UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE DESENVOLVIMENTO SUSTENTÁVEL DO SEMIÁRIDO
UNIDADE ACADÊMICA DE ENGENHARIA DE PRODUÇÃO
CURSO DE ENGENHARIA DE PRODUÇÃO

IGGOR THADEU LAGOS DE SOUZA

ABORDAGENS EXATAS PARA O PROBLEMA DO CAIXEIRO VIAJANTE DUBINS COM VIZINHANÇAS

SUMÉ - PB
2023

IGGOR THADEU LAGOS DE SOUZA

ABORDAGENS EXATAS PARA O PROBLEMA DO CAIXEIRO VIAJANTE DUBINS COM VIZINHANÇAS

Monografia apresentada ao Curso Superior de Engenharia de Produção do Centro de Desenvolvimento Sustentável do Semiárido da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Engenharia de Produção.

Orientador: Professor Dr. Yuri Laio Teixeira Veras Silva.

SUMÉ - PB
2023

S729a Souza, Iggor Thadeu Lagos de.
Abordagens exatas para o Problema do Caixeiro Viajante Dubins com Vizinhanças. / Iggor Thadeu Lagos de Souza. - 2023.

55f.

Orientador: Professor Dr. Yuri Laio Teixeira Veras Silva.
Monografia - Universidade Federal de Campina Grande; Centro de Desenvolvimento Sustentável do Semiárido; Curso de Engenharia de Produção.

1. Pesquisa operacional. 2. Programação linear. 3. Problema de otimização de trajetória. 4. Problema do Caixeiro Viajante Dubins com Vizinhança. 5. Roteamento de veículos. 6. Veículo Dubins. 7. Decomposição generalizada de Benders. 8. Algoritmos exatos. 9. Algoritmo Benders generalizado. I. Silva, Yuri Laio Teixeira Veras. II. Título.

CDU: 519.8(043.1)

Elaboração da Ficha Catalográfica:

Johnny Rodrigues Barbosa
Bibliotecário-Documentalista
CRB-15/626

IGGOR THADEU LAGOS DE SOUZA

ABORDAGENS EXATAS PARA O PROBLEMA DO CAIXEIRO VIAJANTE DUBINS COM VIZINHANÇAS

Monografia apresentada ao Curso Superior de Engenharia de Produção do Centro de Desenvolvimento Sustentável do Semiárido da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Engenharia de Produção.

BANCA EXAMINADORA:

Professor Dr. Yuri Laio Teixeira Veras Silva.
Orientador – UAEP/CDSA/UFCG

Professor Dr. Walton Pereira Coutinho.
Examinador Externo I – UFPE

Professor Dr. Luciano Carlos Azevedo da Costa.
Examinador Externo II – UFPB

Trabalho aprovado em: 30 de novembro de 2023.

SUMÉ - PB

RESUMO

Este trabalho aborda o Problema do Caixeiro Viajante Dubins com Vizinhanças, que é uma variante do TSP no qual o veículo é capaz de atender as demandas de cada localidade ao sobrevoar qualquer local pertencente a sua vizinhança e deve respeitar as dinâmicas de movimento do veículo Dubins. Propõe-se a primeira formulação matemática para o DTSPN, baseada em programação inteira-mista, que considera a otimização integrada de rotas e trajetórias. Além disso, é proposto uma formulação baseada no método de decomposição generalizada de Benders no qual o problema é decomposto em um problema mestre de roteamento e um subproblema de otimização de trajetória. A partir dos experimentos realizados, a formulação baseada em programação inteira-mista mostrou-se numericamente intratável e os *solvers* utilizados não foram capazes de encontrar soluções para o problema. Contudo, o método baseado na decomposição do problema foi capaz de encontrar soluções ótimas para as instâncias testadas e com os parâmetros utilizados, comprovando assim a eficiência do algoritmo proposto.

Palavras-chave: TSP, Dubins, Otimização de Trajetórias, Algoritmos Exatos.

ABSTRACT

This work addresses the Dubins Traveling Salesman Problem with Neighborhoods, which is a variant of the TSP in which the vehicle is capable of meeting the demands of each location by flying over any location belonging to its neighborhood and must respect the movement dynamics of the Dubins vehicle. The first mathematical formulation for DTSPN is proposed, based on mixed-integer programming, which considers the integrated optimization of routes and trajectories. Furthermore, a formulation based on the generalized Benders distribution method is proposed in which the problem is decomposed into a master routing problem and a trajectory optimization subproblem. From the experiments carried out, the formulation based on mixed-integer programming proved to be numerically intractable and the *solvers* used were not able to find solutions to the problem. However, the method based on solving the problem was able to find optimal solutions for the tested instances and with the parameters used, thus proving the efficiency of the proposed algorithm.

Palavras-chave: TSP, Dubins, Trajectory Optimization, Exact Algorithms.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Veículos Aéreos Não Tripulados (VANTs) podem ser descritos como aeronaves que podem voar e permanecer no ar sem um piloto humano a bordo. Esses veículos são capazes de executar operações críticas sem por em risco a vida de um ser humano e são normalmente controlados por um computador embutido ou operados remotamente por um piloto utilizando um controle remoto. Os VANTs modernos foram inicialmente desenvolvidos na década de 1920 e aplicados principalmente em missões militares de reconhecimento, vigilância e transporte de armas. VANTs podem executar uma variedade de atividades, como transporte, ataques armados, missões de busca e salvamento. Drones, planadores não tripulados e helicópteros controlados remotamente são exemplos de VANTs.

Os VANTs vêm se tornando cada vez mais populares em áreas como a logística e vigilância. De acordo com dados da Agência Nacional de Aviação Civil (ANAC), em um período de um ano, a partir de agosto de 2018, saltou de 18.389 para 27.665 o número de drones utilizados para fins profissionais no Brasil. Segundo Coutinho, Battarra e Fliege (2018), os VANTs possuem uma ampla gama de aplicações devido a capacidade de anexar diversos sensores, transmissores e equipamentos fotográficos. Casos de sucesso da aplicação de VANTs podem ser visto em diversas áreas como, por exemplo, agricultura (Kim et al., 2019), comercio online (Poikonen, Wang e Golden, 2017), sensoriamento remoto (Klemas, 2015), controle de tráfego (Kanistras et al., 2013), detecção de incêndios florestais (Sousa e Gamboa, 2020).

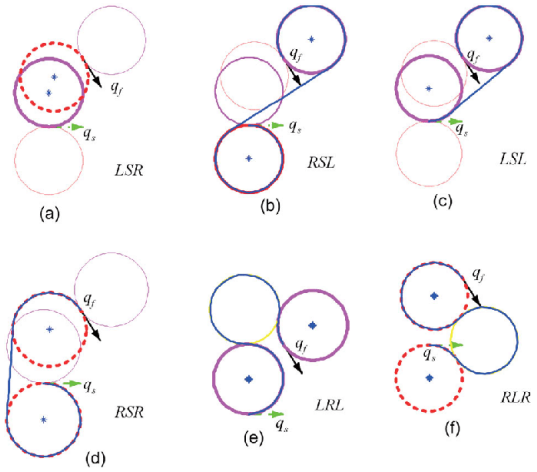
O problema aqui estudado é motivado pelo grande potencial que a utilização de VANTs possui e pelo interesse de empresas e organizações em utilizar VANTs em suas operações para concluir tarefas em um menor intervalo de tempo e com economia. Os VANTs podem ser classificados em diferentes categorias, como mostrado em Elmeseiry, Alshaer e Ismail (2021). Dependendo do modelo utilizado, o veículo irá possuir restrições de movimento que devem ser levadas em consideração na hora do planejamento de trajetórias. No presente trabalho abordamos os drones de asa fixa que são VANTs com essas restrições, possuindo um raio mínimo de curvatura.

Uma forma de abordar o roteamento de veículos é através do Problema do Caixeiro Viajante ou TSP (do inglês *Traveling Salesman Problem*) (Applegate et al., 2007). O TSP consiste em determinar uma rota de menor distância percorrida para um veículo de forma que um conjunto de pontos geograficamente dispersos sejam visitados. Entretanto, o TSP utiliza apenas da distância euclidiana e não leva em consideração as restrições de movimento dos drones de asa fixa.

Dessa forma, para simular as restrições de movimento dos drones de asa fixa, o presente trabalho utiliza do modelo de Dubins (Dubins, 1957). O veículo Dubins consiste em um veículo

que está sempre avançando e possui apenas 3 controles, sendo eles: esquerda (L), direita (R) e seguir em frente (S). Possibilitando 6 diferentes combinações que representam os caminhos mais curtos para um veículo com restrição de curvatura, como mostrado na Figura 1.

Figura 1 – Principais tipos de caminhos de Dubins.



Fonte: Hameed (2017).

O roteamento de veículos Dubins consiste em um problema de roteamento em que o veículo só pode avançar para frente, em velocidade constante e possui um raio de giro limitado. Esse problema pode ser definido como o Problema do Caixeiro Viajante Dubins ou DTSP (do inglês *Dubins Traveling Salesman Problem*) (Ny, Frazzoli e Feron, 2007), uma generalização do TSP. O *Traveling Salesman Problem with Drones* (TSP-D) (Agatz, Bouman e Schmidt, 2015) e o *Flying Sidekick Traveling Salesman Problem* (FS-TSP) (Murray e Chu, 2015) são outras variantes do TSP que consideram a movimentação de drones.

Devido a acoplagem de sensores e equipamentos fotográficos, em aplicações de vigilância e coleta de dados, os VANTs são capazes de coletar as informações necessárias para conclusão de missões a uma distância suficientemente próxima dos alvos sem que seja necessário chegar até o ponto exato de destino. Dessa forma, na literatura, existem diversas variantes do TSP que foram propostas para melhor utilização de recursos como esse dos VANTs, com um raio mínimo ao redor do alvo no qual o veículo pode coletar a informação ou tirar uma foto, de acordo com a missão. Como o Problema do Caixeiro Viajante com Vizinhanças ou TSPN (do inglês *Traveling Salesman Problem with Neighborhoods*) abordado em (Vana e Faigl, 2015).

Diante dessas peculiaridades que os VANTs possuem, Obermeyer (2009) propôs o Problema do Caixeiro Viajante Dubins com Vizinhanças ou DTSPN (do inglês *Dubins Traveling Salesman Problem with Neighborhoods*) para que seja possível aproveitar todos os esses recursos e características. Nesse problema, o veículo se movimenta respeitando as dinâmicas de movimento do veículo Dubins e não precisa visitar a localização exata do alvo, sendo capaz de atender à demanda de cada local quando sobrevoa um ponto pertencente ao conjunto de vizinhanças do alvo. Um caso especial do DTSPN é o Problema de Orientação Suficientemente

Próximo ou CEDOP (do inglês *Close Enough Dubins Orienteering Problem*) (Faigl e Pěnička, 2017) no qual são atribuídos recompensas a cada uma das localidades e busca-se maximizar o premio coletado sem que seja excedido o limite de tempo.

A partir de uma busca sistemática na literatura de roteamento de veículos, identificou-se que nenhum método exato para resolver o DTSPN ou problemas similares que lidam com veículos com restrição de curvatura foi proposto, sendo todos os métodos de resolução utilizados baseados em abordagens heurísticas. Neste contexto, apresenta-se a primeira formulação matemática que considera as rotas e trajetórias simultaneamente para resolução do DTSPN, que é uma formulação de programação inteira-mista. Entretanto, os resultados computacionais demonstram que essa formulação matemática não é numericamente tratável, ou seja, não foi possível encontrar soluções para o problema utilizando os *solvers* disponíveis. Diante disso, desenvolveu-se um método baseado na decomposição do problema em um problema mestre de roteamento e um subproblema de otimização de trajetória. Para resolver o problema mestre, é utilizado o método *branch-and-cut* baseado em algoritmos já conhecidos para o problema do TSP. O subproblema de otimização de trajetória foi resolvido utilizando um método exato baseado em colocação direta (Betts, 2001). Diante disso, o presente trabalho trás as seguintes contribuições:

- Na literatura, encontra-se uma escassez de instâncias para o DTSPN pois diversos artigos não disponibilizam as instâncias utilizadas em suas pesquisas. Por isso, aborda-se a falta de dados para o DTSPN através da criação e disponibilização de um banco de dados e de um gerador de instâncias;
- Diversos trabalhos abordam o DTSPN, entretanto, ainda não foi proposto uma formulação que aborde o problema em sua forma integrada. Diante disso, propõe-se a primeira formulação matemática para o DTSPN, que considera a otimização integrada de rotas e trajetórias;
- A formulação matemática proposta se mostrou intratável, tendo em vista que os *solvers* disponíveis não foram capazes de encontrar soluções viáveis para o problema. Dessa maneira, apresenta-se o primeiro método exato, baseado no método de *branch-and-cut*, capaz de resolver instâncias do DTSPN, de pequeno e médio porte, na otimalidade, em tempo computacional aceitável e níveis altos de acurácia.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Desenvolver uma formulação matemática e um algoritmo Benders generalizado para resolução do DTPSN.

1.1.2 Objetivos Específicos

Para alcançar o objetivo geral, faz-se necessário atender os seguintes objetivos específicos:

- Realizar uma revisão da literatura sobre o roteamento de veículos e o DTSPN;
- Desenvolver uma formulação matemática baseada em otimização de trajetórias para o DTSPN;
- Desenvolver um algoritmo baseado na decomposição de Benders generalizada para o DTSPN;
- Implementar o modelo e algoritmo propostos para realização de experimentos computacionais;
- Desenvolver conclusões e ideias a partir dos resultados computacionais encontrados.

1.2 JUSTIFICATIVA

Com os recentes avanços tecnológicos dos VANTs, empresas de diversos segmentos passaram a investir na fabricação e desenvolvimento de tecnologias de drones que potencializem sua aplicabilidade em seus respectivos setores. As tendências tecnológicas e os avanços no controle, miniaturização e informatização levam a VANTs mais seguros, leves, acessíveis e econômicos. Entretanto, os VANTs ainda enfrentam limitações na operacionalidade em termos de autonomia de voo, planejamento de trajetória, aspectos de segurança aérea e custos operacionais. Diversas aplicações de VANTs em áreas como agricultura, comércio online, sensoriamento remoto, controle de tráfego, detecção de incêndios florestais, já foram bem sucedidas.

Segundo Brasil e Pansonato (2018), na área de logística, o transporte de mercadorias representa uma porção considerável dos custos de uma empresa. Um exemplo da aplicação no setor logístico foi a abordagem de combinar VANTs e caminhões para realizar atividades de distribuição em grandes centros, onde drones são despachados de caminhões para realizar os últimos quilômetros de uma entrega. Dessa forma, a realização de um roteamento de veículos e de um mapeamento da melhor rota a ser tomada durante um processo logístico tem o potencial de minimizar o tempo gasto, a distância percorria e consequentemente reduzir os custos, gerando benefícios para a empresa.

De acordo com Coutinho, Battarra e Fliege (2018), a comunidade acadêmica de roteamento reconheceu o interesse de empresas e organizações na adoção de VANTs em suas operações. Entretanto, as literaturas de roteamento de veículos e atribuição de tarefas de VANTs muitas vezes negligenciam restrições devido à dinâmica de voo dos VANTs.

Como será mostrado na revisão bibliográfica, na literatura de roteamento de veículos aplicada a veículos com restrições de movimento, ainda não foi proposto um método de resolução exato para o DTSPN. Em razão disso, ao propor uma formulação matemática e desenvolver algoritmos que levam em consideração as restrições de movimentos do veículo, utilizando métodos exatos para sua resolução, além de disponibilizar um banco de dados com instâncias para o DTSPN, o presente trabalho pretende preencher um gap na literatura de roteamento de veículo com restrições de movimento.

2 REFERENCIAL TEÓRICO

2.1 PESQUISA OPERACIONAL

A Pesquisa Operacional (PO) faz a utilização de técnicas com embasamento lógico-cientifico para resolução de problemas e auxiliar na tomada de decisões. Segundo Hillier (2013), a origem da PO se dá com a transição adenta da Revolução Industrial e o grande crescimento no tamanho e complexidade das organizações. A PO é capaz de prover informações relevantes à tomada de decisão, tornando-a mais prática e eficiente para organizações e seus tomadores de decisão. A construção de novos modelos e a capacidade de resolver situações cada vez mais complexas proporciona uma aproximação cada vez maior dos eventos reais estudados e mais confiabilidade aos resultados encontrados.

De acordo com Goldbarg e Luna (2000), os modelos de PO são estruturados de forma lógica e suportados por ferramentas de representação matemática destinadas a determinar as condições ótimas de operação do sistema representado. Os principais modelos de PO são denominados de Programação Matemática e constituem uma das mais importantes variedades dos modelos quantitativos.

As técnicas de solução da modelagem matemática são agrupadas em subáreas como a Programação Linear (PL) que é utilizada para modelos com funções objetivas que possuem restrição lineares e que, segundo Taha (2016), é a área mais estudada da PO. Outras áreas incluem a Programação Inteira (PI), na qual as variáveis assumem valores inteiros, implicando maior complexidade computacional, a Programação Dinâmica (PD), em que o modelo original é decomposto em subproblemas menores e mais gerenciáveis, Programação Não Linear (PNL), na qual a função objetivo ou qualquer uma das restrições do modelo são não lineares.

2.1.1 Programação Linear

A PL consiste na representação das características de um problema em forma de um conjunto de equações lineares para descrever um problema real. A palavra programação deve ser vista como um sinônimo de planejamento, enquanto o adjetivo linear evidencia que todas as funções matemáticas utilizadas no modelo devem ser necessariamente funções lineares. A programação linear envolve o planejamento de atividades para obter-se, entre todas as alternativas viáveis, um resultado capaz de atingir o melhor objetivo de acordo com as especificações da situação e do modelo que está sendo trabalhado.

Um problema de PL pode ser matematicamente definido da seguinte forma:

$$Z = c_1 x_1 + c_2 x_2 + ... + c_n x_n \tag{2.1}$$

s.a

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n < b_1, \\ a_{21}x_2 + a_{22}x_2 + ... + a_{2n}x_n < b_2, \\ . \\ . \\ . \\ a_{m1}x_m + a_{m2}x_m + ... + a_{mn}x_n < b_m, \end{aligned} \tag{2.2}$$

Onde: $x_i \geq 0$ e $b_j \geq 0$, para $i = 1, 2, ..., n$ e $j = 1, 2, ..., m$. Nessa formulação, (2.1) representa a função objetivo do problema que deve ser maximizada ou minimizada de acordo com o problema. (2.2) são equações que expressam as restrições do problema.

Para Hillier (2013), o desenvolvimento da programação linear pode ser classificado entre os mais importantes avanços científicos dos meados do século XX e seu impacto tem sido extraordinário. A grande maioria de toda a computação científica realizada em computadores é dedicada a utilização da programação linear. Sua aplicação em diversos setores da sociedade se espalhou rapidamente e desde o seu surgimento já poupou milhares de dólares para negócios de médio porte e grandes empresas em diversos países industrializados ao redor do mundo.

A alocação de recursos para atividades é o tipo de aplicação mais comum da programação linear, entretanto, qualquer problema cujo modelo matemático se encaixe no formato genérico do modelo de programação linear, é um problema que pode ser resolvido utilizando da programação linear. Segundo Dantzig (1997), as primeiras aplicações da programação linear no setor industrial foram para a indústria do petróleo, incluindo a extração, refinamento, mistura e distribuição do petróleo, com a indústria de processamento de alimentos sendo o segundo maior usuário da programação linear, com sua primeira aplicação no setor para atividades de distribuição de produtos.

A vasta aplicabilidade da programação linear em conjunto com o método simplex que é um procedimento de resolução extremamente eficiente e que encontra-se disponível para solucionar até mesmo problemas de programação linear de enormes dimensões, são algumas das razões para o seu extraordinário impacto em décadas recentes.

2.1.2 Programação Não Linear

Em problemas complexos de otimização, a função objetivo e/ou as restrições são expressas por funções não lineares. Tais problemas são chamados de problemas de programação não linear (PNL). Segundo Winston (2012), o mundo real muitas vezes se comporta de maneira não linear, então ao modelar um problema com programação linear, você está normalmente fazendo uma aproximação da realidade. Ao permitir não linearidades nos modelos, muitas vezes é possível criar modelos mais realistas. Entretanto, problemas de PNL são mais difíceis de resolver.

Em geral, um problema de PNL pode ser matematicamente definido da seguinte forma:

$$f(x_1, x_2, ..., x_n) \tag{2.3}$$

s.a

$$\begin{aligned} g_1(x_1, x_2, ..., x_n) &\leq b_1, \\ g_2(x_1, x_2, ..., x_n) &\leq b_2, \\ &\cdot \\ &\cdot \\ &\cdot \\ g_m(x_1, x_2, ..., x_n) &\leq b_m, \end{aligned} \tag{2.4}$$

onde cada uma das restrições g_1 até g_m é dada. A equação 2.3 representa a função objetivo do problema que deve ser maximizada ou minimizada de acordo com o problema.

Para Luenberger e Ye (2015), a PNL pode ser dividida em duas parte, Problemas Irrestritos e Problemas Restritos que quando juntas compreendem o assunto de Programação Não Linear. Apesar de aparentar que os problemas de otimização irrestrita não possuem propriedades estruturais, comprometendo a sua aplicabilidade como modelos úteis para problemas significativos, é possível expor que essa afirmação não é verdadeira por dois motivos. O primeiro deles, é que pode-se argumentar, convincentemente, que ao ampliar o escopo de um problema para considerar todas as variáveis relevantes de decisão, chega-se a um ponto onde não haverá restrições, Luenberger e Ye (2015) explicam que isso ocorre porque as restrições representam delimitações artificiais de escopo e, quando o escopo é ampliado, as restrições desaparecem. Os autores dão o exemplo de um Problema Orçamentário, argumentando que uma restrição orçamentária não é característica de uma formulação de problema significativo, já que ao pegar um empréstimo a dada taxa de juros, é possível obter fundos adicionais e, portanto, um termo que reflita o custo dos fundos deve ser incorporada ao objetivo, ao invés de uma restrição orçamentária.

O segundo motivo pelo qual diversos problemas importantes podem ser considerados como problemas irrestritos, é que as vezes podemos facilmente converter um problema restrito em irrestrito. Por exemplo, as restrições de igualdade tem a função de limitar os graus de liberdade, fazendo essencialmente com que algumas variáveis se tornem funções de outras. As vezes um novo problema com seu número de variáveis igual ao verdadeiro grau de liberdade pode ser determinado ao caracterizar explicitamente essas dependências. Além de representar uma classe significativa de problemas práticos, o estudo de problemas irrestritos fornecem o acesso a casos mais gerais de problemas restritos.

Apesar dos argumentos dados acima, muitos problemas encontrados na prática são formulados como problemas restritos. De acordo com Luenberger e Ye (2015), isso ocorre porque, na maioria dos casos, problemas complexos como, por exemplo, a política de produção deta-

lhada de uma corporação gigante, o planejamento de uma grande agência governamental ou até mesmo o projeto de um dispositivo complexo deve ser decomposto em subproblemas separados, com cada subproblema possuindo restrições que são impostas para restringir seu alcance já que não é possível contabilizar todas as escolhas possíveis desses problemas e tratar-los diretamente em sua totalidade. Portanto, em problemas de planejamento, as restrições orçamentárias são muitas vezes impostas para separar o problema que está sendo trabalhado de um problema mais global. Assim, frequentemente encontramos problemas gerais de programação matemática não linear com restrições.

2.2 PROGRAMAÇÃO INTEIRA MISTA

Em diversos problemas práticos, as variáveis de decisão apenas fazem sentido se tiverem valores inteiros. Por exemplo, problemas de alocação de pessoal, máquinas e veículos para atividades que precisam de quantidades inteiras. A Programação Inteira é uma variação da programação linear, onde o modelo possui uma restrição que impede a divisibilidade das variáveis de decisão, isto é, exige que as variáveis assumam valores inteiros. Sendo assim, se a exigência de valores inteiros for a única característica pela qual o problema se afaste da formulação de um problema de programação linear, então temos um problema de programação inteira. Quando temos um problema no qual parte das variáveis são inteiras e as variáveis restantes podem assumir quaisquer valores em um intervalo de números reais, ou seja, são continuas e a hipótese da divisibilidade permanece válida, o modelo é chamado de Programação Inteira Mista.

2.2.1 Programação Inteira mista não linear

Segundo Belotti et al. (2013), Programas não lineares inteiros mistos (MINLP) são problemas de otimização nos quais algumas das variáveis são restritas a assumir valores inteiros enquanto a função objetivo e a região factível do problema são descritas por funções não lineares. As variáveis inteiras são necessárias para modelar relacionamentos lógicos, restrições disjuntivas, funções lineares por partes e a indivisibilidade de recursos. Já as funções não lineares, são necessárias para representar com exatidão as propriedades físicas, economias de escala e covariância. No mundo real, encontra-se diversas aplicações para tais problemas de otimização.

Um Problema de Programação Inteira Mista Não linear possui a seguinte forma:

$$\min \left\{ f^0(x,y) : f^j(x,y) \leq 0 (j = 1, ..., m), x \in \mathbb{Z}_+^{n_1}, y \in \mathbb{R}_+^{n_2} \right\}, \tag{2.5}$$

onde n_1 representa o número de variáveis com restrição de número inteiro, n_2 presenta o número de variáveis continuas, m é o número de restrições e $f^j(x,y)$ para $j = 0, 1, ..., m$ são funções arbitrárias mapeando $\mathbb{Z}_+^{n_1} x \mathbb{R}_+^{n_2}$ para os reais.

Os MINLPs combinam a dificuldade de otimizar variáveis inteiras com o manuseio de funções não lineares, por isso, formam uma classe ampla e desafiadora de problemas de otimização. Ao restringir o modelo para conter somente funções lineares, o MINLP é reduzido a um Programa Linear Inteiro Misto (MILP) que é um problema considerado NP-Difícil. No entanto, ao restringir o modelo para não possuir variável inteira, mas permitir funções não lineares gerais na função objetivo ou nas restrições, o MINLP é reduzido a um Programa Não Linear (NLP) que é igualmente conhecido como NP-Difícil.

Existem dois tipos de MINLPs, convexos e não convexos. Se as funções $f^0, ..., f^m$ são convexas, o MINLP é considerado convexo, em contra partida, se as funções não forem convexas, o MINLP é considerado não convexo. Ambos os tipos de MINLP são considerados N-Difícil, entretanto, os MINLP convexos são mais fáceis de solucionar quando comparados aos não convexos, tanto na teoria quanto na prática.

Sua combinação de casos especiais de programação linear inteira mista e programação não linear torna atraente tanto como uma ferramenta de modelagem flexível para uma ampla gama de acadêmicos e aplicações industriais e uma área de pesquisa rica em desafios computacionais problemas de otimização

2.2.2 Métodos de resolução exatos

Um dos métodos de resolução exatos mais comuns para o MINLP é o Branch-and-Bound (BB). O algoritmo BB foi proposto por Land e Doig (1960) em 1960 e utilizado para resolução do Problema do Caixeiro Viajante. Em virtude da sua capacidade de resolução de problemas práticos que possuem um número finito de soluções viáveis, esse método tem sido bastante utilizado por pesquisadores de diversas áreas.

Em sua forma básica, o BB resolve os problemas MINLP relaxando as restrições inteiras do problema original e resolvendo relaxações PNL continuas. Segundo Kronqvist et al. (2019), resolver um problema de relaxamento continuo (P-MINLP) resulta em uma solução (x^k, y^k), que fornece um limite inferior ($lowerbound$) válido. Se todos os componentes de y^k, assumirem valores inteiros, então também é uma solução ótima para o problema MINLP, caso contrário, a relaxação continua é dividida em dois novos subproblemas PNL adicionando as restrições $y_i \leq \left\lfloor y_i^k \right\rfloor$ e $y_i \geq \left\lceil y_i^k \right\rceil$ ao problema relaxado. A variável de ramificação y_i é uma variável que assume um valor fracionário e é geralmente escolhido com base em alguns critérios, como por exemplo, a variável mais distante de um valor inteiro. Um novo $lowerbound$ pode ser obtido ao resolver os novos subproblemas, caso um dos subproblemas retorne uma solução inteira, essa solução pode ser utilizada como um limite superior ($upperbound$).

Segundo Mitchell (2009a), uma abordagem BB pode se tornar consideravelmente mais rápida ao empregar um esquema de cutting plane, em cada nó da árvore de ramificações ou apenas no topo. Os métodos de Cutting Plane são algorítimos exatos para resolução de PLI.

Conforme Mitchell (2009b), esses métodos trabalham resolvendo uma sequência de relaxações de Programação Linear do problema de Programação Inteira que são melhoradas gradualmente para fornecer melhores aproximações na vizinhança da solução ótima. Em instâncias consideradas difíceis na qual a otimalidade da solução não pode ser garantida, a utilização dos métodos de cutting plane pode produzir soluções próximas da otimalidade em tempos de computação moderados.

Ao combinar os métodos de Branch-and-Bound com os métodos de Cutting Plane, temos os métodos de Branch-and-Cut. De forma simplificada, Mitchell (2009a) explica que esses métodos funcionam resolvendo uma sequência de relaxações de programação linear do problema de programação inteira onde são aplicados os métodos de cutting plane para melhorar o relaxamento do problema de programação inteira e os algoritmos de branch-and-bound seguem uma abordagem sofisticada de dividir e conquistar para resolução dos problemas.

Outro método de resolução exato para problemas MINLP é a Decomposição Generalizada de Benders ou GBD (do inglês *Generalized Benders Decomposition*) proposto por Geoffrion (1997). O GBD é uma abordagem teórica e algorítmica para resolução de problemas MINLP e problemas que exigem a exploração de sua estrutura matemática por meio de princípios de decomposição. A ideia principal do GBD é decompor um problema MINLP em dois subproblemas: um problema primal e um problema mestre. Esses dois problemas são resolvidos iterativamente até que as soluções convergem.

Segundo Floudas (2009), a ideia básica do GBD é a geração, a cada interação, de um $upperbound$, através do problema mestre, e um $lowerbound$, por meio do problema primal, de uma solução do modelo MINLP. O problema primal corresponde ao problema original com as variáveis complicadas (variáveis que, quando fixadas temporariamente, tornam o problema de otimização restante consideravelmente mais tratável) fixas e sua solução fornece informações sobre o $upperbound$ e os multiplicadores de Lagrange associados às restrições de igualdade e desigualdade do problema. O problema mestre é derivado através da teoria da dualidade não linear, faz uso dos multiplicadores de Lagrange obtidos no problema primal, e sua solução fornece informações sobre o $lowerbound$, bem como o próximo conjunto de variáveis complicadas fixas a serem usadas posteriormente no problema primal.

2.2.3 Métodos de resolução heurísticos

Muitas vezes não é possível utilizar métodos exatos para problemas muito complexos e grandes do mundo real, isso ocorre pois esses problemas são geram uma enorme árvore de busca ou devem ser resolvidos em tempo real. Em situações como essas, obter uma solução rapidamente pode ser mais vantajoso do que uma solução ótima, para isso, uma alternativa é recorrer as heurísticas de busca que são capazes de fornecer uma solução viável mas sem garantir a otimalidade. Segundo Belotti et al. (2013), as heurísticas também podem ser utilizadas para acelerar os métodos exatos, identificando rapidamente uma solução candidata com baixo

valor da função objetivo que pode ser utilizada como $upperbound$ para o método exato que está sendo implementado.

Belotti et al. (2013) distinguem as heurísticas de busca em duas classes: busca probabilística e busca determinística. A busca probabilística refere-se a técnicas que requerem uma escolha aleatória de uma solução candidata ou parâmetros que determinam uma solução a cada interação. Esses métodos são simples de projetar e aplicáveis a diversos problemas de otimização, entretanto, demandam implementação e modificações específicas para a estrutura do problema que está sendo trabalhado. Já as heurísticas de busca determinística, podem ser classificadas em heurísticas de busca no qual ocorre a busca de soluções sem a ajuda de quaisquer soluções conhecidas e heurísticas de melhoria que procuram melhorar uma dada solução ou um conjunto de soluções.

Essas heurísticas podem ser classificados em dois tipos: heurísticas de busca, que buscam uma solução sem a ajuda de quaisquer soluções conhecidas, e heurísticas de melhoria, que melhorar uma dada solução ou um conjunto de soluções.

2.3 O PROBLEMA DE OTIMIZAÇÃO DE TRAJETÓRIA

O Problema de Otimização de Trajetória (POT) é uma variante do Problema de Controle Ótimo (PCO) que consiste, segundo Coutinho (2018), em determinar a trajetória de um sistema, por exemplo, veículos como espaçonaves, aeronaves, VANTs, enquanto minimiza uma determinada medida de desempenho e satisfaz um conjunto condições de limite, restrições de caminho e a dinâmica do sistema.

Um POT possui a seguinte forma:

$$\min \phi(y(t_f), t_f) \tag{2.6}$$

s.a.

$$\dot{y} = f(y(t), u(t)) \tag{2.7}$$

$$\psi(y(t_f), u(t_f), t_f) = 0 \tag{2.8}$$

onde o estado do veículo e as variáveis de controle no tempo t são representadas pelos vetores $y(t)$ e $u(t)$, respectivamente. $\dot{y} = f(y(t), u(t))$ representa a dinâmica do veículo. A função objetivo 2.6 busca minimizar uma medida de desempenho sobre o estado do veículo no tempo t_f. Por fim, a restrição 2.8, assume que as condições iniciais do veículo $y(t_o)$ e $u(t_o)$ no tempo inicial $t_o = 0$ são previamente conhecidas.

De acordo com Coutinho (2018), resolver um POT para uma aeronave consiste em gerar as entradas ($inputs$) para o sistema de controle da aeronave para que o conjunto de manobras desejado seja realizado. O POT recebe como $input$ as restrições dinâmicas da aeronave e de-

volve como $output$ uma sequência de estado e controle indexados ao tempo, como velocidades, posições e acelerações.

Ao considerar que as condições de contorno dependem de variáveis desconhecidas ou se a dinâmica dos veículos muda ao longo do tempo, o problema se torna consideravelmente mais difícil. Quando isso ocorre, é possível dividir os POTs em duas ou mais fases para modelar adequadamente as mudanças nas características físicas ou operacionais dos veículos. As fases podem ser descritas por seu próprio sistema de equações diferenciais, condições de contorno, restrições operacionais e eventos de tempo e definidas como um segmento de uma trajetória no qual o sistema dinâmico permanece inalterado. Por fim, todas as fases podem ser ligadas ou não, dependendo do comportamento do sistema dinâmico.

2.3.1 Métodos de resolução diretos e indiretos

Os métodos diretos e indiretos são os dois principais métodos numéricos de resolução dos POTs. Os métodos diretos dependem da discretização de um PCO de dimensão infinita em um problema de otimização de dimensão finita, essa estratégia consiste de discretizar e depois otimizar o problema.

Se levarmos em consideração a formulação do POT representada pelas equações (2.6-2.8), em um método direto de disparo único, as variáveis de controle são discretizadas em uma grade fixa $t_o = t_0 < t_1 < \cdots < t_n = t_f$ usando um esquema arbitrário de parametrização, onde n é o número de passos de discretização:

$$u(t_j, a) \approx \sum_{i=1}^{m} a_i \alpha_i(t_j), j = 1, ..., n. \tag{2.9}$$

As funções $\alpha_i, i = 1, ..., m$ devem ser definidas previamente e os parâmetros $a_i, i = 1, ..., m$ se tornam variáveis de otimização. Os estados correspondentes $y(t_j, a), j = 1, ..., n$, são considerados como variáveis dependentes e podem ser encontradas por integração numérica utilizando os parâmetros de controle $a_i, i = 1, ..., m$. Com isso, é possível garantir a viabilidade da restrição 2.7 para os controles discretizados.

Por fim, o último passo desse método direto consiste em resolver um problema de PNL definido pelas equações (2.10-2.11) para encontrar um vetor ideal de parâmetros $a = (a_1, ..., a_m)$:

$$\min \phi(y(t_n, a), t_n) \tag{2.10}$$

s.a.

$$\psi(y(t_n, a), u(t_n, a), t_n) = 0 \tag{2.11}$$

Os métodos indiretos consistem em determinar as condições de otimização para um problema de CO e, em seguida, utilizar um método de discretização para resolver as equações resultantes. Em oposição aos métodos diretos, nos métodos indiretos é feita a otimização e em seguida a discretização do problema.

Por uma questão de simplicidade, definimos $\phi_{t_f} = \phi(y(t_f), t_f)$ e $\psi_{t_f} = \psi(y(t_f), u(t_f), t_f)$. Se definirmos $\lambda(t)$ e β como sendo os coeficientes lagrangianos das equações 2.7 e 2.8, respectivamente, o relaxamento lagrangiano do problema definido pelas equações (2.6-2.8) pode ser escrito da seguinte forma:

$$\min \phi_{t_f} + \int_{t_o}^{t_f} \lambda(t)^T [f(y(t), u(t)) - \dot{y}]dt + \beta\psi_{t_f} \tag{2.12}$$

Definimos $H = \lambda(t)^T f(y(t), u(t))$ e $\Phi_{t_f} = \phi_{t_f} + \beta\psi_{t_f}$. Por motivos de simplificação, iremos omitir as etapas de substituição e reorganização dos termos. Condições de otimalidade para a equação 2.12 podem ser alcançados se definirmos sua primeira variação como sendo igual a 0. Dessa forma, podemos definir o conjunto de condições necessárias para o problema (2.6-2.8):

$$\frac{\partial \Phi}{\partial y_{t_f}} - \lambda(t_f) = 0 \tag{2.13}$$

$$\frac{\partial H}{\partial y_{(t)}} - \dot{\lambda} = 0 \tag{2.14}$$

$$\frac{\partial H}{\partial u_{(t)}} = 0 \tag{2.15}$$

$$\frac{\partial H}{\partial \lambda(t)} - \dot{y} = 0 \tag{2.16}$$

Por fim, o ultimo passo de um método indireto consiste na resolução do sistema de EDOs (equações 2.13-2.16) como um problema de valor limite entra a condição inicial descrita e a condição final desejada (2.8).

2.4 O PROBLEMA DO CAIXEIRO VIAJANTE DUBINS COM VIZINHANÇAS

Segundo Matai, Singh e Mittal (2010), o TSP foi inicialmente estudado no século 18 por um matemático da Irlanda chamado Sir William Rowam Hamilton e pelo matemático britânico Thomas Penyngton Kirkman. Entretanto, a forma geral do TSP foi estudada por Kalr Menger na década de 20, em Viana e Havard, e é considerado o primeiro problema envolvendo roteirização. De acordo com Applegate et al. (2007), o TSP consiste em determinar uma rota de menor distância percorrida para um veículo de forma que um conjunto de pontos geograficamente

dispersos sejam visitados. O TSP é, em virtude da sua aplicabilidade, um dos problemas mais estudados na literatura, surgindo assim, diversas variantes do mesmo.

Em operações envolvendo drones equipados com sensores ou câmeras, muitas vezes não é necessário que o veículo visite a exata localização, mas apenas um ponto próximo a localidade e dentro do alcance desses equipamentos. Esse problema pode ser interpretado como o TSPN que é uma generalização do TSP em que o vendedor não precisa visitar a localização de cada cliente, em vez disso, uma região compacta do plano contendo cada nó é especificada como seu conjunto de vizinhanças e o objetivo se torna encontrar a trajetória mais curta que cruze todas essas vizinhanças. O TSPN foi primeiro introduzido por Dumitrescu e Mitchell (2003) e posteriormente estudado em Gentilini, Margot e Shimada (2013).

Os VANTs podem ser classificados em diferentes categorias, como mostrado em Elmeseiry, Alshaer e Ismail (2021). Dependendo do modelo utilizado, o veículo irá possuir restrições de movimento que devem ser levadas em consideração na hora do planejamento de trajetórias para o mesmo. Drones de asa fixa são VANTs com essas restrições, possuindo um raio mínimo de curvatura. Abordagens para lidar com elas foram propostas em Machmudah et al. (2022) e Xu et al. (2018). Outra forma de lidar com essas restrições de movimento, é através do modelo de Dubins (Dubins, 1957).

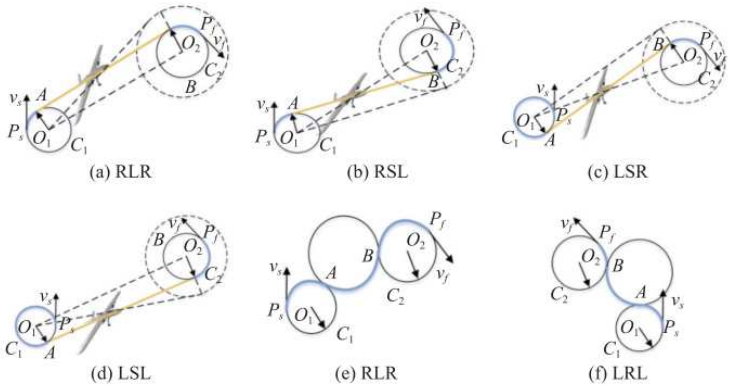
O veículo Dubins é amplamente utilizado para modelar sistemas autônomos como drones, brocas, veículos subaquáticos e robôs, conforme discutido em Balkcom, Furtuna e Wang (2018) e Živojević e Velagić (2019). O caminho mais curto com restrição de curvatura conectando dois pontos $q_i, q_j \in SE(2)$ pode ser calculado analiticamente e é uma das seis manobras possíveis de Dubins, consistindo em um segmento de linha reta e arcos com a curvatura ρ (Dubins, 1957).

O veículo Dubins consiste em um veículo que está sempre avançando e possui apenas 3 controles, sendo eles: esquerda (L), direita (R) e seguir em frente (S). Dubins provou em seu artigo Dubins (1957) que com 6 combinações desses controles, é possível descrever todos os caminhos mais curtos para um veículo com restrição de curvatura: LSR, RSL, RSR, LRL e RLR, como mostrado na Figura 2.

O roteamento de veículos Dubins consiste em um problema de roteamento em que o veículo só pode avançar para frente, em velocidade constante e possui um raio de giro limitado, esse problema pode ser definido como o DTSP (Ny, Frazzoli e Feron, 2007). O DTSP é uma generalização do TSP, no qual o comprimento do caminho não depende apenas da distância entre as localidades, mas também do ângulo de direção específico do veículo em cada uma delas e do seu raio de giro mínimo.

Diante disso, Obermeyer (2009) foi o primeiro a estudar o TSPN aplicado a dinâmica de movimento do veículo Dubins e propôs um algoritmo genético para resolução do problema. Posteriormente, Obermeyer, Oberlin e Darbha (2010) propõem um método de análise sintá-

Figura 2 – Diagrama do caminho de Dubins.



Fonte: Hu et al. (2022).

tico completo baseado em amostragem aleatória, possibilitando a transformação do problema em uma variante do TSP generalizado (GTSP), com conjuntos finitos de nós mutuamente exclusivos, que é então convertida em um TSP assimétrico que é analisado pelo algoritmo LKH (Helsgaun, 2000). Sendo o DTSPN formalmente definido em Isaacs, Klein e Hespanha (2011) que usam uma abordagem semelhante a proposta por Obermeyer (2009) mas a transformação Noon-Bean (Noon e Bean, 1993) é usada para transformar o GTSP com conjuntos de nós sobrepostos para o ATSP.

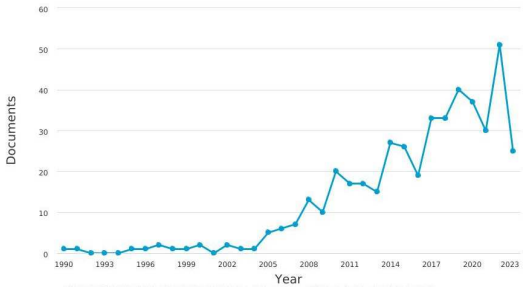
2.5 O ESTADO DA ARTE NO PROBLEMA DTSPN

A literatura sobre roteirização e planejamento de rotas de veículos possui milhares de trabalhos publicados. Esta pesquisa tem como objetivo investigar o estado da arte desses desenvolvimentos. Baseado em uma base de dados bibliométrica extraída do sistema web Scopus, que contém uma das maiores bases de dados de literatura científica revisada.

A busca foi realizada utilizando as palavras-chave “vehicle routing”, “trajectory planning” ou “path planning” combinadas com “curvature-constrained” e “dubins vehicle”. Com o intuito de mostrar todos os trabalhos na área de roteamento de veículos que incluíram veículos com restrição de curvatura, como os VANTs que são o foco deste artigo.

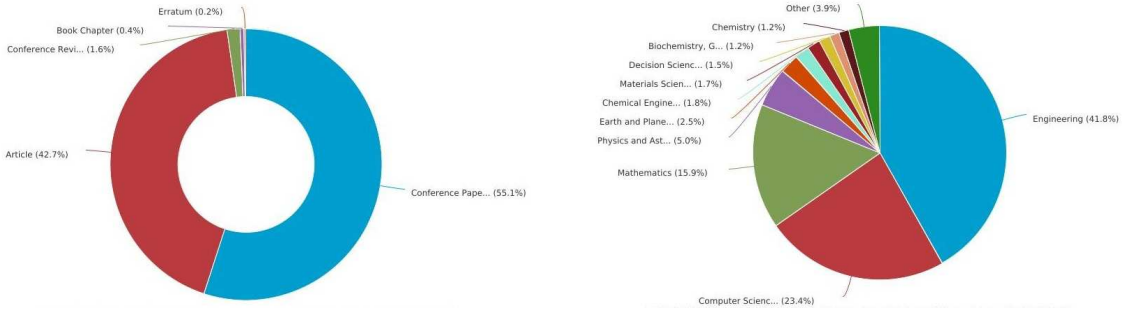
A última pesquisa foi realizada em 3 de setembro de 2023 através dos motores de busca disponibilizados pelo site. Foi recuperado um total de 445 registros, composto por 245 artigos de conferências (55,1%), 190 artigos (42,7%), 7 resenhas de conferências (1,6%), 2 capítulos de livros (0,4%) e 1 errata (0,2%). As publicações estão distribuídas entre 45 países, dos quais os cinco em número de publicações são China (137), Estados Unidos (123), República Tcheca (22), Reino Unido (22) e Itália (20); 159 autores dos quais os com maior número de publicações estão Tsourdos, A. (20) Faigl, J. (18), Shima, T. (15), White, B.A. (13) Zbikowski, R. (13) e Rathinam, S. (12).

Figura 3 – Artigos publicados ao longo dos anos no Scopus Web System (1990-2023).



Fonte: Autoria própria (2023).

Figura 4 – Porcentagem de artigos publicados por tipo de documento/assunto (Scopus Web System).



Fonte: Autoria própria (2023).

Os refinamentos descritos pelas consultas Q2, Q3 e Q4 limitam os resultados a trabalhos escritos em inglês com palavras-chave específicas no resumo do documento, como "Unmanned Aerial Vehicles (UAV)", "Fixed Wing"(para uma compreensão mais clara do pesquisa sistemática relacionada, consulte Tabela 1). Foram identificados 173 documentos como resultado do processo sistemático, dos quais os autores não conseguiram acesso a 11 deles e que portanto foram retirados da análise. Além dos 162 documentos restantes, os autores acrescentaram mais 5 artigos que não foram detectados pela busca sistemática por diversos motivos totalizando 167 artigos.

O veículo Dubins é usado para representar as restrições de movimento dos VANTs em 94,61% (158) dos artigos. Os demais artigos utilizam diferentes métodos, como algoritmo baseado em Euler-Lagrange (Yokoyama e Ochi, 2008) (Yokoyama e Ochi, 2009), Bézier Curves (Yang e Sukkarieh, 2008) (Faigl e Vana, 2018), Fast Marching Square (Li, Ding e Cai, 2009) (Garrido et al., 2022), fuzzy logic control scheme (Wilburn et al., 2012), trochoidal paths (Ayhan et al., 2019), vector fields (Shivam e Ratnoo, 2021).

Em relação às condições ambientais, 26,94% (45) dos artigos estudaram problemas tridimensionais. Evitar obstáculos foi abordado em 37,72% (63) dos artigos. Apenas alguns estudos 3% (5) incluíram os efeitos do vento nas trajetórias dos VANTs. Cerca de 36,52%

Tabela 1 – Cadeias de consulta usadas para realizar a pesquisa sistemática no Scopus

Acronym	**Mention**
(Q1)	(TITLE-ABS-KEY(Vehicle Routing) OR TITLE-ABS-KEY(Trajectory Planning) OR TITLE-ABS-KEY(Path Planning)) AND (TITLE-ABS-KEY(Curvature-constrained) OR TITLE-ABS-KEY(Dubins Vehicle))
(Q2)	(TITLE-ABS-KEY(Vehicle Routing) OR TITLE-ABS-KEY(Trajectory Planning) OR TITLE-ABS-KEY(Path Planning)) AND (TITLE-ABS-KEY(Curvature-constrained) OR TITLE-ABS-KEY(Dubins Vehicle)) AND (EXCLUDE (DOCTYPE,"er") OR EXCLUDE (DOCTYPE,"ch")) AND (LIMIT-TO(LANGUAGE,"English"))
(Q3)	(TITLE-ABS-KEY(Vehicle Routing) OR TITLE-ABS-KEY(Trajectory Planning) OR TITLE-ABS-KEY(Path Planning)) AND (TITLE-ABS-KEY(Curvature-constrained) OR TITLE-ABS-KEY(Dubins Vehicle)) AND (EXCLUDE(DOCTYPE,"er") OR EXCLUDE(DOCTYPE,"ch")) AND (LIMIT-TO(LANGUAGE,"English")) AND (EXCLUDE(SRCTYPE,"k"))
(Q4)	(TITLE-ABS-KEY(Vehicle Routing) OR TITLE-ABS-KEY(Trajectory Planning) OR TITLE-ABS-KEY(Path Planning)) AND (TITLE-ABS-KEY(Curvature-constrained) OR TITLE-ABS-KEY(Dubins Vehicle)) AND (EXCLUDE(SRCTYPE,"k"))AND (EXCLUDE(DOCTYPE,"er") OR EXCLUDE(DOCTYPE,"ch")) AND (LIMIT-TO(LANGUAGE,"English")) AND (LIMIT-TO(EXACTKEYWORD,"Unmanned Aerial Vehicles (UAV)") OR LIMIT-TO(EXACTKEYWORD,"Aerial Vehicle") OR LIMIT-TO(EXACTKEYWORD,"Unmanned Aerial Vehicles") OR LIMIT-TO(EXACTKEYWORD,"Multiple UAVs") OR LIMIT-TO(EXACTKEYWORD,"Unmanned Vehicles") OR LIMIT-TO(EXACTKEYWORD,"UAV") OR LIMIT-TO(EXACTKEYWORD,"Unmanned Aircrafts") OR LIMIT-TO(EXACTKEYWORD,"Fixed Wings"))

Fonte: Autoria própria (2023).

(61) dos artigos detectados pela busca sistemática tratavam do uso de múltiplos VANTs ou de um trabalho coordenado entre VANTs e veículos terrestres. Dos quais 34,42% (21) também consideram ambiente cheio de obstáculos e evitação de obstáculos.

Todos os artigos referentes a ambientes com obstáculos foram retirados da revisão. Após uma análise meticulosa de cada artigo, 22 documentos foram selecionados como trabalhos de destaque, dos quais 5 já estavam incluídos em nossa lista bibliográfica anterior de artigos relevantes. Assim, foram acrescentados 17 documentos, totalizando 22 artigos finais relevantes. Dessa forma, 77,27% dos documentos foram identificados pela busca sistemática e 22,73% acrescentados pelo conhecimento especializado.

No que diz respeito à abordagem utilizada para resolver o problema de roteirização, 54,54% dos 22 artigos selecionados utilizam métodos desacoplados nos quais o problema é desacoplado em dois subproblemas: primeiro, é determinada a sequência de visita aos pontos;

então, o ângulo de direção em cada ponto da sequência é escolhido. O Problema Euclidiano do Caixeiro Viajante (ETSP) (Applegate et al., 2006) é geralmente usado para determinar a sequência alvo. Enquanto isso, a otimização do título pode usar diferentes métodos, como métodos heurísticos (Lim et al., 2010) (Babel, 2020) (Wolek et al., 2021), programação dinâmica (Oberlin, Rathinam e Darbha, 2010), pesquisa determinística (Cons, Shima e Domshlak, 2014) (Cons, Shima e Domshlak, 2011), Sistema de lógica difusa (Ernest e Cohen, 2012), Simulated Annealing (Zhu, Xie e Dai, 2016), baseado em amostragem (Yulong et al., **??**), pesquisa de vizinhança variável (Penicka et al., 2017) (Kucerova, Vana e Faigl, 2021) e otimização iterativa local (Váňa, Sláma e Faigl, 2018).

Enquanto isso, 45,45% implementaram métodos acoplados nos quais a sequência do alvo e os ângulos de rumo são resolvidos simultaneamente. À medida que a otimização avança, a sequência alvo muda e os ângulos de rumo podem ser re-escolhidos, por exemplo, usando algoritmos evolutivos (Zhang et al., 2014) (Ding et al., 2020). Outros métodos de otimização acoplados foram propostos que dependem de pesquisa determinística (Cons, Shima e Domshlak, 2014) (Cons, Shima e Domshlak, 2011), pesquisa de vizinhança variável (Pěnička et al., 2019), aprendizagem não supervisionada (Faigl, 2020) (Faigl e Pěnička,).

Cerca de 18% dos artigos simplificam o problema usando métodos de transformação nos quais as variáveis contínuas, como os ângulos de rumo, são discretizadas usando amostragem aleatória (Obermeyer, Oberlin e Darbha, 2012) (Isaacs e Hespanha, 2013) (Váňa, Sláma e Faigl, 2018) (Wolek et al., 2021) para formular um TSP Generalizado (GTSP). Uma vez definido o GTSP, ele pode ser transformado em TSP Assimétrico (ATSP) usando a transformação Noon-Bean (Noon e Bean, 1991) e então resolvido. Existe uma sobreposição entre esses artigos, uma vez que alguns deles empregam mais de uma metodologia.

Em cerca de 45% dos artigos selecionados o caminho dubins é simplificado usando o caminho dubins com relaxamento terminal, proposto por Bui e Boissonnat (1994), no qual a configuração inicial e o ponto final são fixos, mas o ângulo de chegada do veículo é livre. Com o uso desta abordagem de relaxação, o cálculo do caminho de Dubins pode ser notavelmente simplificado.

Todos os artigos selecionados como trabalhos de destaque tratam do DTSPN ou de um problema semelhante. A busca sistemática mostrou que nenhum método exato para resolver o DTSPN ou problemas similares que lidam com veículos com restrição de curvatura foi proposto na literatura de roteamento de veículos ou planejamento de trajetória.

3 O PROBLEMA DO CAIXEIRO VIAJANTE DUBINS COM VIZINHANÇAS

3.1 DEFINIÇÃO DO PROBLEMA

Seja $G = (V, A)$ onde $V = \{0, 1, ..., n-1\}$ e $A = \{(i, j) : i, j \in V, i \neq j\}$ representam um grafo contendo vértices $i \in V$ conectados por arcos $a \in A$. Cada vértice $i \in V$ é descrita por uma tupla $(\bar{x}_i, \bar{y}_i, \bar{\theta}_i)$, onde $(\bar{x}_i, \bar{y}_i) \in \mathbb{R}^2$ representam a posição no plano e $\bar{\theta}_i$ representa o ângulo em que o veículo deve chegar e sair do vértice. Sem perda de generalidade, assume-se que $\bar{\theta}_0$ é livre. Além disso, para cada vértice $i \in V \setminus \{0\}$ tem-se um conjunto convexo e compacto $C_i \subset \mathbb{R}^2$ associado a qual refere-se como sendo a região de cobertura do vértice i. Um vértice i é considerado visitado se a trajetória do veículo intercepta sua região de cobertura.

O Dubins Traveling Salesman Problem with Neighborhoods (DTSPN) pode ser formalmente definido da seguinte forma: Encontre um caminho Dubins único de menor tempo que visite cada vértice $i \in V$ uma única vez e retorne a origem. Lembrando que um vértice é considerado visitado somente se sua região de cobertura for interceptado pelo caminho.

3.1.1 Veículo Dubins

O veículo Dubins consiste em um veículo que está sempre avançando com velocidade constante e com raio de giro limitado ρ. A posição do veículo pode ser representada pelas funções $x(\tau), y(\tau)$ e $\theta(\tau)$, onde $x(\tau)$ e $y(\tau)$ representam a posição do veículo no plano $2D$ e $\theta(\tau)$ a direção do veículo medida no sentido anti-horário do eixo x, no tempo $\tau \in [\tau^o, \tau^f]$. Assume-se que $\tau^o = 0$. A dinâmica do veículo é modelada usando as seguintes equações diferenciais:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \theta \end{bmatrix} = v \begin{bmatrix} \cos\theta(\tau) \\ \sin\theta(\tau) \\ u(\tau) \cdot \rho^{-1} \end{bmatrix}, \ |u| \leq 1. \tag{3.1}$$

onde $v \in \mathbb{R}^+$ é a velocidade do veículo e $u(t)$ é a variável de controle, que deve estar em um intervalo simétrico $u(t) \in [-1, 1]$. O veículo Dubins é um modelo bastante difundido nas áreas de robótica e teoria de controle para planejar trajetórias de robôs com rodas, aviões e veículos subaquáticos.

3.2 FORMULAÇÃO MATEMÁTICA

O problema é formulado como um Problema de Otimização de Trajetória Inteiro Misto com múltiplas fases sobre o grafo G. Nossa formulação associa cada arco $a \in A$ para uma fase de possíveis trajetórias. Nesse trabalho, usa-se os temos arco e fase intercambiavelmente.

Neste caso, cada fase pode ser entendida como um conjunto de parâmetros, como condições inicias e finais. de modo que o sistema dinâmico descrito pelas EDOs (3.1) continue valido. Dessa forma, o DTSPN pode ser reformulado como: (I) as variáveis de estado $S_{ij}(\tau_{ij}) = (x_{ij}(\tau_{ij}), y_{ij}(\tau_{ij}), \theta_{ij}(\tau_{ij})) : \mathbb{R}^+$ e de controle $u_{ij}(\tau_{ij}) : \mathbb{R}^f; \tau_{ij} \in \tau_{ij}^o, \tau_{ij}^f$; (II) o intervalo de tempo τ_{ij}^o (inicial) e τ_{ij}^f (final) para cada arco percorrido na trajetória e; (III) uma permutação $s = (i_0, ..., i_{n-1}), i_j \in V$, de modo que todo vértice $i \in V$ é visitado, por exemplo, $S_{ij}(t_{ij}^f) \cap C_j \neq \phi \forall i, j \in V, i \neq j$, e o tempo total de trajeto é minimizado.

Cabe ressaltar que os estados e controles desconhecidos são interpretados como a evolução do sistema dinâmico (EM_s) onde τ é a variável independente. Sem perda de generalidade assume-se que cada vértice $i \in V$ pode ser visitado no último passo de tempo do arco (i, j), por exemplo, $S_{ij}(t_{ij}^f) \cap C_j \neq \phi \forall i, j \in V$.

A seguir, as Equações de Movimento (EMs) do veículo Dubins atravessando o arco (i, j) são expressas por:

$$s_{ij} = f_{ij}(s_{ij}(\tau_{ij}), u_{ij}(\tau_{ij}, \tau_{ij})), \forall (i, j) \in A, \tag{3.2}$$

em que $f_{ij}(s_{ij}(\tau_{ij}), tau_{ij})$ representa o lado direto das EMs. Ambas as variáveis de estado e de controle são limitadas por limites inferiores e superiores, ou seja, $s^{lb} \leq s_{ij}(\tau_{ij}) \leq s^{ub}$ e $-1 \leq u_{ij}(\tau_{ij}) \leq 1$. Os limites s^{lb} e s^{ub} podem ser definidos de acordo com a aplicação. As condições iniciais s^o e u^o devem ser fornecidas no tempo $\tau = 0$, ou seja, $s_{ij}(0) = s^o$ e $u_{ij}(0) = u^o \forall i, j \in A$.

Agora vamos definir as variáveis $a_{ij} \in [0, 1]$ tal que $a_{ij} = 1$ se o arco $(i, j) \in A$ for usado em uma determinada rota e $a_{ij} = 0$ caso contrário. Para simplificar, o conjunto A' é definido como o conjunto de arcos que não partem do vértice inicial $i_o = 0$. Portanto, uma formulação multifásica de otimização de trajetória inteira mista para o DTSPN pode ser escrita da seguinte forma:

3.2.1 Conjuntos

- $V = \{1, \ldots, n\}$: Conjunto de n localidades.
- $A = \{1, \ldots, N\}$: Conjunto de N arcos.

3.2.2 Parâmetros

- v: Velocidade do veículo Dubins.
- ρ: Raio mínimo de giro do veículo Dubins.

3.2.3 Variáveis

- a_{ij}: Variável binária que assume valor 1 se o veículo realiza o percurso da localidade i para a localidade j, ou 0 caso contrário.
- s_{ij}: Variável contínua que representa as variáveis de estado do veículo no trajeto da localidade i para a localidade j.
- u_{ij}: Variável contínua que representa o comportamento da variável de controle trajeto da localidade i para a localidade j, que deve estar em um intervalo simétrico $u \in [-1, 1]$.
- τ_{ij}: Variável contínua que representa o tempo gasto no trajeto da localidade i para a localidade j.

3.2.4 Modelo

$$\min \tau_{n-1}^{f} \tag{3.3}$$

s.a.

$$\sum_{j \in V} a_{ij} = 1, \forall i \in V \tag{3.4}$$

$$\sum_{i \in V} a_{ij} = 1, \forall j \in V \tag{3.5}$$

$$s_{ij}(t_{ij}^{f}) \cap C_j \neq \phi \Leftrightarrow a_{ij} = 1, \forall (i,j) \in A \tag{3.6}$$

$$\dot{s}_{ij} = f_{ij}(s_{ij}(\tau_{ij}), u_{ij}(\tau_{ij}, \tau_{ij}), \forall (i,j) \in A, \tag{3.7}$$

$$s_{jk}(\tau_{jk}^{o}) a_{ij} = s_{ij}(\tau_{jk}^{f}) a_{ij} a_{jk}, \forall (i,j)(j,k) \in A \tag{3.8}$$

$$u_{jk}(\tau_{jk}^{o}) a_{ij} = u_{ij}(\tau_{jk}^{f}) a_{ij} a_{jk}, \forall (i,j)(j,k) \in A \tag{3.9}$$

$$\tau_{jk}^{o} a_{ij} = \tau_{ij}^{f} a_{ij} a_{jk}, \forall (i,j)(j,k) \in A \tag{3.10}$$

$$\tau_{ij}^{o} \leq \tau_{ij} \leq \tau_{ij}^{f} \forall (i,j) \in A \tag{3.11}$$

$$a_{ij} \in \{0,1\}, s_{ij}(\tau_{ij}) \in \mathbb{R}^3, u_{ij}(\tau_{ij}) \in [-1,1], \tau_{ij}^{o}, \tau_{ij}^{f} \in \mathbb{R}, \forall (i,j) \in A \tag{3.12}$$

Na formulação acima, (3.3) representa a função objetivo do problema que consiste em minimizar o tempo de trajeto. (3.4) e (3.5) são restrições que exigem que o veículo entre e saia de cada nó exatamente uma vez, fazendo com que cada localidade seja visitada uma única ocasião. As restrições (3.6) garantem que todos os vértices sejam visitados e que a trajetória passe pela vizinhança de cada vértice i em V

As restrições (3.7) representam o sistema de ODE(s) responsáveis pelas restrições de movimento do veículo Dubins em que o veículo só pode se movimentar para frente em linha reta e para direita ou esquerda com um raio de giro de ρ.

As restrições (3.8) - (3.10) são restrições de ligação entre arcos que são responsáveis por conectar o fim de um arco com o início do outro para que ocorra a continuidade da trajetória. As restrições (3.8) conectam a posição do veículo no plano entre dois arcos, a posição inicial do arco a_{jk} vai ser igual a posição final do arco a_{ij}, caso ambos os arcos forem utilizados na trajetória. As restrições (3.9) atuam de maneira semelhante, ligando as variáveis de controle finais de um arco com o inicio do arco seguinte. Já as restrições (3.10), conectam as variáveis responsáveis por calcular o tempo de trajeto.

As restrições (3.11) garantem que o tempo de percurso τ_{ij} é menor ou igual ao tempo final τ_{ij}^f e maior ou igual ao tempo inicial τ_{ij}^o para todo i e j em A. Por fim, as restrições (3.12) definem o domínio das variáveis.

3.2.5 Restrições de Visita

Existem muitas opções de funções para descrever as regiões compactas $C_i \cap \mathbb{R}^2, \forall i \in V\ \{0\}$, por exemplo, elas podem ser usadas para representar (a) a interseção de um conjunto de semiplanos, (b) elipsoides ou (c) vizinhanças delimitadas por segmentos de reta conectando uma sequência de pontos finitos. Um caso especial de (b) consiste no Problema de Orientação Dubins Suficientemente Próximo (CEDOP) proposto por Faigl e Pěnička (2017). No CEDOP, as regiões de cobertura $C_i, i \in V \setminus \{0\}$, são definidas como círculos com raio r, centrados em $(\bar{x}_i, \bar{y}_i)$. Isso leva às seguintes restrições:

$$((x_{ij}(\tau_{ij}^f) - \bar{x}_j)^2 + (y_{ij}(\tau_{ij}^f) - \bar{y}_j)^2)a_{ij} \leq \bar{r}_j^2, \forall i, j \in A \tag{3.13}$$

3.2.6 Discretização das Equações de Movimento

Um método de colocação direta é aplicado para converter o problema de otimização de trajetória de dimensão infinita proposto em um problema MINLP de dimensão finita. Por uma questão de simplicidade, o índice de arco $(i, j) \in A$ é omitido das variáveis de estado, controle e tempo. A seguir, é definido o conjunto de índices de tempo $T = \{0, ..., N-1\}$, N determina o tamanho da discretização, tal que $\tau_i = \tau_o + ih, \tau_i \in [\tau^o, \tau^f], h = \tau^f - \tau^o/N - 1$, onde h denota o tamanho do passo.

Isso define uma discretização de grade uniforme da forma: $\tau^o < \tau^1, ..., \tau^f$. Deixando s_t e u_t aproximar as variáveis de estado e controle $S(\tau^t)$ e $u(\tau^t)$ no instante de tempo $t \in T$. A derivada temporal de $s(\tau)$ é aproximada da seguinte forma:

$$\dot{S} \approx \frac{S_{t+1} - S_t}{h} \tag{3.14}$$

Ao aplicar esta aproximação à dinâmica do veículo de Dubins, é possível definir uma discretização explícita das EMs de Dubins que podem ser usadas como restrições na formulação proposta.

O Método de Euler aplicado as EMs do veiculo Dubins nos dá:

$$s_{t+1} = s_t + h(f(y_t, u_t, t)), \forall t \in T \tag{3.15}$$

O Método Trapezoidal aplicado as EMs do veiculo Dubins nos dá:

$$s_{t+1} = s_t + \frac{1}{2}h(f(y_{t+1}, u_{t+1}, t+1) + f(y_t, u_t, t)), \forall t \in T \tag{3.16}$$

3.2.6.1 Linearização das restrições de conexão

Restrições de conexão do tipo: $s^o_{jk}a_{ij} = s^f_{ij}a_{ij}a_{jk}$ ou $s^o_{jk} = s^f_{ij}$ se $a_{ij} = a_{jk} = 1$, que modelam restrições de caminho envolvendo um par de caminhos subsequentes em uma determinada rota, apresenta restrições bilineares que representam um desafio para solucionadores de PNL.

Linearização proposta:

$$\begin{array}{l} s^o_{jk} \geq s^f_{ij} - M(1 - a_{ij}) - M(1 - a_{jk}) \\ s^o_{jk} \leq s^f_{ij} + M(1 - a_{ij}) + M(1 - a_{jk}) \end{array}, M >> 1. \tag{3.17}$$

M deve ser escolhido de forma que $M > s^f_{ij} \forall (i, j) \in A$. Diferentes M para variáveis diferentes.

3.3 UMA ABORDAGEM DE DECOMPOSIÇÃO PARA O DTSPN

Devido a complexidade do problema, os *solvers* disponíveis na literatura se mostraram incapazes de encontrar soluções viáveis para a formulação matemática proposta (3.3-3.12). Por esse motivo, é aplicado o método GBD, no qual o problema é dividido em um problema mestre de roteamento, responsável por encontrar a melhor sequência de visitação, e um subproblema de otimização de trajetória que, a partir da rota encontrada no problema mestre, busca a melhor trajetória para a sequência encontrada. A seguir, explicasse em mais detalhes cada etapa do algoritmo proposto.

3.3.1 O problema mestre de roteamento de veículos

Começamos construindo um grafo secundário $G' = (V', A')$ tal que os conjuntos de vértice e arco V' e A', respectivamente, sejam desacoplados dos EMs de Dubins. Porém, o significado desses conjuntos permanece o mesmo, ou seja, V' coleta vértices $i \in V$ e A' coleta arcos $a \in A$. Deixe τ^{lb}_{ij} representar um limite inferior do tempo de voo entre os vértices i e j. Além disso, deixe S denotar o conjunto de todas as rotas que visitam n vértices. As variáveis de decisão incluem a_{ij}, que assume valor 1 se o arco (i, j) for usado em um passeio e 0 caso contrário, e Δ que é uma variável auxiliar que representa o custo de uma rota viável $s \in S$.

A formulação pode ser escrita da seguinte forma:

$$\min \sum_{i \in V'} \sum_{j \in V'} \tau_{ij}^{lb} x_{ij} + \Delta \tag{3.18}$$

s.a.

$$\Delta \geq \Delta_s - M(n - \sum_{(i,j) \in s} x_{ij}), \forall s \in S \tag{3.19}$$

$$\sum_{j \in V'}^{n} x_{ij} = 1, \forall j \in V' \tag{3.20}$$

$$\sum_{i \in V'}^{n} x_{ij} = 1, \forall i \in V' \tag{3.21}$$

$$\sum_{i,j \in S'}^{n} x_{ij} \leq |S'| - 1, \forall S' \in \{0, ..., n-1\}, 2 \leq |S'| \leq n. \tag{3.22}$$

$$x_{ij} \in \{0, 1\}, i, j = 0, ..., n. \tag{3.23}$$

$$\Delta \geq 0 \tag{3.24}$$

Na formulação (3.18 a 3.24), a função objetivo (3.18) busca minimizar o tempo de trajeto. A restrição (3.19) garante que a variável Δ seja maior ou igual a diferença entre o tempo real e o tempo relaxado para a sequência s. As restrições (3.20) e (3.21) exigem que o veículo entre e saia de cada nó exatamente uma vez. (3.22) diz que o somatório das arestas do subconjunto de vértices x_{ij} deve ser menor ou igual ao modulo do subconjunto $S' - 1$, evitando *subtours*. Por fim, as restrições (3.23) e (3.24) definem o domínio das variáveis.

O algoritmo é implementado inicialmente sem as restrições (3.19) e (3.22). As restrições de *subtour* e de trajetória são aplicadas ao modelo, utilizando o *branch-and-cut*. Ao encontrar uma solução, um teste lógico é realizado para verificar se a solução encontrada possui *subtour*, caso possua, a restrição (3.22) é aplicada para o subconjunto de vértices x_{ij} dessa solução. Caso a solução encontrada não possua *subtour*, a solução é valida.

3.3.2 Um subproblema de otimização de trajetórias

Deixe X representar uma solução inteira viável para o problema mestre e $s = (a_0, a_1, .., a_{n-1})$ sua sequência de visita correspondente. Encontrar um caminho de Dubins entre os vértices i e j para cada arco subsequente $a \in s$, pode ser formulado como o seguinte problema de otimização de trajetória multifásica.

$$\min \tau_{n-1}^{f} \tag{3.25}$$

s.a.

$$s_{ij}(t_{ij}^{f}) \cap C_j \neq \phi \Leftrightarrow a_{ij} = 1, \forall (i,j) \in A \tag{3.26}$$

$$\dot{s}_{ij} = f_{ij}(s_{ij}(\tau_{ij}), u_{ij}(\tau_{ij}, \tau_{ij}), \forall (i,j) \in A, \tag{3.27}$$

$$s_{jk}(\tau^o_{jk}) = s_{ij}(\tau^f_{jk}), \forall (i,j)(j,k) \in A \tag{3.28}$$

$$u_{jk}(\tau^o_{jk}) = u_{ij}(\tau^f_{jk}), \forall (i,j)(j,k) \in A \tag{3.29}$$

$$\tau^o_{jk} = \tau^f_{ij}, \forall (i,j)(j,k) \in A \tag{3.30}$$

$$\tau^o_{ij} \leq \tau_{ij} \leq \tau^f_{ij} \forall (i,j) \in A \tag{3.31}$$

$$s_{ij}(\tau_{ij}) \in \mathbb{R}^3, u_{ij}(\tau_{ij}) \in [-1,1], \tau^o_{ij}, \tau^f_{ij} \in \mathbb{R}, \forall (i,j) \in A \tag{3.32}$$

Seja $s = (i_0, ..., i_n)$ qualquer rota viável para o problema de roteamento, o subproblema de otimização de trajetória calcula o tempo real para percorrer a sequência s. Se uma solução viável for encontrada para o o subproblema, é utilizado o *branch-and-cut* para adicionar a restrição (3.19) ao problema mestre.

3.3.2.1 Cálculo do limite inferior do tempo de voo

Calcular bons valores de limite inferior $\tau^{lb}_{ij} \forall (i,j) \in A'$ não é uma tarefa trivial. Para o DTSP, no qual se assume configurações inicial e final fixas, $(s^o_{ij}, u^o_{ij}, \tau^o_{ij})$ e $(s^f_{ij}, u^f_{ij}, \tau^f_{ij})$, respectivamente, as distâncias euclidianas representam uma aproximação razoável se a distância entre os vértices for grande o suficiente. Para o DTSPN, a qualidade da aproximação euclidiana depende não apenas da distância entre os vértices, mas também do tamanho ou área das regiões vizinhas. É possível melhorar o *lower bound* euclidiano levando em consideração a dinâmica de Dubins e configurações iniciais e finais possivelmente fixas:

O *lower bound* pode ser calculado resolvendo o seguinte problema de otimização de trajetória:

$$\min \tau^f \tag{3.33}$$

s.a.

$$\dot{s} = f(s(\tau), u(\tau), \tau)., \tau \in [0, \tau^f] \tag{3.34}$$

$$s(\tau^o) = (x_i, y_i, \theta^o) \tag{3.35}$$

$$s(\tau^f) = (x_j, y_j, \theta^f) \tag{3.36}$$

$$s(\tau^f) \cap C_j \neq \phi \tag{3.37}$$

$$(x_i, y_i), (x_j, y_j) \in \mathbb{R}^2 \tag{3.38}$$

$$\tau^f \geq 0 \tag{3.39}$$

$$s(\tau) \in \mathbb{R}^3 \tag{3.40}$$

$$u(\tau) \in [-1, 1] \tag{3.41}$$

4 RESULTADOS

Os modelos descritos anteriormente foram implementados utilizando a linguagem de programação Python e as bibliotecas Gurobi e Pyomo. Os experimentos computacionais foram executados em um computador com sistema operacional Windows 10 de 64 bits, processador Intel Core i5-9300H de 2.40 GHz e 16.0 GB de memoria RAM. Os softwares de otimização usados foram Gurobipy 9.5.2 e IPOPT 1.0.3.

4.1 GERAÇÃO DE INSTÂNCIAS PARA O DTSPN

Apesar de já ter sido abordado em outros trabalhos, o DTSPN não possui um banco de dados disponível com instâncias que possam ser utilizadas para futuras pesquisas. Diante disso, a partir das instâncias disponibilizadas por Behdani e Smith (2014) para o Problema do Caixeiro Viajante Suficientemente Próximo ou CETSP (do inglês *Close-enough Dubins Traveling Salesman Problem*), foi criado um gerador de instâncias para o DTSPN.

Todas as instâncias do CETSP foram geradas por Behdani e Smith (2014) de acordo com o seguinte procedimento. Os $|M|$ locais alvo e o depósito são escolhidos aleatoriamente em um retângulo (comprimento 16 e largura 10). A vizinhança definida para todos os alvos são discos de raio idêntico r que no nosso algoritmo abordamos como sendo um parâmetro que variamos em nossos testes computacionais.

O gerador adiciona θ a uma instância já existente, que possui apenas as coordenadas (x, y), utilizando uma distribuição uniforme que atribui o valor $0 \leq \theta \leq 2\pi$ para cada uma das localidades da instância. A primeira linha de cada arquivo contém as coordenadas x e y do depósito e o ângulo de aproximação no qual o veículo deve chegar e sair do local para aquela instância. As linhas subsequentes contêm as coordenadas x e y dos locais alvo e o ângulo de aproximação dos locais alvo.

O arquivo "DTSPN-X-Y"contém as informações para a instância do problema "DTSPN-X-Y", independentemente do valor do raio de cobertura. Por exemplo, "DTSPN-12-1"contém as coordenadas dos locais alvo e o ângulo de aproximação e o depósito para todos os problemas de teste associados a "DTSPN-12-1"(para cada valor do raio de cobertura obteremos uma nova instância de teste). O banco de dados contendo as instâncias será disponibilizado para que possam ser utilizadas por outros pesquisadores que abordarem o DTSPN.

4.2 EXPERIMENTOS COMPUTACIONAIS

Todos os experimentos foram realizados utilizando as instâncias geradas e que possuem 7, 9, 11, 13, 15 e 17 vértices. Foi atribuído os seguintes valores aos parâmetros, raio de giro

mínimo $\rho = 1$, raio mínimo de proximidade $r = 1$ e com discretização igual a 30. Além disso, foi estipulado um tempo computacional máximo para cada instância de 3600 segundos (1 hora), após o tempo estipulado, o *solver* encerra a otimização. O valor da função objetivo e o tempo de CPU foram coletados em segundos. As tabelas com os resultados de todas as instâncias testadas encontram-se no APÊNDICE 5.

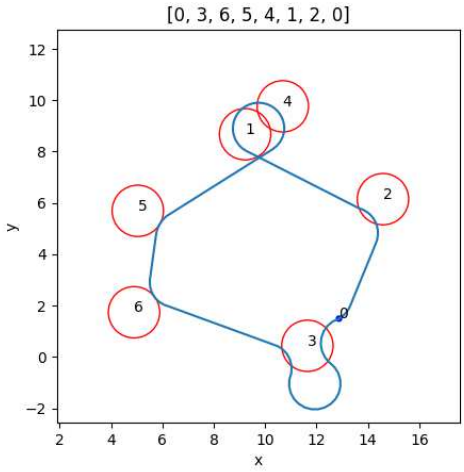
A Tabela 8 apresenta os resultados encontrados para as instâncias com 7 vértices. Todas as instancias foram resolvidas na otimalidade. A instância DTSPN-6-18 foi a que apresentou o menor tempo para ser resolvida (3,509 segundos), enquanto a instância DTSPN-6-27 foi a que demandou mais tempo (377,521 segundos $\approx$ 6,3 minutos), as Figuras 5 e 6 mostram a sequência e a rota utilizada para essas instâncias. Como apresentado na Tabela 2, todas as instâncias foram resolvidas na otimalidade, com valor médio da função objetivo de 46,483 segundos e tempo computacional médio de 123,048 segundos, além disso, todas as soluções apresentaram um *gap* de 0%.

Tabela 2 – Média dos resultados para instâncias com 7 vértices.

Status	Instâncias	Objetivo	GAP	Tempo	Soluções
Optimal	25	46,483	0%	123,048	4
Tempo Limite	-	-	-	-	-

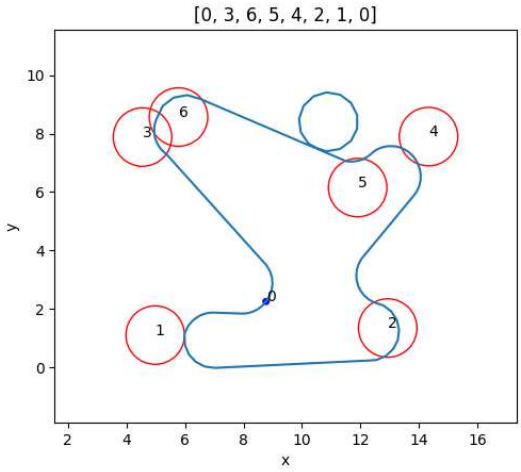
Fonte: Autoria própria (2023).

Figura 5 – Trajetória ótima DTSPN-6-18.



Fonte: Autoria própria (2023).

Figura 6 – Trajetória ótima DTSPN-6-27.



Fonte: Autoria própria (2023).

A Tabela 3 mostra a média dos resultados para as instâncias com 9 vértices. Apesar de não conseguir encontrar soluções ótimas para todas as instâncias, das 28 instâncias testadas, 22 chegaram em um status de otimalidade, enquanto 6 instâncias atingiram o tempo computacional limite sem encontrar uma solução ótima. Para as instâncias com soluções ótimas, o tempo computacional necessário foi de (942,526 segundos $\approx$ 15,7 minutos). Já as instâncias que atingiram

o tempo computacional limite, apesar de não encontrar soluções ótimas para o problema, o *gap* médio foi de 14,7%.

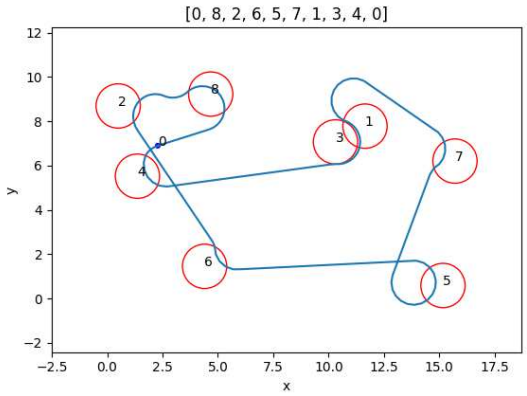
Na Tabela 9 são apresentados os resultados dos experimentos realizados para as instâncias com 9 vértices. Dentre as instâncias as instâncias com 9 vértices que foram resolvidas na otimalidade, as instâncias DTSPN-8-25 e DTSPN-8-2 requereram apenas (51,907 segundos) e (74,491 segundos), respectivamente, enquanto a instância DTSPN-8-8 precisou de (2466,719 segundos $\approx$ 41 minutos) para encontrar uma solução ótima, as Figuras 7 e 8 mostram a sequência e a rota utilizada para as instâncias DTSPN-8-25 e DTSPN-8-8, respectivamente.

Tabela 3 – Média dos resultados para instâncias com 9 vértices.

Status	Instâncias	Objetivo	GAP	Tempo	Soluções
Optimal	22	50,542	0	942,526	4
Tempo Limite	6	58,566	14,7%	3645,823	4

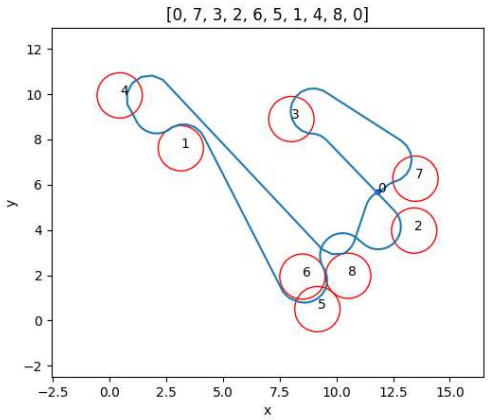
Fonte: Autoria própria (2023).

Figura 7 – Trajetória ótima DTSPN-8-25.



Fonte: Autoria própria (2023).

Figura 8 – Trajetória ótima DTSPN-8-8.



Fonte: Autoria própria (2023).

Para as instâncias com 11 vértices, apresentadas na Tabela 10, o número de instâncias que atingiram a otimalidade diminui drasticamente quando comparamos com as de 7 e 9 vértices. Enquanto para as instâncias com 7 e 9 vértices foi possível chegar na otimalidade em 100% e 78,6% delas, respectivamente, para as instâncias com 11 vértices apenas 17,9% atingiram a otimalidade. Apesar do número de instâncias que atingiram o tempo computacional limite ter aumentado quando comparado as instâncias de 9 vértices, o *gap* médio das soluções encontradas para essas instâncias foi de 21,5%.

Dentre as instâncias com 11 vértices que foram resolvidas na otimalidade, a instância DTSPN-10-26 precisou de (753,863 segundos $\approx$ 12,56 minutos), enquanto a instância DTSPN-10-4 precisou de (3258,104 segundos $\approx$ 54,3 minutos), chegando próximo ao tempo computa-

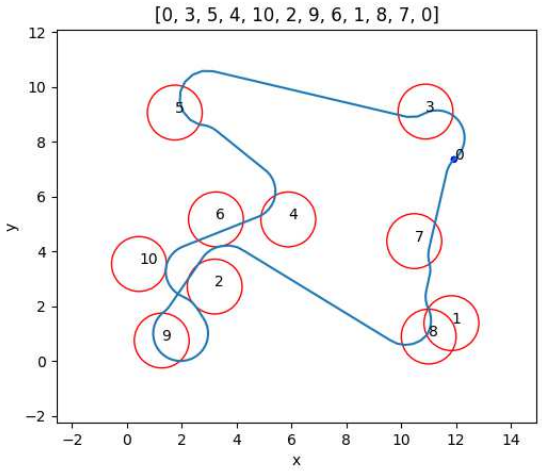
cional limite para encontrar uma solução ótima, as Figuras 9 e 10 mostram a sequência e a rota utilizada para essas instâncias.

Tabela 4 – Média dos resultados para instâncias com 11 vértices.

Status	Instâncias	Objetivo	GAP	Tempo	Soluções
Optimal	5	53,081	0%	1814,449	5
Tempo Limite	23	60,82	21,5%	3646,631	5

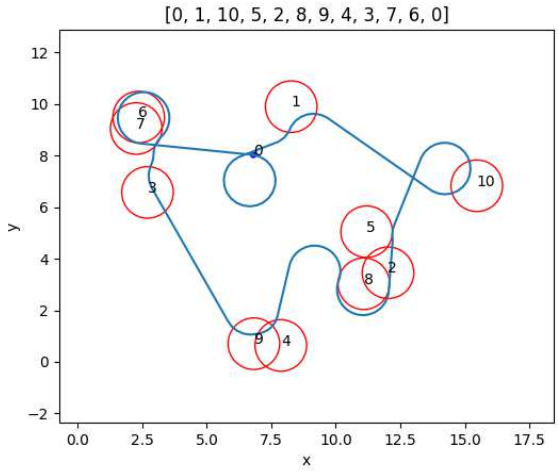
Fonte: Autoria própria (2023).

Figura 9 – Trajetória ótima DTSPN-10-26.



Fonte: Autoria própria (2023).

Figura 10 – Trajetória ótima DTSPN-10-4.



Fonte: Autoria própria (2023).

Ao aumentar o tamanho de vértices, o problema se torna consideravelmente mais desafiador e, consequentemente, o tempo computacional necessário para resolução da instância aumenta. O algoritmo proposto não foi capaz de encontrar soluções ótimas para as instâncias com 13 vértices no tempo computacional estabelecido nos experimentos, sendo capaz apenas de chegar em soluções com um *gap* de 34,7% como mostra a Tabela 5.

Tabela 5 – Média dos resultados para instâncias com 13 vértices.

Status	Instâncias	Objetivo	GAP	Tempo	Soluções
Optimal	-	-	-	-	-
Tempo Limite	27	71,779	34,7%	3641,448	3

Fonte: Autoria própria (2023).

Entretanto, foi capaz de chegar na otimalidade para uma instância com 15 vértices, com valor da função objetivo de 113,344 segundos e tempo computacional de (848,947 segundos $\approx$ 14 minutos), como mostra a Tabela 6. Para as demais instâncias com 15 vértices, o tempo computacional limite foi atingido e as soluções encontradas possuíam um *gap* médio de 46,3%.

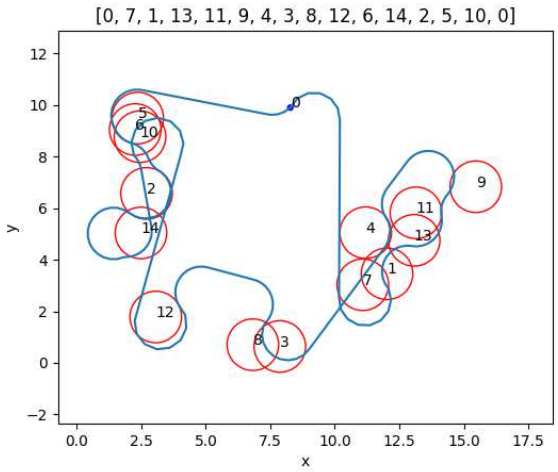
Tabela 6 – Média dos resultados para instâncias com 15 vértices.

Status	Instâncias	Objetivo	GAP	Tempo	Soluções
Optimal	1	113,344	0%	848,947	10
Tempo Limite	23	88,581	46,3%	3629,953	2

Fonte: Autoria própria (2023).

A Figura 11 mostra a trajetória tomada pelo veiculo para atingir a otimalidade na instância DTSPN-14-3. Os círculos em vermelho ilustram a vizinhança de cada localidade e possuem um raio $r = 1$ e a sequência de visitas utilizada pelo veículo foi S = (0, 7, 1, 13, 11, 9, 4, 3, 8, 12, 6, 14, 2, 5, 10, 0).

Figura 11 – Trajetória ótima DTSPN-14-3.



Fonte: Autoria própria (2023).

Assim como com as instâncias com 13 vértices, o algoritmo também não foi capaz de encontrar soluções ótimas para as instâncias com 17 vértices no tempo computacional estabelecido nos experimentos, sendo capaz apenas de chegar em soluções com um *gap* de 46,8% como mostra a Tabela 7.

Tabela 7 – Média dos resultados para instâncias com 17 vértices.

Status	Instâncias	Objetivo	GAP	Tempo	Soluções
Optimal	-	-	-	-	-
Tempo Limite	9	92.897	46.8%	3625.792	2

Fonte: Autoria própria (2023).

5 CONCLUSÃO E TRABALHOS FUTUROS

O presente trabalho abordou o Problema do Caixeiro Viajante Dubins com Vizinhanças, sendo proposta uma formulação matemática que considera as rotas e trajetórias simultaneamente, baseada em programação inteira-mista, entretanto, a formulação proposta mostrou-se numericamente intratável tendo em vista que os *solvers* disponíveis não foram capazes de encontrar soluções viáveis para o problema.

Dessa forma, foi proposta uma abordagem baseada no método de Decomposição Generalizada de Benders proposto por (GEOFFRION, 1997), no qual o problema foi decomposto em um problema mestre de roteamento de veículos e um subproblema de otimização de trajetórias. Para a resolução do problema mestre, utilizou-se do método branch-and-cut, enquanto para o subproblema utilizou-se um método exato baseado em colocação direta. Além disso, realizou a criação de um gerador de instâncias e a disponibilização de um banco de dados contendo instâncias para o DTSPN.

Através dos experimentos computacionais, o método de decomposição apresentou resultados satisfatórios, sendo capaz de encontrar soluções ótimas para instâncias com 7, 9, 11 e 15 vértices em um tempo computacional aceitável. Todas as instâncias com 7 vértices foram resolvidas na otimalidade, enquanto para as instâncias com 9 e 11 vértices, foi possível chegar na otimalidade em 78,5% e 17,9%, respectivamente, já para as instâncias com 15 vértices, apenas uma instância chegou na otimalidade, comprovando a eficiência do modelo proposto. Para as demais instâncias, apesar de não encontrar soluções ótimas, o algoritmo desenvolvido foi capaz de encontrar soluções com um gap médio de 14,7%, 21,5%, 34,7%, 46,3% e 46,8% para as instâncias com 9, 11, 13, 15 e 17 vértices, respectivamente.

Para a realização de trabalhos futuros, deve-se considerar a utilização de métodos que otimizem a implementação das restrições de fluxo da formulação proposta, visando a redução do tempo computacional necessário para resolução das instâncias. Além disso, a utilização de métodos heurísticos para fornecer um *upper bound* inicial e facilitar a resolução do problema.

REFERÊNCIAS

AGATZ, N.; BOUMAN, P.; SCHMIDT, M. Optimization approaches for the traveling salesman problem with drone. **SSRN Electronic Journal**, v. 32, 2015.

APPLEGATE, D. L.; BIXBY, R. E.; CHVATáL, V.; COOK, W. J. **The Traveling Salesman Problem: A Computational Study**. Princeton University Press, 2006. ISBN 9780691129938. Disponível em: http://www.jstor.org/stable/j.ctt7s8xg.

APPLEGATE, D. L.; BIXBY, R. E.; CHVáTAL, V.; COOK, W. J. **The Traveling Salesman Problem: A Computational Study**. Princeton, New Jersey: Princeton University Press, 2007.

AYHAN, B.; KWAN, C.; BUDAVARI, B.; LARKIN, J.; GRIBBEN, D. Preflight contingency planning approach for fixed wing uavs with engine failure in the presence of winds. **Sensors (Switzerland)**, v. 19, 2019.

BABEL, L. New heuristic algorithms for the dubins traveling salesman problem. **Journal of Heuristics**, v. 26, p. 503–530, 2020.

BALKCOM, D.; FURTUNA, A.; WANG, W. The dubins car and other arm-like mobile robots. In: **2018 IEEE International Conference on Robotics and Automation (ICRA)**. Brisbane, QLD, Australia: [s.n.], 2018. p. 380–386.

BEHDANI, B.; SMITH, J. An integer-programming-based approach to the close-enough traveling salesman problem. **INFORMS Journal on Computing**, v. 26, p. 415–432, 2014.

BELOTTI, P.; KIRCHES, C.; LEYFFER, S.; LINDEROTH, J.; LUEDTKE, J.; MAHAJAN, A. Mixed-integer nonlinear optimization. **Acta Numerica**, Cambridge University Press, p. 1–131, 2013.

BETTS, J. T. **Practical Methods for Optimal Control using Nonlinear Programming**. Society for Industrial and Applied Mathematics, Philadelphia, PA.: Advances in design and control, 2001.

BRASIL, C.; PANSONATO, R. **Logística dos canais de distribuição**. 1. ed. [S.l.]: Editora Intersaberes, 2018.

BUI, X.-N.; BOISSONNAT, J.-D. Accessibility region for a car that only moves forwards along optimal paths. 03 1994.

CONS, M.; SHIMA, T.; DOMSHLAK, C. Integrating task and motion planning for unmanned aerial vehicles. In: . [S.l.: s.n.], 2011. v. 2, p. 1181–1200. ISBN 9781617824012.

____. ____. In: . [S.l.: s.n.], 2014. v. 2, p. 1181–1200. ISBN 9781617824012.

COUTINHO, W. P. **Unmanned Aerial Vehicle Routing and Trajectory Optimisation Problems**. Tese (Doutorado) — University of Southampton, 2018.

COUTINHO, W. P.; BATTARRA, M.; FLIEGE, J. The unmanned aerial vehicle routing and trajectory optimisation problem, a taxonomic review. **Computers Industrial Engineering**, v. 120, p. 116–128, 2018.

DANTZIG, M. N. T. G. B. **Linear Programming 1: Introduction**. NY: Springer New York, 1997.

DING, Y.; XIN, B.; ZHANG, H.; CHEN, J. A memetic algorithm for curvature-constrained path planning of messenger uav in air-ground coordination. In: . [S.l.: s.n.], 2020. v. 2020-Octob, p. 1465–1472. ISBN 9781728185262.

DUBINS, L. E. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. **American Journal of Mathematics**, v. 79, no. 3, p. 497–516, 1957.

DUMITRESCU, A.; MITCHELL, J. S. Approximation algorithms for tsp with neighborhoods in the plane. **Journal of Algorithms**, v. 48, p. 135–159, 2003.

ELMESEIRY, N.; ALSHAER, N.; ISMAIL, T. A detailed survey and future directions of unmanned aerial vehicles (uavs) with potential applications. **Aerospace**, v. 8, p. 363, 2021.

ERNEST, N.; COHEN, K. Fuzzy clustering based genetic algorithm for the multi-depot polygon visiting dubins multiple traveling salesman problem. In: . [S.l.: s.n.], 2012. ISBN 9781600869396.

FAIGL, J. Unsupervised learning-based solution of the close enough dubins orienteering problem. **Neural Computing and Applications**, Springer Science and Business Media Deutschland GmbH, v. 32, p. 18193–18211, 12 2020. ISSN 14333058.

FAIGL, J.; PěNIčKA, R. **On Close Enough Orienteering Problem with Dubins Vehicle**.

____. On close enough orienteering problem with dubins vehicle. In: . [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2017. p. 5646–5652. ISBN 978-1-5386-2682-5. ISSN 2153-0866.

FAIGL, J.; VANA, P. Surveillance planning with bézier curves. **IEEE Robotics and Automation Letters**, v. 3, p. 750–757, 2018.

FLOUDAS, C. A. Generalized benders decomposition. In: ____. **Encyclopedia of Optimization**. Boston, MA: Springer US, 2009. p. 1162–1175. ISBN 978-0-387-74759-0. Disponível em: https://doi.org/10.1007/978-0-387-74759-0_201.

GARRIDO, S.; MUñOZ, J.; LóPEZ, B.; QUEVEDO, F.; MONJE, C.; MORENO, L. Fm² path planner for uav applications with curvature constraints: A comparative analysis with other planning approaches. **Sensors**, v. 22, 2022.

GENTILINI, I.; MARGOT, F.; SHIMADA, K. The travelling salesman problem with neighbourhoods: Minlp solution. **Optimization Methods and Software**, v. 28, p. 364–378, 2013.

GEOFFRION, A. M. Generalized benders decomposition. **Journal of Optimization Theory and Applications**, v. 10, p. 237–260, 10 1997.

GOLDBARG, M. C.; LUNA, H. P. L. **Otimização Combinatória e Programação Linear: Modelos e Algoritmos**. Rio de Janeiro: Editora Campus, 2000.

HAMEED, I. Coverage path planning software for autonomous robotic lawn mower using dubins' curve. In: . [S.l.: s.n.], 2017.

HELSGAUN, K. An effective implementation of the lin–kernighan traveling salesman heuristic. **European Journal of Operational Research**, v. 126, p. 106–130, 2000.

HILLIER, G. J. L. F. S. **Introdução à Pesquisa Operacional**. [S.l.]: AMGH, 2013.

HU, L.; XI, B.; YI, G.; ZHAO, H.; ZHONG, J. A multiple heterogeneous uavs reconnaissance mission planning and re-planning algorithm. **Journal of Systems Engineering and Electronics**, v. 33, n. 6, p. 1190–1207, 2022.

ISAACS, J. T.; HESPANHA, J. P. Dubins traveling salesman problem with neighborhoods: A graph-based approach. **Algorithms**, v. 6, p. 84–99, 2013. ISSN 19994893.

ISAACS, J. T.; KLEIN, D. J.; HESPANHA, J. P. Algorithms for the traveling salesman problem with neighborhoods involving a dubins vehicle. In: **Proceedings of the 2011 American Control Conference**. San Francisco, CA, USA: [s.n.], 2011. p. 1704–1709.

KANISTRAS, K.; MARTINS, G.; RUTHERFORD, M. J.; VALAVANIS, K. P. A survey of unmanned aerial vehicles (uavs) for traffic monitoring. In: **2013 International Conference on Unmanned Aircraft Systems (ICUAS)**. [S.l.: s.n.], 2013. p. 221–234.

KIM, J.; KIM, S.; JU, C.; SON, H. I. Unmanned aerial vehicles in agriculture: A review of perspective of platform, control, and applications. **IEEE Access**, v. 7, p. 105100–105115, 2019.

KLEMAS, V. V. Coastal and Environmental Remote Sensing from Unmanned Aerial Vehicles: An Overview. **Journal of Coastal Research**, v. 31, p. 1260 – 1267, 2015.

KRONQVIST, J.; BERNAL, D. E.; LUNDELL, A.; GROSSMANN, I. E. A review and comparison of solvers for convex minlp. **Optimization and Engineering**, v. 20, p. 397–450, 2019.

KUCEROVA, K.; VANA, P.; FAIGL, J. Variable-speed traveling salesman problem for vehicles with curvature constrained trajectories. In: . [S.l.: s.n.], 2021. p. 4714–4719. ISBN 9781665417143.

LAND, A. H.; DOIG, A. G. An automatic method of solving discrete programming problems. **Econometrica**, [Wiley, Econometric Society], v. 28, n. 3, p. 497–520, 1960. ISSN 00129682, 14680262. Disponível em: http://www.jstor.org/stable/1910129.

LI, S.; DING, M.; CAI, C. Path planning using fmm with direction and curvature constrained. In: . [S.l.: s.n.], 2009. v. 7498. ISBN 9780819478092.

LIM, C.-W.; PARK, S.; RYOO, C.-K.; CHOI, K.; CHO, J.-H. A path planning algorithm for surveillance uavs with timing mission constrains. In: . [S.l.: s.n.], 2010. p. 2371–2375. ISBN 9781424474530.

LUENBERGER, D. G.; YE, Y. **Linear and Nonlinear Programming**. [S.l.]: Springer Publishing Company, Incorporated, 2015. ISBN 3319188410.

MACHMUDAH, A.; SHANMUGAVEL, M.; PARMAN, S.; MANAN, T. S. A.; DUTYKH, D.; BEDDU, S.; RAJABI, A. Flight trajectories optimization of fixed-wing uav by bank-turn mechanism. **Drones**, v. 6, p. 69, 2022.

MATAI, R.; SINGH, S.; MITTAL, M. L. Traveling salesman problem: an overview of applications, formulations, and solution approaches. In: DAVENDRA, D. (Ed.). **Traveling Salesman Problem**. Rijeka: IntechOpen, 2010. cap. 1. Disponível em: https://doi.org/10.5772/12909.

MITCHELL, J. E. Integer programming: Branch and cut algorithms. In: ____. **Encyclopedia of Optimization**. Boston, MA: Springer US, 2009. p. 1643–1650. ISBN 978-0-387-74759-0. Disponível em: https://doi.org/10.1007/978-0-387-74759-0_287.

____. Integer programming: Cutting plane algorithms. In: ____. **Encyclopedia of Optimization**. Boston, MA: Springer US, 2009. p. 1650–1657. ISBN 978-0-387-74759-0. Disponível em: https://doi.org/10.1007/978-0-387-74759-0_288.

MURRAY, C.; CHU, A. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. **Transportation Research Part C: Emerging Technologies**, v. 54, 2015.

NOON, C.; BEAN, J. An efficient transformation of the generalized traveling salesman problem. **INFOR. Information Systems and Operational Research**, v. 31, 1993.

NOON, C. E.; BEAN, J. C. A lagrangian based approach for the asymmetric generalized traveling salesman problem. **Operations Research**, Institute for Operations Research and the Management Sciences (INFORMS), v. 39, p. 623–632, 8 1991. ISSN 0030-364X.

NY, J. L.; FRAZZOLI, E.; FERON, E. **The Curvature-Constrained Traveling Salesman Problem For High Point Densities**. New Orleans, LA, USA: IEEE, 2007.

OBERLIN, P.; RATHINAM, S.; DARBHA, S. Combinatorial motion planning for a dubins vehicle with precedence constraints. In: . [S.l.: s.n.], 2010. p. 1627–1634. ISBN 9780791848920.

OBERMEYER, K. Path planning for a uav performing reconnaissance of static ground targets in terrain. In: . [S.l.: s.n.], 2009. ISBN 9781563479786.

OBERMEYER, K.; OBERLIN, P.; DARBHA, S. Sampling-based roadmap methods for a visual reconnaissance uav. In: . [S.l.: s.n.], 2010. ISBN 9781600869624.

____. Sampling-based path planning for a visual reconnaissance unmanned air vehicle. **Journal of Guidance, Control, and Dynamics**, v. 35, p. 619–631, 2012.

PENICKA, R.; SASKA, M.; REYMANN, C.; LACROIX, S. Reactive dubins traveling salesman problem for replanning of information gathering by uavs. In: . [S.l.: s.n.], 2017. ISBN 9781538610961.

POIKONEN, S.; WANG, X.; GOLDEN, B. The vehicle routing problem with drones: Extended models and connections. **Networks**, v. 70, 2017.

PěNIčKA, R.; FAIGL, J.; SASKA, M.; VáňA, P. Data collection planning with non-zero sensing distance for a budget and curvature constrained unmanned aerial vehicle. **Autonomous Robots**, v. 43, p. 1937–1956, 2019.

SHIVAM, A.; RATNOO, A. Curvature-constrained vector field for path following guidance. In: . [S.l.: s.n.], 2021. p. 853–857. ISBN 9780738131153.

SOUSA, J. Vasconcelos Reinolds de; GAMBOA, P. V. Aerial forest fire detection and monitoring using a small uav. **KnE Engineering**, v. 5, p. 242–256, 2020.

TAHA, H. A. **Operations Research An Introduction, 10th edition**. [S.l.]: Pearson Education, 2016.

VANA, P.; FAIGL, J. On the dubins traveling salesman problem with neighborhoods. In: **2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. Hamburg, Germany: [s.n.], 2015. p. 4029–4034.

VáňA, P.; SLáMA, J.; FAIGL, J. The dubins traveling salesman problem with neighborhoods in the three-dimensional space. In: . [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2018. p. 374–379. ISBN 9781538630815. ISSN 10504729.

WILBURN, J.; COLE, J.; PERHINSCHI, M.; WILBURN, B. Comparison of a fuzzy logic controller to a potential field controller for real-time uav navigation. In: . [S.l.: s.n.], 2012. ISBN 9781600869389.

WINSTON, S. C. A. W. L. **Practical Management Science**. 4. ed. Mason, OH: South-Western Cengage Learning, 2012.

WOLEK, A.; MCMAHON, J.; DZIKOWICZ, B. R.; HOUSTON, B. H. The orbiting dubins traveling salesman problem: planning inspection tours for a minehunting auv. **Autonomous Robots**, Springer, v. 45, p. 31–49, 1 2021. ISSN 15737527.

XU, M.; DOU, L.; XIN, B.; WANG, Y.; FANG, H.; CAI, T. **Curvature-Constrained UAV Path Planningin Tracking a Moving Air Target**. Anchorage, AK, USA: IEEE, 2018.

YANG, K.; SUKKARIEH, S. Real-time continuous curvature path planning of uavs in cluttered environments. In: . [S.l.: s.n.], 2008. ISBN 9781424420346.

YOKOYAMA, N.; OCHI, Y. Optimal path planning for skid-to-turn unmanned aerial vehicle. In: . [S.l.: s.n.], 2008. ISBN 9781563479458.

____. Path planning algorithms for skid-to-turn unmanned aerial vehicles. **Journal of Guidance, Control, and Dynamics**, v. 32, p. 1531–1543, 2009.

YULONG, D.; BIN, X.; JIE, C.; HAO, F.; YANGGUANG, Z.; GUANQIANG, G.; LIHUA, D. Path planning of messenger uav in air-ground coordination. In: . [S.l.: s.n.], 2017. v. 50, p. 8045–8051.

ZHANG, X.; CHEN, J.; XIN, B.; PENG, Z. A memetic algorithm for path planning of curvature-constrained uavs performing surveillance of multiple ground targets. **Chinese Journal of Aeronautics**, v. 27, p. 622–633, 2014.

ZHU, K.; XIE, Z.; DAI, L. Two layers solving strategy for path planning of unmanned aerial vehicle. In: . [S.l.: s.n.], 2016. v. 1, p. 69–73. ISBN 9781509007684.

ŽIVOJEVIć, D.; VELAGIć, J. Path planning for mobile robot using dubins-curve based rrt algorithm with differential constraints. In: **2019 International Symposium ELMAR**. University of Maryland, College Park.: [s.n.], 2019. p. 139–142.

APÊNDICES

APÊNDICE A – RESULTADO DOS TESTES COMPUTACIONAIS.

Tabela 8 – Testes com instâncias com 7 vértices.

Instância	Vértices	Objetivo (s)	Gap	Tempo (s)	Soluções	Status
DTSPN-6-5	7	33,671	0,000	133,481	2	Optimal
DTSPN-6-6	7	49,550	0,000	101,914	4	Optimal
DTSPN-6-7	7	53,746	0,000	52,276	3	Optimal
DTSPN-6-8	7	40,464	0,000	112,086	4	Optimal
DTSPN-6-9	7	44,049	0,000	140,781	4	Optimal
DTSPN-6-10	7	54,912	0,000	367,187	4	Optimal
DTSPN-6-11	7	44,453	0,000	41,747	3	Optimal
DTSPN-6-12	7	47,986	0,000	168,549	7	Optimal
DTSPN-6-13	7	37,673	0,000	69,221	2	Optimal
DTSPN-6-14	7	41,313	0,000	25,787	4	Optimal
DTSPN-6-15	7	53,955	0,000	305,196	8	Optimal
DTSPN-6-16	7	39,718	0,000	377,521	2	Optimal
DTSPN-6-17	7	32,931	0,000	59,689	3	Optimal
DTSPN-6-18	7	37,707	0,000	3,509	2	Optimal
DTSPN-6-19	7	41,326	0,000	20,900	2	Optimal
DTSPN-6-20	7	46,256	0,000	52,034	2	Optimal
DTSPN-6-21	7	55,133	0,000	144,698	5	Optimal
DTSPN-6-22	7	47,164	0,000	161,381	6	Optimal
DTSPN-6-23	7	47,739	0,000	144,167	5	Optimal
DTSPN-6-24	7	54,473	0,000	88,608	3	Optimal
DTSPN-6-25	7	47,109	0,000	124,573	5	Optimal
DTSPN-6-26	7	36,002	0,000	10,288	2	Optimal
DTSPN-6-27	7	44,762	0,000	46,460	1	Optimal
DTSPN-6-28	7	51,753	0,000	40,455	4	Optimal
DTSPN-6-29	7	78,230	0,000	283,680	5	Optimal

Fonte: Autoria própria (2023)

Tabela 9 – Testes com instâncias com 9 vértices.

Instância	Vértices	Objetivo (s)	Gap	Tempo (s)	Soluções	Status
DTSPN-8-2	9	36,321	0	74,491	2	Optimal
DTSPN-8-3	9	59,941	0,117	3689,165	3	Tempo Limite
DTSPN-8-4	9	56,038	0,141	3608,293	7	Tempo Limite
DTSPN-8-5	9	51,784	0	971,201	4	Optimal
DTSPN-8-6	9	45,978	0	2000,083	3	Optimal
DTSPN-8-7	9	44,107	0	1676,677	3	Optimal
DTSPN-8-8	9	56,157	0	2466,719	8	Optimal
DTSPN-8-9	9	56,966	0	359,869	6	Optimal
DTSPN-8-10	9	46,58	0,118	3645,617	4	Tempo Limite
DTSPN-8-11	9	46,752	0	1553,328	8	Optimal
DTSPN-8-12	9	45,908	0	1327,733	4	Optimal
DTSPN-8-13	9	41,062	0	907,866	4	Optimal
DTSPN-8-14	9	60,891	0	603,177	4	Optimal
DTSPN-8-15	9	48,961	0	1625,363	5	Optimal
DTSPN-8-16	9	57,065	0	322,876	5	Optimal
DTSPN-8-17	9	38,385	0	799,722	3	Optimal
DTSPN-8-18	9	56,889	0	537,453	3	Optimal
DTSPN-8-19	9	65,3	0,136	3655,942	2	Tempo Limite
DTSPN-8-20	9	49,986	0	2374,561	6	Optimal
DTSPN-8-21	9	59,534	0,125	3644,13	4	Tempo Limite
DTSPN-8-22	9	59,782	0	696,094	3	Optimal
DTSPN-8-23	9	51,164	0	978,872	1	Optimal
DTSPN-8-24	9	52,841	0	471,177	5	Optimal
DTSPN-8-25	9	60,477	0	51,907	5	Optimal
DTSPN-8-26	9	49,44	0	102,16	1	Optimal
DTSPN-8-27	9	64,005	0,243	3631,792	4	Tempo Limite
DTSPN-8-28	9	47,22	0	636,394	4	Optimal
DTSPN-8-29	9	53,792	0	197,842	5	Optimal

Fonte: Autoria própria (2023)

Tabela 10 – Testes com instâncias com 11 vértices.

Instância	Vértices	Objetivo (s)	Gap	Tempo (s)	Soluções	Status
DTSPN-10-2	11	65,968	0,244	3659,389	3	Tempo Limite
DTSPN-10-3	11	50,007	0,327	3612,481	7	Tempo Limite
DTSPN-10-4	11	55,766	0	3258,104	2	Optimal
DTSPN-10-5	11	56,799	0,208	3612,926	6	Tempo Limite
DTSPN-10-6	11	54,424	0,059	3654,297	7	Tempo Limite
DTSPN-10-7	11	77,786	0,378	3629,105	3	Tempo Limite
DTSPN-10-8	11	69,274	0,414	3696,807	4	Tempo Limite
DTSPN-10-9	11	53,187	0,081	3616,439	7	Tempo Limite
DTSPN-10-10	11	53,796	0,084	3732,857	7	Tempo Limite
DTSPN-10-11	11	46,863	0,079	3634,167	8	Tempo Limite
DTSPN-10-12	11	55,834	0,315	3659,071	4	Tempo Limite
DTSPN-10-13	11	60,955	0	932,171	7	Optimal
DTSPN-10-14	11	88,884	0,576	3616,185	2	Tempo Limite
DTSPN-10-15	11	53,811	0	1794,2	7	Optimal
DTSPN-10-16	11	44,682	0	2333,906	2	Optimal
DTSPN-10-17	11	68,339	0,241	3638,59	6	Tempo Limite
DTSPN-10-18	11	64,114	0,267	3675,104	4	Tempo Limite
DTSPN-10-19	11	58,341	0,094	3741,034	7	Tempo Limite
DTSPN-10-20	11	51,505	0,033	3641,503	8	Tempo Limite
DTSPN-10-21	11	68,918	0,211	3600,391	5	Tempo Limite
DTSPN-10-22	11	67,681	0,355	3667,476	1	Tempo Limite
DTSPN-10-23	11	62,412	0,236	3652,267	5	Tempo Limite
DTSPN-10-24	11	68,044	0,249	3650,889	5	Tempo Limite
DTSPN-10-25	11	57,586	0,116	3606,437	5	Tempo Limite
DTSPN-10-26	11	50,191	0	753,863	6	Optimal
DTSPN-10-27	11	50,873	0,199	3620,5	5	Tempo Limite
DTSPN-10-28	11	53,83	0,16	3607,088	2	Tempo Limite
DTSPN-10-29	11	54,386	0,019	3647,5	7	Tempo Limite

Fonte: Autoria própria (2023)

Tabela 11 – Testes com instâncias com 13 vértices.

Instância	Vértices	Objetivo (s)	Gap	Tempo (s)	Soluções	Status
DTSPN-12-0	13	90,603	0,499	3656,679	3	Tempo Limite
DTSPN-12-1	13	76,767	0,528	3640,716	3	Tempo Limite
DTSPN-12-2	13	82,61	0,26	3618,512	2	Tempo Limite
DTSPN-12-3	13	75,803	0,467	3601,497	3	Tempo Limite
DTSPN-12-4	13	51,067	0,079	3633,928	7	Tempo Limite
DTSPN-12-5	13	73,878	0,405	3677,934	2	Tempo Limite
DTSPN-12-6	13	78,431	0,417	3619,761	2	Tempo Limite
DTSPN-12-8	13	65,861	0,304	3620,744	2	Tempo Limite
DTSPN-12-9	13	55,806	0,323	3601,857	3	Tempo Limite
DTSPN-12-10	13	83,247	0,325	3610,704	1	Tempo Limite
DTSPN-12-11	13	92,958	0,49	3633,958	5	Tempo Limite
DTSPN-12-12	13	73,171	0,454	3647,623	2	Tempo Limite
DTSPN-12-13	13	75,514	0,414	3619,87	5	Tempo Limite
DTSPN-12-14	13	74,547	0,36	3658,791	4	Tempo Limite
DTSPN-12-15	13	83,708	0,414	3612,79	3	Tempo Limite
DTSPN-12-16	13	74,778	0,302	3696,385	2	Tempo Limite
DTSPN-12-17	13	79,79	0,4	3675,493	2	Tempo Limite
DTSPN-12-18	13	63,729	0,28	3614,803	3	Tempo Limite
DTSPN-12-19	13	60,877	0,306	3600,504	4	Tempo Limite
DTSPN-12-21	13	72,651	0,416	3618,507	3	Tempo Limite
DTSPN-12-22	13	61,851	0,179	3666,109	3	Tempo Limite
DTSPN-12-24	13	55,716	0,127	3614,634	1	Tempo Limite
DTSPN-12-25	13	85,018	0,511	3644,134	2	Tempo Limite
DTSPN-12-26	13	55,702	0,313	3635,843	5	Tempo Limite
DTSPN-12-27	13	77,887	0,441	3686,75	8	Tempo Limite
DTSPN-12-28	13	68,871	0,332	3676,604	5	Tempo Limite
DTSPN-12-29	13	47,192	0,033	3733,969	2	Tempo Limite

Fonte: Autoria própria (2023)

Tabela 12 – Testes com instâncias com 15 vértices.

Instância	Vértices	Objetivo (s)	Gap	Tempo (s)	Soluções	Status
DTSPN-14-0	15	92,482	0,453	3627,283	2	Tempo Limite
DTSPN-14-1	15	92,456	0,559	3664,5	1	Tempo Limite
DTSPN-14-2	15	86,97	0,46	3615,789	6	Tempo Limite
DTSPN-14-3	15	113,344	0	848,947	10	Optimal
DTSPN-14-4	15	74,034	0,464	3638,329	3	Tempo Limite
DTSPN-14-5	15	82,776	0,548	3601,455	3	Tempo Limite
DTSPN-14-7	15	71,192	0,299	3670,981	2	Tempo Limite
DTSPN-14-9	15	67,284	0,342	3623,765	3	Tempo Limite
DTSPN-14-10	15	84,899	0,539	3615,306	2	Tempo Limite
DTSPN-14-11	15	102,657	0,579	3663,216	1	Tempo Limite
DTSPN-14-13	15	59,847	0,139	3606,505	3	Tempo Limite
DTSPN-14-14	15	84,431	0,354	3643,371	1	Tempo Limite
DTSPN-14-15	15	75,161	0,214	3643,636	6	Tempo Limite
DTSPN-14-17	15	99,847	0,593	3631,783	2	Tempo Limite
DTSPN-14-18	15	96,167	0,517	3634,593	1	Tempo Limite
DTSPN-14-19	15	88,194	0,476	3626,565	3	Tempo Limite
DTSPN-14-21	15	72,054	0,395	3623,527	4	Tempo Limite
DTSPN-14-22	15	126,121	0,614	3677,314	1	Tempo Limite
DTSPN-14-23	15	89,628	0,547	3639,61	1	Tempo Limite
DTSPN-14-24	15	104,887	0,677	3613,4	1	Tempo Limite
DTSPN-14-25	15	114,912	0,5	3615,593	3	Tempo Limite
DTSPN-14-26	15	95,162	0,441	3605,484	1	Tempo Limite
DTSPN-14-28	15	75,866	0,332	3601,453	6	Tempo Limite
DTSPN-14-29	15	100,326	0,598	3605,468	2	Tempo Limite

Fonte: Autoria própria (2023)

Tabela 13 – Testes com instâncias com 17 vértices.

Instância	Vértices	Objetivo (s)	Gap	Tempo (s)	Soluções	Status
DTSPN-16-1	17	108,925	0,486	3647,938	1	Tempo Limite
DTSPN-16-3	17	83,747	0,484	3630,513	1	Tempo Limite
DTSPN-16-11	17	78,105	0,384	3605,971	3	Tempo Limite
DTSPN-16-12	17	108,89	0,448	3610,795	2	Tempo Limite
DTSPN-16-13	17	110,468	0,548	3600,855	4	Tempo Limite
DTSPN-16-16	17	74,67	0,433	3623,678	2	Tempo Limite
DTSPN-16-17	17	78,897	0,421	3609,318	3	Tempo Limite
DTSPN-16-18	17	103,755	0,558	3653,386	3	Tempo Limite
DTSPN-16-20	17	88,617	0,452	3649,674	3	Tempo Limite

Fonte: Autoria própria (2023)