Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Detecção e Resolução de Conflitos em ATM Utilizando Modelagens de Trajetórias 4D Baseadas em Banco de Dados NoSQL e Algoritmos de Busca

Lucas Borges Monteiro

Tese apresentada como requisito parcial para
conclusão do Doutorado em Informática

Orientador
Prof. Dr. Li Weigang

Brasília
2023

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Detecção e Resolução de Conflitos em ATM Utilizando Modelagens de Trajetórias 4D Baseadas em Banco de Dados NoSQL e Algoritmos de Busca

Lucas Borges Monteiro

Tese apresentada como requisito parcial para
conclusão do Doutorado em Informática

Prof. Dr. Li Weigang (Orientador)
CIC/UnB

Prof. Dr. Claudio Barbieri da Cunha
POLI/USP

Prof. Dr. Zhao Liang
FFCLRP/USP

Prof. Dr. Vinicius Ruela Pereira Borges
CIC/UnB

Prof. Dr. Geraldo Rocha Filho
DCET/UESB

Prof. Dr. Ricardo Pezzuol Jacobi
Coordenador do Programa de Pós-graduação em Informática

Brasília, 26 de maio de 2023

Dedicatória

Dedico este trabalho à minha filha Maria Eduarda, ao meu filho Bruno e à minha esposa Leiliane por toda força, carinho e compreensão. À minha mãe que me ensinou desde cedo o valor da educação. E à minha irmã Paula e meu cunhado Ricardo que são exemplos de dedicação à carreira acadêmica.

Agradecimentos

Agradeço primeiramente à Deus pelo dom da vida.

Ao professor Li Weigang, pela orientação e motivação ao longo do mestrado e do doutorado.

Ao mestrando Cristiano Perez Garcia que contribuiu enormemente com o trabalho. Ao colega Vitor Filincowsky Ribeiro que foi fundamental para definição dos rumos desta pesquisa, mesmo estando dedicado ao término do seu doutorado.

Às professoras e professores do CIC/UnB, pelos ensinamentos ao longo dos últimos anos e, em especial, ao professor Geraldo Pereira Rocha Filho, pelo acompanhamento durante as atividades da pesquisa.

E, por fim, à toda minha família, que foi fundamental em todos os momentos.

Resumo

Os avanços tecnológicos aumentaram significativamente a quantidade de dados gerados em vários domínios, incluindo o transporte aéreo. O tratamento correto desse grande volume de dados pode trazer resultados importantes, pois torna a tomada de decisão mais precisa. Neste sentido, com foco no novo paradigma de Operações Baseadas em Trajetórias (TBO) do Gerenciamento de Tráfego Aéreo (ATM), este trabalho apresenta duas modelagens para detecção e resolução de conflitos (CDR): a primeira baseada em banco de dados NoSQL e algoritmos de busca; e a segunda, denominada `4DNavMCTS`, que aplica os conceitos de *Monte Carlo Tree Search* (MCTS) e Modelos de Espaço Vetorial (MEV) também a uma modelagem baseada em banco de dados NoSQL.

Os principais objetivos alcançados com a pesquisa foram: i) aumento da segurança para o ATM; ii) processamento de grandes volumes de dados gerados pelo tráfego regional e global em navegação 4D; iii) tratamento de incertezas de fatores humanos e ambientais, como clima e temperatura; e iv) gerenciamento das trajetórias de forma a garantir cenários livres de conflito, mesmo com eventuais intervenções do próprio modelo.

Os testes mostraram que a primeira modelagem possibilitou a identificação, no pior caso, de todos os conflitos em no máximo 1,5 milissegundos, e a resolução de todos os conflitos simulados, no cenário mais complexo, isto é, 30 aeronaves com 30 trajetórias para cada aeronave (espaço de busca de 10^{44} estados possíveis), em 47,6 segundos. Já a segunda abordagem também obteve resultados satisfatórios, com eliminação, em média, de 98,95% das zonas de conflito, com aproximadamente 1/4 das intervenções no espaço aéreo em comparação com o método alternativo.

Palavras-chave: Trajetórias 4D, Detecção e Resolução de Conflitos, Algoritmos de Busca, Not Only SQL, Monte Carlo Tree Search.

Abstract

The progress of science and technology has greatly increased the amount of data generated in various fields, including air transportation. Dealing with these massive data correctly will bring important results, because it can make decision-making more accurate. In this sense, focusing on the new paradigm of Trajectory-Based Operations (TBO) of Air Traffic Management (ATM), this work presents two models for conflict detection and resolution (CDR). The first is based on NoSQL database and search algorithms. The second one is called `4DNavMCTS`, which also applies the concepts of Monte Carlo Tree Search (MCTS) and Vector Space Model (MEV) to modeling based on a NoSQL database.

The main objectives achieved with the research were: i) increased security for the ATM; ii) processing a large amount of data generated by regional and global traffic in four-dimensional navigation; iii) dealing with uncertainties of human and environmental factors such as climate and temperature; and iv) trajectory management to ensure conflict-free scenes, even though the model itself occasionally interferes.

The tests shows that the first modeling can identify all conflicts within 1.5 milliseconds in the worst case, and solve all simulated conflicts within 47.6 seconds in the most complicated cases, which include 30 planes with 30 trajectories (search space is 10^{44} possible states). The second model, `4DNavMCTS`, has also achieved good results, eliminating 98.95% of the conflict zones on average. Compared with the alternative method, the intervention in the airspace is about 1/4.

Keywords: 4 Dimensional Trajectory, Conflict Detection and Resolution, Search Algorithms, Not Only SQL, Monte Carlo Tree Search.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

4D Trajetórias Quadridimensionais.

ACAS Sistema Automático de Prevenção de Colisão Aérea (*Automatic Air Collision Avoidance System*).

ACC Centro de Controle de Área (*Area Control Center*).

ADSB Sistema de Vigilância Dependente Automática (*Automatic Dependent Surveillance - Broadcast*).

AI Inteligência Artificial (*Artificial Inteligence*).

AIDL *Aircraft Intent Description Language.*

ASM Gerenciamento do Espaço Aéreo (*Airspace Management*).

ATC Controle de Tráfego Aéreo (*Air Traffic Controller*).

ATFM Gerenciamento do Fluxo de Tráfego Aéreo (*Air Traffic Flow Management).*

ATM Gerenciamento do Tráfego Aéreo (*Air Traffic Management*).

BFS Busca em Largura (*Breadth First Search*).

BIP Programação Inteira Binária.

BSON JSON Binário (*Binary JSON*).

CDR Detecção e Resolução de Conflitos (*Conflict Detection and Resolution*).

CGNA Centro de Gerenciamento de Navegação Aérea.

COMAER Comando da Aeronáutica.

CPVR Centro de Plano de Vôo Repetido.

CQL *Cassandra Query Language.*

DECEA Departamento de Controle do Espaço Aéreo.

DFS Busca em Profundidade (*Depth First Search*).

DL Aprendizagem Profunda (*Deep Learning*).

DNN Redes Neurais Profundas (*Deep Neural Networks*).

FIR Regiões de Informações de Voo.

FIXM *Flight Information Exchange Model.*

FMS Sistemas de Gerenciamento de Voo (*Flight Management Systems*).

GNSS Sistema Global de Navegação por Satélite (*Global Navigation Satellite System*).

GPS Sistema de Posicionamento Global (*Global Positioning System*).

HMM Modelo Oculto de Markov (*Hidden Markov Model*).

ICAO Organização Internacional da Aviação Civil (*International Civil Aviation Organization*).

IFM Gerenciamento do Fluxo de Mensagens (*Information Flow Management).*

ILP Programação Linear Inteira (*Integer Linear Programing*).

JSON Notação de Objetos JavaScrip (*JavaScript Object Notation*).

KML *Keyhole Markup Language.*

LSTM Redes Neurais Profundas de Memória de Curto Prazo (*Long Short-Term Memory* Neural Networks).

MA Algoritmo Memético (*Memetic Algorithm*).

MCTS Árvore de Busca de Monte Carlo (*Monte Carlo Tree Search*).

MDP Processo de Decisão de Markov (*Markov Decision Process*).

MEV Modelos de Espaço Vetorial.

NoSQL Não-SQL (*Not Only SQL*).

OR Pesquisa Operacional (*Operational Research*).

ORCA *Optimal Reciprocal Collision Avoidance.*

PBN Navegação Baseada em Performance (*Performance-Based Navigation*).

PDF *Portable Document Format.*

RNA Redes Neurais Artificiais.

RPL Plano de Voo Repetido (*Repeated Flight Plan*).

RVSM *Reduced Vertical Separation Minimum.*

SA Recozimento Simulado (*Simulated Annealing*).

SISCEAB Sistema de Controle do Espaço Aéreo Brasileiro.

SQL Linguagem de Consulta Estruturada (*Structured Query Language*).

SWIM *System Wide Information Management.*

TBO Operação Baseada em Trajetória (*Trajectory-Based Operation*).

TFM Gerenciamento de Fluxo de Tráfego (*Traffic Flow Management*).

TP Previsão de Trajetória (*trajectory prediction*).

UAM Mobilidade Aérea Urbana (*Urban Air Mobility*).

XML *Extensible Markup Language.*

ZEE Zona Econômica Exclusiva.

Capítulo 1

Introdução

Em uma ampla variedade de áreas de aplicação, os dados estão sendo coletados em uma escala sem precedentes. As decisões que costumavam ser baseadas em suposições ou modelos realistas feitos à mão agora podem ser tomadas usando modelos matemáticos orientados por dados. Atualmente, a análise de *big data* promoveu quase todos os aspectos da sociedade, incluindo serviços móveis, varejo, manufatura, serviços financeiros e transporte aéreo [4, 5].

É desafiador resolver os problemas relacionados à complexidade e incerteza de uma grande quantidade de dados. Um exemplo disso é o transporte aéreo global, que desempenha um papel vital na promoção do desenvolvimento econômico e social sustentável. Ele, direta e indiretamente, sustenta o emprego de 58,1 milhões de pessoas, contribui com mais de US$ 2,4 trilhões para a economia global e transporta mais de 3,3 bilhões de passageiros e US$ 6,4 trilhões de dólares americanos em mercadorias todos os anos. Desde 1977, a escala do tráfego aéreo global dobrou a cada 15 anos e provavelmente continuará a dobrar. Essa tendência foi interrompida pela pandemia de Coronavírus em 2020, mas até o final de 2021, a demanda de tráfego aéreo atingiu ou superou o nível de 2019. Embora o ciclo de demanda seja mais abrangente, esse tipo de crescimento ainda aparece e ajuda a explicar que o investimento na aviação é o fator chave para apoiar a recuperação econômica [6]. O crescimento da demanda mostra a necessidade de métodos mais orientados a dados como forma de melhorar a segurança e o desempenho do voo [7].

1.1 Motivação

Um requisito essencial para um Gerenciamento de Tráfego Aéreo (ATM) eficiente é a qualidade das informações disponíveis, que devem ser atualizadas, altamente precisas e confiáveis. Isso permite que o usuário tome as decisões certas no momento certo [8], suportando, assim, a previsão de cenários, alocação de recursos e gerenciamento de trajetória.

Neste contexto, é fundamental o desenvolvimento de novas soluções de gestão do tráfego aéreo. Por isso, diferentes programas estão sendo desenvolvidos pela Organização Internacional de Aviação Civil (ICAO), com foco no conceito de gestão global do tráfego aéreo, buscando melhorar a segurança e a eficiência de todo o transporte aéreo [9].

Dentre os objetivos específicos de cada projeto, destaca-se o propósito comum de otimizar as chegadas nos aeroportos por meio do conceito emergente de Operação Baseada em Trajetória (TBO). O TBO é inspirado no conhecimento e compartilhamento das posições atuais e planejadas das aeronaves, que estão restritas a uma sequência de pontos no espaço-tempo, ou seja, um espaço quadridimensional (latitude x, longitude y, altitude z e tempo t) [10]. No TBO, todo voo tem uma trajetória compartilhada e gerenciada, que é utilizada como um plano de voo comum, melhorando, assim, a tomada de decisão dos operadores de aeronaves e prestadores de serviços de navegação aérea [11]. A representação quadridimensional de trajetórias, juntamente com diversas variáveis relacionadas, gera uma grande quantidade de dados e se configura como um exemplo óbvio de um problema relacionado ao *big data*.

1.2 Definição do Problema e Objetivos da Pesquisa

Trajetórias Quadridimensionais (4D) e Navegação Baseada em Desempenho (PBN) são temas de pesquisa que ganharam destaque desde que as autoridades do espaço aéreo em todo o mundo estabeleceram diretrizes para implementar essas tecnologias. O escopo deste trabalho é o desenvolvimento de um sistema inteligente que deve apoiar a operação do Controle de Tráfego Aéreo (ATC) para manter um ambiente de transporte aéreo seguro e eficiente. Nossa pesquisa visa desenvolver um framework para Detecção e Resolução de Conflitos (CDR) em rotas 4D selecionadas. A estrutura proposta utiliza bancos de dados NoSQL e algoritmos de busca para gerenciar as trajetórias, atendendo ao paradigma SWIM.

A adoção da navegação 4D na implementação do paradigma TBO aumenta o desafio para CDR. Embora esta tarefa seja um tema tradicional em ATM [12], CDR pode ser dividido em dois cenários principais: sem TBO e com TBO. O procedimento geral do CDR pode ser descrito em quatro passos. O primeiro passo (1) é prever e apresentar os dados da trajetória 4D, que podem ser estimados a partir de um preditor de trajetória. O segundo passo (2) é detectar os conflitos. Havendo conflito, o procedimento é encaminhado para o terceiro passo (3) para resolução do conflito. Como resultado do passo (3), os dados de trajetória 4D ajustados são propagados de volta para o passo (1) até que não haja conflito. Em seguida, o procedimento é encaminhado para o quarto passo (4) para gerar os dados da trajetória 4D livre de conflitos.

Os seguintes aspectos representam um desafio significativo em CDR com a implementação do TBO e, consequentemente, são objetivos específicos deste trabalho:

- maior requisito de segurança para ATM [13];
- volume massivo de dados gerados pelo tráfego regional e global em navegação 4D [14];
- incerteza de fatores humanos e ambientais, como clima e temperatura [10];
- computação e gerenciamento repetidos para alcançar cenários livres de conflito [15].

Existem algumas tentativas, incluindo o uso de unidades de processamento gráfico (GPU) para melhorar o desempenho da computação [16], mas essa limitação permanece sem solução desde a declaração do problema em 1997 [17]. No entanto, o problema crítico é a alta complexidade computacional para detecção de conflitos, que é aproximadamente $O(n^2)$, onde n é o número de aeronaves em um determinado momento, sendo necessário verificar se todas as aeronaves estão devidamente separadas de um para o outro.

1.3 Abordagem Proposta

O objetivo principal deste trabalho é propor uma nova modelagem para detecção e resolução de conflitos, denominada `4DNavMCTS`, que é utilizada para navegação 4D baseada em algoritmos de busca, especialmente Monte Carlo Tree Search (MCTS). Nesse modelo, o MCTS é utilizado para manipular os pontos das trajetórias a serem analisadas, incluindo as trajetórias padrão e possíveis trajetórias alternativas, possibilitando avaliar a combinação desse massivo número de estados. Quando aplicado ao MCTS, o Modelo de Espaço Vetorial (MEV) compara as estratégias simuladas com a estratégia ideal. Com o modelo proposto, no modo de navegação 4D, espera-se obter uma solução de gerenciamento de tráfego aéreo mais eficaz.

1.4 Contribuições

Como principal contribuição desta pesquisa, desenvolve-se um novo framework de CDR para a gestão da navegação 4D utilizando um esquema particular de uma base de dados NoSQL. Juntamente com o NoSQL, em comparação com outros algoritmos CDR (geralmente usando o método $O(n^2)$ pairwise), o algoritmo proposto usa o MCTS, que permite operar uma grande quantidade de dados reduzir o custo computacional economicamente. Além disso, o framework apresenta dois métodos de resolução de conflitos, que se beneficiam do modelo de detecção de conflitos proposto e encontram as melhores soluções

em tempo satisfatório. A pesquisa mostra que o desenvolvimento da inteligência artificial promove o desenvolvimento sustentável do transporte aéreo inteligente, melhorando efetivamente o gerenciamento do tráfego aéreo.

As contribuições concretas deste estudo são as seguintes:

- com base no MCTS, o `4DNavMCTS` pode lidar satisfatoriamente com problemas complicados de TBO;
- considerando o *big data* da navegação, o `4DNavMCTS` pode realizar detecção e resolução de conflitos (CDR) sob o paradigma da inteligência artificial (AI);
- usando uma solução de tomada de decisão com um algoritmo MCTS, o `4DNavMCTS` pode encontrar e resolver os potenciais conflitos entre aeronaves e melhorar a segurança de voo com previsão razoável;
- adotando o MEV, o `4DNavMCTS` pode ajudar os controladores a escolher quantitativamente a trajetória apropriada e reduzir o risco de conflito.

1.5 Organização da Tese

Nesse trabalho são explorados diversos conceitos, metodologias e tecnologias referentes às diferentes disciplinas envolvidas na solução do problema. Para uma melhor compreensão, o trabalho está organizado da seguinte forma:

- o Capítulo 2 apresenta os conceitos relacionados ao ATM, fundamentais para entendimento do contexto, tais como nagevação 4D, conceitos sobre o voo e dados de navegação;
- no Capítulo 3 são descritas as metodologias da Ciência da Computação adotadas na modelagem da solução, tais como bancos de dados NoSQL, algoritmos de busca e MEV;
- o Capítulo 4 apresenta o estado da arte na pesquisa sobre CDR, dividindo os trabalhos em 3 blocos: soluções que adotam pesquisa operacional; soluções que empregam AI, e; trabalhos que apresentam metodologias comparativas e revisões de diferentes pesquisas;
- o Capítulo 5 apresenta a primeira modelagem do framework, que combina algoritmos de busca com um banco de dados NoSQL;
- no Capítulo 6 é apresentada a evolução do modelo, que integra o algoritmo de busca baseado em MCTS diretamente na modelagem, de forma que a detecção e a resolução de conflitos sejam tratadas de forma concomitante;

- o Capítulo 7 apresenta os casos de teste para as duas abordagens, com as análises dos resultados obtidos em todos os cenários;
- por fim, o Capítulo 8 apresenta as conclusões, revisa os objetivos e indica pontos a serem explorados futuramente.

Capítulo 2

Conceitos Fundamentais em Gerenciamento de Tráfego Aéreo

Esse Capítulo aborda conceitos fundamentais sobre gerenciamento de tráfego aéreo (ATM), especialmente relacionados à navegação 4D e ao problema de detecção e resolução de conflitos (CDR), necessários para entendimento da proposta de solução.

2.1 Gerenciamento de Tráfego Aéreo

O gerenciamento de tráfego aéreo (ATM) tem como objetivo garantir que as aeronaves cumpram seus planos de voo da melhor maneira possível, de acordo com os horários de partida e chegada previstos, e com observância à segurança operacional inerente à aviação civil [18]. A principal tarefa do ATM é estabelecer as condições necessárias para o uso do espaço aéreo com base na demanda atual e futura por tráfego aéreo [19].

As atividades relacionadas ao ATM preocupam-se tanto com o controle em solo quanto do espaço aéreo, e é realizada pelos controladores de voo. Esses profissionais devem evitar ao máximo o uso de informações empíricas uma vez que uma de suas responsabilidades é evitar acidentes aéreos [20]. Portanto, a atividade dos controladores de voo é revestida de grande relevância, o que tem motivado diversos trabalhos de pesquisa em todo mundo, por meio dos quais são investigadas melhorias no processo de ATM como um todo.

Dentre as funções que compõem o ATM destacam-se [21]:

- controle de tráfego aéreo (ATC): tem como objetivo manter a separação entre aeronaves e entre obstáculos aéreos e terrestres (colisões);
- gerenciamento do fluxo de tráfego aéreo (ATFM): regula o fluxo de aeronaves de forma eficiente para evitar congestionamentos do espaço aéreo e aeroportos sem comprometer a segurança;

- gerenciamento do espaço aéreo (ASM): estruturação do espaço aéreo para facilitar os serviços de ATC e como o uso do espaço aéreo é alocado aos diferentes usuários.

Estas funções são organizadas a nível nacional ou regional, e são coordenadas em todo o mundo pela Organização da Aviação Civil Internacional (ICAO) [21].

Alterações inesperadas ou indesejadas nos parâmetros de voo das trajetórias interferem significativamente no custo do voo, e representam alto impacto financeiro no consumo de combustível ou no custo relacionado ao tempo [22]. Dessa forma, o uso de modelos computacionais para auxiliar as atividades dos controladores de voo, na aplicação de medidas restritivas para uma distribuição justa do espaço aéreo, não só representa um incremento na segurança de voo como também evitam a inserção desnecessária de custos adicionais aos operadores.

2.2 Nagevação 4D

Uma trajetória pode ser definida como a rota de vôo quadridimensional (4D) de uma aeronave através do espaço (tridimensional) e do tempo (unidimensional) [23]. A trajetória 4D pode ser entendida como uma descrição precisa do caminho de uma aeronave no espaço e no tempo, incluindo incerteza de posição em todas as quatro dimensões [24]. As Operações Baseadas em Trajetória (TBO) visam integrar a capacidade de navegação de uma aeronave no espaço e no tempo para melhorar a eficiência e a previsibilidade no ATM [25].

Este objetivo requer principalmente: trajetórias de referência armazenadas; uma trajetória continuamente recalculada; indicadores eletrônicos de situação; e um sistema de controle para seguir a trajetória geral no espaço e no tempo [19]. A aeronave deve atender às restrições de tempo especificadas nos *waypoints* designados ao longo de sua rota. Assim, as trajetórias precisam, eventualmente, serem recalculadas nos pontos de referência selecionados de modo a atingir o tempo de chegada desejado [26].

Esse paradigma está sujeito a alguns esforços de pesquisa em todo o mundo. Os sistemas de gerenciamento de vôo 4D (FMS) são projetados para guiar aeronaves ao longo de uma trajetória de voo pré-especificada, de modo que a aeronave chegue ao destino no horário especificado pelo controlador de voo [26]. Tais sistemas implementam recursos de planejamento de voo que incluem definição de caminho horizontal e vertical para operação eficiente [27].

2.3 Fases do Voo

Com o aumento esperado da demanda pelo espaço aéreo, considerações ambientais e requisitos de eficiência nos anos seguintes, o ATM deve ser aprimorado para lidar com um cenário de maior complexidade. O ATM é composto por vários subsistemas locais, por exemplo, aeroportos (movimentação de terra e aproximação final), áreas de controle de terminal e Centros de Controle de Trânsito Aéreo que interagem juntos para gerenciar todos os vôos de portão a portão [28].

A autoridade de aviação tem uma visão global para coordenar todas as operações e equilibrar a demanda dos operadores de aeronaves e a capacidade do espaço aéreo, conhecidas como operações de Gerenciamento de Fluxo de Tráfego (TFM). Em termos gerais, as principais fases do planejamento e implementação do TFM são [28]:

- **estratégica** (de seis a onze meses antes). A autoridade identifica áreas, aeroportos e centros de rota, associadas a grandes atrasos. Com dados históricos, técnicas e simulações de modelagem do espaço aéreo, a autoridade responsável aprimora a estrutura e os procedimentos do espaço aéreo, a fim de se adaptar aos crescentes tráfegos e eventos especiais.
- **pré-tática** (cinco dias antes). As áreas são confirmadas de acordo com a previsão de tráfego e o manual de operações é estabelecido. O manual é um plano que contém os regulamentos a serem usados durante a fase tática, a fim de solucionar possíveis problemas de congestionamento.
- **tática** (no mesmo dia do voo). A autoridade supervisiona as operações de um ponto de vista global e compartilha as informações de ocupação do espaço com os centros de controle de tráfego aéreo.

2.4 Plano de Voo

O plano de vôo é o documento que indica todas as informações necessárias ao movimento pretendido pelas aeronaves, possibilitando que sejam disponibilizados e alocados recursos de tráfego aéreo para as aeronaves [29].

Um Plano de Vôo Repetitivo (RPL) é uma categoria de planos de voo que descreve vôos comerciais regulares. Esses vôos geralmente operam com muita frequência e com recursos básicos idênticos, usando repetidamente os mesmos recursos de ATM.

O RPL é usado para voos operando pelo menos uma vez por semana, com um mínimo de 10 voos, quando houver intenção de pelo menos dois meses de validade. Os documentos RPL são muito confiáveis, no sentido de que os dados fornecidos têm um grande grau

de estabilidade; portanto, eventuais alterações que venham a ser necessárias podem ser facilmente aplicadas. Naturalmente, os RPL são típicos para aeronaves comerciais que freqüentemente executam a mesma rota em horários bem definidos [19].

As propostas de planos de voo devem ser apresentadas ao provedor com uma antecedência mínima de 10 dias a partir do início previsto do período de validade [29]. No Brasil, esse provedor é a Central de Plano de Vôo Repetitivo (CPVR) no Centro de Gerenciamento de Navegação Aérea (CGNA) [23]. Após o processamento, o provedor publica essas informações para os Centros de Controle de Área (ACC) envolvidos no movimento pretendido.

2.5 Preditor de Trajetórias

A Previsão de Trajetória (TP) é um problema de estimativa onde o tempo em que uma estimativa é desejada ocorre após a última medição disponível [30]. Seu objetivo é calcular a posição da aeronave em 4D no futuro, sabendo seu estado inicial. Matematicamente, é descrito como um conjunto ordenado de tempo de vetores de estado de aeronave. A maioria dos preditores de trajetória atualmente implantados pressupõe que a aeronave mantenha uma altitude, velocidade e aceleração constantes. O TP está vinculado ao gerenciamento de separação e sua precisão afeta a investigação estratégica de conflito em rota e o alerta tático de conflito. O objetivo final de melhorar o TP é aumentar a capacidade do espaço aéreo, reduzindo a separação mínima entre aeronaves, reduzindo a ocorrência de alertas de conflito de trajetória falsa e evitando a perda de alertas de conflito de trajetória válidos [10]. A Figura 2.1 apresenta um *framework* genérico para o TP.

Os sistemas ou ferramentas de TP podem ser classificadas de diferentes maneiras:

- sistemas da aeronave *versus* sistemas em solo;
- abordagens determinísticas *versus* abordagens probabilísticas;
- modelos matemáticos adotados;
- conforme a fase do voo (em rota *versus* em solo).

Uma descrição da trajetória da aeronave pode incluir detalhes adicionais sobre o movimento da aeronave [21]:

- aspectos geométricos, que se referem à posição do centro de massa da aeronave em relação a um determinado sistema de coordenadas de referência e à atitude da aeronave, geralmente dada pelos ângulos de Euler;

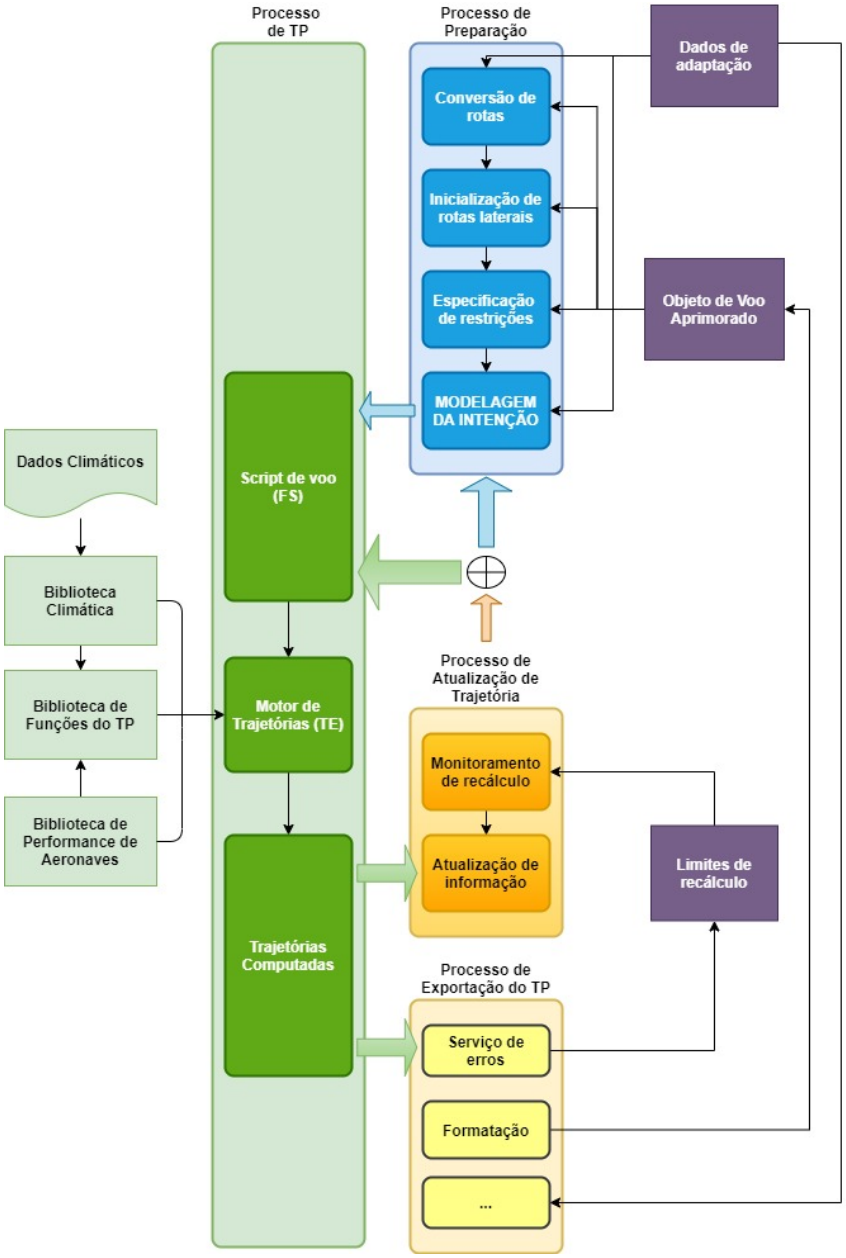

Figura 2.1: Framework genérico do TP, adaptado de (Fonte: [21]).

- aspectos cinemáticos, que se referem à evolução temporal da posição e velocidade do centro de gravidade e da atitude da aeronave e das velocidades angulares;
- aspectos cinéticos, que se referem às forças e momentos que atuam na aeronave, causas diretas de seu movimento, bem como aos controles de voo que influenciam essas forças e à evolução temporal da massa da aeronave.

Dentre as abordagens com modelos de cálculos não classificados destacam-se: técnicas de aprendizado de máquina sem modelar a performance da aeronave; regressão linear

aplicada a dados históricos de radar; e redes neurais treinadas com conjuntos de trajetórias reais [10].

2.6 *Automatic Dependent Surveillance - Broadcast* (ADSB)

O sistema de vigilância dependente automática (ADSB) é uma crescente tecnologia de vigilância que permite aos controladores de tráfego aéreo rastrear aeronaves e veículos terrestres em aeroportos, sem a necessidade de radar (aeronaves e demais veículos transmitem sua posição e outros parâmetros). O sistema ADSB consiste em aviônicos de aeronaves e uma infraestrutura terrestre, o que pode possibilitar a substituição do radar na vigilância de tráfego aéreo [10].

O conceito ADSB emergiu na década de 1990 devido à necessidade do uso mais flexível e eficiente do espaço aéreo, e com a ideia de redução de custos de vigilância. A tecnologia baseia-se em dois componentes [10]:

- dispositivo para calcular a posição da aeronave;
- veículo para transmitir a posição (e demais informações de momento) via link para demais atores.

O ADSB dispensa a necessidade de comunicação entre o controlador de voo e a aeronave (a aeronave transmite sua posição), e a posição é baseada no Sistema Global de Navegação por Satélite (GNSS). Demais informações transmitidas via ADSB são calculadas pelos equipamentos contidos na própria aeronave, em especial o FMS, [10]. A Figura 2.2 descreve o funcionamento do ADSB.

As aeronaves podem ser equipadas com receptores ADSB para receber os sinais emitidos por outras aeronaves, veículos terrestres ou retransmissões de estações terrestres de vigilância dependente automática. Diferentes soluções estão sendo desenvolvidas para utilizar os dados ADSB recebidos no reconhecimento da situação do tráfego, separação entre aeronaves, dentre outras. Essas soluções são chamadas de ADSB IN. A transmissão periódica de dados ADSB é chamada de ADSB OUT [10].

As informações ADSB são um componente importante desse trabalho. Uma vez que os dados transmitidos via ADSB refletem parâmetros reais das aeronaves, espera-se com o uso do ADSB a proposição de modelo que possibilite mapear fontes de incerteza às quais as aeronaves estão constantemente sujeitas em situações reais.

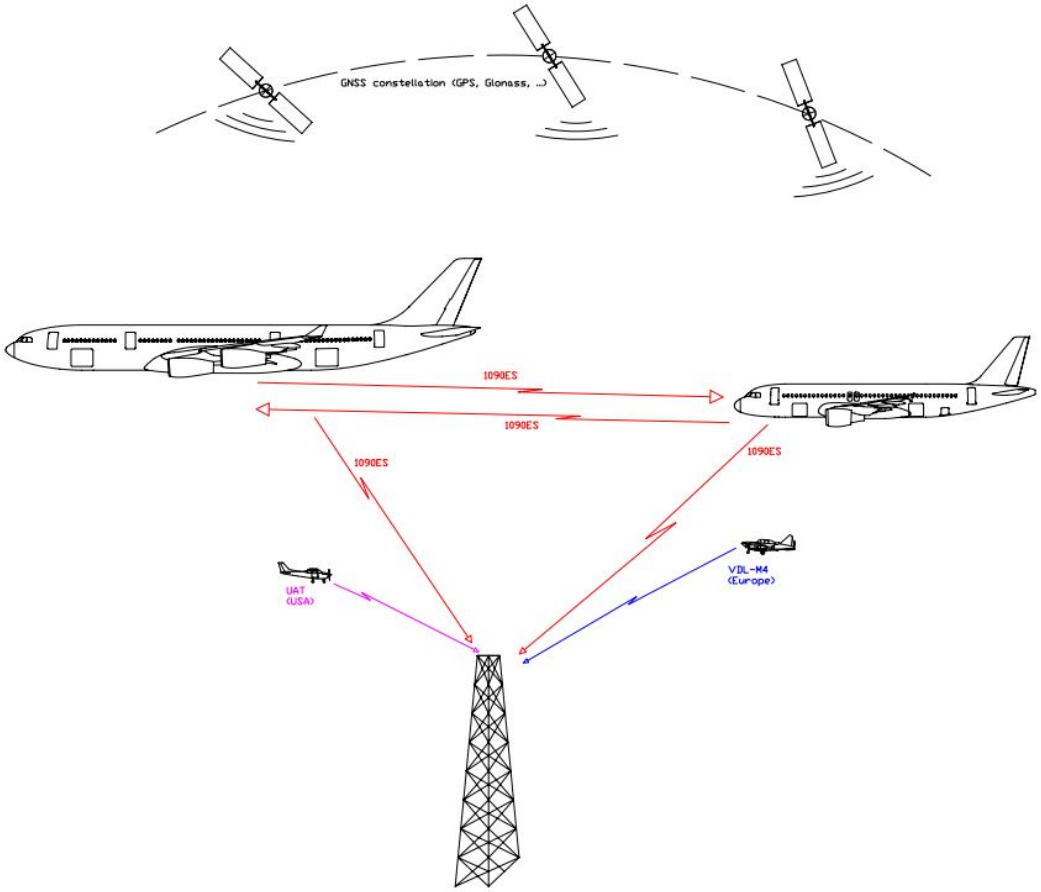

Figura 2.2: Princípios do ADSB (Fonte: [10]).

2.7 ATM no Brasil

No Brasil, o gerenciamento do tráfego aéreo é realizado pelo Sistema de Controle do Espaço Aéreo Brasileiro (SISCEAB), o "conjunto de órgãos e instalações - tais como auxílio à navegação aérea, radares de vigilância, centros de controle e torres de controle de aeródromo, estações de telecomunicações, recursos humanos, etc. - que tem como objetivo proporcionar regularidade, segurança e eficiência do fluxo de tráfego nos aeroportos e no espaço aéreo" [31]. Dentre as atividades do SISCEAB, destacam-se:

- controle de tráfego aéreo (ATC);
- telecomunicações aeronáuticas e auxílios à navegação aérea;
- meteorologia aeronáutica;
- cartografia e informações aeronáuticas;
- busca e salvamento;

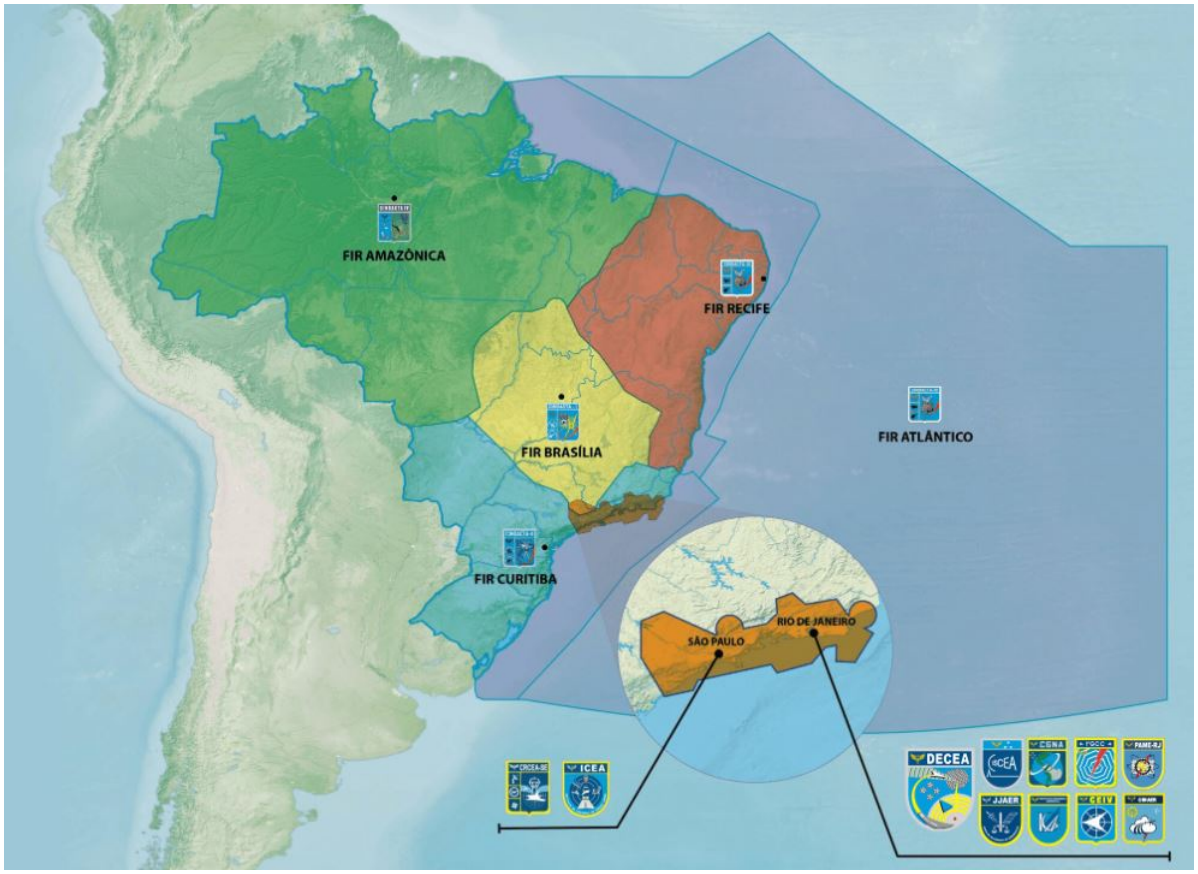

Figura 2.3: Subdivisão do espaço aéreo brasileiro.

- inspeção em voo;
- coordenação e fiscalização de ensino técnico específico; e
- supervisão de fabricação, reparo, manutenção e distribuição de equipamentos terrestres de auxílio à navegação aérea.

O órgão central do SISCEAB é o Departamento de Controle do Espaço Aéreo (DECEA), organização do Comando da Aeronáutica (COMAER), que tem a responsabilidade de planejar, gerenciar, controlar e proporcionar o apoio logístico e a segurança necessários à realização das atividades relacionadas à provisão dos Serviços de Navegação Aérea no território nacional e no espaço aéreo sob sua jurisdição. Atualmente, conforme ilustrado na Figura 2.3, o espaço aéreo brasileiro está subdividido em 5 Regiões de Informações de Voo (FIR), que cobrem mais de 22 milhões de km², sendo 8,5 milhões acima do continente, 3,5 milhões sobre a Zona Econômica Exclusiva (ZEE) e 10 milhões sobre parte do Oceano Atlântico, em consonância com acordos internacionais no âmbito da Organização Internacional da Aviação Civil (ICAO)[1].

[1]https://performance.decea.mil.br/

A dimensão do SISCEAB pode ser compreendida a partir dos seguintes números:

- mais de 1,5 milhão de movimentos (pousos, decolagens, sobrevoos e toques e arremetidas);
- 57 torres de controle de aeródromos;
- aproximadamente 4,7 mil aeródromos;
- mais de 5 mil voos diários; e
- pouco menos de 12 mil profissionais ligados ao DECEA.

Recentemente, o DECEA iniciou o Programa SIRIUS, instrumento do Comando da Aeronáutica voltado para a evolução do SISCEAB, em resposta às demandas provenientes do crescimento e do aumento da diversidade do tráfego aéreo previsto para as próximas décadas e das evoluções tecnológicas no campo da aviação.

O Projeto SIRIUS busca, por meio do emprego de soluções de alta tecnologia, da implantação de procedimentos operacionais inovadores e da ênfase na contínua elevação da performance dos recursos humanos, permitir à sociedade brasileira usufruir de um sistema de navegação aérea ágil, seguro, sustentável ambientalmente, de alto rendimento e adequado aos níveis de interoperabilidade mundial, bem como de uma infraestrutura de defesa da soberania do espaço aéreo adequada às necessidades do Brasil[2].

[2]https://sirius.decea.mil.br/

Capítulo 3

Metodologias da Ciência da Computação

Nesse Capítulo serão apresentados conceitos fundamentais da Ciência da Computação que integram as metodologias utilizadas nas propostas de solução. Inicialmente, o conceito de banco de dados NoSQL, com dois exemplos de implementação, será apresentado. Em seguida, serão abordados os algoritmos de busca que foram empregados na pesquisa, com as respectivas demonstrações de implementação. Por fim, a metodologia de Modelo de Espaço Vetorial (MEV), componente fundamental na identificação da melhor estratégia de seleção de trajetórias, será descrito.

3.1 Bancos de Dados Not only SQL

Bancos de dados NoSQL (Not Only SQL) constituem uma abordagem desenvolvida para gerenciamento de *big data* a ser distribuído entre vários nós em uma rede [32]. Eles se referem a um grupo crescente de sistemas de gerenciamento de dados não relacionais em que os bancos de dados não são construídos principalmente em tabelas e geralmente não usam a linguagem SQL para manipulação de dados. Os sistemas de gerenciamento de banco de dados NoSQL são úteis quando se trabalha com uma grande quantidade de dados quando a natureza dos dados não requer um modelo relacional [33]. Uma das limitações dos modelos relacionais é que eles têm problemas de escalabilidade, então o desempenho diminui rapidamente à medida que os volumes de dados aumentam [34].

Para garantir a integridade dos dados, a maioria dos sistemas clássicos de banco de dados são baseados em transações. Isso garante a consistência dos dados em todas as situações de gerenciamento de dados. Essas características transacionais também são conhecidas como ACID (Atomicidade, Consistência, Isolamento e Durabilidade). No entanto, a expansão de sistemas compatíveis com ACID mostrou ser um problema. Conflitos

estão surgindo entre os diferentes aspectos de alta disponibilidade em sistemas distribuídos que não são totalmente solucionáveis - conhecido como o teorema CAP [33]:

- consistência: todos veem a mesma versão dos dados, mesmo em atualizações no conjunto de dados;
- disponibilidade: todos sempre podem encontrar pelo menos uma cópia dos dados solicitados, mesmo que algumas das máquinas de um cluster estejam inativas; e
- tolerância à partição: o sistema total mantém sua característica mesmo quando implantado em servidores diferentes, transparente para o cliente.

O Teorema CAP postula que apenas dois dos três aspectos diferentes da expansão podem ser alcançados totalmente ao mesmo tempo, conforme ilustrado na Figura 3.1.

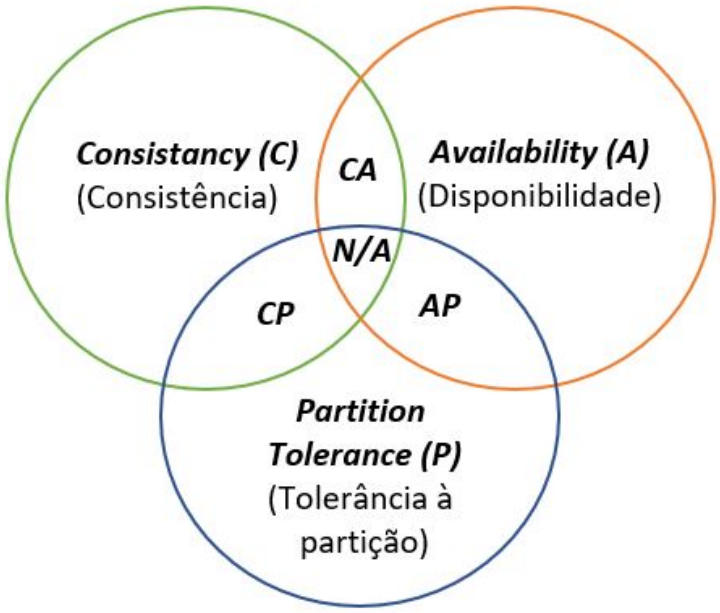

Figura 3.1: Características dos bancos de dados NoSQL (Fonte: [33]).

Bancos de dados NoSQL fornecem armazenamento relativamente barato e altamente escalável para dados históricos de alto volume e pequenos pacotes, como logs, registros de dados de chamadas, leituras de medidores e instantâneos de ticker (ou seja, armazenamento em "*big bit bucket*") e para armazenamento de dados semiestruturado ou não estruturado (arquivos de e-mail, arquivos xml, documentos, etc.). Sua estrutura distribuída também os torna ideais para processamento massivo de dados em lote (agregação, filtragem, classificação, processamento algorítmico, etc.), e são adequados para recuperação de

informações e comunicação máquina a máquina, além de processamento de transações de alto volume, desde que as restrições ACID possam ser relaxadas ou, pelo menos, impostas no nível do aplicativo [33].

Existem diferentes implementações ou arquiteturas de bancos de dados NoSQL, geralmente categorizadas da seguinte forma [34]:

- **bancos de dados orientados a documentos**: Consiste de arquiteturas que armazenam seus dados na forma de documentos. Os repositórios de documentos oferecem excelente desempenho e opções de escalabilidade horizontal. Os documentos dentro de um banco de dados orientado a documentos são um pouco semelhantes aos registros em bancos de dados relacionais, mas são muito mais flexíveis, pois não têm esquemas. Em bancos de dados relacionais, todos os registros de uma mesma tabela apresentarão os mesmos campos, e os campos de dados não utilizados são mantidos vazios. No caso dos bancos de dados NoSQL orientados a documentos, cada documento pode ter uma estrutura própria, com campos semelhantes e diferentes dos demais documentos/registros, representado por formatos como XML, PDF, JSON, etc.

  Fazem parte deste grupo as seguintes tecnologias: **MongoDB**, Apache CouchDB, IBM Cloudant, CrateDB, Azure Cosmos DB,etc.

- **bancos de dados chave-valor**: são implementações NoSQL relativamente simples, porém bastante eficientes. Um armazenamento de dados chave-valor permite que o usuário armazene dados de uma maneira sem esquema, consistindo de duas partes: uma string que representa a chave e os dados reais que devem ser referidos como valor, criando assim um par "chave-valo". Esses armazenamentos são semelhantes a tabelas *hash* onde as chaves são usadas como índices, tornando o NoSQL mais rápido que os bancos de dados relacionais.

  O modelo de dados é simples: um mapa ou um dicionário que permite ao usuário solicitar os valores de acordo com a chave especificada. Os armazenamentos de dados de chave-valor modernos preferem alta escalabilidade em vez de consistência. Portanto, os recursos de consulta e análise *ad hoc*, como junções e operações agregadas, são preteridos. Alta simultaneidade, pesquisas rápidas e opções para armazenamento em massa são fornecidas pelos armazenamentos chave-valor.

  Dentre as principais implementações, destacam-se: Amazon DynamoDB, Oracle NoSQL Database, InfinityDB, Redis, etc.

- **bancos de dados orientados a coluna**: Os armazenamentos orientados a colunas em NoSQL são, na verdade, armazenamentos híbridos de linha/coluna, ao contrário

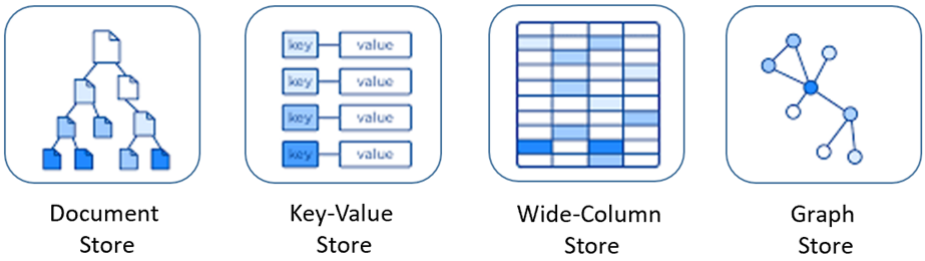

Figura 3.2: Modelos de banco de dados NoSQL, disponível em https://learn.microsoft.com/pt-br/dotnet/architecture/cloud-native/relational-vs-nosql-data.

dos bancos de dados de colunas relacionais puros. Nesta abordagem, cada chave é associada a um ou mais atributos (colunas), de maneira que os dados possam ser agregados rapidamente com menos atividade de E/S, oferecendo alta escalabilidade no armazenamento de dados. Os dados armazenados no banco de dados são baseados na ordem de classificação da família de colunas.

A lista de representantes dos bancos de dados NoSQL orientados a colunas conta com: **Apache Cassandra**, DataStax, Microsoft Azure Cosmos DB, ScyllaDB, etc.

- **bancos de dados orientados a grafos**: Os bancos de dados NoSQL orientados a grafos armazenam seus dados nesta forma de estrutura de dados, que consiste em nós e arestas, onde os nós atuam como os objetos e as arestas atuam como o relacionamento entre os objetos. Nesta representação, cada nó consiste em um ponteiro direto que aponta para o nó adjacente, permitindo que milhões de registros possam ser percorridos. Em um banco de dados orientado a grafos, a ênfase principal está na conexão entre os dados, fornecendo menos esquema e armazenamento eficiente de dados semiestruturados. As consultas são expressas como "travessias", tornando os bancos de dados orientados a grafos geralmente mais rápidos do que os bancos de dados relacionais.

  Nesta categoria, destacam-se: Neo4j, Amazon Neptune, OrientDB, etc.

A Figura 3.2 ilustra as arquiteturas NoSQL descritas acima. Nas próximas seções, serão detalhadas as implementações NoSQL que foram adotadas neste trabalho.

3.1.1 Apache Cassandra

Apache Cassandra (ou simplesmente Cassandra) é um banco de dados NoSQL projetado para ser um sistema de gerenciamento *peer-to-peer* através da comunicação entre vários nós distribuídos em uma rede. Cada nó tem a mesma hierarquia e carga de trabalho e

cada um pode gravar e ler dados. Os dados do aplicativo são particionados por todos os nós no cluster e a replicação de dados é bastante aceita neste sistema de gerenciamento de dados distribuído [35].

Ao contrário dos modelos relacionais, um modelo de dados em Cassandra não define o conceito de Entidade-Relacionamento (ER). Em vez disso, os dados são estruturados em colunas permitindo a classificação de cada detalhe de um item em uma única linha, indexada por sua chave. Essa implementação favorece a realização de operações de leitura e escrita quando a aplicação é destinada a *big data*. Em um banco de dados Cassandra, os dados são particionados em famílias de colunas, de maneira semelhante aos bancos de dados ER. Uma família de colunas é simplesmente uma coleção de linhas rotuladas por um nome. As famílias de colunas são agrupadas logicamente em uma estrutura chamada Keyspace. Keyspaces funcionam como um escopo isolado para os nomes das famílias de colunas.

Um banco de dados Cassandra permite alta escalabilidade, projetado especificamente para gerenciamento de *big data*. A distribuição de dados entre os nós da rede é transparente para o administrador, desde que nenhum esforço de programação seja exigido nesta tarefa [35]. Os nós em um banco de dados Cassandra são distribuídos em uma topologia em anel. A arquitetura não apresenta um único ponto de falha e a disponibilidade de dados é garantida mesmo que um nó fique fora de serviço. Além disso, a capacidade de processamento cresce linearmente em relação ao número de nós, que podem ser adicionados ou removidos online. A replicação de dados é configurada de acordo com as necessidades do aplicativo.

Como qualquer banco de dados NoSQL, o Cassandra não suporta transações ACID. O comportamento padrão não garante consistência de dados em todas as réplicas, ou seja, se pelo menos um nó possuir um determinado dado, a transação é considerada concluída [32]. Isso acontece para diminuir o tempo de leitura/gravação, pois são extremamente eficientes nessa arquitetura.

Comandos comuns em modelos de dados relacionais, como junções e subconsultas, não são permitidos no Cassandra. Por esse motivo, a linguagem SQL não é usada nas consultas do Cassandra. Em vez disso, CQL (Cassandra Query Language) é definido como a linguagem de consulta. Além disso, a estrutura limita a referência a cláusulas *where* às colunas que fazem parte da chave primária da família de colunas correspondente [35].

3.1.2 MongoDB

MongoDB é outro banco de dados NoSQL, mas que implementa armazenamento de documentos sem esquema, onde os documentos são agrupados em coleções. Os documentos

são armazenados no formato BSON (Binary JSON) e para aumentar o desempenho, os arquivos de dados são mapeados na memória. Os dados são enviados para o disco periodicamente e quando novos arquivos são criados, tudo é descarregado no disco, liberando memória. O MongoDB usa índices em coleções de maneira semelhante aos bancos de dados relacionais e oferece suporte a redução de mapa para agregações complexas entre documentos para aumentar o desempenho [33]. Cada documento é identificado por um campo *_id* e um índice único é criado sobre este campo.

Além da criação automática de índices no campo *_id*, índices adicionais podem ser criados pelo administrador do banco de dados. Por exemplo, um índice pode ser definido em um campo dentro de uma coleção específica para funcionar com uma chave específica. Este recurso do MongoDB é chamado de *índice composto*. No entanto, todos os índices usam a mesma estrutura de árvore B, e cada consulta usa apenas um índice escolhido pelo mecanismo otimizador de consulta, dando preferência ao índice mais eficiente [36].

Nesta pesquisa, a escolha pelo MongoDB foi baseada em seu desempenho quando comparado a outros bancos de dados relacionais e NoSQL. Muitos estudos mostram que o MongoDB tem um desempenho melhor quando comparado a bancos de dados SQL e outros tipos de bancos de dados NoSQL [33]. Dependendo do tamanho das coleções e da modelagem adotada, o MongoDB pode alcançar o melhor desempenho em operações de leitura quando comparado a outros bancos de dados otimizados para *big data* [1]. O MongoDB usa um esquema de expansão que é flexível contra a expansão de hardware e oferece suporte à fragmentação automática. Assim, a distribuição automática de dados por vários servidores pode ser convenientemente realizada. A configuração do MongoDB de particionamento horizontal de dados (*sharding*) e conjuntos de réplicas garante alta disponibilidade, segurança e consistência de dados. Eles também permitem a expansão distribuída para processamento de dados envolvendo grandes quantidades de dados [37].

As funções dos componentes do MongoDB são as seguintes [37]:

- *mongod*: lida com requisições de dados, gerencia o acesso aos dados e executa operações de gerenciamento em segundo plano;
- *mongos*: processa as consultas da camada de aplicação e determina a localização destes dados no cluster compartilhado;
- *shard*: armazena dados. Para fácil disponibilidade e consistência de dados, cada fragmento é um conjunto de réplicas;
- *servidor de configuração*: armazena os metadados do cluster. Esses dados contêm um mapeamento do conjunto de dados do cluster para os estilhaços;

- *conjuntos de réplicas*: um grupo de processos mongod que mantêm o mesmo conjunto de dados. Se o mongod primário estiver indisponível, o conjunto de réplicas elegerá um novo mongod primário.

A distribuição uniforme de dados entre estilhaços é controlada por uma chave de estilhaço. Uma chave de fragmentação é um único campo indexado ou um campo composto indexado que existe em todos os documentos. Os dados no MongoDB são divididos em blocos, unidades lógicas de dados armazenados, de acordo com a chave de estilhaço usando particionamento baseado em intervalo ou particionamento baseado em hash. Portanto, a seleção apropriada da chave de fragmentação é um fator de decisão muito importante no design do MongoDB [37].

3.2 Algoritmos de Busca em Árvores

O campo dos algoritmos de busca em árvore reúne uma grande quantidade de abordagens, com as mais variadas características. Nesta seção optou-se por focar nas metodologias que se mostraram mais adequadas à modelagem proposta.

3.2.1 *Breadth First Search* (BFS)

A busca em largura, do inglês *breadth first search* (BFS) expande os nós em ordem de profundidade a partir da raiz, gerando um nível da árvore por vez até que uma solução seja encontrada, como pode ser verificado na Figura 3.3. É mais facilmente implementado mantendo uma fila de nós primeiro a entrar, primeiro a sair, inicialmente contendo apenas a raiz, e sempre removendo o nó no início da fila, expandindo-o e adicionando seus filhos ao final da fila [38].

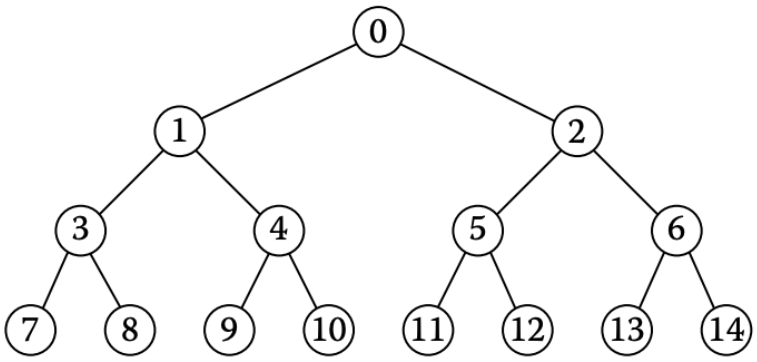

Figura 3.3: Ordem da geração dos nós na BFS. (Fonte: [38]).

Uma vez que nunca gera um nó na árvore até que todos os nós em níveis mais rasos tenham sido gerados, BFS sempre encontra um caminho mais curto para um objetivo. Como cada nó pode ser gerado em tempo constante, a quantidade de tempo usada pela busca em largura é proporcional ao número de nós gerados, porém tem como principal desvantagem o requisito de memória. Como cada nível da árvore deve ser armazenado para gerar o próximo nível, e a quantidade de memória é proporcional ao número de nós armazenados, BFS é severamente limitada pelo espaço na prática e esgotará a memória disponível em computadores típicos em questão de minutos [38].

Em termos matemáticos, o desempenho do BFS, que pode estar relacionado ao tempo necessário para consumir um grafo com V vértices e A arcos, baseia-se no fato de que cada arco do grafo é percorrido no máximo uma vez durante a execução da iteração. Assim, a execução de todas as iterações consome, no pior caso, tempo proporcional a A:

$$O_{BFS} = V + A \tag{3.1}$$

Se o grafo for representado por matriz de adjacências, uma análise semelhante indica que a complexidade apontada na Equação 3.1 representa V^2 no pior caso.

3.2.2 *Depth First Search* (DFS)

A busca em profundidade, do inglês *depth first search* (DFS) remove a limitação de espaço da busca em largura por sempre gerar um filho a partir nó não expandido mais profundo, como pode ser observado na Figura 3.4. Ambos os algoritmos podem ser implementados usando uma lista de nós não expandidos, com a diferença que DFS gerencia a lista como uma fila de primeiro a entrar, primeiro a sair, enquanto a pesquisa em profundidade trata a lista como uma pilha de último a entrar, primeiro a sair [38].

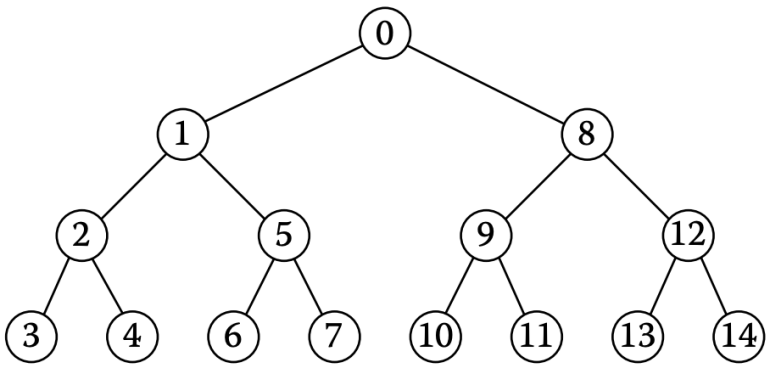

Figura 3.4: Ordem da geração dos nós na DFS. (Fonte: [38]).

A vantagem da DFS é que seu requisito de espaço é apenas linear na máximo profundidade de pesquisa, em oposição a exponencial para BFS. A principal desvantagem da DFS é que ela pode não terminar em uma árvore infinita, mas simplesmente desça o caminho mais à esquerda para sempre. A solução habitual para este problema é impor uma profundidade de corte na busca [38].

De maneira semelhante ao BFS, verifica-se que o desempenho do DFS considerando um grafo com V vértices e A arcos corresponde à Equação 3.2:

$$O_{DFS} = V + A \tag{3.2}$$

Na seção de resultados é possível verificar que tal abordagem apresenta excelentes resultados em cenários menores, ou seja, menos aeronaves e/ou menos trajetórias por aeronave. É provável que o DFS atenda a maioria das necessidades atuais de tráfego aéreo, mas considerando a perspectiva de um aumento contínuo na demanda de tráfego aéreo [39], foi necessário desenvolver um segundo modelo que permitisse a identificação de trajetórias ótimas em complexos cenários.

3.2.3 *Alfa-Beta*

Na abordagem inspirada no clássico algoritmo Alfa-Beta, a montagem e a busca em árvore tentam evitar que todos os nós possíveis sejam investigados. Esta característica é obtida com o cálculo, em simultâneo com a montagem da árvore, da pontuação acumulada até ao nó de estudo, eliminando a necessidade de adicionar novos nós caso já seja identificado um custo inferior ao mínimo alcançado até ao momento.

A eliminação de nós/caminhos conhecidos piores permite que o algoritmo reduza o espaço de busca, aumentando a capacidade de analisar árvores maiores considerando os mesmos recursos computacionais usados no DFS. No Capítulo 7 será possível visualizar essa redução no espaço de busca e, consequentemente, a análise de cenários bem mais complexos, provavelmente suficientes para a situação atual e futura do tráfego aéreo global.

Se considerarmos o pior caso do Alfa-Beta, na qual temos todos os nós da árvore investigados, sem qualquer redução no espaço de busca, temos que a complexidade pode ser representada por:

$$O_{Alfa-Beta} = b^m \tag{3.3}$$

onde b indica a quantidade de movimentos possíveis em cada estado e m a profundidade máxima da árvore. Porém, observa-se que, utilizando a melhor ordem, a complexidade do Alfa-Beta passa a ser $b^{m/2}$, e de $b^{3m/4}$ quando se consideram ordem aleatória

3.2.4 *Monte Carlo Tree Search* (MCTS)

O algoritmo básico MCTS envolve a construção iterativa de uma árvore de pesquisa até que algum custo computacional predefinido - normalmente um tempo, memória ou restrição de iteração - seja alcançado, ponto no qual a pesquisa é interrompida e a ação raiz de melhor desempenho é retornada. Cada nó na árvore de pesquisa representa um estado do domínio e os links direcionados para nós filhos representam ações que levam a estados subsequentes [3].

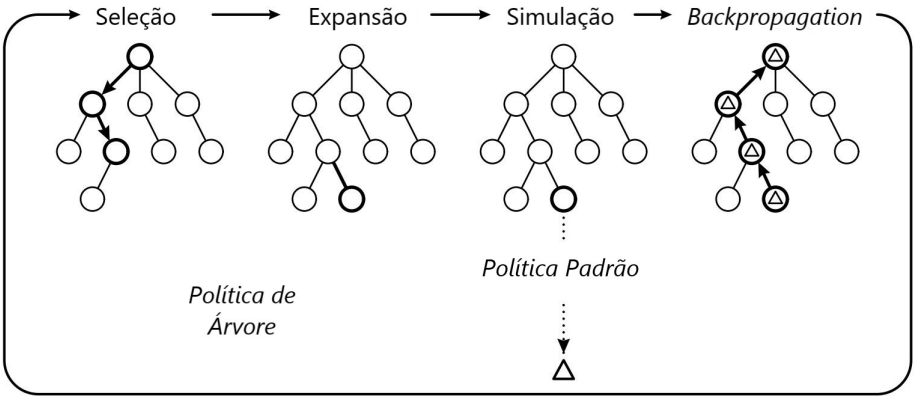

Figura 3.5: Representação da iteração do MCTS [1].

Quatro etapas são aplicadas por iteração de pesquisa [40], conforme ilustrado na Figura 3.5:

1. *seleção*: Iniciando no nó raiz, uma política de seleção de filhos é aplicada recursivamente para descer pela árvore até que o nó expansível mais urgente seja alcançado. Um nó é expansível se representar um estado não terminal e tiver filhos não visitados (ou seja, não expandidos).
2. *expansão*: Um (ou mais) nós filhos são adicionados para expandir a árvore, de acordo com as ações disponíveis.
3. *simulação*: uma simulação é executada a partir do (s) novo (s) nó (s) de acordo com a política padrão para produzir um resultado.
4. retropropagação: O resultado da simulação é "compartilhado" (ou seja, retropropagado) através dos nós selecionados para atualizar suas estatísticas.

Eles podem ser agrupados em duas políticas distintas [1]:

1. *política da árvore*: selecione ou crie um nó folha a partir dos nós já contidos na árvore de pesquisa (seleção e expansão).

2. *política padrão*: execute o domínio a partir de um determinado estado não terminal para produzir uma estimativa de valor (simulação).

Na etapa de seleção, é importante garantir que cada nó tenha uma chance "justa" de escolha. Uma das abordagens mais populares da família MCTS, talvez a mais utilizada, é a *Upper Confidence Bounds for Trees* (UCT) [1], cujo cálculo para seleção ideal de nós é fornecido pela Equação 3.4.

$$F = \frac{v_i}{i_i} + e.\sqrt{\frac{\ln t}{i_i}} \tag{3.4}$$

Na Equação 3.4, v_i representa o número de resultados positivos após o movimento $i-th$; i_i indica o número de simulações após o movimento $i-th$; e representa o parâmetro de exploração (teoricamente igual a $\sqrt{2}$ [3]) e t é igual ao número total de simulações para o nó pai. A fórmula garante que nenhum estado será esquecido ao passo que visitará com maior frequência ramos mais promissores.

Todos os jogos de informação perfeita têm uma função de valor ótimo, $v*(s)$, que determina o resultado do jogo a partir de cada posição do tabuleiro ou estado s, sob jogo perfeito por todos os jogadores. Esses jogos podem ser resolvidos computando recursivamente a função de valor ideal em uma árvore de pesquisa contendo aproximadamente b^d possíveis sequências de movimentos, onde b é a amplitude do jogo (número de movimentos legais por posição) e d é sua profundidade (duração do jogo). Em jogos grandes, como xadrez ($b \approx 35$; $d \approx 80$) e especialmente Go ($b \approx 250$; $d \approx 150$), a busca exaustiva é inviável, mas o espaço de busca efetivo pode ser reduzido por dois princípios gerais. Primeiro, a profundidade da pesquisa pode ser reduzida pela avaliação da posição: truncando a árvore de pesquisa no estado s e substituindo a subárvore abaixo de s por uma função de valor aproximado $v(s) \approx v*(s)$ que prevê o resultado dos estados s. Em segundo lugar, a amplitude da pesquisa pode ser reduzida amostrando ações de uma política $p(a|s)$ que é uma distribuição de probabilidade sobre possíveis movimentos a na posição s. O MCTS estima o valor de cada estado em uma árvore de pesquisa. À medida que mais simulações são executadas, a árvore de pesquisa fica maior e os valores relevantes se tornam mais precisos [41].

3.3 Modelos de Espaço Vetorial

Uma das maiores dificuldades para se fazer pleno uso do poder dos computadores é a completa compreensão do significado da linguagem humana. Esse desafio impulsiona o impacto transformador que tecnologias semânticas mais profundas têm tido recentemente.

Os Modelos de Espaço Vetorial (MEV) integram esse grupo de tecnologias semânticas transformadoras [42].

A semântica aqui é tratada de um modo geral, como o significado de uma palavra, uma frase, uma sentença, ou qualquer texto em linguagem humana. Os sentidos mais estritos de semântica, como a web semântica ou abordagens semânticas baseadas na lógica formal não serão considerados. O foco está nos MEV e na sua relação com a hipótese de distribuição como uma abordagem para representar alguns aspectos da semântica da linguagem natural.

Criada em 1971 para o sistema de recuperação de informação SMART [43], a representação baseada em MEV tem como intenção representar cada documento em uma coleção como um ponto em um espaço (um vetor em um espaço vetorial). Pontos que estão próximos neste espaço são semanticamente similares e pontos que estão distantes são semanticamente distantes. A pesquisa que se deseja fazer, na forma de uma consulta, é também representada como um ponto no mesmo espaço em que os documentos são inseridos (a consulta é considerada uma espécie de pseudo-documento). Os documentos são, então, classificados em ordem crescente de distância (ordem de diminuição da semelhança semântica) a partir da consulta.

Esta é a principal característica que impulsionou o estudo do MEV como componente da modelagem para CDR, ou seja, a capacidade de identificar semelhanças entre um banco de objetos (no caso do CDR, estratégias de gerenciamento do tráfego) e uma consulta (a estratégia desejada ou esperada).

O primeiro passo na abordagem MEV é a geração de uma matriz de frequências. Em seguida pode ser necessário ajustar os pesos dos elementos da matriz, porque as ocorrências comuns terão alta freqüência, no entanto, podem ser menos representativas. Em terceiro lugar pode ser necessário suavizar a matriz para reduzir a quantidade de extremos e preencher alguns elementos contendo zero numa matriz esparsa. Por fim, há muitas maneiras diferentes de se medir a semelhança entre dois vetores.

Portanto, a construção de um MEV pode ser descrita como sendo um processo de quatro etapas: calcular as frequências, aplicar pesos, suavizar (redução de dimensionalidade) e calcular as semelhanças [44].

Um elemento em uma matriz de freqüências corresponde a um evento: um determinado item que ocorre em uma determinada situação um certo número de vezes. Em um nível abstrato, a construção de uma matriz de frequências é uma simples questão de contagem de eventos. Na prática, isto pode ser complicado quando o conjunto de situações possíveis é grande.

Uma abordagem típica para a construção de uma matriz de frequências envolve duas etapas. Em primeiro lugar verifica-se sequencialmente o conjunto de documentos, arma-

zenando os eventos e suas frequências em uma tabela, um banco de dados ou um índice motor de busca. Em segundo lugar, usa-se a estrutura de dados resultante para gerar a matriz de frequências, com uma representação de matriz esparsa [45].

A ideia da ponderação é atribuir maior peso aos eventos surpreendentes e menos peso aos eventos esperados. A hipótese é de que os eventos surpreendentes, se compartilhados por dois vetores, são mais discriminativos da semelhança entre os vetores do que eventos menos surpreendentes. Por exemplo, na medição da semelhança semântica entre as palavras *rato* e *camundongo*, os contextos *dissecar* e *exterminar* são mais discriminativos de sua semelhança do que os contextos *têm* e *gostam*. Em teoria da informação, um evento surpreendente tem mais conteúdo que um evento esperado [46].

A forma mais popular de formalizar esta ideia para matrizes *termo-documento* é a família TF-IDF (frequência do termo *versus* inverso da frequência no documento) de funções de ponderação [47]. Um elemento recebe um peso elevado quando o termo correspondente é frequente no documento correspondente (ou seja, TF é alta), mas o termo é raro em outros documentos do conjunto (ou seja, DF é baixa e, assim, IDF é alta). As funções de ponderação da família TF-IDF podem produzir melhoras significativas em tarefas de recuperação de informação quando comparadas com a frequência bruta [48].

A $tf_{t,d}$ representa a frequência do termo t no documento d. A df_t de um termo é uma medida inversa da informatividade do termo t. Representa o número de documentos em que um termo aparece, isto é, dft não pode ser maior do que o número de documentos (N).

O inverso da frequência de documento pode ser calculado da seguinte forma:

$$idf_t = \log_{10} \frac{N}{df_t} \tag{3.5}$$

A função *log* é utilizada em vez de N/df_t para "amortecer" o efeito da idf_t. Por exemplo, se temos 1 milhão de documentos e um termo que aparece apenas em um documento, então idf_t é igual a 6, ou seja, temos um termo relevante. Por outro lado, se tivermos um termo que aparece em todos os documentos, a idf_t será 0, indicando tratar-se de um termo comum que provavelmente não é relevante.

Podemos aplicar este mesmo conceito para obter o termo ponderado de tf, neste caso, o coeficiente da frequência logarítmica ($w_{t,d}$) de um termo t em um documento d, que é definido por:

$$w_{t,d} = 1 + \log_{10} tf_{t,d} \tag{3.6}$$

Finalmente, a TF-IDF ponderada de um termo é o produto das componentes tf ponderada e idf ponderada:

$$w_{t,d} = (1 + \log_{10} tf_{t,d}) \times \log_{10} \frac{N}{df_t} \tag{3.7}$$

Outro tipo de ponderação, muitas vezes combinada com a ponderação TF-IDF, é a normalização de comprimento. Em recuperação de informação, se o comprimento do documento for ignorado, os motores de busca tendem a ter um viés em favor de documentos mais longos. A normalização do comprimento corrige esse viés [49].

A ponderação de termos também pode ser utilizada para corrigir termos correlacionados. Por exemplo, os termos *refém* e *reféns* tendem a ser correlacionados, ainda assim, pode não ser recomendável normalizá-los para o mesmo termo porque eles possuem significados ligeiramente diferentes. Como uma alternativa para a normalização, pode-se reduzir seus pesos quando quando ambos ocorrerem em um documento [50].

A maneira mais simples de melhorar o desempenho de um processo de recuperação de informação é limitar o número de componentes do vetor. Manter apenas as partes que representem palavras de conteúdo que ocorrem mais frequentemente é uma maneira; no entanto, as palavras comuns, tais como *o* e *têm*, possuem pouco poder de discriminação semântica. Componentes simples de suavização heurística, com base nas propriedades de esquemas de ponderação, têm demonstrado tanto a manutenção do poder de discriminação semântica quanto a melhora do desempenho de cálculos de similaridade.

Calcular a semelhança entre todos os pares de vetores é uma tarefa computacionalmente intensiva. No entanto, apenas os vetores que compartilham uma coordenada não-zero devem ser comparados (isto é, dois vetores que não compartilham nenhuma coordenada são diferentes).

A maneira mais popular de se medir a similaridade entre dois vetores é através do cálculo do cosseno entre eles. Sejam $\mathbf{x}$ e $\mathbf{y}$ dois vetores, cada um com n elementos:

$$\begin{aligned} x &= \langle x_1, x_2, ..., x_n \rangle \\ y &= \langle y_1, y_2, ..., y_n \rangle \end{aligned} \tag{3.8}$$

O cosseno do ângulo entre $\mathbf{x}$ e $\mathbf{y}$ pode ser calculado da seguinte forma:

$$\cos(x, y) = \frac{x}{||x||} . \frac{y}{||y||} \tag{3.9}$$

Em outras palavras, o cosseno do ângulo entre dois vetores é o produto interno desses vetores depois da normalização (vetores unitários). Se $\mathbf{x}$ e $\mathbf{y}$ são vetores de freqüências de palavras, palavras comuns terão longos vetores enquanto que palavras raras terão vetores curtos, mesmo se forem sinônimos. O cosseno capta a ideia de que o comprimento dos vetores é irrelevante; o importante é o ângulo entre eles.

O cosseno varia entre -1 quando os vetores apontam em sentidos opostos (o ângulo é de 180 graus) e 1 quando apontam na mesma direção (o ângulo é 0). Quando os vetores são ortogonais (o ângulo é de 90 graus), o cosseno é zero. Normalmente quando não existem elementos negativos nos vetores, o cosseno também não é negativo, mas a ponderação e a suavização podem introduzir elementos negativos.

Uma medida da distância entre os vetores pode ser facilmente convertida em uma medida de similaridade por inversão ou subtração:

$$sim(x, y) = \frac{1}{dist(x, y)} \tag{3.10}$$

$$sim(x, y) = 1 - dist(x, y) \tag{3.11}$$

Muitas medidas de similaridade foram propostas em recuperação de informação [51]. Costuma-se dizer que, com vetores devidamente normalizados, a diferença de desempenho da recuperação usando diferentes medidas é insignificante [52].

Capítulo 4

Revisão Bibliográfica

Neste Capítulo serão apresentados diferentes estudos desenvolvidos ao longo dos últimos anos que tem como objetivo investigar problemas relacionados ao transporte aéreo. Como forma de melhor organizar as informações, os trabalhos foram agrupados em 3 seções: pesquisas baseadas em técnicas de pesquisa operacional (OR); estudos que utilizam metodologias da inteligência artificial (AI); e, por fim, pesquisas que consolidam e analisam comparativamente outros conjuntos de trabalhos.

4.1 Pesquisa Operacional Aplicada ao ATM

Na literatura sobre detecção e resolução de conflitos (CDR), técnicas de pesquisa operacional (OR) são amplamente utilizadas, e com resultados bastante satisfatórios. Dentre os estudos publicados na última década, Dias *et. al.* [53] abordam o problema de resolução de conflitos de aeronaves utilizando formulações de programação inteira mista, com controle de velocidade e direção em uma primeira abordagem 2D e, em seguida, com controle adicional de altitude, em uma segunda abordagem 3D. Na abordagem 2D é proposto um algoritmo de pré-processamento que identifica pares de aeronaves livres de conflitos, o que reduz o tamanho do espaço de busca, e na abordagem 3D é proposta uma otimização que visa diminuir o número de mudanças de nível de voo (alteração de altitude) para solucionar conflitos remanescentes utilizando a abordagem 2D.

A proposta de solução foi comparada com outras duas abordagens baseadas em programação linear, e se mostrou melhor tanto em termos da quantidade de conflitos resolvidos quanto do tempo necessário para execução dos algoritmos. Porém, a abordagem proposta depende de várias suposições que podem ser limitantes na prática, tais como: leis de movimento uniforme, que se traduzem em taxas infinitas de aceleração e desaceleração; garantia de recuperação de trajetórias livres de conflitos; e incertezas na previsão

de trajetórias, impactadas, por exemplo, por condições climáticas adversas ou falhas de comunicação dos sistemas de navegação.

Wang *et. al.* [54] apresentam uma estrutura genérica para o problema de resolução de conflitos que separa a trajetória e os modelos de conflito da resolução, e é capaz de lidar com qualquer tipo de manobra e modelos de detecção. Na resolução do problema, é introduzida uma estrutura genérica para a cooperação de algoritmos de otimização. A cooperação entre um algoritmo memético (MA) e um programa linear inteiro (ILP) possibilita tratar instâncias com até 60 aeronaves de forma otimizada em poucos minutos, diferentemente do que ocorre quando cada algoritmo é considerado individualmente.

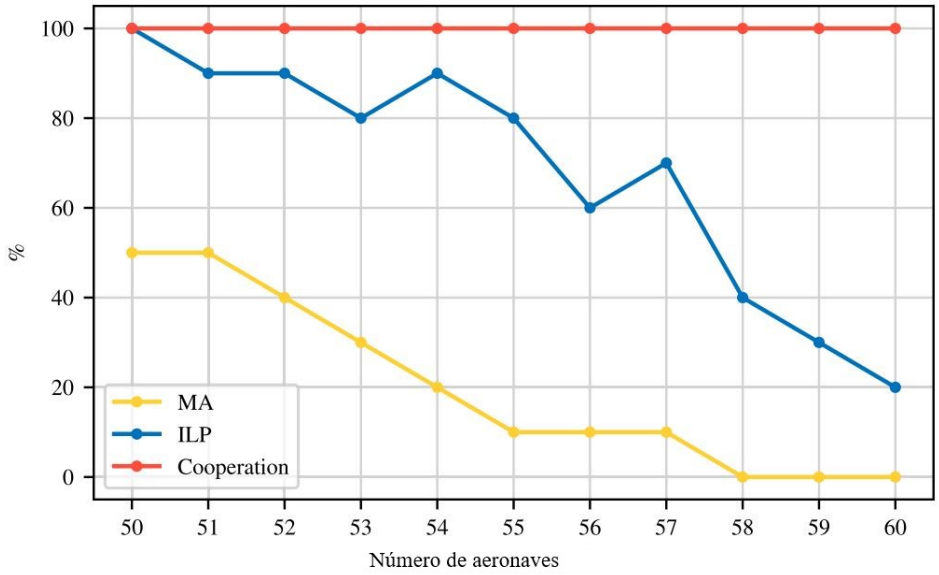

Figura 4.1: Porcentagem de sucesso para encontrar uma solução ótima com um limite de tempo de 300s com MA, ILP e sua cooperação, adaptado de (Fonte: [54]).

Nos testes realizados foi possível demonstrar que a cooperação entre os algoritmos apresenta resultados consistentemente melhores quando comparados com os resultados individuais dos algoritmos, como pode ser verificado na Figura 4.1, que apresenta o percentual de sucesso na identificação da escolha ótima, considerando duração máxima do algoritmo de 300 segundos, para os MA, ILP, e a cooperação entre eles.

Pelegrín e D'Ambrosio [55] apresentam uma síntese dos estudos produzidos pela comunidade de OR, nas últimas décadas, relacionados ao desenvolvimento de modelos matemáticos para auxiliar na detecção e resolução de conflitos no nível tático. Na análise comparativa dos 27 trabalhos analisados, consolidada na Figura 4.2, desenvolvidos entre 2001 e 2020, foram consideradas as seguintes características:

- *dimensões*: se foram consideradas duas ou três dimensões;

- *movimento*: se as trajetórias eram retilíneas ou aceleradas uniformemente;
- *separação*: se considerou restrições de separação ou se a região de segurança for um retângulo;
- *manobras*: se trabalhou apenas com alteração de velocidade, ângulo de direção ou vários níveis de voo;
- *recuperação de trajetória*; se as aeronaves retornaram à trajetória original (comum em trabalhos baseados em alteração no ângulo de direção);
- *incertezas*; se o estudo trabalhou com incertezas;
- *discretização*: se baseada em tempo, espaço ou manobras;
- *mp*: conforme o tipo de modelo de programação (OR) adotado;
- *obj*: conforme função objetivo definida.

Uma importante contribuição do trabalho está na demonstração, por meio de seis equações diferentes, das ligações e equivalências relativas às condições de separação *pairwise* de aeronaves, cuja aplicação direta fornece modelos não lineares não convexos, enquanto que as alternativas incluem simplificações que afetam a confiabilidade do modelo e/ou restrições matemáticas disjuntivas.

4.2 Inteligência Artificial no Contexto da CDR

Muitos dos trabalhos sobre CDR utilizam diferentes técnicas de inteligência artificial (AI) nas mais variadas modelagens, para estudar tanto cenários completos de gerenciamento do tráfego aéreo (ATM) como, também, investigar tópicos específicos da aviação civil comercial.

Neste sentido, especificamente sobre o impacto do clima, Rosenow *et. al.* [56] apresentam um estudo com foco na necessidade de reotimização de trajetórias em função de condições meteorológicas instáveis, com consequente impacto nas trajetórias das aeronaves e no consumo de combustível. A pesquisa investiga o benefício de uma otimização dinâmica da trajetória horária durante um voo entre Seattle e Nova York, considerando previsões meteorológicas horárias emitidas em 93 dias consecutivos.

As atualizações meteorológicas horárias são usadas para reotimizar a trajetória durante o voo e enfatizar o benefício de executar um plano de voo reotimizado assim que uma nova trajetória estiver disponível, conforme demonstrado na Figura 4.3. As trajetórias otimizadas são obtidas por meio do *TOolchain for Multicriteria Aircraft Trajectory*

Authors	Year	D	$\mathcal{M}$	Sep.	Maneuvers			R	U	Discretiz.			MP	Obj.
					v	ha	ℓ			T	S	M		
Frazzoli et al.	2001	2	u	(12)	✗	✗							MIQP	Deviation
Pallottino et al.	2002	2	u	(6)	✗								MILP	Time
Pallottino et al.	2002	2	u	(7)		✗							MILP	Time
Richards & How	2002	2	u	r	✗					✗			MILP	Time
Vela et al.	2009b	2	u	(23)	✗		✗				✗		MILP	Fuel, time
Vela et al.	2009a	2	u	(19)	✗				✗			✗	NLP	Fuel
Alonso-Ayuso et al.	2010	2	u	(6)	✗		✗	✗					MILP	Deviation, num. changes
Vela et al.	2010	2	u	(6)	✗	✗						✗	MILP	Fuel, fairness
Rey et al.	2012	2	u	(23)	✗						✗		MILP	Conf. duration
Omer & Farges	2013	2	ua	(1)	✗	✗		✗		✗			NLP	Deviation
Omer & Farges	2013	2	ua	(1)	✗	✗		✗		✗			MILP	Deviation
Cafieri & Durand	2014	2	u	(12)	✗			✗					MINLP	Deviation
Rey et al.	2014	2	u	(23)	✗						✗		MILP	Num. conf., fairness, time
Alonso-Ayuso et al.	2014	2	u	(6)		✗		✗					MINLP	Deviation
Omer	2015	2	ua	(23)	✗	✗		✗			✗	✗	MILP	Fuel, time
Rey et al.	2016	2	u	(23)	✗						✗		MILP	Num. conf., conf. duration
Alonso-Ayuso et al.	2016	2	u	(6)	✗	✗	✗						MINLP	Deviation
Cafieri & Omheni	2017	2	u	(12)		✗		✗					MINLP	Deviation
Cafieri & Omheni	2017	2	u	(12)	✗								MINLP	Num. conf.
Cafieri & Rey	2017	2	u	(12)	✗								MINLP	Largest conflict-free set
Rey & Hijazi	2017b	2	u	(17)	✗	✗							MINLP	Deviation
Lehouillier et al.	2017b	3	ua	(1)	✗	✗	✗	✗		✗		✗	MILP	Fuel, time
Lehouillier et al.	2017a	3	ua	(19)	✗	✗		✗	✗	✗		✗	MILP	Fuel, time
Cafieri & D'Ambrosio	2018	2	u	(12)	✗								MINLP	Deviation
Dias et al.	2020	2	u	(17)	✗	✗	✗						MI(Q)P	Deviation
Cerulli et al.	2020	3	u	(1)	✗			✗					BP	Deviation
Cerulli et al.	2020	2	u	(1)		✗		✗					BP	Deviation

Figura 4.2: Resumo das formulações de Programação Matemática existentes para detecção e resolução de conflitos (Fonte: [55]).

Optimization - TOMATO (Rosenow *et. al.* [57]), que atua sobre variáveis dinâmicas de entrada para recalcular as trajetórias.

No campo das redes neurais artificiais (RNA), Zhang e Mahadevan [58] apresentam uma abordagem bayesiana para lidar com incertezas na previsão de trajetórias com base em algoritmo de aprendizado profundo (DL), com foco na segurança de voo. A metodologia proposta consiste em quatro etapas. Na primeira etapa, é utilizada uma arquitetura Spark para processar um grande volume de mensagens no formato Flight Information Exchange Model (FIXM). Na segunda etapa, duas redes neurais são utilizadas: redes neurais profundas (DNN) são treinadas para fazer uma previsão um passo à frente no desvio ao longo da latitude e longitude entre a trajetória de voo real e a trajetória de voo alvo; redes neurais profundas de memória de curto prazo (LSTM) são treinadas para fazer previsões de longo prazo sobre a trajetória de voo em vários instantes de tempo subsequentes. Na terceira etapa, os dois tipos diferentes de modelos de aprendizado profundo são combi-

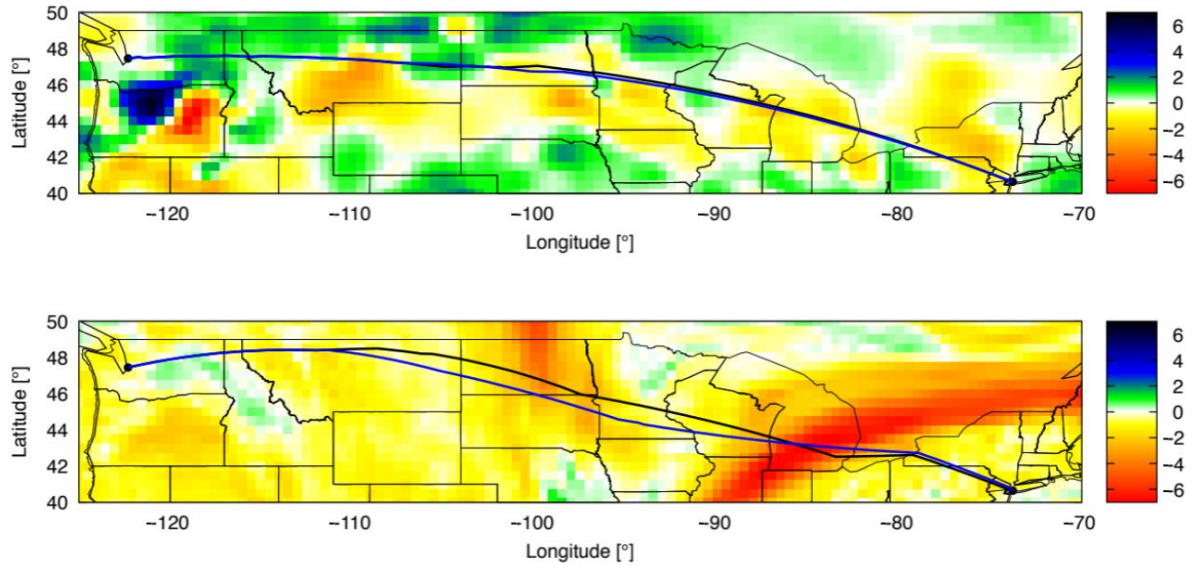

Figura 4.3: (Acima): Trajeto otimizado às 12h. (meia-noite, preto) e caminho reotimizado à 1h (azul) em 10 de agosto de 2019. (Inferior): caminhos reotimizados à 1h (preto) e 2h (azul) em 4 de janeiro de 2020. A cor de fundo indica diferenças na velocidade do vento ms^{-1} entre as previsões RAP correspondentes. (Fonte: [56]).

nados para criar uma previsão de fidelidade múltipla, utilizando a previsão DNN para corrigir a previsão LSTM da trajetória de voo ao longo dos instantes de tempo subsequentes. Na quarta e última etapa, o modelo fundido é utilizado para fazer previsões de trajetória em vários voos e avaliar a segurança de dois voos, onde a distância de separação é usada como uma métrica quantitativa de segurança.segurança de voo em rota.

O estudo apresenta contribuições significativas no campo da previsão de trajetórias, tais como a adoção de uma arquitetura distribuída para processar grandes volumes de dados e uma abordagem bayesiana para tratar incertezas. Por outro lado, os autores destacam a necessidade de incorporar o impacto do clima na trajetória de voo, de forma explicita, bem como a realização de testes com dados de voo de fora dos Estados Unidos, que apresentam características particulares.

Courchelle *et. al.* [59] apresentam uma abordagem baseada em *simulated annealing* (SA) para minimizar o número de conflitos entre aeronaves na fase estratégica do voo. A modelagem foca em alterações de velocidades para evitar conflitos, e considera incertezas na posição das aeronaves em virtude do vento.

Nos testes realizados, que contou com um conjunto de dados contendo mais de 1.000 voos, a abordagem possibilitou resolver cerca de 80-90% dos conflitos apenas com a alteração das velocidades antes da partida das aeronaves, ou seja, modificando os tempos de voo. Assim, mostra-se que a abordagem estratégica de resolução de conflitos poderia,

a priori, diminuir a carga de trabalho dos controladores e aumentar indiretamente a capacidade de operação. Uma extensão natural do trabalho consiste na implementação de outros tipos de manobras de separação, como mudanças de rumo ou nível de voo.

Liu e Hansen [60] propõem uma nova abordagem para previsão de trajetórias 4D reais de aeronaves, a partir de informações meteorológicas e dos últimos planos de voo arquivados. A abordagem é composta por um algoritmo de correspondência, um modelo generativo profundo, uma estrutura de treinamento e uma estrutura de inferência.

O algoritmo de correspondência combina com eficiência trajetórias de voo reais com informações climáticas (clima convectivo, temperatura do ar e velocidade do vento), gerando uma representação na forma de cubos de recursos. Os estados dos voos reais são modelados como misturas gaussianas condicionais com parâmetros a serem aprendidos do modelo generativo proposto, que consiste em um codificador multicamada LSTM, um decodificador multicamada LSTM e um conjunto de camadas convolucionais.

A rede do codificador insere a última sequência de plano de vôo arquivada e produz uma variável de estado oculto de tamanho fixo que é posteriormente alimentada na rede do decodificador. As camadas convolucionais aprendem as representações de recursos dos cubos de recursos correspondentes e são integradas à rede decodificadora para que a perda seja propagada de volta para seus pesos. Por fim, no processo de inferência, os planos de voo, os estados de voo observados e seus cubos de recursos correspondentes são alimentados no modelo treinado, obtendo-se os parâmetros de mistura gaussiana para o primeiro estado de voo previsto. Em seguida, é aplicado recursivamente um filtro de Kalman adaptativo que melhora o poder de previsão.

A abordagem proposta em [60] foi aplicada em conjunto de dados com 1.342 voos de treinamento, e avaliados com outros 337 voos, tendo sido observado que os filtros aprendidos localizam com sucesso o clima convectivo e generalizam bem os recursos relacionados ao clima. Também foram observados grandes erros de previsão para voos (outlier) com procedimentos de partida incomuns, que devem ser explorados em pesquisas futuras.

Ayhan *et. al.* [61] propõe um modelo de análise prescritiva para o problema de detecção e resolução de conflitos de aeronaves de longo alcance que, a partir de um conjunto de trajetórias previstas, identifica um conflito quando uma zona protegida de uma aeronave em sua trajetória é infringida por outra aeronave. A proposta indica soluções alternativas para resolução dos conflitos a partir do aprendizado com padrões descritivos de trajetórias históricas e registros meteorológicos, modelado com o uso de um Modelo Oculto de Markov (HMM).

O espaço aéreo de uma aeronave é modelado como um conjunto concatenado horizontal e verticalmente de data cubos, formando uma unidade atômica, conforme ilustrado na Figura 4.4.

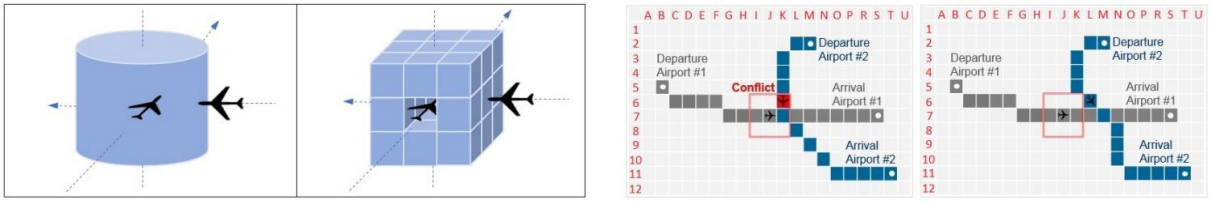

Figura 4.4: Exemplo de conflito de cruzamento em representação regular e na modelagem utilizando cubos, e representação simplificada da proposta de CDR. (Fonte: [61]).

A proposta apresentada em [61] obteve precisão média superior a 97% na resolução de conflitos, em todos os casos de teste realizados. Os resultados indicam a eficácia da modelagem, porém não garante que os conflitos são detectdos e resolvidos de forma definitiva, não ocorrendo conflitos posteriores. Provavelmente, haverá padrões climáticos convectivos após a partida que causarão conflitos potenciais que devem ser abordados taticamente por sistemas CDR de curto e/ou médio alcance.

No campo da Mobilidade Aérea Urbana (UAM), Yang e Wei [62] apresentam um algoritmo de orientação espcial a fim de permitir operações de voo livre sob demanda, autônomas, seguras e eficientes, com capacidade de evitar colisões. O algoritmo proposto formula o problema como um Processo de Decisão de Markov (MDP) e utiliza um algoritmo online basado em Monte Carlo Tree Search (MCTS) para solução.

Nas simulações realizadas com diferentes comportamentos, a proposta apresentada obteve desempenho promissor quando comparado com a abordagem clássica Optimal Reciprocal Collision Avoidance (ORCA) em cenários de tráfego aéreo denso. O algoritmo proposto fornece uma estrutura de solução potencial para permitir operações autônomas de voo livre sob demanda em UAM.

Em [63], Malakis *et. al.* apresentam um estudo focado na resposta de controladores de voo à compexidade de um cenário de tráfego, com o objetivo de explorar como árvores de decisão e regras de classificação podem ser usadas para classificação realista de cenários de tráfego aéreo, e para explocar quais fatores refletem melhor a complexidade operacional.

Dois algoritmos de aprendizado de máquina são utilizados na estruturação das árvores de classificação: CRT, que baseia-se em escolhas binárias mais facilmente compreensíveis, mas que leva a árvores e regras de decisão mais complexas; e, CHAID, que permite vários níveis de escolha. O estudo demonstra que a comparação entre as classificações do modelo e as reais feitas pelas equipes operacionais mostra um descompasso que pode acarretar não apenas problemas de eficiência, mas também questões de segurança.

Ribeiro *et. al.* [64] propõem um modelo baseado em dois algoritmos distintos, SA e Programação Inteira Binária (BIP) para resolução de conflitos, com foco no espaço aéreo brasileiro. Neste modelo proposto, a vizinhança de soluções é projetada para ajustar

alguma aeronave que já esteja envolvida em um conflito. Assim, a avaliação do estado energético das soluções inclui a atribuição de uma probabilidade uniforme para as aeronaves que atendem a esse requisito. O diagrama de fluxo para detecção e resolução de conflitos é mostrado na Figura 4.5.

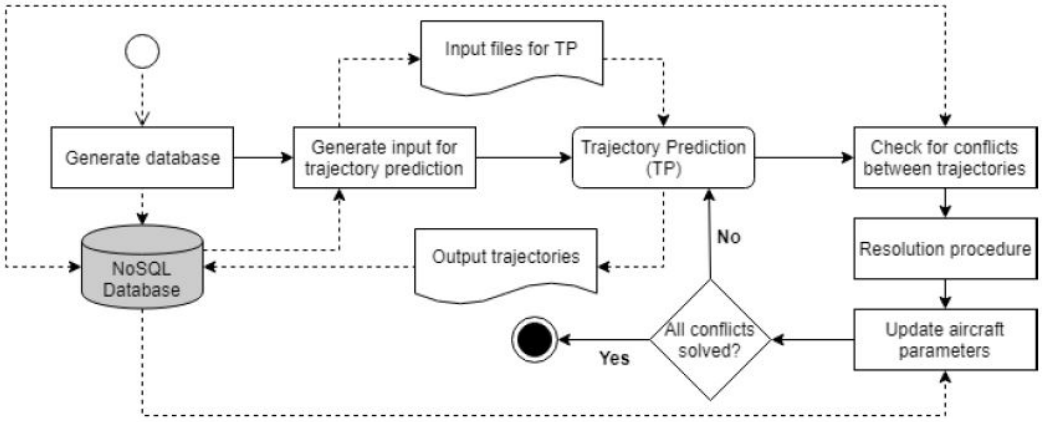

Figura 4.5: Diagrama de fluxo para detecção e resolução de conflitos. (Fonte: [64]).

O modelo BIP apresentou melhor desempenho que o modelo SA. O processo de SA tem a propriedade inerente de imprimir soluções piores sob uma dada probabilidade de evitar soluções sub-ótimas, enquanto a abordagem binária é conduzida para o mínimo custo percebido em cada etapa. Assim, menos atualizações são necessárias e o cenário gerado apresenta custos menores. Além disso, observou-se que o SA não atende à propriedade de equidade, o que significa que a distribuição de custos é desigual entre as aeronaves atualizadas.

O estudo de Pamplona *et. al.* [65] foca no problema do atraso no tráfego aéreo, que resulta em aumento de custos para as companhias aéreas e desconforto para os passageiros. Um dos esforços envolvidos no ATM consiste em reduzir os níveis de atraso, e o trabalho apresentado aplica Redes Neurais Artificiais (RNA) em um modelo de previsão de atrasos aéreos na rota São Paulo (Congonhas) - Rio de Janeiro (Santos Dumont).

A aplicação da RNA proposta foi capaz de auxiliar na previsão dos atrasos aéreos utilizando os dados reais dos cenários de tráfego aéreo brasileiro. Para medir o desempenho da predição, são aplicadas as métricas *recall*, *precision* e *f-score*, e os resultados do estudo de caso apresentaram uma capacidade preditiva de acertos superior a 90%, e demonstrou, em contraponto a outros estudos, que o dia da semana, o horário e a companhia aérea possuem maior influência do que a meteorologia no atraso aéreo.

No campo dos sistemas multiagentes, Weigang *et. al.* [66] descrevem dois sistemas, ATFMGC e SISCONFLUX, permitindo demonstrar que devem ser considerados dois ti-

pos de fluxo de tráfego distintos para o ATFM: o fluxo de voo propriamente dito (ATFM) e o fluxo de mensagens de comunicação (gerenciamento do fluxo de mensagens, IFM). Nos testes realizados, verificou-se que o fluxo de mensagens aumentava consideravelmente em determinados horários, onde um agente realizava e recebia diversas requisições simultaneamente. No modelo desenvolvido, esse problema foi inicialmente resolvido com a execução completa do ciclo de análise de conflitos, mas esse procedimento gerou processamentos extras que podem ser potencialmente evitados. Identificou-se, portanto, a necessidade de priorizar e organizar as filas de pedidos, de forma a otimizar os resultados do ATM.

Uma referência fundamental para este trabalho, Vitor [19] investiga o problema de descobrir um esquema eficiente para armazenar e gerenciar as trajetórias na complexa rede de tráfego, com massiva quantidade de dados, para então ser possível detectar e resolver os conflitos, por meio do desenvolvimento de um framework para CDR para o gerenciamento de trajetórias quadridimensionais (4D), ilustrado na Figura 4.6.

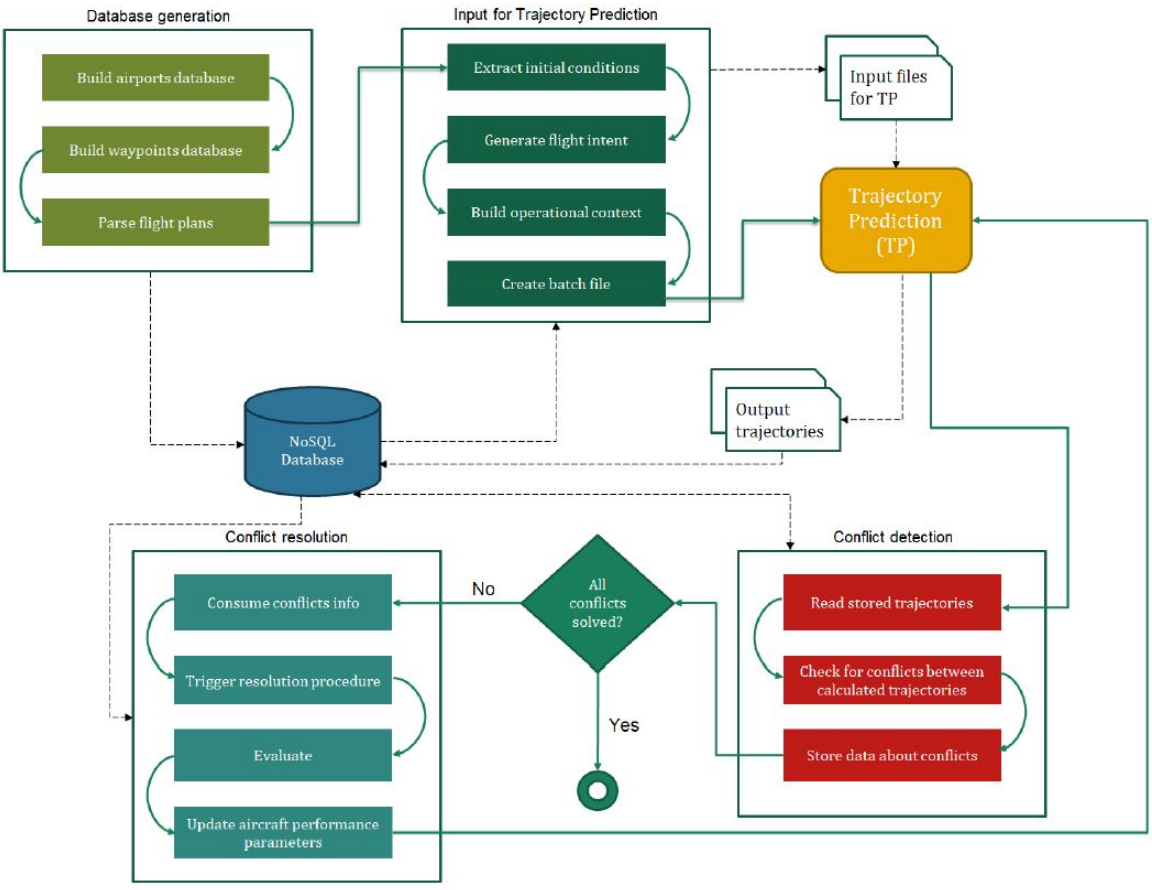

Figura 4.6: Diagrama de sequência para detecção e resolução de conflitos. (Fonte: [19]).

O framework proposto em [19] utiliza uma arquitetura baseada em banco de dados

NoSQL para armazenar, manipular e computar as trajetórias 4D de voos comerciais, e os eventuais conflitos são detectados em complexidade computacional na ordem de O(nlog2(n)) no caso médio, superando técnicas do estado da arte. Por fim, o trabalho apresenta uma proposta para resolução de conflitos baseado em busca local, responsável por ajustar os parâmetros do plano de voo a fim de resolver os conflitos em tempo de planejamento estratégico do voo.

4.3 Trabalhos Consolidados com Análises Comparativas

O trabalho de Fan [67] apresenta uma revisão das abordagens tradicionalmente utilizadas para gerenciamento de separação no ATM. Tais abordagens são classificadas como: **descentralizadas**, quando as aeronaves tomam decisões individuais com base nas informações de bordo; ou **centralizadas**, quando há uma entidade central tomando decisões por todas as aeronaves.

Incluídas no rol de abordagens **descentralizadas**, destacam-se [67]:

- **Teoria dos Jogos não cooperativa** [68]. Situação em que os agentes tomam decisões conforme as decisões dos demais agentes. O comportamento dos agentes é modelado matematicamente e a estratégia mais segura é considerar a pior ação dos demais agentes. A impossibilidade de se saber o exato estado das demais aeronaves dificulta a aplicação dessa metodologia em cenários com incerteza.
- **Abordagem Míope** [69]. Utiliza uma estratégia orientada a usuário que enfatiza a eficiência individual das aeronaves. Um conflito ocorre se for detectado em um período de tempo de até 8 minutos, e os conflitos são resolvidos aos pares: apenas o conflito imediato será resolvido. O modelo determina a solução mais eficiente e executa as manobras que requerem a menor mudança de rota. É uma abordagem clássica, porém falta capacidade de resolução em cenários com voos em formação (quando há uma aeronave "líder").
- ***Look-ahead***. Abordagem similar a anterior onde primeiramente a manobra mais eficiente é determinada, e verifica-se a criação de novo conflito em um intervalo menor de tempo do que o conflito resolvido. Se um novo conflito é detectado, outra solução é avaliada (mudanças de rota de 5 graus) até encontrar uma rota livre de conflito. Ademais, essa estratégia contorna a desvantagem da abordagem anterior relacionada ao efeito dominó, porém carece da capacidade de resolver múltiplos efeitos dominó. A Figura 4.7 apresenta a comparação entre as abordagens Míope e *Look-ahread*.

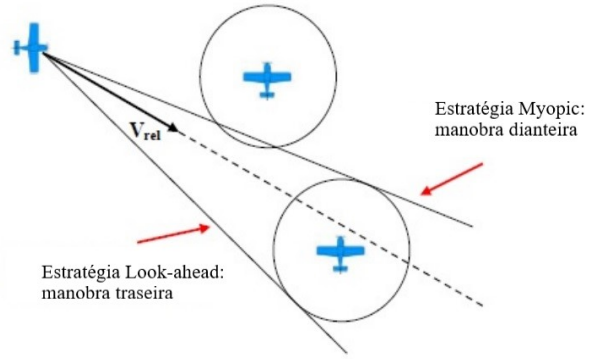

Figura 4.7: Comparativo entre as abordagens Míope e *Look-ahread*, adaptado de (Fonte: [67]).

- **Sistema Automático de Prevenção de Colisão Aérea – Auto ACAS** [70]. Nessa abordagem são utilizadas manobras de escape ideais para evitar colisão, de forma que o sistema assume o controle da aeronave no último instante possível para evitar colisão. É projetado para funcionar mesmo com falha de GPS ou queda no link de dados, sendo complementar às demais abordagens de separação de aeronaves. O sistema calcula três manobras possíveis e compartilha com as aeronaves próximas, então o sistema seleciona a melhor manobra (que maximiza a separação mínima). Apesar dos resultados positivos, o impacto de baixas taxas de comunicação não foi investigado. Outra necessidade é desenvolver o algoritmo visando garantir proteção para aeronaves não equipadas com o Auto ACAS.
- **Modelo de Controle Preditivo** [71], [72]. O modelo, utilizado na indústria desde 1980, busca prever o comportamento de variáveis dependentes (saídas) em relação a alterações nas variáveis independentes (entradas). O controle de horizonte de recuo permite a inclusão de requisitos de prevenção de colisão entre pares de aeronaves. O esforço computacional é viável, porém há limitações relacionadas a omissão de rajadas de vento e dinâmica de voo. Outras abordagens consideram requisitos de comunicação (incerteza), porém com impacto na performance da computação.
- **Kripke** [73]. A metodologia utiliza-se de uma semântica formal para sistemas lógicos não clássicos, onde foram utilizadas declarações lógicas temporais para lidar com diferentes tipos de incerteza e dinâmica em múltiplas aeronaves cooperantes. A abordagem possui a vantagem de representar incerteza no mundo real usando um modelo formal/intuitivo de grafo direcionado, porém sem abordar explicitamente o desempenho.

- **Previsão de Colisão Reativa Descentralizada** [74]. É adotado o conceito de cone de colisão. Primeiramente, os conflitos são resolvidos com manobras iniciais de desconexão. Em seguida, a aeronave muda progressivamente para direções desejadas após a resolução do conflitos. O modelo possibilita que veículos tenham tamanhos, velocidades, limitações de atuação e ganhos diferentes, porém é incapaz de garantir uma resolução de conflitos bem-sucedida. Adicionalmente, o método não lida com aeronaves não cooperativas, dificultando a usabilidade.

Por sua vez, entre as abordagens **centralizadas**, destacam-se [67]:

- **Força Bruta** [75]. Envolve enumeração sistemática de todas as soluções possíveis para um problema. Pode ser satisfatório para um número reduzido de aeronaves, porém para um número grande de aeronaves o tempo de execução é inviável.
- **Algoritmo Genético** [76]. Técnica de busca inspirada na biologia evolucionária para encontrar soluções ótimas globais ou locais. O algoritmo genético considera incertezas relacionadas à velocidade da aeronave e, por conta disso, as manobras devem começar o quanto antes para evitar novos conflitos. A desvantagem é o custo computacional.
- **Programação Semidefinida** [77]. Muitos problemas práticos em pesquisa operacional e otimização combinatória utilizam tal abordagem. O problema é formulado como um programa quadrático não-convexo, e as restrições são definidas como sendo de três tipos: colisão (localização da aeronave); manobra (velocidade) e função de custo (minimizar desvios de velocidade). A otimização é resolvida por meio de programação semidefinida combinada com um esquema de randomização. Simulações envolvendo duas aeronaves demonstram boa habilidade do algoritmo, porém falta aprofundar a análise de problemas como falta de comunicação.
- **Programação Inteira Mista** [78]. Programação linear envolvendo restrições modeladas por equações lineares cujas variáveis podem ser tanto inteiras quanto reais (mista). O sistema possui duas partes: a primeira parte define restrições relacionadas aos conflitos, tais como velocidade e ângulo da aeronave; e a segunda trata da otimização do tempo de voo (minimização). O sistema é capaz de tratar situações com muitas aeronaves, porém problemas de medição não foram abordados.
- **Grid-Based** [79]. A abordagem utiliza representação 4D na qual as células que representam trajetórias são setadas com 1 e zonas restritas podem ser representadas na grid de forma apropriada. É possível representação estocástica com o armazenamento da probabilidade setando a célula com valor entre 0 e 1. A abordagem

possibilita representar a incerteza da trajetória, porém com alto custo computacional.

- **Abordagem satisfatória** [80]. Abordagem multiagente envolvendo Teoria dos Jogos na qual ações conjuntas são realizadas pelos agentes. As ações são determinadas a partir de duas propriedades, e as opções são: -5, -2,5, 0, +2,5 e +5 grau. Possui as seguintes propriedades: seletividade, considera o impacto na trajetória desejada; e rejeitabilidade, considera os conflitos envolvidos. Não busca a melhor solução, mas sim propor um conjunto de manobras possíveis, e custo computacional aumenta conforme o aumento de aeronaves.
- **Algoritmo de Separação de Classificação Baseado em Atraso** [81]. Baseia-se no anterior: as aeronaves são ordenadas conforme horas de voo e atraso. A ordem de mudança na trajetória é baseada no ranking. O custo computacional aumenta conforme o aumento de aeronaves.

Por fim, Tang [17] apresenta um levantamento contendo vários estudos sobre detecção e resolução de conflitos, mapeando características apontadas em cada estudo que contribuem para a formação de uma taxonomia específica. Foram consideradas as seguintes características:

- *detection*: possui ou não algoritmo de detecção de conflitos;
- *resolution*: considera ou não manobras para resolução de conflitos;
- *cooperation*: indica se o modelo é cooperativo, não cooperativo ou híbrido (ambos);
- *maneuvers*: controle de angulação, controle de altitude, controle de velocidade, combinação dos controles anteriores ou outras situações;
- *domino effects*: se contempla conflitos provocados pelo próprio modelo ou não;
- *multiple conflicts*: se utiliza abordagem *pairwise* ou global;
- *uncertainties*: considera ou não fontes de incerteza;
- *state variables*: continuas, discretas ou híbrida;
- *equipment location*: em solo ou na aeronave;
- *method status*: se o método já está em uso pelo ATM ou se é objeto de pesquisa.

Os trabalhos estudados foram classificados em três categorias distintas: *long term CDR*, que abrange o planejamento estratégico, lidando com horizontes de tempo superiores a 30 minutos; *medium term CDR*, que considera o planejamento tático do voo, com

Summary of Models Addressing the Short-Term CDR Problem

Model	Detection	Resolution	Cooperation	Maneuvers	Domino effects	Multiple conflicts	Uncertainties	State variables	Equipped locations	Methods status
Ford [79]	√	–	CO	–	–	MP	–	CT	AB	U
Gong [80]	√	–	CO	–	–	MP	√	DC	GB	R
Liu [81]	√	–	CO	–	–	MG	√	CT	GB	R
Clements [82]	–	√	CO	C(AS)	–	MG	√	CT	GB/AB	R
Alonso-Ayuso [83]	–	√	CO	C(HAS)	–	MG	–	CT	GB/AB	R
Chen [84]	√	√	CO	A	–	MG	–	DC	GB/AB	R
Barhydt [85]	√	√	BC	A	–	MP	√	DC	AB	R
Cafieri [86]	√	√	CO	C(HS)	–	MG	–	CT	GB	R
Blom [87]	√	√	CO	H	–	MG	√	CT	AB	R
Hu [88]	√	√	NC	C(HA)	–	MP	√	CT	GB/AB	R
Archambault [89]	√	√	NC	C(HAS)	–	MP	√	CT	GB	R
Chaloulos [90]	√	√	NC	C(HS)	–	MP	–	CT	GB/AB	R

Figura 4.8: Resumo de modelos que abordam o problema *short term CDR*. (Fonte: [17]).

horizontes de tempos de até 30 minutos; e *short term CDR*, cujo foco é a fase operacional do voo, abordando horizontes de tempo de até 10 minutos. Na Figura 4.8, apresentamos o consolidado obtido para os estudos incluídos na categoria *short term CDR*.

O estudo feito por Tang [17] preenche uma lacuna importante da pouca discussão ou avaliação comparativa entre as várias abordagens para CDR, além de sugerir uma taxonomia que se mostrou adequada para comparações. Como trata-se de um contexto dinâmico, é proposta a inclusão de novas questões para uma análise mais precisa e maior alcance dos levantamentos dessa natureza.

4.4 Contribuições desta Pesquisa

Como pode ser observado nas seções anteriores, existe uma variedade de estudos e metodologias que se propõem a tratar o problema de CDR. Devido à complexidade do problema, que envolve a manipulação de diversas variáveis, os estudos geralmente concentram-se em grupos de variáveis, sendo obrigados a definir algumas aproximações para que os modelos possam ser desenvolvidos.

Exemplificando, alguns trabalhos focam em componentes de incerteza mas não consideram a navegação 4D; outras abordagens trabalham com navegação 4D mas tem dificuldade de lidar com eventos não programados; e assim por diante. Uma provável causa dessas lacunas pode estar relacionada ao volume de dados envolvidos em cada uma dessas variáveis, o que torma computacionalmente custoso trabalhos mais abrangentes. Este trabalho busca justamente lidar de forma eficiente com grandes volumes de dados, compatibilizando as lacunas existentes entre trabalhos anteriores.

Capítulo 5

Modelagem 1: Framework Baseado em NoSQL

Neste Capítulo será apresentada a primeira abordagem para detecção e resolução de conflitos, baseada em bancos de dados NoSQL e algoritmos de busca. Inicialmente, serão apresentados os conceitos utilizados na previsão de trajetórias. Em seguida, a modelagem dos dados e a proposta de detecção de conflitos são descritas. Por fim, a metodologia de resolução de conflitos e detalhes sobre a complexidade são abordados. O fluxo de trabalho completo para a detecção e resolução de conflitos propostos é mostrado na Figura 5.1.

5.1 Previsão de Trajetórias

Uma trajetória de uma aeronave é uma sequência de pontos bidimensionais que formam um conjunto conectado de segmentos. Resolver conflitos significa ajustar os planos de voo predefinidos para atualizar as trajetórias previstas. O conjunto de *waypoints* $P = (p_1, ...p_n)$ é capaz de descrever uma trajetória em um plano 2D, mas deve ser refinado para acomodar outras variáveis de controle e uma restrição cronológica para descrever uma trajetória 4D completa [82]. Em outras palavras, precisamos de um conjunto $W = w_1, ..., w_n$ desde que $w_i = (x_i, y_i, z_i, t_i) \in \mathbb{R}^4$ seja um ponto de trajetória 4D onde x_i, y_i, z_i são dimensões espaciais e t_i é tempo.

O cálculo da trajetória deve considerar não apenas o conjunto predefinido de *waypoints* informados no plano de voo (origem, destino e alguns pontos fixos intermediários), mas também os parâmetros que afetam o perfil de voo, como massa e velocidade de decolagem. Modelos de clima e desempenho de aeronaves formam o roteiro de voo e são entradas necessárias para qualquer sistema preditor de trajetória [83]. O produto do processo de cálculo da trajetória é chamado de Intenção da Aeronave, que informa exatamente como

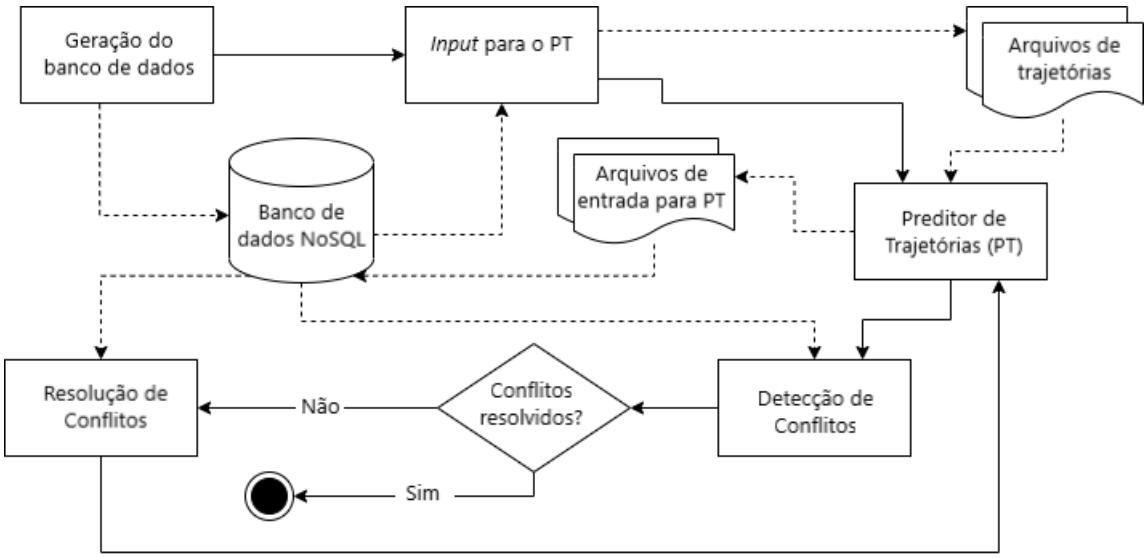

Figura 5.1: Diagrama de sequência para detecção e resolução de conflitos [2]

a aeronave deve cumprir o plano de voo. Como resultado, é fornecida uma descrição inequívoca da trajetória de voo pretendida.

A comunicação contínua de informações completas e atualizadas entre a aeronave e os prestadores de serviços de solo possibilita processos adequados de tomada de decisão [84]. No contexto de navegação 4D, a principal informação são as trajetórias livres de conflito calculadas pelos operadores terrestres que devem ser emitidas para a aeronave em voo. Isso só é possível quando há consciência comum do cenário do tráfego aéreo, o que significa entendimento inequívoco das intenções de voo. A *Aircraft Intent Description Language* (AIDL) é usada para esse fim [85]. Esta linguagem foi originalmente proposta por Vilaplana *et. al.* [86].

O framework construído sob o conceito AIDL, vide Figura 5.1, é utilizado como serviço para o cálculo de trajetória na ferramenta proposta por Ribeiro [19] e revisitada neste trabalho. A previsão de trajetória utilizada neste framework recebe os parâmetros da aeronave e outras informações sobre a intenção de voo em um determinado momento, e retorna a trajetória calculada. Este processo é executado de forma iterativa, e a cada novo ciclo, os pontos de trajetória calculados pela previsão de trajetória são utilizados no processo de detecção de conflitos. Uma descrição detalhada das equações de movimento e seus resultados podem ser encontrados no trabalho de Vilaplana *et. al.* [85].

Uma característica importante dessa estrutura é que a modelagem não considera apenas as trajetórias preditas padrão, mas também considera um grupo de trajetórias alternativas. Toda a trajetória definida é calculada simultaneamente, o que permite identificar um conflito entre as trajetórias previstas e agrupar as aeronaves em clusters de conflitos. Portanto, é possível buscar uma solução que combine as trajetórias previstas e as

trajetórias alternativas para garantir que a decisão de realizar uma manobra (a escolha de uma trajetória alternativa) não gere um novo conflito. Além disso, como a trajetória alternativa também foi calculada, os possíveis conflitos que essa nova trajetória poderia causar podem ser avaliados pelo framework no momento da seleção.

5.2 Proposta de Modelagem de Dados 4D

Os bancos de dados NoSQL projetados por Ribeiro [19] foram propostos para serem compatíveis com o paradigma SWIM, pois podem ser usados para armazenamento massivo de trajetórias, modelos meteorológicos e dados de aeronaves em qualquer nível de detalhe.

Embora não haja vantagem para um tipo específico de banco de dados NoSQL, em comparação com o banco de dados SQL tradicional, a vantagem de desempenho para NoSQL é óbvia [87]. Por isso, foram realizados testes preliminares em outros tipos de banco de dados NoSQL, dentre os quais Cassandra e MongoDB tiveram os melhores desempenhos entre as tecnologias avaliadas. Para implementar a funcionalidade de detecção de conflito em trajetórias 4D, deve-se definir o tipo de dado a ser recuperado. Relacionamentos entre as entidades não são possíveis por meio de chaves estrangeiras. Assim, todo relacionamento distinto depende da implementação de entidades distintas: famílias de colunas para Cassandra e coleções para MongoDB, que são exemplos importantes desses tipos de bancos de dados. Isso pode resultar em replicação de dados, mas no paradigma NoSQL, na verdade, é considerado um recurso em vez de uma desvantagem.

As prováveis consultas desejadas nesta implementação devem ser: plano de voo de uma aeronave, trajetória de uma aeronave, pontos em uma trajetória, pontos ocupados no tempo t e conflitos de trajetória.

O Preditor de Trajetórias, conforme descrito na Seção 2.5, é capaz de calcular trajetórias descritas em AIDL e gerar um arquivo KML como saída. Este arquivo usa a sintaxe XML para representar cada ponto de amostra em uma trajetória 4D. Assim, os atributos do ponto de amostra (x, y, h, t) de uma trajetória prevista podem ser usados como entrada para qualquer ferramenta de detecção de conflito. Na abordagem proposta, a chave de partição é formada pelo número do voo e pela estratégia de trajetória. Uma ordem de agrupamento na coluna denominada tempo é aplicada para que os *waypoints* da trajetória sejam ordenados por seu carimbo de data/hora em ordem crescente. As entidades são projetadas de forma semelhante tanto nos bancos de dados Cassandra quanto no MongoDB e podem ser consultadas para trazer todo o conjunto de *waypoints* (ordenados) que formam a trajetória de alguma aeronave. O Preditor de Trajetórias adotado neste trabalho, descrito na Seção 2.5, foi desenvolvido pela Boeing Research.

```
CREATE TABLE IF NOT EXISTS conflictpairs (
    flightnum text,
    interceptorid text,
    strategy1st int,
    strategy2nd int,
    cost1st double,
    cost2nd double,
    PRIMARY KEY ((flightnum, interceptorid),
        strategy1st, strategy2nd
    ));
CREATE INDEX ON conflictpairs( interceptorid );
```

```
{
  "flightNum":"GLO1461",
  "interceptorId":"TAM4537",
  "strategy1st":"1",
  "strategy2nd":"1",
  "cost1st":4100.97481,
  "cost2nd":4414.68067
}
```

Figura 5.2: Exemplo de estrutura de dados Cassandra e MongoDB.

O sistema proposto implementa um banco de dados Cassandra e um MongoDB para armazenar dados para facilitar a detecção de conflitos. Como o algoritmo de detecção de conflitos precisa manipular uma grande quantidade de dados, cada domínio de informação foi modelado como uma família de colunas distinta ou coleção de documentos para melhor desempenho na leitura e gravação de dados, dependendo do banco de dados do assunto. Cada entidade modelada representa a informação correspondente a um registro em uma tabela relacional. A Figura 5.2 apresenta um exemplo da estrutura de dados final na modelagem Cassandra e MongoDB. Também publicamos o dataset no GitHub, veja link: https://github.com/lucasbmonteiro/translab-cdr.

É necessário reforçar a importância do banco de dados para a solução proposta, não só no armazenamento de trajetórias originais e na capacidade de processar grandes volumes de dados, mas também por ser um agente importante em praticamente todas as etapas do framework. Além das trajetórias pré-estabelecidas, utilizadas no início do processamento, o banco de dados é responsável por armazenar as trajetórias alternativas calculadas (todos os pontos de todas as trajetórias de todas as aeronaves), além de registrar, também os vários estados que são gerados e representados nas árvores de busca. Trata-se, portanto, de uma componente essencial do trabalho, que demandou dedicação considerável em pesquisa e desenvolvimento.

5.3 Detecção de Conflitos

Um conflito envolvendo duas ou mais aeronaves em voo é o cenário em que a separação mínima entre elas é comprometida em um ou mais pontos no tempo. As distâncias mínimas horizontais e verticais devem ser asseguradas durante toda a execução dos voos, que são definidas respectivamente como 1.000 pés em níveis de voo abaixo de 41.000 pés em espaço aéreo RVSM e cinco milhas náuticas. Portanto, toda zona exclusiva ao redor de uma aeronave é um cilindro centrado nela. A restrição de tempo é adicionada às trajetórias 4D, então o conflito é detectado se esta zona espacial exclusiva for violada

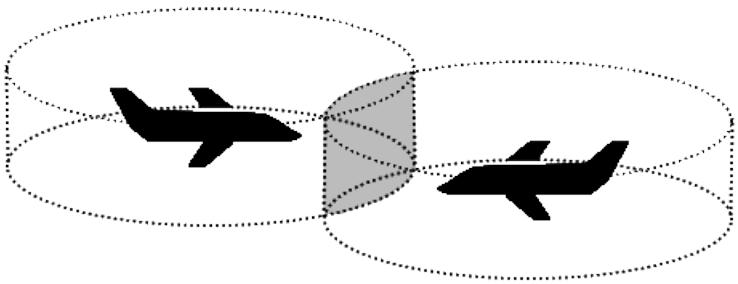

Figura 5.3: Violação de zonas exclusivas entre duas aeronaves.

dentro de uma janela de tempo específica. A Figura 5.3 ilustra uma situação em que duas aeronaves estão em conflito.

Então, é possível identificar a relação lógica que define a existência de um conflito c entre as aeronaves A_i e A_j em um instante t, conforme

$$c^{A_iA_j}(t) \leftrightarrow ({d_h}^{A_iA_j}(t) < S_h) \wedge ({d_v}^{A_iA_j}(t) < S_v) \tag{5.1}$$

onde ${d_h}^{A_iA_j}(t)$ é a distância horizontal entre A_i e A_i no instante t, t, ${d_v}^{A_iA_j}(t)$ é a distância vertical entre A_i e A_j no instante t e S_h e S_v são, respectivamente, a separação horizontal e vertical mínima exigida.

A distância horizontal entre os pontos w_i e w_j é dada pela fórmula de Haversine, conforme Equação 5.2. A equação calcula a distância geodésica entre as projeções de pontos 2D para a superfície da esfera da Terra, onde latitude e longitude são expressas em radianos, e R é o raio da Terra, e equivale a 6.371 km. Na Equação 5.2, $\Delta\phi$ corresponde à diferença entre as latitudes das duas aeronaves i e j, $\Delta\lambda$.

$$\begin{aligned} \Delta\phi &= |lat_i - lat_j| \\ \Delta\lambda &= |long_i - long_j| \\ a &= sin^2\Big(\frac{\Delta\phi}{2}\Big) + cos(lat_i) \cdot cos(lat_j) \cdot sin^2\Big(\frac{\Delta\lambda}{2}\Big) \\ c &= 2 \cdot arctan\Big(\sqrt{a}, \sqrt{1-a}\Big) d_h^{w_i,w_j} = R.c \end{aligned} \tag{5.2}$$

O algoritmo de detecção de conflitos implementa as restrições de separação entre as aeronaves descritas na Equação 5.1: em um determinado intervalo de tempo, dois pontos devem estar a uma distância mínima tanto na vertical quanto na horizontal. Se todas as condições expressas pela equação forem satisfeitas, os pontos comparados entram em conflito; caso contrário, se pelo menos uma das condições falhar, não há conflito entre os pontos analisados.

A abordagem proposta permite que os pontos sejam analisados aos pares, garantindo

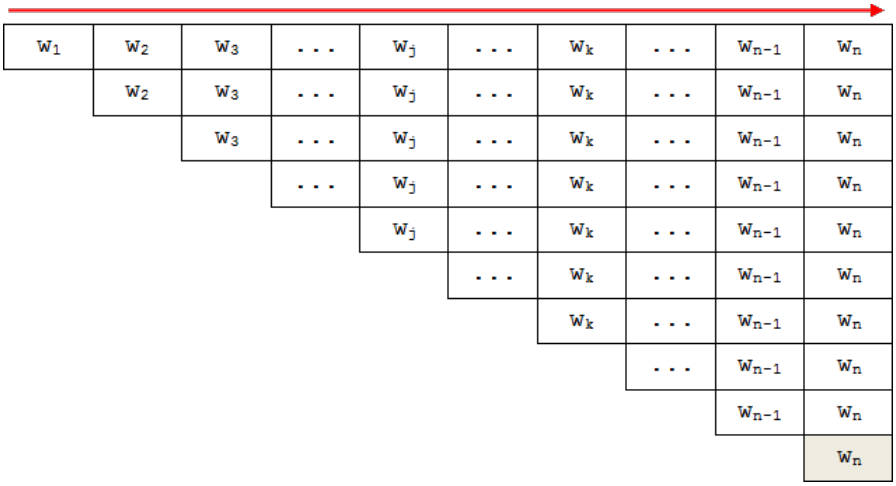

Figura 5.4: Esquema de consultas comparativas entre *waypoints*.

maior eficiência na redução do espaço de busca: como os pontos são ordenados por sua altitude, se houver separação vertical entre o primeiro e o segundo ponto, é possível inferir que não há conflito entre o primeiro e terceiro pontos e os pontos seguintes. Em outras palavras, nenhuma verificação adicional envolvendo o primeiro ponto é necessária. Esta abordagem permite, no melhor caso, a identificação de todos os conflitos em tempo linear. Se houver um conflito vertical entre dois pontos sucessivos, o algoritmo então verifica se existe uma separação horizontal mínima entre ambos, registrando um caso de conflito positivo. Se não houver conflito, o algoritmo verifica se há violação da separação vertical mínima entre o primeiro e o terceiro pontos (pior caso e caso médio), conforme ilustrado na Figura 5.4, onde o waypoint w_1 é comparado com todos os *waypoints* da sequência, até w_n; w_2 é comparado com todos os *waypoints* subseqüentes (exceto w_1 que é anterior a ele) até w_n, e assim por diante. A partir do ponto em que não há violação de separação vertical, o primeiro ponto é decaído, ou seja, não precisa ser analisado novamente.

5.4 Resolução de Conflitos

A proposta de resolução de conflitos ora apresentada difere da contida no trabalho de Ribeiro [19] tratando-se, portanto, de uma inovação do framework. Na etapa de detecção de conflitos, foram formados clusters de conflito em que cada cluster contém a totalidade de aeronaves envolvidas em algum conflito e suas respectivas trajetórias, default e alternativas. O desafio passa a ser identificar, para cada cluster, uma combinação ótima de trajetórias individuais que representem não apenas a não ocorrência de conflito, mas também a combinação mais satisfatória, com base em alguma política previamente escolhida (custo do voo, prioridades, impacto no espaço aéreo, etc).

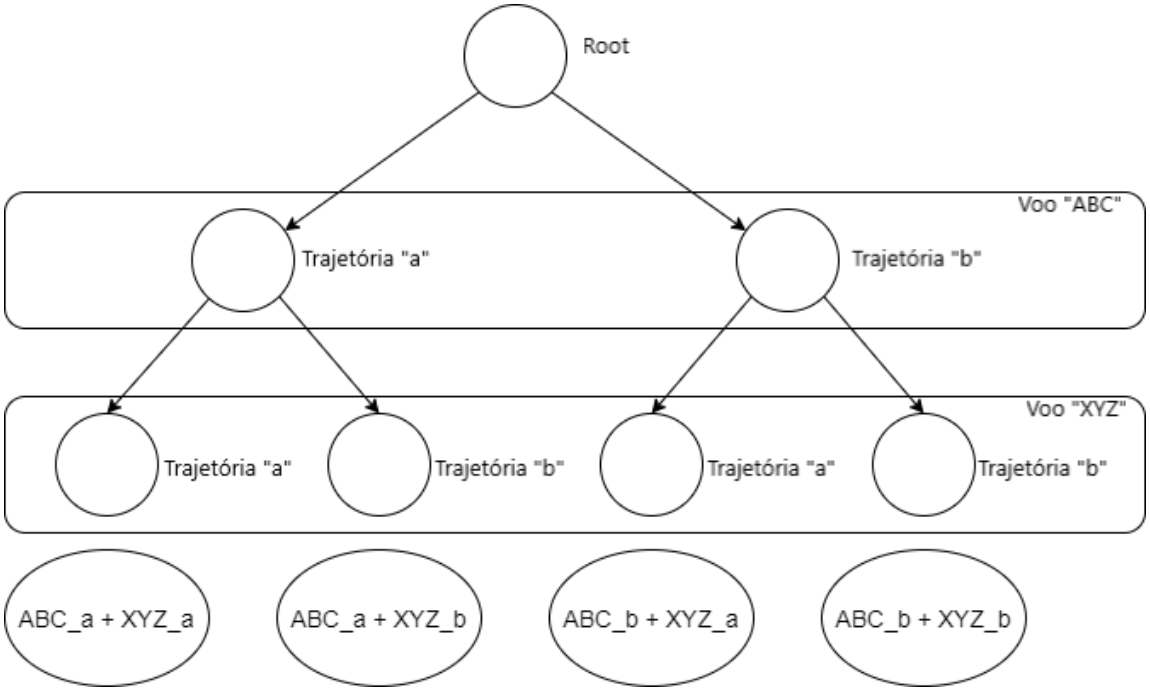

Figura 5.5: Exemplo do modelo de árvore de combinação de trajetória.

Na modelagem de resolução de conflitos, os estados possíveis, ou seja, as combinações de todas as trajetórias de todas as aeronaves, foram organizados na forma da estrutura de dados em árvore, de forma que cada nível da árvore corresponde a uma aeronave, cada nó de um nível corresponde a uma trajetória dessa aeronave e um caminho entre o nó raiz e um nó folha representa a escolha de uma combinação de trajetórias. Essa modelagem está representada na Figura 5.5.

No exemplo ilustrado na Figura 5.5, constam 4 estados possíveis oriundos da combinação de duas trajetórias,“a” e “b”, da aeronave “ABC”, e duas trajetórias, “a” e “b”, da aeronave “XYZ”, que leva aos estados: 1) voos “ABC” e “XYZ” adotam suas trajetórias “a”; 2) voo “ABC” adota trajetória “a”, enquanto que o voo “XYZ” adota trajetória “b”, 3) voo “ABC” adota trajetória “b”, enquanto que o voo “XYZ” adota trajetória “a”; e 4) ambas as aeronaves adotam suas trajetórias “b”.

A cada nó da árvore foi atribuído um *score* (parâmetro) que representa o quão adequada é aquela escolha, ou seja, os nós que representam maior pontuação indicam uma pior escolha, enquanto os nós que representam uma melhor escolha têm menor pontuação. Desta forma, nós que indicam trajetórias envolvidas em conflitos possuem pontuação máxima, de forma a desestimular a escolha de qualquer combinação que contenha aquele nó (caminhos que chegam a um nó folha por este nó). Alguns parâmetros que podem ser adotados como *score* são: consumo de combustível, atraso acumulado no voo; ordem de priorização de aeronaves; dentre outros. Nos testes, optou-se por um parâmetro genérico

que, em cenários reais, pode representar, inclusive uma função de diferentes fatores.

A modelagem sugerida permite identificar uma combinação ótima de trajetórias, sem conflitos, usando algoritmos clássicos de busca. Duas abordagens baseadas nesta classe de algoritmos foram implementadas: i) força bruta, baseada no algoritmo de busca em profundidade (DFS); e ii) baseado em Alfa-Beta.

5.4.1 Abordagem DFS

A abordagem baseada em DFS consiste na montagem completa da árvore (todos os estados possíveis), e a busca por meio de força bruta, ou seja, todos os nós são visitados. É um algoritmo vantajoso no sentido de que requer espaço linear em relação à profundidade de busca [38], mas tem a desvantagem de se tornar inviável à medida que o tamanho da árvore aumenta (em número de voos e trajetórias).

Cada nó da árvore representa uma combinação de trajetórias, e a seleção da combinação ótima consiste na identificação do caminho, da raiz até um nó folha, cuja acumulação da função *score* indique uma pontuação ótima ou adequada.

Na Seção 7.1 é possível verificar que tal abordagem apresenta excelentes resultados em cenários menores, ou seja, menos aeronaves e/ou menos trajetórias por aeronave. É provável que o DFS atenda a maioria das necessidades atuais de tráfego aéreo, mas considerando a perspectiva de um aumento contínuo da demanda de tráfego aéreo [39], foi necessário desenvolver um segundo modelo que permitisse a identificação de trajetórias ótimas em cenários complexos.

5.4.2 Abordagem Alfa-Beta

Na abordagem inspirada no clássico algoritmo Alfa-Beta, a montagem e a busca em árvore tentam evitar que todos os nós possíveis sejam investigados. Esta característica é obtida com o cálculo, em simultâneo com a montagem da árvore, da pontuação acumulada até o nó de estudo, eliminando a necessidade de adicionar novos nós caso já seja identificado um custo inferior ao mínimo alcançado até ao momento.

Exemplificando, considerando uma situação hipotética de um cenário com 3 aeronaves em conflito e 3 trajetórias para cada aeronave, a árvore contará com 9 nós folha, ou seja, 9 combinações possíveis. Ao montar o primeiro caminho até o nó folha, registra-se o somatório da pontuação, que passa a impactar a montagem do restante da árvore conforme a seguinte estratégia: se ao longo da montagem da árvore, a pontuação para aquele caminho indica um resultado menos favorável do que o já registrado, a montagem desse caminho ou ramo é imediatamente interrompida (poda), passando-se para o próximo ramo da árvore, visto que foi encontrado um caminho pior do que o já conhecido. Se a

montagem de um caminho avança até o nó folha, e verifica-se um valor mais adequado do que o já registrado, o algoritmo substitui o valor anterior pela nova pontuação, e o procedimento continua até a investigação de todos os caminhos da árvore, completos ou não.

A eliminação de nós/caminhos conhecidos piores permite que o algoritmo reduza o espaço de busca, aumentando a capacidade de analisar árvores maiores considerando os mesmos recursos computacionais usados no DFS. Na seção Seção 7.1 será possível identificar essa redução no espaço de busca e, consequentemente, a análise de cenários bem mais complexos, provavelmente suficientes para a situação atual e futura do tráfego aéreo global.

5.5 Espaço de Busca

Um espaço de problema consiste em um conjunto de estados do problema e um conjunto de operadores que alteram o estado. Por exemplo, no jogo Eight Puzzle, os estados são as diferentes permutações possíveis das peças, e os operadores deslizam uma peça para a posição em branco. Uma instância de problema é um espaço de problema junto com um estado inicial e um estado objetivo. Um grafo de espaço de problema é freqüentemente usado para representar um espaço de problema. Os estados do espaço são representados por nodos do grafo, e as operações por arestas entre nodos. As arestas podem ser não direcionadas ou direcionadas, dependendo se seus operadores correspondentes são reversíveis ou não. A tarefa em um problema de localização de caminho de agente único é encontrar um caminho no grafo do nó inicial para um nó objetivo [38].

Armazenar dados sobre esses estados possíveis requer um enorme poder de computação. por exemplo, se considerarmos 5 aeronaves no mesmo cluster, com 40 trajetórias para cada aeronave (1 padrão e 39 alternativas), para identificar a solução ótima, será necessário avaliar 10^8 combinações possíveis entre cada uma das trajetórias de cada voo. O número de estados possíveis que podem ser analisados aumenta exponencialmente e também pode ser expresso pela Equação 5.3:

$$E(Q_a, Q_t) = Q_t^{Q_a} \tag{5.3}$$

onde, Q_a indica o número de aeronaves e Q_t o número de trajetórias por aeronave, padrão e alternativas. A Equação 5.3 pode ser usada para calcular o número de estados possíveis (espaço de busca). Para configurações complexas, é necessário usar tecnologia de computação para avaliar o maior número possível de situações. Também pode mostrar eficácia no ambiente de big data por meio do método sugerido. Nos testes realizados,

conforme será demonstrado no Capítulo 7, foram considerados cenários envolvendo até 10^{44} combinações possíveis.

Por fim, com relação à análise computacional das duas abordagens, tem-se que a complexidade do pior caso da abordagem Alfa-Beta equivale à complexidade da abordagem DFS visto que, nesta situação, não há nó a ser podado, de forma que a árvore será examinada integralmente, obtendo-se complexidade equivalente a $O(b^d)$, onde b indica a quantidade de filhos por nó (conhecido como *branch factor*), e d corresponde à profundidade da árvore. No melhor caso, em cada nó serão examinados $2b - 1$ filhos para a tomada de decisão, significando que, essenciamente, o algoritmo geral examina $O(b^{b/2})$ nós, o mesmo que um algoritmo de pior caso cujo ponto de corte é a metade de d, o que é bastante significativo.

Capítulo 6

Modelagem 2: Abordagem 4DNavMCTS

Esse Capítulo apresenta a proposta `4DNavMCTS`, que é a solução sugerida para navegação quadridimensional baseada em MCTS. Para simplificar o entendimento, abordaremos os conceitos de navegação 4D e os desafios a serem resolvidos, o método inspirado no algoritmo MCTS e a combinação desses diferentes conceitos para chegar à solução proposta.

6.1 Motivação para Nova Modelagem de Navegação 4D

Considerando o aumento da demanda por espaço aéreo e a necessidade de reduzir a carga de trabalho dos controladores de tráfego aéreo, os sistemas e procedimentos atualmente utilizados na gestão do tráfego aéreo precisam ser constantemente desenvolvidos. Atualmente, os métodos limitam-se a instruções dadas pelos controladores de tráfego aéreo à tripulação de voo, devendo esta cumprir rigorosamente as instruções recebidas e cumprir as restrições de desempenho da aeronave [82]. Embora o Controle de Tráfego Aéreo (ATC) tenha como objetivo garantir a separação entre as aeronaves, essa pesquisa foi realizada para otimizar os resultados das intervenções de trajetória [19].

A trajetória quadridimensional 4D é uma descrição precisa do trajeto percorrido por uma aeronave no espaço e no tempo [19]. Quando um controlador observa o tráfego na tela do radar, ele tenta identificar se há uma aeronave conflitante em um futuro próximo. Em caso afirmativo, emite instruções para manobrar e mantê-los separados. O problema é estimar a próxima posição da aeronave no horizonte dentro de 10 a 30 minutos. A previsão de trajetória 4D contém dados que especificam as posições horizontais e verticais previstas de uma aeronave durante um determinado período de tempo. Sob diferentes

condições de voo, a capacidade de prever com precisão as trajetórias de diferentes tipos de aeronaves é um fator importante na determinação da precisão e eficácia de um sistema ATM [10].

Em cenários de navegação 4D, a restrição de tempo também deve ser considerada para redefinir o conflito como a violação da distância mínima entre aeronaves dentro de uma determinada janela de tempo e um determinado segmento do espaço aéreo [88]. As Operações Baseadas em Trajetória (TBO) visam integrar a capacidade de navegação de uma aeronave no espaço e no tempo para melhorar a eficiência e previsibilidade do ATM [25]. Este objetivo requer principalmente: trajetórias de referência armazenadas; uma trajetória continuamente recalculada; indicadores eletrônicos de status; e um sistema de controle para acompanhar a trajetória geral no espaço e no tempo [19].

Tabela 6.1: Combinações de estados possíveis.

Número de	**Número de Estados por Nível**			
Voos	**Nível 1**	**Nível 2**	**Nível 3**	**Nível 20**
3	13	169	2,197	10^{22}
5	21	441	9,261	10^{26}
7	29	841	24,389	10^{29}
9	37	1,369	50,653	10^{31}

Armazenar dados sobre essas variáveis e seus possíveis estados requer um enorme poder computacional. Por exemplo, considerando que velocidade e altitude são as únicas variáveis modificáveis, cada aeronave pode escolher um dos cinco estados possíveis a cada momento. Um é o estado normal (seguindo o plano de voo pré-definido), o outro é o aumento e diminuição da velocidade/altitude. Conforme detalhado na Tabela 6.1, que apresenta valores para quantidades crescentes de voos, o número de estados possíveis que podem ser analisados aumenta exponencialmente e também pode ser expresso pela Equação 6.1:

$$E(Q_a, Q_v, Q_n) = (1 + Q_a.Q_v)^{Q_n} \tag{6.1}$$

em que, Q_a indica o número de aeronaves, Q_v o número de variações para cada voo em cada novo estado e Q_n o número de níveis projetados. A Equação 3.5 pode ser usada para calcular o número de estados possíveis (espaço de busca). Para configurações complexas, é necessário usar tecnologia de computação para avaliar o maior número possível de situações. Também pode-se mostrar eficácia no ambiente de *big data* por meio do método sugerido.

6.2 Modelagem MCTS para TBO

O algoritmo básico do MCTS, detalhado na Seção 3.2.4, inclui a construção iterativa de uma árvore de busca até que um determinado custo computacional predeterminado (geralmente tempo, memória ou limite de iteração) seja alcançado. Nesse ponto, a pesquisa será interrompida e retornará a operação de raiz com melhor desempenho. Cada nó na árvore de busca representa um estado de domínio e os links para os nós filhos representam a operação que leva aos estados subsequentes [3].

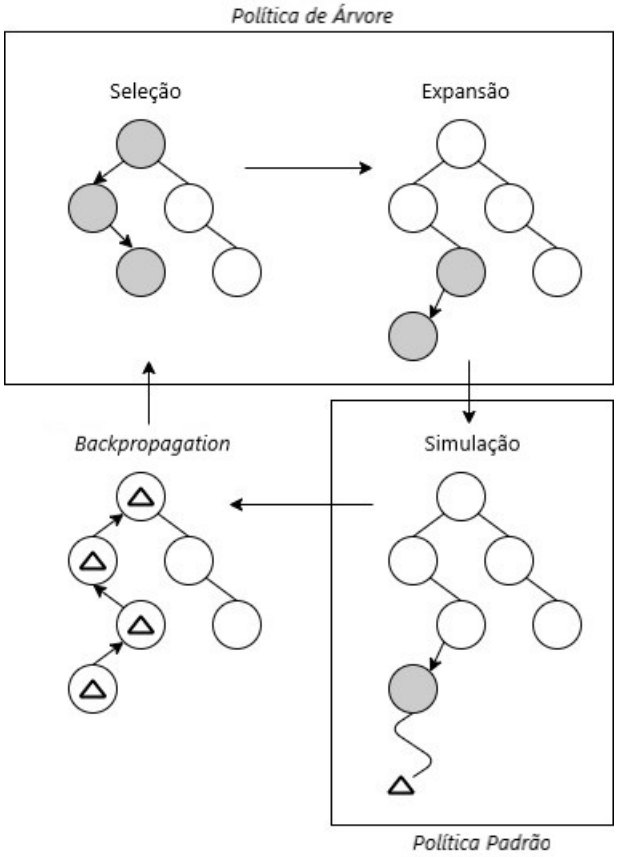

Figura 6.1: Representação de iteração do MCTS baseada em [3].

A modelagem da aplicação do conceito TBO na navegação 4D requer a manipulação das posições atuais e planejadas da aeronave em um espaço 4-dimensional. O monitoramento preciso requer que esta operação seja realizada com o máximo de informações de trajetória para reduzir quaisquer incertezas. Considerando um recorte do espaço aéreo de um dia de operação, que inclui apenas os voos com origem e destino no Brasil, o número de pontos 4D descrevendo as trajetórias das aeronaves ultrapassa facilmente 2 milhões de registros. Neste caso, não são consideradas as simulações para trajetórias de mudança, ou seja, são consideradas apenas trajetórias padrão, sendo necessário fazer a modelagem que possa representar, avaliar e atualizar esta grande quantidade de dados. Em termos

de computação, geralmente é necessário reduzir o espaço de busca para lidar com uma quantidade tão grande de informações, e é essa característica do MCTS que inspirou a modelagem abaixo.

Um dos problemas das abordagens atuais é que, ao tentar solucionar um conflito existente, a trajetória precisa ser recalculada caso seja detectado um novo conflito. Quando um conflito é resolvido, o modelo sugere uma determinada estratégia que poderá levar a um conflito futuro que não existia antes. Isso requer uma nova aplicação do modelo para resolver esse novo conflito. Como forma de resolver este problema, a modelagem aqui apresentada não considera apenas as trajetórias previstas de cada aeronave, mas também considera um grupo de trajetórias alternativas. Todo o grupo de trajetórias é calculado simultaneamente, o que permite identificar um conflito entre as trajetórias previstas. É possível buscar imediatamente uma solução para combinar as trajetórias previstas e as trajetórias alternativas para garantir que a decisão de realizar uma manobra (a escolha de uma trajetória alternativa) não gere um novo conflito porque as trajetórias alternativas são sempre calculadas.

Como método de representação de trajetórias e simulação de mudanças de rota, para prever possíveis interseções e/ou conflitos neste trabalho, serão construídas árvores MCTS contínuas. Cada camada da árvore representa as posições de todas as aeronaves em um instante t, conforme mostrado na Figura 6.2. A transição de um nó pai para um nó filho indica o deslocamento da aeronave representada na árvore durante o intervalo de tempo Δt. Ou seja, os nós filhos representam as novas posições de cada aeronave após o deslocamento. Cada nó filho recém-adicionado tem uma combinação diferente de trajetórias. Por exemplo, o primeiro nó filho pode indicar o deslocamento padrão no tempo de todas as aeronaves; O segundo nó pode indicar o deslocamento da aeronave, levando em consideração qualquer alteração de rota de qualquer voo da combinação (por exemplo, velocidade); o terceiro nó apresenta uma terceira combinação diferente, com uma mudança distinta na mesma trajetória que foi alterada no segundo nó, ou uma mudança em outras trajetórias de voo, e assim por diante.

No exemplo mostrado na Figura 6.2, a posição relativa de duas aeronaves é mostrada em cada estado. O estado s_0 indica a posição atual da aeronave, considerando que o momento atual é t_0. O próximo nível da árvore, t_1, indica a posição de cada aeronave após um intervalo de tempo Δt, com três estados possíveis para aquele instante. Considerando apenas as mudanças no parâmetro velocidade: s_1 mostra que não há mudança na posição da nova aeronave; s_2 representa um aumento da velocidade da aeronave superior em relação à inferior; e; s_3 significa que a velocidade da aeronave superior diminui (ou a velocidade da aeronave inferior aumenta). Essas mesmas mudanças de estado das aeronaves podem ser replicadas em outros níveis da árvore, resultando em um estado em

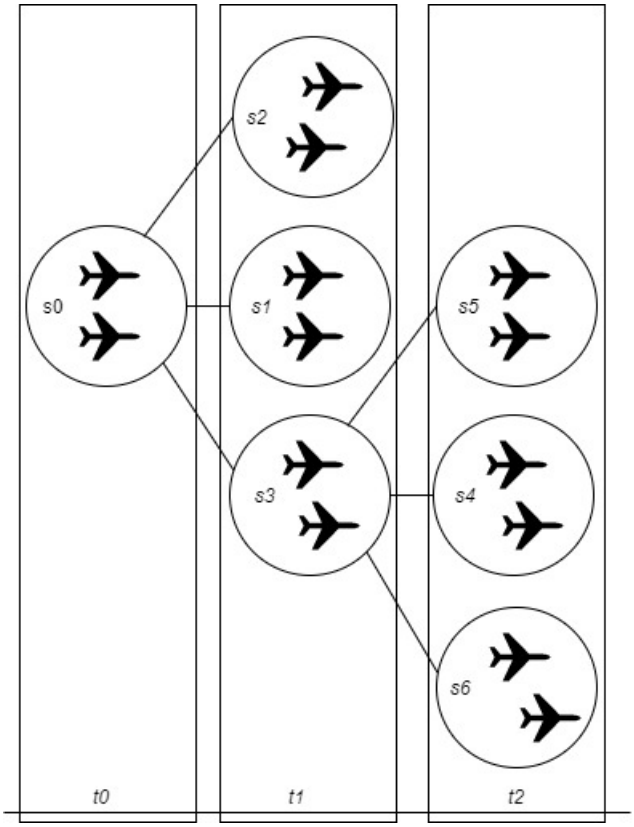

Figura 6.2: Representação de trajetórias na árvore MCTS.

que as distâncias entre as aeronaves aumentam e diminuem com o crescimento da árvore.

Outra particularidade da modelagem `4DNavMCTS` é o fato de ser um *"single player game"*, ou seja, o objetivo aqui não é maximizar as chances de vitória dos jogadores, mas minimizar as chances de vitória dos adversários.

No modelo TBO proposto, o objetivo é garantir que a montagem na árvore permita manter as condições ideais de voo em cada nível. Em seguida, buscar uma condição de jogo subótima, ou seja, o estado que representa a operação ideal para o controle de tráfego aéreo. Podemos definir a operação desejável da seguinte maneira:

- priorização de trajetórias padrão: há a menor interferência possível no espaço aéreo, ou seja, desde que não haja conflito, as aeronaves podem seguir suas trajetórias esperadas sem nenhuma interferência;
- evite intervir em vários voos: se forem necessárias intervenções contínuas, priorize a estratégia que afete o menor número possível de voos, pois é melhor alterar um voo várias vezes, em vez de vários voos;
- assegurar a distância mínima: quando as aeronaves seguem suas trajetórias padrão, ou intervêm para eliminar conflitos futuros, é necessário manter a separação mínima;

- Avaliar o espaço aéreo como um todo: significa considerar situações de conflitos simultâneos e/ou considerar outros voos envolvidos ou não no conflito potencial sob análise.

Nesse sentido, a modelagem `4DNavMCTS` procura garantir que a simulação possa alcançar a operação desejável ao máximo. Em suma, se nenhum conflito futuro for identificado durante a simulação, as respostas do `4DNavMCTS` não irão sugerir nenhuma intervenção, ou seja, as aeronaves poderão seguir suas rotas previamente definidas. No entanto, se algum conflito for identificado, a modelagem buscara garantir que o impacto seja o menor possível.

6.3 Avaliação Baseada em MEV

A avaliação das simulações adotadas pelo `4DNavMCTS` foi inspirada pela metodologia do Modelo de Espaço Vetorial (MEV) originalmente desenvolvida para sistemas de recuperação de informação. A ideia do MEV é representar cada documento do conjunto como um ponto no espaço multidimensional (um vetor em um espaço vetorial). Pontos adjacentes no espaço são semanticamente semelhantes, enquanto pontos distantes são semanticamente diferentes. As consultas também são expressas como um ponto no mesmo espaço (a consulta é do tipo pseudo-documento). De acordo com a distância da consulta, os documentos são classificados em ordem crescente [89].

Na abordagem MEV tradicional, a similaridade entre vetores-documentos é calculada pela montagem de uma matriz, geralmente de frequências. Cada linha representa uma entidade (documento) e cada coluna representa uma palavra (termo). Existem diferentes métodos para calcular a similaridade entre dois vetores. Uma forma muito comum de recuperação de informação é considerar o espaço composto por vetores unitários, sendo a similaridade dada pelo cosseno do ângulo formado entre dois vetores.

Considerando espaços vetoriais formados por vetores não unitários, uma forma de avaliar as relações é calcular a distância entre os pontos representados por cada vetor. Nesse caso, uma forma de obter essa medida é utilizando a distância euclidiana, que consiste na aplicação repetida do teorema de Pitágoras [90].

Sejam x e y dois vetores, cada um contendo n elementos:

$$\begin{aligned} x &= \langle x_1, x_2, ..., x_n \rangle \\ y &= \langle y_1, y_2, ..., y_n \rangle \end{aligned} \tag{6.2}$$

A distância euclidiana entre dois pontos, $d(x, y)$ representada por esses vetores pode ser calculada da seguinte forma:

$$\begin{aligned} d(x,y) &= \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2} \\ &= \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \end{aligned} \tag{6.3}$$

Essa análise permite identificar quando dois vetores são menos semelhantes (com distâncias maiores) ou mais semelhantes (com distâncias menores).

Na modelagem para TBO, buscou-se implementar uma função de avaliação que representasse a operação desejável descrita na Seção 6.2. A modelagem inclui a representação de cada combinação de estados ou caminhos da árvore MCTS como vetores em um espaço vetorial, e o eixo do espaço vetorial refere-se a fatores que contribuem para uma operação padrão. Nessa representação, foram escolhidas as seguintes variáveis: F_s, representa a frequência da trajetória padrão, ou seja, é o número total de caminhos para cada nó atingi-la de acordo com seu plano de voo inicial; F_i representa a frequência de intervenções, ou seja, é o número total de aeronaves em cada nó que sofreram intervenção/alteração em sua trajetória até alcançá-lo; e F_v, que indica em quantos níveis na representação da árvore houve estados onde os critérios de separação foram violados.

$$V_i = \langle \alpha.F_s, \beta.F_i, \gamma.F_v \rangle \tag{6.4}$$

A definição adotada para o vetor de estratégia detalhado acima pode ser expressa pela função de avaliação descrita na Equação 6.4, onde V_i refere-se ao nó ith e as variáveis α, β e γ atribuem pesos a cada um dos eixos e são definidas empiricamente, através de testes sucessivos.

A operação desejável descrita na Seção 6.2, e modelada como um MEV, foi definida com o auxílio de especialistas em controle de tráfego aéreo. Essa abordagem foi escolhida em virtude da característica on-line da solução, ou seja, que os conflitos sejam identificados e resolvidos em tempo real, considerando os valores instantâneos de todas as variáveis envolvidas, independentemente do histórico dessas mesmas trajetórias. Com isso, o aprendizado a partir de bases históricas de trajetórias passou a não ter representatividade para a solução, justificando o uso do MEV como metodologia para a seleção das estratégias promissoras, alternativamente às metodologias baseadas em aprendizagem.

6.4 Implementação do `4DNavMCTS`

Esta seção descreve a implementação de `4DNavMCTS` incluindo o desenho do banco de dados, gerador de trajetória, arquitetura de solução e algoritmos relacionados.

6.4.1 Projeto de Banco de Dados

Para implementar TBO, é importante integrar a capacidade de navegação de uma aeronave no espaço e no tempo para melhorar a eficiência e previsibilidade do ATM [25]. Esta visualização requer quatro partes principais para garantir as trajetórias de referência armazenadas: trajetórias de referência armazenadas; uma trajetória de captura continuamente recalculada; mostradores eletrônicos de situação; e um sistema de controle para rastrear a trajetória geral no espaço e no tempo. Todos esses componentes precisam dos bancos de dados para organizar melhor uma grande quantidade de informações.

Alguma estrutura de banco de dados deve ser projetada para lidar com as trajetórias das aeronaves, e quaisquer operações finais devem ser realizadas de acordo com os dados históricos da trajetória. As bases de dados NoSQL (Not Only SQL) constituem uma abordagem desenvolvida para gerir *big data* a distribuir entre vários nós numa rede, adequada para esta tarefa [32].

Algumas abordagens de bancos de dados relacionais foram inicialmente avaliadas, porém o requisito principal de ser capaz de processar grandes volumes de dados se mostrou um obstáculo à continuidade da adoção dessas metodologias tradicionais. Diante disso, a avaliação de tecnologias NoSQL balisou a continuidade do estudo para implementação da estrutura de dados necessária para a modelagem definidal, com uma vantagem adicional de lidar mais facilmente com fontes de dados flexíveis, ou seja, tornar a absorção de especificidades menos trabalhosa em termos de arquitetura de banco de dados.

O MongoDB, apresentado na Seção 3.1.2 é um banco de dados NoSQL orientado a documentos e sem esquema, no qual os documentos são agrupados em coleções. Os documentos são armazenados no formato BSON (JSON[1] binário). Para melhorar o desempenho, os arquivos de dados são mapeados na memória. Por padrão, os dados são enviados para o disco a cada 60 segundos. Quando um novo arquivo é criado, tudo é gravado no disco, o que libera a memória. O MongoDB usa índices em coleções de maneira semelhante aos bancos de dados relacionais e oferece suporte a uma tecnologia chamada map-reduce para agregações complexas entre documentos para melhorar o desempenho [33].

MongoDB foi selecionado com base no desempenho comparado a outros bancos de dados relacionais e bancos de dados NoSQL. Muitos estudos mostram que o MongoDB tem

[1] https://www.json.org/

melhor desempenho do que bancos de dados SQL e outros tipos de bancos de dados NoSQL [33]. Comparado com outros bancos de dados otimizados para *big data*, o MongoDB pode obter o melhor desempenho nas operações de leitura de acordo com o tamanho do conjunto e a modelagem adotada [87].

O banco de dados MongoDB (coleções) foi modelado conforme descrição contida na Seção 5.2 para conseguir obter um desempenho satisfatório ao ler grandes quantidades de dados. Nesse sentido, os dados da trajetória a serem processados são armazenados em uma coleção, e cada ponto/registro dessa coleção é convertido em um documento JSON.

6.4.2 Gerador de Trajetórias

A solução `4DNavMCTS` foi projetada para processar dados reais e, diante dessa preocupação, foram realizados testes iniciais com informações do Automatic Dependent Surveillance - Broadcast (ADSB). Esta é uma tecnologia de vigilância em crescimento, que permite aos controladores de tráfego aéreo rastrear aeronaves e veículos terrestres em aeroportos sem radar (aviões e outros veículos transmitem sua posição e outros parâmetros). O sistema ADSB é composto por aviônicos de aeronaves e infraestrutura terrestre, o que possibilita a substituição do radar na vigilância do tráfego aéreo [10].

Os fatos provaram que é bastante desafiador focar no espaço aéreo brasileiro e fazer uso das informações do ADSB. Em algumas áreas, verificou-se que as informações obtidas apresentam muito ruído, o que pode ser devido a vários motivos: falha instantânea de comunicação (transmissão de dados ADSB); nenhuma cobertura ou cobertura insuficiente; entre outros.

Na análise preliminar das trajetórias obtidas a partir dos dados do ADSB, constatou-se que em alguns períodos e regiões, apenas cerca de 30% dos voos fornecem informações completas em toda a trajetória, o que limita a avaliação da capacidade de sobrecarga do `4DNavMCTS`. Para resolver este problema, um gerador de trajetória aleatória foi desenvolvido para processar os seguintes parâmetros de entrada:

- timestamp refere-se à janela inicial, ou seja, o indicador de tempo do primeiro ponto da primeira trajetória de simulação;
- o perímetro é representado por 4 coordenadas geográficas, formando um polígono no espaço aéreo, contendo os limites inicial e final de cada trajetória;
- número de aeronaves indica quantas trajetórias simular;
- os limites de velocidade são usados para selecionar diferentes velocidades de cruzeiro para cada trajetória dentro de uma faixa razoável;

- a altitude inicial e o nível de voo da aeronave são baseados no algoritmo proposto. Com o aumento do número de aeronaves em uma área, as aeronaves são distribuídas em diferentes altitudes de voo, melhorando assim a capacidade do espaço aéreo;
- as janelas neutras correspondem ao número de níveis da árvore MCTS, e são utilizadas como áreas neutras no início e no final da simulação, ou seja, os conflitos ocorridos nestes intervalos de tempo serão ignorados; Deve-se ressaltar que a última variável é incluída para eliminar falsos negativos de `4DNavMCTS`, pois os conflitos iniciais e finais na fase de cruzeiro são difíceis de resolver (não há margem de manobra). Na operação real, eles são tratados de outras maneiras e além da faixa inicial da modelagem.

A direção de geração de cada trajetória também é definida aleatoriamente (norte-sul, sul-norte, oeste-leste ou leste-oeste) de forma que o voo comece em uma ponta do perímetro e termine na outra ponta, favorecendo a existência de potenciais conflitos.

Embora as trajetórias simuladas sejam usadas, o modelo atende totalmente aos requisitos do ADSB e é universal o suficiente para ser facilmente transplantado para outra fonte de dados real, mantendo assim sua capacidade online para TBO.

6.4.3 Arquitetura da Solução

O presente modelo consiste na arquitetura ilustrada na Figura 6.3, cujos componentes serão detalhados ao longo desta seção. Antes da execução do algoritmo base do `4DNavMCTS`, há uma etapa para definir os parâmetros iniciais do modelo (etapa zero).

O primeiro passo para a operação completa do `4DNavMCTS` inclui a obtenção de dados de trajetória, que podem ser captados a partir de informações ADSB em tempo real, planos de voo previamente definidos e suas projeções completas de trajetória 4D. Esta previsão pode ser obtida a partir de ferramentas como o preditor de trajetória (TP) [91], ou mesmo para fins de simulação, podem ser utilizados os dados históricos das trajetórias das aeronaves (por exemplo, ADSB). Neste trabalho, foram utilizadas trajetórias do gerador de trajetória aleatória descrito na subseção anterior. Além disso, modelos matemáticos, como as fórmulas de Vincent [92], podem ser usados para calcular o deslocamento da aeronave de acordo com determinados parâmetros de voo (posição, proa, etc.).

Depois de preparar a base (armazenada no MongoDB), o núcleo `4DNavMCTS` baseado no algoritmo MCTS agora pode ser executado. É necessário definir os seguintes parâmetros para sua implementação:

- critério de parada (sc): é composto pela duração (em segundos) da execução de cada algoritmo MCTS;

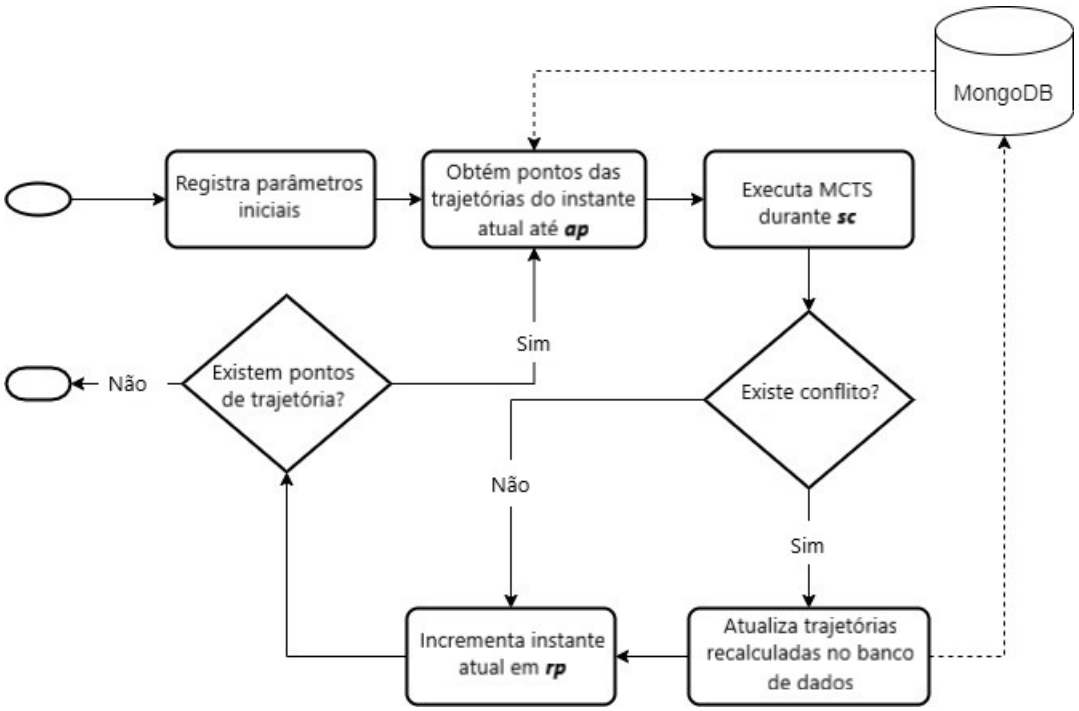

Figura 6.3: Arquitetura da solução.

- período de repetição (rp): é composto pelo intervalo de tempo entre o início da execução de cada algoritmo MCTS;
- período de antecipação (ap): Consiste na janela de tempo futuro, que passará a fazer parte do intervalo de simulação.

Por exemplo, quando definimos um critério de parada de 15 segundos, um período de repetição de 30 segundos e um período de antecipação de 10 minutos, significa que o algoritmo é executado a cada 30 segundos, ou seja, roda por 15 segundos em 0, 30s, 60s, etc. Para que a próxima execução não comece sem que a anterior termine, é fundamental que o rp seja maior que o sc. Durante a execução do algoritmo, a árvore será construída para levar em conta os pontos futuros até o limite de ap. Ou seja, para o exemplo dado, a simulação da trajetória será realizada pelos próximos 10 minutos. Esta é uma limitação da profundidade das árvores e deve fazer sentido no cenário real. Ou seja, deve considerar o maior número possível de níveis, de forma a poder modificar a trajetória no tempo, evitando assim conflitos. Este exemplo é ilustrado na Figura 6.4.

Uma vez definidos os parâmetros iniciais, a execução iterativa do algoritmo é realizada da seguinte forma. De acordo com rp, o algoritmo será executado repetidamente quantas vezes forem necessárias. A cada rodada, o algoritmo MCTS será executado até que o critério de parada seja atingido, durante o qual a árvore de busca ficará limitada ao período de antecipação. Cada nível da árvore representará o deslocamento da aeronave no tempo,

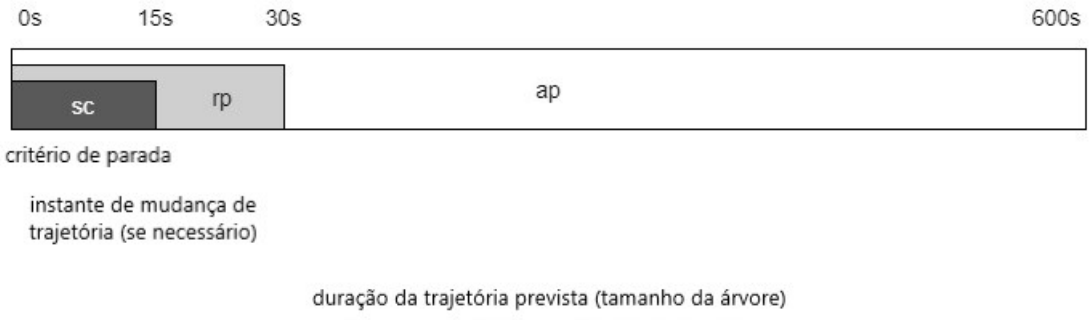

Figura 6.4: Parâmetros iniciais do modelo.

que equivale a rp, e os nós representarão as novas posições de todas as aeronaves. Cada nó armazenará as posições de todas as aeronaves naquele momento, e os nós no mesmo nível mostrarão as diferenças específicas de posições específicas de aeronaves e representarão com precisão quaisquer mudanças de trajetória. Essas diferenças serão geradas pelo algoritmo na etapa de simulação de acordo com requisitos previamente definidos, podendo considerar: mudanças de velocidade; mudanças/deslocamentos de trajetória; entre outros.

Algoritmo 1: Implementação do `4DNavMCTS`

Dados: conjunto de trajetórias 4D
Resultado: conjunto modificado de trajetórias 4D
$sc \leftarrow criterioParadaMCTS$;
$rp \leftarrow periodoRepeticaoMCTS$;
$ap \leftarrow periodoAntecipacaoMCTS$;
$BD \leftarrow baseTrajetorias4D$;
for *cada intervalo em rp* **do**
  $Traj \leftarrow BD.getTrajetorias(rp, ap)$;
  $Arvore \leftarrow Traj$;
  for *cada intervalo em sc* **do**
    $UCT\_Selecao(Tree)$;
    $Expansao(Tree)$;
    $Simulacao(Tree)$;
    $MEV\_Retropropagacao(Arvore)$;
  end
  $BD \leftarrow Traj.melhorNo()$;
end
return BD modificado

Ao final de cada execução, caso algum conflito seja identificado, o algoritmo proporá alterações nas rotas das aeronaves envolvidas a partir daquele ponto, atualizando os pontos de trajetória armazenados no banco de dados (MongoDB). Uma interface pode ser criada para permitir que os controladores de tráfego aéreo decidam se aceitam as mudanças propostas. Esses novos pontos serão levados em consideração na implementação subsequente

do algoritmo. Se nenhum conflito for identificado, o algoritmo não fará nenhuma alteração no banco de dados, que o preparará para a próxima execução.

O Algoritmo 1 resume o processo de implementação descrito acima.

Capítulo 7

Casos de Testes e Experimentos

Neste Capítulo são detalhados os casos de testes definidos para cada um das duas abordagens apresentadas, bem como as simulações que foram realizadas para avaliação da modelagem sugerida. Adicionalmente, são apresentadas as análises dos resultados obtidos, também para cada uma das abordagens.

7.1 Abordagem DFS e Alfa-Beta - Estudo de Caso

A seguir serão apresentados os testes realizados com a abordagem baseada nos algoritmos de busca Alfa-Beta e Busca em Profundidade (DFS), com a descrição das simulações realizadas e dos resultados obtidos.

7.1.1 Descrição das Simulações

Duas bases de dados diferentes foram utilizadas para analisar o framework proposto no Capítulo 5: uma baseada em dados reais, com foco no desempenho do componente de detecção de conflitos; e outro com dados gerados aleatoriamente, para avaliação do componente de resolução de conflitos. Em ambos os casos, são adotados os seguintes atributos principais: coordenadas de posição (altitude, latitude e longitude); instante de tempo (quarta dimensão do ponto de trajetória); e a identificação do voo (*callsign*). No caso da base de dados contendo informações reais, dados adicionais estão disponíveis, tais como ângulo de ataque e altitude barométrica, por exemplo.

O cenário de simulação para detecção de conflitos consta detalhado no trabalho proposto por Vitor [19], e é composto por uma seleção de nove aeroportos no Brasil. Optou-se por utilizar informações do tráfego aéreo local, apesar do Brasil estar na vanguarda mundial em termos de segurança operacional na aviação civil [93], pois o crescimento da demanda de tráfego aéreo exige o desenvolvimento contínuo de novos métodos e modos

de operação, o que pode garantir a segurança do transporte aéreo mesmo quando a escala aumenta. O país é o segundo do mundo em número de aeroportos, e os selecionados atendem a maior parte da população brasileira. Juntos, eles foram responsáveis por 58% do tráfego aéreo nacional em 2018 [94]. Portanto, são uma representação adequada do cenário do tráfego aéreo brasileiro para esta simulação.

A base de dados de voos repetitivos foi coletada do site gratuito do Centro de Gerenciamento de Navegação Aérea (CGNA), que contém uma lista de planos de voos com seus respectivos prazos de validade. Dentre os voos válidos, foram selecionados aqueles com validade até junho de 2019. Essa seleção não considerou a frequência e periodicidade desses voos em geral. Dentre os planos selecionados, 1.221 voos objeto são filtrados para simulação, desde que estes voos tenham aeroportos de origem e destino no conjunto de aeroportos selecionado. Para realização dos testes, duas abordagens diferentes podem ser adotadas para a inserção de conflitos em uma base real: alterar o plano de voo, de maneira que haja coincidência de partes da trajetória com outro voo; e/ou aumentar a distância mínima de separação entre as aeronaves, de forma que um conflito não existente no mundo real possa ser considerado um conflito para fins de teste.

Uma vez que o banco de dados esteja totalmente preenchido, todos os Planos de Voo Repetidos (RPL) são buscados e enviados ao motor que gera os arquivos de entrada para o preditor de trajetórias (TP). Em seguida, um script *batch* desenvolvido para executar o TP, passando todos os arquivos/parâmetros de entrada necessários, é executado para obtenção da previsão de trajetória para cada voo. O aplicativo cliente acessa as trajetórias de saída e insere cada *waypoint* amostrado no banco de dados. Após executar o módulo de previsão de trajetória, foi possível ter uma visão abrangente da ocupação do cenário em estudo.

Em um segundo cenário de simulação, os planos de voo foram replicados com níveis de voo atualizados e respectivas velocidades para criar diferentes estratégias, equivalentes a trajetórias alternativas. No total, as distintas combinações de parâmetros resultaram em cinco diferentes trajetórias alternativas passíveis de serem selecionadas para a aeronave envolvida na tomada de decisão.

Cenários simulados com dados gerados aleatoriamente foram utilizados para avaliar a proposta de resolução de conflitos, permitindo maior controle e complexidade dos testes, pois nas execuções com dados reais, os *clusters* de conflitos identificados foram compostos, em sua maioria, por poucas combinações de aeronaves. Dessa forma, foram considerados aglomerados fictícios com poucas a dezenas de aeronaves em conflito, e um número de trajetórias da ordem de até centenas de casos possíveis. Para estes cenários, fatores incertos como ruído e condições climáticas não foram considerados, porém o modelo pode se adaptar a restrições diversas (obstáculos virtuais ou perturbações aleatórias). É impor-

tante ressaltar que tais fatores incertos estão naturalmente presentes nos testes realizados com dados reais.

Os testes foram realizados considerando os mesmos parâmetros de entrada, tanto para a abordagem DFS quanto para a abordagem Alfa-Beta. Cenários com diferentes números de aeronaves no mesmo *cluster* foram considerados (3, 5, 7, 9, 12, 15, 18, 20, 25 e 30 aeronaves), e, para cada cenário, diferentes quantidades de trajetórias consideradas para cada aeronave, ou seja, a trajetória padrão e outras tantas trajetórias alternativas (3, 7, 10, 15, 20, 25, 30, 40 e 50).

Uma vez que na análise do método de resolução de conflitos foram utilizados dados simulados, optou-se por definir o tempo de processamento como o critério de parada, ou seja, o número de voos e de trajetórias por voo aumenta até um determinado limite de tempo de execução, com foco em cenários que pudessem ser aplicados em condições reais (viabilidade). Portanto, foram considerados apenas os resultados do algoritmo de resolução de conflitos cuja duração não ultrapassou o limiar de $120s$ (segundos).

Adicionalmente, para cada combinação de número de aeronaves e número de trajetórias por aeronave, foram realizados testes sucessivos (30 repetições) a fim de garantir que os resultados obtidos sejam estatisticamente sustentáveis, totalizando 2.700 execuções para cada caso.

7.1.2 Análise dos Resultados

Nesta subseção serão apresentados os resultados dos testes realizados com o framework proposto, focando principalmente no desempenho dos modelos de CDR apresentados no Capítulo 5, considerando o grande volume de dados de trajetórias processadas.

Detecção de Conflitos e Avaliação de Desempenho

Conforme demonstrado por Vitor [19], a trajetória de voo de cada aeronave é representada como uma sequência de *waypoints* amostrados por meio do TP. Um total de 160.022 *waypoints* foram computados pelo TP, formando uma base composta por dados relativos a um dia inteiro de operação.

Após a previsão da trajetória, o procedimento de detecção de conflitos é acionado, e os conflitos encontrados também são inseridos no banco de dados. A base de dados de *waypoints* é então consultada por janela de tempo, e são retornados os pontos que ocupam o espaço aéreo em determinado momento. Para fins de simulação, foram realizados três procedimentos de detecção de conflitos: (i) *conflitos padrão* os conflitos encontrados naturalmente se os planos de voo forem executados conforme definido originalmente pelo RPL; (ii) *conflitos alternativos*, os conflitos encontrados entre todas as trajetórias origi-

nais mais as trajetórias alternativas; e (iii) *conflitos de janela de tempo*, os conflitos são avaliados dentro de um tempo específico de antecipação.

As simulações foram realizadas com as seguintes configurações: uma Workstation HP Z220CMT BR equipada com uma CPU Intel® Xeon® E3-1270 V2 3.50GHz, 32GB DDR3 SDRAM 800MHz e Microsoft Windows 7 Professional 64 bits como sistema operacional. Essa situação permite que o processo de detecção de conflitos seja executado em paralelo e em sequência, e a alocação de threads seja gerenciada programaticamente. A Tabela 7.1 apresenta o desempenho da detecção de conflitos usando apenas as trajetórias padrão. A execução foi realizada usando até 2^n threads, onde $0 \leq n \leq 5$. Da mesma forma, a abordagem estratégica também foi avaliada. A Tabela 7.2 mostra o desempenho da detecção de conflitos usando a combinação de todas as trajetórias possíveis de serem executadas pela aeronave, com diferentes combinações de execuções do plano de voo.

Tabela 7.1: Desempenho de detecção de conflitos entre trajetórias originais. O número de conflitos é sempre o mesmo porque os testes foram executados com o mesmo conjunto de dados.

			Cassandra	**MongoDB**
Threads	Waypoints	Conflitos	Duração (s)	Duração (S)
1	160.022	330	0,936	0,518
2	160.022	330	0,516	0,269
4	160.022	330	0,331	0,2
8	160.022	330	0,286	0,159
16	160.022	330	0,298	0,19
32	160.022	330	0,25	0,207

Tabela 7.2: Atuação na detecção de conflitos entre todas as trajetórias estratégicas. Esses números mostram que os resultados são consistentes aumentando o número de threads para acelerar a execução do algoritmo, e o tempo de execução continua diminuindo quando 8 threads são adicionados.

			Cassandra	**MongoDB**
Threads	Waypoints	Conflitos	Duração (s)	Duração (S)
1	833.072	8.466	1m12,025s	1m20,319s
2	833.072	8.466	0m38,126s	0m42,293s
4	833.072	8.466	0m23,075s	0m24,802s
8	833.072	8.466	0m15,798s	0m19,169s
16	833.072	8.466	0m15,498s	0m18,605s
32	833.072	8.466	0m16,216s	0m17,042s

A Figura 7.1 mostra um exemplo de conflito entre dois voos. A trajetória representada por um caminho laranja pertence a um voo partindo do Galeão para Congonhas,

enquanto a trajetória representada por uma linha azul é realizada por um voo partindo de Santos Dumont para Congonhas. Embora seus distintos níveis de cruzeiro (FL340 e FL300 respectivamente) garantam a separação segura na maior parte do voo, a perda de separação é detectada quando ambas as aeronaves estão em procedimento de descida.

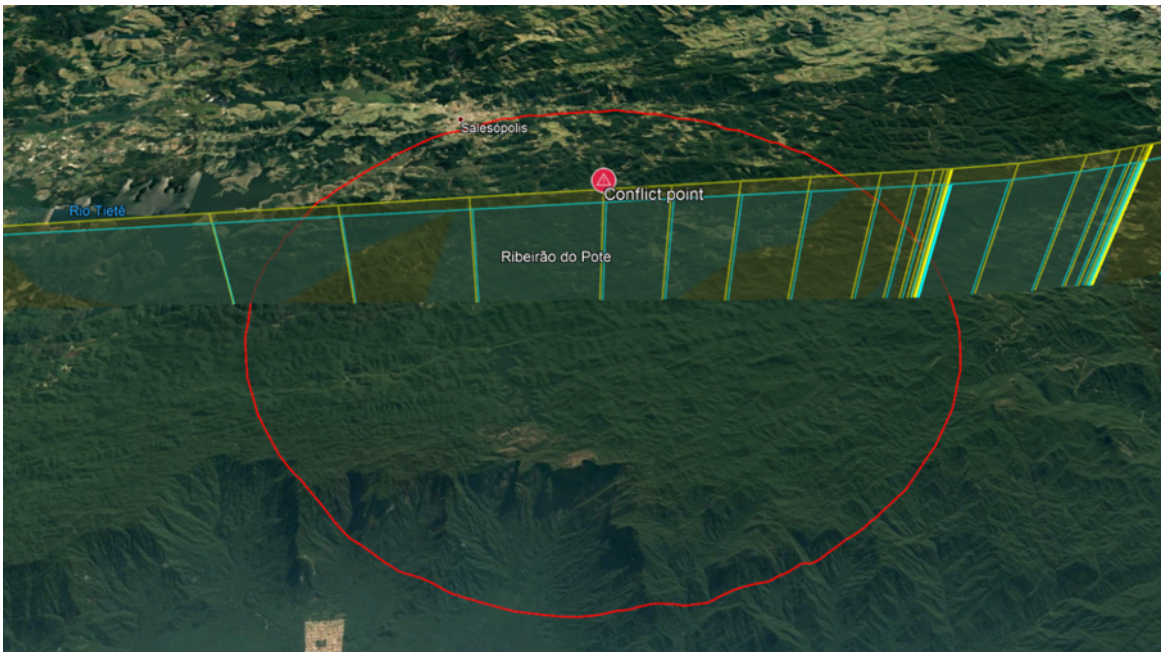

Figura 7.1: Conflito detectado entre duas aeronaves.

É fundamental esclarecer que o procedimento de previsão de conflito proposto compreende um dia inteiro de operações. Assim, cada voo programado é avaliado. Para cada voo, são fornecidas mais quatro estratégias, o que significa que cada aeronave possui cinco trajetórias alternativas diferentes a serem executadas em caso de conflito. Isso justifica o grande número de conflitos encontrados: primeiramente, os planos de voo padrão inserem 330 conflitos no cenário ao longo do dia. A nova metodologia de detecção de conflitos apresentada neste trabalho também leva em consideração os prováveis conflitos que devem aparecer em todas as combinações possíveis de trajetórias alternativas pré-selecionadas totalizando 6.105 trajetórias e 833.072 *waypoints*.

Embora esse procedimento complete a previsão do conflito estratégico em um dia inteiro, uma avaliação adicional foi realizada para janelas de tempo sequenciais compreendendo a avaliação de uma hora a partir do momento atual. Esta avaliação é importante para demonstrar a eficiência do algoritmo em um cenário dinâmico onde os conflitos devem ser detectados sob demanda, tipicamente em uma operação tática de curto prazo, a ser implementada como parte de um trabalho futuro. As Tabelas 7.3 e 7.4 mostram o desempenho obtido para detecção de conflitos para trajetórias padrão e trajetórias alter-

nativas durante o período ímpar. Ambos os cenários foram executados em paralelo com 16 *threads*.

Tabela 7.3: Detecção de conflito de janela de tempo para trajetórias padrão.

			Cassandra	**MongoDB**
Hora	Waypoints	Conflitos	Duração (s)	Duração (s)
1:00	4.764	2	0,012	0,007
3:00	1.018	0	0,002	0,002
5:00	0	0	0,002	0,002
7:00	405	0	0,002	0,001
9:00	6.086	8	0,015	0,016
11:00	10.220	22	0,013	0,012
13:00	11.135	30	0,021	0,008
15:00	9.530	24	0,012	0,01
17:00	8.613	14	0,017	0,024
19:00	9.112	16	0,017	0,008
21:00	10.692	29	0,02	0,011
23:00	9.320	22	0,033	0,046
Total	**160.022**	**330**	**0,372**	**0,278**

Tabela 7.4: Detecção de conflito de janela de tempo para trajetórias alternativas.

			Cassandra	**MongoDB**
Hora	Waypoints	Conflitos	Duração (s)	Duração (s)
1:00	24.316	96	0,296	0,228
3:00	5.133	11	0,096	0,074
5:00	0	0	0,002	0,001
7:00	2.158	0	0,002	0,002
9:00	32.484	179	0,282	0,307
11:00	53.166	563	1,505	1,421
13:00	58.029	711	1,277	1,3
15:00	50.101	673	0,952	0,929
17:00	44.495	401	0,496	0,521
19:00	47.306	410	0,484	0,52
21:00	55.789	768	0,626	0,544
23:00	48.065	550	0,39	0,297
Total	**833.072**	**8.466**	**13,772**	**13,951**

Verificou-se que o algoritmo para detecção de conflitos é bem mais eficiente do que os algoritmos encontrados no levantamento do estado da arte em relação ao tempo de execução. Na verdade, o método de banco de dados NoSQL elimina a necessidade de pré-processamento de dados, que é comumente encontrado no estado da arte, e visa possibilitar

o processamento de grandes volumes de dados, tais como indexação, padronização, etc., de forma que o conjunto de dados fornecido é considerado um espaço de pesquisa podado. Sobre adicionar *threads* para acelerar a execução, encontramos resultados consistentes. Como esperado, a Figura 7.2 mostra que o tempo de execução caiu consistentemente até 8 *threads* devido ao recurso de virtualização do núcleo da CPU.

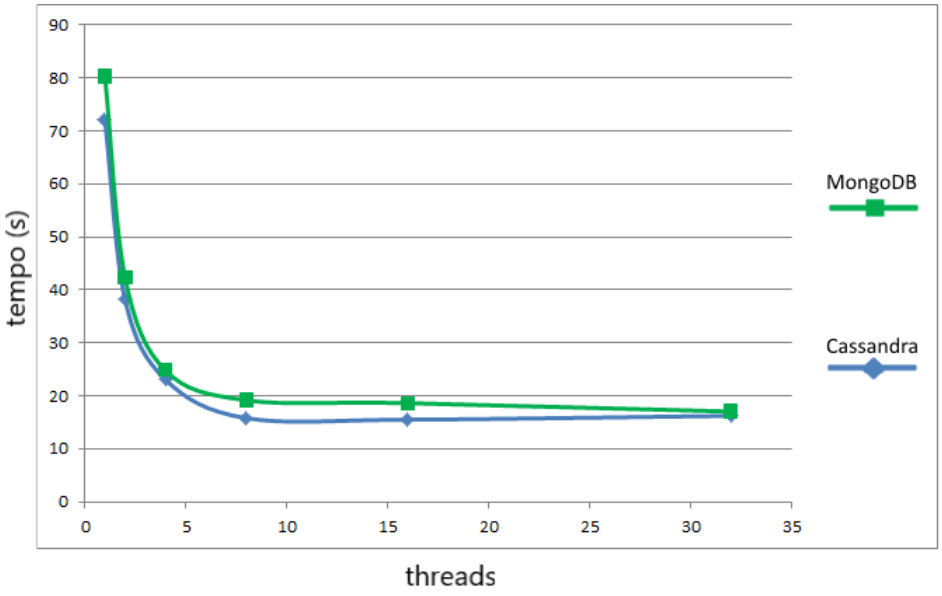

Figura 7.2: Comparação da performance com diferentes configurações de *thread*.

Desempenho da Resolução de Conflitos

A Tabela 7.5 mostra que o algoritmo DFS pode encontrar a solução ótima dentro do limite de tempo predeterminado para até 10^8 combinações possíveis de espaços de busca. Essa limitação faz com que com o aumento do número de voos, o número de trajetórias diminua, o que é um comportamento previsível. Em outras palavras, se o modelo considerar até três trajetórias (uma trajetória padrão e duas trajetórias alternativas) para cada aeronave em um *clusters* contendo 18 aeronaves em situação de conflito, é possível encontrar a melhor combinação de trajetórias dentro do limite de tempo, ou seja, $1m46s$ em média, verificando 100% do espaço de busca, 10^8 estados possíveis. No outro extremo, considerando *clusters* com três aeronaves conflitantes, o algoritmo consegue encontrar a solução ótima considerando 50 trajetórias para cada aeronave em apenas $22,3ms$, verificando 100% do total de 10^5 estados possíveis.

Esses números mostram que o modelo se torna viável em cenários com um número menor de aeronaves, isto é, em situações de menor complexidade, o que pode ser suficiente para grande parte do transporte aéreo. Como exemplo, considerando a hipótese de que 5 aeronaves estão em conflito, o algoritmo seria capaz de processar até 40 trajetórias

alternativas para cada aeronave em cerca de $18s$, em média, que pode ser interpretado como um tempo adequado para a realização das manobras necessárias.

Tabela 7.5: Resultados dos testes para a abordagem DFS.

Voos	Trajetória	Duração (ms)	Total de Estados
3	3	0,33	10^1
3	20	2,43	10^3
3	50	22,30	10^5
9	3	7,83	10^4
9	7	8.737,60	10^7
12	3	145,27	10^5
15	3	3.858,57	10^7
18	3	106.181,87	10^8

A Tabela 7.6 indica que a abordagem Alfa-Beta pode lidar com *clusters* de conflito ainda maiores, com parâmetros altos o suficiente para garantir a cobertura dos casos encontrados em um cenário real complexo. Foi possível resolver, em um extremo, *clusters* de conflitos com 30 aeronaves e 30 trajetórias para cada aeronave, com solução encontrada, em média, em menos de $50s$. Se considerarmos que, em cenários reais, os *clusters* não devem ultrapassar 15 aeronaves distintas, o modelo encontrou a solução ótima em um tempo médio de $250,57ms$, processando 50 trajetórias para cada uma das aeronaves envolvidas no conflito (espaço de busca equivalente a 10^{25}). Para cenários com menos aeronaves, situação que deve prevalecer, a solução foi encontrada em aproximadamente 1 ms.

Tabela 7.6: Resultados dos testes para a abordagem Alfa-Beta.

Voos	Trajetórias	Duração (ms)	Estados Possíveis (p)	Estados Verificados (v)	Busca (v/p)
3	10	0,00	10^3	10^1	10^{-2}
3	50	0,43	10^5	10^2	10^{-3}
9	3	0,13	10^4	10^1	10^{-3}
9	7	0,63	10^7	10^2	10^{-5}
9	30	7,07	10^{13}	10^3	10^{-10}
9	50	15,20	10^{15}	10^4	10^{-11}
18	3	3,57	10^8	10^2	10^{-6}
18	10	58,07	10^{18}	10^3	10^{-15}
18	50	924,20	10^{30}	10^5	10^{-25}
30	7	3.939,13	10^{25}	10^3	10^{-22}
30	25	39.099,73	10^{41}	10^4	10^{-37}
30	30	47.622,43	10^{44}	10^5	10^{-39}

A estabilidade do modelo pode ser avaliada repetindo continuamente o mesmo cenário (a mesma combinação de quantidades de voos e trajetórias). A Figura 7.3 mostra a distribuição do tempo de execução de diferentes quantidades de aeronaves, focando no cenário com maior número de trajetórias. É possível identificar, pelo gráfico, que a maioria das execuções ficaram próximas ao desempenho médio, ou seja, tempo médio de execução, indicando que o modelo proposto é estável.

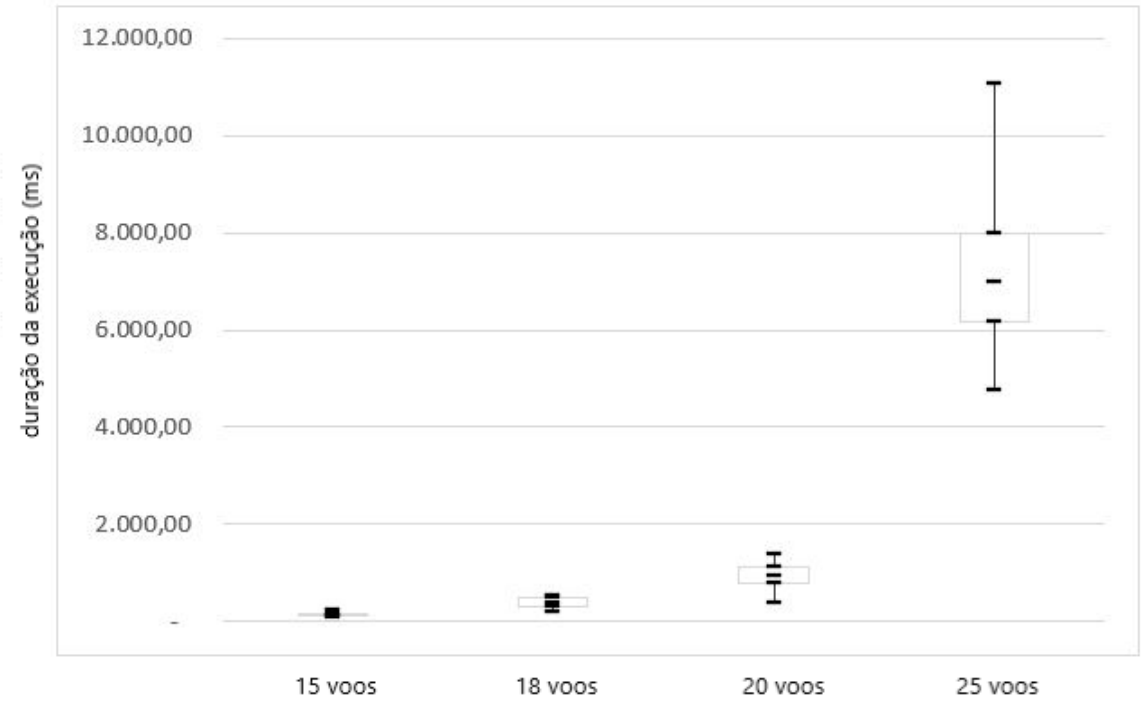

Figura 7.3: Distribuição da duração da abordagem baseada no algoritmo Alfa-Beta.

Por fim, uma análise conclusiva considera a comparação entre o método DFS e Alfa-Beta com relação ao cenário real, ou seja, o número de aeronaves presente no tráfego aéreo atual e a estimativa de trajetórias de cada aeronave. A figura 7.4 apresenta a superfície correspondente à distribuição dos resultados obtidos por DFS (a) e Alfa-Beta (b). Considerando o tempo de execução do eixo y, o número de voos em um grupo de conflito e as trajetórias computadas de cada aeronave do grupo, verifica-se a diferença comportamental entre esses dois métodos. Considerando que, no cenário real, o algoritmo não deve durar mais de $120s$ para encontrar a solução, a maior parte da superfície correspondente à abordagem DFS (a) está acima do plano virtual de $120.000ms$, enquanto que a superfície correspondente à abordagem Alfa-Beta (b) está inteiramente abaixo do mesmo plano virtual. Portanto, para todas as combinações avaliadas, o Alfa-Beta conseguiu encontrar a solução ótima em tempo satisfatório, enquanto que para o método DFS esse comporta-

mento só foi verificado em cenários mais simples (próximo ao ponto de partida, quando há menos aeronaves e menos trajetórias por aeronave).

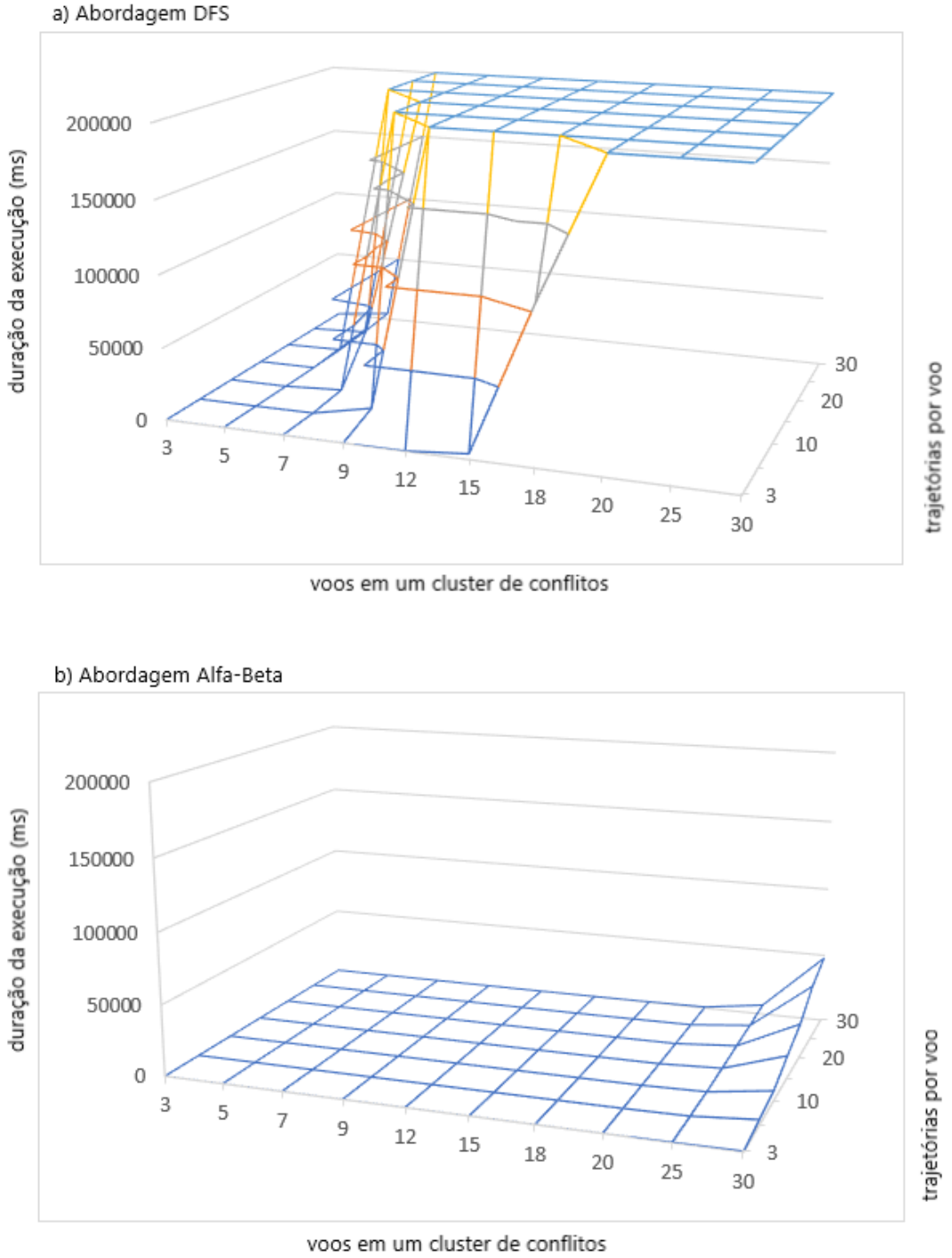

Figura 7.4: Comparação entre as abordagens propostas, DFS (a) e Alfa-Beta (b).

Os resultados mostram que os métodos propostos são viáveis no mundo real em termos de tempo de execução. Pode-se observar na Tabela 7.6 que, considerando a possibilidade

de executar os dois métodos em um tempo aceitável (menos de $120s$), o método Alfa-Beta sempre apresenta melhores resultados que o método DFS. Nos casos mais complicados, considerando 18 aeronaves em conflito e 3 trajetórias para cada aeronave, em comparação com o método DFS ($106,2s$), o método Alfa-Beta é quase 30.000 vezes mais rápido ($3,57ms$). Também nestes casos, considerando o método Alfa-Beta, é possível encontrar uma solução satisfatória para os conflitos do *cluster* em menos de 1 minuto ($47,6s$). Esses cenários são mais complexos que os que devem ser observados no mundo real (conflito envolvendo 30 aeronaves diferentes, 30 trajetórias alternativas são computadas para cada aeronave e um espaço de busca de 10^{44} estados possíveis é gerado). Uma possibilidade de adoção pelas autoridades é integrar o modelo com as ferramentas necessárias para os insumos esperados pelo framework, comparando os resultados obtidos pelo modelo com as instruções emitidas pelo ATC. Nos testes executados, conforme descrito na Seção 7.1.1, fatores incertos como ruído e condições climáticas não foram considerados.

7.2 Abordagem 4DNavMCTS - Estudo de Caso

Nesta seção serão descritos os cenários de testes definidos para avaliação da modelagem `4DNavMCTS`, bem como a análise dos resultados obtidos com os citados testes.

7.2.1 Cenários de Testes

Nos experimentos realizados foram utilizadas bases de trajetórias criadas através do gerador de trajetórias descrito na Subseção 6.4.2. A escolha do gerador de trajetórias visa um melhor controle dos experimentos, pois, desta forma, o número de voos no espaço de simulação pode ser aumentado ou diminuído, o que é difícil de se obter quando avaliamos dados ADS-B reais.

Os experimentos foram planejados para permitir, para cada voo, além da trajetória padrão, 4 trajetórias alternativas: aumentar a velocidade, reduzir a velocidade, aumentar o nível de voo e diminuir o nível de voo. Essa combinação de trajetórias alternativas torna esse procedimento ainda mais complexo porque o espaço de busca aumenta consideravelmente, conforme apresentado na Subseção 5.5.

Os experimentos do `4DNavMCTS` foram comparados com duas abordagens distintas: *Baseline* (BL) e *Heuristicless* (HL). O método BL representa a situação em que o ATC não toma nenhuma ação (conflitos detectados acontecerão). No método HL, ao identificar uma situação de conflito (violação da separação mínima entre as aeronaves), o modelo seleciona aleatoriamente uma ação de contorno, ou seja, seleciona um dos estados possíveis, sem se preocupar com a influência futura dessa escolha.

A distribuição das aeronaves em diferentes níveis de voo dificulta ainda mais o problema. Aumentar o nível de voo de uma aeronave para evitar um conflito no mesmo nível de voo pode resultar em conflito com aeronaves em diferentes níveis de voo. A fim de obter um maior alcance, a metodologia foi testada com diferentes números de aeronaves. Para se obter uma melhor visualização do desempenho da solução proposta, foram realizados testes com diferentes quantidades de aeronaves. Os cenários mais simples são compostos por três e cinco aeronaves no mesmo nível de voo, e os dois cenários mais complexos possuem sete e nove aeronaves, distribuídas entre dois níveis de voo diferentes. Em termos de tamanho do espaço de busca, os cenários experimentais e a complexidade da busca são mostrados na Tabela 7.7.

Tabela 7.7: Experimentos e respectivas complexidades.

Cenário	Voos	Níveis de Voo	Espaço de Busca
#1	3	1	10^{33}
#2	5	1	10^{39}
#3	7	2	10^{33}
#4	9	2	10^{47}

Como forma de avaliar objetivamente o desempenho do `4DNavMCTS`, são definidas as seguintes métricas:

- *violações (V_i)*: indica quantos pares de aeronaves estão próximos o suficiente um do outro em um determinado ponto para representar a violação do parâmetro mínimo de separação. Ou seja, não indica necessariamente uma colisão;
- *duração da violação (D_{vi})*: indica quanto tempo durou o conflito. Este é um parâmetro discreto, que é usado para calcular quantos momentos são afetados pela situação de conflito, que equivale a um instante de rp, definido na Seção 6.4.3;
- *intervenções (I_n)*: indica quantas vezes a trajetória de uma aeronave mudou, ou seja, indica a quantidade de situações nas quais um nó diferente do padrão (trajetória esperada) foi escolhido. O objetivo dessa métrica é avaliar se o modelo é mais ou menos intrusivo do que uma operação real realizada pelo ATC;
- *duração da intervenção(D_{in})*: indica quanto tempo dura a intervenção (totalizador discreto de instantes), semelhante à duração do conflito.

As métricas acima nos permitem avaliar a sensibilidade do `4DNavMCTS` para tomar decisões sobre a operação ideal. A redução do número de violações e a extensão da duração das violações representam melhoria da segurança de voo trazida pelo modelo

proposto. A diminuição nas métricas relacionadas às intervenções mede a eficácia do `4DNavMCTS` em explorar o espaço de busca.

As simulações foram realizadas utilizando-se um computador pessoal HP, modelo HSN-I18C, equipado com uma CPU Intel® Core® i5-8350U 1.70GHz 1.90GHz, 20,0 GB SDRAM e Microsoft Windows 10 Enterprise 64 bits.

7.2.2 Avaliação do Desempenho da Abordagem 4DNavMCTS

Os resultados do experimento mostraram que, para cenários com três e cinco aeronaves no mesmo nível de voo, `4DNavMCTS` possibilitou identificar escolhas quase ótimas, mas muito próximas da abordagem HL. Essa similaridade é razoável, pois as trajetórias alternativas que representam a mudança de nível de vôo podem resolver imediatamente o conflito na mesma altitude. De fato, metade das trajetórias alternativas são indicadas pela mudança de nível de voo.

Como pode ser observado na Tabela 7.8, para um conjunto de experimentos, espera-se que ocorram em média 1,445 violações mínimas de separação, com duração média de 29,723 instantes sem nenhuma intervenção (BL). Ambos HL e `4DNavMCTS` foram capazes de eliminar todas as violações para encontrar a solução ótima para essas duas métricas.

Tabela 7.8: Resultados dos testes com aeronaves no mesmo nível de voo.

		Métricas			
Modelo	Medida	V_i	D_{vi}	I_n	D_{in}
Baseline (BL)	*mínimo*	1	12	-	-
	máximo	2	40	-	-
	média	1,445	29,723	-	-
Heuristicless (HL)	*mínimo*	0	0	1	1
	máximo	0	0	6	13
	média	0	0	2,611	4,222
`4DNavMCTS`	*mínimo*	0	0	1	1
	máximo	0	0	2	2
	média	0	0	1,167	1,167

Em termos de intervenção no espaço aéreo (informação não presente no Baseline, pois não se prevê intervenção neste caso), a abordagem `4DNavMCTS` obteve resultados consideravelmente mais satisfatórios, com uma média de 1,167 intervenções contra 2,611 da abordagem HL, e tempo de intervenção ainda menor, obtendo 1,167 contra 4,222 verificados no HL.

Pode-se observar na Figura 7.5 que o método `4DNavMCTS` é mais estável que HL para o boxplot da métrica D_{in}, ou seja, a dispersão dos resultados é menor. Este fato mostra que o modelo proposto encontrou melhores soluções para eliminar as violações observadas.

Este parâmetro é muito importante, o que mostra que, embora ambos os modelos tenham resolvido todas as violações identificadas, o `4DNavMCTS` se mostra mais adequado por causar menor impacto no espaço aéreo.

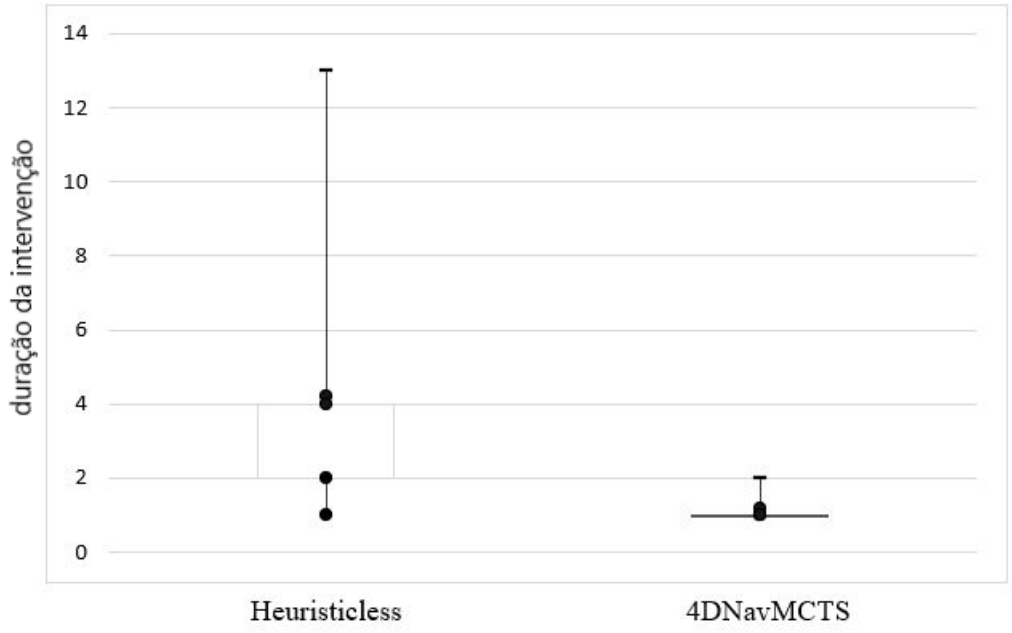

Figura 7.5: Distribuição estatística da duração das intervenções para o cenário com aeronaves no mesmo nível de voo.

Alguns comportamentos diferentes foram observados nas simulações com aeronaves em diferentes níveis de voo, embora ambos os modelos convirjam para resolver os conflitos. Conforme detalhado na Tabela 7.9, apesar de não resolver todos os conflitos em algumas das execuções (métrica Vi maior que 0 no pior caso, ou seja, valor máximo), uma convergência robusta de `4DNavMCTS` é percebida. Em alguns experimentos, `4DNavMCTS` pode ter apresentado algum desequilíbrio na montagem da árvore, o que levou a este comportamento, ou seja, à não resolução de todos os conflitos em uma composição consideravelmente pequena dos testes.

A Tabela 7.9 mostra que para o conjunto de experimentos, esperava-se que houvesse em média 2,461 violações mínimas de separação, com duração média de 80,076 instantes, sem nenhuma intervenção (BL). `4DNavMCTS` conseguiu reduzir o número médio de violações para 0,077, enquanto o número médio de violações em HL foi de 0,385; a duração das violações foi reduzida para 0,769 com `4DNavMCTS` enquanto que a média de violações para HL foi de 4,846. Em relação à intervenção no espaço aéreo, a abordagem `4DNavMCTS` obteve resultados mais satisfatórios: o número médio de intervenções foi de 2,769, enquanto o método HL obteve 8,462; a duração das intervenções verificadas por HL foi de 17,0, enquanto que para o `4DNavMCTS` verificou-se menos da metade, 8,385 instantes em média.

Tabela 7.9: Resultados dos testes com aeronaves em diferentes níveis de voo

		Métricas			
Modelo	Medida	V_i	D_{vi}	I_n	D_{in}
Baseline (BL)	*mínimo*	1	16	-	-
	máximo	3	270	-	-
	média	2,461	80,076	-	-
Heuristicless (HL)	*mínimo*	0	0	1	1
	máximo	1	23	15	36
	média	0,385	4,846	8,462	17,0
`4DNavMCTS`	*mínimo*	0	0	1	1
	máximo	1	10	8	41
	média	0,077	0,769	2,769	8,385

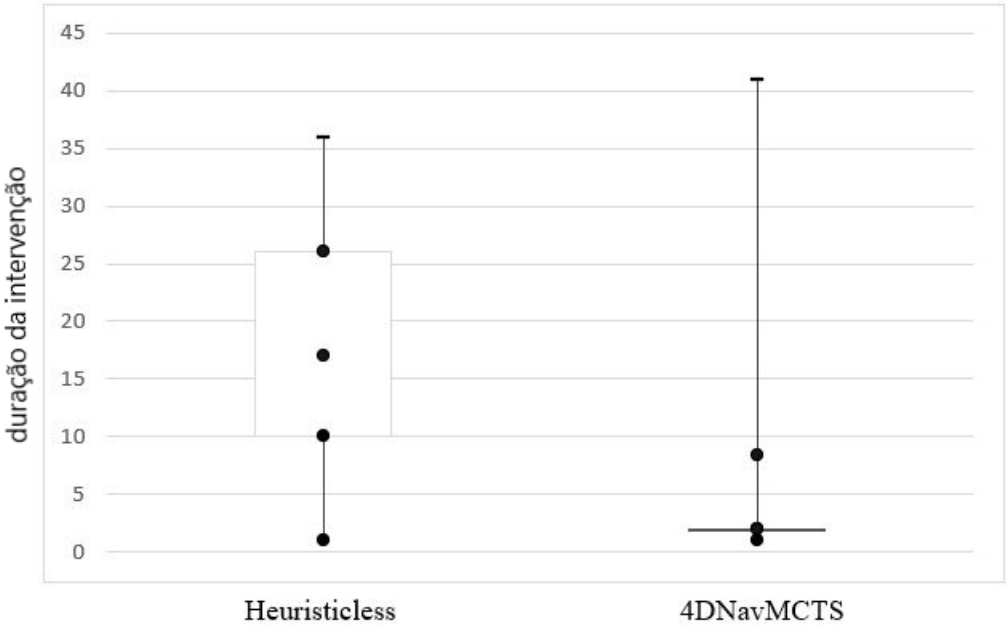

Figura 7.6: Distribuição estatística da duração das intervenções para o cenário com aeronaves em diferentes níveis de voo.

Uma observação importante a ser considerada na análise da Tabela 7.9 é que a existência de violações no caso médio e no pior caso não é um fator limitante ao uso do modelo. Isso se deve a dois pontos principais: 1) em uma operação real, o controlador de tráfego aéreo pode complementar a análise do algoritmo com base em sua experiência, ou seja, trata-se de uma solução de suporte à decisão do controlador; e, 2) mesmo nos casos em que um conflito não é totalmente eliminado, a solução sugere uma decisão que torne esse conflito 'o mais distante possível', de forma que o controlador tenha mais tempo para uma tomada de decisão. Neste segundo ponto, é possível que nas execuções posteriores esse conflito seja definitivamente resolvido, o que não ocorreu antes apenas pelo fato da

janela de tempo futura considerada não contar, ainda, com os estados totalmente livres de conflito, no momento da simulação inicial.

A distribuição observada nos testes envolvendo diferentes níveis de voo, especialmente com relação à duração da intervenção no espaço aéreo, é mostrada na Figura 7.6. O método `4DNavMCTS` também apresenta uma menor dispersão dos resultados, o que indica que o modelo proposto encontrou melhores soluções para eliminar as violações observadas considerando todas as repetições.

Os resultados mostraram uma característica do `4DNavMCTS`, com ênfase no funcionamento ideal do ATC: prever ao máximo situações evitáveis para obter o mínimo de impacto no espaço aéreo. Diferentes configurações de parâmetros de `4DNavMCTS` permitem uma exploração mais aprofundada da árvore de pesquisa. Por exemplo, aumentar o parâmetro *sc* pode ajudar a determinar soluções mais eficazes.

Capítulo 8

Conclusão e Trabalhos Futuros

Analisar grandes quantidades de dados é um dos principais desafios da atualidade, pois o avanço tecnológico no processamento de *big data* trouxe notáveis benefícios em todos os âmbitos da sociedade. No que diz respeito ao transporte aéreo, uma tomada de decisão mais rápida e qualificada traz benefícios econômicos (redução de custo e otimização de rota), e, principalmente, com relação à segurança de voo.

Neste trabalho, um novo modelo de detecção e resolução de conflitos, `4DNavMCTS`, foi desenvolvido para fornecer melhores resultados no cenário de navegação 4D. Além disso, como uma grande quantidade de dados precisa ser analisada, o MCTS foi implementado para resolver esse problema dentro de um prazo razoável e rápido. A modelagem baseada em MCTS desenvolvida pela permite gerenciar as trajetórias das aeronaves de acordo com a implementação do TBO.

Novos métodos para avaliação de trajetória 4D foram apresentados por comparação entre Cassandra e MongoDB. A arquitetura de armazenamento proposta é projetada para atender ao paradigma SWIM, pois esta arquitetura pode gerenciar e fornecer informações para apoiar o processo de tomada de decisão dos *stakeholders* do tráfego aéreo. No entanto, esta aplicação não pretende substituir o controlador de tráfego aéreo ATC, mas sim uma ferramenta de apoio à decisão do operador humano. Esta parte da integração homem-máquina no ATM será mais estudada como uma nova direção para o transporte aéreo inteligente.

8.1 Contribuições da Pesquisa

Os objetivos destacados na definição do problema foram alcançados conforme detalhamento abaixo:

- **maior segurança**: no contexto da CDR, o incremento da segurança ocorre na identificação tempestiva de situações que requeiram alguma intervenção, acompanhada

de um processo decisório rápido e adequado, de forma a solucionar definitivamente o problema. Neste trabalho foi possível verificar que, nas duas abordagens apresentadas, soluções ótimas para resolução de conflitos foram obtidas em tempo satisfatório.

- **volume massivo de dados**: a modegalem de dados baseada em banco de dados NoSQL aplicou conceitos e tecnologias modernas para gerenciamento de *big data*, e possibilitou manipular todos os pontos de todas as trajetórias descritas para os cenários propostos, sem a identificação de qualquer gargalo.
- **incertezas de fatores humanos e ambientais**: especialmente na segunda abordagem proposta, que possibilita manipulação on-line dos dados, a incerteza é incorporada ao modelo a cada iteração, cujo intervalo é definido pelo operador, podendo ser da ordem de segundos. Situações inesperadas e excepcionais como, por exemplo, uma condição climática adversa não prevista, podem ser incorporadas ao modelo assim que se apresentarem, e passam a sensibilizar o algoritmo a partir de então.
- **repetições para obtenção de cenários livres de conflitos**: nas duas abordagens apresentadas o framework trabalha com o conceito de trajetórias alternativas, garantindo que uma combinação de trajetórias escolhida não insira um novo conflito no modelo, elimintando a necessidade de repetição/reprocessamento.

8.2 Resultados Obtidos

Os principais resultados desse trabalho são:

1. Com base no MCTS, `4DNavMCTS` pode lidar satisfatoriamente com problemas complicados de TBO.

   No paradigma TBO, existe um grande foco na interoperabilidade entre os diversos sistemas do ecossistema do transporte aéreo, o que aumente a complexidade desses mesmos sistemas em virtude do grande volumes de dados que necessitam estar disponíveis para diversos agentes.

   Os resultados dos testes demonstraram que a modelagem proposta possibilita configurar os dados de forma a incorporar à solução toda a complexidade do paradigma TBO. A possibilidade de incorporar, de forma imeditada, situações inesperadas, e considerar tais situações na continuidade da solução do problema, favorece a manipulação de diferentes fontes de incerteza envolvidas no contexto do ATM.

2. Considerando o *big data* da navegação, o `4DNavMCTS` pode realizar CDR sob o paradigma da inteligência artificial (AI).

Tratar grandes volumes de dados de forma a possibilitar insights adequados ao problema estudado, sem o uso de técnicas apropriadas, pode-se configurar uma tarefa impossível. Diante disso, o paradigma da inteligência artificial se apresenta como uma disciplina cujo estudo é necessário para o alcance dos objetivos da pesquisa.

A abordagem proposta possibilita com o uso de metodologias adequadas de inteligência artificial uma exploração mais adequada do grande espaço de busca, com foco na exploração das soluções mais adequadas sob o ponto de vista do ATC.

3. Usando uma solução de tomada de decisão com um algoritmo MCTS, `4DNavMCTS` pode encontrar e resolver os potenciais conflitos entre aeronaves e melhorar a segurança de voo com previsão razoável.

   É frequente na literatura a necessidade de reprocessamento do modelo, em virtude de novos conflitos inseridos na tentativa de se solucionar um conflito anterior, característica que pode ser observada em abordagens baseadas em *pairwise*, por exemplo. Neste trabalho foram desenvolvidos modelos que não necessitam de reprocessamento, característica fundamental para que a proposta possa ser aplicada ao cenário real.

   A complexidade inserida na modelagem com o processamento de trajetórias alternativas, que contribui para evitar o surgimento de conflitos futuros, não se mostra um obstáculo ao uso do algoritmo, uma vez que a flexibilidade do MCTS possibilita identificar no espaço de busca os grupos de soluções mais vantajosos.

4. Adotando o Modelo de Espaço Vetorial MEV, `4DNavMCTS` pode ajudar os controladores a escolher quantitativamente a trajetória apropriada e reduzir o risco de conflito.

   A metodologia proposta busca possibilitar ao ATC a tomada de decisão de forma mais confortável, por meio da diminuição do stress da atividade com base na quantidade de informações processadas e na sugestão de escolhas baseadas em cálculos consistentes.

   O MEV se mostrou bastante satisfatório na tarefa de representar as estratégias possíveis, bem como avaliar das estratégias as que apresentam maior similaridade com o comportamento padrão do ATC, possibilitando a escolha de trajetórias subótimas.

8.3 Trabalhos Futuros

O modelo proposto incopra diferentes metodologias com particularidades distintas, que possibilita uma maior exploração futura.

Com relação ao gerenciamento dos dados, foram adotadas duas tecnologias de bancos de dados NoSQL, porém existem outras tecnologias que podem ser experimentadas. Nesse sentido, avaliar a adoção de tecnologias de bancos de dados orientadas a grafos podem ser interessantes.

Especificamente em relação ao MCTS, a função de seleção é um componente bastante sensível do algoritmo. Explorar diferentes parâmetros dessa função ou até mesmo diferentes funções possibilitará uma análise mais abrangente do modelo.

Já com relação à metodologia para seleção de trajetórias, o MEV possibilita a adoção de diferentes funções de similaridade que também podem ser exploradas.

Por fim, uma maior variedade de casos de testes podem sugerir novas configurações do framework com a intenção de se conseguir resultados ainda mais satisfatórios em termos de CDR.

Referências

[1] Li, Y. e S. Manoharan: *A performance comparison of sql and nosql databases*. Em *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, páginas 15–19, Aug 2013. ix, 20, 24, 25

[2] De Oliveira, Italo Romani, Vitor Filincowsky Ribeiro, Li Weigang e Reinaldo Crispiniano Garcia: *Detecting violation of aircraft separation requirements*, 2020. US Patent 10,553,121. ix, 46

[3] Browne, Cameron B., Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis e Simon Colton: *A survey of monte carlo tree search methods*. IEEE Transactions on Computational Intelligence and AI in Games, 4(1):1–43, 2012. x, 24, 25, 57

[4] Jagadish, H. V., Johannes Gehrke, Alexandros Labrinidis, Yannis Papakonstantinou, Jignesh M. Patel, Raghu Ramakrishnan e Cyrus Shahabi: *Big data and its technical challenges*. Commun. ACM, 57:86–94, 2014. 1

[5] Kieckbusch, Diego S, Geraldo P Rocha Filho, Vitor F Ribeiro, Jose ATG Fregnani, Bento S Mattos e Li Weigang: *Negotiation approach by reinforcement learning for takeoff sequencing decision in airports*. Em *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, páginas 4477–4482. IEEE, 2019. 1

[6] ICAO: *Doc 9750-AN/963. 2016–2030 Global Air Navigation Plan. Fifth Edition*. Montréal, Canada, 2016. 1

[7] Murça, Mayara Condé Rocha, Marcelo Xavier Guterres, McWillian de Oliveira, João Basílio Tarelho Szenczuk e Wallace Silva Sant'anna Souza: *Characterizing the brazilian airspace structure and air traffic performance via trajectory data analytics*. Journal of Air Transport Management, 85:101798, 2020, ISSN 0969-6997. `https://www.sciencedirect.com/science/article/pii/S0969699719306349`. 1

[8] Dieudonne, J. E., H. L. Crane, S. R. Jones, C. J. Smith, S. A. Remillard e G. Snead: *Neo (nextgen 4d tm) provided by swim's surveillance soa (sdn asp for rnp 4d ops)*. Em *2007 Integrated Communications, Navigation and Surveillance Conference*, páginas 1–12, April 2007. 1

[9] ICAO: *Doc 9750-AN/963. 2013-2028 Global Air Navigation Plan. Fourth Edition*. Montréal, Canada, 2013. 2

[10] Legrand, Karim: *Correction and Optimization of 4D aircraft trajectories by sharing wind and temperature information.* Tese de Doutoramento, Université de Toulouse, 2019. 2, 3, 9, 11, 12, 56, 63

[11] Mondoloni, Stephane e Nicholas Rozen: *Aircraft trajectory prediction and synchronization for air traffic management applications.* Progress in Aerospace Sciences, 119, setembro 2020. 2

[12] Kuchar, J. K. e L. C. Yang: *A review of conflict detection and resolution modeling methods.* IEEE Transactions on Intelligent Transportation Systems, 1(4):179–189, 2000. 2

[13] Nieto, Francisco Javier Sáez: *The long journey toward a higher level of automation in atm as safety critical, sociotechnical and multi-agent system.* Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 230(9):1533–1547, 2016. https://doi.org/10.1177/0954410015596763. 3

[14] Chen, Yukun, Luhua Zhou, Junyi Yang e Yuedan Yan: *Big data platform of air traffic management.* Em *2019 IEEE 1st International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, 2019. 3

[15] Calvo Fernández, Esther, Luis Sanz, Jose Cordero Garcia e Rosa Valdés: *Conflict-free trajectory planning based on a data-driven conflict-resolution model.* Journal of Guidance, Control, and Dynamics, 40:615–627, março 2017. 3

[16] Wang, Zhengyi, Man Liang, Daniel Delahaye e Weilu Wu: *Learning real trajectory data to enhance conflict detection accuracy in closest point of approach.* Em *USA/Europe Air Traffic Management Research and Development Seminar (ATM2019)*, páginas 1–8, 2019. 3

[17] Tang, Jun: *Conflict detection and resolution for civil aviation: A literature survey.* IEEE Aerospace and Electronic Systems Magazine, 34(10):20–35, 2019. 3, 42, 43

[18] ICAO: *International Standards and Recommended Practices - Air Traffic Services.* International Civil Aviation Organization, 2001. 6

[19] Ribeiro, Vitor Filincowsky: *A Novel Approach for Conflict Detection and Resolution for Trajectory-Based Operations in 4D-Navigation using NoSQL Databases and Local Search Algorithms.* Tese de Doutoramento, Universidade de Brasília, 2019. 6, 7, 9, 38, 46, 47, 50, 55, 56, 68, 70

[20] DECEA: *Diretrizes para Implementação de Programas de Garantia de Qualidade nos Serviços de Tráfego Aéreo.* Brasil, 2004. 6

[21] Magana, Casado e Enrique Díaz Greene Juan: *Trajectory prediction uncertainty modelling for Air Traffic Management.* Tese de Doutoramento, University of Glasgow, 2016. 6, 7, 9, 10

[22] Roberson, Bill: *Fuel Conservation Strategies: Cost Index Explained.* Quarterly publication of Boeing Aeromagazine, 2007. 7

[23] DECEA: *Regras do Ar e Serviços de Tráfego Aéreo*. Volume ICA 100-12, Brasil, 2006. 7, 9

[24] JPDO: *Operational concept for the next generation air transportation system (nextgen)*. Technical Report, 2009. 7

[25] Teller, T. L.: *4d fms tbo pilot-controller human-in-the-loop simulation*. Technical report, Massachusetts Institute of Technology (MIT), 2001. 7, 56, 62

[26] Lee, H.Q. e G.H. Hardy: *4d area navigation system description and flight test results*. Technical Report Technical Note no. 7874, NASA, 1975. 7

[27] Williams, D.H. e S.M. Green: *Airborne four-dimensional flight management in a time-based air traffic control environment*. Technical Report Technical Memorandum no. 4249, NASA, 1991. 7

[28] Caron, Gaétan Marceau: *Optimization and Uncertainty Handling in Air Traffic Management*. Tese de Doutoramento, Université Pariz-Sud, 2014. 8

[29] FAB: *Plano de voo*. Technical Report ICA 100-11, 2000. 8, 9

[30] Gelb, A.: *Applied Optimal Estimation. Applied Optimal Estimation*. MIT Press, 1974. 9

[31] Silva, G. C.: *A eficiência do SISCEAB diante de um cenário desafiador*. Tese de Doutoramento, Universidade do Sul de Santa Catarina, 2019. 12

[32] Jr, C. F. M. Teixeira e G. R. Ferreira: *Uma comparaçãoo entre sgbd's relacionais e nosql para dados de proveniência em workflows científicos*. Universidade Federal Fluminense, Instituto de Computaçãoao, 2014. 15, 19, 62

[33] Moniruzzaman, A. B. M. e S. Hossain: *Nosql database: New era of databases for big data analytics - classification, characteristics and comparison*. Int J Database Theor Appl, 2013. 15, 16, 17, 20, 62, 63

[34] Nayak, A., A. Poriya e Dikshay Poojary: *Article: Type of nosql databases and its comparison with relational databases*. International Journal of Applied Information Systems, 5:16–19, janeiro 2013. 15, 17

[35] Datastax: *Apache cassandra 2.1 documentation*. Relatório Técnico, Apache Cassandra, 2015. 19

[36] Abramova, V. e J. Bernardino: *Nosql databases: Mongodb vs cassandra*. Proceedings of the International C* Conference on Computer Science and Software Engineering, 2013. 20

[37] Y. Kang, I. Park, J. Rhee e Y. Lee: *Mongodb-based repository design for iotgenerated rfid/sensor big data*. IEEE Sensors Journal, página 485–497, 2016. 20, 21

[38] Korf, Richard E.: *Artificial Intelligence Search Algorithms*, página 22. 2ª edição, 2010, ISBN 9781584888208. 21, 22, 23, 52, 53

[39] IATA: *Iata forecast predicts 8.2 billion air travelers in 2037*, 2018. Press Release No.: 62. Date: October 24th, 2018. 23, 52

[40] Chaslot, Guillaume, Sander Bakkes, Istvan Szita e Pieter Spronck: *Monte-carlo tree search: A new framework for game ai*. Em *Proceedings of the Fourth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, AIIDE'08, página 216–217. AAAI Press, 2008. 24

[41] Silver, David, Aja Huang, Christopher J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel e Demis Hassabis: *Mastering the game of go with deep neural networks and tree search*. Nature, 529:484–503, 2016. `http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html`. 25

[42] Turney, Peter D. e Patrick Pantel: *From frequency to meaning: Vector space models of semantics*. J. Artif. Int. Res., 37(1):141–188, jan 2010, ISSN 1076-9757. 26

[43] Salton, G.: *The SMART Retrieval System—Experiments in Automatic Document Processing*. Prentice-Hall, Inc., USA, 1971. 26

[44] Lowe, Will: *Towards a theory of semantic space*. Proceedings of the Twenty-Third Annual Conference of the Cognitive Science Society, janeiro 2001. 26

[45] Gilbert, John, Cleve Moler e Robert Schreiber: *Sparse matrices in matlab: Design and implementation*. SIAM Journal on Matrix Analysis and Applications, 13, maio 1997. 27

[46] Shannon, C. E.: *A mathematical theory of communication*. The Bell System Technical Journal, 27(3):379–423, 1948. 27

[47] Sparck Jones, Karen: *A Statistical Interpretation of Term Specificity and Its Application in Retrieval*, página 132–142. Taylor Graham Publishing, GBR, 1988, ISBN 0947568212. 27

[48] Salton, Gerard e Chris Buckley: *Term weighting approaches in automatic text retrieval*. Relatório Técnico, USA, 1987. 27

[49] Singhal, Amit, Chris Buckley e Mandar Mitra: *Pivoted document length normalization*. Em *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '96, página 21–29, New York, NY, USA, 1996. Association for Computing Machinery, ISBN 0897917928. `https://doi.org/10.1145/243199.243206`. 28

[50] Church, Kenneth Ward: *One term or two?* Em *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '95, página 310–318, New York, NY, USA, 1995. Association for Computing Machinery, ISBN 0897917146. `https://doi.org/10.1145/215206.215376`. 28

[51] Jones, William P. e George W. Furnas: *Pictures of relevance: A geometric analysis of similarity measures.* J. Am. Soc. Inf. Sci., 38(6):420–442, nov 1987, ISSN 0002-8231. 29

[52] Rijsbergen, Cornelis: *The Geometry of Information Retrieval.* agosto 2004, ISBN 978-0-521-83805-4. 29

[53] Dias, Fernando, Hassan Hijazi e David Rey: *Disjunctive linear separation conditions and mixed-integer formulations for aircraft conflict resolution.* agosto 2020. 30

[54] Wang, Ruixin, Richard Alligier, Cyril Allignol, Nicolas Barnier, Nicolas Durand e Alexandre Gondran: *Cooperation of combinatorial solvers for en-route conflict resolution.* Transportation Research Part C: Emerging Technologies, 114:36–58, 2020, ISSN 0968-090X. https://www.sciencedirect.com/science/article/pii/S0968090X19301093. 31

[55] Pelegrín, Mercedes e Claudia D'Ambrosio: *Aircraft deconfliction via mathematical programming: Review and insights.* Transportation Science, 56(1):118–140, jan 2022, ISSN 1526-5447. https://doi.org/10.1287/trsc.2021.1056. 31, 33

[56] Rosenow, Judith, Martin Lindner e Joachim Scheiderer: *Advanced flight planning and the benefit of in-flight aircraft trajectory optimization.* Sustainability, 2021:1383, janeiro 2021. 32, 34

[57] Förster, Stanley, Judith Rosenow e Martin Lindner: *A toolchain for optimizing trajectories under real weather conditions and realistic flight performance.* outubro 2016. 33

[58] Zhang, Xiaoge e Sankaran Mahadevan: *Bayesian neural networks for flight trajectory prediction and safety assessment.* Decis. Support Syst., 131(C), apr 2020, ISSN 0167-9236. https://doi.org/10.1016/j.dss.2020.113246. 33

[59] Soler, Manuel, Valentin Courchelle, Daniel González Arribas e Daniel Delahaye: *A simulated annealing approach to 3d strategic aircraft deconfliction based on en-route speed changes under wind and temperature uncertainties.* Transportation Research Part C Emerging Technologies, 103:194–210, abril 2019. 34

[60] Liu, Yulin e Mark Hansen: *Predicting aircraft trajectories: A deep generative convolutional recurrent neural networks approach.* dezembro 2018. 35

[61] Ayhan, Samet, Pablo Costas e Hanan Samet: *Prescriptive analytics system for long-range aircraft conflict detection and resolution.* páginas 239–248, 2018. 35, 36

[62] Yang, Xuxi e Peng Wei: *Autonomous free flight operations in urban air mobility with computational guidance and collision avoidance.* Trans. Intell. Transport. Sys., 22(9):5962–5975, sep 2021, ISSN 1524-9050. https://doi.org/10.1109/TITS.2020.3048360. 36

[63] Malakis, Stathis, Panagiotis Psaros, Tom Kontogiannis e Christina Malaki: *Classification of air traffic control scenarios using decision trees: Insights from a field study in terminal approach radar environment.* Cogn. Technol. Work, 22(1):159–179, feb 2020, ISSN 1435-5558. https://doi.org/10.1007/s10111-019-00562-7. 36

[64] Ribeiro, Vitor, Henrique Rodrigues, Vitor Faria, Li Weigang e Reinaldo Crispiniano Garcia: *Conflict Detection and Resolution with Local Search Algorithms for 4D-Navigation in ATM*, páginas 129–139. janeiro 2020, ISBN 978-3-030-16659-5. 36, 37

[65] Pamplona, Daniel Alberto, Li Weigang, Alexandre Gomes de Barros, Elcio Hideiti Shiguemori e Claudio Jorge Pinto Alves: *Supervised neural network with multilevel input layers for predicting of air traffic delays.* Em *2018 International Joint Conference on Neural Networks (IJCNN)*, páginas 1–6, 2018. 37

[66] Weigang, Li, Marcos Dib, Daniela Pereira e Antonio Crespo: *Intelligent computing methods in air traffic flow management.* Transportation Research Part C: Emerging Technologies, 18:781–793, outubro 2010. 37

[67] Fan, Jiezhen Sean: *Measurement and Communication Uncertainty in Automated Aircraft Separation Management.* Tese de Doutoramento, Queensland University of Technology, 2016. 39, 40, 41

[68] Tomlin, C., G. Pappas e S. Sastry: *Conflict resolution for air traffic management: A study in multiagent hybrid systems.* IEEE Transactions on vol. 43, página 509–521, 1998. 39

[69] Krozel, J., M. Peters, K. Bilimoria, C. Lee e J. Mitchell: *System performance characteristics of centralized and decentralized air traffic separation strategies.* USA/Europe Air Traffic Management Research and Development Seminar, 2001. 39

[70] Ikeda, Y., B. Nguyen, A. Barfield, B. Sundqvist e S. Jones: *Automatic air collision avoidance systems.* Proceedings of the 41st SICE Annual Conference, vol. 1. IEEE, página 630–635, 2002. 40

[71] Dunbar, W. e R. Murray: *Model predictive control of coordinated multi-vehicle formations.* Proceedings of the 41st IEEE Conference, vol. 4. IEEE, página 4631–4636, 2002. 40

[72] Keviczky, T., F. Borrelli, K. Fregene, D. Godbole e G. Balas: *Decentralized receding horizon control and coordination of autonomous vehicle formations.* Control Systems Technology, IEEE Transactions, vol. 16, página 19–33, 2008. 40

[73] Tsourdos, A., R. Zbikowski e B. White: *Cooperative control strategies for swarm of unmanned aerial vehicles under motion uncertainty.* Institution of Engineering and Technology Conference IET, páginas 1–5, 2007. 40

[74] Lalish, E. e K. Morgansen: *Decentralized reactive collision avoidance for multivehicle systems.* 47th IEEE Conference in Decision and Control, página 1218–1224, 2008. 41

[75] DeJong, P.: *Coalition formation in multi-agent uav systems.* Tese de Doutoramento, University of Central Florida, 2005. 41

[76] Durand, N., J. Alliot e J. Noailles: *Automatic aircraft conflict resolution using genetic algorithms.* ACM symposium on Applied Computing, página 289–298, 1996. 41

[77] Frazzoli, E., J. Oh Z. Mao e E. Feron: *Resolution of conflicts involving many aircraft via semidefinite programming.* Journal of Guidance Control and Dynamics, 1999. 41

[78] Pallottino, L., E. Feron e A. Bicchi: *Conflict resolution problems for air traffic management systems solved with mixed integer programming.* Intelligent Transportation Systems, IEEE Transactions, vol. 3, página 3–11, 2002. 41

[79] Jardin, M.: *Grid-based strategic air traffic conflict detection.* AIAA Guidance, Navigation, and Control Conference and Exhibit, página 1–11, 2005. 41

[80] Archibald, J., J. Hill, N. Jepsen, W. Stirling e R. Frost: *A satisficing approach to aircraft conflict resolution.* Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions, vol. 38, página 510–521, 2008. 42

[81] Hui, Q.: *A satisficing approach to aircraft conflict resolution.* Ship Electronic Engineering, vol. 5, 2008. 42

[82] Besada, Juan A., Guillermo Frontera, Jesus Crespo, Enrique Casado e Javier Lopez-Leones: *Automated aircraft trajectory prediction based on formal intent-related language processing.* Trans. Intell. Transport. Sys., 14(3):1067–1082, 2013. 45, 55

[83] EUROCONTROL e FAA: *Common trajectory predictor structure and terminology.* Relatório Técnico Version 1.0, Action Plan 16 White Paper, Jan 29th 2010. 45

[84] Ball, Michael O, Robert L Hoffman, Dave Knorr, James Wetherly e Mike Wambsganss: *Assessing the benefits of collaborative decision making in air traffic management.* Progress in Astronautics and Aeronautics, 193:239–252, 2001. 46

[85] López-Leonés, J., M. A. Vilaplana, E. Gallo, F. A. Navarro e C. Querejeta: *The aircraft intent description language: A key enabler for air-ground synchronization in trajectory-based operations.* Em *2007 IEEE/AIAA 26th Digital Avionics Systems Conference*, páginas 1.D.4–1–1.D.4–12, Oct 2007. 46

[86] Vilaplana, M., E. Gallo e F. Navarro: *Towards a formal language for the common description of aircraft intent.* 24th Digital Avionics Systems Conference, 1:3.C.5–3.1–9, 2005. 46

[87] Li, Y. e S. Manoharan: *A performance comparison of sql and nosql databases.* IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), página 15–19, 2013. 47, 63

[88] Ribeiro, Vitor Filincowsky, Daniel Alberto Pamplona, José Alexandre T. G. Fregnani, Ítalo Romani de Oliveira e Li Weigang: *Modeling the swarm optimization to build effective continuous descent arrival sequences.* Em *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, páginas 760–765, 2016. 56

[89] Monteiro, Lucas Borges, Li Weigang e Ahmed Abdelfattah Saleh: *An approach of vector space model to link concrete concepts with wiki entities*. Em *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, páginas 313–320, 2015. 60

[90] Li, Jing e Bao Liang Lu: *An adaptive image euclidean distance*. Pattern Recognition, 42(3):349–357, 2009, ISSN 0031-3203. `https://www.sciencedirect.com/science/article/pii/S0031320308003130`. 60

[91] Eurocontrol e FAA: *Human performance in air traffic management safety, sep 2010*. 2010. 64

[92] Kim, Yong Kyun, Yun Hyun Jo, Jin Won Yun, Taeck Keun Oh, Hee Chang Roh, Sang Bang Choi e Hyo Dal Park: *En-route trajectory calculation using flight plan information for effective air traffic management*. JIPS, 6:375–384, setembro 2010. 64

[93] Aviação Civil, Agência Nacional de: *Relatório operacional de segurança operacional (raso) - 2020*. Relatório Técnico, ANAC, 2020. 68

[94] DECEA: *Anuário estatístico de tráfego aéreo*. Relatório Técnico, Headquarters of Air Navigation Management (CGNA), 2018. 69

Apêndice A

Produção Científica

4D Trajectory Conflict Detection and Resolution Using Decision Tree Pruning Method

Qualis Periódico B2.

Monteiro, L. B., Ribeiro, V. F., Garcia, C. P., Rocha Filho, G. P., Weigang, L.

The aviation community develops Trajectory Based Operations (TBO) as an advancement in Air Traffic Management (ATM). There is still the need for an efficient scheme to present the trajectories, manage their associated data, and further detect and resolve the conflicts (CDR) that should eventually occur. In this research, we develop a CDR framework for managing predicted 4-Dimensional Trajectory (4DT). Using Not Only SQL (NoSQL) database (Cassandra and MongoDB), the 4D trajectories of related routes are presented, and the possible conflicts are detected using the strategy of Computing in NoSQL Database. Compared with other conflict detection algorithms, usually by the pairwise method with O(n2) at least, the proposed Decision Tree Pruning Method (DTPM) effectively treats massive data sets. The 4DT data are collected by Trajectory Predictor (TP) concerning 58% of the whole Brazilian air traffic. The comparison results between Cassandra and MongoDB from the case studies show the effectiveness of the proposed methods for conflict detection. In addition, we prove that the conflict resolution approach is viable for application in real scenarios, finding near-optimal solutions for the conflicts identified by the framework. Finally, we also demonstrated the development of sustainable artificial intelligence in intelligent air transportation to improve safety in air traffic management.

IEEE Latin America Transactions, 21(2), 277–287.

Development of SWIM Registry for Air Traffic Management with the Blockchain Support

Qualis Conferência A1.

Igor S. Bonomo, Iuri R. Barbosa, Lucas Borges Monteiro, Camila Bassetto, Alexandre de Barros Barreto, Vinicius Ruela Pereira Borges, Li Weigang.

System Wide Information Management (SWIM) including SWIM Registry for Air Traffic Management (ATM) has been successfully developed and applied in Europe and United States. The most developing countries have just started to study the employment of SWIM concept, which its establishment is required prior to the development of SWIM Registry. In this paper, we introduce the experience of the development of SWIM Registry Brazil, which comprises the study of the architecture, components, services and data accessing. In order to encourage consumers and providers to participate in the SWIM community, we developed a prototype of SWIM Registry Demonstration for the Brazilian ATM society. We propose a model based on Blockchain for managing services currently provided by Brazilian ATM in order to certificate operations which are performed by consumers, authorities and involved stakeholders. The proposed model is expected to provide services for SWIM Registry with integrity, efficiency, security and authenticity, which are fundamental for the proper operation of Brazilian aviation system.

ITSC 2018: 3544-3549.

Using Severe Convective Weather Information for Flight Planning

Qualis Conferência B1.

Iuri Souza Ramos Barbosa, Igor Silva Bonomo, Leonardo L. B. V. Cruciol, Lucas Borges Monteiro, Vinicius Ruela Pereira Borges, Weigang Li.

Aircraft fly in an environment that is subject to constant weather changes, which considerably influences the decision-making process in (ATM). The stakeholders in ATM track weather conditions to appropriately respond against new environmental settings. The proposed work builds an intelligent system to constantly monitor the impact of severe weather on airways, which are corridors with specific width and height connecting two locations in

the airspace. The proposed approach integrates weather information on convection cells obtained from ground-based weather radars, and flight tracking information detailing flight positions in real time. To delimit the boundaries of airways, the set of flight positions is transformed to a more convenient one using linear interpolation. Then, a cluster analysis via (DBSCAN) is performed into this new set. The algorithm compares the positions of the clusters found with the positions of convection cells to identify possible intersections between them. A case study was set up consisting of two phases: 1) flight positions were tracked, and the boundaries of the airways were identified; and 2) convection cell locations were monitored, and compared against the airways in order to identify possible intersections. The solution showed the clusters representing the boundaries of the underlying airways, and some intersections were found during the case study.

ISDA 2018: 140-149.

Segunda Demonstração do SWIM no Cenário Nacional

Camila Bassetto, Lorrene Carolline Vieira Nunes, Ivan Matias da Silva, Inaldo Capistrano Costa, Li Weigang, Lucas Borges Monteiro and Iuri Souza Ramos Barbosa.

The System Wide Information Management (SWIM) is a concept recommended by the International Civil Aviation Organization (ICAO) meant to improve the interoperability of data and systems in the context of the ATM System once adopted. As part of the initiatives headed to SWIM implementation in the Brazilian scenario, technical demonstrations are being held. This paper presents the contextualization of the SWIM Brazilian demonstrations, as well as the results of the second one, which took place in May 2019.

Sitraer 2019.

Utilização do Blockchain para Suporte do Controle de Tráfego Aéreo

Igor Bonomo, Iuri Ramos, Lucas Monteiro, Li Weigang, Vinicius Borges, Camila Bassetto and Alexandre Barreto.

Blockchain is a fundamental technology employed in cryptocurrency, that can be defined as a data structure working as sequence of linked blocks, each one storing a finite set of tran-

sactions(operations or data record). Blockchain have some characteristics that improve the data security, authenticity check for users, transparency in the information within blocks and efficient information sharing. In the development of the System of Wide Information Management (SWIM), Blockchain can provide Air Traffic Management (ATM) data security, sharing information and authenticity of the users for the access to this data. In this paper, we will present the proposed model, algorithm and implementation of Blockchain for SWIM Registry.

Sitraer 2018.

SWIM Registry BR: Serviço de Compartilhamento de Estados

Helena Silva, Diego Santos, Lucas Monteiro, Iuri Ramos and Li Weigang.

O gerenciamento de tráfego aéreo é uma atividade complexa que envolve diversos agentes (companhias aéreas, aeroportos, autoridades, etc) e que está sujeito a diferentes fatores externos (variações climáticas, restrições legais, etc). Para que os agentes tomem decisões otimizadas, é imprescindível que esses tenham conhecimento dos estados uns dos outros. Diante disso, propõe-se o desenvolvimento de um sistema que uniformize a geração de informação pelos agentes, além de providenciar a distribuição da informação para os interessados. Trata-se de um serviço pioneiro para o System Wide Information Management (SWIM) brasileiro, o Compartilhamento de Estados, que disponibilizará informações atualizadas aos participantes do SWIM.

Sitraer 2018.

Desenvolvimento do SWIM Registry Brasileiro

Iuri Souza Ramos Barbosa, Lucas Borges Monteiro, Li Weigang TransLab, Alexandre de Barros Barreto, Camila Bassetto.

The System Wide Information Management (SWIM) consists of standards, infrastructure and governance enabling the management of air traffic information and services. The SWIM registry is crucial to improve interoperability in SWIM, and it behaves as a central repository for services and service related information. SESAR (Europe) and NextGen

(USA) embraced SWIM for modernizing their Air Traffic Management programs. In Brazil, the SWIM ecosystem and the SWIM registry are at an early stage of development. This work presents a review of the main concepts of SWIM and introduces an architecture for a SWIM registry in a Brazilian scenario.

Sitraer 2017.

A System Wyde Information Management Architecture Proposal for Brazilian Scenarios

Lucas Borges Monteiro, Iuri Souza Ramos Barbosa, Li Weigang, José Alexandre Fregnani, Ítalo Romani de Oliveira, Gláucia Balvedi.

The System Wide Information Management (SWIM) aims to provide information sharing and integrate technologies related to Air Traffic Management (ATM), increasing the airspace security and the level of decision-making. The two main examples of SWIM implementation are SESAR, in Europe, and NextGen, in the USA. In Brazil, despite the initiatives related to ATM information processing, such as SAGITARIO and SIGMA, a deployment of a SWIM ecosystem is still required for the implementation of interoperable services. This work presents a review of the main concepts of SWIM and an architecture proposal for the implementation of Brazilian SWIM.

Sitraer 2017.