Autonomous Air Traffic Controller: A Deep Multi-Agent Reinforcement Learning Approach

自主空中交通控制器: 一种深度多代理强化学习方法

Marc Brittain [1] Peng Wei [1]
Marc Brittain [1] Peng Wei [1]

Abstract

摘要

Air traffic control is a real-time safety-critical decision making process in highly dynamic and stochastic environments. In today's aviation practice, a human air traffic controller monitors and directs many aircraft flying through its designated airspace sector. With the fast growing air traffic complexity in traditional (commercial airliners) and low-altitude (drones and eVTOL aircraft) airspace, an autonomous air traffic control system is needed to accommodate high density air traffic and ensure safe separation between aircraft. We propose a deep multi-agent reinforcement learning framework that is able to identify and resolve conflicts between aircraft in a high-density, stochastic, and dynamic en-route sector with multiple intersections and merging points. The proposed framework utilizes an actor-critic model, A2C that incorporates the loss function from Proximal Policy Optimization (PPO) to help stabilize the learning process. In addition we use a centralized learning, decentralized execution scheme where one neural network is learned and shared by all agents in the environment. We show that our framework is both scalable and efficient for large number of incoming aircraft to achieve extremely high traffic throughput with safety guarantee. We evaluate our model via extensive simulations in the BlueSky environment. Results show that our framework is able to resolve 99.97% and 100% of all conflicts both at intersections and merging points, respectively, in extreme high-density air traffic scenarios.

空中交通管制是在高度动态和随机环境中进行的实时安全关键决策过程。在今天航空实践中，一名人类空中交通管制员监控并指导在其指定空域内飞行的多架飞机。随着传统 (商业客机) 和低空 (无人机和 eVTOL 飞机) 空域内航空交通复杂性的快速增长，需要一种自主空中交通管制系统来容纳高密度航空交通并确保飞机之间的安全间隔。我们提出了一种深度多代理强化学习框架，能够识别并解决高密度、随机、动态航路区域内的多个交叉点和汇合点处的飞机冲突。所提出的框架采用了一种演员-评论家模型，即 A2C，它融入了来自近端策略优化 (PPO) 的损失函数以帮助稳定学习过程。此外，我们使用了一种集中学习、分布式执行的方案，其中一个神经网络被学习并共享给环境中的所有代理。我们展示了我们的框架对于大量 incoming 飞机来说是可扩展且有效的，能够实现极高的交通吞吐量并确保安全。我们在 BlueSky 环境中通过大量模拟来评估我们的模型。结果显示，我们的框架能够在极端高密度航空交通场景中分别解决 99.97% 和 100% 的所有冲突。

1. Introduction

1. 引言

1.1. Motivation

1.1. 研究动机

With the rapid increase in global air traffic and expected high density air traffic in specific airspace regions, to guarantee air transportation safety and efficiency becomes a critical challenge. Tactical air traffic control (ATC) decisions to ensure safe separation between aircraft are still being made by human air traffic controllers in en-route airspace sectors, which is the same as compared to 50 years ago (Council et al., 2014). Heinz Erzberger and his NASA colleagues first proposed autonomous air traffic control by introducing the Advanced Airspace Concept (AAC) to increase airspace capacity and operation safety by designing automation tools such as the Autoresolver and TSAFE to augment human controllers (Erzberger, 2005; Erzberger & Heere, 2010; Farley & Erzberger, 2007) in conflict resolution. Inspired by

Erzberger, we believe that a fully automated ATC system is the ultimate solution to handle the high-density, complex, and dynamic air traffic in the future en-route and terminal airspace for commercial air traffic.

随着全球航空交通的迅速增长以及特定空域区域预期的高密度航空交通，确保航空运输的安全和效率成为了一个关键挑战。战术空中交通管制 (ATC) 决策，以确保飞机之间的安全间隔，仍然由在航路空域扇区的空中交通管制员人工进行，这与 50 年前相比并无二致 (Council et al., 2014)。Heinz Erzberger 及其 NASA 同事首次提出通过引入先进空域概念 (AAC) 来自主控制空中交通，以设计自动化工具如 Autoresolver 和 TSAFE 来增强人工管制员在冲突解决中的能力，从而提高空域容量和运行安全性 (Erzberger, 2005; Erzberger & Heere, 2010; Farley & Erzberger, 2007)。受到 Erzberger 的启发，我们相信完全自动化的 ATC 系统是处理未来商业航空在航路和终端空域的高密度、复杂且动态航空交通的最终解决方案。

In recent proposals for low-altitude airspace operations such as UAS Traffic Management (UTM) (Kopardekar et al., 2016), U-space (Undertaking, 2017), and urban air mobility (Mueller, Kopardekar, and Goodrich, 2017), there is also a strong demand for an autonomous air traffic control system to provide advisories to these intelligent aircraft, facilitate on-board autonomy or human operator decisions, and cope with high-density air traffic while maintaining safety and efficiency (Air, 2015; Airbus, 2018; Google, 2015; Holden & Goel, 2016; Kopardekar, 2015; Mueller et al., 2017; Uber, 2018). According to the most recent study by Hunter and Wei (Hunter & Wei, 2019), the key to these low-altitude airspace operations is to design the autonomous ATC on structured airspace to achieve envisioned high throughput. Therefore, the critical challenge here is to design an autonomous air traffic control system to provide real-time advisories to aircraft to ensure safe separation both along air routes and at intersections of these air routes. Furthermore, we need this autonomous ATC system to be able to manage multiple intersections and handle uncertainty in real time.

在近期关于低空空域运行的建议中，如无人机交通管理 (UTM)(Kopardekar 等人，2016 年)、U 空间 (Undertaking，2017 年) 和城市空中出行 (Mueller，Kopardekar 和 Goodrich，2017 年)，也强烈需求一个自主空中交通控制系统，以向这些智能飞机提供咨询，便利机上自主系统或人类操作员的决策，并在保持安全和效率的同时应对高密度空中交通 (Air，2015 年；Airbus，2018 年；Google，2015 年；Holden & Goel，2016 年；Kopardekar，2015 年；Mueller 等人，2017 年；Uber，2018 年)。根据 Hunter 和 Wei 的最新研究 (Hunter & Wei，2019 年)，这些低空空域运行的关键是设计基于结构化空域的自主 ATC 以实现预期的高通量。因此，这里的重大挑战是设计一个自主空中交通控制系统，以实时向飞机提供咨询，确保在航线及其交汇处保持安全间隔。此外，我们需要这个自主 ATC 系统能够实时管理多个交汇点并处理不确定性。

To implement such a system, we need a model to perceive the current air traffic situation and provide advisories to aircraft in an efficient and scalable manner. Reinforcement learning, a branch of machine learning, is a promising way to solve this problem. The goal in reinforcement learning is to allow an agent to learn an optimal policy by interacting with an environment. The agent is trained by first perceiving the state in the environment, selecting an action based on the perceived state, and receiving a reward based on this perceived state and action. By formulating the tasks of human air traffic controllers as a reinforcement learning problem, the trained agent can provide dynamic real-time air traffic advisories to aircraft with extremely high safety guarantee under uncertainty and little computation overhead.

为了实现这样一个系统，我们需要一个模型来感知当前的空中交通情况，并以高效和可扩展的方式为飞机提供咨询。强化学习，作为机器学习的一个分支，是解决这个问题的有前途的方法。强化学习的目标是让一个智能体通过与环境的交互来学习一个最优策略。智能体通过首先感知环境中的状态，根据感知到的状态选择一个动作，并基于这个感知到的状态和动作接收一个奖励来进行训练。将人类空中交通管制员的任务定义为强化学习问题，训练后的智能体能够在不确定性和很小的计算开销下，为飞机提供动态的实时空中交通咨询，并具有极高的安全保障。

Artificial intelligence (AI) algorithms are achieving performance beyond humans in many real-world applications today. An artificial intelligence agent called AlphaGo built by DeepMind defeated the world champion Ke Jie in three matches of Go in May 2017 (Silver & Hassabis, 2016). This notable advance in the AI field demonstrated the theoretical foundation and computational capability to potentially augment and facilitate human tasks with intelligent agents and AI technologies. To utilize such techniques, fast-time simulators are needed to allow the agent to efficiently learn in the environment. Until recently, there were no open-source high-quality air traffic control simulators that allowed for fast-time simulations to

[1] Department of Aerospace Engineering, Iowa State University, Ames, IA, 50021, USA. Correspondence to: Marc Brittain < mwb@iastate.edu > , Peng Wei < pwei@iastate.edu > .

[1] 航空航天工程系，爱荷华州立大学，艾姆斯，IA，50021，美国。通讯作者:Marc Brittain < mwb@iastate.edu > ，Peng Wei < pwei@iastate.edu > 。

enable an AI agent to interact with. The air traffic control simulator, BlueSky, developed by TU Delft allows for realistic real-time air traffic scenarios and we decide to use this software as the environment and simulator for performance evaluation of our proposed framework (Hoekstra & Ellerbroek, 2016).

人工智能 (AI) 算法在许多现实世界应用中的性能已经超越了人类。由 DeepMind 构建的人工智能体 AlphaGo 在 2017 年 5 月的比赛中击败了世界冠军柯洁 (Silver & Hassabis, 2016)。这一在 AI 领域的显著进步展示了理论基础的坚实和计算能力，有可能通过智能代理和 AI 技术增强和促进人类任务的完成。为了利用这些技术，需要快速时间模拟器以允许代理在环境中高效学习。直到最近，还没有开源的高质量空中交通控制模拟器能够进行快速时间模拟，以使 AI 代理能够与之交互。由代尔夫特理工大学开发的空中交通控制模拟器 BlueSky 能够提供逼真的实时空中交通场景，我们决定使用这个软件作为环境和模拟器来评估我们提出的框架的性能 (Hoekstra & Ellerbroek, 2016)。

In this paper, a deep multi-agent reinforcement learning framework is proposed to enable autonomous air traffic separation in en-route airspace, where each aircraft is represented by an agent. Each agent will comprehend the current air traffic situation and perform online sequential decision making to select speed advisories in real-time to avoid conflicts at intersections, merging points, and along route. Our proposed framework provides another promising potential solution to enable an autonomous air traffic control system.

在本文中，我们提出了一个深度多代理强化学习框架，以实现在航路空域中的自主空中交通分离，其中每架飞机都由一个代理表示。每个代理将理解当前的空中交通情况，并执行在线顺序决策，以实时选择速度建议，避免在交叉点、合并点和沿航线发生冲突。我们提出的框架为实自主空中交通控制系统提供了另一个有前景的潜在解决方案。

1.2. Related Work

1.2. 相关工作

Deep reinforcement learning has been widely explored in ground transportation in the form of traffic light control (Genders and Razavi, 2016; Liang, Du, Wang, and Han, 2018). In these approaches, the authors deal with a single intersection and use one agent per intersection to control the traffic lights. Our problem is similar to ground transportation in the sense we want to provide speed advisories to aircraft to avoid conflict, in the same way a traffic light advises cars to stop and go. The main difference with our problem is that we need to control the speed of each aircraft to ensure there is no along route conflict. In our work, we represent each aircraft as an agent instead of the intersection to handle along route and intersection conflicts.

深度强化学习已经在地面交通中得到了广泛研究，其形式为交通信号灯控制 (Genders 和 Razavi，2016；Liang、Du、Wang 和 Han，2018)。在这些方法中，作者们处理的是单个交叉口，并为每个交叉口使用一个代理来控制交通信号灯。我们的问题与地面交通类似，因为我们希望向飞机提供速度建议以避免冲突，就像交通信号灯建议汽车停止和行驶一样。我们问题的主要区别在于，我们需要控制每架飞机的速度，以确保航路上没有冲突。在我们的工作中，我们将每架飞机表示为一个代理，而不是交叉口，以处理航路和交叉口的冲突。

There have been many important contributions to the topic of autonomous air traffic control. One of the most promising and well-known lines of work is the Autoresolver designed and developed by Heinz Erzberger and his NASA colleagues (Erzberger, 2005; Erzberger & Heere, 2010; Farley & Erzberger, 2007). It employs an iterative approach, sequentially computing and evaluating candidate trajectories, until a trajectory is found that satisfies all of the resolution conditions. The candidate trajectory is then output by the algorithm as the conflict resolution trajectory. The Autore-solver is a physics-based approach that involves separate components of conflict detection and conflict resolution. It has been tested in various large-scale simulation scenarios with promising performance.

自主空中交通控制领域有许多重要的贡献。其中最有前景且广为人知的工作之一是由 Heinz Erzberger 和他的 NASA 同事们设计开发的 Autoresolver(Erzberger，2005；Erzberger & Heere，2010；Farley & Erzberger，2007)。它采用迭代方法，依次计算和评估候选轨迹，直到找到满足所有解决条件的轨迹。然后算法输出候选轨迹作为冲突解决轨迹。Autoresolver 是一种基于物理的方法，涉及冲突检测和冲突解决分开的组件。它已经在各种大规模仿真场景中进行了测试，并表现出了有希望的性能。

Strategies for increasing throughput of aircraft while minimizing delay in high-density sectors are currently being designed and implemented by NASA. These works include the Traffic Management Advisor (TMA) (Erzberger and Itoh, 2014) or Traffic Based Flow Management (TBFM), a central component of ATD-1 (Baxley, Johnson, Scardina, and Shay, 2016). In this approach, a centralized planner determines conflict free time-slots for aircraft to ensure separation requirements are maintained at the metering fix.

Our algorithm also is able to achieve a conflict free metering fix by allowing the agents to learn a cooperative strategy that is queried quickly online during execution, unlike TBFM. Another main difference with our work is that our proposed framework is a decentralized framework that can handle uncertainty. In TMA or TBFM, once the arrival sequence is determined and aircraft are within the "freeze horizon" no deviation from the sequence is allowed, which could be problematic if one aircraft becomes uncooperative.

美国国家航空航天局 (NASA) 目前正在设计和实施旨在提高飞机在高密度区域吞吐量同时最小化延迟的策略。这些工作包括交通管理顾问 (TMA)(Erzberger 和 Itoh，2014) 或基于交通的流量管理 (TBFM)，这是 ATD-1(Baxley、Johnson、Scardina 和 Shay，2016) 的核心组成部分。在这种方法中，集中式规划器为飞机确定无冲突的时间槽，以确保在计量点保持分离要求。我们的算法也能通过允许代理学习一种快速在线查询的协作策略来实现无冲突的计量点，这与 TBFM 不同。我们的工作与 TMA 或 TBFM 的另一个主要区别在于，我们提出的框架是一个分布式框架，能够处理不确定性。在 TMA 或 TBFM 中，一旦确定到达顺序且飞机进入 "冻结范围"，就不允许偏离该顺序，如果有一架飞机变得不合作，这可能会带来问题。

Multi-agent approaches have also been applied to conflict resolution (Wollkind, Valasek, and Ioerger, 2004). In this line of work, negotiation techniques are used to resolve identified conflicts in the sector. In our research, we do not impose any negotiation techniques, but leave it to the agents to derive negotiation techniques through learning and training.

多代理方法也已被应用于冲突解决 (Wollkind、Valasek 和 Ioerger，2004)。在这类工作中，使用谈判技术来解决区域内识别的冲突。在我们的研究中，我们没有强加任何谈判技术，而是让代理通过学习和训练自行推导谈判技术。

Reinforcement learning and deep Q-networks have been demonstrated to play games such as Go, Atari and Warcraft, and most recently Starcraft II (Amato and Shani, 2010; Mnih, Kavukcuoglu, Silver, Graves, Antonoglou, Wierstra, and Riedmiller, 2013; Silver, Huang, Maddison, Guez, Sifre, Van Den Driessche, Schrittwieser, Antonoglou, Panneer-shelvam, Lanctot, et al., 2016; Vinyals, Ewalds, Bartunov, Georgiev, Vezhnevets, Yeo, Makhzani, Küttler, Agapiou, Schrittwieser, et al., 2017). The results from these papers show that a well-designed, sophisticated AI agent is capable of learning complex strategies. It was also shown in previous work that a hierarchical deep reinforcement learning agent was able to avoid conflict and choose optimal route combinations for a pair of aircraft (Brittain and Wei, 2018).

强化学习和深度 Q 网络已经在玩围棋、雅达利游戏和魔兽世界等游戏方面得到展示，最近还扩展到了星际争霸 II(Amato 和 Shani，2010；Mnih，Kavukcuoglu，Silver，Graves，Antonoglou，Wierstra 和 Riedmiller，2013；Silver，Huang，Maddison，Guez，Sifre，Van Den Driessche，Schrittwieser，Antonoglou，Panneer-shelvam，Lanctot 等人，2016；Vinyals，Ewalds，Bartunov，Georgiev，Vezhnevets，Yeo，Makhzani，Küttler，Agapiou，Schrittwieser 等人，2017)。这些论文的结果表明，设计精良、复杂的人工智能代理能够学习复杂的策略。之前的论文还展示了分层深度强化学习代理能够避免冲突，并为一对飞机选择最优的航线组合 (Brittain 和 Wei，2018)。

Recently the field of multi-agent collision avoidance has seen much success in using a decentralized noncommunicating framework in ground robots (Chen, Liu, Everett, and How, 2017; Everett, Chen, and How, 2018). In this work, the authors develop an extension to the policy-based learning algorithm (GA3C) that proves to be efficient in learning complex interactions between many agents. We find that the field of collision avoidance can be adapted to conflict resolution by considering larger separation requirements, so our framework is inspired by the ideas set forth by (Everett et al., 2018).

最近，在多代理碰撞避免领域，使用去中心化非通信框架在地面机器人上取得了很大成功 (Chen，Liu，Everett 和 How，2017；Everett，Chen 和 How，2018)。在这项工作中，作者开发了一种基于策略学习算法 (GA3C) 的扩展，该算法在学习和许多代理之间的复杂交互方面证明是有效的。我们发现，通过考虑更大的分离要求，可以将避障领域适应为冲突解决，因此我们的框架受到了 (Everett 等人，2018) 提出的想法的启发。

In this paper, the deep multi-agent reinforcement learning framework is developed to solve the separation problem for autonomous air traffic control in en-route dynamic airspace where we avoid the computationally expensive forward integration method by learning a policy that can be quickly queried. The results show that our framework has very promising performance.

在本文中，我们开发了一种深度多代理强化学习框架，用于解决在航路动态空域中自主空中交通控制的分离问题，我们通过学习一个可以快速查询的策略，避免了计算成本高昂的前向积分方法。结果显示，我们的框架具有非常令人期待的性能。

The structure of this paper is as follows: in Section II, the background of reinforcement learning, policy based learning, and multi-agent reinforcement learning will be introduced. In Section III, the description of the problem and its mathematical formulation of deep multi-agent reinforcement learning are presented. Section IV presents our designed deep multi-agent reinforcement learning framework to solve this problem. The numerical experiments and results are shown in Section V, and Section VI

concludes this paper.

本文的结构如下: 第二部分将介绍强化学习、基于策略的学习和多代理强化学习的背景。第三部分将描述问题及其深度多代理强化学习的数学表述。第四部分呈现我们设计的深度多代理强化学习框架来解决这个问题。第五部分展示了数值实验和结果，第六部分对本论文进行总结。

2. Background

2. 背景

2.1. Reinforcement Learning

2.1. 强化学习

Reinforcement learning is one type of sequential decision making where the goal is to learn how to act optimally in a given environment with unknown dynamics. A reinforcement learning problem involves an environment, an agent, and different actions the agent can select in this environment. The agent is unique to the environment and we assume the agent is only interacting with one environment. If we let t represent the current time, then the components that make up a reinforcement learning problem are as follows:

强化学习是一种顺序决策，目标是学习如何在具有未知动态的给定环境中如何最优地行动。一个强化学习问题包括环境、代理以及代理在这个环境中可以选择的不同动作。代理对环境是唯一的，我们假设代理只与一个环境交互。如果我们让 t 表示当前时间，那么构成强化学习问题的组成部分如下:

- S - The state space S is a set of all possible states in the environment
- S - 状态空间 S 是环境中所有可能状态的集合
- A - The action space A is a set of all actions the agent can select in the environment
- A - 动作空间 A 是代理在环境中可以选择的所有动作的集合
- $r\left(s_t, a_t\right)$ - The reward function determines how much reward the agent is able to acquire for a given $\left(s_t, a_t\right)$ transition
- $r\left(s_t, a_t\right)$ - 奖励函数决定了代理在给定的 $\left(s_t, a_t\right)$ 转移中能够获得多少奖励
- $\gamma \in [0,1]$ - A discount factor determines how far in the future to look for rewards. As $\gamma \rightarrow 0$, immediate rewards are emphasized, whereas, when $\gamma \rightarrow 1$, future rewards are prioritized.
- $\gamma \in [0,1]$ - 折扣因子决定了要查看多远的未来以获取奖励。当 $\gamma \rightarrow 0$ 时，会强调即时奖励，而当 $\gamma \rightarrow 1$ 时，会优先考虑未来奖励。

S contains all information about the environment and each element s_t can be considered a snapshot of the environment at time t . The agent accepts s_t and with this, the agent then selects an action, a_t . By selecting action a_t , the state is now updated to s_{t+1} and there is an associated reward from making the transition from $\left(s_t, a_t\right) \rightarrow s_{t+1}$. How the state evolves from $s_t \rightarrow s_{t+1}$ given action a_t is dependent upon the dynamics of the system, which is often unknown. The reward function is user defined, but needs to be carefully designed to reflect the goal of the environment.

S 包含了关于环境以及每个元素 s_t 的所有信息，可以将其视为在时间 t 的环境快照。智能体接受 s_t ，并在此基础上选择一个动作 a_t 。通过选择动作 a_t ，状态现在更新为 s_{t+1} ，并且有一个与从 $\left(s_t, a_t\right) \rightarrow s_{t+1}$ 转变的关联奖励。状态如何从 $s_t \rightarrow s_{t+1}$ 通过动作 a_t 发展取决于系统的动态特性，这通常是未知的。奖励函数由用户定义，但需要仔细设计以反映环境的目标。

From this framework, the agent is able to extract the optimal actions for each state in the environment by maximizing a cumulative reward function. We call the actions the agent selects for each state in the environment a policy. Let π represent some policy and T represent the total time for a given environment, then the optimal policy can be defined as follows:

在这个框架下，智能体能够通过最大化累积奖励函数来提取环境中每个状态的最优动作。我们称智能体在环境中每个状态选择的一系列动作为策略。设 π 表示某种策略，T 表示给定环境的总时间，那么最优策略可以定义如下:

$$\pi^* = \arg\max_{\pi} E\left[\sum_{t=0}^{T}\left(r\left(s_t, a_t\right) \mid \pi\right)\right]. \tag{1}$$

If we design the reward to reflect the goal in the environment, then by maximizing the total reward, we have obtained the optimal solution to the problem.

如果我们设计奖励以反映环境中的目标，那么通过最大化总奖励，我们就获得了问题的最优解。

2.2. Policy-Based Learning

2.2. 基于策略的学习

In this work, we consider a policy-based reinforcement learning algorithm to generate policies for each agent to execute. The advantage of policy-based learning is that these algorithms are able to learn stochastic policies, whereas value-based learning can not. This is especially beneficial in non-communicating multi-agent environments, where there is uncertainty in other agent's action. A3C (Mnih, Badia, Mirza, Graves, Lillicrap, Harley, Silver, and Kavukcuoglu, 2016), a recent policy-based algorithm, uses a single neural network to approximate both the policy (actor) and value (critic) functions with many threads of an agent running in parallel to allow for increased exploration of the state-space. The actor and critic are trained according to the two loss functions:

在这项工作中，我们考虑了一种基于策略的强化学习算法，为每个代理生成执行策略。基于策略学习的优点在于这些算法能够学习随机策略，而基于价值的学习则不能。这在非通信多代理环境中尤其有益，因为其他代理的行为存在不确定性。A3C(Mnih, Badia, Mirza, Graves, Lillicrap, Harley, Silver, 和 Kavukcuoglu, 2016)，一种最近的基于策略的算法，使用单个神经网络来近似策略 (演员) 和价值 (评论家) 函数，并使用并行运行代理的多个线程来允许对状态空间的增加探索。演员和评论家根据两个损失函数进行训练:

$$L_\pi = \log \pi\left(a_t, \mid s_t\right)\left(R_t - V\left(s_t\right)\right) + \beta \cdot H\left(\pi\left(s_t\right)\right) \tag{2}$$

$$L_v = \left(R_t - V\left(s_t\right)\right)^2 \tag{3}$$

where in (2), the first term $\log \pi\left(a_t, \mid s_t\right)\left(R_t - V\left(s_t\right)\right)$ reduces the probability of sampling an action that led to a lower return than was expected by the critic and the second term, $\beta \cdot H\left(\pi\left(s_t\right)\right)$ is used to encourage exploration by discouraging premature convergence to suboptimal deterministic polices. Here H is the entropy and the hyperparameter β controls the strength of the entropy regularization term. In (3), the critic is trained to approximate the future discounted rewards, $R_t = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V\left(s_{t+k}\right)$.

其中在 (2) 中，第一项 $\log \pi\left(a_t, \mid s_t\right)\left(R_t - V\left(s_t\right)\right)$ 减少了采样导致低于评论家预期回报的动作的概率，第二项 $\beta \cdot H\left(\pi\left(s_t\right)\right)$ 用于通过阻止过早收敛到次优确定性策略来鼓励探索。这里的 H 是熵，超参数 β 控制熵正则项的强度。在 (3) 中，评论家被训练来近似未来折现奖励 $R_t = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V\left(s_{t+k}\right)$ 。

One drawback of L_π is that it can lead to large destructive policy updates and hinder the final performance of the model. A recent algorithm called Proximal Policy Optimization (PPO) solved this problem by introducing a new type of loss function that limits the change from the previous policy to the new policy (Schulman, Wolski, Dhariwal, Radford, and Klimov,2017). If we let $r_t\left(\theta\right)$ denote the probability ratio and θ represent the neural network weights at time $t, r_t\left(\theta\right) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$, the PPO loss function can be formulated as follows:

L_π 的一个缺点是它可能导致大的破坏性策略更新，并阻碍模型的最终性能。一种名为近端策略优化 (PPO) 的最近算法通过引入一种新的损失函数解决了这个问题，该损失函数限制了从旧策略到新策略的变化 (Schulman, Wolski, Dhariwal, Radford, 和 Klimov, 2017)。如果我们让 $r_t\left(\theta\right)$ 表示概率比，θ 代表时间 $t, r_t\left(\theta\right) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ 的神经网络权重，PPO 损失函数可以表述如下:

$$L^{\mathrm{CLIP}}\left(\theta\right) =$$

$$E_t\left[\min\left(r_t\left(\theta\right)\left(A\right), \mathrm{clip}\left(r_t\left(\theta\right), 1-\epsilon, 1+\epsilon\right)\left(A\right)\right)\right], \tag{4}$$

where $A := R_t - V(s_t)$ and ϵ is a hyperparameter that determines the bound for $r_t(\theta)$. This loss function allows the previous policy to move in the direction of the new policy, but by limiting this change it is shown to lead to better performance (Schulman et al., 2017).

其中 $A := R_t - V(s_t)$ 和 ϵ 是一个超参数，它决定了 $r_t(\theta)$ 的界限。这个损失函数允许之前的策略向新策略的方向移动，但通过限制这种变化，它被证明能够带来更好的性能 (Schulman et al., 2017)。

2.3. Multi-Agent Reinforcement Learning

2.3. 多智能体强化学习

In multi-agent reinforcement learning, instead of considering one agent's interaction with the environment, we are concerned with a set of agents that share the same environment (Bu, Babu, De Schutter, et al., 2008). Fig. 1 shows the progression of a multi-agent reinforcement learning problem. Each agent has its own goals that it is trying to achieve in the environment that is typically unknown to the other agents. In these types of problems, the difficulty of learning useful policies greatly increases since the agents are both interacting with the environment and each other. One strategy for solving multi-agent environments is Independent Q-learning (Tan, 1993), where other agents are considered to be part of the environment and there is no communication between agents. This approach often fails since each agent is operating in the environment and in return, creates learning instability. This learning instability is caused by the fact that each agent is changing its own policy and how the agent changes this policy will influence the policy of the other agents (Matignon, Laurent, and Le Fort-Piat, 2012). Without some type of communication, it is very difficult for the agents to converge to a good policy.

在多智能体强化学习中，我们关注的不是单个智能体与环境的交互，而是一组共享同一环境的智能体 (Bu, Babu, De Schutter, et al., 2008)。图 1 展示了多智能体强化学习问题的进展。每个智能体在环境中都有自己的目标要实现，这些目标通常对其他智能体是未知的。在这类问题中，由于智能体既与环境交互，也彼此交互，学习有用的策略的难度大大增加。解决多智能体环境的一种策略是独立 Q 学习 (Tan, 1993)，在这种策略中，其他智能体被视为环境的一部分，智能体之间没有通信。这种方法通常失败，因为每个智能体都在环境中操作，并反过来造成学习的不稳定性。这种学习不稳定性是由每个智能体都在改变自己的策略这一事实造成的，一个智能体如何改变策略将会影响其他智能体的策略 (Matignon, Laurent, and Le Fort-Piat, 2012)。没有某种形式的通信，智能体很难收敛到好的策略。

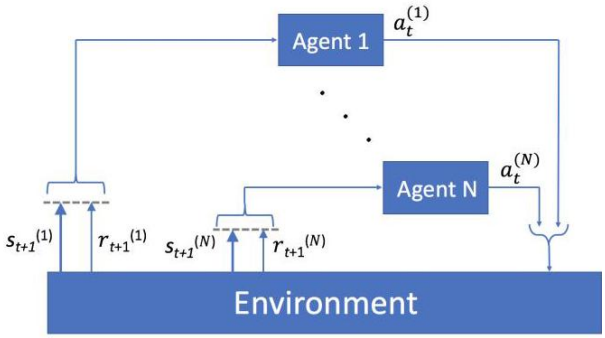

Figure 1. Progression of a multi-agent reinforcement learning problem.
图 1. 多智能体强化学习问题的进展。

3. Problem Formulation

3. 问题公式化

In real world practice, air traffic controllers in en-route and terminal sectors are responsible for separating aircraft. In our research, we used the BlueSky air traffic control simulator as our deep reinforcement learning environment. We developed two challenging Case Studies: one with multiple intersections (Case Study 1) and one with a merging point (Case Study 2), both with high-density air traffic to evaluate the performance of our deep multi-agent reinforcement learning framework.

在实际操作中，航路和终端区的空中交通管制员负责分离飞机。在我们的研究中，我们使用了 BlueSky 空中交通管制模拟器作为深度强化学习的环境。我们开发了两个具有挑战性的案例研究: 一个是具有多个交叉点的案例 (案例研究 1)，另一个是具有合并点的案例 (案例研究 2)，两个案例均在高密度空中交通条件下评估我们深度多代理强化学习框架的性能。

3.1. Objective

3.1. 目标

The objective in these Case Studies is to maintain safe separation between aircraft and resolve conflict for aircraft in the sector by providing speed advisories. In Case Study 1, three routes are constructed with two intersections so that the agents must navigate through the intersection with no conflicts. In Case Study 2, there are two routes that reach a merging point and continue on one route, so ensuring proper separation requirements at the merging point is a difficult problem to solve. In order to obtain the optimal solution in this environment, the agents have to maintain safe separation and resolve conflict and every time step in the environment. To increase the difficulty of the Case Studies and to provide a more realistic environment, aircraft enter the sector stochastically so that the agents need to develop a strategy instead of simply memorizing actions.

这些案例研究的目标是在扇区内的飞机之间保持安全间隔并解决飞机的冲突，通过提供速度建议。在案例研究 1 中，构建了三条航线并设有两个交叉点，因此代理必须在无冲突的情况下穿越交叉点。在案例研究 2 中，有两条航线在合并点汇合并继续一条航线，因此在合并点确保适当间隔要求是一个难以解决的问题。为了在这个环境中获得最优解，代理必须在环境的每个时间步长保持安全间隔并解决冲突。为了增加案例研究的难度并提供更真实的环境，飞机随机进入扇区，因此代理需要制定策略，而不仅仅是简单地记忆动作。

3.2. Simulation Settings

3.2. 模拟设置

There are many settings we imposed to make these Case Studies feasible. For each simulation run, there is a fixed max number of aircraft. This is to allow comparable performance between simulation runs and to evaluate the final performance of the model. In BlueSky, the performance metrics of each aircraft type impose different constraints on the range of cruise speeds. We set all aircraft to be the same type, Boeing 747-400, in both Case Studies. We also imposed a setting that all aircraft can not deviate from their route. The final setting in the Case Studies is the desired speed of each aircraft. Each aircraft has the ability to select three different desired speeds: minimum allowed cruise speed, current cruise speed, and maximum allowed cruise speed which is defined in the BlueSky simulation environment.

我们设定了许多条件以使这些案例研究可行。对于每次模拟运行，都有一个固定的最大飞机数量。这是为了允许各次模拟运行之间进行比较，并评估模型的最终性能。在 BlueSky 中，每种飞机类型的性能指标对巡航速度的范围施加了不同的约束。我们在两个案例研究中将所有飞机设置为同一类型，波音 747-400。我们还设定了一个条件，即所有飞机不得偏离其航线。案例研究的最后一个设定是每架飞机的期望速度。每架飞机都有选择三种不同期望速度的能力: 允许的最小巡航速度、当前巡航速度和 BlueSky 模拟环境中定义的允许的最大巡航速度。

3.3. Multi-Agent Reinforcement Learning Formulation

3.3. 多代理强化学习公式化

Here we formulate our BlueSky Case Study as a deep multi-agent reinforcement learning problem by representing each aircraft as an agent and define the state space, action space, termination criteria and reward function for the agents.

在这里，我们将我们的 BlueSky 案例研究公式化为一个深度多代理强化学习问题，通过将每架飞机表示为一个代理，并定义代理的状态空间、动作空间、终止条件和奖励函数。

3.3.1. STATE SPACE

3.3.1. 状态空间

A state contains all the information the AI agent needs to make decisions. Since this is a multi-agent environment, we needed to incorporate communication between the agents. To allow the agents to communicate, the state for a given agent also contains state information from the N -closest agents. We follow a similar state space definition as in (Everett et al., 2018), but instead we use half of the loss of separation distance as the radius of the aircraft. In this way the sum of the radii between two aircraft is equal to the loss of separation distance. The state information includes distance to the goal, aircraft speed, aircraft acceleration, distance to the intersection, a route identifier, and half the loss of separation distance for the N -closest agents, where the position for a given aircraft can be represented as (distance to the goal, route identifier). We also included the distance to the N -closest agents in the state space of the agents and the full loss of separation distance. From this, we can see that the state space for the agents is constant in size, since it only depends on the N -closest agents and does not scale as the number of agents in the environment increase. Fig. 2 shows an example of a state in the BlueSky environment.

状态包含 AI 代理做出决策所需的所有信息。由于这是一个多代理环境，我们需要在代理之间加入通信。为了允许代理之间进行通信，特定代理的状态还包含 N -最近代理的状态信息。我们遵循与 (Everett et al., 2018) 相似的态空间定义，但改为使用分离距离损失的一半作为飞机的半径。这样两个飞机之间的半径之和等于分离距离的损失。状态信息包括距离目标、飞机速度、飞机加速度、距离交叉点、路线标识符以及 N -最近代理的分离距离损失的一半，其中给定飞机的位置可以表示为 (距离目标，路线标识符)。我们还在代理的状态空间中包含了距离 N -最近代理的距离和完整的分离距离损失。由此我们可以看出，代理的状态空间大小是恒定的，因为它只依赖于 N -最近代理，并且不会随着环境中代理数量的增加而扩大。图 2 展示了 BlueSky 环境中的一个状态示例。

We found that defining which N -closest agents to consider is very important to obtain a good result since we do not want to add irrelevant information in the state space. For example, consider Fig. 2. If the ownship is on R_1 and one of the closest aircraft on R_3 has already passed the intersection, there is no reason to include its information in the state space of the ownship. We defined the following rules for the aircraft that are allowed to be in the state of the ownship:

我们发现，定义要考虑的 N -最近代理对于获得良好结果非常重要，因为我们不希望在状态空间中添加无关信息。例如，考虑图 2。如果本机位于 R_1 并且其中一个最近的飞机已经在 R_3 通过了交叉点，那么没有理由将其信息包含在本机的状态空间中。我们定义了以下规则，以确定哪些飞机可以被包含在本机的状态中:

- aircraft on conflicting route must have not reached the intersection
- 处于冲突航线的飞机必须尚未到达交叉点
- aircraft must either be on the same route or on a conflicting route.
- 飞机必须位于相同的航路或者存在冲突的航路上。

By utilizing these rules, we eliminated useless information which we found to be critical in obtaining convergence to this problem.

通过利用这些规则，我们消除了在获得该问题收敛性时发现的无用信息，这对我们来说至关重要。

Figure 2. BlueSky sector designed for our Case Study. Shown is Case Study 1 for illustration: there are three routes, R_1, R_2 , and R_3 , along with two intersections, I_1 and I_2 .

图 2. 为我们的案例研究设计的 BlueSky 区域。所示为案例研究 1 的示例: 存在三条航路 R_1, R_2 和 R_3 ，以及两个交叉点 I_1 和 I_2 。

If we consider Fig. 2 as an example, we can acquire all of the state information we need from the aircraft. If we let $I^{(i)}$ represent the distance to the goal, aircraft speed, aircraft acceleration, distance to the intersection, route identifier, and half the loss of separation distance of aircraft i , the state will be represented as follows:

如果我们将图 2 作为示例，我们可以从飞机获取我们需要的所有状态信息。如果我们让 $I^{(i)}$ 代表到目标点的距离，飞机速度，飞机加速度，到交叉点的距离，航路识别符，以及飞机 i 分离距离损失的一半，状态将表示如下:

$$s_t^o = \Big(I^{(o)}, d^{(1)}, \mathrm{LOS}\,(o,1)\,, d^{(2)}, \mathrm{LOS}\,(o,2) \ldots, d^{(n)},$$

$$\mathrm{LOS}\,(o,n)\,, I^{(1)}, I^{(2)}, \ldots, I^{(n)}),$$

where s_t^o represents the ownship state, $d^{(i)}$ represents the distance from ownship to aircraft $i, \mathrm{LOS}\,(o,i)$ represents the loss of separation distance between aircraft o and aircraft i , and n represents the number of closest aircraft to include in the state of each agent. By defining the loss of separation distance between two aircraft in the state space, the agents should be able to develop a strategy for non-uniform loss of separation requirements for different aircraft types. In this work we consider the standard uniform loss of separation requirements and look to explore this idea in future work.

其中 s_t^o 代表自身飞机的状态，$d^{(i)}$ 代表从自身飞机到其他飞机的距离，$i, \mathrm{LOS}\,(o,i)$ 代表两架飞机之间的分离距离损失，o 代表其他飞机，i ，以及 n 代表每个代理状态中包含的最近飞机的数量。通过在状态空间中定义两架飞机之间的分离距离损失，代理应该能够制定出针对不同类型飞机的非均匀分离距离要求的策略。在这项工作中，我们考虑了标准的均匀分离距离要求，并计划在未来的工作中探索这个想法。

3.3.2. ACTION SPACE

3.3.2. 动作空间

All agents decide to change or maintain their desired speed every 12 seconds in simulation time. The action space for the agents can be defined as follows:

所有代理在模拟时间的每 12 秒决定改变或维持它们的期望速度。代理的动作空间可以定义如下:

$$A_t = [v_{\min}, v_{t-1}, v_{\max}]$$

where $v_{\min}$ is the minimum allowed cruise speed (decelerate), v_{t-1} is the current speed of the aircraft (hold), and $v_{\max}$ is the maximum allowed cruise speed (accelerate).

其中 $v_{\min}$ 是允许的最小巡航速度 (减速)，v_{t-1} 是飞机的当前速度 (保持)，$v_{\max}$ 是允许的最大巡航速度 (加速)。

3.3.3. TERMINAL STATE

3.3.3. 终止状态

Termination in the episode was achieved when all aircraft had exited the sector:

当所有飞机都已离开该区域时，该情节的终止得以实现:

$$N_{\text{aircraft}} = 0$$

3.3.4. REWARD FUNCTION

3.3.4. 奖励函数

The reward function for the agents were all identical, but locally applied to encourage cooperation between the agents. If two agents were in conflict, they would both receive a penalty, but the remaining agents

that were not in conflict would not receive a penalty. Here a conflict is defined as the distance between any two aircraft is less than 3nmi . The reward needed to be designed to reflect the goal of this paper: safe separation and conflict resolution. We were able to capture our goals in the following reward function for the agents:

代理的奖励函数都是相同的，但是局部应用以鼓励代理之间的合作。如果有两个代理发生冲突，它们都会受到惩罚，但未发生冲突的其余代理不会受到惩罚。这里的冲突定义为任意两架飞机之间的距离小于 3nmi 。奖励的设计需要反映本文的目标: 安全分离和冲突解决。我们能够在以下代理的奖励函数中捕捉到我们的目标:

$$r_t = \begin{cases} -1 & \text{if } d_o^c < 3 \\ -\alpha + \beta \cdot d_o^c & \text{if } d_o^c < 10 \text{ and } d_o^c \geq 3, \\ 0 & \text{otherwise} \end{cases}$$

where d_o^c is the distance from the ownship to the closest aircraft in nautical miles, and α and β are small, positive constants to penalize agents as they approach the loss of separation distance. By defining the reward to reflect the distance to the closest aircraft, this allows the agent to learn to select actions to maintain safe separation requirements.

其中 d_o^c 是本机到最近飞机的距离，单位为海里，α 和 β 是小的正常数，用以在代理接近失去分离距离时对它们进行惩罚。通过定义奖励以反映到最近飞机的距离，这允许代理学会选择行动以保持安全分离要求。

4. Solution Approach

4. 解决方案方法

To solve the BlueSky Case Studies, we designed and developed a novel deep multi-agent reinforcement learning framework called the Deep Distributed Multi-Agent Reinforcement Learning framework (DD-MARL). In this section, we introduce and describe the framework, then we explain why this framework is needed to solve this Case Study.

为了解决 BlueSky 案例研究，我们设计并开发了一种新颖的深度多代理强化学习框架，称为深度分布式多代理强化学习框架 (DD-MARL)。在本节中，我们介绍并描述该框架，然后解释为什么需要这个框架来解决本案例研究。

To formulate this environment as a deep multi-agent reinforcement learning problem, we utilized a centralized learning with decentralized execution framework with one neural network where the actor and critic share layers of same the neural network, further reducing the number of trainable parameters. By using one neural network, we can train a model that improves the joint expected return of all agents in the sector, which encourages cooperation between the agents. We utilized the synchronous version of A3C, A2C (advantage actor critic) which is shown to achieve the same or better performance as compared to the asynchronous version (Schulman et al., 2017). We also adapted the A2C algorithm to incorporate the PPO loss function defined in (6), which we found led to a more stable policy and resulted in better final performance. We follow a similar approach to (Everett et al., 2018) to split the state into the two parts: ownship state information and all other information, which we call the local state information, s^{local} . We then encode the local state information using a fully connected layer before combining the encoded state with the ownship state information. From there, the combined state is sent through

为了将这个环境构建为一个深度多智能体强化学习问题，我们采用了一个集中学习与去中心执行框架，使用一个神经网络，其中演员 (actor) 和评论家 (critic) 共享相同神经网络的层，进一步减少了可训练参数的数量。通过使用一个神经网络，我们可以训练一个模型，以提高该区域所有智能体的联合期望回报，从而促进智能体之间的协作。我们使用了同步版本的 A3C，即 A2C(优势演员评论家)，该算法相较于异步版本 (Schulman 等人，2017 年) 展现出相同或更好的性能。我们还调整了 A2C 算法，以纳入 (6) 中定义的 PPO 损失函数，我们发现这导致了更稳定的策略，并带来了更好的最终性能。我们遵循 (Everett 等人，2018 年) 的类似方法，将状态分为两部分: 自身状态信息和其他所有信息，我们称之为本地状态信息 s^{local} 。然后我们使用一个全连接层对本地状态信息进行编码，之后将编码状态与自身状态信息相结合。从那里开始，组合状态被送入

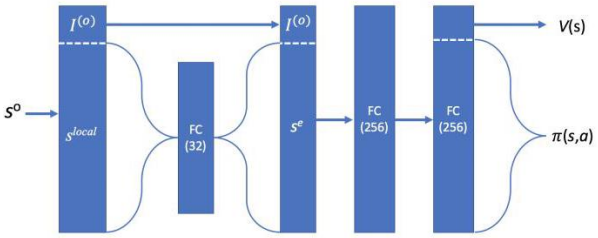

Figure 3. Illustration of the neural network architecture for A2C with shared layers between the actor and critic. Each hidden layer is a fully connected (FC) layer with 32 nodes for the encoded state and 256 nodes for the last two layers.

图 3. 展示了带有共享层的 A2C 神经网络架构。每个隐藏层都是一个全连接 (FC) 层，编码状态有 32 个节点，最后两层有 256 个节点。

two fully connected layers and produces two outputs: the policy and value for a given state. Fig. 3 shows an illustration of the the neural network architecture. With this framework, we can implement the neural network to all aircraft, instead of having a specified neural network for all individual aircraft. In this way, the neural network is acting as a centralized learner and distributing knowledge to each aircraft. The neural network's policy is distributed at the beginning of each episode and updated at the end of each episode which reduces the amount of information that is sent to each aircraft, since sending an updated model during the route could be computationally expensive. In this formulation, each agent has identical neural networks, but since they are evolving different states their actions can be different.

两个全连接层并产生两个输出: 给定状态下的策略和价值。图 3 展示了神经网络架构的示意图。使用这个框架，我们可以将神经网络应用到所有飞机上，而不是为每架飞机指定一个专用的神经网络。这样，神经网络充当集中学习者并将知识分发给每架飞机。神经网络的策略在每轮开始时分发，并在每轮结束时更新，这减少了发送到每架飞机的信息量，因为在飞行途中发送更新的模型可能是计算上昂贵的。在这个公式中，每个代理拥有相同的神经网络，但由于它们处于不同的状态，它们的行动可能有所不同。

It is also important to note that this framework is invariant to the number of aircraft. When observing an en-route sector, aircraft are entering and exiting which creates a dynamic environment with varying number of aircraft. Since our approach does not depend on the number of aircraft, our framework can handle any number of aircraft arriving based on stochastic inter-arrival times.

还需要注意的是，这个框架对飞机的数量是不变的。在观察航路扇区时，飞机正在进入和退出，这创造了一个动态环境，飞机数量在变化。由于我们的方法不依赖于飞机的数量，我们的框架可以处理任何数量的基于随机到达时间的飞机到达。

5. Numerical Experiments

5. 数值实验

5.1. Interface

5.1. 接口

To test the performance of our proposed framework, we utilized the BlueSky air traffic control simulator. This simulator is built around python so we were able to quickly obtain the state space information of all aircraft [1] . By design, when restarting the simulation, all objectives were the same: maintain safe separation and sequencing, resolve conflicts, and minimize delay. Aircraft initial positions and available speed changes did not change between simulation runs.

为了测试我们提出框架的性能，我们使用了 BlueSky 空中交通控制模拟器。这个模拟器是基于 python 构建的，因此我们能够快速获取所有飞机的状态空间信息 [1] 。按照设计，在重新启动模拟时，所有目标都是相同的: 保持安全间隔和序列，解决冲突，并最小化延误。飞机的初始位置和可用速度变化在模拟运行之间没有改变。

[1] Code will be made available at https://github.com/marcbrittain
[1] 代码将会在 https://github.com/marcbrittain 上提供。

5.2. Environment Setting

5.2. 环境设置

For each simulation run in BlueSky, we discretized the environment into episodes, where each run through the simulation counted as one episode. We also introduced a time-step, Δt , so that after the agents selected an action, the environment would evolve for Δt seconds until a new action was selected. We set Δt to 12 seconds to allow for a noticeable change in state from $s_t \rightarrow s_{t+1}$ and to check the safe-separation requirements at regular intervals.

在 BlueSky 中的每次模拟运行中，我们将环境离散化为若干个情节，每次运行模拟过程计为一个情节。我们还引入了时间步长 Δt ，以便在代理选择一个动作后，环境会进化 Δt 秒钟直到选择新的动作。我们将 Δt 设为 12 秒，以允许从 $s_t \rightarrow s_{t+1}$ 状态中观察到明显的变化，并定期检查安全间隔要求。

There were many different parameters that needed to be tuned and selected for the Case Studies. We implemented the adapted A2C concept mentioned earlier, with two hidden layers consisting of 256 nodes. The encoding layer for the N -closest aircraft state information consisted of 32 nodes and we used the ReLU activation function for all hidden layers. The output of the actor used a Softmax activation function and the output of the critic used a Linear activation function. Other key parameter values included: learning rate $lr = 0.0001, \gamma = 0.99, \epsilon = 0.2, \alpha = 0.1, \beta = 0.005$, and we used the Adam optimizer for both the actor and critic loss (Kingma and Ba, 2014).

对于案例研究，需要调整和选择许多不同的参数。我们实施了之前提到的适应型 A2C 概念，包含两个隐藏层，每层 256 个节点。用于编码 N -最近飞行器状态信息的编码层包含 32 个节点，并且我们为所有隐藏层使用了 ReLU 激活函数。演员 (actor) 的输出使用 Softmax 激活函数，评论家 (critic) 的输出使用线性激活函数。其他关键参数值包括: 学习率 $lr = 0.0001, \gamma = 0.99, \epsilon = 0.2, \alpha = 0.1, \beta = 0.005$ ，并且我们为演员和评论家的损失函数使用了 Adam 优化器 (Kingma 和 Ba，2014 年)。

5.3. Case Study 1: Three routes with two intersections

5.3. 案例研究 1: 三条路线与两个交叉点

In this Case Study, we considered three routes with two intersections as shown in Fig. 2. In our DD-MARL framework, the single neural network is implemented on each aircraft as they enter the sector. Each agent is then able to select its own desired speed which greatly increases the complexity of this problem since the agents need to learn how to cooperate in order to maintain safe-separation requirements. What also makes this problem interesting is that each agent does not have a complete representation of the state space since only the ownship (any given agent) state information and the N -closest agent state information are included.

在这个案例研究中，我们考虑了如图 2 所示的三条路线和两个交叉点。在我们的 DD-MARL 框架中，每个飞行器进入扇区时都会实施单个神经网络。每个代理都能够选择自己的期望速度，这极大地增加了问题的复杂性，因为代理需要学习如何合作以维持安全间隔要求。使这个问题变得有趣的是，每个代理都没有完整的状态空间表示，因为只包含了自身的 (任何给定代理) 状态信息和 N -最近代理的状态信息。

5.4. Case Study 2: Two routes with one merging point

5.4. 案例研究 2: 两条路线与一个合并点

This Case Study consisted of two routes merging to one single point and then following one route thereafter (see Fig. 4. This poses another critical challenge for an autonomous air traffic control system that is not present in Case Study 1: merging. When merging to a single point, safe separation requirements need to be maintain both before the merging point and after. This is particularly challenging since there is high density traffic on each route that is now combining to one route. Agents need to carefully coordinate in order to maintain safe separation requirements after the merge point.

本案例研究包括两条航线合并到一个单一节点，然后继续遵循一条航线 (见图 4)。这对于自主空中交通控制系统来说提出了另一个关键挑战，这在案例研究 1 中是不存在的: 合并。在合并到一个单一节点时，需要在合并点之前和之后都保持安全间隔要求。这尤其具有挑战性，因为现在每条航线上的高密度交通都合并到了一条航线上。代理需要仔细协调以保持合并点之后的安全间隔要求。

In both Case Studies there were 30 total aircraft that entered the airspace following a uniform distribution over 4,5, and 6 minutes. This is an extremely difficult problem to solve because the agents cannot simply memorize actions, the agents need to develop a strategy in order to solve the problem. We also included the 3 closest agents state information in the state of the ownship. All other agents are not included in the state of the ownship. The episode terminated when all 30 aircraft had exited the sector, so the optimal solution in this problem is 30 goals achieved. Here we define goal achieved as an aircraft exiting it the sector without conflict.

在两个案例研究中，共有 30 架飞机在 4、5 和 6 分钟内以均匀分布进入空域。这是一个极其困难的问题，因为代理不能简单地记住动作，代理需要制定策略来解决问题。我们还将在状态中包括自身最近的 3 个代理的状态信息。所有其他代理不包括在自身状态中。当所有 30 架飞机都离开该区域时，这一集结束，所以这个问题中的最优解是实现 30 个目标。在这里，我们将目标实现定义为飞机在没有冲突的情况下离开该区域。

Figure 4. Case Study 2: two routes, R_1 and R_2 merge to a single route at M_1 .
图 4。案例研究 2: 两条航线，R_1 和 R_2 在 M_1 处合并为一条航线。
Table 1. Performance of the policy tested for 200 episodes.
表 1。在 200 个情节中测试的策略性能。

CASE STUDY	MEAN	MEDIAN
1	29.99 ± 0.141	30
2	30	30

案例研究	MEAN	中位数
1	29.99 ± 0.141	30
2	30	30

5.5. Algorithm Performance

5.5. 算法性能

In this section, we analyze the performance of DD-MARL on the Case Studies. We allowed the AI agents to train for 20,000 episodes and 5,000 episodes for Case Study 1 and Case Study 2, resepctively. We then evaluated the final policy for 200 episodes to calculate the mean and standard deviation along with the median to evaluate the performance of the final policy as shown in Table 1^2 . We can see from Fig. 5 that for Case Study 1, the policy began converging to a good policy by around episode 7,500, then began to further refine to a near optimal policy for the remaining episodes. For Case Study 2, we can see from Fig. 6 that a near optimal policy was obtained in only 2,000 episodes and continued to improve through the remainder of the 3,000 episodes. Training for only 20,000 episodes (as required in Case Study 1) is computationally inexpensive as it equates to less than 4 days of training. We suspect that this is due to the approach of distributing one neural network to all aircraft and by allowing shared layers between the actor and critic.

在本节中，我们分析了 DD-MARL 在案例研究中的性能。我们让 AI 代理分别在案例研究 1 和案例研究 2 中进行 20,000 个回合和 5,000 个回合的训练。然后，我们评估了最终策略 200 个回合，以计算均值、标准差以及中位数，以评估如表 1^2 所示的最终策略的性能。从图 5 中我们可以看到，对于案例研究 1，策略在大约 7,500 回合时开始收敛到良好的策略，然后开始进一步细化，在剩余回合中接近最优策略。对于案例研究 2，从图 6 中我们可以看到，在仅 2,000 个回合就获得了接近最优的策略，并在剩余的

3,000 个回合中继续改进。仅训练 20,000 个回合 (如案例研究 1 所要求) 在计算上是低成本的，因为它相当于不到 4 天的训练时间。我们怀疑这是由于将一个神经网络分布到所有飞机上，并允许演员和评论家之间共享层的做法所致。

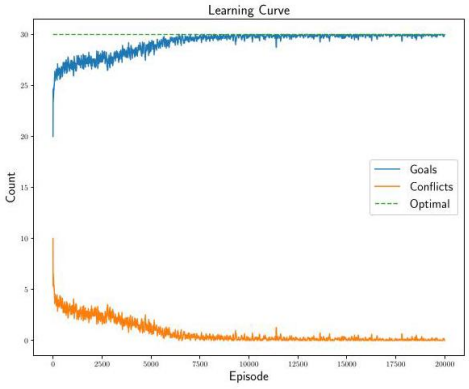

Figure 5. Learning curve of the DD-MARL framework for Case Study 1. Results are smoothed with a 30 episode rolling average for clarity.

图 5. DD-MARL 框架在案例研究 1 中的学习曲线。结果通过 30 个回合的滚动平均值进行平滑处理以提高清晰度。

We can see from Table 1, that on average we obtained a score of 29.99 throughout the 200 episode testing phase for Case Study 1 and 30 for Case Study 2. This equates to resolving conflict 99.97% at the intersections, and 100% at the merging point. Given that this is a stochastic environment, we speculate that there could be cases where there is an orientation of aircraft where the 3nmi loss of separation distance can not be achieved, and in such cases we would alert human ATC to resolve this type of conflict. The median score removes any outliers from our testing phase and we can see the median score is optimal for both Case Study 1 and Case Study 2.

从表 1 中我们可以看到，在案例研究 1 的 200 个测试阶段中，我们平均获得了 29.99 分的成绩，案例研究 2 中为 30 分。这等同于在交叉点解决冲突 99.97% ，以及在合并点 100% 解决冲突。鉴于这是一个随机环境，我们推测可能存在飞机方向使得 3nmi 分离距离无法达成，在这种情况下我们会提醒人类空中交通管制员解决此类冲突。中位数得分排除了测试阶段的任何异常值，我们可以看到中位数得分对于案例研究 1 和案例研究 2 都是最优的。

6. Conclusion

6. 结论

We built an autonomous air traffic controller to ensure safe separation between aircraft in high-density en-route airspace sector. The problem is formulated as a deep multi-agent reinforcement learning problem with the actions of selecting desired aircraft speed. The problem is then solved by using the DD-MARL framework, which is shown to be capable of solving complex sequential decision making problems under uncertainty. According to our knowledge, the major contribution of this research is that we are the first research group to investigate the feasibility and performance of autonomous air traffic control with a deep multi-agent reinforcement learning framework to enable an automated, safe and efficient en-route airspace. The promising results from our numerical experiments encourage us to conduct future work on more complex sectors. We will also investigate the feasibility of the autonomous air traffic controller to replace human air traffic controllers in ultra dense, dynamic and complex airspace in the future.

我们构建了一个自主空中交通管制系统，以确保在高密度航路空域中飞机之间的安全间隔。问题被构建为一个深度多智能体强化学习问题，其动作是选择期望的飞机速度。然后，使用 DD-MARL 框架解决这个问题，该框架已被证明能够解决在不确定性下的复杂序列决策问题。据我们所知，本研究的主要贡献是，我们是第一个研究小组探讨使用深度多智能体强化学习框架来自主控制空中交通的可行性和性能，

[2] A video of the final converged policy can be found at https://www.youtube.com/watch?v=sjRGjiRZWxg and
[2] 最终收敛策略的视频可以在 https://www.youtube.com/watch?v=sjRGjiRZWxg 找到 https://youtu.be/NvLxTJNd-q0

以实现自动化、安全和高效的航路空域。我们数值实验的积极结果鼓励我们在更复杂的空域进行未来的工作。我们还将研究自主空中交通管制系统在未来替代人类空中交通管制员在超密集、动态和复杂空域中的可行性。

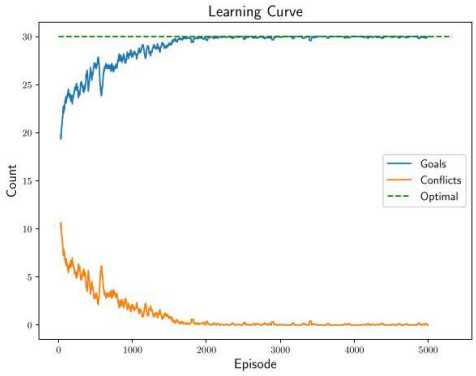

Figure 6. Learning curve of the DD-MARL framework for Case Study 2. Results are smoothed with a 30 episode rolling average for clarity.

图 6. 案例研究 2 中 DD-MARL 框架的学习曲线。结果通过 30 个测试阶段的滚动平均值进行平滑处理以提高清晰度。

Acknowledgements

致谢

We would like to thank Xuxi Yang, Guodong Zhu, Josh Bertram, Priyank Pradeep, and Xufang Zheng, whose input and discussion helped in the success of this work.

我们要感谢杨绪熙、朱国栋、Josh Bertram、Priyank Pradeep 和郑徐芳，他们的意见和建议有助于这项工作的成功。

References

参考文献

Air, A. P. Revising the airspace model for the safe integration of small unmanned aircraft systems. Amazon Prime Air, 2015.

Air, A. P. 修订空域模型以保障小型无人机系统的安全整合。Amazon Prime Air, 2015。

Airbus. Blueprint for the sky, 2018. URL

空中客车公司。天空蓝图，2018。URLhttps://storage.googleapis.com/

blueprint/Airbus_UTM_Blueprint.pdf.

blueprint/Airbus_UTM_Blueprint.pdf。

Amato, C. and Shani, G. High-level reinforcement learning in strategy games. In Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1, pp. 75-82. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

Amato, C. 和 Shani, G. 在策略游戏中进行高级强化学习。在《第 9 届国际自主代理和多代理系统会议论文集: 第 1 卷-第 1 卷》中，第 75-82 页。国际自主代理和多代理系统基金会，2010 年。

Baxley, B. T., Johnson, W. C., Scardina, J., and Shay, R. F. Air traffic management technology demonstration-1 concept of operations (atd-1 conops), version 3.0. 2016.

Baxley, B. T.，Johnson, W. C.，Scardina, J. 和 Shay, R. F. 空中交通管理技术演示-1 概念运作 (ATD-1 CONOPS)，版本 3.0。2016 年。

Brittain, M. and Wei, P. Autonomous aircraft sequencing and separation with hierarchical deep reinforcement learning. In Proceedings of the International Conference for Research in Air Transportation, 2018.

Brittain, M. 和 Wei, P. 使用分层深度强化学习进行自主飞机排序和分离。在《国际航空运输研究会议论文集》中，2018 年。

Bu, L., Babu, R., De Schutter, B., et al. A comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 38(2):156-172, 2008.

Bu, L., Babu, R., De Schutter, B. 等人。多代理强化学习的全面调查。IEEE 系统、人与网络安全学杂志，第 C 部分 (应用与评论)，38(2):156-172，2008 年。

Chen, Y. F., Liu, M., Everett, M., and How, J. P. Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In 2017 IEEE international conference on robotics and automation (ICRA), pp. 285-292. IEEE, 2017.

Council, N. R. et al. Autonomy research for civil aviation: toward a new era of flight. National Academies Press, 2014.

Erzberger, H. Automated conflict resolution for air traffic control. 2005.

Erzberger, H. and Heere, K. Algorithm and operational concept for resolving short-range conflicts. Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 224(2):225-243, 2010.

Erzberger, H. and Itoh, E. Design principles and algorithms for air traffic arrival scheduling. 2014.

Everett, M., Chen, Y. F., and How, J. P. Motion planning among dynamic, decision-making agents with deep reinforcement learning. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3052-3059. IEEE, 2018.

Farley, T. and Erzberger, H. Fast-time simulation evaluation of a conflict resolution algorithm under high air traffic demand. In 7th USA/Europe ATM 2007 R&D Seminar, 2007.

Genders, W. and Razavi, S. Using a deep reinforcement learning agent for traffic signal control. arXiv preprint arXiv:1611.01142, 2016.

Google. Google uas airspace system overview, 2015. URL https://utm.arc.nasa.gov/docs/ GoogleUASAirspaceSystemOverview5 \pager [1]. pdf.

Hoekstra, J. M. and Ellerbroek, J. Bluesky atc simulator project: an open data and open source approach. In Proceedings of the 7th International Conference on Research in Air Transportation, pp. 1-8. FAA/Eurocontrol USA/Europe, 2016.

Holden, J. and Goel, N. Fast-forwarding to a future of on-demand urban air transportation. San Francisco, CA, 2016.

Hunter, G. and Wei, P. Service-oriented separation assurance for small uas traffic management. In Integrated Communications Navigation and Surveillance (ICNS) Conference, Herndon, VA, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

Kopardekar, P., Rios, J., Prevot, T., Johnson, M., Jung, J., and Robinson, J. E. Unmanned aircraft system traffic management (utm) concept of operations. 2016.

Kopardekar, P. H. Safely enabling civilian unmanned aerial system (uas) operations in low-altitude airspace by unmanned aerial system traffic management (utm). 2015.

Liang, X., Du, X., Wang, G., and Han, Z. Deep reinforcement learning for traffic light control in vehicular networks. arXiv preprint arXiv:1803.11115, 2018.

Matignon, L., Laurent, G. J., and Le Fort-Piat, N. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. The Knowledge Engineering Review, 27(1):1-31, 2012.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602, 2013.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In International conference on machine learning, pp. 1928- 1937, 2016.

Mueller, E. R., Kopardekar, P. H., and Goodrich, K. H. Enabling airspace integration for high-density on-demand mobility operations. In 17th AIAA Aviation Technology, Integration, and Operations Conference, pp. 3086, 2017.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

Silver, D. and Hassabis, D. Alphago: Mastering the ancient game of go with machine learning. Research Blog, 9, 2016.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. nature, 529(7587):484, 2016.

Tan, M. Multi-agent reinforcement learning: Independent vs. cooperative agents. In Proceedings of the tenth international conference on machine learning, pp. 330-337, 1993.

Uber. Uber elevate - the future of urban air transport, 2018. URL https://www.uber.com/info/elevate.

Undertaking, S. J. U-space blueprint. SESAR Joint Undertaking. Accessed September, 18, 2017.

Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhn-evets, A. S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., et al. Starcraft ii: A new challenge for reinforcement learning. arXiv preprint arXiv:1708.04782, 2017.

Wollkind, S., Valasek, J., and Ioerger, T. Automated conflict resolution for air traffic management using cooperative multiagent negotiation. In AIAA Guidance, Navigation, and Control Conference and Exhibit, pp. 4992, 2004.