.
    para cada ponto médio calculado faça
        se não houver UAV posicionado sobre este ponto então
            Posicionar 1 HUB sobre o ponto médio.
        fim
    fim
fim
retorna a quantidade de HUBs encontrados.
```

táxis de Zhao *et al.* (2015), e adaptada ao contexto de drones por Iellamo, Lehtomaki e Khan (2017). Os pontos para posicionamento dos drones são identificados utilizando instancias com as coordenadas (X,Y) dos alvos ao longo de um determinado período de tempo (por exemplo, um dia), e verificando quais são as regiões mais densas ao longo do dia, isto é, que existem as maiores concentrações de alvos.

Assim, o algoritmo recebe como entrada a base de dados as coordenadas das posições dos alvos presentes na região *R* registradas ao longo de um período de tempo *t*, e a quantidade máxima de UAVs de observação disponíveis para posicionar.

Primeiramente, é utilizado o *trajectory clustering algorithm* proposto em Zhao *et al.* (2015), considerando os pontos de trajetória, como as variações de posicionamentos de cada alvo dentro da região *R*, ou seja, para cada alvo esse algoritmo é executado de forma que os pontos identificados como *hotspots*, isto é, os centros dos *clusters* encontrados para a trajetória do alvo sejam coordenadas candidatas a posicionamento de um drone.

Algoritmo 13: ALGORITMO PARA ALVOS ESTÁTICOS

Entrada: P ← Base de dados com o posicionamento dos alvos; MaxUAVs ← Quantidade máxima de UAVs disponíveis para posicionar.

1 Calcular a quantidade *n* de *clusters* de alvos em *P* usando o algoritmo DBSCAN.
2 A partir de *n*, calcular o melhor valor para *k* do K-Means, utilizando o método do cotovelo de forma automatizada (*Kneedle algorithm*), considerando o valor máximo de *k* igual a MaxUAVs.
3 Utilizar melhor valor *k* encontrado como entrada do algoritmo K-Means, e calcular as coordenadas dos centros dos novos *clusters* formados por ele.
4 Posicionar *n* drones no plano *R* sobre os centros dos *clusters* encontrados, e posicionar a estação base *b*.
5 Construir o grafo *C* de comunicação com algoritmo 10.
6 Calcular a quantidade de componentes conexas de *C* com algoritmo 11.
7 **repita**
8 Calcular os centroides dos polígonos formados por cada componente conexa em *C*.
9 Posicionar uma UAV para cada ponto médio entre dois centroides. Caso haja sobreposição de UAVs não é incrementado o valor de *n*, e considere-se que há apenas 1 drone naquele ponto.
10 Construir o grafo *C* de comunicação.
11 Calcular a quantidade de componentes conexas de *C* com algoritmo 11.
12 **até** *a quantidade de componentes conexas for igual a* 1;
13 **retorna** *a quantidade de UAVs encontrada.*

Após todos os *hotspots* para todos os alvos presentes na região *R* forem calculados, esses pontos são posicionados em um plano cartesiano (caso haja sobreposição de pontos, considerasse que há apenas 1 ponto na posição). Os passos seguintes são os mesmos do Algoritmo 13, porém o DBSCAN é aplicado utilizando os *hotspots* de alvos, ao invés dos próprios alvos. Assim os drones são posicionados em coordenadas candidatas a possuírem alto fluxo de alvos. O Algoritmo 14 resume os passos descritos do método para alvos móveis.

4.4 ESTUDO DE COMPLEXIDADE DOS ALGORITMOS PROPOSTOS

Ambos os algoritmos são formados por chamadas de outros métodos, logo a complexidade deles está diretamente ligada a complexidade dos algoritmos internos nas chamadas. Assim, primeiramente são relembrados as complexidades no pior caso dos algoritmos da literatura utilizados como componentes, como mostrado na Tabela 1.

Para o cálculo da complexidade do Algoritmo 9, considere *m* a quantidade de valores de *k* testados. Como o valor de *m* define a quantidade de chamadas do K-Means neste método, e também é o tamanho de cada lista dada como entrada da chamada do *Kneedle algorithm*, e como há apenas 2 dimensões para agrupar os alvos como K-Means, ou seja, *d* passa a ser uma

Algoritmo 14: ALGORITMO PARA ALVOS MÓVEIS

Entrada: P ← Base de dados com os posicionamentos dos alvos ao longo do período de tempo t.

1 Aplicar o algoritmo *trajectory clustering algorithm* para as trajetórias de cada um dos alvos em P dentro da região de interesse R ao longo do período de tempo t.
2 Posicionar os centros dos *clusters* (*hotspots*) encontrados para cada um dos alvos em um plano cartesiano. Em caso de vários pontos com a mesma coordenada (X,Y), considere que há apenas 1 ponto posicionado nessa coordenada.
3 Calcular a quantidade n de *clusters* dos *hotspots* de alvos usando o algoritmo DBSCAN.
4 A partir de n, calcular o melhor valor para k do K-Means, utilizando o método do cotovelo de forma automatizada (*Kneedle algorithm*).
5 Utilizar melhor valor k encontrado como entrada do algoritmo K-Means, e calcular as coordenadas dos centros dos novos *clusters* formados por ele.
6 Posicionar n drones no plano R sobre os centros dos *clusters* encontrados, e posicionar a estação base b.
7 Construir o grafo C de comunicação com algoritmo 10.
8 Calcular a quantidade de componentes conexas de C com algoritmo 11.
9 **repita**
10 Calcular os centroides dos polígonos formados por cada componente conexa em C.
11 Posicionar uma UAV para cada ponto médio entre dois centroides. Caso haja sobreposição de UAVs não é incrementado o valor de n, e considere-se que há apenas 1 drone naquele ponto.
12 Construir o grafo C de comunicação.
13 Calcular a quantidade de componentes conexas de C com algoritmo 11.
14 **até** *a quantidade de componentes conexas for igual a* 1;
15 **retorna** *a quantidade de UAVs encontrada.*

Tabela 1 – Complexidades dos métodos da literatura utilizados nos algoritmos propostos.

Algoritmo	Complexidade
DFS	$O(v+a)$
K-Means	$O(ndki)$
Kneedle algorithm	$O(n^3)$
DBSCAN	$O(n^2)$
Trajectory clustering algorithm	$O(n^2)$

Fonte: Elaborada pelo autor.

constante, a complexidade no pior caso é $O(mnki+m^3)$, onde n é o número de alvos a serem agrupados e i é o número de interações até o K-Means convergir. O número de interações i é definido como uma constante, logo pode ser ignorado na função de complexidade. Considerando o pior caso, em que é determinado a quantidade de UAVs a serem posicionadas igual a quantidade n de alvos, os valores de m e k seriam no máximo iguais ao valor de n, assim a expressão de complexidade no pior caso poderia ser reescrita como: $O(n^3+n^3)$, que reduzido ficaria $O(n^3)$.

O método para construir o grafo C (Algoritmo 10) possui complexidade igual a $O(v^2)$, definido pelo rotina de "para"duplo, em cada "para"percorre todo a lista de v vértices de

C. Como a quantidade de vértices no grafo é igual ao número de UAVs posicionadas na região R, pode-se considerar $v = n$, já que no pior caso, a quantidade de UAVs é igual a quantidade de alvos presentes na região R. Portando a complexidade no pior caso é igual a $O(n^2)$.

A complexidade do Algoritmo 11 para calcular as componentes conexas de C é definida pelas v chamadas do algoritmo DFS, onde v é quantidade de vértices em C, isto é, a quantidades de UAVs utilizadas mais a estação base. Portanto a complexidade deste algoritmo é $O(v(v+a))$, onde a é o número de arestas em C. No pior caso o grafo C sería o grafo completo, portanto $a = n(n-1)/2 = n^2$, e como $v = n$, tem-se que a complexidade no pior caso deste algoritmo seria $O(n(n+n^2))$, que reduzido ficaria $O(n^3)$.

O método para posicionar HUBs (Algoritmo 12) possui complexidade definida pelas duas rotinas "para", com respectivos tamanhos c e p, onde c é o número de centroides dos polígonos formados pelas componentes conexas e p é o número de pontos médios entre o centroide selecionado da interação e todos os outros centroides. Existe uma operação para calcular os pontos médios em o centróide selecionado e os outros centroides, resultando em um "para"parelelo ao "para"de tamanho "p", que vai possuir tamanho $c-1$. Como o valor de p é a quantidade de pontos médios definidos pela rotina de tamanho $c-1$, então $p = c-1$. Assim, a complexidade do algoritmo 12 é $O(c((c-1)+(c-1)))$, que pode ser reduzida para $O(c^2)$.

A operação de posicionar drones é feito por uma simples rotina "para"de tamanho $v-1$, considerando que a estação base já está posicionada. Portanto, como $v = n$, possui complexidade $O(n)$.

Para o algoritmo de alvos fixos, o laço "repita"define o operações, onde o é a quantidade de componentes conexas encontradas pelo Algoritmo 11. Esse laço então define o chamadas do algoritmo 12, o chamadas para posicionar UAVs/Hubs, e $o+1$ chamadas dos algoritmos 10 e 11, visto que eles são chamados uma vez antes do laço. Considerando a complexidade do algorito 12, tem-se $c = o$, visto que a quantidade de centroides é a mesma de componentes conexas. Além disso, tem-se $o = n$, pois no pior caso do procedimento de encontrar as componentes conexas o grafo seria totalmente desconexo, isto é, sem arestas entre os vértices, portanto existiram n componentes conexas. Assim a complexidade do Algoritmo 13 é dada por:

$$O((n^2)+O(n^3)+O(n^3)+(n-1)+(n+1)(n^2)+(n+1)(n(n+n))+o(n^2)+o(n-1))$$

$$O((n^2)+O(n^3)+O(n^3)+(n)+(n^3)+(n^3)+o(n^2)+o(n))$$

$$O(n^3)$$

E, por fim, para o algoritmo de alvos móveis, existe a execução do *Trajectory clustering algorithm* e um rotina para posicionar os *hotspots*, as quais não existem no algoritmo para alvos estáticos.

O *Trajectory clustering algorithm* é aplicado para encontrar o *hotspot* de cada um dos n alvos, fazendo que esta etapa tenha complexidade igual a $O(n * n^3)$, ou seja, $O(n^3)$.

Como são gerados *hotspots* na mesma quantidade de alvos, assim a rotina de posicionar os *hotspots* tem complexidade $O(n)$.

Considerando a complexidade do algoritmo de alvos estáticos, visto que os próximos passos do algoritmo de alvos móveis são iguais, a complexidade fica $O(n^3 + n^3)$, que reduzindo ficaria $O(n^3)$.

5 METODOLOGIA PARA OS EXPERIMENTOS

Neste capítulo são descritos as ferramentas utilizadas para implementação dos algoritmos propostos, os métodos utilizados para geração das instancias de teste, a escolha e o uso de uma base de dados externa para testes, assim como os meios de aferição da eficiência dos algoritmos propostos.

5.1 IMPLEMENTAÇÃO DOS ALGORITMOS PROPOSTOS

Os algoritmos foram implementados na linguagem Python (Versão 3.7.4) com o auxilio do ambiente de programação Jupyter Notebook (PEREZ; GRANGER, 2015). Foram utilizadas as seguintes bibliotecas do Python para auxiliar nas implementações:

- Numpy (NUMPY DEVELOPERS, 2018): Para diversas operações matemáticas, como cálculo de distancia, matrizes e geração de valores aleatórios;
- Networkx (HAGBERG; SWART; CHULT, 2008): Diversas utilidades para trabalhar com grafos, tais como, construção do grafos, desenho de grafos e algoritmos em grafos;
- Scikit Learn (PEDREGOSA *et al.*, 2011): Para os algoritmos de agrupamento, DBSCAN e K-Means;
- Kneed (SATOPAA *et al.*, 2011): Para o método que identificar o melhor número de *clusters* para o K-Means, a partir do método do cotovelo;
- Matplotlib (HUNTER *et al.*, 2012): Para plotar gráficos.

Para os métodos de agrupamento, foi considerado a distancia euclidiana para verificar a pertinência de cada um dos alvos em seu respectivo *cluster*.

5.2 GERAÇÃO DE INSTANCIAS

Para cada um dos algoritmos, foram geradas 90 instancias para cenários com 100, 500 e 1000 alvos distribuídos em uma região com dimensões $100x100$ ($10000m^2$), $500x500$ ($250000m^2$) e $1000x1000$ ($1000000m^2$). Os alvos são definidos como objetos simétricos que ocupam uma área no mapa de 1 m^2. As instancias foram geradas de acordo com a Tabela 2:

Para as instancias do algoritmo para alvos fixos, as instancias foram geradas por *scripts* escritos em Python, onde os alvos foram distribuídos aleatoriamente na região em cada um das 90 cenários descritos na Tabela 2. É garantido que não há sobreposição de alvos nas instancias, cada coordenada (X,Y) só pode haver 1 alvo.

Tabela 2 – Os tipos de cenários para a geração de instâncias.

Número de instancias	Quantidade de alvos	Área da região
30	100	$10000m^2$
30	500	$10000m^2$
30	1000	$10000m^2$
30	100	$250000m^2$
30	500	$250000m^2$
30	1000	$250000m^2$
30	100	$1000000m^2$
30	500	$1000000m^2$
30	1000	$1000000m^2$

Fonte: Elaborada pelo autor.

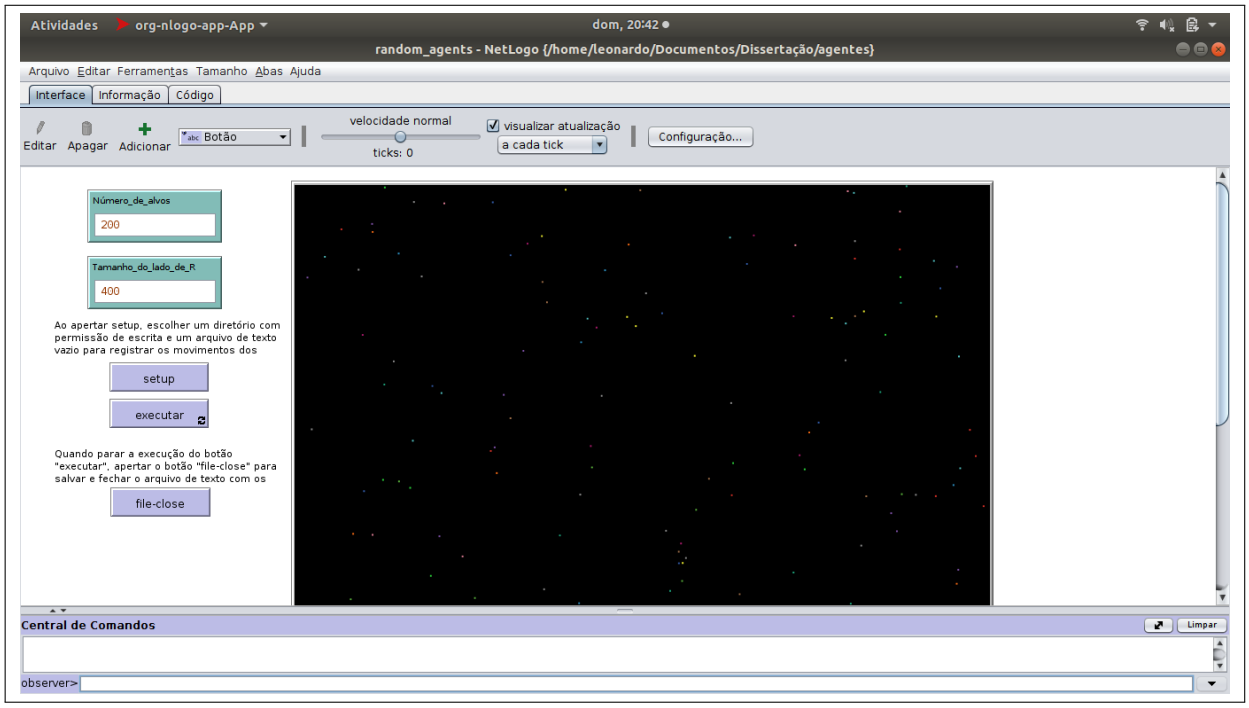
Para as instancias do algoritmo para alvos móveis, os alvos foram modelados de duas formas:

1. Alvo com movimento aleatório: Move-se aleatoriamente no mapa, e possui apenas as restrições de não colidir com outro alvo ou ultrapassar as delimitações da área;
2. Alvo com movimento inteligente: Possui as mesmas restrições do tipo de alvo anterior, porém pode apenas dar 15 passos por dia e andar em forma de "L"(5 passos para oeste e 10 passos para norte). Caso haja algum obstáculo no caminho, ele simplesmente vira para esquerda até que possa seguir com a sua cota de passos. (A ideia deste tipo de alvo é simular um alvo móvel com uma determinada rotina como, por exemplo, animais de fazenda).

Ambos os tipos de alvos percebem os outros alvos e os delimitadores da região. Eles podem, a cada instante de tempo, andar orientados para norte, sul, leste ou oeste, ou simplesmente ficarem parados. Para geração de instancias dos alvos móveis, eles foram implementados como agentes reativos simples (RUSSELL; NORVIG, 2016), e os cenários foram gerados com o auxílio da ferramento NetLogo (TISUE; WILENSKY, 2004). A Figura 37 ilustra uma instancia com 200 alvos dentro de uma região $400x400$, gerada utilizando o NetLogo. Os registro de movimentos de cada alvo ao longo dos instantes de tempo definidos, são armazenados em um arquivo de texto ao final da execução da aplicação no NetLogo.

Os cenários com o alvo do tipo 1 indicam o pior caso, onde os alvos não possuem uma rotina de deslocamentos, o que dificulta a eficiência do algoritmo. E os do tipo 2, já descrevem um situação um pouco melhor, onde os agentes simulam uma rotina dado os movimentos pré-determinados. Para cada uma das 9 variações descritas na Tabela 2, 10 dos cenários foram feitos utilizando os alvos com movimento aleatório e os outros 10 foram feitos utilizando os alvos com movimento inteligente.

Figura 37 – Visualização de uma instancia gerada no Netlogo de uma região R de dimensões $400x400$, com 200 alvos.



Fonte: Elaborada pelo autor.

Para cada um dos 90 cenários, foram gerados 4 registros de deslocamentos dos alvos, cada um mostrando a posição dos alvos no respectivo instante t de tempo, onde $t = 1,2,3,4$, representando o monitoramento dos alvos ao longo do período de 1 dia, onde cada valor de t representa uma etapa do dia (madrugada, manhã, tarde, noite), como no trabalho de Iellamo, Lehtomaki e Khan (2017).

5.3 BASE DE DADOS EXTERNA

Além das instancias de teste geradas, foi utilizado também uma base de dados da literatura para verificar a eficiência do algoritmo para alvos móveis. Isto porque este algoritmo é aplicado em situações um pouco mais complexas, e os cenários gerados podem não apresentar uma experiencia completa de uma situação realista em que esse algoritmo seria aplicado na prática. Além que o algoritmo para alvos móveis engloba o algoritmo para alvos fixos, representando uma evolução dele.

A base escolhida foi a T-Drive trajectory[1] (YUAN *et al.*, 2010; YUAN *et al.*, 2011), que consiste em coletas de trajetórias de 10357 táxis em um período de uma semana. O número

[1] A base de dados encontra-se disponível em https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/?from=https%3A%2F%2Fresearch.microsoft.com%2Fapps%2Fpubs%2F%3Fid%3D152883

total de pontos neste conjunto de dados é de cerca de 15 milhões e a distância total das trajetórias atinge 9 milhões de quilômetros, considerando uma região de $1000000m^2$.

A ideia é considerar os táxis como os alvos a serem observados, para aplicar o algoritmo para alvos móveis e encontrar os melhores posicionamentos para as UAVs nesse contexto.

5.4 EXPERIMENTOS

Os experimentos foram conduzidos em uma máquina com um sistema operacional Linux (Ubuntu 18.04.3) com processador AMD Rysen 5 2400G, com 16 GB de memória ram e placa de vídeo intel 1050TI 4G.

Nos experimentos com as instancias gerados foram feitos, para cada um dos algoritmos propostos, teste das 270 instancias feitas para cada um dos tres tipos de posicionamentos da estação base (origem, meio e aleatório), ou seja, ao todo foram realizados 810 testes para o algoritmo de alvos fixos, e 1620 testes para o algoritmo de alvos móveis.

Quanto aos experimentos com a base dados externa, o algoritmo para alvos móveis foi aplicado 3 vezes, uma vez para cada tipo de posicionamento da estação base (origem, meio e aleatório).

Para todos os experimentos, foi fixado o raio de observação dos drones igual a $26m$, que é aproximadamente o valor máximo considerado pelo trabalho de Caillouet, Giroire e Razafindralambo (2018), com altura do drone fixada em $45m$ e angulo de observação fixado em 60^o, calculado usando a equação 4.1 (Seção 4.1). E o alcance de comunicação entre drones foi definido igual a $300m$, considerando o limite padrão de comunicação Wi-Fi (IEEE 802.11 a/b/g/n) entre drones (MAMMADOV; GUEAIEB, 2014). A quantidade de UAVs máxima para cada execução dos algoritmos foi estabelecida como a quantidade de alvos presentes na respectiva instancia.

Para investigar a influencia dos valores de entrada dos algoritmos, testes em algumas instancias foram repetidos, limitando a quantidade de UAVs disponíveis ao valor encontrado de drones necessários pelos algoritmos, para cobertura da respectiva instancia. No caso das instancias geradas, foram repetidos os testes para as regiões com 1000000 m^2 de área. E, para a base de dados externa, todos os testes foram repetidos.

Por fim, quanto a aferição dos algoritmos, eles foram avaliados com relação aos seus respectivos resultados quanto ao monitoramento de cada cenário de teste, com relação ao tempo

de execução, a quantidade de drones encontrados para posicionar sobre a região R, quantidade de HUBs necessárias, e a taxa de cobertura de alvos possível com a quantidade de drone que ambos os algoritmos obtiveram para o monitoramento dos respectivos cenários.

6 RESULTADOS

Neste capítulo são apresentados os resultados do experimentos descritos no Capítulo 5. Primeiramente, os algoritmos propostos foram testados com as instâncias geradas. A seguir, o algoritmo para alvos móveis é testando utilizando a base de dados externa

6.1 RESULTADOS DOS TESTES DOS ALGORITMOS PROPOSTOS UTILIZANDO AS INSTÂNCIAS GERADAS

As Figuras 38 – 46 apresentam, cada uma respectivamente, os resultados dos algoritmos propostos quanto:

A Quantidade média de UAVs encontradas pelos algoritmos (Considerando a quantidade UAVs para observação dos alvos mais a quantidade de HUBs que foram utilizadas);

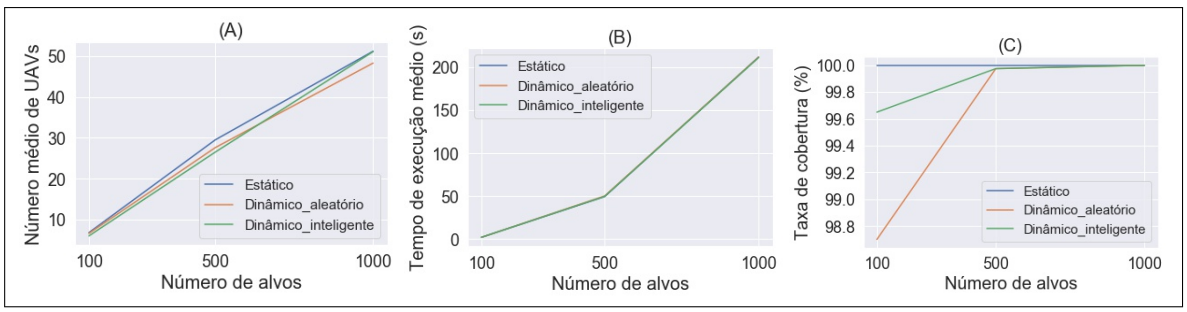
B Tempo médio de execução (em segundos);

C Taxa média de cobertura de alvos pelos drones.

Nestas figuras são mostrados resultados variando entre si os parâmetros de tamanho da região R (10000 m^2, 250000 m^2 ou 1000000 m^2) e posicionamento da estação base (Origem, meio ou aleatório).

No caso do algoritmo para alvos móveis, os resultados para alvos de comportamento aleatório e para alvos que se movimentam de forma inteligente são apresentados de forma separada nos gráficos a seguir.

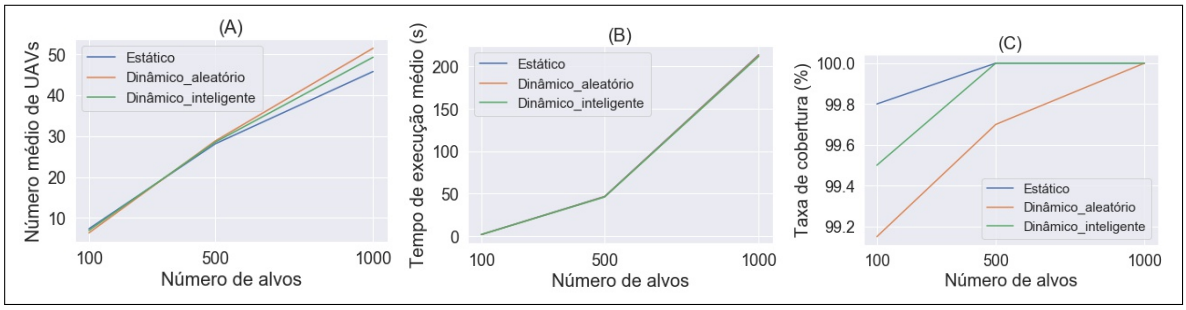
Figura 38 – Resultados para os algoritmos propostos aplicados em uma região de 10000 m^2 **e com a estação base posicionada na origem (0,0).**



Fonte: Elaborada pelo autor.

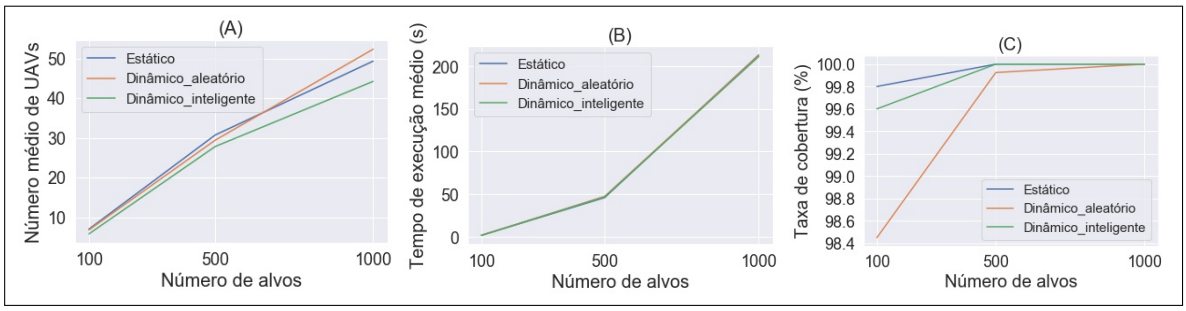
É notável a influencia da densidade da região com relação a taxa de cobertura média encontrada pelos algoritmos, isto é, quanto maior a quantidade de alvos da instancia e menor for o tamanho da região R, maiores serão as quantidades alvos observados pelos UAVs. Os resultados

Figura 39 – Resultados para os algoritmos propostos aplicados em uma região de 10000 m^2 **e com a estação base posicionada no meio.**



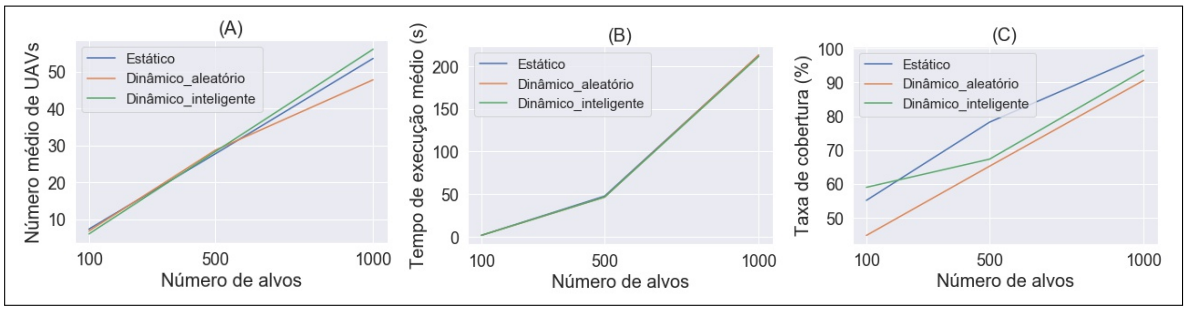
Fonte: Elaborada pelo autor.

Figura 40 – Resultados para os algoritmos propostos aplicados em uma região de 10000 m^2 **e com a estação base posicionada aleatoriamente.**



Fonte: Elaborada pelo autor.

Figura 41 – Resultados para os algoritmos propostos aplicados em uma região de 250000 m^2 **e com a estação base posicionada na origem (0,0).**

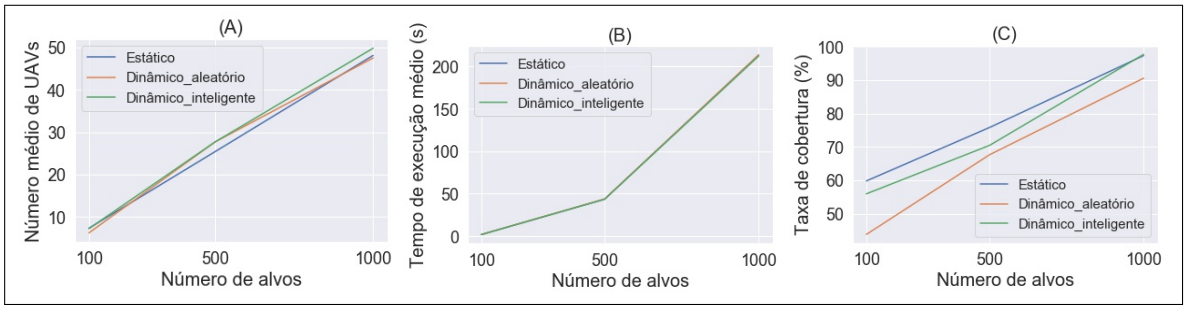


Fonte: Elaborada pelo autor.

dos algoritmos para alvos móveis quanto a este quesito foram piores que os do algoritmo de alvos fixos, porque ele posiciona os UAVs sobre pontos calculados a partir de estimativas de posicionamentos dos alvos, enquanto o outro calcula os posicionamentos a partir das localizações exatas dos alvos.

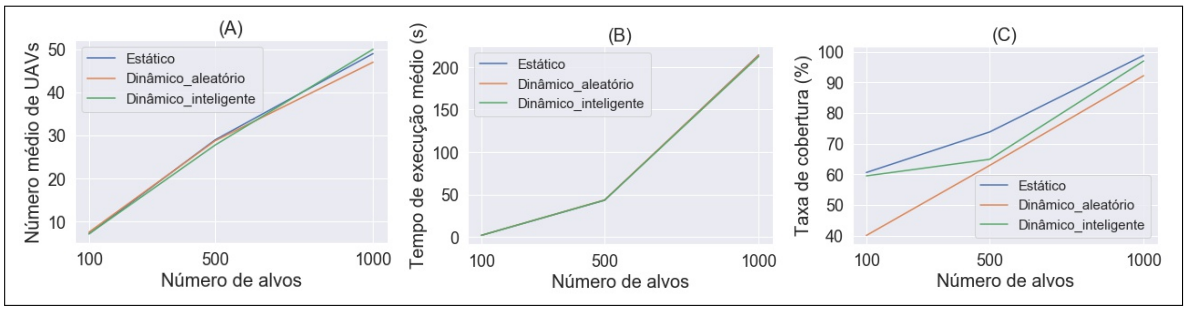
O comportamento do alvo móvel também influenciou nos resultados do algoritmo para alvos móveis, onde as execuções sobre as instancias de geradas a partir de agentes reativos

Figura 42 – Resultados para os algoritmos propostos aplicados em uma região de 250000 m^2 **e com a estação base posicionada no meio.**



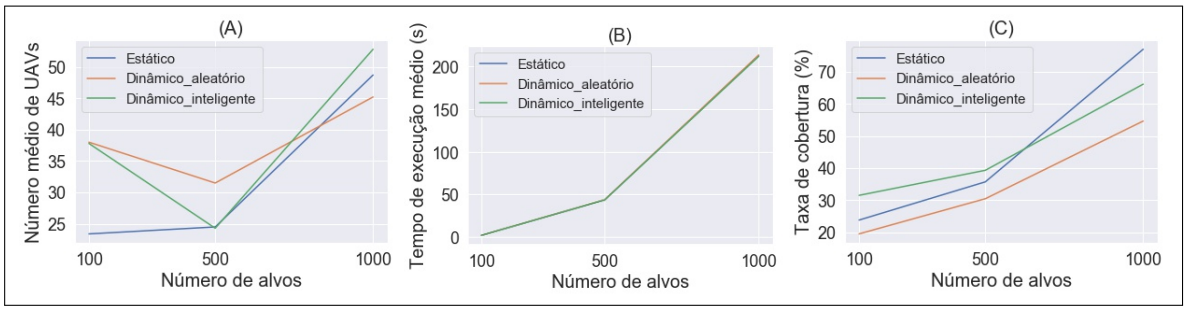
Fonte: Elaborada pelo autor.

Figura 43 – Resultados para os algoritmos propostos aplicados em uma região de 250000 m^2 **e com a estação base posicionada aleatoriamente.**



Fonte: Elaborada pelo autor.

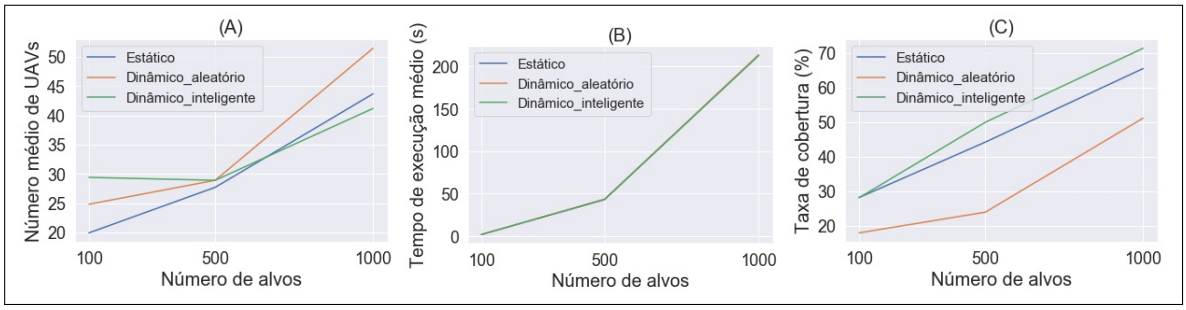
Figura 44 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 **e com a estação base posicionada na origem (0,0).**



Fonte: Elaborada pelo autor.

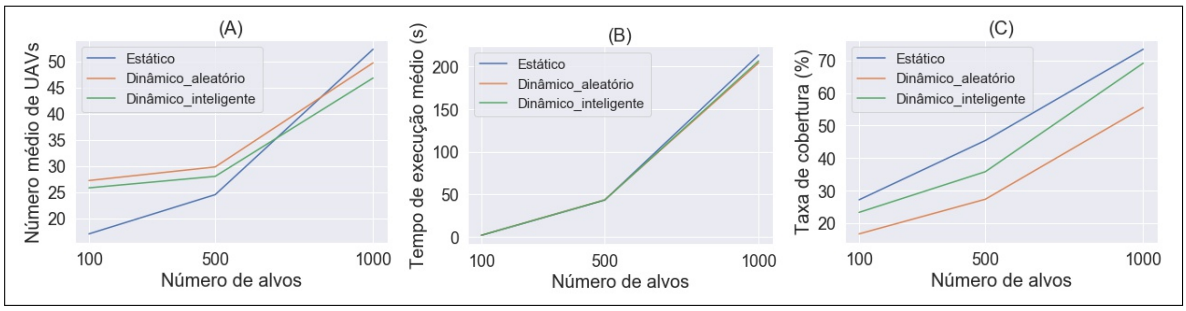
simples com movimentos "inteligentes"obtiveram melhores resultados quanto a taxa média de cobertura do que aquelas nas instancias com alvos que se movem de forma aleatória. Isso ocorre porque o alvo que se move de forma inteligente tende a ser mais fácil de ser encontrado, pois ele se move de maneira previsível, como se tivesse uma rotina. Inclusive os resultados para as instancias com alvos de movimentos não aleatórios não foram muito distantes dos obtidos pelos testes do algoritmo de alvos fixos.

Figura 45 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 e com a estação base posicionada no meio.



Fonte: Elaborada pelo autor.

Figura 46 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 e com a estação base posicionada aleatoriamente.



Fonte: Elaborada pelo autor.

Quanto aos tempos médios de execução, independentemente dos parâmetros de tamanho da região R e posicionamento da estação base, os algoritmos obtiveram resultados bastante próximos entre si (podendo considerar iguais de tão próximos), sendo a quantidade de alvos o fator determinante para o tempo de execução, visto que ele aumenta consideravelmente quanto maior a quantidade de alvos.

Entretanto não é necessariamente verdade que a quantidade de alvos é o fator determinante para o crescimento do tempo de execução. Considerando as respectivas complexidades dos algoritmos, descritas na Tabela 3:

Tabela 3 – Complexidades dos algoritmos propostos.

Algoritmo	Complexidade
Alvos estáticos	$O(mnki + m^3 + o(v^2 + ova + o^2))$
Alvos móveis	$O(n^3 + mnki + m^3 + o(v^2 + ova + o^2))$

Fonte: Elaborada pelo autor.

As expressões de complexidade indicam que o número de UAVs disponíveis dado

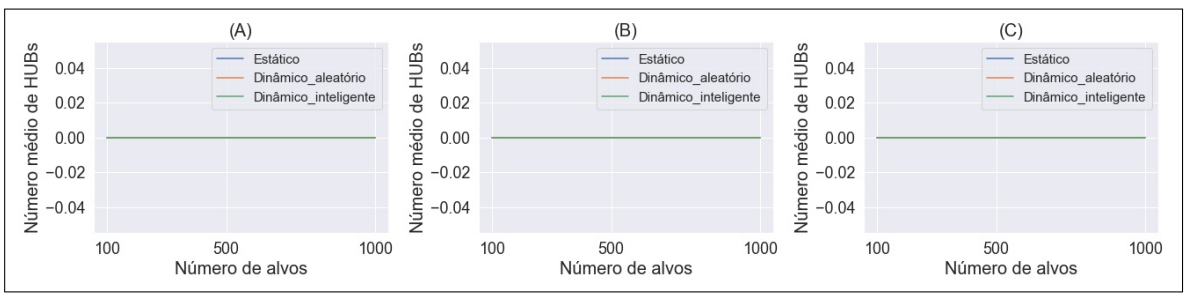
como entrada é mais suscetível de influenciar no tempo de execução. Como m é o número de interações para encontrar o valor de k ideal para o K-Means, ele é definido pela diferença entre o número máximo de UAVs disponíveis e quantidade de *clusters* encontradas pela etapa do DBSCAN.

Além disso os valores de o componentes conexas, v vértices e a arestas do grafo C, são todos determinados inicialmente a partir do número de $UAVs$ encontrados, isto é, o número de *clusters* (valor de k) definido como ideal para o K-Means. Assim, o número de $UAVs$ disponíveis para posicionar influencia muitas mais variáveis dos algoritmos do que o número de alvos a serem observados, portanto possui maior probabilidade de influenciar no tempo de execução.

A quantidade média de UAVs encontradas pelos algoritmos também variou pouco de acordo com os parâmetros de tamanho da região R e posicionamento da estação base, seguindo o mesmo comportamento crescente em relação ao número de alvos que o tempo de execução médio. Entretanto, as médias para ambos os algoritmos em relação a situação de menor densidade, isto é, 100 alvos distribuídos em um região de área igual a 1000000 m^2, fugiram um pouco da lógica de crescimento da quantidade de UAVs diretamente proporcional a quantidade de alvos.

Observando os gráficos das Figuras 47 – 49, nota-se que o caso de menor densidade descrito acima foi a única situação que necessitou de uso significativo de HUBs para garantir a conectividade do grafo de comunicação C. Como a quantidade média de UAVs das Figuras 38 – 46 são obtidas a partir da quantidade total de UAVs + HUBs de cada execução dos algoritmos, o uso das HUBs ocasiona no comportamento diferente nos gráficos de número médio de UAVs posicionadas das Figuras 44 – 46.

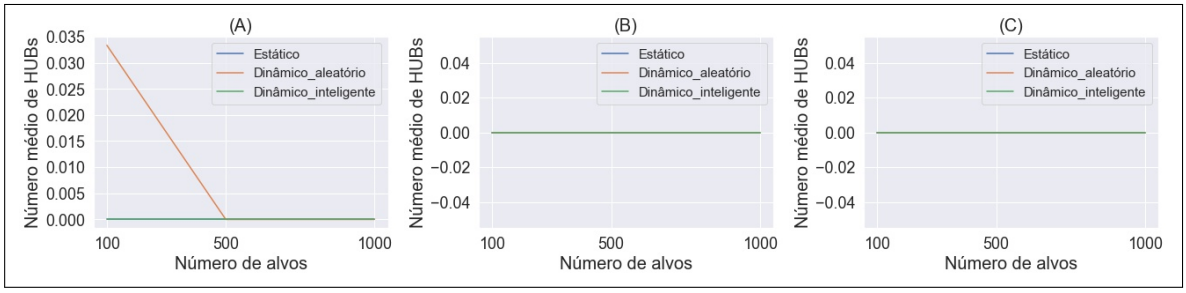
Figura 47 – Quantidade média de HUBs utilizadas nos testes em regiões de 10000 m^2**.**



Fonte: Elaborada pelo autor.

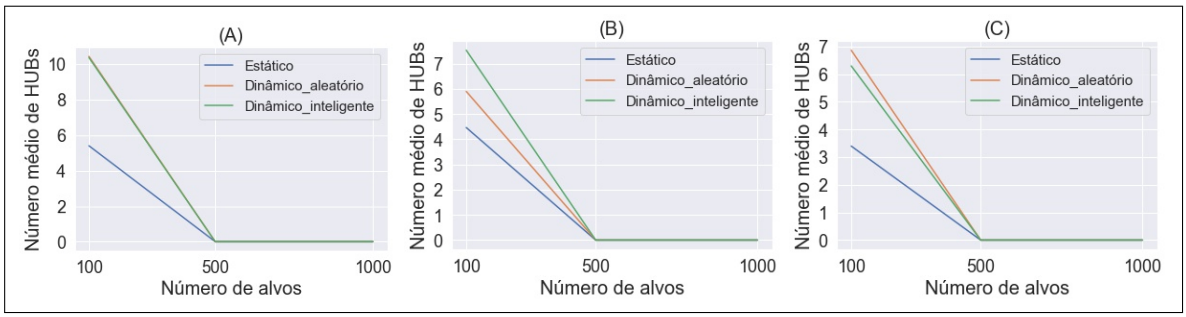
A ideia de estabelecer o número de UAVs máximo igual a quantidade de alvos da instancia de teste foi para investigar se, dentro das instancias de testes, ocorreria algum caso

Figura 48 – Quantidade média de HUBs utilizadas nos testes em regiões de $250000\ m^2$**.**



Fonte: Elaborada pelo autor.

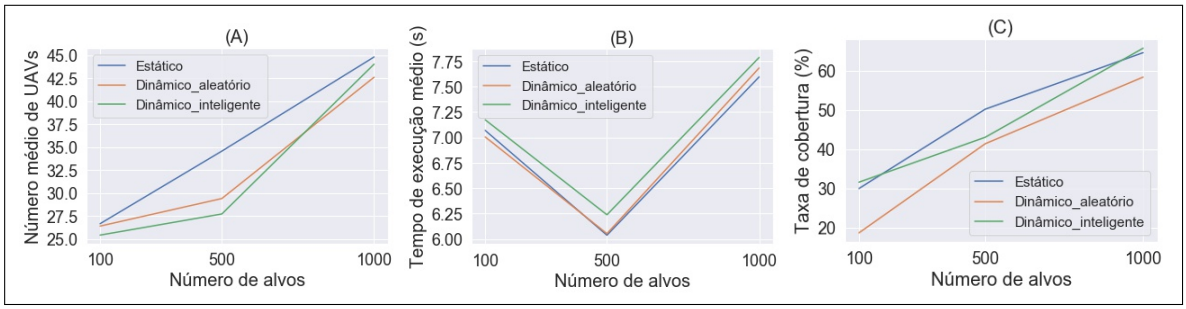
Figura 49 – Quantidade média de HUBs utilizadas nos testes em regiões de $1000000\ m^2$**.**



Fonte: Elaborada pelo autor.

dos algoritmos posicionarem 1 UAV por alvo, o que não ocorreu. Porém, para ilustrar melhor a influencias dos fatores citados acima quanto ao tempo de execução, mais uma sequencia de testes foi realizada, desta vez limitando a quantidade de UAVs para o valor de drones encontrado pelos algoritmos para a respectiva instancia. As Figuras 50 – 52 mostram o resultados desses novos testes.

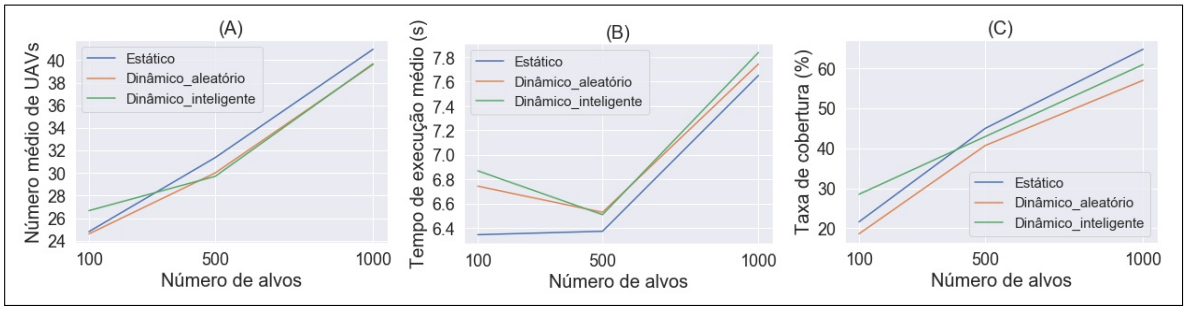
Figura 50 – Resultados para os algoritmos propostos aplicados em uma região de $1000000\ m^2$ **e com a estação base posicionada na origem (0,0), fixando a quantidade de UAVs máxima.**



Fonte: Elaborada pelo autor.

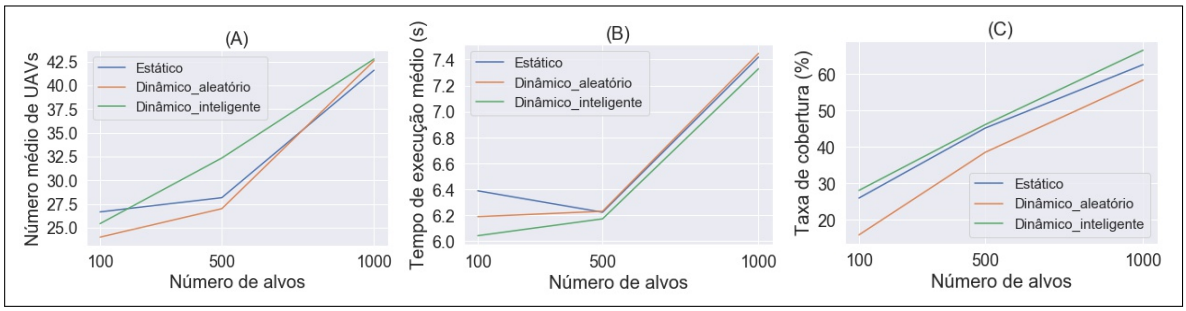
Os resultados comprovam que a quantidade de UAVs máxima influencia mais que a

Figura 51 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 e com a estação base posicionada no meio, fixando a quantidade de UAVs máxima.



Fonte: Elaborada pelo autor.

Figura 52 – Resultados para os algoritmos propostos aplicados em uma região de 1000000 m^2 e com a estação base posicionada aleatoriamente, fixando a quantidade de UAVs máxima.



Fonte: Elaborada pelo autor.

quantidade de alvos, tamanho da região e posicionamento da estação base, pelo menos quanto ao tempo de execução dos algoritmos. Como os valores máximo das UAVs dados como entrada foram os respectivos valores obtidos pelas instancias na execução anterior, os algoritmos puderam reproduzir resultados quanto a quantidade de drones e cobertura de alvos bastante próximos dos obtidos anteriormente.

6.2 RESULTADOS DO ALGORITMO PARA ALVOS MÓVEIS UTILIZANDO A BASE DE DADOS EXTERNA

A Tabela 4 apresenta os resultados do algoritmo para alvos móveis aplicado na base de dados T-Drive trajectory.

Como nas instancias geradas, a posição das UAVs pouco importou quanto aos resultados, que foram bastante próximos entre si. A quantidade de alvos (10357 táxis) e de

Tabela 4 – Resultados do algoritmo de alvos móveis com relação a base de dados externa.

Posição da estação base	Total de UAVs	HUBs	Porcentagem de cobertura (%)	Tempo de execução (s)
Origem	129	0	65,22	2940,67
Meio	123	0	62,82	2926,87
Aleatória	127	0	64,36	3031,03

Fonte: Elaborada pelo autor.

pontos de trajetória (cerca de 1450 pontos em média por táxi) muito superior a quantidade utilizada nas instancias geradas contribuíram para o tempo relativamente elevado de execução, com relação ao obtido mesmo nos piores casos dos cenários gerados.

Independendo da posição da estação base, o posicionamento encontrado para os drones garantiu a observação de em média 64% dos alvos. E, a ausência do uso de HUBs é esperada dado os resultados da subseção anterior, visto que em uma situação com a região de mesma área (1000000 m^2), o algoritmo para alvos móveis conseguiu posicionar UAVs de forma que mais da metade dos alvos foram cobertos e grafo de comunicação de C ficasse conexo, com 1000 alvos, que representa uma caso bem menos denso.

Da mesma forma que na subseção anterior, os testes com a base de dados externa foram refeitos, limitando a quantidade máxima de UAVs para os valores encontrados na Tabela 4. A Tabela 5 mostra os novos resultados.

Tabela 5 – Resultados do algoritmo de alvos móveis com relação a base de dados externa, limitando a quantidade de UAVs máxima.

Posição da estação base	Total de UAVs	HUBs	Porcentagem de cobertura (%)	Tempo de execução (s)
Origem	119	0	62,26	324,76
Meio	114	0	60,99	213,87
Aleatória	121	0	62,88	276,29

Fonte: Elaborada pelo autor.

Os conclusões foram as mesmas da sessão anterior, onde é evidenciado que a quantidade de UAVs máxima dado como entrada influencia o tempo de execução mais que qualquer outro parâmetro.

7 CONCLUSÃO

O problema de cobertura de alvos por UAVs estabelece que eles devem ser posicionados de modo que o maior número de alvos na região seja coberto usando o menor número possível de drones. Além disso, a conectividade entre os UAVs e a estação base deve ser garantida. Assim, foi proposto neste trabalho a concepção de algoritmos para alocação de drones em uma região de interesse, cobrindo o maior número possível de alvos naquela região, com auxilio de métodos de agrupamento e de teoria dos grafos. Foram dois algoritmos propostos, um específico para alvos fixos e outro para alvos móveis.

Para os cenários considerados, os resultados foram bastante promissores, onde dadas as restrições de conectividade, os algoritmos foram capazes de posicionar UAVs de forma que a maior parte dos alvos foram cobertos. Nos cenários gerados, mesmo nos piores casos com regiões mais esparsas, a taxa de cobertura dos alvos permaneceu próximo ou superior a 60%, e o mesmo o ocorre para os resultados com a base de dados externa. Essa taxa é equivalente ou superior a boa parte dos trabalhos da literatura considerados, como os trabalhos de Iellamo, Lehtomaki e Khan (2017), Strumberger *et al.* (2017) e Huang e Savkin (2018).

7.1 CONTRIBUIÇÕES DO TRABALHO

Neste trabalho foram apresentados dois algoritmos para posicionamento de UAVs em uma determinada região, de forma que a maioria dos alvos a serem observados sejam cobertos, e que a rede de comunicação entre UAVs e estação base seja conexa após o posicionamento de forma automatizada. Os algoritmos são, respectivamente, específicos para alvos estáticos e para alvos móveis.

Os algoritmos utilizam método de agrupamento para calcular os melhores posicionamentos para os drones, e visualizam a rede de comunicação entre drones e estação base como uma estrutura de grafo, o que facilitou na aplicação de abordagens para verificar e corrigir a conectividade dessa estrutura de comunicação.

7.2 LIMITAÇÕES

A principal limitação deste trabalho foi a ausência de algoritmos da literatura para comparar com os métodos propostos neste trabalho. Isso ocorre visto que não foram encontradas outras abordagens que encontrem posicionamentos de UAVs de forma automatizada, para a situ-

ação de haver apenas 1 estação base e que garantissem a conectividade da rede de comunicação entre drones e estação base. A maioria das abordagens encontradas resolve apenas uma parte do problema abordado, e o método proposto muitas vezes necessita da interferência e ajustes durante o procedimento.

7.3 TRABALHOS FUTUROS

Para trabalhos futuros, pretende-se utilizar generalizar os testes para outro tipos de drones fora do contexto de observação. Pretende-se considerar o custo de implantação de forma mais ativa para o posicionamento das UAVs. Também deseja-se considerar a otimização da altura, assim como em Caillouet, Giroire e Razafindralambo (2018), além do posicionamento dos drones.

E, por fim, tem-se intenção de elaborar algoritmos de agrupamento que sejam mais apurados e específicos para o problema de alocação de drones, de forma que apenas esse algoritmo seja necessário para o agrupamento de alvos.

REFERÊNCIAS

AGGARWAL, Charu. **Data classification**: algorithms and applications. Nova Iorque: CRC Press, 2014. 707 p.

AMOROSI, Lavinia et al. Energy-efficient mission planning of UAVs for 5G coverage in rural zones. *In:* INTERNATIONAL CONFERENCE ON ENVIRONMENTAL AND ELECTRICAL ENGINEERING, 18., 2018, Milão. **Proceedings** [...] Milão: IEEE, 2018. p. 1-9.

BARR, Nicholas. **Economics of the Welfare state**. Oxônia: OUP Oxford, 2012. 386 p.

BASA, Gungor. **Intelligent agents**. [2018]. Disponível em: <https://gungorbasa.com/intelligent-agents-dc5901daba7d>. Acesso em: 14 out. 2019.

BONDY, Adrian; MURTY, Uppaluri Siva Ramachandra. **Graph theory**. 244. ed. Londres: Springer, 2008. 676 p.

BOURKE, Paul. **Calculating the area and centroid of a polygon**. [1988]. Disponível em: https://www.seas.upenn.edu/~sys502/extra_materials/Polygon%20Area%20and%20Centroid.pdf. Acesso em: 14 dez. 2019.

BRADEN, Bart. The Surveyor's area formula. **The college mathematics journal**, Washington, v. 17, n. 4, p. 326-337, set. 1986.

BRITO, Fernando. **As sete pontes de Königsberg**. [2015]. Disponível em: http://matematicacomge.blogspot.com/2015/01/as-sete-pontes-de-konigsberg.html. Acesso em: 13 maio 2019.

CAILLOUET, Christelle; GIROIRE, Frédéric; RAZAFINDRALAMBO, Tahiry. Optimization of mobile sensor coverage with UAVs. *In:* INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATIONS, 3., 2018, Honolulu. **Proceedings** [...] Honolulu: IEEE, 2018. p. 622-627.

CAILLOUET, Christelle; RAZAFINDRALAMBO, Tahiry. Efficient deployment of connected unmanned aerial vehicles for optimal target coverage. *In:* GLOBAL INFORMATION INFRASTRUCTURE SYMPOSIUM, 10., 2017, Saint-Pierre. **Proceedings** [...] Saint-Pierre: IEEE, 2017. p. 1-8.

CHAUHAN, Darshan; UNNIKRISHNAN, Avinash; FIGLIOZZI, Miguel. Maximum coverage capacitated facility location problem with range constrained drones. **Transportation research part c**: emerging technologies, Amsterdã, v. 99, n. 1, p. 1-18, fev. 2019.

CORMEN, Thomas H. et al. **Introduction to algorithms.** 3. ed. Boston: The MIT press, 2009. 1292 p.

ESTER, Martin et al. A density-based algorithm for discovering clusters in large spatial databases with noise. *In:* INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 2., 1996, Munique. **Proceedings** [...] Munique: ACM, 1996. p. 226-231.

EULER, Leonhard. Solutio problematis ad geometriam situs pertinentis. **Commentarii academiae scientiarum petropolitanae.** São Petersburgo, v. 8, n. 53, p. 128-140. ago. 1741.

GOLDBARG, Marco Cesar; GOLDBARG, Elizabeth. **Grafos**: conceitos, algoritmos e aplicações. Rio de Janeiro: Elsevier, 2012. 640 p.

GREEN, Peter; SILVERMAN, Bernard. **Nonparametric regression and generalized linear models**: a roughness penalty approach. Londres: Chapman and Hall, 1993. 184 p.

GUPTA, Alind. **Elbow method for optimal value of k in kmeans.** 2019. Disponível em: https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/. Acesso em: 23 dez. 2019.

HAGBERG, Aric; SWART, Peter; SCHULT, Daniel. Exploring network structure, dynamics, and function using networkx. *In:* SCIENTIFIC COMPUTING WITH PYTHON CONFERENCE, 8., 2008, Pasadena. **Proceedings** [...] Pasadena: Los Alamos, 2008. p. 1-5.

HARJU, Tero. **Lecture notes on graph theory.** Scotts Valley: Createspace independent publishing platform, 2014. 100 p.

HUANG, Hailong; SAVKIN, Andrey. An algorithm of efficient proactive placement of autonomous drones for maximum coverage in cellular networks. **IEEE wireless communications letters.** Piscataway, p. 994-997. jun. 2018.

HUNTER, John et al. **Matplotlib**: visualization with python. [2012]. Disponível em: https://matplotlib.org/. Acesso em: 20 jul. 2019.

IELLAMO, Stefano; LEHTOMAKI, Janne J.; KHAN, Zaheer. Placement of 5g drone base stations by data field clustering. *In:* VEHICULAR TECHNOLOGY CONFERENCE, 85., 2017, Sydney. **Proceedings** [...] Sydney: IEEE, 2017. p. 1-5.

INABA, Mary; KATOH, Naoki; IMAI, Hiroshi. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering. *In:* THE ANNUAL SYMPOSIUM ON COMPUTATIONAL GEOMETRY, 10., 1994, Nova Iorque. **Proceedings** [...] Nova Iorque: ACM, 1994. p. 332-339.

INSITU. **Insitu**: a boeing company. [2018]. Disponível em: https://www.insitu.com/. Acesso em: 14 jun. 2019.

JAIN, Anil. Data clustering: 50 years beyond k-means. **Pattern recognition letters**, Amsterdã, v. 31, n. 8, p. 651-666, jun. 2010.

KETCHEN, David; SHOOK, Christopher. The application of cluster analysis in strategic management research: an analysis and critique. **Strategic management journal**, Hoboken, v. 17, n. 6, p. 441-458, jun. 1996.

KHAN, Mouhyemen et al. Mobile target coverage and tracking on drone-be-gone uav cyber-physical testbed. **IEEE systems journal**, Piscataway, v. 12, n. 4, p. 3485-3496, dez. 2018.

LI, Deyi; DU, Yi. **Artificial intelligence with uncertainty.** Orlando: CRC Press, 2007. 310 p.

LI, Junzhi; TAN, Ying. The bare bones fireworks algorithm: a minimalist global optimizer. **Applied soft computing**, Amsterdã, v. 62, p. 454-462, jan. 2018.

LI, Xiaohui. Deployment of drone base stations for cellular communication without apriori user distribution information. *In:* CHINESE CONTROL CONFERENCE, 37., 2018, Wuhan. **Proceedings** [...] Wuhan: IEEE, 2018. p. 7274-7281.

MA, Ping; HUANG, Jianhua Z.; ZHANG, Nan. Efficient computation of smoothing splines via adaptive basis sampling. **Biometrika**, Oxônia, v. 102, n. 3, p. 631-645, jun. 2015.

MACQUEEN, James et al. Some methods for classification and analysis of multivariate observations. *In:* BERKELEY SYMPOSIUM ON MATHEMATICAL STATISTICS AND PROBABILITY, 5., 1967, Berkeley. **Proceedings** [...] Berkeley: University of California Press, 1967. p. 281-297.

MADEIRA, Daniel. **A centróide de um polígono.** [2009]. Disponível em: http://dan-scientia.blogspot.com/2009/10/centroide-de-um-poligono.html. Acesso em: 13 out. 2019.

MAMMADOV, Elchin; GUEAIEB, Wail. Long-range communication framework for multi-agent autonomous UAVs. *In:* INTERNATIONAL CONFERENCE ON UNMANNED AIRCRAFT SYSTEMS, 3., 2014, Orlando. **Proceedings** [...] Orlando: IEEE, 2014. p. 26-29.

MCGOLDRICK, Ciaran; SHIVARAM, Smita; HUGGARD, Meriel. Experiences of integrating uavs into the curriculum through multidisciplinary engineering projects. *In:* ANNUAL CONFERENCE AND EXPOSITION, 123., 2016, Nova Orleans. **Proceedings** [...] Nova Orleans: American Society for Engineering Education, 2016. p. 26-29.

MELO, Priscila. **Polígonos.** [2014]. Disponível em: https://www.estudokids.com.br/poligonos/. Acesso em: 14 dez. 2019.

NAIK, Azad. **Data clustering algorithms.** [2010]. Disponível em: https://sites.google.com/site/dataclusteringalgorithms/. Acesso em: 23 jun. 2019.

NUMPY DEVELOPERS. **Numpy.** [2018]. Disponível em: http://www.numpy.org/. Acesso em: 4 out. 2019.

OTTO, Alena et al. Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: a survey. **Networks**, Hoboken, v. 72, n. 4, p. 411-458, 25 mar. 2018.

PARETO, Vilfredo. **Cours d'économie politique.** Genebra: Librairie Droz, 1964. 424 p.

PARK, Jaehyun. **Network flow problems.** [2015]. Disponível em: https://web.stanford.edu/class/cs97si/08-network-flow-problems.pdf. Acesso em: 5 mar. 2019.

PEDREGOSA, Fabian et al. Scikit-learn: machine learning in python. **Jefferson-madison regional library**, Charlottesville, v. 12, n. 1, p. 2826-2830, out. 2011.

PEREZ, Fernando; GRANGER, Brian. **Project jupyter**: computational narratives as the engine of collaborative data science. 2015. Disponível em: http://archive.ipython.org/JupyterGrantNarrative-2015.pdf. Acesso em: 28 fev. 2019.

PINHO, José Luiz Rosas; BATISTA, Eliezer; CARVALHO, Neri Terezinha Both. **Geometria I.** 2. ed. Florianópolis: EAD/UFSC/CED/CFM, 2010. 330 p.

RUSSELL, Stuart; NORVIG, Peter. **Artificial intelligence**: a modern approach. 3. ed. Nova Jersey: Pearson Education, 2010. 1016 p.

SAEED, Ahmed et al. Argus: realistic target coverage by drones. *In:* INTERNATIONAL CONFERENCE ON INFORMATION PROCESSING IN SENSOR NETWORKS, 16., 2017, Pittsburgh. **Proceedings** [...] Pittsburgh: IEEE, 2017. p. 155-166.

SATOPAA, Ville et al. Finding a kneedle in a haystack: detecting knee points in system behavior. *In:* INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS WORKSHOPS, 31., 2011, Minneapolis. **Proceedings** [...] Minneapolis: IEEE, 2011. p. 166-171.

SHANNON, Claude. A mathematical theory of communication. **The bell system technical journal**, Nova Iorque, v. 27, n. 1, p. 379-423, jul. 1948.

STEINHAUS, Hugo. Sur la division des corps materiels en parties. **Bulletin de l'academie polonaise des sciences**, Paris, v. 4, n. 12, p. 801-804, out. 1956.

STRUMBERGER, Ivana et al. Static drone placement by elephant herding optimization algorithm. *In:* TELECOMMUNICATION FORUM, 25., 2017, Belgrade. **Proceedings** [...] Belgrade: IEEE, 2017. p. 1-4.

THORNDIKE, Robert. Who belongs in the family? **Psychometrika**, Berlim, v. 18, n. 4, p. 267-276, dez. 1953.

TISUE, Seth; WILENSKY, Uri. Netlogo: a simple environment for modeling complexity. *In:* INTERNATIONAL CONFERENCE ON COMPLEX SYSTEMS, 2., 2004, Boston. **Proceedings** [...] Boston: ICCS, 2004. p. 16-21.

TUBA, Eva et al. Efficient drone placement for wireless sensor networks coverage by bare bones fireworks algorithm. *In:* INTERNATIONAL SYMPOSIUM ON DIGITAL FORENSIC AND SECURITY, 6., 2018, Antalya. **Proceedings** [...] Antalya: IEEE, 2018. p. 1-5.

VALAVANIS, Kimon; VACHTSEVANOS, George. **Handbook of unmanned aerial vehicles.** Heidelberg: Springer, 2015. 3022 p.

WANG, Gai-ge; DEB, Suash; COELHO, Leandro. Elephant herding optimization. *In:* INTERNATIONAL SYMPOSIUM ON COMPUTATIONAL AND BUSINESS INTELLIGENCE, 3., 2015, Bali. **Proceedings** [...] Bali: IEEE, 2015. p. 1-5.

WONG, Angi Ma; HARTIGAN, John. Algorithm as 136: a k-means clustering algorithm. **Applied statistics**, Hoboken, v. 28, n. 1, p. 100-108, jan. 1979.

XU, Jun et al. **Internet of things applications:** animal monitoring with unmanned aerial vehicle. [2016]. Disponível em: https://arxiv.org/pdf/1610.05287.pdf. Acesso em: 15 maio 2019.

YUAN, Jing et al. Driving with knowledge from the physical world. *In:* INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, 17., 2011, São Diego. **Proceedings** [...] São Diego: ACM, 2011. p. 316-324.

YUAN, Jing et al. T-drive: driving directions based on taxi trajectories. *In:* SIGSPATIAL INTERNATIONAL CONFERENCE ON ADVANCES IN GEOGRAPHIC INFORMATION SYSTEMS, 18., 2010, Nova Iorque. **Proceedings** [...] Nova Iorque: ACM, 2010. p. 99-108.

ZHAO, Patrick et al. Detecting hotspots from taxi trajectory data using spatial cluster analysis. *In:* INTERNATIONAL WORKSHOP ON SPATIOTEMPORAL COMPUTING, 10., 2015, Fairfax. **Proceedings** [...] Fairfax: ISPRS, 2015. p. 131-135.

ZHENG, Yu; XIE, Xing; MA, Wei-ying. Geolife: a collaborative social networking service among user, location and trajectory. **Data engineering bulletin**, Redmond, v. 33, n. 2, p. 32-39, jun. 2010.

ZORBAS, Dimitrios; PUGLIESE, Luigi di Puglia; RAZAFINDRALAMBO, Tahiry; GUERRIERO, Francesca. Optimal drone placement and cost-efficient target coverage. **Journal of network and computer applications**, Nova Iorque, v. 75, n. 3, p. 16-31, nov. 2016.