2024

EFFICIENT CONNECTIVITY MANAGEMENT AND PATH PLANNING FOR IOT AND UAV NETWORKS

Amirahmad Chapnevis
Virginia Commonwealth University

EFFICIENT CONNECTIVITY MANAGEMENT AND PATH PLANNING FOR IOT AND UAV NETWORKS

A Proposal submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

by

AMIRAHMAD CHAPNEVIS

Doctorate in Computer Science - 2019-2024

Director: Dr. Eyuphan Bulut,

Associate Professor, Department of Computer Science

Virginia Commonwealth University

Richmond, Virginia

May, 2024

Acknowledgements

I want to express my gratitude to my advisor, Dr. Eyuphan Bulut, who has played a significant role and provided guidance throughout my doctoral studies. I extend my sincere appreciation to Dr. Kostadin Damevski, Dr. Changqing Luo, Dr. Ismail Guvenc, and Dr. Arupjyoti Bhuyan for graciously agreeing to be a part of my committee. I am also grateful for the continuous support of my family. Lastly, I appreciate the impact of individuals in my life who have contributed to shaping my journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract

EFFICIENT CONNECTIVITY MANAGEMENT AND PATH PLANNING FOR IOT AND UAV NETWORKS

By Amirahmad Chapnevis

A Proposal submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

Virginia Commonwealth University, 2024.

Director: Dr. Eyuphan Bulut,

Associate Professor, Department of Computer Science

This dissertation explores how to better manage resources in mobile networks, especially for enhancing the performance of Unmanned Aerial Vehicles (UAV)-supported IoT networks. We explored ways to set up a flexible communication architecture that can handle large IoT deployments by making good use of mobile core network resources like bearers and data paths. We developed strategies that meet the needs of IoT networks and enhance network performance. We also developed and tested a system that combines traffic from several mobile devices that use the same user identity and network resources within the core mobile network. We used everyday smartphones, SIM cards, and the Amarisoft Callbox, which includes core network and eNB components, for our tests.

UAVs have changed many fields due to their flexibility and versatility. This dissertation looks at how UAVs and IoT can work together, addressing important challenges to make systems work better, increase efficiency, and guarantee strong communication between UAVs and ground control stations. Collecting data efficiently from

ground sensors and IoT networks is crucial, and our research is centered on planning paths that make UAVs as efficient as possible in this process. We use UAVs as relay nodes, optimizing their paths and flight plans to reduce delays and make sure data is collected and delivered on time. Additionally, we introduced a new way to route UAVs, taking into account the Age of Information (AoI) concept. We calculate AoI from when data is generated to when it is delivered through cellular-connected UAVs, making mission time as short as possible while keeping UAV connectivity. Our tests show that our heuristic approach works well in different scenarios. Utilizing UAVs as relays to facilitate communication reduces mission time and accelerates IoT data delivery, presenting an innovative advantage over alternative methods. In solving each problem, we first use an Integer Linear Programming (ILP) solution and then introduce a faster heuristic algorithm to save time. This combination ensures strong solutions while also providing quicker computational results.

CHAPTER 1

INTRODUCTION

UAVs have revolutionized numerous domains, including wireless communication, agriculture, and search and rescue operations [1]. These agile flying machines offer immense possibilities when combined with the power of the IoT [2]. In this dissertation, we embark on a journey to explore the dynamic relationship between UAVs and IoT, addressing key challenges to optimize system performance, improve efficiency, and ensure reliable communication between UAVs and ground base stations (GBSs) [3]. By delving into the interconnected domains of path planning, resource management, and collaborative connectivity maintenance, we aim to unlock the true potential of UAV-assisted IoT networks, ushering in a new era of possibilities and advancements.

In addition, resource management within the mobile core network plays a pivotal role in enhancing the performance of UAV-assisted IoT networks [4]. As the IoT ecosystem expands, mobile network operators face the challenge of optimizing resource utilization while addressing the limitations of wireless spectrum availability [5]. Our research focuses on harnessing the potential of mobile core network resources, such as bearers and data paths in the Evolved Packet Core (EPC), to establish a scalable communication architecture for massive IoT deployments. By smartly grouping machine-type devices (MTDs) and optimizing traffic patterns, we strive to enhance efficiency and ensure reliable connectivity. Our resource management strategies cater to the unique demands and constraints of IoT networks, allowing for effective resource allocation and improved network performance.

Efficiently collecting data from ground sensor nodes and IoT networks using

UAVs is crucial in various applications [6]. However, achieving optimal data collection poses unique challenges. In this research, we focus on path planning strategies that maximize the efficiency of UAVs during the data collection process. By carefully considering factors such as battery capacity, flight duration, and data generation times at IoT devices, we aim to minimize delays and ensure timely data collection. Our innovative approach integrates UAVs as relay nodes, collecting data from IoT devices and delivering it to base stations for further transmission. Through meticulous mission planning, we optimize the path and flight trajectory of UAVs, guaranteeing efficient data collection and delivery.

Furthermore, ensuring continuous and reliable connectivity between UAVs and ground control stations is vital for the success of UAV-assisted IoT networks [7, 8, 9]. In this dissertation, we explore collaborative connectivity maintenance strategies that leverage cellular connections to enhance the communication capabilities of UAVs. While traditional UAV control systems rely on direct LoS communication, our approach taps into the potential of cellular networks. We address the challenge of limited GBS coverage by fostering collaboration between UAVs. Through resource sharing and a multi-hop approach, UAVs support each other, mitigating cellular outage durations and ensuring seamless communication. Our collaborative connectivity framework optimizes mission completion time and guarantees reliable communication, even in areas with limited GBS coverage.

In conclusion, this dissertation represents a comprehensive exploration of UAV-assisted IoT networks, with a focus on path planning, resource management, and collaborative connectivity maintenance. By addressing these interconnected challenges, we strive to optimize system performance, improve efficiency, and enable reliable communication between UAVs and ground control stations. Through innovative approaches and practical solutions, we aim to unlock the true potential of UAV-assisted

Fig. 1. Collaborative UAV control systems establish scalable communication for massive IoT deployments

IoT networks, paving the way for transformative applications in various domains. This research not only contributes to the advancement of UAV technology but also drives the evolution of IoT networks, fostering a future where UAVs and IoT seamlessly integrate to create a more connected and efficient world.

1.1 Industry Interest

Telecommunication providers are increasingly recognizing the potential of UAVs to enhance network coverage and connectivity, particularly in challenging or remote areas [10]. By deploying UAVs equipped with communication equipment, such as small cellular base stations or relays, telecommunications companies can extend network coverage to serve regions or areas affected by natural disasters. UAVs can quickly and flexibly establish temporary communication links, providing essential connectivity for emergency response efforts or supporting ongoing operations in remote locations [11].

Integration with IoT networks offers significant advantages in the telecommunications industry [12]. With the proliferation of IoT devices, ranging from smart homes [13, 14] to industrial sensors, the demand for seamless connectivity and reliable data transmission has grown exponentially. UAVs integrated into the IoT ecosystem can serve as aerial nodes, facilitating communication between IoT devices and the core network. This integration enables efficient and scalable connectivity for a vast number of IoT devices, ensuring reliable data transmission, monitoring, and control [15].

Moreover, UAVs can act as flying relays, extending network coverage to areas with weak signals or gaps in traditional infrastructure. By strategically positioning UAVs, telecommunications companies can overcome obstacles such as physical terrain, geographical barriers, or temporary events that may hinder network connectivity [16]. UAVs equipped with communication equipment can establish temporary communication links, bridging the connectivity gap and providing seamless coverage for IoT devices in those areas.

In addition, the use of UAVs and IoT networks in environmental monitoring has gained significant interest among environmental agencies [17], conservation organizations, and researchers. UAVs equipped with advanced sensors, such as multispectral or hyperspectral cameras, can capture high-resolution aerial imagery and collect data on various environmental parameters. For instance, in air quality monitoring, UAVs can measure pollutant concentrations in real-time, providing valuable insights into pollution sources, distribution patterns [18], and potential health risks.

Integration with IoT networks offers enhanced capabilities for environmental monitoring. UAVs can serve as data collection nodes, transmitting environmental data to centralized servers or cloud platforms wirelessly [17]. This real-time data transmission enables prompt analysis, interpretation, and decision-making, empowering environmental agencies to take timely action in response to environmental chal-

lenges. Additionally, UAVs can be integrated with ground-based IoT sensors to create a comprehensive monitoring network, combining aerial and terrestrial data for a more holistic understanding of environmental conditions.

By leveraging the power of IoT networks, environmental monitoring efforts can be streamlined and automated. UAVs can be deployed to monitor large areas efficiently, collecting data from multiple locations simultaneously. The integration with IoT networks allows for data fusion and analysis, enabling the identification of trends, patterns, and anomalies in environmental parameters. This integrated approach not only improves the accuracy and efficiency of environmental monitoring but also supports timely interventions and conservation strategies.

1.2 Dissertation Organization

The remainder of this dissertation is organized as follows:

1. Literature Review: In Chapter 2, we begin by reviewing the existing literature related to UAV path trajectory findings, resource management in cellular networks, and the AoI area. This chapter provides a comprehensive overview of the current state of research in these areas, setting the stage for our proposed solutions.

2. Resource Management in Mobile Core Networks: Chapter 3 presents our proposed approach for managing resources within mobile core networks. We focus on leveraging mobile core network resources, such as bearers or data paths in the EPC, to establish a scalable communication architecture for massive IoT deployments. Our research explores the concept of connecting MTDs to local gateway devices, reducing resource usage in the core network. By smartly grouping devices and optimizing traffic patterns, we enhance efficiency while

maintaining reliable connectivity. This chapter investigates resource management strategies that foster more effective utilization of available resources.

3. Collaborative Connectivity for UAVs: In Chapter 4, we introduce our approach to designing paths for UAVs with collaboration while maintaining connectivity to GBSs. UAVs present considerable challenges, especially given the limited coverage of GBSs. Our research explores the concept of collaboration between UAVs to overcome these obstacles and enhance connectivity. By leveraging the power of shared resources and adopting a multi-hop approach, UAVs can support each other and avoid cellular outage durations. This collaborative framework optimizes mission completion time and ensures reliable communication, even in areas with limited GBS coverage.

4. Data Collection in UAV-Assisted IoT Networks: Chapter 5 delves into data collection from ground sensor nodes or IoT networks using UAVs. We consider a realistic scenario where UAVs act as relay nodes, collecting data from IoT devices and delivering it to base stations for further transmission. In this chapter, we introduce both an ILP based approach and a greedy heuristic-based solution that factor in data generation times and the locations of IoT devices to minimize the maximum AoI while optimizing mission time and path length. These approaches ensure timely data collection and efficient delivery, considering the inherent constraints of UAV operations.

5. AoI in Mesh Networks: Chapter 6 delves into exploring the path planning problem for a UAV mesh network that collects data from ground IoT devices considering the minimization of the maximum age of information. We have studied several scenarios where the data delivery to the backhaul happens through satellite connection as well as through a few existing base stations in the area.

Through simulations, we have shown that the results in different scenarios are optimal and they have pros and cons to one another.

6. Concluding Remarks: Finally, in Chapter 7, we provide concluding remarks that summarize the key findings of our research, highlight the contributions made, and discuss potential avenues for future work.

By organizing the dissertation in this manner, we provide a comprehensive exploration of UAV-assisted IoT networks, covering path planning, resource management, collaborative connectivity, and data collection aspects. This structure ensures a logical flow of ideas and facilitates a deeper understanding of the research presented.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we explore scalable connection management for massive IoT systems. We begin by presenting a comparison of the previous studies on IoT resource management. Then, we delve into the literature that study the trajectory optimization for cellular-connected UAVs, discussing their contributions and drawbacks. Finally, we touch upon the AoI metric, emphasizing its importance for timely data collection from IoT devices using UAVs.

2.1 Scalable Connection Management for Massive IoTs

Subscriber identity sharing aims to efficiently use core network resources by grouping multiple IoT devices' traffic [19, 20]. This is done by giving a shared International Mobile Subscriber Identity (IMSI) to a group of IoT devices and allowing the core network to see them as one device. Devices send data in turns on this shared line to prevent traffic overlap. A related patent from Qualcomm discusses managing such wireless device networks [21].

This method of IMSI sharing communication helps in using fewer core network resources like the number of cellular bearers, especially when considering IoT devices in large service regions [19, 20]. Yet, these studies only allow pre-determined devices with similar data sending intervals to share the IMSI. Recent work [22] expanded this, considering all IoT devices and dynamic IMSI sharing using modern subscriber ID solutions like virtual SIMs [23] and e-SIM cards [24, 25]. These modern solutions aid in dynamically providing network connectivity to IoT devices [26].

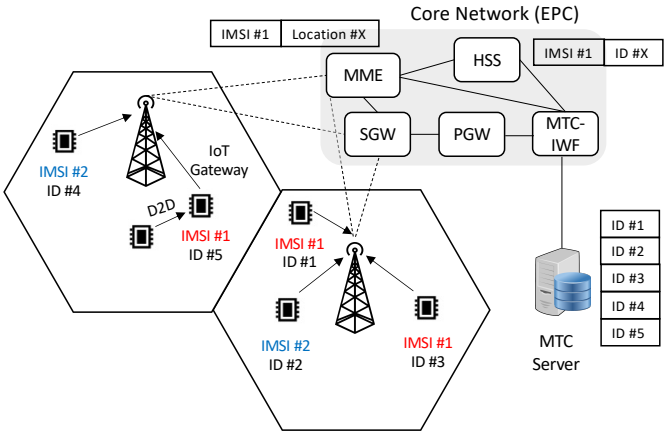


Fig. 2. Overview of IMSI sharing based aggregated IoT communication in EPC, as a representative of mobile core network.

Consider the EPC network in Fig. 2, as a representative core network architecture which is currently the most common system in use. The IoT devices that share the same IMSI are considered as the same device by the core network. However, the list of the IoT devices using the same IMSI are still being tracked by the MTC server in the background through the usage of external identifiers (EID) and MTC interworking function (MTC-IWF) [27] that is serving as an intermediary function between the core network and the MTC server. Note that MTC server does not deal with IP addresses and cellular IDs (e.g., IMSI), which is managed by Packet Data Network Gateway (PGW), and just uses external identifiers (EID) to communicate with the IoT devices. The mapping of IMSI and application port ID to EID is achieved through communication of MTC-IWF with HSS. The interested readers can refer to [20, 19, 22] for further details.

Previous studies have proposed an aggregated connection model that enables scalability for massive IoT by allowing devices to share an IMSI and flexibly shifting their traffic patterns as a practical solution without major architectural changes.

Several prior works [28, 29, 30, 31, 32, 33, 34, 35] have proposed solutions to handle the escalating connection demands from massive numbers of IoT devices. These include the core network functions [28], separating control and data planes via Software Defined Networking (SDN) and Network Function Virtualization (NFV) [29, 30, 31, 32, 33, 34, 35], and device-centric techniques like virtual bearers and group communication. Though promising, most solutions have limitations for practical large-scale deployment. Virtual bearers [34] require close device proximity for device-to-device communication. Group communication [35] necessitates devices be within the same base station coverage area. A lightweight stateless MME architecture [30] improves just one core network gateway, not the overall core network.

In contrast, more scalable and practical aggregated connection models using IMSI sharing have been studied [20, 22] without radically changing the core network architecture. These aggregate IoT devices to share a subscriber identity and take turns transmitting data.

Several studies shed light on the challenges and innovations in IoT resource management and communication. The study by [36] emphasizes multi-operator network sharing, aiming to enhance resource utilization and reduce costs but faces specific challenges in mobility and access channel efficiencies. [37] presents an adaptive sharing priority for IoT communications over LTE-A networks, yet grapples with scalability concerns. [22] offers a solution for managing hyper-dense IoT traffic using a computationally intensive algorithm. Lastly, [38] introduces a flexible spectrum sharing model for multi-tenant cellular networks, but its outcomes are somewhat limited. These investigations collectively underscore the evolving strategies and challenges in IoT resource management.

In Table 1, we show a comparison of the key studies and presents their concepts and the various metrics they have employed. There is a notable concern about scala-

Title	Main contribution	Metrics measured	Drawbacks and Challenges
Multi-Operator Network Sharing for Massive IoT [36]	Allowing multi-operator network sharing, facilitating the coexistence of IoT and high-speed cellular services, utilizing 3GPP's radio access network sharing architecture to improve resource use and reduce roll-out expenses **(Network Slicing)**.	• Total number of supported IoT devices • Ability to coexist with other cellular services	• Limited mobility management and traffic dynamic control • Excessive overhead and inefficiency of random access channel procedure
Analysis of an Adaptive Priority-Based Resource Sharing Scheme for Multiservice IoT Communications Over LTE-A Networks [37]	Designing efficient resource partitioning for different types of IoT device communications with an adaptive sharing priority (AS) scheme	• Average waiting delay • System utilization • Blocking Probabilities	• Lack of scalability in the proposed algorithm
Dynamically Shared Wide-Area Cellular Communication for Hyper-dense IoT Devices [22]	Developing a dynamically shared connectivity model to manage hyper dense IoT traffic using the same resources and to group the IoT devices based on their traffic patterns.	• Number of bearer usage • Resource utilization • Signaling cost	• Proposed solution used Genetic algorithm which is computationally intensive. • Does not consider non-zero delays threshold which may reduce the number of bearers further.
Traffic Load-Based Spectrum Sharing for Multi-Tenant Cellular Networks for IoT Services [38]	Proposing a spectrum sharing model that can be used to model any spectrum sharing policy between various tenants.	• Blocking probability • Spectrum utilization	• Limited numerical results and analyzed data • Does not consider non-zero delays threshold which may reduce the number of bearers further.

Table 1. Comparative overview of key previous studies on IoT resource management and communication

bility, computational intensity, and the ability to handle dynamic and diverse traffic. Several studies have not addressed non-zero delay thresholds, which are crucial in optimal resource management. Overhead and inefficiency appear in the management of IoT devices, highlighting the complexity of the proposed optimized algorithms and procedures.

2.2 Trajectory Optimization for Cellular-connected Collaborating UAVs

In this section, we review studies on cellular-connected UAVs and their connectivity issues. We start reviewing the studies that aim to optimize the single UAV's

path. Cellular-connected UAVs act as user equipment UE and rely on base stations for connectivity critical to successful mission completion [39, 40, 41, 42, 39, 43, 44, 45]. Recent studies have explored various issues in cellular-enabled UAV networks, including interference management [41], power control [42], and trajectory optimization [46, 39, 43] under constraints like privacy [44] and fading [40].

Due to limited UAV flight times, minimizing mission completion time has been a major focus [47, 48, 40, 49, 45]. Works have also jointly optimize time with other objectives. One study [45] jointly optimizes UAV trajectories and terrestrial user scheduling to maximize downlink throughput. Others [46, 39, 43] jointly optimize trajectories and transmit power to maximize minimum user rates. To enable solutions, non-convex optimizations have leveraged techniques like coordinate descent and successive convex optimization [45].

Given the increasing reliance on UAVs in diverse fields, from agriculture and delivery to defense and communication – ensuring their reliable operation via robust path planning that considers potential outages is essential [50, 51]. Recent works have incorporated outage constraints into path planning [47, 48, 40, 49], with different proposed solutions. A dynamic programming approach [47] provides approximate polynomial-time paths. Graph-based methods [48, 40, 49] find approximate paths via shortest path algorithms. A study [40] further includes fading effects for realistic results. However, these works optimize single UAV paths based only on base station coverage.

For multi-UAV missions, as shown in Fig. 1, this overlooks collaboration opportunities between UAVs. This work extends prior efforts by considering collaborative path planning for multiple UAVs. This differs from literature studying UAV relaying to reduce data transmission latency [52, 53, 54], which focuses on multi-hop data routing from sensors to base stations. In contrast, we study optimizing UAV paths

themselves in a collaborative way to improve cellular connectivity. In our research, we explore collaboration among UAVs to enhance connectivity maintenance. This is primarily achieved through multi-hop connections between UAVs, with at least one of them connected to a GBS. Moreover, such collaboration can sometimes reduce the total mission time. This is because one UAV can traverse non-connected areas while maintaining a relayed connection to another UAV that is linked to the GBS.

Several other studies delve into the trajectory optimization of cellular-connected UAVs. The research by [55] focuses on optimizing both the trajectory of a cellular-connected UAV and the information collection order of ground targets, aiming to minimize mission completion time. However, it investigates only one UAV and overlooks the age of information (AoI) while minimizing flight distance. The study by [56] proposes an iterative trajectory optimizing algorithm using graph theory. This approach emphasizes minimizing mission completion time, enhancing area coverage, and ensuring affordable computation time. Nonetheless, the model lacks collaboration and doesn't account for UAV movement outside the range of GBSs. [57] introduces a design where a UAV forms a mobile bistatic synthetic aperture radar (SAR) with its serving base station. The focus here is on energy conservation and maintaining sensing resolutions. The study, however, also limits its approach to a single UAV and its testing to maps with a limited number of stations and sensing areas. Lastly, [58] suggests a new trajectory planning scheme for multiple UAVs, combining dynamics analysis, path search, and nonlinear optimization. Their model primarily targets obstacle avoidance and minimizing mission time, but its optimization time extends with the number of UAVs, and larger maps with more obstacles have not been tested. Collectively, these papers highlight innovative methods in UAV path trajectory optimization but also underscore prevalent challenges in scalability and collaboration.

In Table 2, we summarize the key studies that are closest to our work and

Title	Main contribution	Metrics measured	Drawbacks and Challenges
Trajectory Optimization of Cellular-Connected UAV for Information Collection and Transmission [55]	Proposing a structured communication protocol between the UAV and the cellular network, establishing an equivalent graph-based model	• Minimize the mission completion time for information collection and transmission	• Only one UAV has been investigated. • Does not consider AoI while minimizing the total flying distance
An efficient trajectory planning for cellular-connected UAV under the connectivity constraint [56]	Optimizing UAV trajectories using geometric planning and graph theory, Identifying potential UAV-GBS associations based on their topological relationships.	• Minimizing mission completion time under the connectivity constraint • More area coverage • Affordable computation time	• Finding path for only one UAV investigated. No Collaboration! • Does not consider movement of UAV when it is out of range of GBSs.
Trajectory Planning of Cellular-Connected UAV for Communication-Assisted Radar Sensing [57]	Forming a mobile bistatic SAR for high-resolution large-area sensing without extra spectrum needs	• Minimizing propulsion energy • Guarantee the required sensing resolutions • Minimizing the flight distance in terms of energy saving and effective consumption fluctuation	• Finding path for only one UAV investigated. No Collaboration! • The first priority is to minimize the propulsion energy while satisfying the range and resolutions of sensing. • The algorithm tested on maps with limited number of BS and sensing areas.
Trajectory Planning for Collaborative Transportation by Tethered Multi-UAVs [58]	Proposing an approach combines dynamics analysis, kinodynamic path search, and nonlinear optimization for each UAV's path optimization.	• Obstacle avoidance • Minimizing the mission time	• The optimization time is relatively long and affected by the number of UAVs • The authors do not simulate their algorithm with large maps and more number of obstacles.

Table 2. Comparative overview of key studies on UAVs path trajectory optimization with collaboration

compare their contributions and drawbacks. These studies underline the importance of a more holistic and inclusive approach to UAV trajectory optimization, including the potential benefits and challenges of multi-UAV operations and a broader range of operational parameters and map scenarios.

2.3 Optimizing UAV Paths for Timely IoT Data Collection with AoI

Recently, a new metric called Age of Information or AoI has gained traction for UAV missions that collect data from ground sensors or IoT devices and deliver it to destinations [54]. Since timely delivery and information freshness is critical for some applications, AoI has attracted significant research attention [15, 59, 60, 54, 47, 61]. AoI has also been studied alongside considerations like wireless energy transfer, data acquisition mode selection, energy consumption, power optimization [19], etc. Proposed solutions utilize optimization techniques [54, 61], dynamic programming [15], and learning models like reinforcement learning [59].

Despite extensive AoI-related UAV path planning research, most studies only consider data collection periods and do not focus on UAV-core network/Internet communication. With multiple base stations, uploading data through different ones can affect device-specific AoI. Moreover, most works assume data generation before UAV dispatch, while practical scenarios involve continuous data generation. A recent work [62] considers these aspects but overlooks UAV mission time and path optimization, potentially resulting in longer paths. It also relies solely on complex optimization unlike our proposed heuristic-based fast solution. In our approach, we first determine the optimal solution which is finding UAV path trajectory using ILP, and then introduce a heuristic method that approximates this optimal path solution. The sequence in which IoTs are visited can influence the AoI and data freshness for IoT delivery.

The importance of optimizing AoI in UAV-assisted data collection for IoT networks has attracted significant attention in recent literature. For instance, [63] introduced a scheme superior to deep Q-network and actor-critic-based algorithms in terms of AoI and energy efficiency, yet it does not consider multi-UAV scenarios or certain relationships during the UAV-assisted IoT network data collection process.

[64] proposed an algorithm for UAV trajectory planning to aid cluster-based IoT networks, emphasizing the minimization of total AoI. However, it restricts its considerations to single UAV scenarios and assumes a specific type of IoT network. A deep reinforcement learning approach is adopted by [65] to enhance the UAV's trajectory for efficient data gathering, aiming to reduce AoI and packet drop rate. Still, the simulations are bound to a predefined map grid and single UAV settings.

In a similar vein, [66] integrates UAV trajectory and uplink transmission power optimization to curtail AoI under energy constraints, though it limits its perspective to single UAVs. A unique contribution from [67] delves into collaborative path planning for energy-efficient data collection, employing a Hexagonal Area Search algorithm combined with a multi-agent Deep Q-Network. This study notably highlights coverage, energy, and data collection metrics. However, it lacks a mathematical model to relate the path to the environmental state, making RL-based solution validation challenging. Finally, the work of [68] addresses multi-UAV assisted data collection aiming to enhance information freshness using an improved ant colony algorithm.

Table 3 presents a summary of key studies in the domain of UAV-assisted data collection in IoT networks with a considerable emphasis on optimizing the AoI. Various advanced methodologies, including deep reinforcement learning and algorithms surpassing traditional deep Q-networks, have been employed to address AoI while also enhancing other factors such as energy conservation. Notably, investigations have been conducted in both single and multi-UAV settings. However, many studies tend to focus predominantly on single UAVs, highlighting an area ripe for further inquiry, especially in multi-UAV systems.

.

Title	Main contribution	Metrics measured	Drawbacks and Challenges
AoI-Energy-Aware UAV-Assisted Data Collection for IoT Networks: A Deep Reinforcement Learning Method [63]	Proposing a scheme outperforming the deep Q-network and actor-critic-based algorithms in terms of achievable AoI and energy efficiency.	• Minimizing average Age of Information • Minimizing energy consumption	• Does not consider multiple UAVs • The relationship between UAV trajectory, device association, and IoT device power allocation for extending network lifetime hasn't been explored.
UAV Trajectory Planning for AoI-Minimal Data Collection in UAV-Aided IoT Networks by Transformer [64]	Proposing an algorithm for solving the trajectory planning problem of a UAV used to aid a cluster based IoT network	• Minimizing total age of information	• Does not consider multiple UAVs • The authors jointly optimize UAV hovering points and their visiting order in a cluster-based IoT network.
Deep Reinforcement Learning for Efficient Data Collection in UAV-Aided Internet of Things [65]	Proposing a deep reinforcement learning algorithm to optimize the UAV's trajectory for effective data collection from IoT ground sensors.	• Reducing AoI • Reducing packet drop rate	• Does not consider multiple UAVs • Simulation results run on a virtually divided map with size 10x9 grids
AoI-Minimal Power and Trajectory Optimization for UAV-Assisted Wireless Networks [66]	Given the problem's non-convex nature, the problem is solved using Lagrangian dual, convex optimization, and block coordinate descent methods.	• Minimizing the average AoI • Considering energy consumption limitation	• Does not consider multiple UAVs
Path planning of multi-UAVs based on deep Q-network for energy-efficient data collection in UAVs-assisted IoT [67]	Introducing a Hexagonal Area Search (HAS) algorithm combined with a multi-agent Deep Q-Network (DQN)	• Coverage ratio • Energy ratio • Data collection	• The authors skip a mathematical model because of the unclear path-environment relationship. • The authors do not validate their RL approach.
AoI-Sensitive Data Collection in Multi-UAV-Assisted Wireless Sensor Networks [68]	Introducing a strategy that minimizes SNs' AoIs through a three-step iterative process, Optimizing UAV path with improved ant colony algorithm.	• Minimizing the average AoI • Minimizing the peak AoI	• Does not consider multiple UAVs

Table 3. Comparative overview of key studies on AoI and UAV path trajectory optimization

CHAPTER 3

IMSI SHARING BASED MASSIVE IOT CONNECTIVITY/RESOURCE MANAGEMENT

3.1 Introduction

The proliferation of IoT technology has brought about a paradigm shift in various aspects of our daily lives. It has ushered in an era of interconnected devices across a multitude of applications, encompassing smart cities [69], environmental monitoring [70], and home automation [14]. This transformation has steered communication from being human-centric to machine-based, giving rise to a surge in the number of machine-type devices (MTD) [5]. This surge, however, has posed novel challenges for mobile network operators, particularly in the context of limited wireless spectrum and finite resources within core networks.

In response to these challenges, considerable research efforts have been dedicated to devising solutions spanning various network layers. Furthermore, new standards like Narrowband-IoT (NB-IoT) [71] have been developed to accommodate the demands of next-generation IoT networks.

This chapter delves into these challenges, with a particular focus on the efficient utilization of mobile core network resources. The core network gateways, typically designed to handle traffic from mobile users, may lead to resource under-utilization when MTDs, which sporadically transmit data, connect directly to macro-cell base stations (BS) and the core network. Even with the implementation of power-saving modes (PSM) [27] at the device level, core network resources continue to be consumed.

To address these inefficiencies, one common approach is to connect nearby MTDs

to a local gateway device, enabling them to collectively utilize the gateway's connection. This can be realized through a star topology or by employing device-to-device (D2D) communication [72, 73]. However, D2D technology's limited range restricts this solution to a local context, limiting the number of MTDs that can be accommodated. Moreover, the capacity of the back-haul connectivity from the gateway to the macro BS must be substantial enough to handle the traffic from all connected devices. Alternative solutions have been proposed to manage MTD connections to a macro BS using group-based Radio Resource Connection (RRC) and bearer establishment [74]. Still, they are confined to MTDs within the same macro BS range.

To achieve scalability, the concept of "aggregated communication" has been introduced, allowing a group of MTDs with similar data transmission intervals to share the same subscriber identity and take turns for data communication [19]. This grouping of devices occurs at the core network level, potentially spanning multiple macro BSs within a serving region of a core network gateway. Core network resources treat the communication from devices in the same group as if it were emanating from a single device, thereby conserving resources significantly.

The contributions of this chapter encompass a comprehensive exploration of solutions to these challenges:

- Development of ILP-based models to address the traffic shifting-based aggregated IoT communication problem optimally at each network moment.
- Introduction of a heuristic-based, polynomial-time algorithm for device grouping based on a novel metric rooted in traffic characteristics.
- Proposition of a polynomial-time algorithm that reorganizes device groups to adapt to dynamic scenarios where new IoT devices join the network, and others depart.

- Extensive simulations to assess the effectiveness of the proposed algorithms across diverse scenarios, highlighting their resource-saving benefits.

The subsequent sections of this chapter are structured in the following manner. The system model, problem description, and ILP formulations are detailed in Section 3.2. Our heuristic-driven solutions are thoroughly explained in Section 3.3. The assessment and outcomes of these solutions, based on various simulation scenarios, are covered in Section 3.4. In section 3.5, we execute a system for combining traffic from multiple mobile devices that use the same subscriber identity and network resources within the mobile core network. Finally, we culminate the chapter with our conclusions in Section 3.6.

Data traffic model. We assume that there is a set $G = \{I_1, I_2 \dots I_M\}$ of M IoT devices or MTDs[1] where each of them sends their data (e.g., measurements, computations) to their server in some constant intervals. Their data sending intervals and the required connectivity duration within each of these intervals vary due to different application specific requirements. To this end, we assume that for each device $I_i \in G$, the data upload happens at every λ_i time units and each data upload occurs for a duration of δ_i time units, starting at s_i and ending at e_i, within each λ_i duration (i.e., $\delta_i = e_i - s_i$), as shown in Fig. 3. We have chosen this model for simplicity, however, it could be extended to more complicated models (e.g., Gaussian distribution with a mean). We assume that the time is also divided into equal slots and all time related parameters are a multiple of the slot size. We also assume that each MTD uses its own bearer initially, and after aggregation process they are partitioned into groups. The group of MTDs that use the i^{th} bearer is denoted by G_i, and for convenience, we will also refer to the i^{th} bearer as G_i.

[1]We use IoT devices and MTDs interchangeably throughout the text.

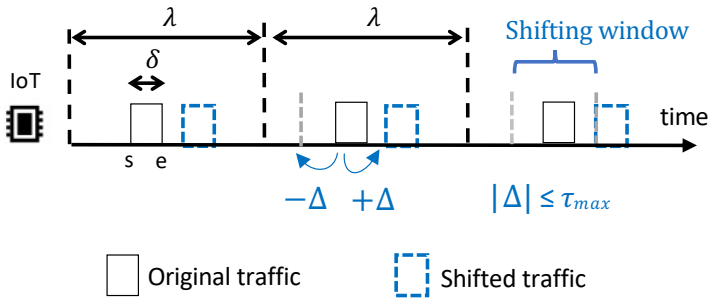


Fig. 3. Original and shifted traffic model for IoT devices.

In the following section we present the way we model this problem and our assumptions.

3.2 System Model

3.2.1 IoT Traffic Model and Traffic Shifting

Flexible traffic model. As it is shown in Fig. 3, we consider some slight shifts in the traffic pattern of the IoT devices. That is, the timing of each data upload instance for an IoT device can either happen earlier or later than its originally scheduled upload time without exceeding a given time threshold denoted with $\tau_{\max}$. Note that this threshold can be defined by the network management considering the application requirements (e.g., 5 minute shifting for humidity measurements that is happening twice a day may be considered fine).

In our initial study, we also considered an inconsistent model for the traffic pattern changes. That is, we let the individual data sending instances of the same IoT device to be shifted (i.e., delayed or scheduled earlier) differently without exceeding the threshold. While this gives more flexibility to the data uploads of the IoT devices, and hence provides an opportunity to group more IoT devices in the same cellular line, due to its complexity in modeling as well as only a slight benefit over consistently

Notations	Description
I_i	MTD or IoT device i
M	Number of MTDs
G (G^t)	The set of all MTDs (at time t)
G_i	The group of MTDs on i^{th} bearer, which is also denoted as bearer G_i.
G_{new}	The set of new MTDs joined to the network.
λ_i	Data sending interval of MTD i
δ_i	Duration of data communication in each data sensing interval for MTD i
s_i	Starting time of data communication within each interval by MTD i
e_i	Ending time of data communication within each interval by MTD i
$\mathcal{T}$ ($\mathcal{T}_j$)	Least Common Multiple (LCM) of data sending intervals (λ) of all MTDs (in group j)
x, y	Number of MTDs leaving and joining the network in every moment, respectively, in dynamic environments
b_i	Set to 1 if bearer i is used by at least one MTD and at any time (otherwise 0)
b_{ik}	Set to 1 if bearer i is used by at least one MTD at time slot k (otherwise 0)
$IMSI_j^t$	Temporary IMSI number or bearer ID assigned to MTD j at network moment t
b_{ijk}	Set to 1 if MTD i uses bearer j at the time slot k (otherwise 0)
$\tau_{\max}$	Maximum shifting allowed
$diff_j$	Set to 1 if IMSI number for MTD j is not equal to its IMSI number in previous moment.

Table 4. Notations used in Chapter 3

shifted traffic pattern model, we did not consider that type of inconsistent shifting model.

Dynamic network model. The number of IoT devices deployed and connected to the network of Mobile Network Operators (MNOs) has been growing massively. Similarly, the existing IoT devices have been replaced, moved or upgraded. In order to model such a dynamic network environment, we first define each time frame without a change in the set of devices as a network *moment*, and use two parameters to define the node joins and leaves between consecutive moments. That is, we assume that x of the existing IoT devices in the current moment will be leaving the network and there will be y new devices will be joining the network in the next moment. Note that depending on the relation between x and y values the network size can be affected differently i.e., when $x < y$, the network size will grow; when $x > y$, it will shrink; otherwise it will stay the same. In any case, the existing group structure among IoT devices can be affected dramatically and regrouping of devices or introduction of new cellular lines may be needed to carry the traffic of all IoT devices.

The notations used throughout this chapter and their descriptions are summarized in Table 4.

3.2.2 Problem Formulation and ILP Models

3.2.2.1 Initial Network

The primary goal of aggregating traffic from MTDs through IMSI sharing is to minimize the number of active bearers used by all devices and optimize resource usage in the core network. In Section 3.5, we have some experiments showing how it is possible to create and execute a system for combining traffic from multiple mobile devices that use the same subscriber identity and network resources within the mobile

core network. We achieve this by utilizing commercial smartphones, programmable SIM cards, and the Amarisoft Callbox [75], which includes both core network and eNB (evolved NodeB) components.

When no shifting is allowed in the initially scheduled traffic patterns of MTDs, devices can still be grouped to some extent, provided there is no overlap in the traffic patterns of different devices within the same group. However, if the devices are allowed to shift their upload times slightly (i.e., less than $\tau_{\max}$) within their long data sending intervals, there will be more opportunities to reduce the number of groups and actual bearers used, leading to increased resource savings. To address this, we utilize ILP to formulate the problem (P1) for finding the optimal aggregation at the initial network moment, considering the flexible traffic model (which allows shifting), as shown below:

$$
\begin{aligned}
&(P1): \\
&\min \quad \sum_{j=1}^{M} b_j, \\
&\text{s.t.} \quad b_j = \min\left\{1, \sum_{k=1}^{\mathcal{T}} b_{jk}\right\}, \forall j \in [1, M], && (3.1)\\
&\qquad b_{jk} = \min\left\{\sum_{i=1}^{M} b_{ijk}, 1\right\}, \forall j \in [1, M], \forall k \in [1, \mathcal{T}], && (3.2)\\
&\qquad \sum_{i=1}^{M} b_{ijk} \leq 1, \forall j \in [1, M], \forall k \in [1, \mathcal{T}], && (3.3)\\
&\qquad \exists! \Delta \in [-\tau_{\max}, +\tau_{\max}], \\
&\qquad \sum_{d=1}^{\delta_i} b_{ij(r\lambda_i + ((s_i + \Delta + d) \mod (\lambda_i)))} = \delta_i, \\
&\qquad \forall i, j \in [1, M], \forall r \in [0, \mathcal{T}/\lambda_i - 1], && (3.4)\\
&\qquad b_{ij((r-1)\lambda_i + d)} = b_{ij((r\lambda_i + d))}, \forall d \in [1, \lambda_i], \\
&\qquad \forall i, j \in [1, M], \forall r \in [1, \mathcal{T}/\lambda_i - 1], && (3.5)
\end{aligned}
$$

where,

$$
\begin{aligned}
&\mathcal{T} = \text{LCM}\{\lambda_1, \ldots, \lambda_M\}, \\
&b_{ijk} = \begin{cases} 1, & \text{if } I_i \text{ uses bearer } j \text{ at time slot } k, \\ 0, & \text{otherwise.} \end{cases}
\end{aligned}
$$

The objective function in Equation (3.1) seeks to minimize the number of actively used bearers, where b_j equals 1 if there is an MTD device utilizing it. Equations (3.1) and (3.2) determine the usage of each bearer (up to M total active bearers if each MTD uses a separate one) by verifying if at least one MTD is employing it during any time slot. Since the data transmission intervals (λ) can differ across MTDs, we first find the least common multiple (LCM) of their intervals to define a common timeline

$\mathcal{T} = \text{LCM}(\lambda_i, \forall i)$.

Equation (3.3) permits at most one MTD to use each slot, while (3.4) mandates exactly one ($\exists!$) *shifting* amount (Δ) between $-\tau_{\max}$ and $\tau_{\max}$ that makes all δ_i consecutive slots used by the i^{th} MTD (i.e., I_i) at a given bearer j. Here, r determines the timing of repeated data uploads within $\mathcal{T}$.

We also use (3.5) to provide some flexibility to the traffic model while controlling it to stay within acceptable thresholds. Overall, this enforces consistency in the shifting between the distinct data transmission intervals of an MTD at its utilized bearer.

3.2.2.2 Dynamic Network

The previously mentioned model specifically addresses the optimal arrangement of IoT devices (i.e., combining their data traffic) in a given network instance. Therefore, it is primarily employed at the initial stage. In scenarios with changing conditions, it becomes essential to account for the transition from the current state to the subsequent one. In this context, our aim involves not only reducing the count of communication channels (bearers) utilized but also minimizing modifications to the existing group configuration from the preceding moment. Any alteration in the identity of current devices necessitates reconfiguration, leading to additional control traffic and delays. Nonetheless, since the latter objective is of lesser importance compared to the primary goal of diminishing the number of bearers, we employ a scalarization approach to incorporate this prioritization in the objective function 3.6 of the second problem (P2). This problem pertains to determining the optimal data aggregation in dynamic settings.

$$(P2):$$

$$\min \quad \left(\sum_{i=1}^{M} IMSI_i^t\right) \times \mathcal{L} + \sum_{i=1}^{M} diff_i, \tag{3.6}$$

$$\text{s.t.} \quad IMSI_i = j, \text{if} \sum_{k=1}^{\mathcal{T}} b_{ijk} \geq 1, \forall i, j \in [1, M], \tag{3.7}$$

$$diff_i = \begin{cases} 1, & IMSI_i^t \neq IMSI_i^{t-1} \\ 0, & \text{otherwise.} \end{cases}. \tag{3.8}$$

In this context, Equation (3.7) guarantees the uniform assignment of the same IMSI number to devices sharing a common bearer. Simultaneously, we assign the bearer ID (denoted as j) as a provisional IMSI number for devices within that bearer. This temporary IMSI number can subsequently be associated with an authentic IMSI from a SIM card. Furthermore, Equation (3.8) identifies devices that will experience an IMSI alteration in the current instance (t) compared to the previous instance ($t-1$).

In the formulation of the objective function (3.6), the parameter $\mathcal{L}$ is introduced as a constant constraint. This constraint ensures that the cumulative IMSI changes across all devices in the system exert a controlled influence on the optimization process, prioritizing the reduction in the utilization of distinct IMSIs (groups or bearers) by the devices. Notably, the initial segment of the optimization function involves the summation of IMSI numbers across all devices. This summation not only minimizes the number of employed groups but also arranges devices within bearers sequentially, without introducing any unoccupied bearers between consecutive ones (e.g., preventing the use of bearers 1 and 5 instead of bearers 1 and 2). This design choice serves to mitigate the frequency of IMSI alterations among devices during successive time

intervals.

By analyzing how all devices send data, a MNO can use special models to decide which IoT device should be connected to which communication bearer at different times. This helps the MNO update important network details, like IMSI numbers for devices, using an online model explained in this section. However, these models, even for a small number of IoT devices (about 10-15), take a long time to run (e.g., more than 1 day processing time with a 2.5 GHz Quad-Core Intel Core i7 computer). So, if we need to use them often, like when the IoT devices or their data patterns change a lot, they might not be very practical.

To address this concern, the subsequent section presents heuristic-based solutions characterized by reduced computational complexities.

3.3 Heuristic-based Solution

3.3.1 Initial Aggregation

3.3.1.1 Overview

To consolidate the data traffic of multiple MTDs onto the fewest possible communication paths, we adopt an iterative strategy and make a series of informed selections at each stage. An outline of this procedure is depicted in Fig. 4. Initially, we assume that each device operates on a distinct communication path or group. Subsequently, we initiate the process by identifying all permissible pairs of communication paths that can be merged. This determination hinges on examining whether there exists a shared time slot allocated to both of these communication paths. Among the set of permissible communication path pairs (with no time slot overlap), we prioritize the pair that yields the highest *addition score (AS)*, computed as follows:

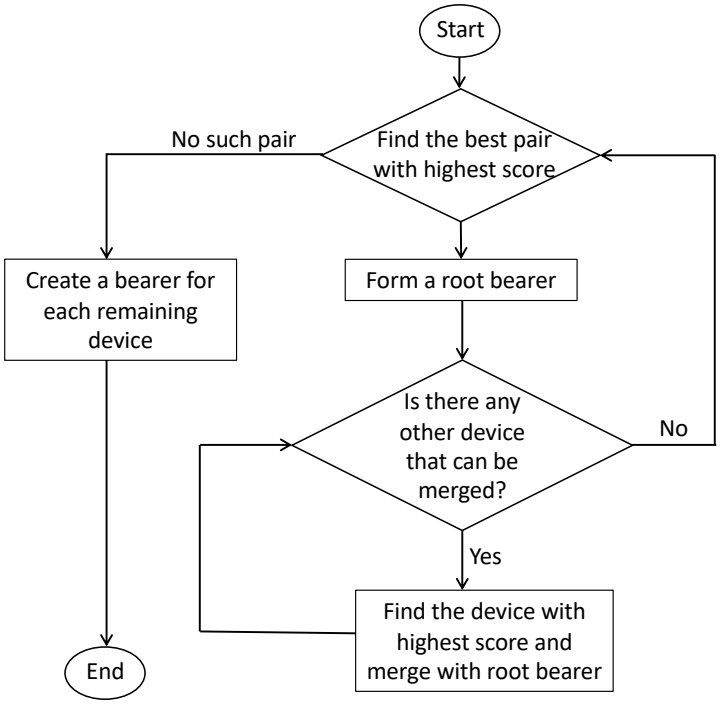


Fig. 4. Overview of heuristic based initial aggregation (HIA) procedure.

$$(I_x^{max}, I_y^{max}) = \underset{\substack{\forall I_x, I_y \in G \\ I_x \neq I_y}}{\arg\max}\, AS(I_x, I_y). \tag{3.9}$$

Subsequently, the traffic from these two bearers and devices is consolidated into a single bearer, referred to as the *root bearer* (G_{root}), while the other bearer is released.

In successive stages, we assess all remaining MTDs on their individual bearers to determine if they are suitable for integration with the traffic of the root bearer. Among those deemed suitable, we select the one that contributes the highest addition score and combine its traffic with the root bearer. Specifically, assuming G' represents the set of MTDs that have not yet been merged into any other bearer, we identify:

$$I_z^{max} = \underset{\forall I_z \in G'}{\arg\max}\, AS(I_z, G_{root}). \tag{3.10}$$

This iterative procedure persists until no further MTD traffic is qualified for integration into the existing root bearer. Subsequently, the process continues by establishing a new root bearer from the pool of single-MTD bearers that have not yet been aggregated. The subsequent steps involve identifying the most favorable pair of bearers, consolidating their traffic, and systematically appending other device/bearer traffic to this chosen bearer, sequentially, until no additional eligible bearers remain.

Here, note that, if there is no eligible pair of bearers that can be merged and assigned as root bearer, we stop the entire process and leave each of the single-MTD bearers as a separate bearer without any aggregation. A formal depiction of this heuristic-driven methodology is presented in Algorithm 1. The creation of root bearers is detailed in lines 4-11, and the gradual inclusion of other bearers onto the root bearer is described in lines 16-32. In instances where no further root bearer can be established, each remaining MTD retains its separate bearer, as outlined in lines 35-39.

3.3.1.2 Addition Score (AS) Function

In this iterative and greedy heuristic based approach, the critical part is the score function. As our goal is to aggregate as many MTD traffic as possible on a single bearer, we select the root bearer as well as the next added bearers to it such that the allocated time slots in the entire timeline are distributed in a way that adding a new MTD traffic will be easier. To attain this goal, we consider three different criteria:

- *Active Timeline* ($\mathcal{A}$): It is the duration from the first allocated time slot until the last allocated one. For bearer or group j, G_j, we find the minimum start time and maximum end time of all IoT devices on this bearer, and take the

Algorithm 1: Initial Aggregation (G)

Input: G: Initial set of MTDs

$AS_{max} = 0$
$\alpha = 0$ // Next bearer id to assign MTDs
while $|G| > 0$ **do**
  foreach (I_x, I_y) *s.t.* $I_x, I_y \in G, I_x \neq I_y$ **do**
    if I_x *and* I_y *are eligible to be merged* **then**
      if $AS(I_x, I_y) > AS_{max}$ **then**
        $AS_{max} = AS(I_x, I_y)$
        $(I_x^{max}, I_y^{max}) = (I_x, I_y)$
      end
    end
  end
  if $AS_{max} \neq 0$ **then**
    $G_\alpha = \{I_x^{max}, I_y^{max}\}$
    $G = G \setminus G_\alpha$
    $E = G,\ AS_{max} = 0$
    while $|E| > 0$ **do**
      foreach $I_z \in E$ **do**
        if I_z *can be merged on* G_α **then**
          if $AS(I_z, G_\alpha) > AS_{max}$ **then**
            $AS_{max} = AS(I_z, G_\alpha)$
            $I_z^{max} = I_z$
          end
        end
      end
      if $AS_{max} \neq 0$ **then**
        $G_\alpha = G_\alpha \bigcup \{I_z\}$
        $E = E \setminus \{I_z\}$
        $AS_{max} = 0$
      else
        $E = \emptyset$
      end
    end
    $\alpha = \alpha + 1$
  else
    foreach $I \in G$ **do**
      $G_\alpha = \{I\}$
      $G = G \setminus G_\alpha$
      $\alpha = \alpha + 1$
    end
  end
end

difference:

$$\mathcal{A}_j = e^j_{max} - s^j_{min}, \text{where} \tag{3.11}$$

$$s^j_{min} = \min\{s_i,\ \forall I_i \in G_j\}, \tag{3.12}$$

$$e^j_{max} = \max\{e_i,\ \forall I_i \in G_j\}. \tag{3.13}$$

- *Utilization* ($\mathcal{U}$): It refers to the percentage of time slots allocated within the active timeline. Given that $b_{ijk} = 1$ when MTD i allocates bearer j at time slot k, for all MTDs on a given bearer or group j, G_j, we calculate

$$\mathcal{U}_j = \left(\sum_{k=s^j_{min}}^{e^j_{max}} a_k \right) / \mathcal{A}_j, \text{ where} \tag{3.14}$$

$$a_k = \begin{cases} 1, & \text{if } \exists I_i \in G_j \text{ s.t. } b_{ijk} = 1 \\ 0, & \text{otherwise.} \end{cases} . \tag{3.15}$$

- *Border Score* ($\mathcal{B}$): This indicates how close the active timeline is to the end points of the entire timeline. As the allocated time slots get close to the sides of the entire timeline, the likelihood of allocating another MTD to the same bearer increases. Thus, we first find the minimum of distances to the start and end of entire timeline from the start and end of active timeline and take their sum. That is, for bearer or group j, G_j, we compute

$$\mathcal{B}_j = \min\{\mathcal{T}_j - s^j_{min}, s^j_{min}\} + \min\{\mathcal{T}_j - e^j_{max}, e^j_{max}\}. \tag{3.16}$$

Here, $\mathcal{T}_j$ is the timeline considered for bearer j and defined by $LCM(\lambda_i, \forall I_i \in G_j)$. We consider these criteria in a prioritized manner with the following order:

$$\min(\mathcal{A}_j) \succ \max(\mathcal{U}_j) \succ \min(\mathcal{B}_j). \tag{3.17}$$

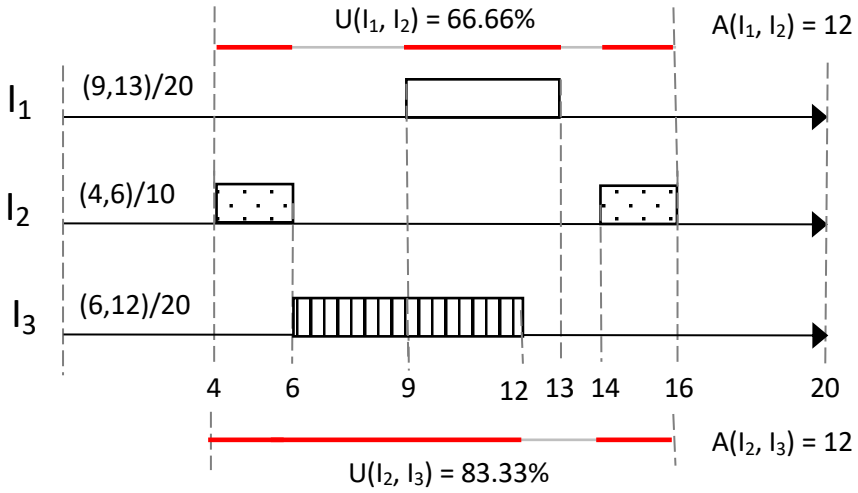


Fig. 5. Example: Only devices (I_1, I_2) and (I_2, I_3) are eligible to be merged on the same bearer as their traffic patterns do not overlap. They both have the same active timeline score, but latter has larger utilization score, thus is selected.

If the active time is the same, we go for the ones that use more of the available space. And if both active time and usage are equal, we give more importance to situations that are nearer to the edges.

Let's take a look at the scenario shown in Fig. 5 involving three MTDs. Here, we have two choices: combining devices I_1 and I_2, or I_2 and I_3. This is because we can't merge I_1 and I_3 due to overlapping traffic. When we calculate the score for the active timeline, it turns out to be $\mathcal{A} = 12$ for both options.

Next, we consider the second criterion, which is utilization. For the first choice, we have $\mathcal{U} = (2 + 4 + 2)/12 = 66.66\%$, and for the second choice, we get $\mathcal{U} = (2 + 6 + 2)/12 = 83.33\%$. This means that the second combination has a higher utilization, so we prefer to aggregate the traffic of I_2 and I_3.

It is important to note that the border score is the same for both cases, which is $\mathcal{B} = 4 + 4 = 8$. However, in this example, we don't consider the border score as it is the third criterion.

3.3.2 Complexity Analysis

In Algorithm 1, the maximum number of single-MTD bearer pairs that need to be examined to identify the optimal root bearer is $\binom{M}{2}$. If another single device can be added to the current root bearer, finding the best one will require less effort, roughly $\mathcal{O}(M)$. However, if no other device can be added to the root bearer and a new root bearer needs to be determined by comparing scores of pairs in the remaining set of unassigned devices, an additional $\binom{M-2}{2}$ pairwise comparisons will be needed.

The worst-case scenario occurs when the root bearer is repeatedly selected without adding any third device, resulting in approximately $\mathcal{O}(M^3)$ eligibility checks and score calculations. It is worth noting that the computational cost of score calculations remains consistent with or without shifting. However, the cost of eligibility checks increases. Without shifting, it merely compares each time slot within $\mathcal{T}$ to identify overlaps, resulting in an overall complexity of $\mathcal{O}(M^3\mathcal{T})$. On the other hand, with shifting, each device's time slot is considered within a shifting range of $[-\tau_{max}, \tau_{max}]$, requiring approximately $\mathcal{O}(\tau_{max}^2)$ combinations, each incurring a cost of $\mathcal{T}$. Consequently, the overall cost becomes approximately $\mathcal{O}(M^3\mathcal{T}\tau_{max}^2)$.

3.3.3 Dynamic Aggregation During Transition Between Moments

3.3.3.1 Overview

In a dynamic network environment, when the devices or their traffic patterns change, a new network situation begins. As a result, we need to reevaluate the existing groups or combined bearer setups. We provide an overview of this process in Fig. 6.

It is important to note that for each new situation, we have the option to use Algorithm 1 to create a fresh grouping among the new set of devices, without consid-

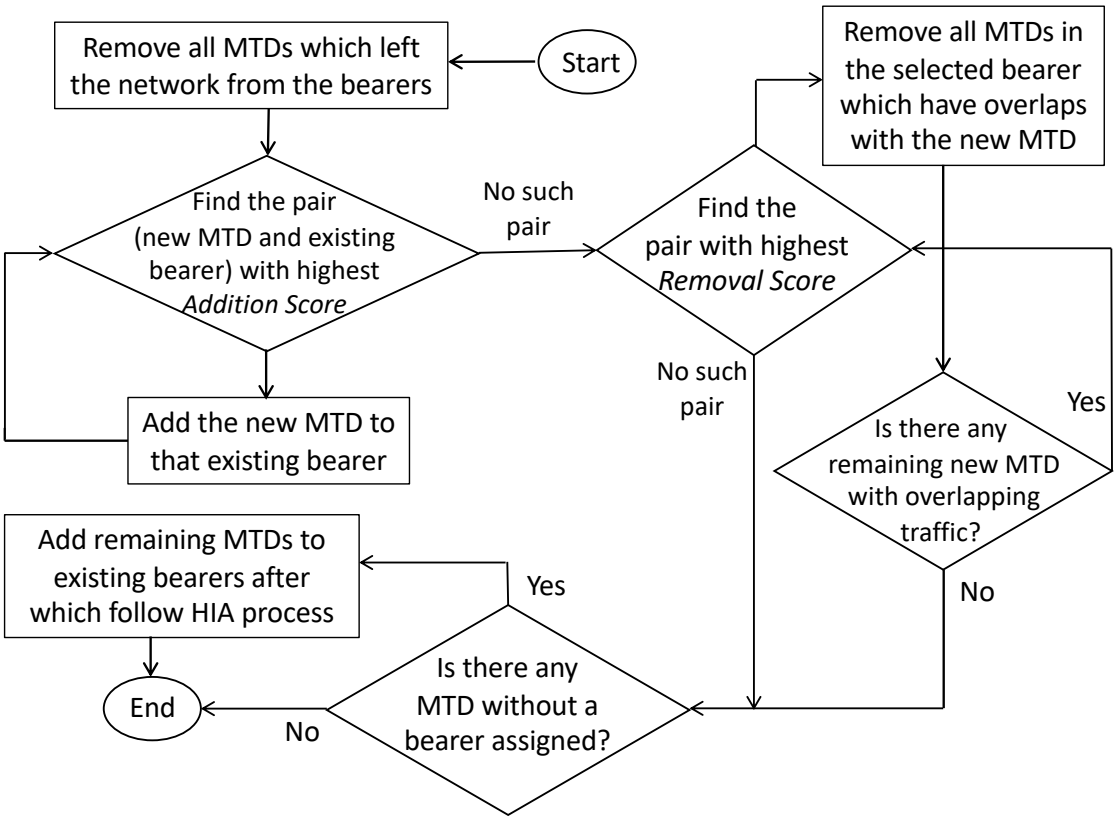


Fig. 6. Overview of heuristic based dynamic aggregation (HDA) procedure.

ering the previous groupings of devices that are also present in the current situation. However, this approach might lead to unnecessary changes in the assignments of device IMSI numbers. Furthermore, each change in IMSI or bearer for a device triggers a new provisioning process, which introduces some delay and incurs control traffic costs.

To address this concern, when new devices join the network and some existing ones depart, we adopt a strategy to first attempt integrating the newly joined MTDs into existing groups. This process mirrors the approach used for incorporating other devices into the root bearer, as outlined in Algorithm 1 (lines 16-32). However, in this case, we consider all potential combinations of new devices (denoted as $I_z \in G_{new}$) and existing groups (denoted as $G_j \in G_{cur}$). The sequence of additions is determined

by consistently identifying the pair that yields the highest score after each addition. In other words, we find:

$$(I_z^{max}, G_j^{max}) = \underset{\substack{\forall I_z \in G_{new} \\ G_j \in G_{cur}}}{\arg\max} AS(I_z, G_j). \tag{3.18}$$

Following this, we include I_z^{max} in G_j^{max} and remove it from G_{new}. This process is reiterated until no further additions are feasible.

Algorithm 2: Dynamic Aggregation (G_{cur}, G_{new})

Input: G_{cur}: Set of existing groups of MTDs
G_{new}: Set of new MTDs joined
Keep merging (I_z^{max}, G_j^{max}) from (3.18) until no more possible.
$RS_{max} = 0$
$G_{tba} = \emptyset$ // Set of MTDs to be assigned a group
while $G_{new} \neq \emptyset$ **do**
  foreach $I_{new} \in G_{new}$ **do**
    foreach $G_i \in G_{cur}$ **do**
      if I_{new} *overlaps with an MTD in* G_i **then**
        if $RS(I_{new}, G_i) \geq RS_{max}$ **then**
          $RS_{max} = RS(I_{new}, G_i)$
          $(I_{best}, G_{best}) = (I_{new}, G_i)$
        end
      end
    end
  end
  if $RS_{max} \neq 0$ **then**
    Remove each MTD in G_{best} that overlaps with I_{best} and add to G_{tba}
    $G_{tba} = G_{tba} \cup I_{best}$
    $G_{new} = G_{new} \setminus \{I_{best}\}$
  else
    break
  end
end
$G_{new} = G_{tba} \cup G_{new}$
Keep merging (I_z^{max}, G_j^{max}) from (3.18) until no more possible.
Add each remaining device to one individual bearer as in lines 35-39 of Alg.1.

Keep in mind that this procedure might lead to placing each new device within an existing bearer if their traffic patterns don't overlap (i.e., $G_{new} = \emptyset$). However, when this is not the scenario, an ideal approach could be to generate new bearers and allocate the remaining new devices to them. But, prior to doing so, we consider an intermediate step to optimize these new bearer assignments. This involves temporarily removing some of the existing MTDs. The goal here is to enhance the

allocation process by accounting for the new devices that have joined the network since the previous assessment. The details of this *smart removal* process are outlined in Algorithm 2 (lines 2-23).

To elaborate, we begin by identifying the pair consisting of an existing group and a new MTD that has the highest *removal score (RS)* (lines 5-13). Subsequently, we eliminate the MTDs in that group which overlap with the new MTD (line 16) and include them, along with the new MTD, in a set G_{tba} of MTDs. These MTDs will be assigned a new group ID and IMSI together (lines 17-18). We continue this process by finding the next best pair until no more overlaps are encountered. It is important to note that this process terminates either after considering all newly joined MTDs or when no further overlaps are found between existing groups and the remaining new MTDs.

In case the latter scenario occurs, meaning no more overlaps exist, we include all remaining new MTDs in the set of MTDs to be assigned new groups (line 23). Subsequently, we recommence the recursive addition process until no more additions are feasible. Following this, we proceed to create new bearers for the remaining devices (as seen in lines 35-39 in Algorithm 1).

3.3.3.2 Removal Score (RS) Function:

The critical part during this process is the removal score function, for which we consider the following three criteria:

- *Count of Overlapping MTDs ($\mathcal{C}$):* This count signifies the number of MTDs in an existing bearer j's timeline that share a data sending interval with the newly joined MTD. Assuming that I_{new} is temporarily assigned to G_j, and using b_{ijk}

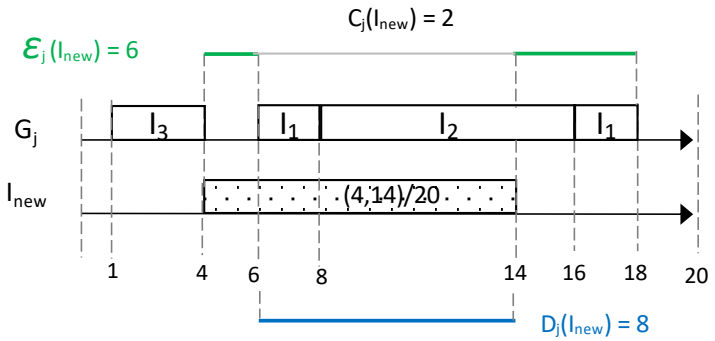


Fig. 7. Example scenario for smart removal process between an existing bearer and group (G_j) from previous moment and a new MTD (I_{new}) joined.

(set to 1 when MTD i allocates bearer j at time slot k), we define

$$\mathcal{C}_j(I_{new}) = |\{I \in G_j \mid \exists k \in \mathcal{T}, b_{I_{new}jk} = 1, b_{Ijk} = 1\}|. \tag{3.19}$$

Removing MTDs from a bearer with a higher count of overlapping MTDs increases the potential for reassignment to other bearers during the final addition process. This aids in reducing the overall number of bearers, as each removed MTD can be efficiently assigned to different bearers. Fig. 7 illustrates bearer G_j with two MTDs (I_1, I_2) overlapping with the new MTD.

- *Overlap Duration ($\mathcal{D}$):* This measure captures the portion of data sending intervals where the new MTD and an existing bearer j's timeline intersect. Specifically, we have

$$\mathcal{D}_j(I_{new}) = \sum_{\forall k \in \mathcal{T}} \mid (b_{I_{new}jk} + b_{jk} = 2). \tag{3.20}$$

Note that b_{jk} is 1 when bearer j is used by at least one MTD at time slot k. A shorter duration of overlap increases the likelihood that removing MTDs from that group will help reduce the total number of bearers. This is particularly relevant when considering shifting possibilities. In Figures 7, $\mathcal{I}_{new}$ has overlap

from time slot 6 to 14 with MTDs already present in bearer j's timeline.

- *Non-overlap Duration ($\mathcal{E}$):* This metric represents the duration of data sending intervals where intersecting MTDs and the new MTD don't overlap in an existing bearer j. It is defined as:

$$\mathcal{E}_j(I_{new}) = \sum_{\forall k \in \mathcal{T}} \mid (b_{I_{new}jk} + b_{jk} = 1). \tag{3.21}$$

  A longer non-overlap duration increases the potential for reducing the total number of bearers by removing MTDs from that group. This is because freeing up more space by removing these MTDs enables more unassigned devices to be accommodated. In Fig. 7, non-overlapping parts exist from time slot 4 to 6 and 14 to 18.

These criteria are considered in the following order for the smart removal score calculation:

$$\max(\mathcal{C}_j) \succ \min(\mathcal{D}_j) \succ \max(\mathcal{E}_j). \tag{3.22}$$

This implies that priority is given to cases with a higher count of overlapping MTDs. If there is a tie, we consider cases with a shorter overlap duration. If a tie persists, we then prioritize cases with a longer non-overlap duration (random selection if the tie persists).

3.3.4 Complexity Analysis

In Algorithm 2, during the initial addition process (line 1), in the worst scenario, each MTD might be placed in its own bearer (i.e., M bearers from the previous moment), and with the inclusion of y new devices, we may need to assess all yM pairings over a duration of $\mathcal{T}$. When considering potential shifting, this results in a

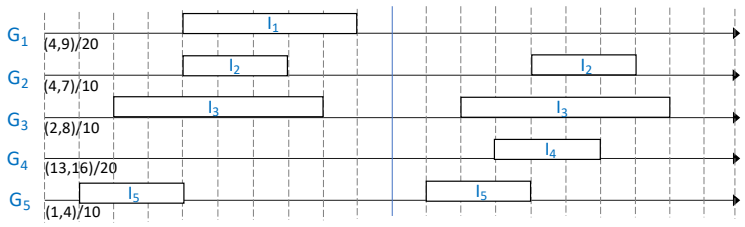


Fig. 8. Moment0: Original traffic without grouping

complexity of $\mathcal{O}(y\tau_{max}^2 M\mathcal{T})$.

In a situation where, at worst, most of the new MTDs are initially placed within existing bearers individually and later require removal (e.g., due to a single long MTD) during the smart removal process, determining the order of their addition necessitates score calculations for each remaining new MTD and existing bearer pairing (i.e., $(y-1)M, (y-2)M \ldots$). As a result, the overall cost of the initial addition can approach $\mathcal{O}(y^2\tau_{max}^2 M\mathcal{T})$.

Throughout the removal process, the worst-case scenario entails removing each MTD in existing groups one by one, incurring a cost of $\mathcal{O}(y^2 M\mathcal{T})$ as shifting is not considered during removal. Subsequently, following a removal process that leads to the removal of all MTDs from every bearer, we initiate the final addition process (lines 24-25) to obtain new bearers, resembling the process in Algorithm 1 with a complexity of $\mathcal{O}(M^3\mathcal{T}\tau_{max}^2)$. Therefore, the overall complexity of Algorithm 2 per network moment is approximately $\mathcal{O}(M\mathcal{T}\tau_{max}^2(M^2+y^2))$.

3.3.5 Toy Example

In this part, we provide how the proposed algorithms work on an example set of MTDs and two network moments. Initial moment of the network with five MTDs and active bearer utilization in different scenarios are shown in Figures 8, 9, and 10.

We examine a group of 5 MTDs, each initially assigned to its own individual

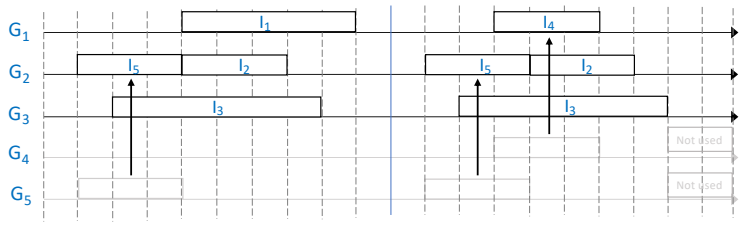


Fig. 9. Moment0: Grouping MTDs with no shifting

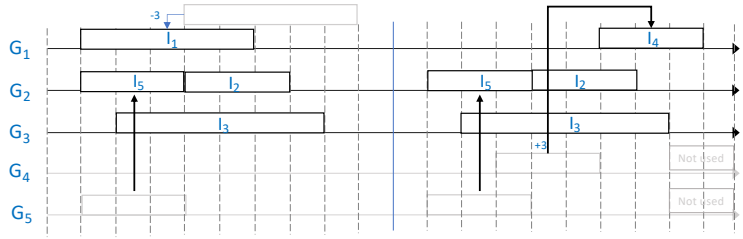


Fig. 10. Moment0: Grouping MTDs with shifting

bearer (for instance, I_1 on bearer G_1), as illustrated in Fig. 8. The corresponding traffic patterns are also depicted in Fig. 8. For instance, I_1 sends data between the 4th and 9th time units within every 20 time units. To accommodate all repetitions of data communication for each device throughout this common timeline, we display their patterns.

Upon applying the initial aggregation algorithm without utilizing a shifting model, it identifies pairs that can be merged, such as (I_2, I_5) and (I_1, I_4). By computing their addition scores, the algorithm determines that (I_2, I_5) has the highest score and thus merges their traffic onto a single bearer (namely, G_2). However, attempting to merge other MTDs into this root bearer doesn't yield further improvements. Consequently, a new round of pairwise assessment commences among the remaining MTDs (I_1, I_3, I_4).

In the next iteration, (I_1, I_4) is selected and merged to create a new root bearer (i.e., G_1). Since I_3 remains the sole MTD left and cannot be added to this root

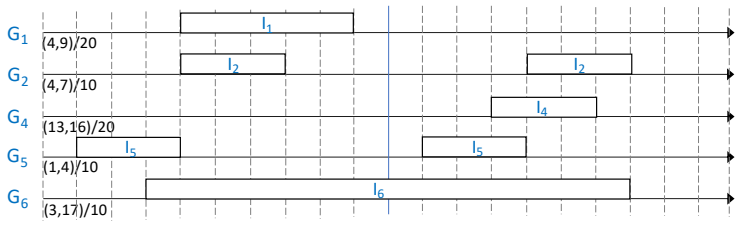


Fig. 11. Moment1: Original traffic without grouping

bearer, and no new root bearer can be formed, the algorithm concludes. As a result, I_3 remains on a separate bearer (specifically, G_3). This completes the initial aggregation procedure without employing shifting, resulting in the usage of 3 active bearers for handling the traffic of 5 MTDs, depicted in Fig. 9.

When traffic shifting is considered with a maximum shift of $\tau_{max} = 3$, we introduce the concept of adjusting each MTD's traffic within the range of $[-\tau_{max}, +\tau_{max}]$ during the eligibility assessment of pairwise merges between MTDs.

This time, in addition to the previously identified pairs from the case without shifting, the algorithm discovers two more eligible pairs, namely (I_2, I_4) and (I_4, I_5), thanks to the flexibility offered by shifting. Nevertheless, despite these new options, (I_2, I_5) still yields the highest score, making it the preferred choice for forming the initial root bearer. In the second iteration of the algorithm, with only one eligible pair remaining (namely, (I_1, I_4)), this pair is selected, and their respective MTDs are merged onto bearer G_1. It is important to note that the data patterns of I_1 and I_4 have been shifted by -3 and +3 time slots, respectively. This adjustment aims to create more space for future additions due to the influence of the border score. Since no further MTDs can be added to this root bearer and only one unmerged MTD (I_3) remains, the process concludes by assigning I_3 to a separate bearer, just as in the previous case.

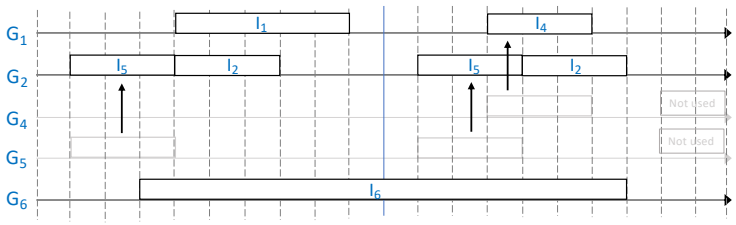


Fig. 12. Moment1: Grouping MTDs with no shifting

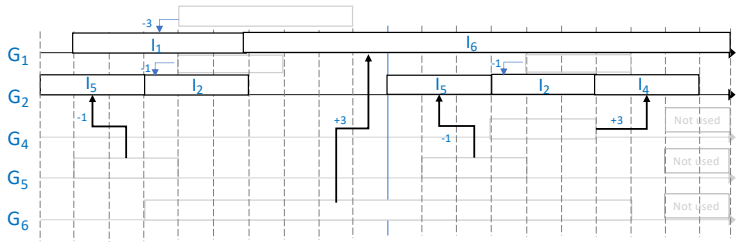


Fig. 13. Moment1: Grouping MTDs with shifting

After this initial aggregation, we make one of the MTDs (i.e., I_3) leave and a new MTD (i.e., I_6) join the network, as shown in Figures 11, 12, and 13, and run the dynamic aggregation algorithm in Algorithm 2.

In the absence of shifting, when I_3 departs, only two active bearers (G_1, G_2) remain in use. Given that I_6 cannot be accommodated in these existing bearers, we initiate the smart removal process. Since I_6 has overlaps with both groups/bearers G_1 and G_2, we determine the preferred option based on the removal score.

The newly introduced MTD I_6 has overlaps with 2 MTDs within each bearer's timeline, making them equal in the context of the first metric in the removal score function. Subsequently, we examine the second priority (i.e., duration of intersection) and opt for G_1 due to its shorter intersection duration with the new MTD (i.e., 8 versus 10). We then proceed to remove all MTDs within G_1 and commence the process of adding unassigned devices (i.e., I_1, I_4, I_6). However, since none of them can be added to the sole remaining active bearer (i.e., G_2), we seek pairs among them

to establish a root bearer. Given that only I_1 and I_4 can be grouped without overlap, we place them into a new root bearer (i.e., G_1). As for I_6, it cannot be merged into this bearer, thus retaining its own separate bearer. Consequently, this series of actions culminates in distributing these devices across three bearers, as illustrated in Fig. 12.

After the departure of I_3 (as seen in Fig. 10), only two bearers are used, employing traffic shifting limited to $\tau_{max} = 3$. When I_6 joins, and since it cannot be accommodated in bearers G_1 and G_2, we initiate the smart removal procedure. We examine all potential shifts within the range of [-3, +3] for I_6, calculating the smart removal scores when paired with each active bearer.

Upon shifting I_6's traffic by +3 time units, only one overlapping device (I_4) remains in bearer G_1 alongside I_6 (all possible shiftings place both MTDs' traffic from G_2 in sync with I_6). This leads to the removal of I_4 from bearer G_1, initiating the addition process for unallocated devices (I_4 and I_6).

By following the steps in the proposed algorithm, we attempt to merge these devices into existing bearers. The priority is given to adding I_6 to bearer G_1 due to its highest addition score (with +3 shifting). Subsequently, I_4 is added to bearer G_2 with +3 shifting. This procedure results in the distribution of these devices' traffic across two bearers, as depicted in Fig. 13.

It is important to note that in the previous scenario, if we attempted to merge the new MTD without eliminating any existing MTDs (like I_4 in this instance), we would end up with a total of 3 bearers. This demonstrates the advantage of the smart removal process in further reducing the count of active bearers.

Furthermore, it is worth mentioning that when we apply the ILP-based solution to this example, we achieve the same number of bearer usages in each scenario as we do with the heuristic-based approaches.

Although the proposed algorithms might not always discover the optimal solution like the ILP solutions do, the simulation results will demonstrate that they can produce outcomes that are very close to optimal across most situations.

In order to evaluate the performance of the proposed solutions, we perform simulations in different settings. We also compare the heuristic based approximate solutions with the optimal solutions obtained by CPLEX from ILP models.

3.3.6 Settings

In accordance with the traffic model outlined in Section 3.4, we initiate the process by generating data upload traffic for each MTD. This is achieved by first selecting a data upload interval (λ_i) at random from the set $10, 20, 40$ minutes. Subsequently, we assign a data communication duration, $\delta_i = s_i - e_i$, within each data upload interval using three distinct traffic load models. In the scenario of low traffic load, we consider 10-15% of the data sending interval or λ_i for data communication. For medium and high traffic loads, we utilize 15-25% and 25-50% of the interval, respectively. The start time of data upload (s_i) within the data sending interval is randomly determined from the range $[0, \lambda_i - \delta_i]$. The end time of data communication is automatically established as $e_i = s_i + \delta_i$.

Throughout the simulations, we vary the count of MTDs from 5 to 500. Specifically, for the sake of comparison with the optimal outcomes derived from the ILP-based approach, we utilize smaller values of M since obtaining ILP results becomes time-consuming with a larger number of MTDs. To get the results on heuristics, we investigate MTD counts as high as 500 and analyze the impact of different parameters. In dynamic network scenarios, we also consider a Dynamicity level defined as the percentage of devices that join or leave during each moment. In the primary simulations, we assume an equal number of joinings and departures (i.e., $x = y$) during

	Traffic Load		
Parameter	**Low**	**Medium**	**High**
Data communication per interval (δ in % within λ)	10-15%	15-25%	25-50%
Number of MTDs (M)	5-500		
Maximum shifting allowed (τ_{max})	0-6 time slots		
Data sending interval (λ) array	{10,20,40} time slots		
Start time for data sending (s_i)	Uniformly distributed in λ_i		
End time of data sending (e_i)	$s_i + \delta_i$ if it is $\leq \lambda_i$		
Dynamicity	10-50%		

Table 5. Simulation parameters and values

every network moment. However, we also explore scenarios where this balance does not hold. The simulation parameters and their respective values are summarized in Table 5.

3.4 Evaluation

3.4.1 Algorithms in Comparison

- *Optimal Initial Aggregation via ILP (ILP-IA)*: This approach solves the ILP-based model presented in (3.1), focusing solely on the initial aggregation phase.

- *Optimal Dynamic Aggregation via ILP (ILP-DA)*: This method solves the ILP-based model from (3.6), accounting for dynamic aggregation across multiple network moments.

- *Heuristic Initial Aggregation (HIA)*: This heuristic algorithm, outlined in Algorithm 1, targets the initial aggregation phase.

- *Heuristic Dynamic Aggregation (HDA)*: This heuristic, detailed in Algorithm 2,

handles dynamic aggregation during various network moments.

- *Heuristic Dynamic Aggregation without Smart Removal (HDA_noSR)*: A variant of the heuristic dynamic aggregation algorithm, excluding the smart removal process (lines 4-22 in Algorithm 2).

We explore both shifting and non-shifting aggregation scenarios for each of these algorithms.

3.4.2 Performance Metrics

We assess the effectiveness of the proposed methods using the following metrics:

- *Percentage of Savings (%)*: This measures the reduction in the number of bearers used. Given a certain number of MTDs M, if the aggregation model determines that X bearers are adequate to handle the traffic from all M devices, the percentage of savings is calculated as $\left(\frac{M-X}{M} \times 100\right)\%$.

- *Percentage of MTDs with Updated IMSI*: This indicates the average percentage of MTDs whose IMSI changes between consecutive moments in dynamic network scenarios. Alterations in group assignments may require some MTDs to switch groups, leading to control data traffic for assigning new IMSI numbers. Minimizing these changes is a secondary goal after maximizing savings. Only MTDs present in both network moments are considered, and the percentage is calculated as $\left(\sum_{t=2}^{Z}\left(\frac{\sum_{\forall I_i \in G^t} \text{diff}_i}{|G_t|}\right)\right) / (Z-1)$, where Z is the number of moments in the network with varying MTDs, and G^t represents the set of MTDs at moment t.

- *Running Time*: To showcase scalability, we present the running times of the algorithms as the number of MTDs increases. Testing is conducted on an Intel

Core i7 processor with 16 GB memory and a 2.5 GHz clock speed.

We investigate the impact of different traffic load models, MTD counts, network dynamics, and maximum allowable shifting (τ_{max}) on these metrics. The presented results are averaged over 20 iterations.

Please note that although prior studies like [20, 19] explored aggregated IoT communication without shifting, their approach assumes that only MTDs with identical data sending intervals (λ) and communication durations (δ) share the same bearer. These works mainly focus on modifying call flows to achieve aggregated communication through IMSI sharing. In contrast, our study addresses the challenge of grouping IoT devices with varying traffic patterns, as well as updating these groupings in dynamic environments. While we cannot directly compare our solutions with existing literature due to differences in methodology, we treat the no-shifting case as a benchmark solution by applying our methods with $\tau_{max} = 0$. This allows us to measure the added benefits of shifting-based models. Additionally, we analyze the advantages of the proposed smart removal process, which considers previous moment's group structure, in comparison to independent groupings for each moment.

3.4.3 Results

3.4.3.1 Comparison of ILP and Heuristic Solutions

We start by comparing the optimal solutions derived from ILP with those from heuristics. Initially, we focus on the initial aggregation procedure and the arrangement of MTDs upon their initial network entry.

Figures 16, 17, and 18 illustrate the percentage of savings achieved by both the ILP and HIA algorithms across various traffic load models as the MTD count increases. Notably, in all three graphs, the percentage of savings grows as the MTD

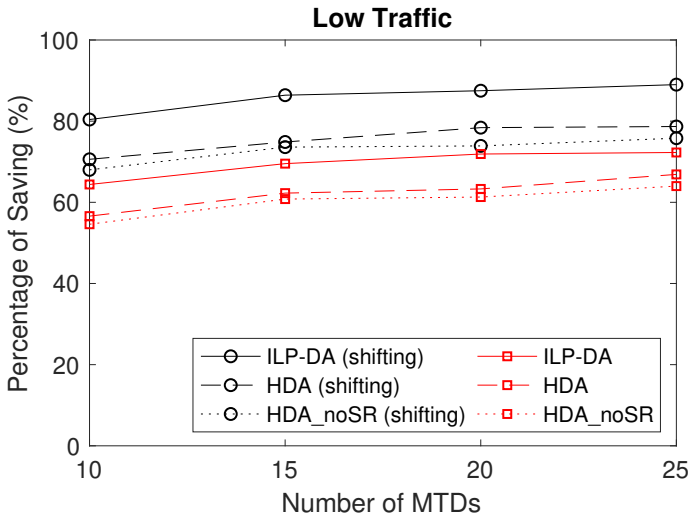


Fig. 14. ILP vs. Heuristic Algorithms in dynamic aggregation: Percentage of saving averaged over 100 moments with 10% dynamicity ($\tau_{max} = 3$ for shifting based aggregation).

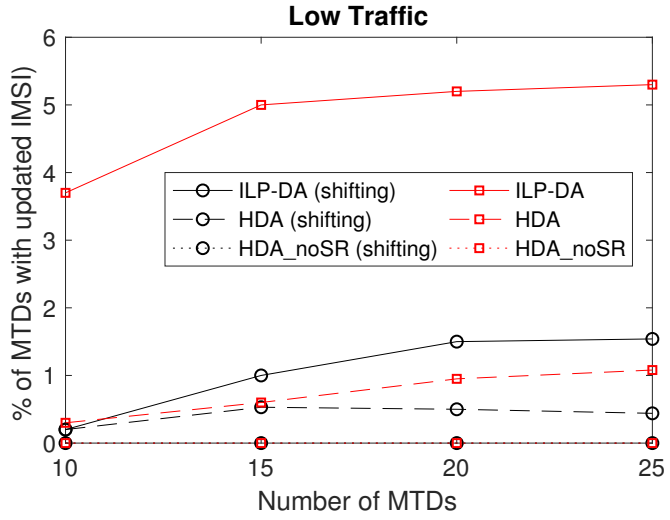


Fig. 15. ILP vs. Heuristic Algorithms in dynamic aggregation: Percentage of MTDs with updated IMSI averaged over 100 moments with 10% dynamicity ($\tau_{max} = 3$ for shifting based aggregation).

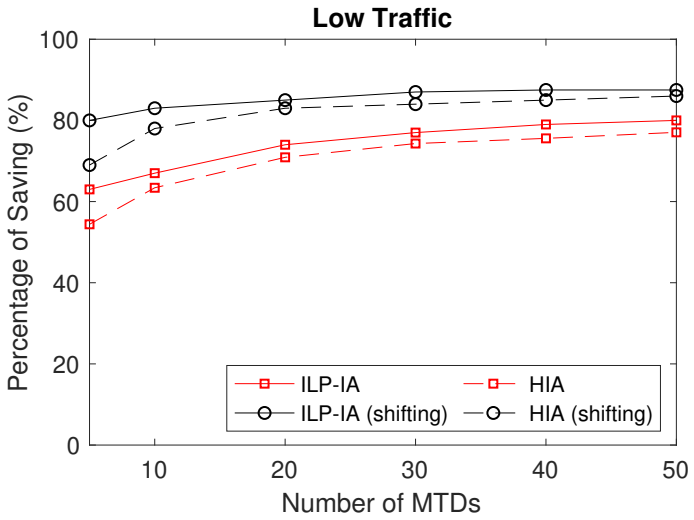


Fig. 16. ILP vs. Heuristic Algorithms in initial aggregation: Percentage of saving in the initial network moment with low traffic models ($\tau_{max} = 3$ for shifting based aggregation).

count rises and eventually stabilizes. Although we did not pursue results beyond 50 MTDs due to the long execution time of ILP, this is unnecessary since the savings reach convergence. When comparing the attained savings, the highest percentage is evident when the traffic load remains low. Furthermore, across all scenarios, the percentage of savings ascends with an increase in MTD count. This is attributed to the greater potential for grouping multiple MTDs into a single bearer under low traffic conditions. However, the rate of saving enhancement fluctuates among different traffic loads.

When contrasting solutions based on shifting versus no shifting, the superiority of shifting in terms of savings is conspicuous in all instances, courtesy of the adaptable data upload times of MTDs. Analyzing the comparison between ILP and heuristic outcomes, it becomes apparent that heuristics can approximate ILP results quite closely. Nonetheless, the gap between heuristic and ILP outcomes widens notably in high traffic scenarios, where finding improved groupings becomes more challenging

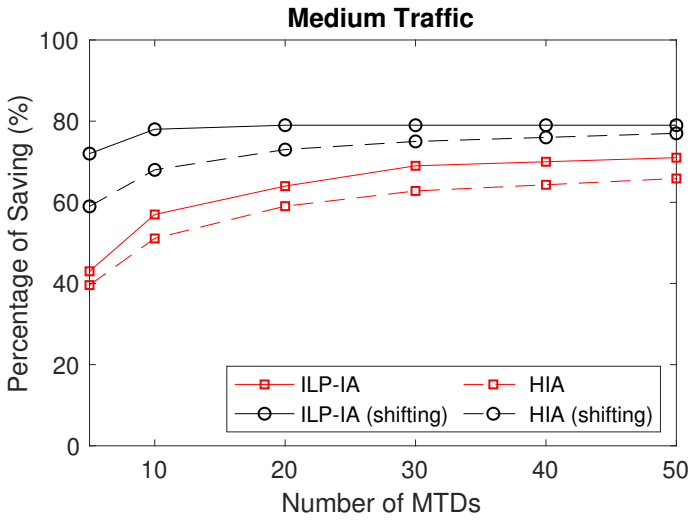


Fig. 17. ILP vs. Heuristic Algorithms in initial aggregation: Percentage of saving in the initial network moment with medium traffic models ($\tau_{max} = 3$ for shifting based aggregation).

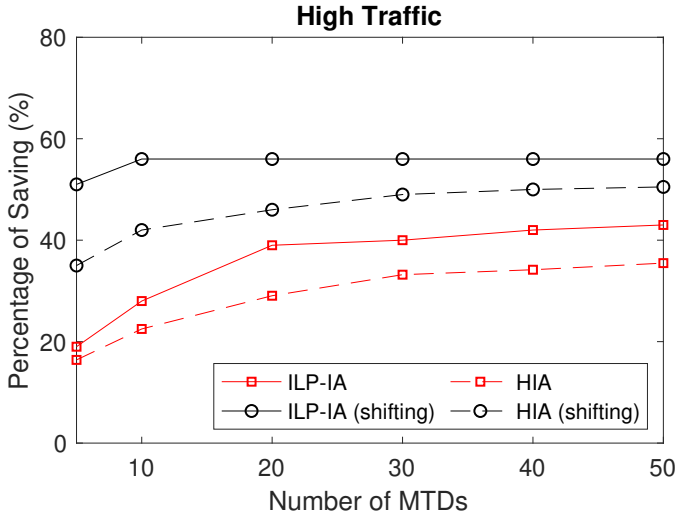


Fig. 18. ILP vs. Heuristic Algorithms in initial aggregation: Percentage of saving in the initial network moment with high traffic models ($\tau_{max} = 3$ for shifting based aggregation).

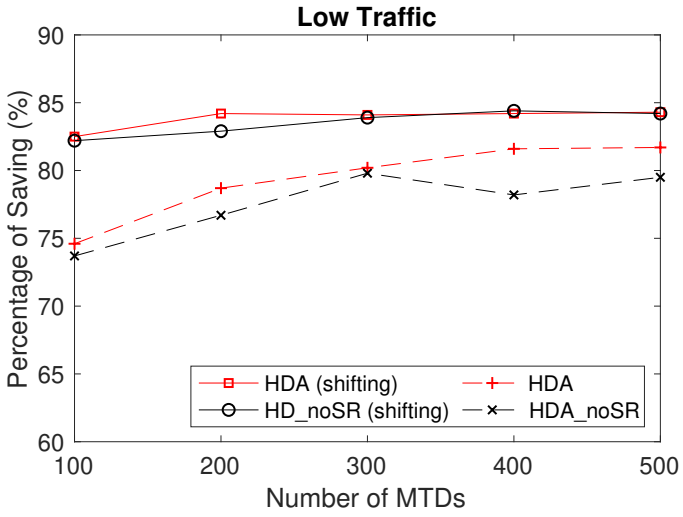


Fig. 19. **Impact of MTD count:** The percentage of savings in dynamic environments (10% dynamicity) with low patterns ($\tau_{max} = 3$)

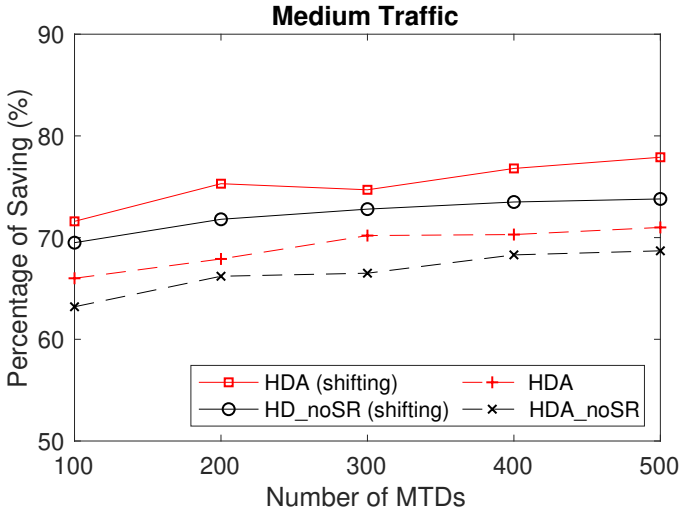


Fig. 20. **Impact of MTD count:** The percentage of savings in dynamic environments (10% dynamicity) with medium patterns ($\tau_{max} = 3$)

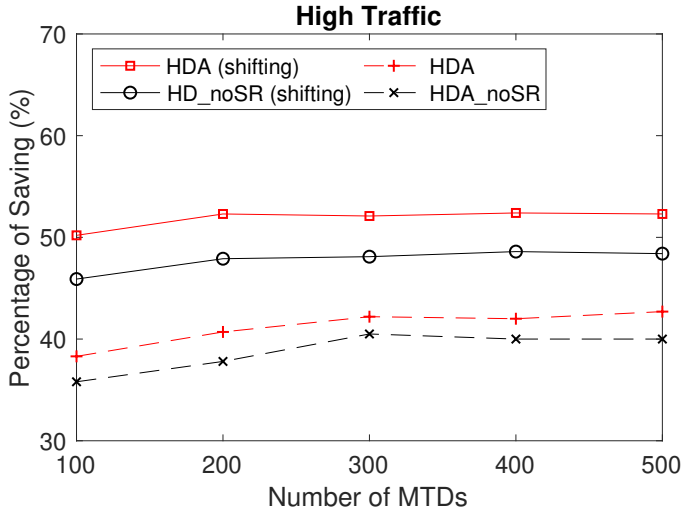


Fig. 21. **Impact of MTD count:** The percentage of savings in dynamic environments (10% dynamicity) with high patterns ($\tau_{max} = 3$)

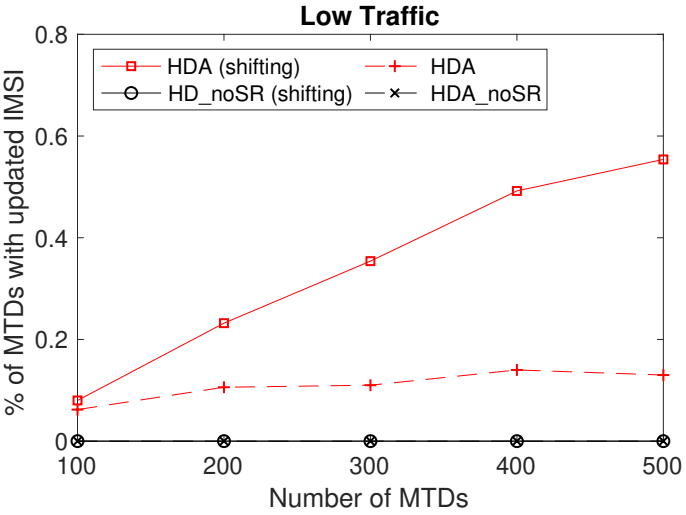


Fig. 22. **Impact of MTD count:** The percentage of savings with updated IMSI counts in dynamic environments (10% dynamicity) with low patterns ($\tau_{max} = 3$)

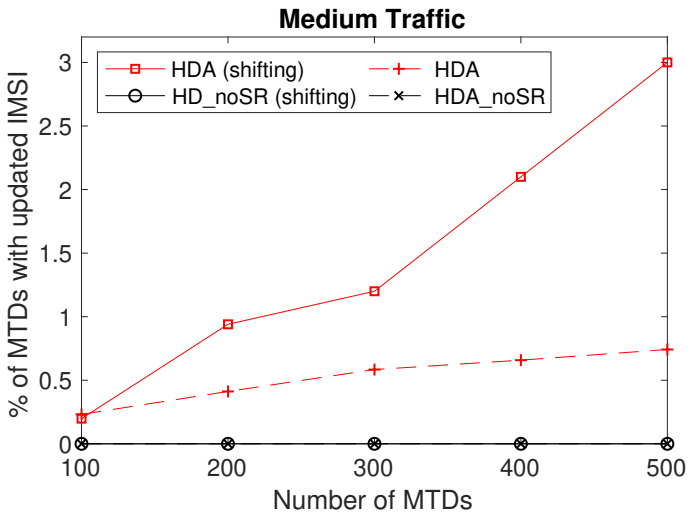


Fig. 23. **Impact of MTD count:** The percentage of savings with updated IMSI counts in dynamic environments (10% dynamicity) with medium patterns ($\tau_{max} = 3$)

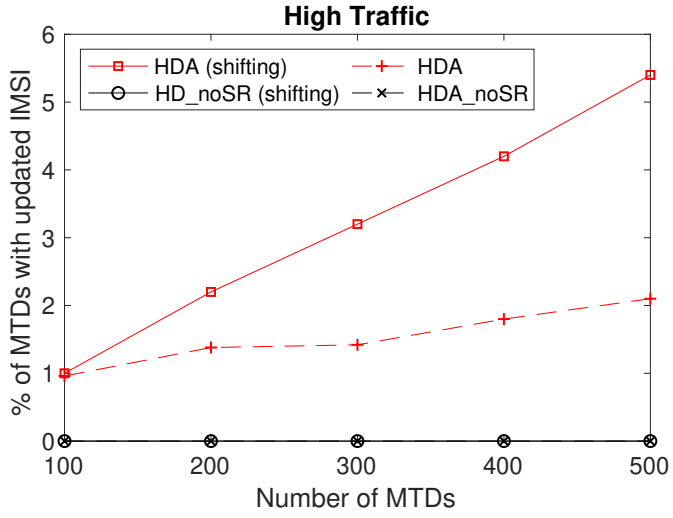


Fig. 24. **Impact of MTD count:** The percentage of savings with updated IMSI counts in dynamic environments (10% dynamicity) with high patterns ($\tau_{max} = 3$)

for the heuristic method due to the heightened utilization of MTD timelines.

Moving forward, we compared the ILP solution with heuristic-based approaches in dynamic network settings. Figures 14 and 15 demonstrate the outcomes for varying MTD counts within the low traffic model (similar trends are observed with other models). To manage computational time, we present results up to 25 MTDs, given that executing ILP-DA takes significantly more time than ILP-IA. For each MTD count, we conducted simulations across 100 instances, with 10% of existing MTDs leaving the network while an equivalent number of new MTDs with distinct traffic patterns join the network in each instance.

Looking at the saving results in Fig. 14, we notice a similar relation as in Fig. 16, but the gap between ILP-DA and HDA is a bit larger. This discrepancy may arise from the fact that while ILP-DA can yield marginally higher savings than ILP-IA, it necessitates more IMSI changes among MTDs during different instances, as depicted in Fig. 15. The HDA-noSR algorithm, which excludes the smart removal process, exhibits slightly diminished savings compared to HDA, underscoring the advantage of the smart removal procedure. We will further demonstrate that this benefit becomes more pronounced as the network accommodates a greater number of MTDs. Additionally, it is noteworthy that the HDA-noSR algorithm only introduces new joining MTDs to existing bearers or establishes new ones, consequently avoiding IMSI updates for existing MTDs.

Figures 25, 26, and 27 show how the parameter τ_{max} influences the percentage of savings. For this comparison, we once again consider a small MTD count ($M = 20$) to be able get ILP outcomes. While the results for no shifting remain consistent, they are included to distinctly showcase the advantages of shifting-based models over this benchmark approach.

Observing the data, we note that a rise in the threshold leads to increased savings

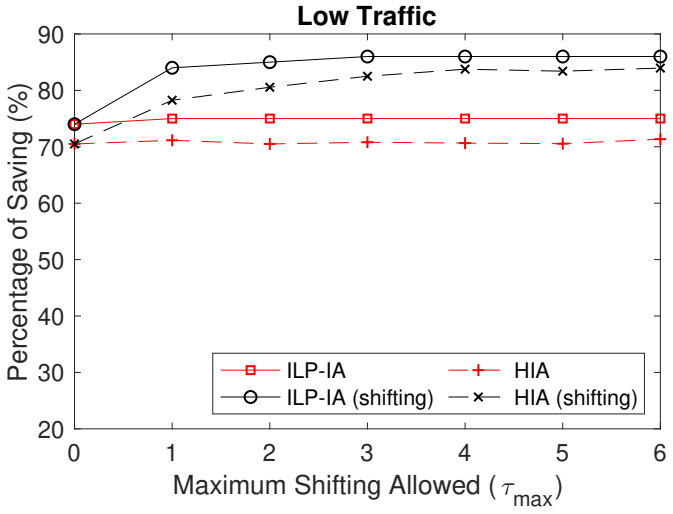


Fig. 25. **ILP vs. Heuristic Algorithms** with different maximum shifting threshold ($\tau_{\max}$): Percentage of saving with low traffic patterns ($M = 20$, $\tau_{max} = 3$).

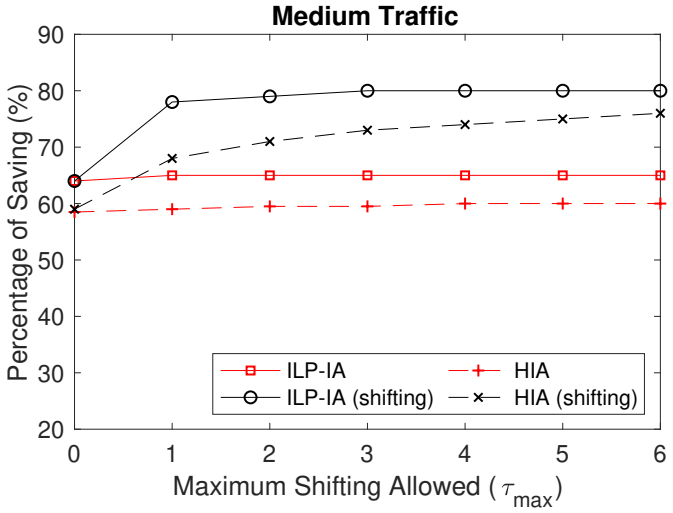


Fig. 26. **ILP vs. Heuristic Algorithms** with different maximum shifting threshold ($\tau_{\max}$): Percentage of saving with medium traffic patterns ($M = 20$, $\tau_{max} = 3$).

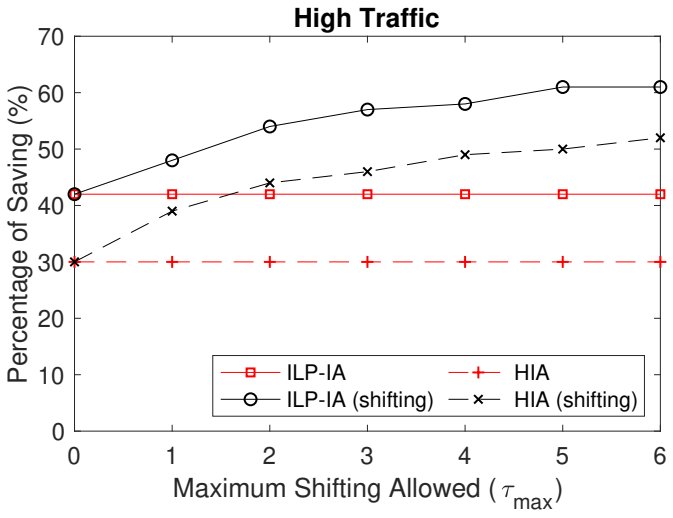


Fig. 27. **ILP vs. Heuristic Algorithms** with different maximum shifting threshold ($\tau_{\max}$): Percentage of saving with high traffic patterns ($M = 20$, $\tau_{max} = 3$).

across all traffic load models. Nonetheless, this enhancement converges more swiftly in low traffic, followed by medium traffic, and relatively more slowly in high traffic scenarios. This trend emerges because higher traffic density restricts maneuverability within group arrangements, allowing maximum benefits with greater flexibility achievable through larger thresholds.

3.4.3.2 Impact of MTD Count

To demonstrate the performance of the suggested algorithms with a greater quantity of MTDs, we also test our algorithms and get outcomes for MTD counts ranging from 100 to 500, in steps of 100. These findings are presented for dynamic scenarios and exclude ILP results due to extended processing times.

In Figures 19, 20, 21, 22, 23, and 24, we depict both the proportion of savings and the proportion of MTDs with updated IMSI for three different traffic models. The savings percentage shown in Figures 19, 20, and 21 illustrate the advantage of shifting, as in Figures 16, 17, and 18, respectively. Furthermore, the savings tend to

remain relatively steady across the various traffic models.

When comparing the HDA and HDA_noSR algorithms, we also notice more advantage of the smart removal process incorporated in HDA as the traffic load increases. This observation holds true for scenarios involving both shifting and no shifting. Nevertheless, as depicted in Figures 22, 23, and 24, this improvement is accompanied by changes in the IMSI assignments of MTDs. For instance, in a high traffic model with shifting and when M equals 500, HDA yields approximately 10% higher relative savings compared to HDA_noSR. However, this enhancement leads to about 5.4% of MTDs updating their IMSI between consecutive time points. In contrast, as anticipated, HDA_noSR does not trigger any updates in IMSI assignments for existing MTDs.

3.4.3.3 Impact of Dynamicity

In Figures 28,29,30,31,32, and 33, we examine the outcomes under varying levels of dynamicity behavior between successive time points. Specifically, we explore dynamicity ranging from 10% to 50%. In the case of an initial network with 500 MTDs, this translates to 50 and 250 MTDs joining or departing at each time point, respectively. When considering the percentage of savings illustrated in Figures 28, 29, and 30, we note a relatively consistent level of savings across all scenarios. Once again, HDA outperforms HDA_noSR in terms of savings, and the shifting approach contributes to further increased savings. Conversely, the HDA approach leads to changes in IMSI assignments due to its smart removal process, as depicted in Figures 31, 32, and 33.

It is important to highlight that as the percentage of dynamicity increases, the proportion of MTDs with updated IMSI also rises. In certain instances, this proportion can become substantial, resulting in heightened control traffic for the allocation

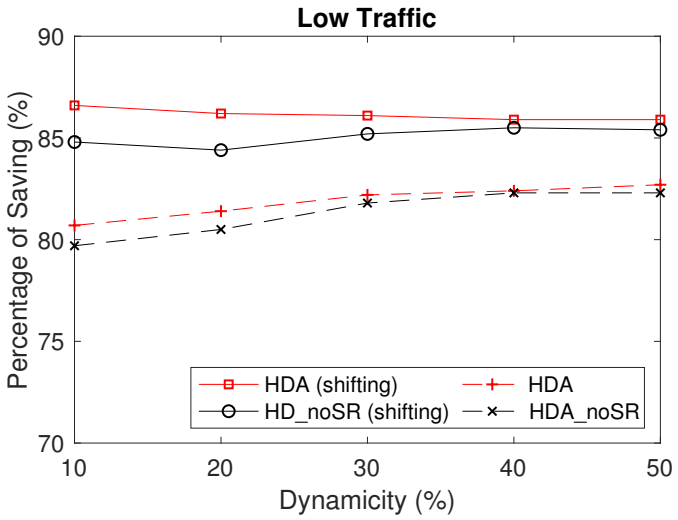


Fig. 28. **Impact of Dynamicity:** The percentage of savings with low traffic patterns ($\tau_{max} = 3$, $M = 500$)

of new IMSI numbers. However, in a practical setting, even a dynamicity level as low as 10% can be considered significant. Our observations indicate that with 10% dynamicity, the proportion of MTDs requiring updated IMSI remains relatively low, ranging from 0.2% to 5%.

3.4.3.4 Impact of Data Sending Interval Array

In Fig. 35, we examine how the selection of data sending interval arrays for MTDs affects the outcomes. As depicted in the figure, when there are more options available and larger values of λ are utilized, the level of savings diminishes for all algorithms. However, in all cases, shifting as well as the smart removal process considered in HDA help increase the saving.

On average, approximately 2% of MTDs experience updates in their IMSI between consecutive time points. This corresponds to around 10 MTDs, thus may not cause too much control traffic.

In the initial version of this paper [60], we also provided a comparison of these

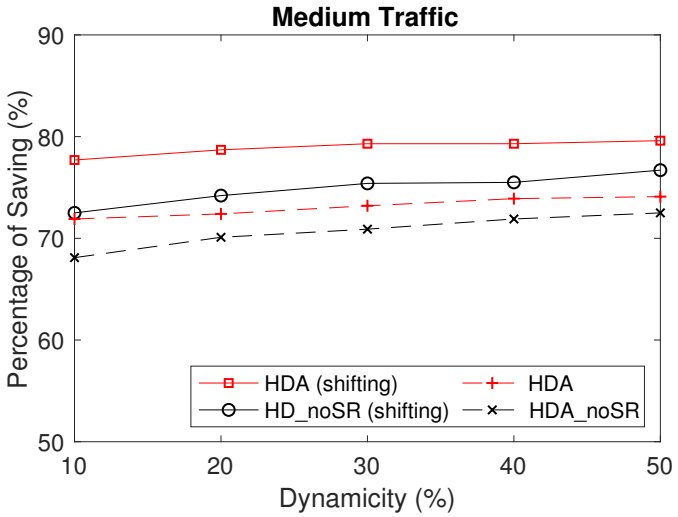


Fig. 29. **Impact of Dynamicity:** The percentage of savings with medium traffic patterns ($\tau_{max} = 3$, $M = 500$)

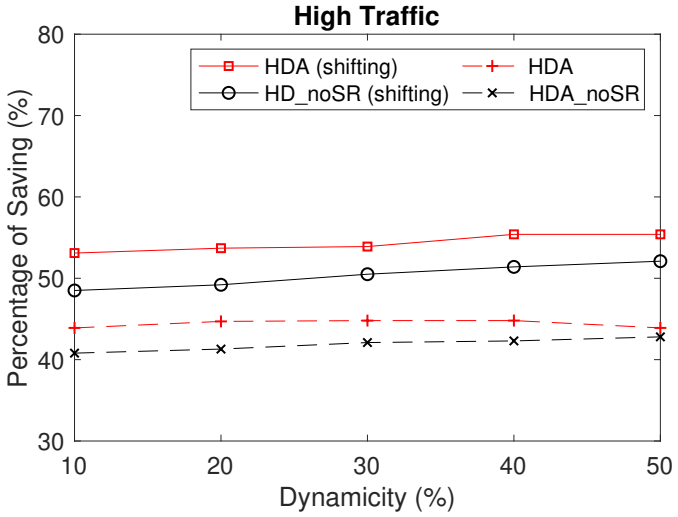


Fig. 30. **Impact of Dynamicity:** The percentage of savings with high traffic patterns ($\tau_{max} = 3$, $M = 500$)

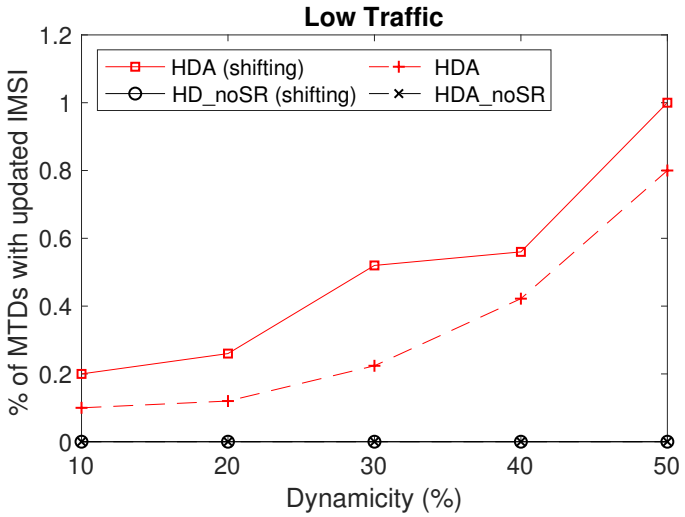


Fig. 31. **Impact of Dynamicity:** The percentage of savings with low traffic patterns ($\tau_{max} = 3$, $M = 500$) with updated IMSI counts

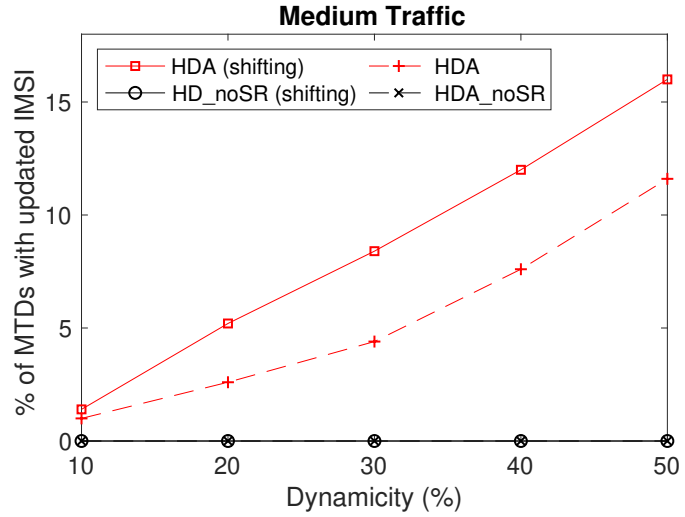


Fig. 32. **Impact of Dynamicity:** The percentage of savings with medium traffic patterns ($\tau_{max} = 3$, $M = 500$) with updated IMSI counts

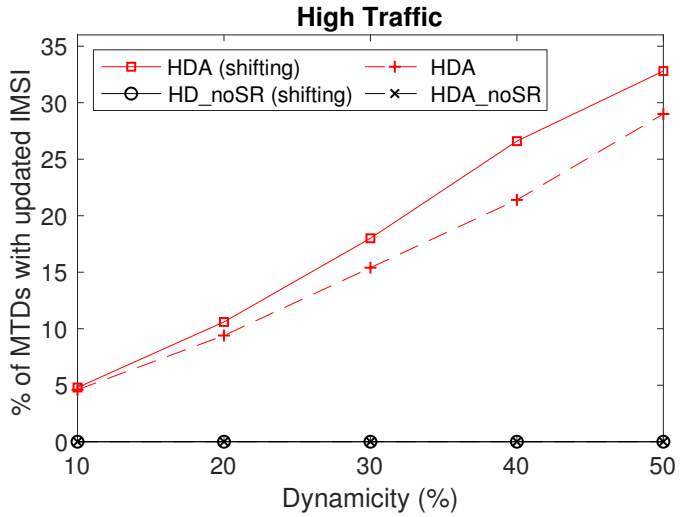


Fig. 33. **Impact of Dynamicity:** The percentage of savings with high traffic patterns ($\tau_{max} = 3$, $M = 500$) with updated IMSI counts

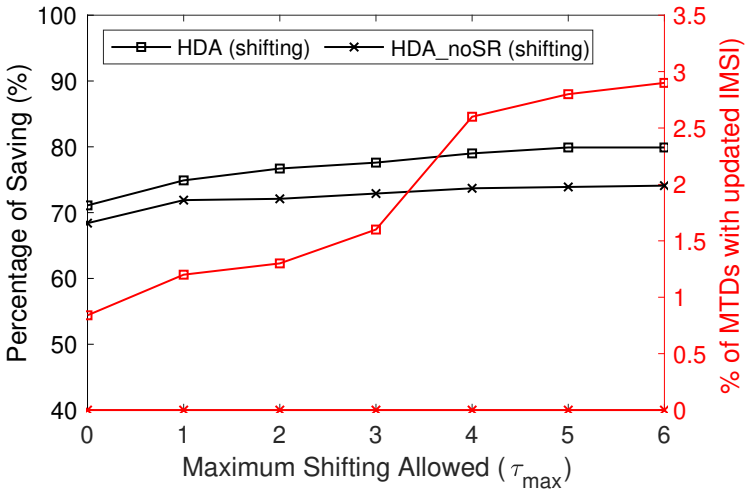


Fig. 34. **Impact of various parameters:** Percentage of saving with different τ_{max} in dynamic scenarios (medium traffic, M=500, τ_{max}=3).

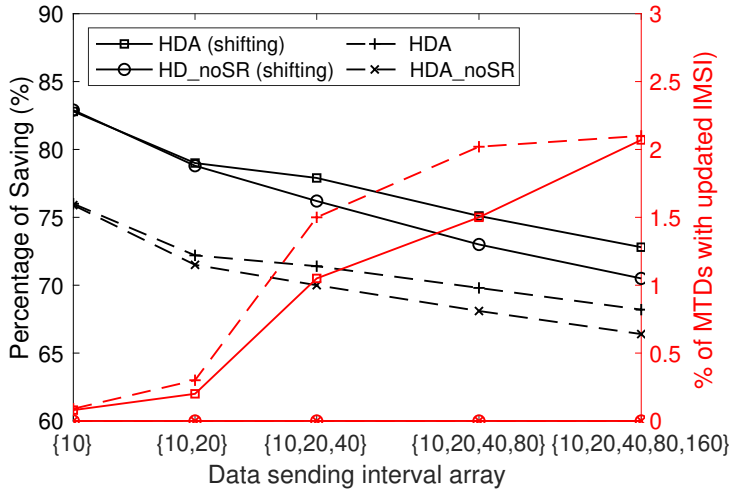


Fig. 35. **Impact of various parameters:** Percentage of saving with different data sending interval arrays in dynamic scenarios (medium traffic, M=500, τ_{max}=3).

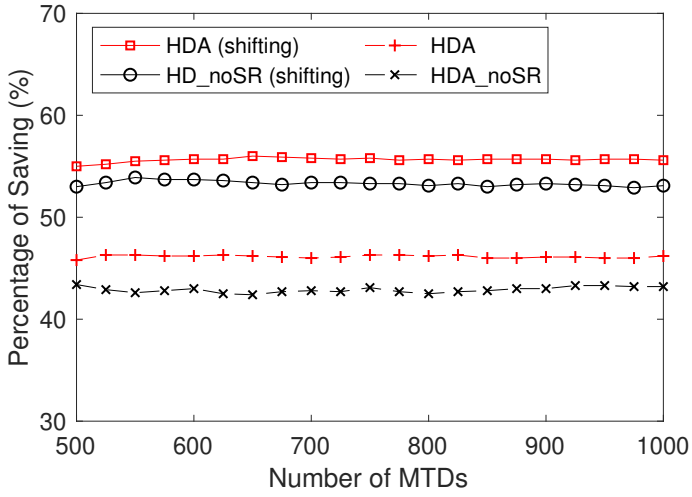


Fig. 36. **Impact of various parameters:** Percentage of saving in a growing network (high traffic).

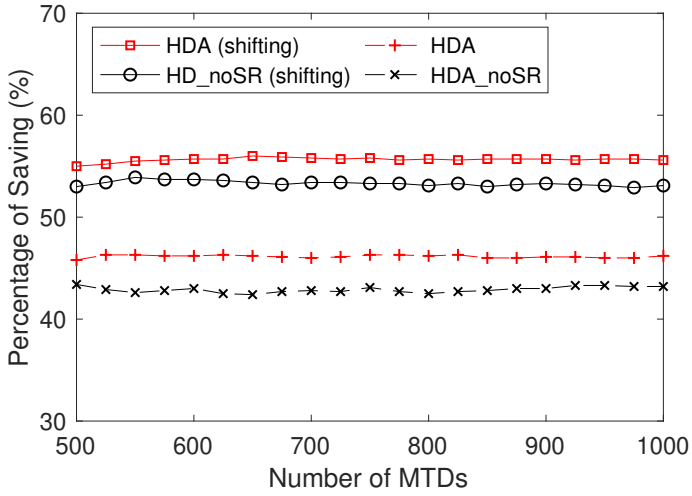


Fig. 37. **Impact of various parameters:** Percentage of saving in a growing network (high traffic).

outcomes with ILP, using a smaller count of MTDs. This comparison showcased how heuristic algorithms can yield outcomes that are close to those produced by ILP.

3.4.3.5 Running Time Comparison

In Figures 38, 39, and 40, we compare how long different algorithms take to run. We're focusing on heuristic-based algorithms because the ILP solutions take too much time, as shown in [60], so we're not considering them here. These algorithms deal with a changing environment, and we run the HIA algorithm independently for each moment, imagining the network starts fresh at that time without remembering the past. This is based on the algorithm described in [60], without accounting for the changing environment.

In Fig. 38, we first compare runtimes using different arrays for when data is sent. The results show that running HIA every moment takes much longer compared to HDA and HDA_noSR, whether or not we consider shifting. Shifting makes all algorithms take more time because we need to check various combinations to make

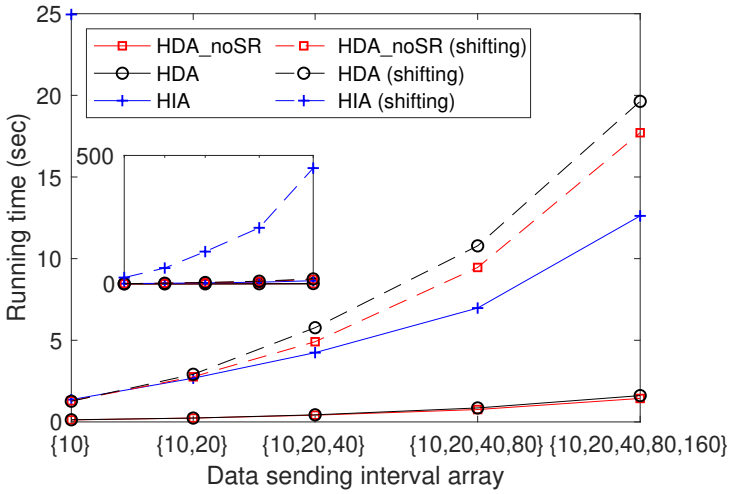


Fig. 38. **Average running time comparison:** Different data sending interval arrays (medium traffic, M=500, τ_{max}=3, dynamicity = 10%).

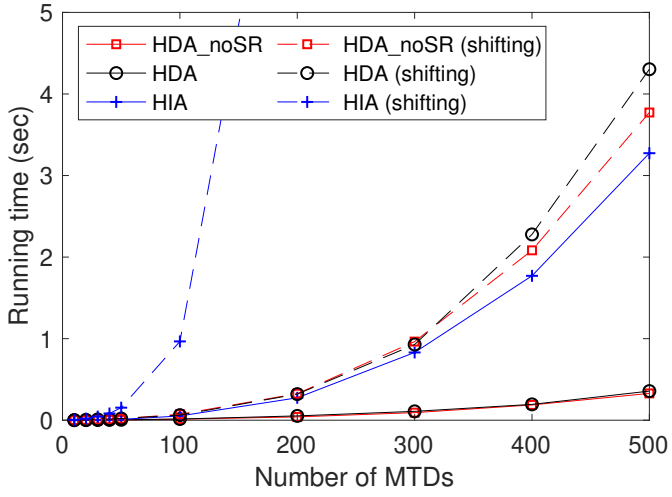


Fig. 39. **Average running time comparison:** Different number of devices in the network (medium traffic, M=500, τ_{max}=3, dynamicity = 10%).

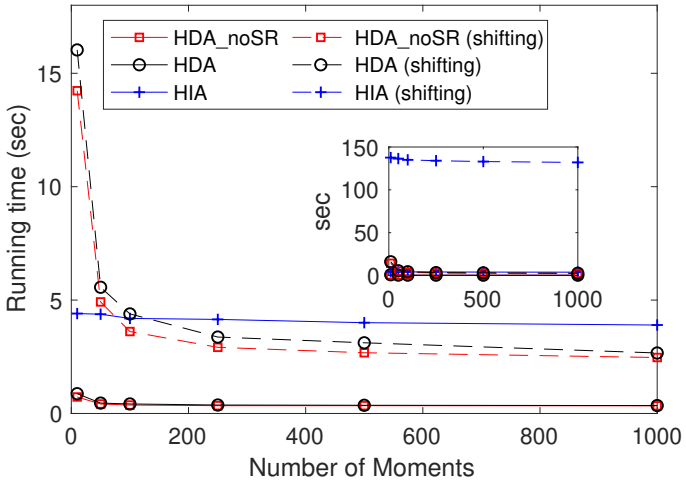


Fig. 40. **Average running time comparison:** Different number of network moments (medium traffic, M=500, τ_{max}=3, dynamicity = 10%).

the most of shifting's flexibility. Comparing HDA and HDA_noSR, HDA takes a bit more time due to its smart removal process, but as we saw earlier, HDA can save up to 10% more than HDA_noSR.

Moving to Fig. 39, we compare runtimes with different numbers of MTDs (assuming 10% change). The order of algorithms in terms of runtime is similar to Fig. 38. With more MTDs, HIA's runtime increases a lot compared to HDA and HDA_noSR. HDA and HDA_noSR have similar runtimes, but HDA offers more savings. Finally, in Fig. 40, we show the average runtimes per moment for different numbers of network moments. As the results indicate, the average runtimes of HDA and HDA_noSR decrease as the number of network moments increases. This is because these algorithms have a high initial grouping running time (using HIA) at the start. Over more moments, their average runtime per moment becomes lower because of the regrouping algorithm used. HDA's average runtime is slightly higher than HDA_noSR due to the smart removal process, but it is still much lower than starting the grouping process (HIA) from scratch at every moment.

3.5 Experimental Proof-of-Concept for IMSI Sharing of IoT Devices

In this section, we create and execute a system for combining traffic from multiple mobile devices that use the same subscriber identity and network resources within the mobile core network. We achieve this by utilizing commercial smartphones, programmable SIM cards, and the Amarisoft Callbox [75], which includes both core network and eNB components. Through this demonstration, we showcase how we can set up and execute an experiment in which each of these devices shares a copy of the same SIM card and connects to the core network at different times for downloading tasks. The outcomes of this experiment indicate that by sharing core network resources in this manner, we can use them efficiently compared to having separate cellular connections for each device. This approach holds promise for enhancing communication in massive IoT scenarios.

Subscriber identity sharing, known as International Mobile Subscriber Identity (IMSI) sharing for connection and communication [19, 60, 76], aims to make efficient use of mobile core network resources for IoT devices with low data rates and infrequent communication needs. For instance, devices like moisture sensors in agricultural fields may only need to transmit data twice a day. This efficiency is achieved by providing a group of IoT devices with a shared Subscriber Identity Module (SIM) profile that uses the same IMSI number, enabling them to take turns connecting to the core network and performing their data communication.

When it is a device's turn, it registers/attaches to the core network, sets up the connection, completes its data transfer tasks, and then deregisters/detaches from the core to free up resources, similar to turning off a phone. Consequently, the core network treats these connections as if they were coming from a single device. This approach differs from devices going idle when not actively transmitting data, as idle

devices still remain registered in the core and consume memory resources. Moreover, this solution offers a more scalable traffic aggregation compared to the commonly used method of connecting multiple IoT devices through a local IoT gateway. IMSI sharing allows aggregation for any device within the core network's service area, which can cover hundreds of base stations, whereas the latter method only permits aggregation among nearby devices.

This IMSI sharing system has been studied in recent works [19, 60, 76]. These studies have focused on the necessary changes in call flows and core network architecture to implement IMSI sharing in existing deployments. Additionally, they have addressed the challenge of efficiently grouping IoT devices based on their traffic patterns. However, it is worth noting that these studies have relied solely on simulations for evaluation.

In this research, we present a groundbreaking realization of this system through experiments involving readily available devices and a commercial core network for the first time.

3.5.1 Design and Implementation

Our laboratory setup is illustrated in Fig. 41. To implement the proposed system, we utilized Amarisoft Callbox Classic, which incorporates the core functionalities outlined by 3GPP (Third Generation Partnership Project), including LTE and 5G NSA (Non-Standalone), with all the necessary components and antennas for connecting UE.

The Amarisoft machine's core network settings offer three distinct algorithms for authentication and key generation: XOR, $MILENAGE$, and $TUAK$. The SIM cards we obtained from Amarisoft shared the same IMSI number and used the XOR

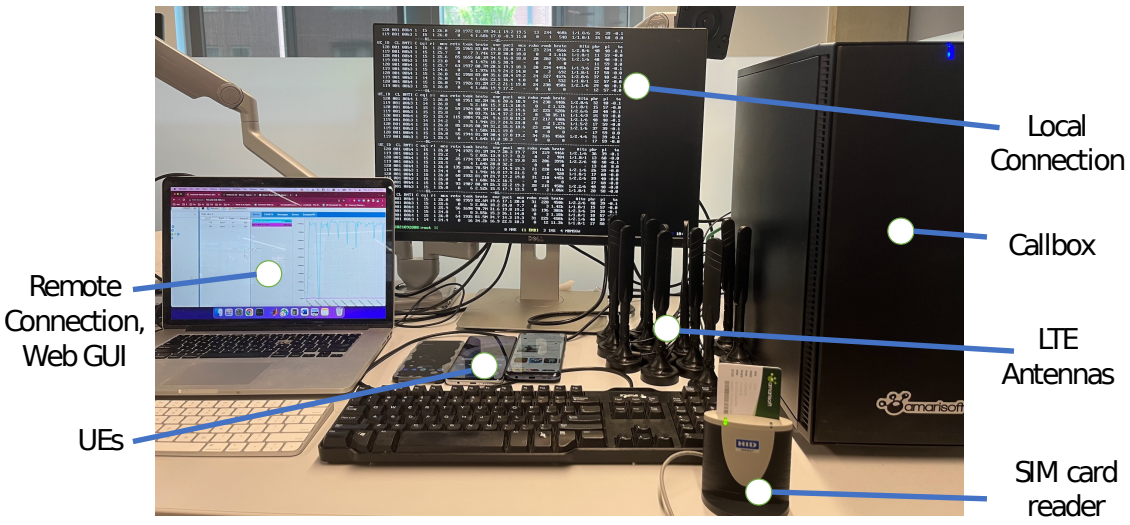


Fig. 41. Our lab setup with Amarisoft core network, LTE antennas, and smartphones.

algorithm. Therefore, when testing the scenario where all devices had the same IMSI number, we used the SIM cards in their default settings. However, for the scenario where each device had a different IMSI, we reconfigured the SIM cards with the $MILENAGE$ algorithm. Unfortunately, reconfiguring SIM cards with different IMSI numbers under the XOR algorithm did not allow them to connect to the core network, and we are actively investigating this issue. The $MILENAGE$ algorithm could also have been used to reconfigure SIMs with the same IMSI but different from the default IMSI number. We employed the HID OMNIKEY smart card reader and the PySim application for reconfiguring the SIM cards.

In the XOR algorithm configuration (located in the */root/enb/config/enb.cfg* file), there exists a specific parameter known as *multi_sim*. This parameter enables multiple devices with the same IMSI but different IMEI to connect to the core network simultaneously. It is important to note that this feature is intended for testing environments and not for real-world deployments. It utilizes both IMSI and IMEI for

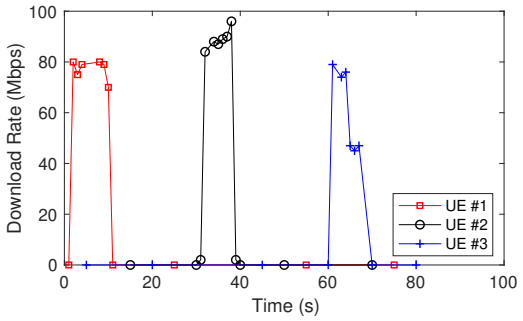
identifying each UE. To ensure that only one device with the same IMSI is actively connected to the core, we set this parameter to false.

For our experiments, we used three smartphones, each representing an IoT device: OnePlus Nord N10 (5G), Google Pixel 5 (5G), and Motorola One 5G ACE.

To automate the registration, deregistration, and data communication processes on these smartphones, we employed an application called Click Assistant [77] along with the airplane mode feature. Click Assistant enables the modeling of automated screen interactions at predefined locations following a repetitive schedule.

Similar to the approach used in [76], we implemented a data traffic model in which each device performs data uploads/downloads for a duration of δ time units at intervals of λ time units. This process starts at time s and concludes at time e within each λ duration. Specifically, we utilized three UE instances, each of which downloaded data during the time intervals of 0-10 seconds, 30-40 seconds, and 60-70 seconds, recurring every 100 seconds (λ).

All devices initially begin with the airplane mode activated. When it is a device's turn, such as the first UE's turn, the Click Assistant app interacts with the airplane mode symbol to deactivate it, allowing the UE to register/attach to the network. Subsequently, Click Assistant opens the Google Play app and initiates the download of a large file. After 10 seconds, Click Assistant interacts with the airplane mode sign to reactivate it, thereby stopping the download and detaching the UE from the network.



Total: 3, registered: 1, LTE: 3

IMSI	IMEISV	RAT	M-TMSI	Reg.
001010123456789	9900168402492632	LTE	0x48601d94	true
001010123456789	3555721104828425	LTE	0x813e8165	false
001010123456789	3524931155344245	LTE	0xf61c214e	false

Fig. 42. (i) Three UEs with the same IMSI connect to the core at different times to download data without overlap. (ii) Registration status from 0-10 sec at the core.

3.5.2 Experimental Results

3.5.2.1 Download Rates and Core Memory Usage

In Fig. 42, we illustrate the download speeds achieved by each UE throughout the duration of the experiment, where the UEs share the same IMSI and connect sequentially. Each UE is capable of reaching and sustaining a download speed of approximately 80 Mbps for their 10-second connection duration. The process of connecting to the network is extremely rapid, ensuring that downloads commence without noticeable delays. When the UEs are detached from the network, their download rates drop to zero. The accompanying table in the figure provides further insight, showing that only the first UE is registered during the initial 10 seconds, confirming that memory resources at the core are allocated to just one UE at a time.

3.5.2.2 Core CPU Usage

In Fig. 43, we present a comparison of CPU usage at the core under two scenarios: when the UEs connect to the network sequentially (i.e., only one UE connected at any given time) and when all three UEs connect to the network simultaneously

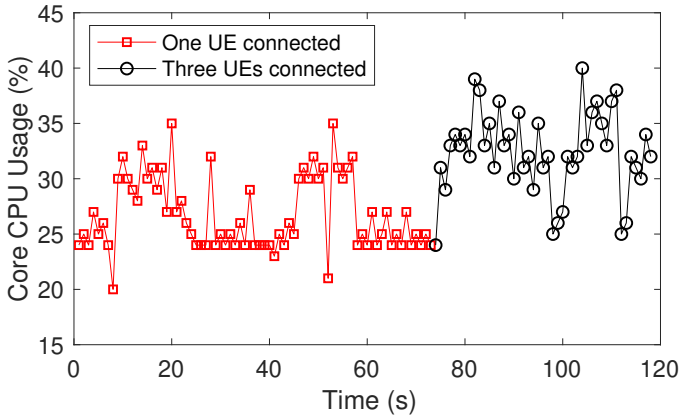


Fig. 43. CPU utilization in the core network when one UE is connected and three UEs are connected at the same time.

(either using different IMSIs or the same IMSI with the XOR algorithm). The figure illustrates that connecting all three UEs concurrently leads to an increase in CPU utilization (attributed to resource management in both the eNB and the core network), with an average usage of 32.4%. Conversely, when only one UE connects at a time, the CPU usage averages around 26.7%. These measurements were taken while the UEs were performing identical download tasks. Notably, in the case of three concurrent UEs, we also observed download speeds of approximately 26-29 Mbps per UE, whereas in the single UE scenario, the download speed was approximately 80 Mbps.

3.6 Summary of Contributions

In this chapter, we delve into an innovative approach, examining a traffic-shifting-based aggregated communication model designed for IoT devices operating in dynamic environments. This newly devised aggregated communication model not only enables devices to share a common subscriber identity (referred to as IMSI) and take turns during their communication cycles but also integrates a subtle traffic pattern adjustment, known as shifting, into the original device traffic flow. This shifting as-

pect aims to further optimize resource utilization, specifically focusing on minimizing the number of actively utilized bearers within the core network.

Our study scenario unfolds in a dynamic environment characterized by devices frequently entering and exiting the network, effectively creating distinct network moments. Our primary objective is to maximize traffic aggregation from these devices while simultaneously ensuring stability in bearer assignments and IMSI usage, even as the composition of devices within the network undergoes continuous changes.

In pursuit of these goals, we initially formulate solutions based on ILP. However, to mitigate the time computational complexity in ILP approaches, we propose a set of heuristic-based aggregation algorithms offering significantly reduced computational demands. Through extensive simulation experiments, we reveal that these heuristic-based solutions yield results that closely approximate those obtained through ILP techniques, all while imposing significantly lower computational overhead.

Furthermore, our exploration highlights that the incorporation of a "smart removal" process between consecutive network moments brings additional benefits. The comprehensive results underscore the efficacy of our proposed HDA algorithm. Notably, the HDA algorithm demonstrates scalability and efficiency, making it particularly well-suited for operation within dynamic and ever-changing network environments.

In addition, we conducted an experiment involving mobile devices, each equipped with a copy of the same SIM card. These devices connected to the core network at different intervals to carry out download tasks. We demonstrated the automation of these sequential connections using readily available smartphones and an auto-clicking application. Data collection was performed using the Amarisoft core network. The outcomes of our experiment indicate that this approach to aggregating traffic not only decreases the memory and CPU resources utilized at the core but also offers a

promising solution for enhancing communication efficiency in scenarios involving a large number of IoT devices.

CHAPTER 4

COLLABORATIVE OUTAGE-AWARE UAV PATH PLANNING

In this chapter, we address the challenge of optimizing the trajectories of UAVs to ensure continuous network connectivity during missions. This research introduces a collaborative approach where multiple UAVs act as relays for each other to minimize mission time. We present both a nonlinear programming-based solution and a faster graph-based approximation, demonstrating that the proposed approach offers efficient trajectory optimization for multi-UAV scenarios.

4.1 Introduction

UAVs have recently been utilized in many different applications such as surveillance and communication [78]. In order to benefit from UAVs truly in practice, however, it is significant to make sure that UAVs have secure, reliable and low-latency communication links with the ground control stations for their command and control. However, current products in the market today rely on direct LoS communication with their pilots in the ground over nonlicensed spectrum. Thus, a new approach [16] that aims to control UAVs through cellular connection has been considered in order to enhance the performance of UAV based systems. In such a scenario with cellular-enabled UAVs [39], the GBSs provide connection to UAVs so that they can communicate with their pilots on the ground as well as with other UAVs.

In rural areas, using UAVs might be challenging because the cellular coverage might not be enough for good communication between UAVs and GBSs due to limited number of GBSs around. Recent studies [47, 79, 40, 49] have set a limit on how

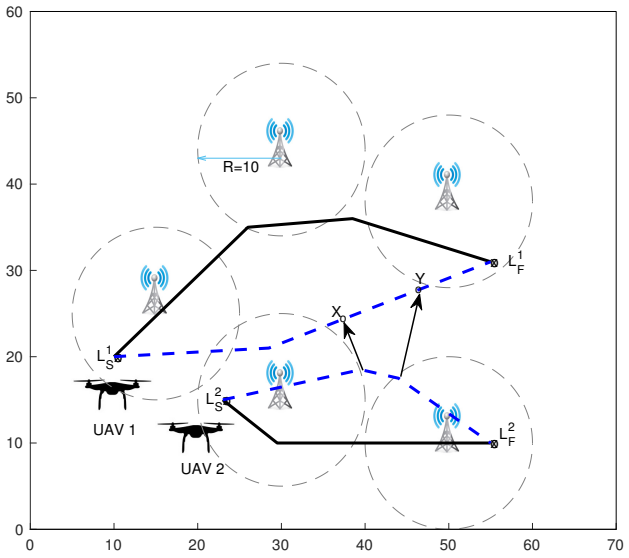


Fig. 44. An example scenario with 5 GBSs and 2 UAVs, where each UAV needs to travel from a starting location (e.g., L_{S}^{1}) to a final destination point (e.g., L_{F}^{1}). The UAV 1 can shorten its path by the help of UAV 2 as shown with dashed lines.

often UAVs can lose their cellular connection (outage) while maintaining the required communication quality. They focus on finding the best paths for UAVs while staying within this limit. However, these studies look at each UAV's path separately and don't consider how UAVs could work together to stay connected.

The GBS ranges are determined based on the shortest distance needed to get the desired communication quality between UAVs and GBSs.

Let's imagine a scenario as shown in Fig. 51, where there are five GBSs and two UAVs. Each UAV has a mission: to fly from a starting point to a final destination without losing their cellular connection for too long. This is important because UAVs need to stay in touch with their operators through control messages. The solid lines represent the best paths for UAVs if they don't help each other, while the dashed lines show what happens when they work together to stay connected. Between points X and Y, the first UAV connects to GBSs through the second UAV, like passing

a message along. Notice that the second UAV adjusts its path a bit to make this happen. Overall, when UAVs collaborate like this, they finish their missions faster than if they work alone, like in the solid lines.

Our main focus is on how UAVs can work together to find good paths. In Section 4.2, we explain how our system works and what the problem is. Since solving this problem perfectly is really hard, in Section 4.3, we present an approximate solution using a graph-based approach with a few steps. We'll share the results of our solution and compare it with a different solution that doesn't consider UAV collaboration in Section 4.4. Finally, in Section 4.5, we wrap up by summarizing our findings and discussing what could be done in the future.

4.2 System Model

4.2.1 Assumptions

We're assuming there are n UAVs, labeled as a group $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$. Each UAV has a mission: to fly from a starting point, $L^u_{\mathrm{S}} = (x^u_{\mathrm{S}},\ y^u_{\mathrm{S}},\ z^u_{\mathrm{S}})$, to a final spot, $L^u_{\mathrm{F}} = (x^u_{\mathrm{F}},\ y^u_{\mathrm{F}},\ z^u_{\mathrm{F}})$, where all UAVs stay at the same height, H. UAVs travel at a specific speed, V, and altitude, H, without having a loss of communication for more than $\tau_{\max}$ time units. We also have a set of GBSs, represented by $\mathcal{G} = \{g_1, g_2, \ldots, g_k\}$, where there are k GBSs. Each GBS's position is given as $(x_i,\ y_i,\ z_i)$, and they're all at the same height, H_{G}.

Just like in previous studies [47, 79, 40], we're keeping things simple by assuming that each communication link (like from GBS to UAV or between UAVs) has its own separate communication channel, so they don't interfere with each other. Both GBSs and UAVs have one antenna each, which sends and receives signals in all directions. Communication happens mainly in LoS. We're using R_G to define the minimum range

Notations	Description
$\mathcal{U}$, $\mathcal{G}$	The set of UAVs and GBSs, respectively.
n, k	Number of UAVs and GBSs, respectively.
L_{S}^{u}, L_{F}^{u}	Start and final location of UAV u, respectively.
$x_u(t)$, $y_u(t)$	Location of UAV u in timeslot t.
$c_u(t)$	Connectivity of UAV u at time t. It is equal to 1 if UAV u can communicate to a GBS directly or over another UAV at timeslot t; otherwise it is 0.
R_G	Max distance/range for a GBS-UAV link to maintain required SNR level.
R_U	Max distance/range for a UAV-UAV link to maintain required SNR level.
T_u	Flight duration time of UAV u
V_u	Maximum speed of UAV u
T_{Total}	Sum of flight durations of all UAVs
τ_{max}	Maximum continuous outage threshold for UAVs

Table 6. Notations used in Chapter 4

needed for the SNR for UAV-GBS communication. It is calculated using: $R = \sqrt{\frac{\gamma_0}{S_{\min}} - (H - H_{\mathrm{G}})^2}$, where $\gamma_0 = \frac{P\beta_0}{\sigma^2}$ is a reference SNR. Here, P is how strong the GBSs send signals, σ^2 is the noise power at a UAV's receiver, and β_0 is how strong the signal is at a reference distance of 1 meter. S_{min} is the smallest SNR needed for good communication between UAVs and GBSs. This SNR at a UAV's receiver tells us how well the cellular-enabled communication is working. For communication between UAVs, we use a similar concept called R_U. All the symbols used in this chapter are listed in Table 6.

4.2.2 Problem Statement

Our scenario, the main goal is to ensure that the UAVs finish their task without experiencing a communication interruption (meaning no direct or multi-hop connection through other UAVs) for more than $\tau_{\max}$ time units. Additionally, the aim is to make the total time taken for all UAVs to complete their missions as short as possible. This total time is determined by the last UAV that arrives at its final destination. Furthermore, there is a goal to reduce the total distance that all UAVs need to travel.

Since the UAVs will be moving, we use $u(t) = (x_u(t),\, y_u(t),\, H)$ to show where UAV u is located at a given time t. Here, $0 \leq t \leq T^u_{\max}$, with $T^u_{\max}$ being the maximum time UAV u can fly at a constant speed of V_u. Based on these concepts, we can describe the optimization problem as follows:

$$\min \quad (\max_{\forall u \in U} T_u)\lambda + T_{Total}, \tag{4.1}$$

$$\text{s.t.} \quad (x_u(0), y_u(0)) = (x_{\mathrm{S}}^u, y_{\mathrm{S}}^u), \forall u \in \mathcal{U}, \tag{4.2}$$

$$(x_u(T_u), y_u(T_u)) = (x_{\mathrm{F}}^u, y_{\mathrm{F}}^u), \forall u \in \mathcal{U}, \tag{4.3}$$

$$\mathrm{dist}_{u(t)}^{u(t+1)} \leq V_u, \forall u \in \mathcal{U}, \forall t \in T, \tag{4.4}$$

$$c_u(t) \;=\; \begin{cases} 1, & \text{if } \exists g_k \in \mathcal{G} \text{ s.t. } \mathrm{dist}_{u(t)}^{g_k} \leq R_G \\ 1, & \text{if } \exists v \in \mathcal{U} \text{ s.t. } \mathrm{dist}_{u(t)}^{v(t)} \leq R_U \;\& \\ & \quad c_v(t) \;=\; 1 \\ 0, & \text{otherwise,} \end{cases}$$

$$\forall u \in \mathcal{U}, \forall t \in T, \tag{4.5}$$

$$\sum_{l=t}^{t+\tau_{max}} c_u(l) \geq 1, \forall u \in \mathcal{U}, t \in T, \tag{4.6}$$

$$T_u = \sum_{t=0}^{T} (\mathrm{dist}_{u(t)}^{u(t+1)} / V_u), \forall u \in \mathcal{U}, \tag{4.7}$$

$$T_{Total} = \sum_{u=0}^{U} T_u, \forall u \in \mathcal{U}, \tag{4.8}$$

where,

$$\mathrm{dist}_u^v = \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2 + (z_u - z_v)^2}\ .$$

In Equation (4.1), we're using a method called scalarization, which involves making the first goal more important by multiplying it with a big number λ. The idea is to first minimize the total time it takes for all UAVs to complete their missions, and then minimize the total travel time for all UAVs within that mission time.

Equations (4.2) and (4.3) determine the starting and ending points for each UAV's journey, respectively. Equation (4.4) makes sure that UAVs don't move more than their set speed between each time step. Equation (4.5) sets the connectivity to

1 if a UAV is within range of a GBS or another connected UAV. Here, $c_u(t)$ shows if UAV u is connected at time t.

Equation (4.6) ensures that no communication outage lasts more than $\tau_{\max}$ time units. Equation (4.7) calculates how long UAV u is in the air, and Equation (4.8) calculates the total flight time for all UAVs combined.

4.3 Proposed Solution

Although the problem could be solved using a nonlinear optimization solver with the model from the previous section, this takes a while to get to the answer. So, in this part, we suggest an easier solution using a graph-based method. First, we check if the UAVs can actually get to their destinations considering all the rules and things like where the GBSs are, and also thinking about how the UAVs can help each other. If it is doable, we then use the graph method to come up with approximate paths for all the UAVs.

4.3.1 Graph-Based Feasibility Check

For every UAV, we make a graph $G_u = (V, E)$, where nodes show GBS positions and the UAV's starting and ending points. If the UAV can fly directly between any of these spots without going over the communication time limit, we add an edge connecting the nodes. The edges' weights are the straight-line distances between the nodes.

More formally,

$$V = G \cup \{L_S^u, L_F^u\}, \tag{4.9}$$

$$E = \{e_{i,j} \mid \forall i \in V, \forall j \neq i \in V \text{ s.t. } \text{dist}_i^j \leq 2R_G + \tau_{max} \;\&\; w_{ij} = \text{dist}_i^j\}. \tag{4.10}$$

Algorithm 3: Feasibility Check

$\mathbf{R} = |\mathcal{U}| \times |\mathcal{G}|$: Reachability matrix
for *each* $u \in \mathcal{U}$ **do**
    Form graph G_u as described in (4.10).
    Run BFS from source u
    for *each* $g \in \mathcal{G}$ *that is reachable from* u **do**
        $\mathbf{R}_{u,g} = 1$
    end
end
for *each* $g \in \mathcal{G}$ **do**
    Calculate A_g using (4.12) to find the total number of UAVs that can reach g
end
continue=true
while *continue* **do**
    for *each* $u \in \mathcal{U}$ **do**
        Update the graph G_u using (4.13).
        Run BFS from source L_{S}^u and update $\mathbf{R}$
    end
    if $\sum_{\forall g \in \mathcal{G}} A_g$ *did not increase* **then**
        continue=false
    end
end
for *each* $u \in \mathcal{U}$ **do**
    if L_{F}^u *is not reachable from* L_{S}^u **then**
        return false
    end
end
return true

With the graph we made and by using the BFS algorithm starting from each UAV's starting point, we can figure out which GBSs each UAV can reach on its own without going over the communication time threshold.

Let $\mathbf{R}$ denote a $|\mathcal{U}| \times |\mathcal{G}|$ matrix showing the reachability of UAVs to the GBS areas. Thus, we set

$$\mathbf{R}_{u,g} = \begin{cases} 1, & \text{if UAV u can fly to GBS range g with a} \\ & \text{desired connection (i.e., no outage} > \tau_{max}\ ) \\ 0, & \text{otherwise.} \end{cases} \tag{4.11}$$

We then find the number of UAVs that can reach to each GBS g as follows:

$$A_g = \sum_{\forall u \in \mathcal{U}} \mathbf{R}_{u,g}. \tag{4.12}$$

After we know A_g for each g, we run the BFS algorithm again on each graph G_u. We add new connections between nodes if this condition is met:

$$\text{dist}_{g_i}^{g_j} \leq 2R_G + (A_i + A_j - 1)R_U + \tau_{max}. \tag{4.13}$$

This equation looks at the best way to position the UAVs (like being in a straight line between a pair of GBSs that are R_U apart) to make sure a UAV can travel from one GBS area to the other one.

Remember, we have to run the BFS again and again until none of the A_g values change. This is because when we add new connections, some A_g values might change. This can create more chances to add new connections. Algorithm 3 has the step-by-step process of this feasibility check. If no more new connections can be added, the algorithm checks if each UAV can make it to its final destination. If any of them can not, it says it is not possible.

4.3.2 Graph-Based Path Approximation

If the solution is feasible, to figure out the paths for all the UAVs, we start by putting new points and connections into the graph. Then, we use Dijkstra's method to find the shortest paths (taking into account the values denoted by w_{ij} in Equation 4.10) from where each UAV starts to where it needs to end up. Now, let's delve into the specifics of these steps.

4.3.2.1 Finding Helping Points

While checking the feasibility, if one UAV needs assistance from another UAV to move between two GBSs without losing connectivity, we identify these points of assistance or relays. Let's use h_i^j to represent a helping point where UAV u_i provides coverage for UAV u_j to travel between two GBS areas with a desired link. Look at

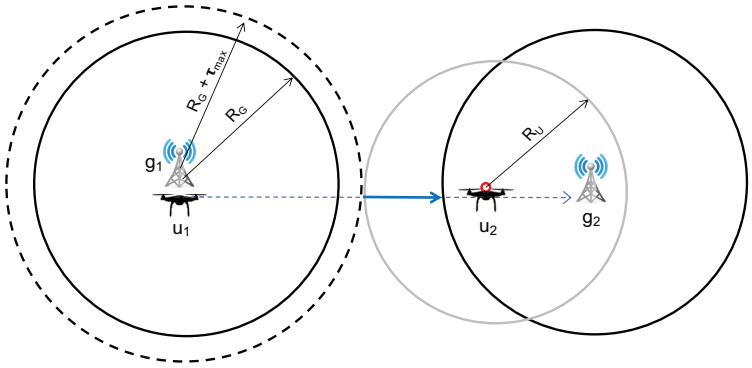


Fig. 45. The exact location of helping point in which u_2 helps u_1 to go to g_2's range from g_1's center.

Fig. 45 as an example. Here, only one UAV can reach these GBS ranges on its own ($A_g = 1$). Let's say that according to the shortest path found by Dijkstra's algorithm, UAV u_1 needs to move from the range of g_1 to the range of g_2 (u_1 relies on u_2, or u_2 needs to assist u_1). In this scenario, we determine that u_2 should help u_1 at a location that's a distance of R_u away from the point when u_1 reaches its communication outage limit (marked by the dashed circle's edge). In Fig. 45, this corresponds to the red point, which is located at a distance of $d = R_G + R_U + \tau_{max}$ from g_1.

To determine the exact position of the helping point, we start by finding a normal vector, denoted as $\hat{\mathbf{n}}(g_1, g_2)$, pointing from the center of g_1 to the center of g_2. We then multiply this vector by the distance of the assistance point from g_1 to get the coordinates. In mathematical terms, we have:

$$\hat{\mathbf{n}}(u_1, u_2) = \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = \begin{bmatrix} \frac{x_{u_2} - x_{u_1}}{\text{dist}_{u_1}^{u_2}} \\ \frac{y_{u_2} - y_{u_1}}{\text{dist}_{u_1}^{u_2}} \end{bmatrix}. \tag{4.14}$$

Using this, we find the specific point where u_1 should be to assist u_2:

$$h_i^j = (x_{u_1} + n_1 d, y_{u_1} + n_2 d),$$

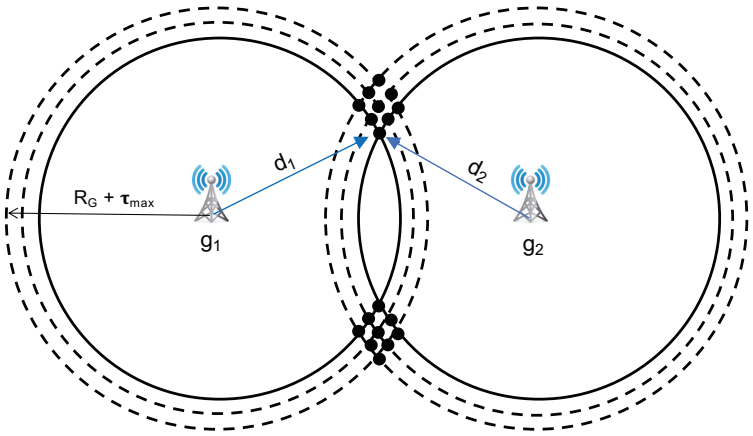


Fig. 46. Intersection of two GBS areas. Dark points are added to graph as new vertices.

where $d = R_G + R_U + \tau_{max}$. At this point, u_1 must wait for a maximum of $\text{dist}_{g_1}^{g_2} - 2R_G - \tau_{max}$ units of time so that u_1 can safely reach the GBS range of g_2.

It might be the case that there are more UAVs that can assist u_1 than actually required. When this happens, we select the UAVs with shorter initial paths calculated using Dijsktra's method. Presently, we're not accounting for situations where there is a circular dependence between UAVs.

4.3.2.2 Finding Intersection Points

To make the UAVs' paths even more efficient, we take a step further. We identify where the coverage areas of the GBSs intersect. These intersections occur where the boundaries of the service areas overlap. We pinpoint these intersection points, which are essentially where the service coverage circles from the GBSs overlap, and we introduce these as new vertices in our graph.

Importantly, these intersections will always be inside the GBS ranges, meaning that as UAVs travel to these from either GBS center, there won't be any loss of connection.

We also focus on the region where the service is transitioning to an outage (rep-

resented by the space between dashed and solid circles) and the outage circles themselves. In this case, we are cautious to ensure that UAV travel remains within acceptable limits. We do this by breaking down the outage area into smaller circular sections, essentially adding more circles between the solid and dashed ones. After creating these sections, we find the intersection points for each circle with one another. For each of these points, we make sure that the combined distance from that point to the considered GBS centers is not greater than a certain value ($2R_G + \tau_{max}$).

For instance, in Fig. 46, we introduce an additional circle between the outage and service area circles. This leads to the identification of 9 intersection points in both the upper and lower sections. We then verify if the sum of distances (d_1 and d_2) from each GBS center is within the limit of $2R_G + \tau_{max}$ before adding these points to the graph.

It is important to understand that these intersection points are just more where a UAV could travel along its route. So, if a UAV is receiving assistance from other UAVs to calculate its shortest path, we can also take into account the coverage area of the assisting UAVs. This area can be thought of as another circle that will overlap with other circles we are considering, like the circles representing the GBS service areas or the additional circles in the outage region.

For instance, in Fig. 45, when we think about this concept, an intersection between the gray circle and the dashed circle appears. In this particular example, there is only one point where they intersect. But in different scenarios, there could be two intersection points resulting from this concept. This shows how we are expanding our consideration to include the impact of helping UAVs' coverage areas. These points are added to G_{u_1} or to the graph of UAV u_1.

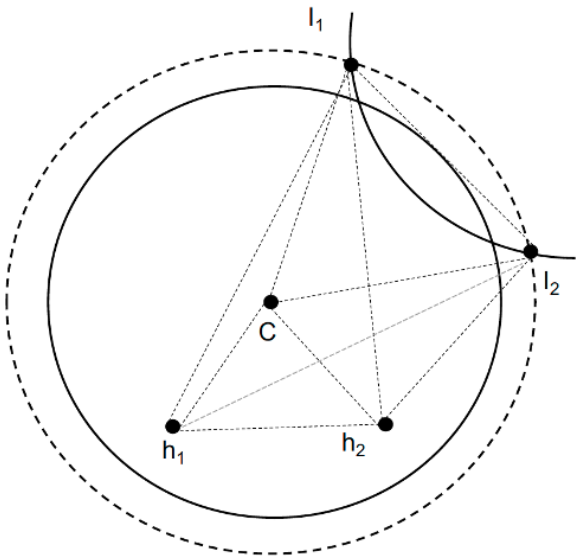


Fig. 47. The vertices and edges in the graph: I_1 and I_2 are intersections of circles, h_1 and h_2 are helping points of a UAV to another UAV, and C is the center of GBS's region. Edges are created between all vertices.

4.3.2.3 Adding Edges Between Nodes on the Graph

As we include new points in the graphs (like G_u for UAV u), we also need to create new connections between these points. To do this, we start by identifying the points that originate from the GBS's locations. These points include the locations of the GBSs as well as the helpful points from the UAVs.

Once we identified these points, for each of them, we determine all the other points that fall within their coverage area. For every pair of points within this area, we establish a connection, or edge, between them. This helps define potential routes or paths between these points.

Fig. 47 provides an example of this process. Here, you can see that edges are drawn between all pairs of nodes within the same GBS service area. This demonstrates how we're building connections in the graph based on the range of a single GBS.

4.3.2.4 Adding Short-cut Edges Between Nodes on the Short-path

To make UAV paths even more efficient, we also consider adding short-cut connections between nodes in the travel path graphs. This is because some routes may not be included based on the usual methods we talked about earlier, possibly because the distance of those routes exceeds certain conditions we have set. So, we are considering creating short-cuts.

Here is how we approach this: We start by drawing a direct connection (an edge) between every pair of nodes that make up the current shortest path for a UAV. Then, we run a check to ensure we are not exceeding the maximum allowable outage. To do this check, we identify where these new short-cut edges intersect with the GBS circles, and we accurately determine their coordinates.

Next, we sort these intersection points based on how far they are from one end of the short-cut edge to the other. We then go through each consecutive pair of these intersection points. If these points fall within the ranges of different GBSs and the distance between them is less than or equal to τ_{max}, we keep the short-cut edge. But if the distance between these consecutive intersection points is greater than τ_{max} and they are in different GBS ranges, we skip adding this short-cut edge.

The procedure detailed above can be found in Algorithm 4. This method systematically determines the inclusion of short-cut connections, with the objective of further optimizing the trajectories of UAVs.

Let's look at an example scenario in Fig. 48 to explain this process. In this scenario, we have a short-cut edge that intersects with the circles of the GBSs. We've identified six points where this edge intersects with the GBS circles, and we call this set of points $M = \{M_1, M_2, M_3, M_4, M_5, M_6\}$.

What we do next is examine pairs of consecutive intersections that come from

Algorithm 4: Adding Short-Cut Edges

```
for each (N1, N2) pair on the current path of UAV do
    Add a temporary edge E' between N1 and N2
    M ← ordered set of intersection points between E' and GBSs.
    for each consecutive points (Mi,Mi+1) in M do
        if dist^Mi_Mi+1 ≥ τmax × V  &  Mi and Mi+1 are in range of different GBS
          service areas then
            Remove E' as it is infeasible
        end
    end
end
```

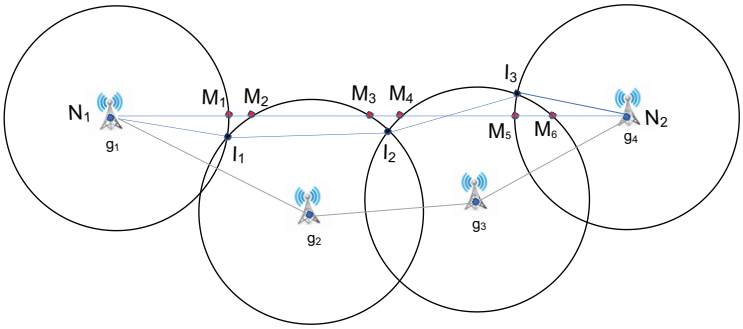


Fig. 48. Checking the feasibility of adding short-cut edge between two nodes, N_1 and N_2, on the current path of a UAV.

different GBS service areas. In this case, we focus on the distance between M_1M_2 and the distance between M_3M_4. If the drone can travel this distance without going beyond the maximum allowable outage, then we consider the short-cut edge as a valid option.

When we apply this check and find that the distance between M_1M_2 and M_3M_4 is within the acceptable outage limit, we determine that the short-cut edge is a good choice. By factoring this into our path optimization process, and running Dijkstra's algorithm again, we can find a shorter path for the drone compared to its current path, which follows the sequence $< g_1, I_1, I_2, I_3, g_4 >$.

4.3.2.5 Path Calculation and Time Synchronization

After we completed the formation of graphs for each UAV, we use Dijkstra's algorithm to discover the shortest paths for them. It is important to note that if a UAV is assisting another UAV, it must be at a specific helping point when the other UAV requires its help. As a result, for these helping UAVs, we determine their routes to the final destination points, taking into account the need to visit and wait at these helping points.

Once we have the paths for each UAV figured out, the next step is to coordinate their timings so they can travel simultaneously. This coordination might involve adding some extra waiting time to the path of a helping UAV. This waiting time becomes necessary if the helping UAV arrives at the assistance point earlier than needed and must wait there until the dependent UAV no longer requires its help.

4.4 Evaluation

In this part, we show the results of simulations that test how well our proposed approximate solution works. We used a map with 12 GBSs and 3 UAVs. The GBSs were positioned at a height of 12.5 meters, while the UAVs were at 90 meters above the ground. The reference SNR at a distance of 1 meter was set at 80 dB, and the minimum required SNR (S_{min}) was set to 26.02 dB. The maximum speed of the UAVs was limited to 50 m/s. With these parameter values, we calculated that the distances R_G and R_U are both equal to 10 units in the x-y axis, which corresponds to a physical distance of 250 meters.

In Fig. 49, we initially compare the best route taken by UAVs using the CPLEX method with the nonlinear model mentioned in Section 4.2, alongside the paths acquired through the approximate solution introduced in Section 4.3. The outcomes

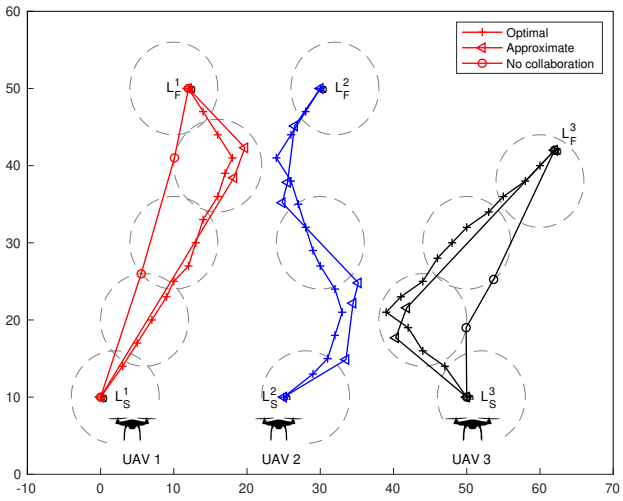


Fig. 49. Optimal and approximate UAV trajectories with collaboration of UAVs. When there is no collaboration UAV 2 cannot fly with given outage threshold ($\tau_{max} = 2.5$ units).

indicate that the suggested approximate solution manages to closely mirror the optimal route. It is important to note that without cooperation, UAV 2 cannot fly from its starting point to the destination within a time limit of $\tau_{max} = 2.5$ units. This implies that earlier studies [47, 79, 40, 49], which did not consider collaboration among UAVs, would have failed to find a path for UAV 2. However, with the assistance of UAVs 1 and 3, such a path becomes feasible. The various scenarios' path lengths for each UAV are detailed in Table 7.

In Fig. 50, we display the flight paths of UAVs achieved using the approximate solution with various outage thresholds. As τ_{max} increases, the lengths of individual UAV paths and the overall mission (represented by the longest UAV path) durations decrease, as expected. These outcomes underscore the reliability of the proposed approximate solution, which remains effective across different scenarios.

When we compare the computation times of the suggested approximate solution

Table 7. Path lengths of UAVs (shown in units of axis)

Method	**UAV 1**	**UAV 2**	**UAV 3**	**Total**	**Mission**
CPLEX	47.02	47.06	47.20	141.28	47.20
Heuristic	48.80	50.69	45.27	144.76	50.69
No collaboration	41.80	N/A	35.39	N/A	N/A
τ_{max}	**UAV 1**	**UAV 2**	**UAV 3**	**Total**	**Mission**
1	50.91	49.56	48.37	148.84	50.91
2.5	48.80	50.69	45.27	144.76	50.69
5	46.37	49.81	42.01	138.19	49.81

and the CPLEX-based optimal solution – for instance, in the case of the results shown in Fig. 49 – the approximate solution takes only 0.1 seconds, whereas the optimal solution using CPLEX takes around 30 minutes. Thus, the approximate solution delivers much quicker results while maintaining a closeness to the optimal outcomes.

4.5 Summary of Contributions

In this chapter, we explore how UAVs that can communicate with GBSs can optimize their flight paths while staying within a specified connectivity limit. We look at a situation where each UAV needs to travel from a starting point to a final point, and all the UAVs collaborate to make sure they stay connected to the GBSs.

We start by setting up a mathematical problem ILP that helps us find the best paths for the UAVs. To make things more manageable in terms of calculations, we create a simplified solution using graphs. This graph-based approach makes things easier to compute. We run simulations, and from the results, we see that our simpler solution is almost as good as the optimal one. This approach works well for various scenarios.

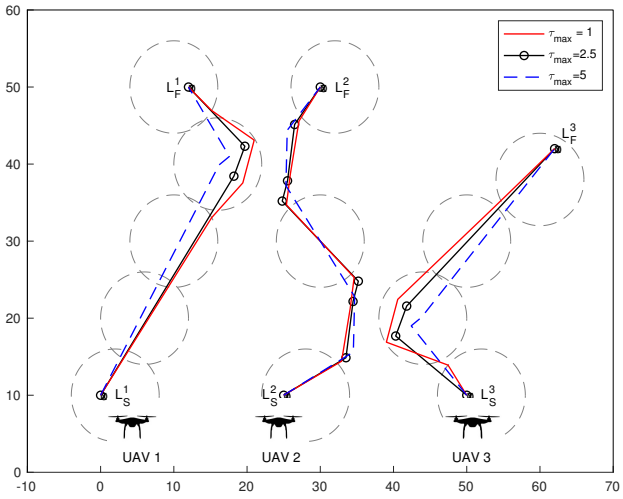


Fig. 50. Comparison of approximate trajectories with different connection outage thresholds.

4.6 Future Work

While this chapter provided a comprehensive discussion on the optimization of UAV path trajectories with collaboration while they have direct or indirect connections to GBSs, there are several aspects that necessitate further investigation.

1. **Improving Graph-Based Solution Precision**: The current graph-based solution, although efficient in time, occasionally deviates from the optimal solution. Enhancing the accuracy of this method is paramount.

2. **Scalability with Increased UAV Numbers**: As UAV operations scale, the number of UAVs in a given scenario may also increase. Our graph-based solution could be generalized to handle scenarios with a larger number of UAVs, ensuring its relevance and efficiency in a variety of operational contexts.

3. **Performance Comparison**: As we introduced both the ILP method and the graph-based solution, a detailed comparative analysis on their performance

metrics, including accuracy, computation time, and resource utilization, would be beneficial. This will provide more nuanced insights into the trade-offs and advantages of each approach.

CHAPTER 5

AOI-OPTIMAL CELLULAR-CONNECTED UAV TRAJECTORY PLANNING FOR DATA COLLECTION FROM IOT NETWORKS

5.1 Introduction

UAVs have revolutionized various fields such as wireless communications, agriculture, and search and rescue operations, thanks to their agility and ability to reach inaccessible areas. This study focuses on a scenario where UAVs serve a crucial role in data collection from ground sensors or IoT networks, acting as a bridge to relay this information to its intended destination. For effective data transfer, a UAV must navigate close to each IoT device along a designated path before concluding its mission.

Given the limited battery life of UAVs, which restricts their flight duration, it is essential to meticulously plan their routes to ensure efficient data collection. This planning must also take into account the specific times when data becomes available at each IoT device, necessitating visits to these devices only after data generation. Contrary to many existing studies that assume data availability before mission commencement [7, 8, 9], this work acknowledges the practical scenario where data generation may not always align with such assumptions.

Once IoT devices generate data, the UAVs need to collect that data and then deliver it to their respective destinations, taking into consideration the requirements of the application at hand. In this study, we consider a broader scenario by including multiple UAVs and a set of GBSs as the delivery points of the collected data from the ground IoT devices. This approach redefines the AoI to cover the period from

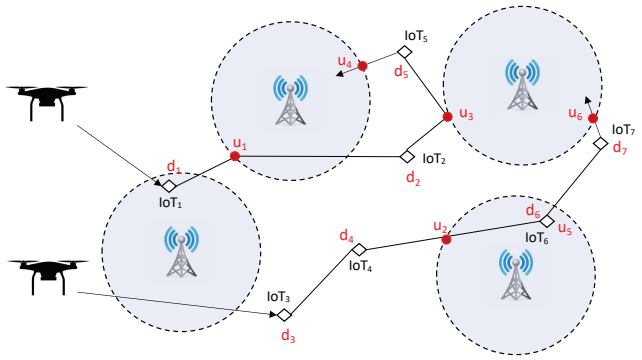


Fig. 51. An example scenario where two UAVs collect data from seven ground IoT devices considering their data generation times and uploads the collected data by visiting a base station. Age of Information is defined from the moment the data is generated at each IoT device until it is uploaded to a GBS.

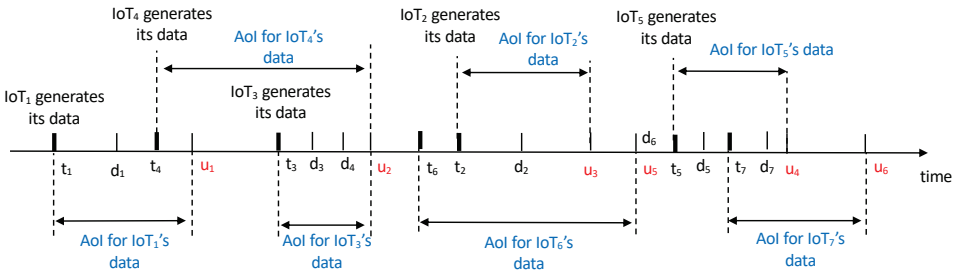


Fig. 52. AoI calculation for the data of each IoT device in Fig.51.

data creation to its upload to a GBS, ensuring the timeliness and relevance of the information relayed.

Recent studies have delved into UAV path planning within IoT networks with varying aims, such as minimizing energy consumption [48], reducing connection outage times [80], and maximizing data collection efficiency [81]. The introduction of AoI as a metric emphasizes the importance of data freshness upon delivery. While previous research has largely focused on AoI in scenarios where data is delivered to a single endpoint, this study proposes a more practical and broader scenario where a cellular-connected UAV can upload the collected data to any available GBS, thus

offering a more flexible and realistic approach to data delivery and finalizing mission requirements.

An example scenario is illustrated in Fig. 51 where two UAVs are collecting data from seven ground IoT devices and uploading the collected data to a GBS. The age of information for each collected data is computed in Fig. 52. Note that the data of each IoT i is collected by a UAV at time d_i after its generation at time $t_i \leq d_i$, and it is uploaded to a GBS at a later time (denoted by u). It is possible multiple IoT data can be uploaded at the same to reduce the mission time without increasing maximum AoI.

In this study, our main goal is to optimize UAV flight paths to minimize the maximum AoI of collected data, considering the generation times and locations of IoT devices. Moreover, we also target reducing mission time and the UAV flight durations/distances, advancing the state-of-the-art in UAV-assisted data collection. Our solution approaches include both an ILP based solution and also a more computationally efficient but approximate greedy heuristic based solution.

The rest of the chapter is organized as follows. In Section 5.2, we provide the system model together with the assumptions made and the problem statement. In Section 5.3, we elaborate on the ILP based approach as well as our greedy heuristic based solution and a brute-force approach. In Section 5.4, we then provide our simulation results in various scenarios. Finally, we provide the concluding remarks and discuss the future work in Section 5.5.

Notations	**Description**
$\mathcal{U}$	Set of UAVs
u	The UAV that travels over the field for data collection from a starting point to an end point.
$\mathcal{I}$	The set of ground sensors or IoT devices.
$\mathcal{G}$	The set of ground base stations (GBS).
V_i^D	The time UAV visits the IoT device i and downloads the generated data.
V_i^U	The time UAV uploads and delivers the data captured from IoT i to one of the GBSs.
L_{S}^u, L_{F}^u	Start and final location of UAV u, respectively.
L, T	The ordered set of critical locations and times on the UAV path, respectively.
$L_u(t)$	Location of the UAV u at time $t \in T$.
l_i	Location of ground IoT device i.
g_i	Location of GBS i.

Table 8. Notations and their descriptions defined in Chapter 5(Part 1).

Notations	Description
$c_i^u(t)$	Connection status of the UAV u to IoT device i at time $t \in T$. It is equal to 1 if the UAV u can communicate to the IoT i and receive the data at time t; otherwise, it is 0.
t_i	The generation time of the data at IoT device i.
$d_i^u(t)$	Collection status of data from IoT device i by UAV u at time $t \in T$. It is equal to 1 if the UAV u collects IoT device i's data at time t; otherwise 0.
$p_i(t)$	Upload status of data that is downloaded from IoT i to a GBS at time $t \in T$. It is equal to 1 if the UAV uploads the data downloaded from IoT i to one of the GBSs at time t; otherwise 0.
$g_i^u(t)$	Connection status of the UAV u to GBS i at time $t \in T$. It is equal to 1 if UAV can communicate to the GBS i and send the data at time t; otherwise, it is 0.
$G(t)$	If the UAV is in range of at least one GBS at time $t \in T$.
R_I	Max distance/range for an IoT-UAV link to maintain required SNR level.
R_G	Max distance/range for a UAV-GBS link to maintain required SNR level.
T_{max}	Maximum possible flight duration for the UAV to reach the destination.
T_F	The first time the UAV arrives at the final location (i.e., mission time).

Table 9. Notations and their descriptions defined in Chapter 5(Part 2).

5.2 System Model

5.2.1 Assumptions

We assume a system model with a set of UAVs, $\mathcal{U}$, a set of ground IoT devices, $\mathcal{I}$, and a set of GBSs, $\mathcal{G}$. Each IoT device is assumed to generate a data at some specific times defined by the application. The location of each IoT device i is represented by l_i and its data generation time is denoted by t_i. Each UAV u has a starting location, L_S^u and a final destination L_F^u which needs to be reached out after collecting data from ground IoT devices assigned to it. The data collection for all UAVs needs to be completed within a given time constraint $T_{\max}$, which is defined as the maximum possible flight time for the UAVs and can be computed based on their hardware specifications.

We assume each GBS, UAV and IoT device is equipped with a single omni-directional antenna and each (UAV, IoT) link or (UAV, GBS) link works in a separate band that is orthogonal to others to avoid the interference (as our focus is not to manage the interference). The collection of data from an IoT device happens when a UAV arrives in the vicinity of the IoT device. More specifically, we assume that when the distance between the UAV and an IoT device is less than R_I, the data can be transmitted. Similarly, the upload of the data from the UAV is assumed to happen to a nearby GBS when the UAV arrives in the range of a GBS, which is assumed to be R_G. It is assumed that UAV can fly with a maximum speed of V at a fixed altitude of H. Note that this will allow the UAV to communicate with the ground IoT devices through LoS based signal without having interference. The actual value of R_I and R_G can be computed by considering the signal level modeling (i.e., minimum SNR necessary) and the required transmission bandwidth for the specific application

data [80, 82]. That is, for example,

$$R_G = \sqrt{\frac{\gamma_0}{S_{\min}} - (H - H_G)^2}.$$

where $\gamma = P\beta_0$ represents the baseline SNR, P the transmission power of a GBS, σ^2 the noise power at the UAV receiver, and β_0 the channel power gain at a reference distance of 1m. The minimum SNR, S_{min}, ensures the communication quality between a UAV and a GBS.

The location of the UAV at time t is denoted by $L_u(t) = (x_u(t),\, y_u(t),\, H)$ until its flight ends at time $T_{\max}$.

The location of each GBS, designated as g_i, is expressed using Cartesian coordinates (x_i, y_i, z_i). To simplify, we assume all GBSs are at a constant altitude, HG, i.e., $z_i = z_j = HG$ for any i, j within the range $[1, K]$. The starting and ending points are denoted as (x_S, y_S, z_S) and (x_F, y_F, z_F), respectively, with both z_S and z_F equal to H. The UAV's location at any time t is indicated by $(x(t), y(t), H)$, where $0 \leq t \leq T_{max}$, and T_{max} is the maximum flight time determined by the UAV's battery capacity and its constant speed V.

5.2.2 Problem Statement

Given the set $\mathcal{U}$, $\mathcal{I}$, and $\mathcal{G}$, together with the locations of IoT devices (l_i), GBS centers (z_i), start and end locations of UAVs $(L_S^u,\, L_F^u)$ and the data generation times of IoT devices (t_i), our goal is to minimize the maximum AoI during the data collection process by UAVs. In addition to this primary objective, we also consider minimizing the mission time as secondary goal, and also aim to minimize the length of the total path travelled by the UAVs as a third objective. Thus, using the scalarization, the

overall objective is formulas as:

$$\min \quad ((A_{max})\lambda + u(T_F))\,\Theta + D_{sum} \tag{5.1}$$

s.t.

$$A_{max} = \max\left\{(V_i^U - t_i)\right\}, \forall i \in \mathcal{I}, \tag{5.2}$$

$$u(T_F) = \max_{\forall u \in \mathcal{U}}(T_F^u), \tag{5.3}$$

$$D_{sum} = \sum_{u \in \mathcal{U}}(L_u^S \rightarrow L_u^F). \tag{5.4}$$

where V_i^U is the upload time of the IoT device i's data to a GBS by the UAV that downloaded its data, T_F^u is the time UAV u reaches its final point and $L_u^S \rightarrow L_u^F$ denotes the trajectory for the UAV u. We use scalars $\lambda \gg \theta$, to prioritize different goals.

5.3 Proposed Solutions

In this section, we first describe the ILP based solution, then discuss the heuristic based more computationally efficient and approximate solution. The notations used throughput this section are given in Table 8 and 9.

5.3.1 ILP Solution

In the proposed problem each UAV starts from a starting point to collect data from ground IoT devices and arrives to an end point (which can be the same location as the initial starting point). Let $L_u = \{L_0^u, L_1^u, L_2^u, ..., L_{2|I|}^u, L_{2|I|+1}^u\}$ be the set of ordered locations that we are trying to identify on the route of each UAV u. These locations correspond to the critical locations that define the path of each UAV u which include the start (L_S^u) and end locations (L_F^u) as well as the download and upload locations for the data of each IoT device. Note that $L_0^u = L_S^u$ and $L_{2|I|+1}^u = L_F^u$. We

also define $T_u = \{T_0^u = 0, T_1^u, T_2^u, ..., T_{2|I|}^u, T_{2|I|+1}^u = T_F^u\}$ as the set of times that the UAV is present at the corresponding locations in L, i.e., $L_u(T_i) = L_i$.

It is essential to ensure that each UAV commences and concludes its mission at specified locations. Furthermore, due to the limitations imposed by the UAV's battery capacity, there is a maximum allowable flight duration that must be adhered to. Thus, we have

$$L_u(0) = L_S^u, \tag{5.5}$$

$$L_u(T_F) = L_F^u \ \& \ T_F^u \leq T_{max}, \forall u \in \mathcal{U}. \tag{5.6}$$

For each consecutive time moment during the mission of UAVs, we ensure that the distance traveled by the UAV is equal to or less than their maximum speed capability by:

$$\begin{aligned} \text{dist}_{L_u(i)}^{L_u(i+1)} \leq V \times (T_u(i+1) - T_u(i)), \\ \forall i \in [0, 2|I|], \forall u \in \mathcal{U}. \end{aligned} \tag{5.7}$$

Each IoT device generates data at a specific time. The UAV is required to visit the IoT device after this data generation to capture the data. Furthermore, the delivery of this data to a GBS must occur only after it has been captured from the IoT device. These are ensured by

$$V_i^D \geq t_i, \forall i \in \mathcal{I}, \tag{5.8}$$

$$V_i^U \geq V_i^D, \forall i \in \mathcal{I}. \tag{5.9}$$

To facilitate the UAV's data capture from an IoT device, it is imperative for the UAV to be within the communication range of the IoT device. Consequently, a variable has been established to monitor the connectivity status between the UAV

and the IoT device. Data download by the UAV is permissible only if a connectivity link is established. Nevertheless, there may be instances where the UAV is within the communication range of the IoT device but chooses not to download any data.

$$c_i^u(t) = \begin{cases} 1, & \text{if } \text{dist}_{l_i}^{L_u(t)} \leq R_I \\ 0, & \text{otherwise.} \end{cases}, \tag{5.10}$$

$$d_i^u(t) \leq c_i^u(t), \tag{5.11}$$

$$\forall t \in T, \forall i \in \mathcal{I}, \forall u \in \mathcal{U}.$$

Each IoT device's data should be captured by one and only one UAV.

$$\sum_{\forall t \in T} \sum_{\forall u \in \mathcal{U}} d_i^u(t) = 1, \forall i \in \mathcal{I}, \forall u \in \mathcal{U}. \tag{5.12}$$

To record the time at which a UAV visits the IoT device and captures its data, we utilize the variable V_i^D, as previously defined. Since the value of $d_i^u(t)$ is set to 1 only for one UAV throughout the entire timeline of all UAVs, by multiplying this value with the time variable, we obtain the exact moment when data is downloaded from the IoT device onto the UAV.

$$V_i^D = \sum_{\forall u \in \mathcal{U}} \sum_{\forall t \in T_u} (d_i^u(t) \times t), \forall i \in \mathcal{I}. \tag{5.13}$$

Similarly, by multiplying the value of d_i^u by u, we obtain the id of the UAV that downloads the data from IoT device i.

$$\mathrm{U}_D(i) = \sum_{u \in \mathcal{U}} \sum_{t \in T_u} (d_i^u \times u), \forall i \in \mathcal{I}. \tag{5.14}$$

Similar to the UAV-IoT connection, for the UAV to upload the collected data

to the GBS, it must be within the communication range of a GBS. We monitor the UAV-GBS connection status using another variable, $G(t)$, which indicates whether the UAV is connected to at least one GBS at time slot t. Furthermore, it should be mentioned that when the UAV enters the communication range of a GBS, it has the capability to upload data to that GBS.

$$g_i^u(t) = \begin{cases} 1, & \text{if } \text{dist}_{z_i}^{L_u(t)} \leq R_G \\ 0, & \text{otherwise.} \end{cases}, \quad \forall u \in \mathcal{U}, \forall t \in T, \forall i \in \mathcal{G}, \tag{5.15}$$

$$G_u(t) = min(1, \sum_{\forall i \in \mathcal{G}} g_i^u(t)), \forall u \in \mathcal{U}, \forall t \in T, \tag{5.16}$$

$$p_i^u(t) \leq G^u(t), \forall u \in \mathcal{U}, \forall i \in \mathcal{I}, \forall t \in T_u. \tag{5.17}$$

Additionally, we impose a constraint to ensure the UAVs upload the captured data. All data captured by the UAVs from the IoT devices must be uploaded to the GBSs.

$$\sum_{\forall u \in \mathcal{U}} \sum_{\forall t \in T_u} p_i^u(t) = 1, \forall i \in \mathcal{I}. \tag{5.18}$$

In order to compute the AoI of each data, we need to record the time at which the UAV delivers the collected data to a GBS. Given that each IoT's data is delivered only once, we leverage this condition by multiplying the value of $u_i(t)$ by t, and then summing this over all critical times.

$$V_i^U = \sum_{\forall u \in \mathcal{U}} \sum_{\forall t \in T_{u'}} (u_i^{u'}(t) \times t), \forall i \in \mathcal{I}. \tag{5.19}$$

Similarly, by multiplying the value of p_i^u by u, we obtain the id of the UAV that

uploads the data of the IoT device i to a GBS.

$$\mathrm{U}_P(i) = \sum_{u' \in \mathcal{U}} \sum_{t \in T_{u'}} (p_i^u \times u), \forall i \in \mathcal{I}. \tag{5.20}$$

This information is needed, as when there are multiple UAVs, we need to make sure that the data is uploaded by the UAV that collected the data from the IoT. To this end, we also consider the following constraint.

$$\mathrm{U}_D(i) = \mathrm{U}_P(i), \forall i \in \mathcal{I}. \tag{5.21}$$

Finally, the path lengths of the UAV trajectories can be computed by

$$D_{sum} = \sum_{u \in \mathcal{U}} \sum_{i=0}^{2|I|} \mathrm{dist}_{L_i^u}^{L_{i+1}^u}. \tag{5.22}$$

where, dist_u^v represents the distance between two coordinates u and v.

Note that these formulations will apply when there is only one UAV too. However, to speed up the running time of the ILP based solution for one UAV scenario, some constraints (e.g., (5.21)) can be removed as they will be always satisfied and will not be necessary.

Algorithm 5: FastestIoTDelivery (L^u_{cur}, T_{cur})

Input : L^u_{cur}: Current location of UAV
T_{cur}: Current time passed since start
Output: I_{best}: Best IoT index
T_{del}: Delivery time to the best IoT
$G_{nearest}$: Nearest GBS to the best IoT

foreach $i \in \mathcal{I}$ **do**
  `// Find intersection point of UAV's trajectory with IoT` i`'s range`
  $I_i \leftarrow \overline{\mathrm{L^u_{cur} l_i}} \cap R_I(i)$
  `// Find time to reach IoT` i`'s range`
  $\Delta_i \leftarrow (\mathrm{dist}(L^u_{cur}, I_i))/V + T_{cur}$
  `// Update download time if arrived before data generation`
  if $\Delta_i \geq t_i$ **then**
    $\Delta_i \leftarrow t_i$
  end
  `// Closest GBS for IoT` i
  $g^* \leftarrow \arg\min_{g \in \mathcal{G}} \mathrm{dist}(\mathrm{I}_i, g)$
  `// Find intersection point with the closest GBS's range`
  $G^*_i \leftarrow \overline{\mathrm{I_i g^*_i}} \cap R_G(g^*))$
  `// Delivery time for data of IoT` i
  $\Delta_g \leftarrow \Delta_i + (\mathrm{dist}(I_i, g^*_i))/V$
  `// Keep the best one`
  if $G_i < T_{del}$ *or* $T_{del} = NULL$ **then**
    $T_{del} \leftarrow \Delta_g$
    $I_{best} \leftarrow i$
    $G_{nearest} \leftarrow g^*$
  end
end

5.3.2 Greedy Heuristic Approach

Due to the high computational complexity of the ILP based solution, getting results at scale is not practical. Thus, in this section, we present our heuristic based solution. We first start with the scenario where there is only one UAV, then discuss how it scales to the multi-UAV scenario.

5.3.2.1 Single UAV

Our greedy heuristic based approach relies on sequentially integrating IoT devices and GBSs into the UAV's trajectory, while also avoiding exhaustive examination of all permutations. Thus, it enables us to approximate the optimal solution with significantly reduced computational time.

We start with a path that includes only the start and the end location of the UAV. We then select an IoT device and try to place it in one of the available spots on the current trajectory. The available spots are considered as the positions between the consecutive elements of the current trajectory. That is, in the initial trajectory, there is only one position possible (i.e., between the starting and end points of the UAV). In order to select the next IoT device to be added onto the trajectory, we use the strategy described in Alg. 5. That is, we find out the IoT device whose data can be delivered (to the nearest GBS) the fastest based on the UAV's current position. Once an IoT device is added onto the UAV trajectory, we then consider inserting the GBS that allow that IoT device quickest delivery of its data. Here, we only consider the positions (between any consecutive element) after the last added IoT location. For example, once the first IoT is added, we only consider the one and only one position after this IoT device for a possible GBS insertion. Moreover, we consider not even inserting this GBS to the path. This is because it is possible that there may be other GBSs already in the current UAV trajectory which can upload the recently added IoT device's data without increasing the current AoI.

The process continues similarly until all the IoT devices are added in the UAV trajectory. The next IoT to be added on the trajectory is determined by finding the IoT device whose data could be delivered the fastest from the final position (and time) of the UAV on the current trajectory. Once the IoT to be added is determined,

we try all possible locations to add it on the trajectory. As the UAV trajectory grows with new the addition of IoT devices and GBSs, the set of possible locations to add the next IoT device expands. For example, for the second IoT device insertion, since there will be the first IoT device and a GBS are already in the trajectory, there are three positions from which we need to find out the one that gives the smallest AoI and insert the second IoT there. Once the second IoT is added on the trajectory, depending on where it is added, there will be one to three different positions for the GBS to be added. Note that we try not only inserting the GBS to any of the positions after the second IoT's position but also try not inserting the GBS at all and proceed with the option that provides the best AoI. In the cases where inserting the GBS to the trajectory provides the same AoI (compared to not adding it), we opt to omit it is insertion as we also aim to minimize the UAV's travel distance.

Upon completing the integration of all IoT devices, we calculate the AoI for the UAV's ultimate trajectory, which must include the defined start and end points, all IoT devices, and at least one GBS. This entire process is illustrated in Algoritm 6.

Algorithm 6: GreedySingleUAVPathFormation

Input: L_S^u, L_F^u, $\mathcal{I}$, $R_{\mathcal{I}}$, $\mathcal{G}$, S
Output: Updated UAV path $\mathcal{P}$

$\mathcal{C} \leftarrow \emptyset$ // Checked IoTs

// Initialize UAV path with start and finish locations
$\mathcal{P} \leftarrow [L_S^u, L_F^u]$
$L_{cur}^u \leftarrow L_S^u$ // Current location of UAV
$T_{cur} \leftarrow 0$ // Current time
while $|\mathcal{C}| \neq |\mathcal{I}|$ **do**
    // Find the IoT whose data can be delivered earliest
    $B_i \leftarrow \text{FastestIoTDelivery}(L_{cur}^u, T_{cur})$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{B_i\}$
    $B_g \leftarrow \arg\min_{g \in \mathcal{G}} \text{dist}(B_i, g)$
    $P_{IoT} \leftarrow |P| - 1$, $P_{GBS} \leftarrow |P|$
    $L_{IoT} \leftarrow -1$, $L_{GBS} \leftarrow -1$
    $\Delta \leftarrow \infty$
    for $i = 1$ **to** P_{IoT} **do**
        for $g = 0$ **to** P_{GBS} **do**
            $P_{temp} \leftarrow P$
            Insert B_i into P_{temp} at position i
            if $g > i$ **then**
                Insert B_g into P_{temp} at position g
            else
                $g \leftarrow -1$; // No GBS insertion if condition not met
            end
            if $AoI(P_{temp}) < \Delta$ **then**
                $\Delta \leftarrow \text{AoI}(P_{temp})$
                $L_{IoT} \leftarrow i$
                $L_{GBS} \leftarrow g$
            end
        end
    end
    $P \leftarrow (P[1 : L_{IoT} - 1], B_i, P[L_{IoT} : \text{end}])$; // Insert B_i at position L_{IoT} in P
    if $L_{GBS} \neq -1$ **then**
        $P \leftarrow (P[1 : L_{GBS} - 1], B_g, P[L_{GBS} : \text{end}])$; // Insert B_g at position L_{GBS} in P
    end
    $L_{cur} \leftarrow P[\text{length}(P) - 1]$; // Update current location as the last UAV location before end point in $\mathcal{P}$
    $T_{cur} \leftarrow \text{CurrentTimeAt}(L_{cur})$; // Update current time based as the time UAV arrives L_{cur}
end

5.3.2.2 Multiple UAVs

Algorithm 7: GreedyMultiUAVPathFormation

Input: L_S^u, L_F^u, $\mathcal{I}$, $R_{\mathcal{I}}$, $\mathcal{G}$, R_G

Output: Optimized paths for each u in $\mathcal{U}$

1 $\mathcal{C} \leftarrow \emptyset$

2 **foreach** u *in* $\mathcal{U}$ **do**

3 $\quad \mathcal{P}^u \leftarrow [L_S^u, L_F^u]$

4 $\quad \mathcal{T}_{cur}^u \leftarrow 0$

5 **end**

6 **while** $|\mathcal{C}| \neq |\mathcal{I}|$ **do**

7 $\quad$ **foreach** u *in* $\mathcal{U}$ **do**

8 $\quad\quad B_i^u, D^u \leftarrow \text{FastestIoTDelivery}(L_{cur}^u, \mathcal{T}_{cur}^u)$

9 $\quad\quad B_g^u \leftarrow \arg\min_{g \in \mathcal{G}} \text{dist}(B_i^u, g)$

10 $\quad$ **end**

11 $\quad u_{\min} \leftarrow \arg\min_u D^u$

12 $\quad B_i \leftarrow B_i^{u_{\min}}$, $B_g \leftarrow B_g^{u_{\min}}$

13 $\quad \mathcal{C} \leftarrow \mathcal{C} \cup \{B_i\}$

14 $\quad$ Insert B_i and B_g to $\mathcal{P}_{u_{min}}$ as in lines 10-34 in Algorithm 6

15 **end**

16 **foreach** u *in* $\mathcal{U}$ **do**

17 $\quad A_u =$ Calculate AoI for $\mathcal{P}^u$

18 **end**

19 Return $min_{u \in \mathcal{U}} A_u$

In this subsection, we leverage the greedy heuristic developed for a single UAV

and adapt it for multi-UAV scenarios to optimize path planning with the objective of minimizing the maximum AoI. This adaptation involves several modifications to the single UAV approach. Specifically, with every movement of the UAVs, we dynamically update the sequence of IoT devices targeted for data delivery, taking into account the current positions of the UAVs. Consequently, the prioritization of IoT devices for each iteration may vary as the UAVs progress along their paths.

Another key modification pertains to the management of IoT device assignments to UAVs. For each UAV, we maintain a prioritized list of IoT devices, sorted based on the anticipated delivery time. Upon the selection of an IoT device by a UAV for data delivery, and subsequent completion of this delivery (i.e., uploaded the data to a GBS), the IoT device's identifier is removed from the lists maintained for all UAVs. This ensures that no UAV is assigned to an IoT device that has already had its data delivered by another UAV, thus preventing redundant visits and optimizing the data collection process.

In Alg. 7, we describe a greedy selection process for multiple UAVs to optimize their paths while interacting with IoT devices and GBSs, aiming to minimize AoI. Initially, each UAV is assigned a start and end location, and a loop begins that continues until all IoT devices delivered their data. Within this loop, for each UAV, the algorithm determines the best IoT device to visit next based on its current location and the delivery time of data from that IoT device. It also identifies the closest GBS to the selected IoT device (lines 8-13). Among all UAVs, it selects the UAV that can deliver the data in the least amount of time and updates the UAV's path to include the IoT device and GBS at optimal positions to minimize AoI (lines 14-21). This process is repeated, updating the path for each UAV with the inclusion of IoT devices and GBSs, until all IoT devices have been processed. The paths for all UAVs are then recalculated to reflect the updated AoI (lines 22-24). Similar to the approach

for a single UAV, and when considering the addition of a specific IoT, we may omit adding a GBS to the UAV paths if the maximum AoI remains unchanged.

5.3.3 Brute-force Approach

For comparison purposes, we also obtain results using a brute-force approach too. To this end, we obtain all the permutations of the IoT devices to first get a visit order in the UAV trajectories, which always start at the starting location of the UAV and ends in the final location of the UAV. In the single UAV scenario, since all IoT devices should be visited by the same UAV, we consider a permutation that includes all the IoT devices. However, in the multiple UAV scenario, we first distribute the IoT devices to each UAV. It is possible that some UAVs may not be assigned any IoT devices at all. Once a UAV knows the ordered set of IoT devices that will be visited, we then consider adding one GBS to be visited before or after each IoT device visited. It is possible that there may not be a GBS added between some IoT device visits, as this is a possible scenario. Also, note that we do not need to consider more than one GBS between IoT device visits, as we assume in the first GBS visited, all the data will be uploaded. For all possible such permutations of IoT devices and potentially visited GBSs in between, we calculate the AoI for the given scenario and find out the best result. In multiple UAV scenario, we compute the max AoI for all UAVs and take the max of all.

5.4 Simulation Results

5.4.1 Toy Example

In this section, we dive into a hands-on example to demonstrate the step by step functionality of the proposed solutions, and how they compare to other solutions. We consider 5 IoT devices and 4 GBSs deployed on a map of 20 units by 20 units. The

Table 10. Locations of GBSs

GBS ID	**1**	**2**	**3**	**4**
Coordinates	(2,2)	(11,10)	(18,2)	(10,18)

Table 11. Locations of IoT devices and data generation times

IoT ID	**1**	**2**	**3**	**4**	**5**
Coordinates	(2,6)	(18,7)	(15,15)	(5,18)	(3,18)
Data generation times (t_i)	5	10	20	10	10

Table 12. Simulation Parameters

Parameters	**Values**
UAV speed (V)	2
Map size	20x20
GBS range (R_G)	2
IoT range (R_I)	1
Number of IoTs	5
Number of GBSs	4
Scale (for ILP)	1,10,100

positions of GBSs and IoT devices are given in Table 10 and Table 11, respectively.

Single UAV: We start with single UAV example where the UAV commences its mission from the coordinates (4,4), aiming to complete its mission at the location (2,12). The other parameter values are given in Table 12.

We start with brute-force results presented in Fig. 53, wherein all possible permutations of IoTs and GBSs are explored to minimize the AoI. In this scenario, it is observed that the UAV opts not to deliver data from IoT1 immediately. Instead, it proceeds to IoT5, then to IoT4, and delivers the data from all three IoT devices simultaneously at GBS2. The UAV's path in this method is delineated as follows:

$$S \rightarrow \text{IoT1} \rightarrow \text{IoT5} \rightarrow \text{IoT4} \rightarrow \text{GBS2} \rightarrow \text{IoT2} \rightarrow \text{GBS3}$$
$$\rightarrow \text{IoT3} \rightarrow \text{GBS2} \rightarrow E$$

In the greedy heuristic based approach, we only consider a selected subset of all

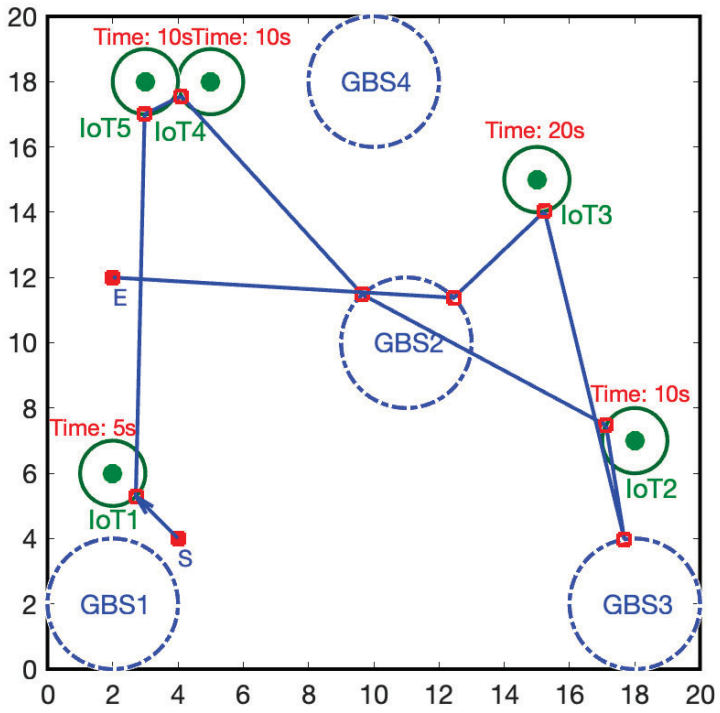


Fig. 53. **Example with a single UAV:** UAV trajectory obtained by brute-force approach.

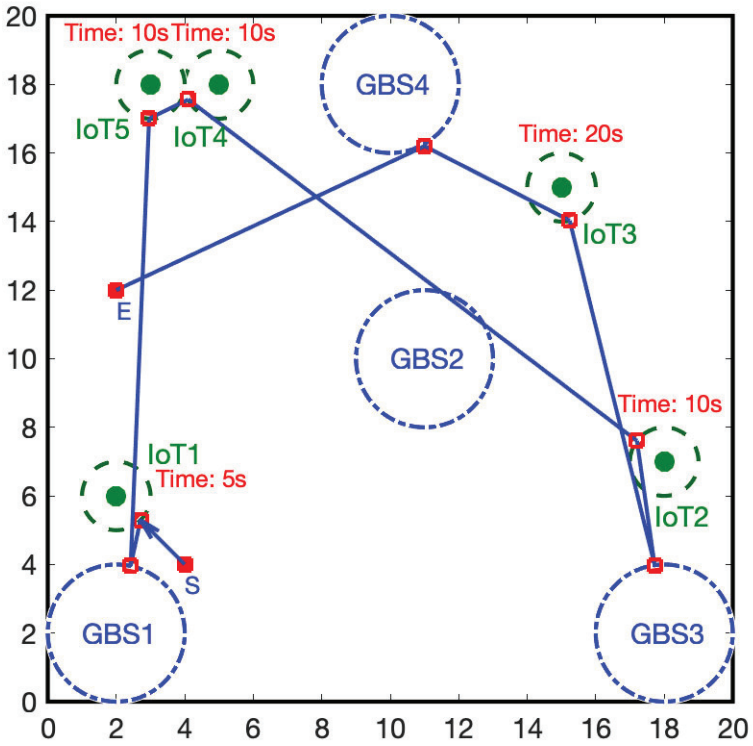


Fig. 54. **Example with a single UAV:** UAV trajectory obtained by the greedy heuristic algorithm.

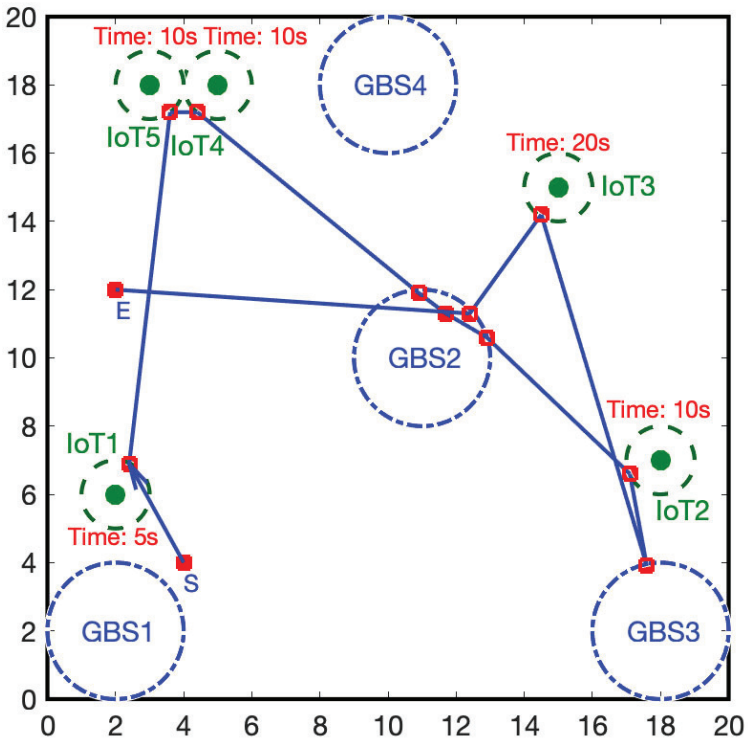


Fig. 55. **Example with a single UAV:** UAV trajectory obtained by the ILP based approach using CPLEX (with scale 10).

Table 13. Steps in the Greedy heuristic algorithm

Step	Decision
Step 1	$S \rightarrow$ **IoT1** $\rightarrow$ **GBS1** $\rightarrow E$
Step 2	$S \rightarrow$ IoT1 $\rightarrow$ GBS1 $\rightarrow$ **IoT2** $\rightarrow$ **GBS3** $\rightarrow E$
Step 3	$S \rightarrow$ IoT1 $\rightarrow$ GBS1 $\rightarrow$ **IoT4** $\rightarrow$ IoT2 $\rightarrow$ GBS3 $\rightarrow E$
Step 4	$S \rightarrow$ IoT1 $\rightarrow$ GBS1 $\rightarrow$ **IoT5** $\rightarrow$ IoT4 $\rightarrow$ IoT2 $\rightarrow$ GBS3 $\rightarrow E$
Step 5	$S \rightarrow$ IoT1 $\rightarrow$ GBS1 $\rightarrow$ IoT5 $\rightarrow$ IoT4 $\rightarrow$ IoT2 $\rightarrow$ GBS3 $\rightarrow$ **IoT3** $\rightarrow$ **GBS4** $\rightarrow E$

possible IoT and GBS permutations as described in Algorithm 5 and in Algorithm 6.

The UAV prioritizes the transmission of data from IoT2 to the nearest GBS, identified as GBS3. Following this prioritized sequence, the UAV is tasked with delivering data from IoT4. At the outset, the closest GBS for each IoT is identified, with GBS4 being the nearest for IoT4. The algorithm then seeks the optimal insertion point for IoT4 and GBS4 within the sequence, aiming to minimize the AoI increase. The strategy emphasizes minimizing GBS additions and finding the most effective IoT and GBS positions to reduce AoI. Consequently, the algorithm directs the UAV to IoT4 immediately after IoT1's data delivery, then to IoT2, allowing for simultaneous data delivery without incorporating GBS4. The algorithm's decisions are detailed in the Tab. 13. The result of this algorithm is depicted in Fig. 54.

Finally, we look at results obtained with ILP optimization. Here we use different scales of the map to show the impact of it in optimal solution. As the scale grows, we can obtain much better solutions, however, the running time increases dramatically. Thus, we stop the run after 2 hours. This method shows the lowest AoI possible in the given scenario an used as baseline to compare the performance of other solutions. The UAV trajectory obtained when we use a scale of 10 (so the map is considered like 200 by 200) is given Fig. 55.

In Table 14, we present a comparative analysis of the AoI for individual IoT devices utilizing various algorithms. First of all, the ILP results with different scales

Table 14. Comparison of AoI for individual IoT devices with all algorithms in single UAV scenario

Algorithms/IoT IDs	**1**	**2**	**3**	**4**	**5**
Brute-force	10.60	**11.61**	8.70	5.60	5.60
Greedy heuristic	0.68	**12.91**	12.17	12.91	12.91
ILP (Scale = 1)	8	**12**	10	3	3
ILP (Scale = 10*)	**10.3**	10.3	9.8	6	4.8
ILP (Scale = 100*)	9.96	**10.18**	8.84	**10.18**	**10.18**

obtained who that, the AoI decreases with more scale. Note that the ILP solutions for scale 10 and scale 100 do not reach 100% optimality (thus * is used), as we have constrained the computational time to a maximum of 2 hours for each scenario. Furthermore, it is evident from the comparison that ILP consistently yields the optimal solution relative to the other algorithms under consideration. On the other hand, greedy heuristic can result slightly higher AoI but runs much faster (finishes within seconds). We will compare the running times extensively later.

Multiple UAVs: For the multiple UAV scenario, we consider two UAVs different starting and ending points. The first UAV begins and concludes its mission at a location with coordinates (2,11), while the starting and ending location for the other UAV is (2,12). We use the same GBSs as in single UAV scenario however, we consider six IoT devices with locations and data generation times given in Table 15.

Table 15. Locations of IoT devices and data generation times in multi-UAV example

IoT ID	**1**	**2**	**3**	**4**	**5**	**6**
Coordinates	(18,7)	(15,15)	(5,18)	(3,18)	(5,5)	(18,18)
Data generation times (t_i)	5	10	10	10	5	1

In Table 16, we compare the AoI for each approach across each IoT device. As observed across all three maps, the IoT devices are divided into two groups, with each group being serviced by one UAV. This strategy assists in maintaining the maximum AoI limit. In Figures 56,57 and 58, we present the outputs from each

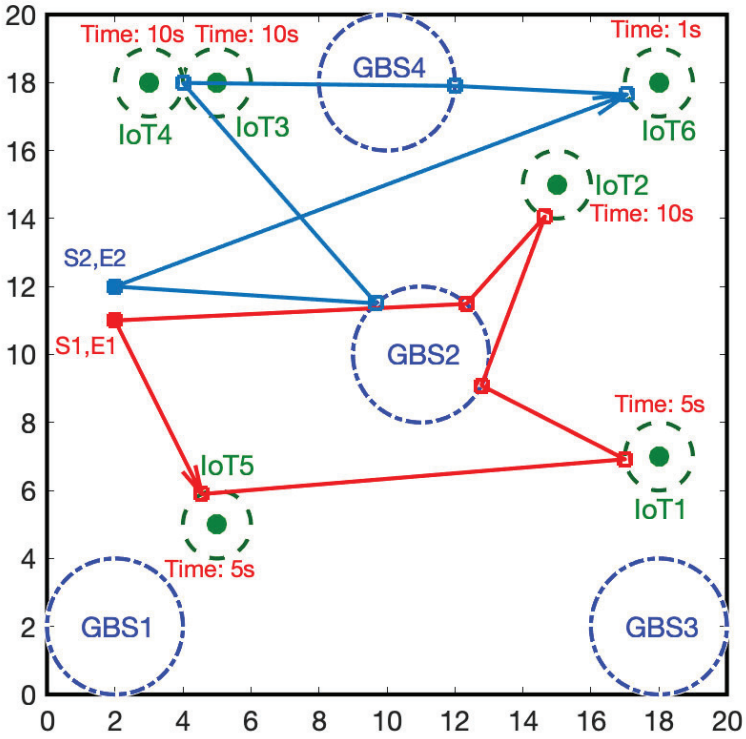


Fig. 56. **Example with two UAVs:** The UAV trajectory obtained by brute force approach.

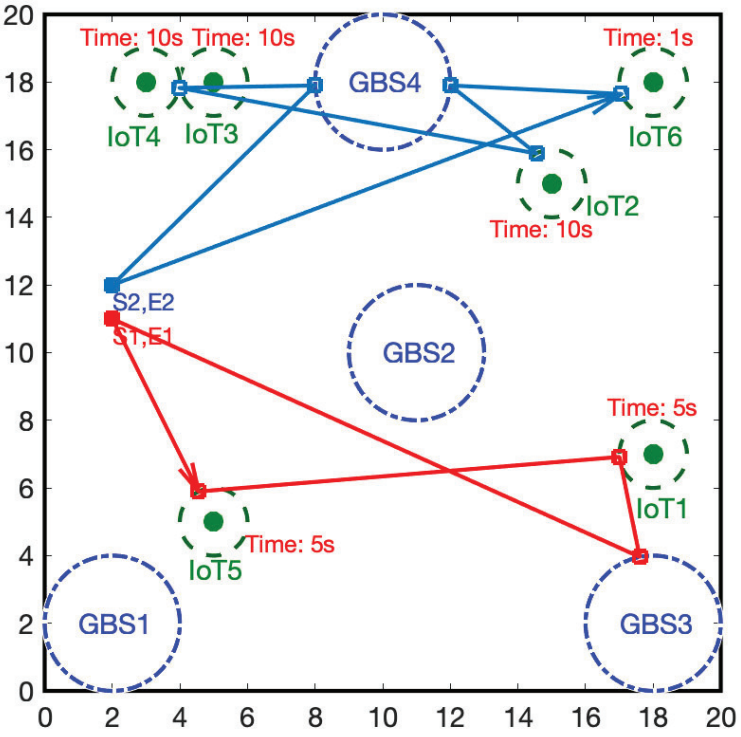


Fig. 57. **Example with two UAVs:** The UAV trajectory obtained by the greedy heuristic algorithm.

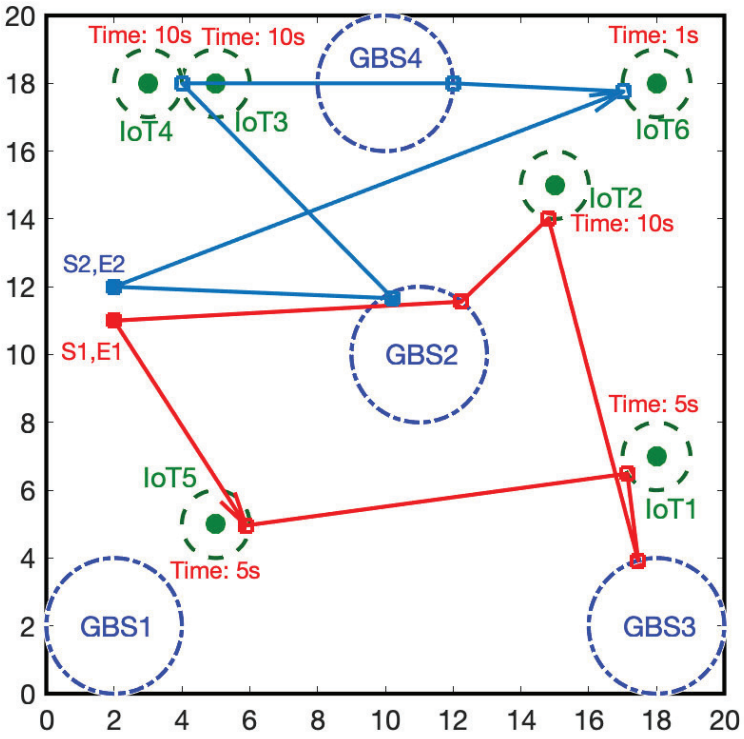


Fig. 58. **Example with two UAVs:** The UAV trajectory obtained by the ILP based approach using CPLEX (scale = 10).

Table 16. AoI for each IoT device across different algorithm in multi-UAV scenarios

Algos/IoT IDs	**1**	**2**	**3**	**4**	**5**	**6**
Brute force	8.62	8.01	8.89	8.89	8.62	**9.58**
Insertion	7.75	**10.58**	**10.58**	**10.58**	7.75	9.58
ILP (Scale = 1*)	10	10	8	8	3	**11**
ILP (Scale = 10*)	8.8	3.6	9.42	9.42	7	**9.6**
ILP (Scale = 100*)	7.11	9.31	9.22	9.22	7.11	**9.57**

of the algorithms we have developed. In Figure 56, we illustrate the output of the brute-force approach. This method considers all possible permutations of IoT devices and combinations of GBSs. Among all potential options, the depicted output yields the minimum maximum AoI. In the greedy heuristic-based approach, as illustrated in Figure 57, we initially identify the first IoT devices based on the delivery time for each UAV. The earliest IoT device that both UAVs can deliver data from is IoT 5. Following this, since the first UAV can deliver the data from IoT 5 faster than the second, it heads directly to IoT 5's location. According to our greedy algorithm, the closest IoT for the first UAV would then be IoT 1. The second UAV is responsible for delivering data from all other IoT devices. Since the data generation at IoT 6 occurs at time 1, it is prudent for the second UAV to deliver this data as early as possible to GBS4 before moving on to collect data from IoTs 2, 3, and 4. Upon gathering the data, it will deliver all of it to GBS4. The sequence in which the IoTs are visited by each UAV is the same as the one presented in Table 13, which applies to a single UAV. In Figure 58, we present the output of the ILP approach. The scale of this map is set to 10, and it achieves the lowest minimum maximum AoI compared to other approaches. However, it takes more time to produce this result.

5.4.2 Random Scenarios

In this part, we consider random scenarios and provide a general performance comparison of compared algorithms.

Single UAV: We generate 100 different scenarios with a specific scenario and get an average performance. We start with deploying different number of IoT devices on the map, while keeping the GBS count fixed at 4. The data generation time of IoT devices is set as randomly between 0 and 20. As it is shown in Fig. 59, the maximum AoI increases as the number of IoT devices increases for all algorithms. While the Cplex results are the best as expected, the permutational brute force approach gives close to that, and the proposed greedy heuristic can provide slightly larger but similar results. Next, we look at the impact of number of GBSs in the same way while keeping the number of IoT devices as 4. As shown in Fig. 60, increasing the number of GBSs results in a reduction of the maximum AoI thanks to the more coverage provided with more GBSs. The relationship between the compared algorithms is also similar to the previous scenario. Finally, in Figure 61, we examine the scalability of our algorithm, specifically within the heuristic approach (as we could not get results with Cplex and brute force for large number of IoT devices). The results depicted in this figure demonstrate that as the number of IoT devices increases in the same area (e.g., map size 40 referring to 40 by 40), maximum AoI increases but it increase in the slope decreases due to some convergence. Note that when the map size gets larger, the AoI gets bigger with the same number of IoT devices, but as new IoT devices are added to the area the convergence happens there eventually too.

Multi-UAV: Next, we obtain results for a multiple UAV scenario. To this end, we consider $|\mathcal{U}| = 2$ UAVs and generate 100 different scenarios with a specific number of randomly placed IoT devices, while maintaining a constant count of 4 GBSs. We

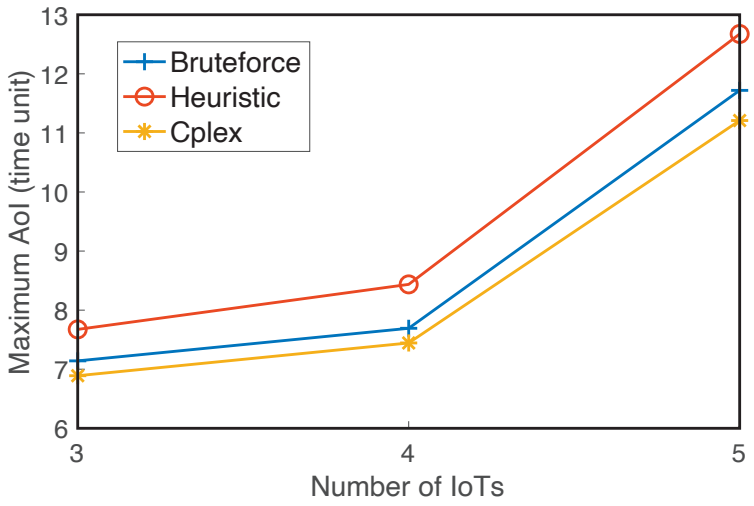


Fig. 59. Single UAV: Impact of varying number of IoT devices (GBS count = 4) on maximum AoI.

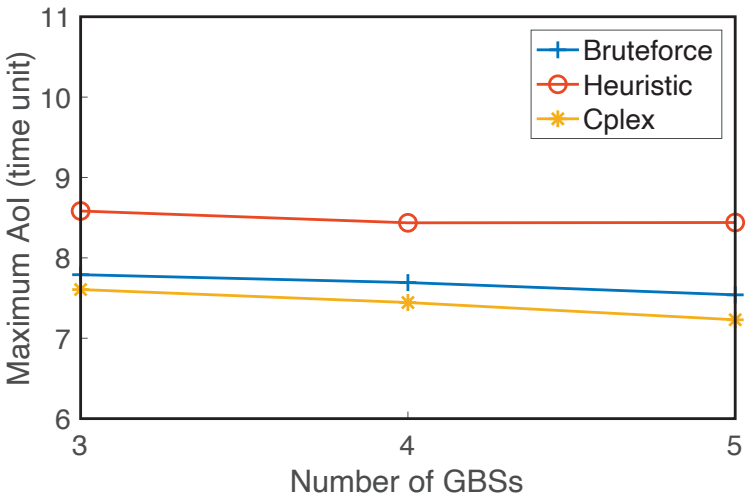


Fig. 60. Single UAV: Impact of varying number of GBSs (with IoT count = 4) on maximum AoI.

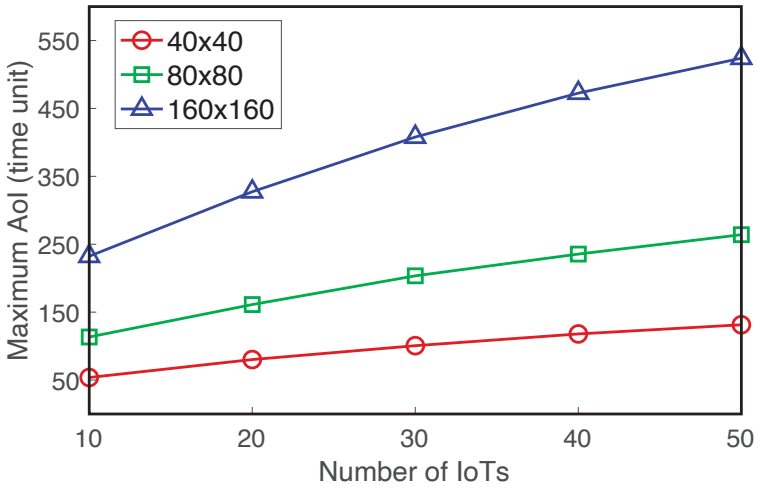


Fig. 61. Single UAV: Results with greedy heuristic with large number of IoT devices on different maps (area sizes).

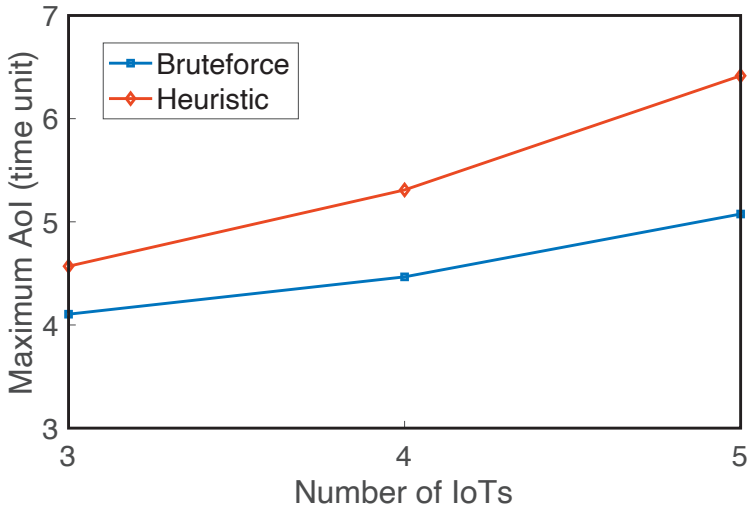


Fig. 62. Multi-UAV scenario: Impact of varying number of IoTs (GBS count = 4) on maximum AoI.

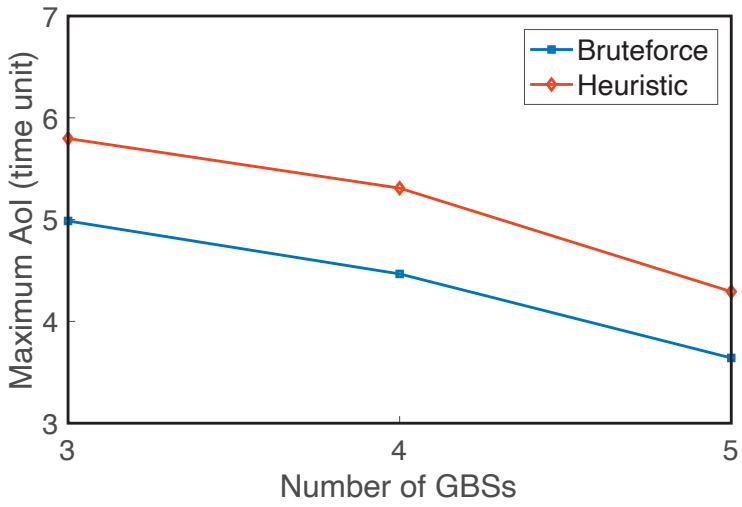


Fig. 63. Multi-UAV scenario: Impact of varying number of GBSs (with IoT count = 4) on maximum AoI.

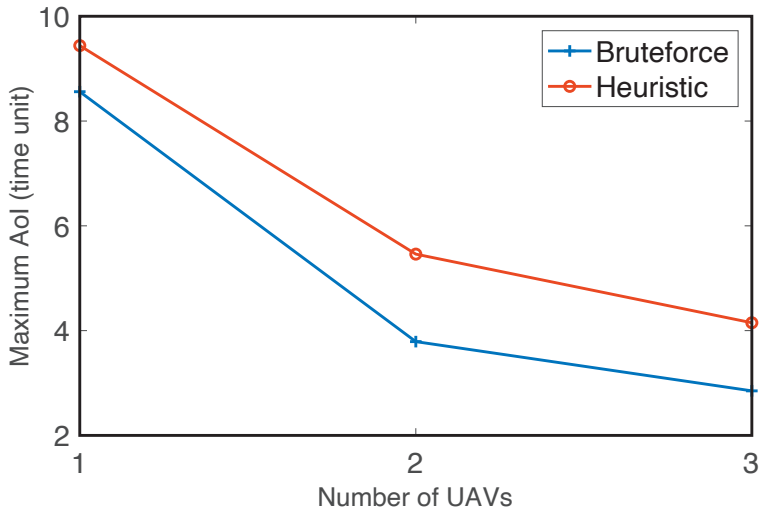


Fig. 64. Multi-UAV scenario: Impact of varying number of UAVs ($|\mathcal{G}|$=4, $|\mathcal{I}|$=4) on maximum AoI.

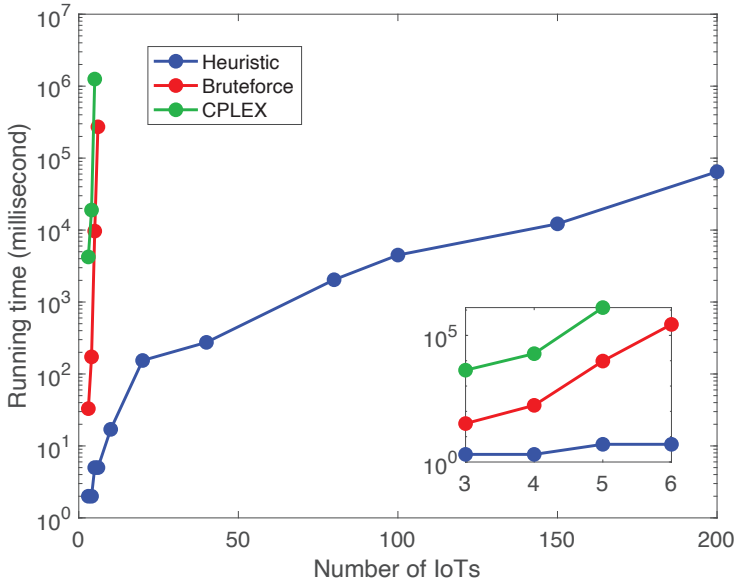


Fig. 65. Runtime comparison in single UAV case.

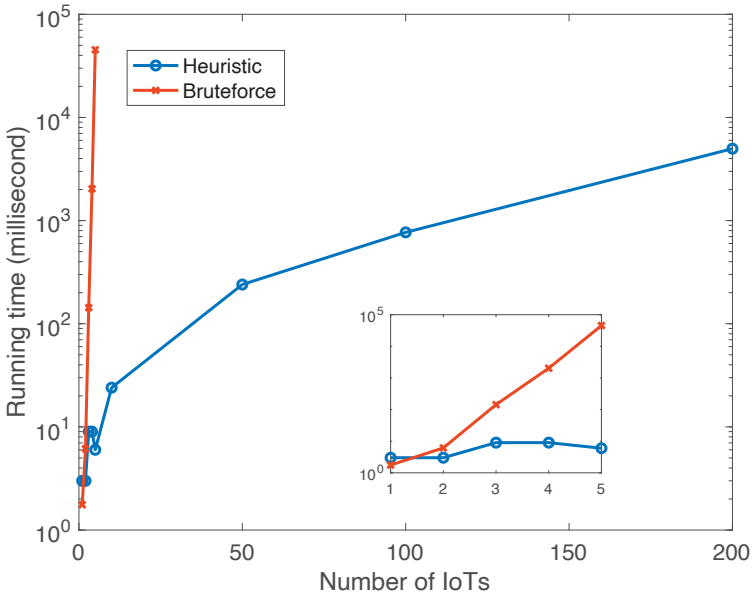


Fig. 66. Runtime comparison with different number of UAVs.

calculate the average maximum AoI. The data generation time for the IoT devices is again randomly set between 0 and 20.

As depicted in Fig. 62, the maximum AoI increases with the number of IoT devices for both algorithms. We could not obtain Cplex results for these scenarios as it took too long thus presenting brute force results, which give closer to Cplex results. We note that heuristic results provide more AoI than brute force, as expected, but the gap is increasing slightly. Subsequently, we analyze the influence of the number of GBSs on this scenario, holding the number of IoT devices constant at four. As the results in Fig. 63 shows, the AoI decreases with more GBSs, as the UAVs find more opportunity to upload the collected data, and the gap between brute force and heuristic based approach gets smaller with more GBSs. Finally, in Fig. 64, we show the impact of different number of UAVs on AoI. As expected, with more UAVs, the data of IoTs is both collected and uploaded to a nearby GBS much faster, yielding a lower AoI. Greedy heuristic based approach can provide close to brute-force results, while running in very short time, which is pivotal for real-time applications requiring up-to-date information.

Processing Time: Next, we compare the running times of different algorithms in different scenarios. We start with results with single UAV as presented in Fig. 65. The results clearly show that the processing time required for the ILP approach far exceeds that of the brute-force method, which in turn significantly surpasses the time taken by the heuristic approach with small IoT counts, but becomes much higher than heuristic approach with large number of IoTs.

In Fig. 66, we analyze the running times of the heuristic and brute-force approaches as the number of UAVs increases. The running time for the heuristic approach escalates due to the increased number of potential scenarios that need exploration. In contrast, with the brute-force approach, increasing the number of UAVs

leads to the distribution of IoT devices across more UAVs. This distribution simplifies the analysis by reducing the number of potential combinations that each UAV must evaluate, consequently decreasing the overall running time.

Next, we consider the scenario with 3 UAVs. As the results in Fig. 67 show, the running time increases heavily for brute force approach (Cplex results will be much longer than it, but we could not receive it as it look very long), underscoring its impracticality for real-time operations within multi-UAV systems. In contrast, the heuristic approach demonstrates a remarkable reduction in processing time, thereby making its suitability for scenarios demanding rapid decision-making and execution.

Finally, in Fig. 68, we show the maximum AoI obtained with brute force and heuristic approach with three UAVs and with larger number of IoTs. Since it takes very long to get results with brute force, we could not obtain results when IoT count is more than 5. However, as the trend with smaller number of IoTs show, the heuristic approach will follow what the brute force gives in different scenarios, while having very small computation cost.

Overall, all these AoI and running time results show the trade-off involved in algorithm selection for multi-UAV scenarios. While the brute-force (and ILP based) method may offer better results its high computational cost renders it less feasible for dynamic and practical environments. On the other hand, the heuristic algorithm, by virtue of its design, offers a balanced compromise between information freshness and computational efficiency, making it a more viable choice for real-time multi-UAV operations.

5.5 Conclusion

In this chapter, we have explored the path planning problem for a cellular-connected UAVs considering minimization of the maximum AoI for any data collected

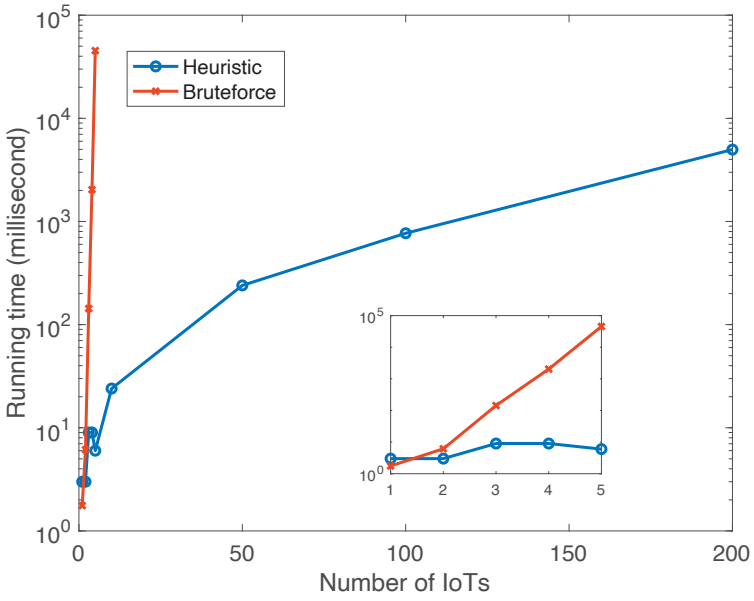


Fig. 67. Runtime comparison with large number of IoTs ($|\mathcal{U}| = 3$, $\mathcal{M} = 80 \times 80$).

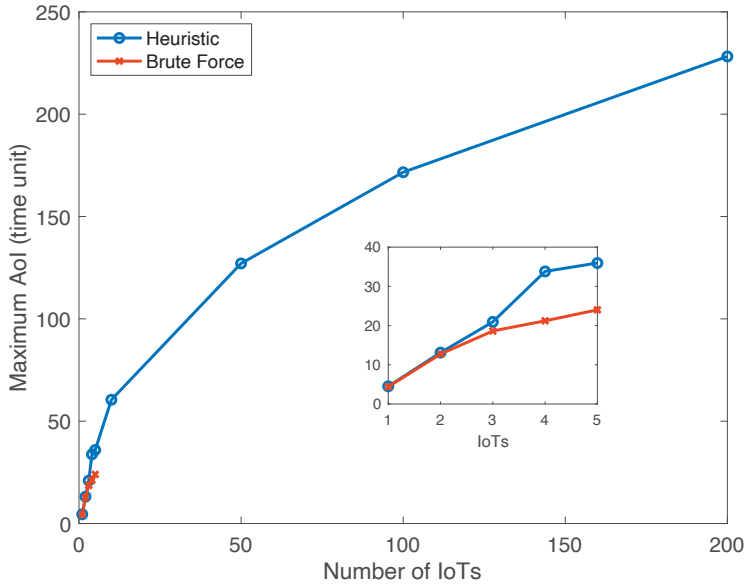


Fig. 68. Effect of number of IoTs on the maximum AoI with 3 UAVs and when the map size is 80x80.

as the main criteria. Different from previous works, AoI is defined as the time passes from the moment data is generated till it is uploaded to any of the nearby ground base stations by the UAVs. We developed both an ILP based model and a greedy heuristic based algorithm to find the path for the UAV. Through simulations with different scenarios, we have compared the results obtained by different methods and showed how their results differ in terms of several metrics.

In the future work, we will consider an online algorithm for the UAVs where only limited information about the IoT devices and the field (e.g., GBS locations) is known. We will also consider multi-UAV scenarios and more realistic communication models.

CHAPTER 6

AOI IN MESH NETWORKS

6.1 Introduction

Following emergencies and disasters, effective response management heavily relies on communication and the collection of environmental data. Communication in these scenarios can be facilitated through various methods, including satellite, ad hoc networks, local base stations, or a combination thereof. UAVs play a crucial role as they can serve as airborne base stations [83], ensuring reliable LoS connectivity with minimal interference, or function as relays to link ground users/devices to the main network infrastructure.

Existing research on UAV deployment in disaster and emergency contexts addresses several challenges, such as optimizing UAV positioning [84] to maximize user coverage and communication data rates, maintaining network topology [85], and efficient data packet routing [86]. Considerations also include the energy limitations and charging protocols of UAVs, as well as managing interference between UAVs and ground users for more practical implementations.

This study delves into the data collection process from terrestrial sensor nodes or IoT devices at emergency sites, particularly focusing on the freshness of the information, known as the AoI. Prior studies have thoroughly examined path planning for UAVs to optimize data collection from ground-based IoT devices [62], and the trajectory optimization concerning various factors like energy usage, time constraints, and environmental conditions [80, 87]. Recent introductions of AoI concepts have led to investigations into AoI-optimized data delivery in UAV-supported IoT frame-

works [61, 88, 9]. These assume data generation prior to UAV missions, whereas in practical scenarios, sensor data might be generated at varying times during UAV operations. Additionally, no prior work has considered the aspects of multi-UAV mesh networks and connectivity persistence. This paper presents a novel approach by integrating these elements to provide a comprehensive examination of UAV-assisted communication and data collection in emergency settings.

Fig. 69 illustrates an example scenario, with a set of UAVs, connected in a mesh network, collecting data from several ground IoT devices. We show two different time moments of the same UAV mesh network. At time t the data from the first two IoT devices is collected while in the next time moment, the data of the other IoT devices are collected. Note that the UAVs maintain their connectivity among each other. This helps up-to-date communication among them, which is vital in emergency sites, where there is no or minimal infrastructure. In this figure, we show that the UAVs also move and get into the range of a base station at a later time to upload the collected data to the backhaul, which makes it reach to the emergency response center to be processed.

Fig. 70 shows the AoI for the two different scenarios considered. If there is an undamaged GBS in the emergency site and the goal of the UAV mesh network is to delivery the collected data from the IoT devices to this GBS, the AoI will be from the moment the data is generated at t_1 until the moment it is delivered to the GBS at t_3. However, if there is a satellite connection possible from one of the UAVs, as soon as one UAV in the mesh network receives the data, it will be delivered to the backhaul through that satellite connection immediately (delay while exchanging data between connected UAVs is neglected assuming the data is very small).

The objective of this research is to develop strategies for the trajectory and connectivity of UAV mesh networks aimed at optimizing the AoI during data collection

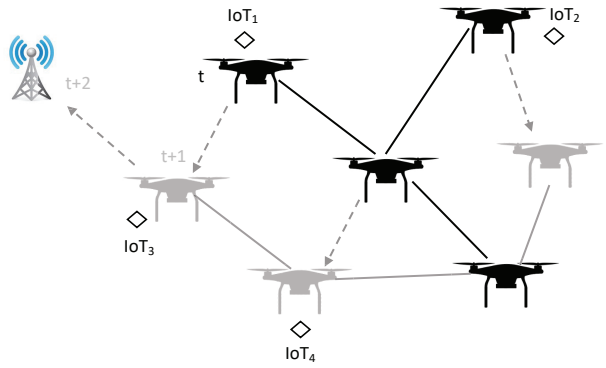


Fig. 69. A UAV mesh network of four UAVs collecting data from four ground IoT devices at time t (black) and $t+1$ (gray) and uploading their data to a base station at time $t+2$.

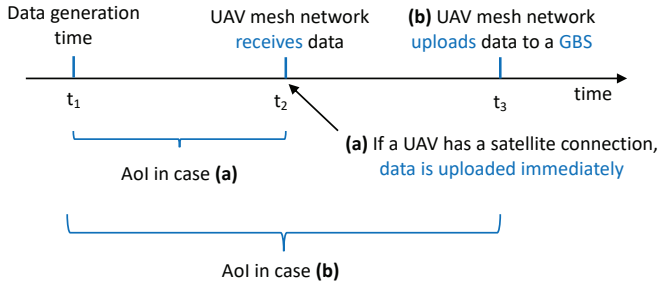


Fig. 70. Age of information in two different scenarios.

from IoT devices in specific scenarios. While some studies, such as [89], consider the AoI in UAV-assisted data collection from IoT devices, none address the transmission of data to a backhaul through satellites or to an operational base station in an emergency setting. In this context, AoI is typically defined as the duration from when data is first collected by a UAV to when it is received. Our definition diverges as we consider AoI from the moment data is generated until it reaches a backhaul entry point, such as a base station or satellite link. To our knowledge, only two studies [90, 91] align with our definition of AoI. However, these do not account for scenarios involving multiple UAVs interconnected within a mesh network in an emergency environment

with limited or no infrastructure.

The remainder of this chapter is structured as follows. Our system model and assumptions are detailed in Section 6.2, where we also define the problem and discuss the optimization models developed for various scenarios. Simulation results for these scenarios are presented in Section 6.3. We conclude the paper and outline future research directions in Section 6.4.

6.2 Problem Statement of Mesh Networks and ILP Formulation

We start with the problem (P1) where UAVs need to collect data from all ground IoT devices while maintaining the connectivity among them and the data delivery happens through a satellite connection from one of the UAVs. We define a decision variable for the location of each UAV at each time moment, defined by a set $L = \{L_0, L_1, L_2, ..., L_T\}$. Our main goal is to minimize the maximum AoI during this data collection process:

$$\min A_{max}. \tag{6.1}$$

Under this objective we first divide the map into a grid and develop an ILP based model.

In order to make sure the UAVs do not move more than their max speed between two consecutive time frames, we use

$$\text{dist}_{L(u,t-1)}^{L(u,t)} \leq V, \forall t \in [1, T], \forall u \in \mathcal{U}, \tag{6.2}$$

where, $\text{dist}_{L_i}^{L_j}$ represents the distance between two coordinates L_i and L_j.

Different UAVs are also not allowed to be in the same location at the same time

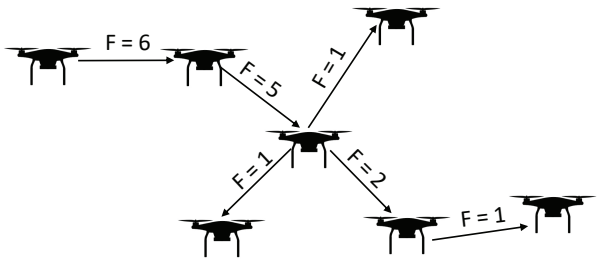


Fig. 71. Flow-based mesh network connectivity modeling.

by

$$\text{dist}_{L(u_i,t)}^{L(u_j,t)} > 0, \forall t \in [0,T], \forall u_i, u_j \in \mathcal{U}, i \neq j. \tag{6.3}$$

Since the data at each IoT may be generated any time, we also make sure that the UAV visits the IoT device's range and downloads or collects its data after its generation time (t_i):

$$V_i^D \geq t_i, \forall i \in \mathcal{I}, \tag{6.4}$$

where V_i^D denotes the time the UAV downloads the data of IoT i. Note that there could be multiple data generated by the same IoT and this formula applies to all such data.

We implement a flow-based connectivity management idea following the similar implementations in previous works that also consider full connectivity among graph nodes or UAVs [92]:

$$0 \leq F_{i,j}(t) \leq (|\mathcal{U}| - 1) \times A_{i,j}(t), \forall t \in T, \forall i, j \in \mathcal{U}, \tag{6.5}$$

where $F_{i,j}(t)$ denotes the virtual flow assumed to go from node (i.e., UAV) i to node j at time t. Here, we also use the connectivity information among the UAVs defined

by

$$A_{i,j}(t) = \begin{cases} 1, & \text{if } \text{dist}_{L(u_i,t)}^{L(u_j,t)} \leq R_U \\ 0, & \text{otherwise.} \end{cases}, \\ \forall t \in T, \forall u_i, u_j \in \mathcal{U}, u_i \neq u_j. \tag{6.6}$$

We define the incoming flow to each UAV by

$$I_F(u,t) = \sum_{u!=u'}^{\mathcal{U}} F_{u',u}(t), \forall u \in \mathcal{U}, \forall t \in T. \tag{6.7}$$

Also, the outgoing flow from each UAV by

$$O_F(u,t) = \sum_{u!=u'}^{\mathcal{U}} F_{u,u'}(t), \forall u \in \mathcal{U}, \forall t \in T. \tag{6.8}$$

Then, to make sure each UAV, except the initial UAV (i.e., u_0) that starts the flow, keeps one item in the flow before releasing it, we set

$$I_F(u,t) - O_F(u,t) = 1, \forall u \in \mathcal{U} \setminus \{u_0\}, \forall t \in T. \tag{6.9}$$

Note that we need at least one flow incoming to each UAV so that it is connected to the other UAVs. Moreover, the max incoming flow should be limited by maximum initial flow defined. To satisfy both, we have

$$1 \leq I_F(u,t) \leq |\mathcal{U}| - 1, \forall u \in \mathcal{U} \setminus \{u_0\}, \forall t \in T. \tag{6.10}$$

The outgoing flow from the initial UAV should be enough to reach all other UAVs, so we have

$$O_F(u_0,t) = |\mathcal{U}| - 1, \forall t \in T, \tag{6.11}$$

$$O_F(u_0,t) - I_F(u_0,t) = |\mathcal{U}| - 1, \forall t \in T. \tag{6.12}$$

The outcome of this flow based approach is illustrated in Fig. 71. The so-called source UAV sends enough flow to reach all UAVs and each gets one flow and sends the rest to others.

The connectivity between the UAV and each IoT device is determined based on the distance between the IoT device and each UAV u at a given time t.

$$c_i(u,t) \quad = \quad \begin{cases} 1, & \text{if } \text{dist}_{l_i}^{L(u,t)} \leq R_I \\ 0, & \text{otherwise.} \end{cases}, \\ \forall t \in T, \forall i \in \mathcal{I}, \forall u \in \mathcal{U}.$$

We then allow the collection of data by each UAV in range of IoT device in (6.13) and only one time as defined in (6.14).

$$d_i(u,t) \leq c_i(u,t), \forall t \in T, \forall i \in \mathcal{I}, \forall u \in \mathcal{U}, \tag{6.13}$$

$$\sum_{u \in \mathcal{U}} \sum_{\forall t \in T} d_i(u,t) = 1, \forall i \in \mathcal{I}. \tag{6.14}$$

In (6.15), we assign the UAV's IoT visit time to its pre-defined variable V_i^D by multiplying the value of $d_i(t)$ by t and then computing the sum. Since $d_i(t)$ is equal to 1 in only one of the ts, the value of V_i^D becomes equal to the IoT visit time.

$$V_i^D = \sum_{u \in \mathcal{U}} \sum_{\forall t \in T} (d_i(u,t) \times t)), \forall i \in \mathcal{I}. \tag{6.15}$$

Finally, we compute max AoI for any data collected from all IoT devices using the following equation. AoI here is defined as the time elapsed from data generation time t_i to its delivery or upload time at V_i^U, which is equal to V_i^D in this case.

$$A_{max} = \max\left\{(V_i^D - t_i)\right\}, \forall i \in \mathcal{I}. \tag{6.16}$$

6.2.1 Relaxed Problem using Critical Times

The problem P1 defined in the previous section considers the computation of each UAV's location at each time moment. However, this can be very costly and may not be very critical as long as the data is delivered with the same maximum AoI. To this end, in this section, we introduce an alternative approach in which we maintain the mesh network, ensuring that all UAVs are connected, but only during critical times, which are defined as time instances when a UAV downloads data from an IoT device. Unlike the previous problem, where we divide the total timeline into unit time slots, in this relaxed scenario, the total number of variables on the timeline is equal to the number of IoT devices. By adopting this strategy, we reduce the number of decision variables in the ILP model, thereby calculating the results more rapidly as the time complexity decreases. The sole limitation in this approach is that we cannot ensure the connectivity for all UAVs between the critical times during their flights.

In this problem, the set of locations that we look for each UAV is defined by the number of IoT devices. Let $L = \{L_0, L_1, \ldots, L_{|\mathcal{I}|}\}$ represent the set of locations we seek to determine along the UAV's route, and let $T = \{T_0, T_1, \ldots, T_{|\mathcal{I}|}\}$ denote the respective time moments. Similar to the previous problem, our primary goal is to minimize the maximum AoI during the data collection process.

In comparison with the previous problem, other than the reduced size of T, we

just update Equation 6.2 as follows:

$$\text{dist}_{L(u,t-1)}^{L(u,t)} \leq V \times (T_t - T_{t-1}), \quad \forall t \in T,\ \forall u \in \mathcal{U}. \tag{6.17}$$

That is, we just need to make sure the path for each UAV between the critical times is possible within the time difference of critical times considering their max speed.

6.2.2 Delivery to Ground Base Stations

In the third problem, we explore the scenario where the data delivery happens to a ground base station or GBS that is still functioning in the emergency site (no satellite connection from a UAV). In this scenario, we assume there are several GBSs across the map. The UAVs' mission now extends beyond merely downloading data from IoT devices. They must also upload this data to the GBSs. Given this expanded role, we revise the definition of the age of information as the time interval starting when the data is generated by an IoT device and ending when the UAV delivers the data to a GBS. As with the previous problems, our objective remains to maintain the mesh network and ensure connectivity among the UAVs while minimizing the maximum AoI.

Here, we consider an approach similar to previous problem (P2) using variables for only time critical moments. Compared to P2, however, we need to double the number of critical times. This adjustment is necessary because, in this scenario, the UAVs are required to perform two tasks for each data from the IoT devices: first, to download the data from the IoT device, and then to upload it to one of the GBSs. Let $L = \{L_0, L_1, \ldots, L_{2|\mathcal{I}|}\}$ denote the set of locations we aim to identify along the UAV's route, and let $T = \{T_0, T_1, \ldots, T_{2|\mathcal{I}|}\}$ represent the corresponding set of times. We also define the location of GBSs $\mathcal{G} = \{g_1, g_2, \ldots, g_n\}$. Our primary objective remains to minimize the maximum AoI throughout the data collection and delivery process.

In addition to constraint (6.4), the UAV must deliver the downloaded data to one of the GBSs, and this delivery must occur after one of the UAVs captures the data (all other UAVs get the same data due to mesh network based connectivity among UAVs). Therefore, we add the following constraint to our model:

$$V_i^U \geq V_i^D, \quad \forall i \in \mathcal{I}. \tag{6.18}$$

This ensures that the visit to upload data at a GBS (V_i^U) occurs on or after the visit to download data from an IoT device (V_i^D) for each IoT device i in the set $\mathcal{I}$.

Next, in addition to the connectivity constraint between a UAV and an IoT device for downloading of data as given in (6.13), we check out the connectivity between the UAV and GBS based on the distance between the GBS and each UAV u at a given time t.

$$g_i(u,t) \quad = \quad \begin{cases} 1, & \text{if } \text{dist}_{l_g}^{L(u,t)} \leq R_G \\ 0, & \text{otherwise.} \end{cases}, \\ \forall t \in T, \forall g \in \mathcal{G}, \forall u \in \mathcal{U}.$$

Given that our network is a mesh network and we operate under the assumption that UAVs are always connected, if one of the UAVs is within the range of a GBS, it can upload or deliver the data. This remains valid even if the UAV performing the upload is not the same one that initially downloaded the data. Equation (6.19) verifies whether at least one of the UAVs is within the range of a GBS. Furthermore, (6.20) indicates that uploading is feasible if any UAV is within the range of a GBS. To ensure that data from all IoT devices are uploaded to the GBSs, we integrate (6.21) into our model. We also keep the times for delivering each IoT's data by adding

Equation (6.22) to our model.

$$G(t) = min(1, \sum_{u \in \mathcal{U}} \sum_{\forall i \in \mathcal{G}} g_i(t)), \forall t \in T, \tag{6.19}$$

$$u_i(t) \leq G(t), \forall i \in \mathcal{I}, \forall t \in T, \tag{6.20}$$

$$\sum_{\forall t \in T} u_i(t) = 1, \forall i \in \mathcal{I}, \tag{6.21}$$

$$V_i^U = \sum_{\forall t \in T} (u_i(t) \times t), \forall i \in \mathcal{I}. \tag{6.22}$$

Finally, in this problem, we compute the maximum AoI for all IoT devices' data using the following equation. In this formula, the AoI for each IoT device is the time elapsed from the data generation time t_i to its delivery time (upload time) to a GBS at time slot V_i^U.

$$A_{max} = \max \left\{ (V_i^U - t_i) \right\}, \quad \forall i \in \mathcal{I}. \tag{6.23}$$

Multiple Objectives: In all problems, our primary objective is to minimize the max AoI. Then, we also set other objectives such as minimizing the average AoI and then minimizing the total path length of UAVs. These objectives are targeted in a prioritized manner using scalarization. However, to expedite solution time in ILP solver, we also consider hierarchical solution. That is, we initially just minimize the max AoI and find an optimized answer. In a subsequent step, we add the max AoI as a constraint in our model and aim to reduce the total AoI across all IoT devices. In the third stage, we additionally impose the total AoI as a constraint and focus on minimizing the overall travel distance of the UAVs.

6.3 Simulation Results

In this section, we present the simulation results for the problems studied using 4 UAVs. Our simulation map is a 40x40 unit grid. For all simulations, the IoT-UAV communication range is set as $R_I = 2$ units, and the UAV-UAV communication range is set as $R_U = 6$ units. In Problems P1 and P2, we look at scenarios with 4 IoT devices positioned at each corner of the map, specifically at coordinates (4,4), (4,36), (36,4), and (36,36). The data from these IoT devices is generated at time slots 0, 0, 3, and 5.

Figures 72, 73,74 and 75 illustrate the critical moments ($t = 0$, 2, 3, and 7) in the mission of UAVs as obtained by P2 model. From these results, it is clear that the UAV first collects data from IoT devices 2 and 1 (which start generating data at time 0) at time slots 0 and 2, respectively. It then proceeds to gather data from IoT devices 3 and 4 at time slots 3 and 7. The results show that the AoI for each IoT device is 2, 0, 0, and 2, with a maximum AoI of 2. Note that the results obtained with P1 has the same max AoI but the solution is obtained in a much longer time as it requires a connectivity among the UAVs at all times.

In Figures 76, 77,78 and 79, we see CPLEX's output for P3. This scenario involves 3 IoT devices and 1 GBS. The UAV-GBS communication range is set as $R_G = 2$ units. The UAV's mission is to collect data from the IoT devices and deliver it to the GBS. The results show that the UAV prioritizes data collection from IoT devices 1 and 3, which start generating data at time 0. After delivering this data to the GBS, the UAV then collects data from IoT device 2, yielding a maximum AoI of 4.

Figures 80, 81, and 82 presents heatmap of the UAV mesh network coverage during their paths. Comparing Fig. 80 and Fig. 81, we observe that our strategy

in P2 enhances UAV movement efficiency. The UAVs tend to position themselves near the map's center, enabling quicker data collection from each IoT device. This demonstrates the advantage of the *critical time* concept introduced in P2 over P1. On the other hand, in P3, the UAVs prioritize collecting data from IoT devices 1 and 3, which have earlier data generation times, before delivering this data to the GBS and then collecting data from the remaining device.

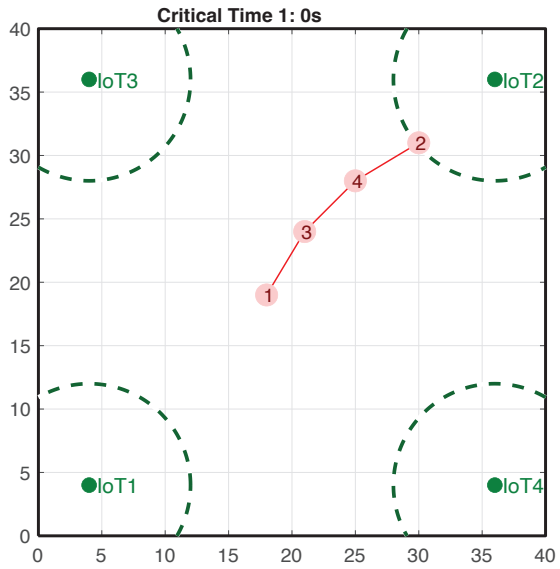


Fig. 72. Snapshot of the UAV mesh network at the first critical time. Data generation time for IoT device 1 is 0.

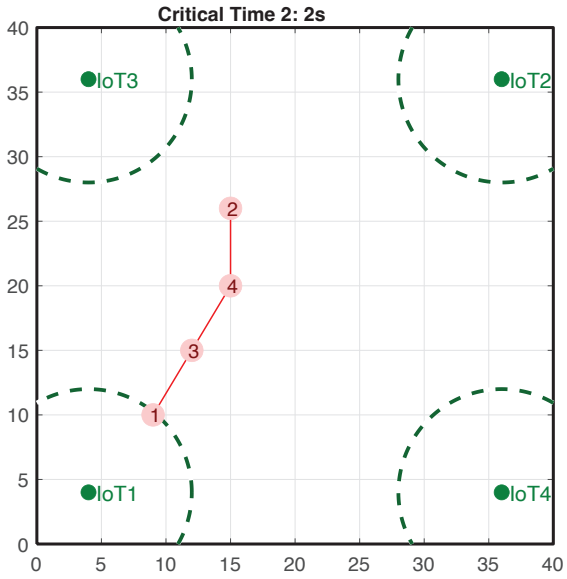


Fig. 73. Snapshot of the UAV mesh network at the second critical time. Data generation time for IoT device 2 is 0.

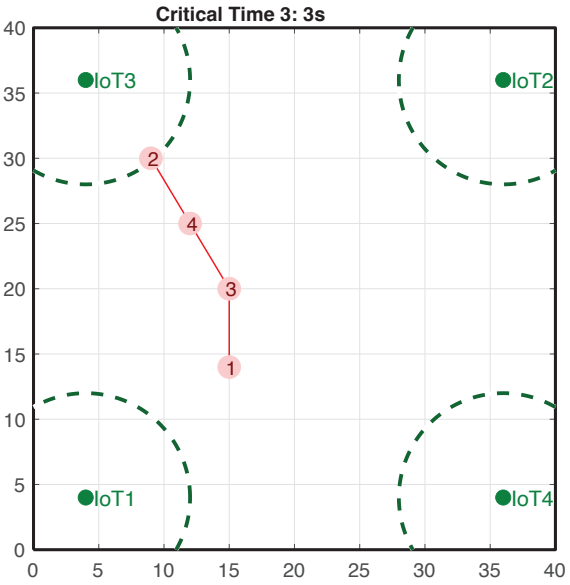


Fig. 74. Snapshot of the UAV mesh network at the third critical time. Data generation time for IoT device 3 is 3.

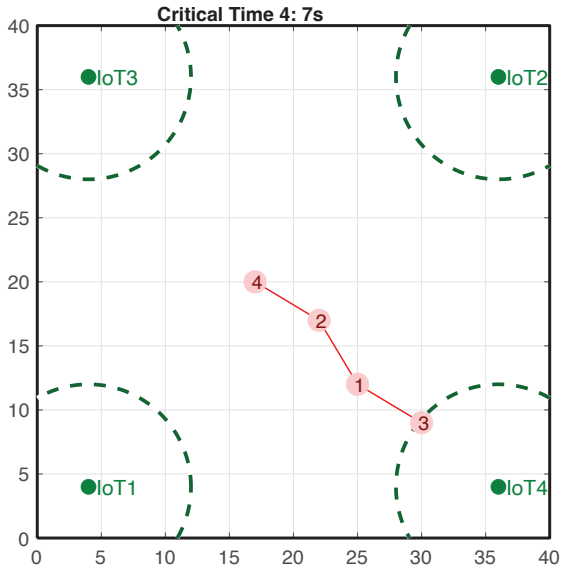


Fig. 75. Snapshot of the UAV mesh network at the fourth critical time. Data generation time for IoT device 4 is 5.

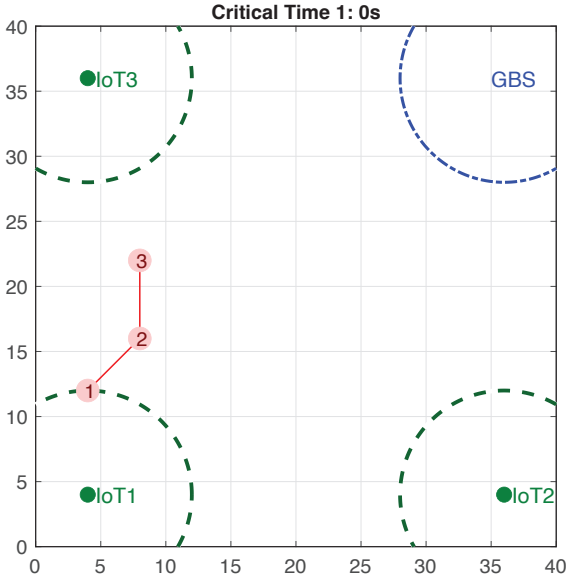


Fig. 76. The snapshot of the UAV mesh network at critical time 1.

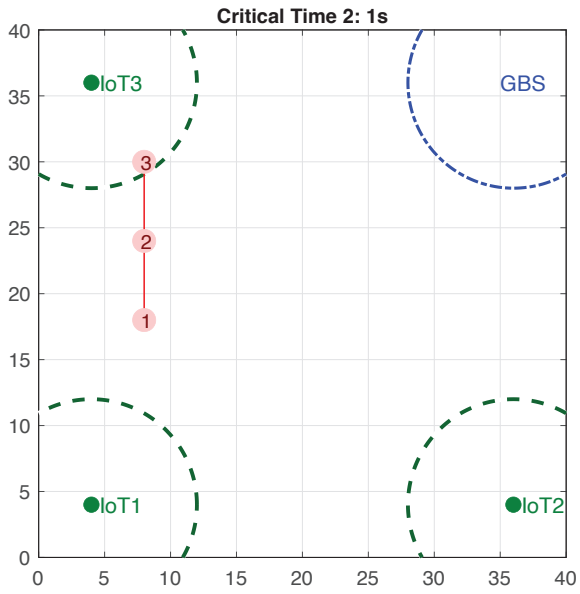


Fig. 77. The snapshot of the UAV mesh network at critical time 2.

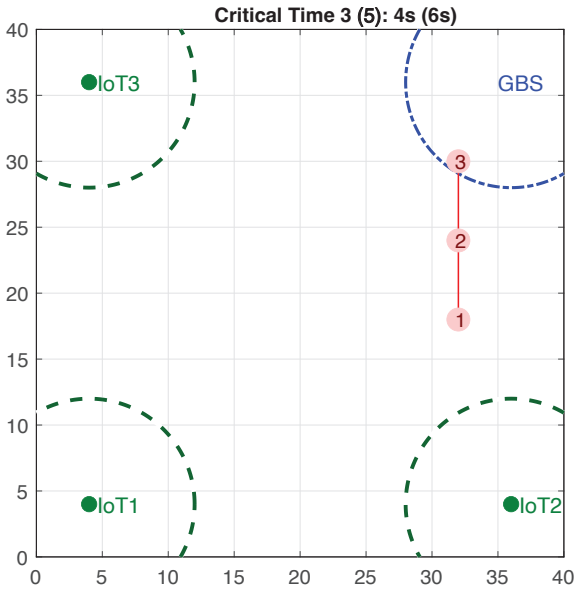


Fig. 78. The snapshot of the UAV mesh network at critical time 3.

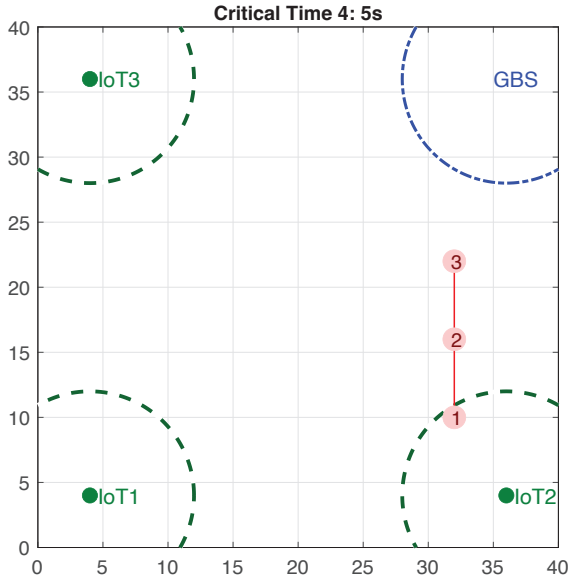


Fig. 79. The snapshot of the UAV mesh network at critical time 4.

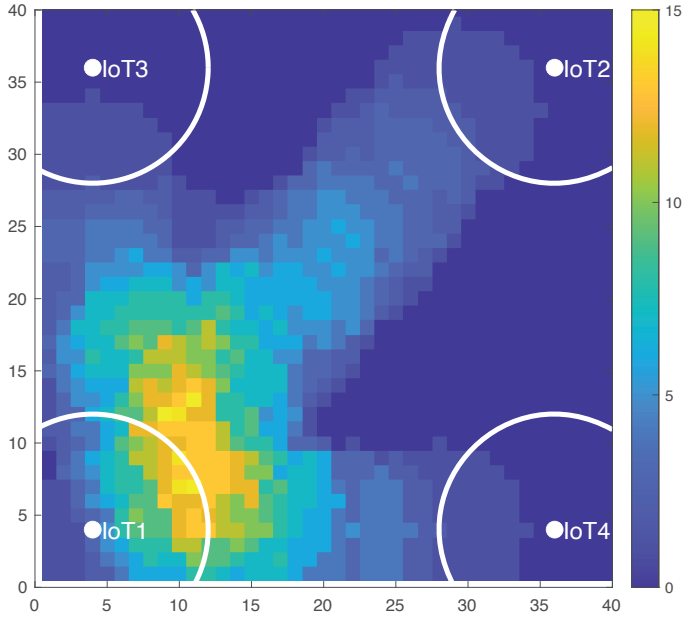


Fig. 80. Coverage heatmap of UAV mesh network during path in P1.

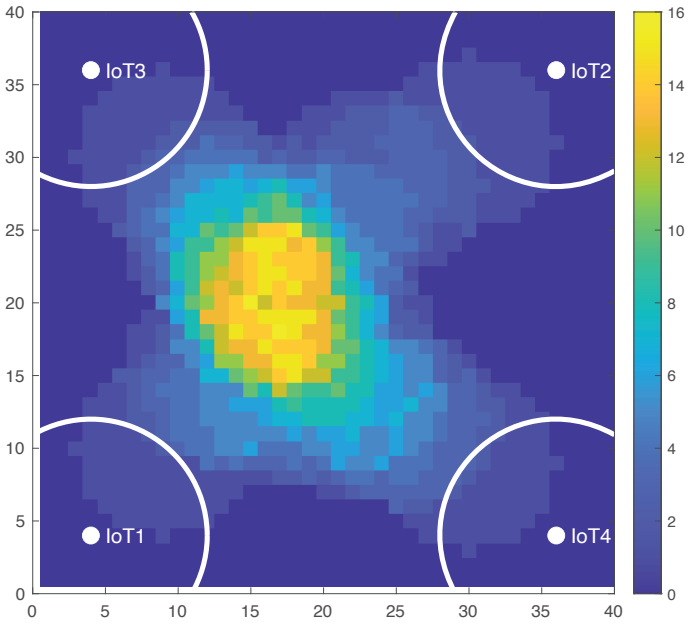


Fig. 81. Coverage heatmap of UAV mesh network during path in P2.

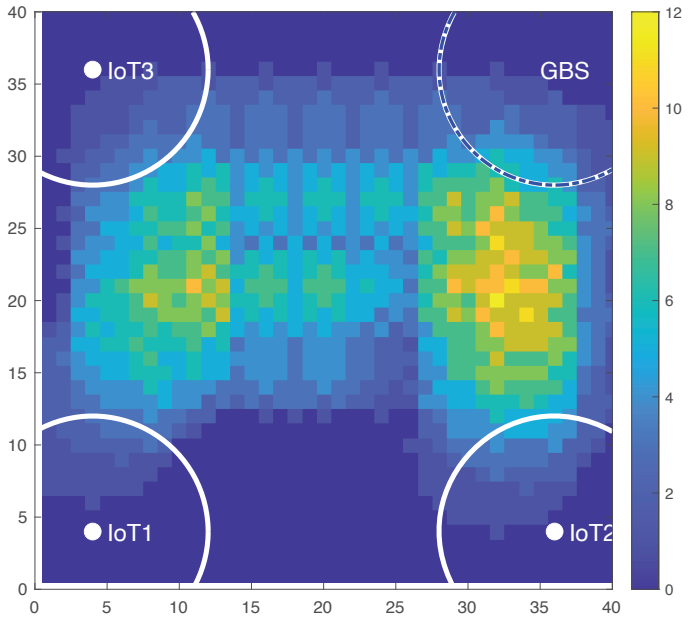


Fig. 82. Coverage heatmap of UAV mesh network during path in P3.

6.4 Summary of Contributions

We have explored the path planning problem for a UAV mesh network that collects data from ground IoT devices considering the minimization of the maximum age of information. We have studied several scenarios where the data delivery to the backhaul happens through satellite connection as well as through a few existing base stations in the area. Depending on the scenario and the associated AoI definition which is determined by the data delivery time to the backhaul, we formulated the problem using ILP to find the optimal path and mesh topology of UAVs towards minimizing the max AoI. We have considered relaxed ILP solutions as well to reduce the time complexity of the solutions. Through simulations, we have shown that the results in different scenarios are optimal and they have pros and cons to one another.

6.5 Future Works

In our future work, we will consider more realistic communication models and also study heuristic based solutions which can provide close to optimal ILP results with a much faster running time.

CHAPTER 7

CONCLUDING REMARKS

7.1 Our Contributions

In this concluding chapter, we draw together the threads of our research to provide a holistic perspective on the innovative approaches we have explored in the context of IoT communication, traffic aggregation, and UAV path optimization. These elements of our study converge to offer novel insights and practical solutions for dynamic and resource-constrained IoT and UAV networks environments. Here's a list of our contributions:

1. **Pioneering Traffic-Shifting-Based Aggregated Communication Model for IoT Devices:**
    - Introduced a novel traffic-shifting-based aggregated communication model for IoT devices.
    - Characterized by common Subscriber Identity Module sharing and traffic shifting.
    - Aimed to optimize resource utilization within the core network.
    - Explored the complexities of dynamic network scenarios with continuously entering and exiting devices.
    - Addressed unique challenges in traffic aggregation and stability within such environments.
    - Proposed heuristic-based aggregation algorithms as computationally efficient alternatives.

- Conducted extensive simulations to validate the efficacy of these heuristics in approximating ILP results while minimizing computational overhead.

2. **Practical Experiments with Mobile Devices:**

   - Conducted practical experiments involving mobile devices.
   - Automated connections to the core network for download tasks.
   - Confirmed the feasibility of the traffic aggregation concept.
   - Demonstrated reductions in core network memory and CPU resource utilization.

3. **UAV Path Optimization with Collaborative Approach:**

   - Explored how UAVs can optimize flight paths while staying within a specified connectivity limit.
   - Introduced a collaborative approach where multiple UAVs act as relays for each other to minimize outage time.
   - Developed mathematical models, including ILP, to find optimal paths for UAVs.
   - Created a simplified solution using graph-based approaches for UAV path optimization.
   - Conducted simulations to show that the simpler solution is almost as good as the optimal one.

4. **Minimizing Maximum AoI for Data Collected by UAVs:**

   - Addressed the problem of minimizing the maximum AoI for data collected by UAVs.

- Developed both ILP-based and heuristic-driven trajectory determination methods.
- Conducted a comprehensive comparative analysis between these two approaches across diverse scenarios.

5. **AoI in Mesh Networks**

- Developed a novel approach using UAVs in a mesh network to enhance data collection and delivery in minimal infrastructure areas, incorporating trajectory planning to maintain connectivity.
- Demonstrated the effectiveness of various strategies in minimizing AoI, comparing outcomes across different models including data transmission through satellites and to ground base stations.

Our journey began with the introduction of a pioneering traffic-shifting-based aggregated communication model for IoT devices. This model, characterized by common Subscriber Identity Module sharing and traffic shifting, seeks to optimize resource utilization within the core network. We navigated through the complexities of dynamic network scenarios, where devices continuously enter and exit the network, presenting unique challenges in traffic aggregation and stability.

To address these challenges, we initially turned to ILP as a mathematical model approach. However, we recognized the need for more computationally efficient solutions, leading us to propose heuristic-based aggregation algorithms. Through extensive simulations, we validated the efficacy of our heuristics algorithms, demonstrating their ability to closely approximate ILP results while minimizing computational overhead. Moreover, the introduction of a "smart removal" process between network moments emerged as a valuable enhancement, further optimizing our approach.

Our exploration extended beyond theoretical frameworks to practical experiments involving mobile devices. Leveraging readily available smartphones and an auto-clicking application, we automated connections to the core network for download tasks. This experiment confirmed the feasibility of our traffic aggregation concept, showcasing reductions in core network memory and CPU resource utilization. These outcomes present a promising solution for enhancing communication efficiency in scenarios involving a large number of IoT devices.

As another contribution in our research we explore how UAVs that can communicate with GBSs can optimize their flight paths while staying within a specified connectivity limit. We're looking at a situation where each UAVs needs to travel from a starting point to a final point, and all the UAVs collaborate to make sure they stay connected to the GBSs.

Again, we start by setting up a mathematical model and then by utilizing ILP we find the best paths for the UAVs. Unlike previous studies that focus on single UAV optimization, we introduce a collaborative approach where multiple UAVs act as relays for each other to minimize outage time.

To make things more manageable in terms of calculations, we create a simplified solution using graphs. This graph-based approach makes things easier to compute. We run simulations, and from the results, we see that our simpler solution is almost as good as the optimal one. This approach works well for various scenarios.

Our unique perspective incorporated the concept of minimizing the maximum AoI for data collected by UAVs. To tackle this problem, we developed both ILP-based and heuristic-driven trajectory determination methods. Simulations across diverse scenarios enabled us to conduct a comprehensive comparative analysis between these two approaches, shedding light on their respective strengths and weaknesses.

In summation, our research contributes a multifaceted toolkit for optimizing

communication in dynamic and resource-constrained environments. Whether through innovative traffic aggregation techniques for IoT devices, practical experiments on mobile devices, or UAV path optimization strategies, our work offers valuable insights and solutions that hold promise for the ever-evolving landscape of modern wireless communication networks. As we look to the future, the concepts explored here serve as a foundation for further advancements and applications in this dynamic field.

7.2 Future Works

The study has illuminated the complex process of improving the flight paths of UAVs by collaboration, and how these paths relate to GBSs both directly and indirectly. There is still much to uncover in this area. To start, even though our graph-based method is quick, it doesn't always find the best path. So, we need to make it more accurate. As the use of UAVs grows, it will be common to have many of them flying at once. This means we have to make our method work well with more UAVs. Also, we want to test how well our heuristic algorithm works in different conditions. In addtion, we want to:

- Test our method with many UAVs and in various maps.
- Look into using machine learning to improve our route decisions.
- Work with UAVs experts to test our methods in real-life situations.
- Make sure our method works well even with complicated maps.

Overall, we're looking to push the boundaries and make sure our research helps in improving how UAVs find their paths when working together. This improvement may help us in AoI studies.

The current heuristic algorithm to find minimum AoI, while fast and efficient,

tends to yield less accurate results in scenarios with numerous IoT devices, creating a gap between heuristic and optimal outcomes. Our research objectives for the future include:

- Investigating an alternative heuristic algorithm capable of achieving results closer to optimality, particularly in complex scenarios with a high number of IoT devices.
- Developing ILP and heuristic algorithms to optimize UAV paths, prioritizing AoI minimization. Additionally, we will explore collaborative strategies among multiple UAVs to further reduce AoI. These efforts aim to enhance the accuracy and efficiency of our approach.

REFERENCES

[1] Milan De Cauwer, Deepak Mehta, and Barry O'Sullivan. "The Temporal Bin Packing Problem: An Application to Workload Management in Data Centres". In: *Proc. IEEE Int. Conf. Tools with Artif. Intel. (ICTAI)*. 2016, pp. 157–164.

[2] Sjaak Wolfert et al. "Big data in smart farming–a review". In: *Agricultural systems* 153 (2017), pp. 69–80.

[3] Shuowen Zhang, Yong Zeng, and Rui Zhang. "Cellular-Enabled UAV Communication: Trajectory Optimization Under Connectivity Constraint". In: *arXiv preprint arXiv:1710.11619* (Oct. 2017).

[4] Zhu Xiao et al. "Resource management in UAV-assisted MEC: state-of-the-art and open challenges". In: *Wireless Networks* 28.7 (2022), pp. 3305–3322.

[5] Fengxian Guo et al. "Enabling massive IoT toward 6G: A comprehensive survey". In: *IEEE Internet of Things Journal* (2021).

[6] Huimin Hu et al. "AoI-minimal trajectory planning and data collection in UAV-assisted wireless powered IoT networks". In: *IEEE Internet of Things Journal* 8.2 (2020), pp. 1211–1223.

[7] Juan Liu et al. "Age-optimal trajectory planning for UAV-assisted data collection". In: *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 2018, pp. 553–558.

[8] Mohamed A Abd-Elmagid et al. "Deep reinforcement learning for minimizing age-of-information in UAV-assisted networks". In: *IEEE Global Communications Conference (GLOBECOM)*. 2019, pp. 1–6.

[9] Shuhang Zhang et al. “Age of information in a cellular internet of UAVs: Sensing and communication trade-off design”. In: *IEEE Transactions on Wireless Communications* 19.10 (2020), pp. 6578–6592.

[10] Syed Agha Hassnain Mohsan et al. “Unmanned aerial vehicles (UAVs): Practical aspects, applications, open challenges, security issues, and future trends”. In: *Intelligent Service Robotics* 16.1 (2023), pp. 109–137.

[11] Tao Li and Haitao Hu. “Development of the Use of Unmanned Aerial Vehicles (UAVs) in Emergency Rescue in China”. In: *Risk Management and Healthcare Policy* (2021), pp. 4293–4299.

[12] Kh Khujamatov et al. “Industry digitalization consepts with 5g-based iot”. In: *2020 International Conference on Information Science and Communications Technologies (ICISCT)*. IEEE. 2020, pp. 1–6.

[13] Dragos Mocrii, Yuxiang Chen, and Petr Musilek. “IoT-based smart homes: A review of system architecture, software, communications, privacy and security”. In: *Internet of Things* 1 (2018), pp. 81–98.

[14] Aimin Yang et al. “Security and privacy of smart home systems based on the Internet of Things and stereo matching algorithms”. In: *IEEE Internet of Things Journal* 7.4 (2019), pp. 2521–2530.

[15] Amirahmad Chapnevis, Ismail Güvenç, and Eyuphan Bulut. “IMSI Sharing-Based Dynamic and Flexible Traffic Aggregation for Massive IoT Networks”. In: *IEEE Internet of Things Journal* 9.19 (2022), pp. 18221–18237.

[16] Yong Zeng, Jiangbin Lyu, and Rui Zhang. “Cellular-connected UAV: Potential, challenges, and promising technologies”. In: *IEEE Wireless Communications* 26.1 (2018), pp. 120–127.

[17] Minghu Zhang and Xin Li. "Drone-enabled Internet-of-Things relay for environmental monitoring in remote areas without public networks". In: *IEEE Internet of Things Journal* 7.8 (2020), pp. 7648–7662.

[18] Meric Yilmaz Salman and Halil Hasar. "Review on Environmental Aspects in Smart City Concept: Water, Waste, Air Pollution and Transportation Smart Applications using IoT Techniques". In: *Sustainable Cities and Society* (2023), p. 104567.

[19] Manabu Ito et al. "Reducing State Information by Sharing IMSI for Cellular IoT Devices". In: *IEEE Internet of Things Journal* 3.6 (2016), pp. 1297–1309.

[20] Manabu Ito et al. "Aggregating cellular communication lines for IoT devices by sharing IMSI". In: *Proc. of IEEE International Conference on Communications (ICC)*. 2016, pp. 1–7.

[21] Dileep Kumar Vayilapelli et al. *Network management of subscriptions for IoT devices*. US Patent 10,212,685. Feb. 2019.

[22] Eyuphan Bulut and Ismail Güvenç. "Dynamically Shared Wide-Area Cellular Communication for Hyper-dense IoT Devices". In: *Proc. of the IEEE 43rd Conference on Local Computer Networks Workshops (LCN Workshops)*. 2018, pp. 64–69.

[23] TechNews. *Xiaomi's MIUI Now Features Virtual SIM Card for Overseas Travels*. 2017. URL: http://technews.co/2015/03/25/xiaomis-miui-now-features-virtual-sim-card-for-overseas-travels/..

[24] McKinsey. *E-SIM for consumers – a game changer in mobile telecommunications?* 2016. URL: https://www.mckinsey.com/industries/telecommunications/

our-insights/e-sim-for-consumers-a-game-changer-in-mobile-telecommunications.

[25] GSMA. *Remote SIM Provisioning for Machine to Machine*. 2017. URL: http://www.gsma.com/connectedliving/embedded-sim/..

[26] Leonardo Militano et al. "Device-to-device communications for 5G internet of things". In: *EAI Endorsed Trans. Internet Things* 1.1 (2015), pp. 1–15.

[27] 3GPP. *Architecture enhancemens to facilitate communication with packet data networks and applications*. v15. 2017. URL: TS%2023.682.

[28] Shun Sakurai et al. "Performance evaluation of a tunnel sharing method for accommodating M2M communication to mobile cellular networks". In: *Proc. of IEEE GLOBECOM Workshops*. 2013, pp. 157–162.

[29] Van-Giang Nguyen et al. "SDN/NFV-based mobile packet core network architectures: A survey". In: *IEEE Communications Surveys & Tutorials* 19.3 (2017), pp. 1567–1602.

[30] Vasudevan Nagendra et al. "MMLite: A Scalable and Resource Efficient Control Plane for Next Generation Cellular Packet Core". In: *Proc. of the ACM Symposium on SDN Research*. 2019, pp. 69–83.

[31] Ali Mohammadkhan and K. K. Ramakrishnan. "Re-Architecting the Packet Core and Control Plane for Future Cellular Networks". In: *Proc. of 27th IEEE International Conference on Network Protocols, ICNP*. 2019, pp. 1–4.

[32] Ali Mohammadkhan et al. "CleanG: A Clean-Slate EPC Architecture and ControlPlane Protocol for Next Generation Cellular Networks". In: *Proc. of the ACM Workshop on Cloud-Assisted Networking, CAN@CoNEXT, USA*. 2016, pp. 31–36.

[33] Xin Jin et al. “Softcell: Scalable and flexible cellular core network architecture”. In: *Proc. of the 9th ACM conference on Emerging networking experiments and technologies*. 2013, pp. 163–174.

[34] Konstantinos Samdanis et al. “Virtual bearer management for efficient MTC radio and backhaul sharing in LTE networks”. In: *Proc. of IEEE Int. Symp. Personal Indoor Mobile Radio Commun. (PIMRC)*. 2013, pp. 2780–2785.

[35] Younghwan Jung, Daehee Kim, and Sunshin An. “Scalable group-based machine-to-machine communications in LTE-advanced networks”. In: *Wireless Networks* 25.1 (2019), pp. 63–74.

[36] Yong Xiao, Marwan Krunz, and Tao Shu. “Multi-Operator Network Sharing for Massive IoT”. In: *IEEE Communications Magazine* 57.4 (Apr. 2019), pp. 96–101.

[37] Salman A. AlQahtani. “Analysis of an Adaptive Priority-Based Resource Sharing Scheme for Multiservice IoT Communications Over LTE-A Networks”. In: *Arabian Journal for Science and Engineering* 44.4 (Sept. 2018), pp. 3457–3472.

[38] Obada Al-Khatib, Wibowo Hardjawana, and Branka Vucetic. “Traffic Load-Based Spectrum Sharing for Multi-Tenant Cellular Networks for IoT Services”. In: *2018 IEEE International Conference on Communications (ICC)*. IEEE. May 2018.

[39] Shuowen Zhang, Yong Zeng, and Rui Zhang. “Cellular-enabled UAV communication: A connectivity-constrained trajectory optimization perspective”. In: *IEEE Trans. on Communications* 67.3 (2018), pp. 2580–2604.

[40] Yu-Jia Chen and Da-Yu Huang. "Trajectory Optimization for Cellular-Enabled UAV With Connectivity Outage Constraint". In: *IEEE Access* 8 (2020), pp. 29205–29218.

[41] Ursula Challita, Walid Saad, and Christian Bettstetter. "Interference management for cellular-connected UAVs: A deep reinforcement learning approach". In: *IEEE Trans. on Wireless Communications* 18.4 (2019), pp. 2125–2140.

[42] Weidong Mei, Qingqing Wu, and Rui Zhang. "Cellular-connected UAV: Uplink association, power control and interference coordination". In: *IEEE Transactions on Wireless Communications* 18.11 (2019), pp. 5380–5393.

[43] Hamidullah Binol et al. "Time optimal multi-uav path planning for gathering its data from roadside units". In: *88th IEEE Vehicular Technology Conference (VTC-Fall)*. 2018, pp. 1–5.

[44] Yan Pan et al. "An unmanned aerial vehicle navigation mechanism with preserving privacy". In: *IEEE International Conf. on Communications (ICC)*. 2019, pp. 1–6.

[45] Qingqing Wu, Yong Zeng, and Rui Zhang. "Joint Trajectory and Communication Design for Multi-UAV Enabled Wireless Networks". In: *arXiv preprint arXiv:1705.02723* (2017).

[46] Guangchi Zhang et al. "Trajectory optimization and power allocation for multi-hop UAV relaying communications". In: *IEEE Access* 6 (2018), pp. 48566–48576.

[47] Eyuphan Bulut and Ismail Guevenc. "Trajectory Optimization for Cellular-Connected UAVs with Disconnectivity Constraint". In: *2018 IEEE Interna-*

tional Conference on Communications Workshops (ICC Workshops). IEEE. May 2018.

[48] Cheng Zhan and Yong Zeng. "Aerial–ground cost tradeoff for multi-UAV-enabled data collection in wireless sensor networks". In: *IEEE Transactions on Communications* 68.3 (2019), pp. 1937–1950.

[49] Omid Esrafilian, Rajeev Gangula, and David Gesbert. "3D-map assisted uav trajectory design under cellular connectivity constraints". In: *IEEE International Conference on Communications (ICC)*. 2020, pp. 1–6.

[50] John D. Smith and Jane E. Doe. "UAV Path Planning under Outage Constraints for Robust Operations". In: *IEEE Transactions on Aerospace and Electronic Systems* 57.6 (2021), pp. 2500–2512.

[51] Richard Jones and Sophia Lee. "Optimization of UAV Trajectories Considering Connectivity Outages". In: *Proceedings of the 2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2022, pp. 362–369.

[52] Ursula Challita and Walid Saad. "Network formation in the sky: Unmanned aerial vehicles for multi-hop wireless backhauling". In: *IEEE Global Communications Conference (Globecom)*. 2017, pp. 1–6.

[53] Shuhang Zhang et al. "Joint trajectory and power optimization for UAV relay networks". In: *IEEE Communications Letters* 22.1 (2017), pp. 161–164.

[54] Md Moin Uddin Chowdhury et al. "3-D trajectory optimization in UAV-assisted cellular networks considering antenna radiation pattern and backhaul constraint". In: *IEEE Trans. on Aerospace and Electronic Systems* 56.5 (2020), pp. 3735–3750.

[55] Xianzhen Guo, Shuowen Zhang, and Liang Liu. "Trajectory Optimization of Cellular-Connected UAV for Information Collection and Transmission". In: *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*. IEEE. Dec. 2022.

[56] Dingcheng Yang et al. "An efficient trajectory planning for cellular-connected UAV under the connectivity constraint". In: *China Communications* 18.2 (Feb. 2021), pp. 136–151.

[57] Shuyan Hu et al. "Trajectory Planning of Cellular-Connected UAV for Communication-Assisted Radar Sensing". In: *IEEE Transactions on Communications* 70.9 (Sept. 2022), pp. 6385–6396.

[58] Chongxu Pei et al. "Trajectory Planning for Collaborative Transportation by Tethered Multi-UAVs". In: *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE. July 2021.

[59] Shubhani Aggarwal and Neeraj Kumar. "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges". In: *Computer Communications* 149 (2020), pp. 270–299.

[60] Amirahmad Chapnevis, Ismail Güvenç, and Eyuphan Bulut. "Traffic Shifting based Resource Optimization in Aggregated IoT Communication". In: *Proc. of 45th IEEE Conference on Local Computer Networks (LCN)*. 2020, pp. 233–243.

[61] Mohamed A Abd-Elmagid and Harpreet S Dhillon. "Average peak age-of-information minimization in UAV-assisted IoT networks". In: *IEEE Transactions on Vehicular Technology* 68.2 (2018), pp. 2003–2008.

[62] Zhiqing Wei et al. “UAV-Assisted Data Collection for Internet of Things: A Survey”. In: *IEEE Internet of Things Journal* 9.17 (2022), pp. 15460–15483.

[63] Mengying Sun et al. “AoI-Energy-Aware UAV-Assisted Data Collection for IoT Networks: A Deep Reinforcement Learning Method”. In: *IEEE Internet of Things Journal* 8.24 (Dec. 2021), pp. 17275–17289.

[64] Botao Zhu et al. “UAV Trajectory Planning for AoI-Minimal Data Collection in UAV-Aided IoT Networks by Transformer”. In: *IEEE Transactions on Wireless Communications* 22.2 (Feb. 2023), pp. 1343–1358.

[65] Peng Tong et al. “Deep Reinforcement Learning for Efficient Data Collection in UAV-Aided Internet of Things”. In: *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE. June 2020.

[66] Xin Zhang et al. “AoI-Minimal Power and Trajectory Optimization for UAV-Assisted Wireless Networks”. In: *2023 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE. Mar. 2023.

[67] Xiumin Zhu et al. “Path planning of multi-UAVs based on deep Q-network for energy-efficient data collection in UAVs-assisted IoT”. In: *Vehicular Communications* 36 (2022), p. 100491. ISSN: 2214-2096. DOI: https://doi.org/10.1016/j.vehcom.2022.100491. URL: https://www.sciencedirect.com/science/article/pii/S2214209622000389.

[68] Xingxia Gao, Xiumin Zhu, and Linbo Zhai. “AoI-Sensitive Data Collection in Multi-UAV-Assisted Wireless Sensor Networks”. In: *IEEE Transactions on Wireless Communications* 22.8 (2023), pp. 5185–5197. DOI: 10.1109/TWC.2022.3232366.

[69] Waleed Ejaz et al. “Efficient energy management for the Internet of Things in smart cities”. In: *IEEE Communications Magazine* 55.1 (2017), pp. 84–91.

[70] Federico Montori, Luca Bedogni, and Luciano Bononi. “A collaborative Internet of Things architecture for smart cities and environmental monitoring”. In: *IEEE Internet of Things Journal* 5.2 (2017), pp. 592–605.

[71] 3GPP. *Standards for IoT*. Dec. 2016. URL: `http://www.3gpp.org/news-events/3gpp-news/1805-iot_r14`.

[72] Navrati Saxena et al. “Efficient IoT Gateway over 5G Wireless: A New Design with Prototype and Implementation Results”. In: *IEEE Commun. Mag.* 55.2 (2017), pp. 97–105.

[73] Oluwatosin Ahmed Amodu and Mohamed Othman. “Machine-to-Machine Communication: An Overview of Opportunities”. In: *Computer Networks* 145 (2018), pp. 255–276.

[74] Galini Tsoukaneri et al. “Group communications in narrowband-IoT: Architecture, procedures, and evaluation”. In: *IEEE Internet of Things Journal* 5.3 (2018), pp. 1539–1549.

[75] Amarisoft. *Callbox Series*. 2022. URL: `https://www.amarisoft.com/products/test-measurements/amari-lte-callbox/`.

[76] Amirahmad Chapnevis and Eyuphan Bulut. “IMSI Sharing based Dynamic and Flexible Traffic Aggregation for Massive IoT Networks”. In: *IEEE IoT Journal* (2022).

[77] *Click Assistant - Auto Clicker*. 2022. URL: `https://play.google.com/store/apps/details?id=com.rise.automatic.autoclicker.clicker`.

[78] Samira Hayat, Evşen Yanmaz, and Raheeb Muzaffar. "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint". In: *IEEE Comm. Surveys & Tutorials* 18.4 (2016), pp. 2624–2661.

[79] Shuowen Zhang and Rui Zhang. "Trajectory design for cellular-connected UAV under outage duration constraint". In: *IEEE International Conference on Communications (ICC)*. 2019, pp. 1–6.

[80] Amirahmad Chapnevis et al. "Collaborative trajectory optimization for outage-aware cellular-enabled UAVs". In: *IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*. 2021, pp. 1–6.

[81] Khoi Khac Nguyen et al. "3D UAV trajectory and data collection optimisation via deep reinforcement learning". In: *IEEE Transactions on Communications* 70.4 (2022), pp. 2358–2371.

[82] Md Moin Uddin Chowdhury et al. "3-D trajectory optimization in UAV-assisted cellular networks considering antenna radiation pattern and backhaul constraint". In: *IEEE Trans. on Aerospace and Electronic Systems* 56.5 (2020), pp. 3735–3750.

[83] Nan Zhao et al. "UAV-assisted emergency networks in disasters". In: *IEEE Wireless Communications* 26.1 (2019), pp. 45–51.

[84] Rooha Masroor, Muhammad Naeem, and Waleed Ejaz. "Efficient deployment of UAVs for disaster management: A multi-criterion optimization approach". In: *Computer Communications* 177 (2021), pp. 185–194.

[85] Maurilio Matracia, Mustafa A Kishk, and Mohamed-Slim Alouini. "On the topological aspects of UAV-assisted post-disaster wireless communication networks". In: *IEEE Communications Magazine* 59.11 (2021), pp. 59–64.

[86] Muhammad Yeasir Arafat and Sangman Moh. "Location-aided delay tolerant routing protocol in UAV networks for post-disaster operation". In: *IEEE Access* 6 (2018), pp. 59891–59906.

[87] Junhai Luo et al. "Path Planning for UAV Communication Networks: Related Technologies, Solutions, and Opportunities". In: *ACM Computing Surveys* 55.9 (2023), pp. 1–37.

[88] Oluwatosin Ahmed Amodu et al. "Age of Information minimization in UAV-aided data collection for WSN and IoT applications: A systematic review". In: *Journal of Network and Computer Applications* (2023), p. 103652.

[89] Xijun Wang et al. "Cooperative Data Collection with Multiple UAVs for Information Freshness in the Internet of Things". In: *IEEE Transactions on Communications* (2023).

[90] Guqiao Chen et al. "Minimizing the Age of Information for Data Collection by Cellular-Connected UAV". In: *IEEE Transactions on Vehicular Technology* (2023).

[91] Amirahmad Chapnevis and Eyuphan Bulut. "AoI-Optimal Cellular-Connected UAV Trajectory Planning for IoT Data Collection". In: *IEEE 48th Conference on Local Computer Networks (LCN)*. 2023, pp. 1–6.

[92] Fatih Senel and Nils Aschenbruck. "Optimizing the Deployment of UAV for Mesh Access Network Coverage". In: *IEEE 48th Conference on Local Computer Networks (LCN)*. 2023, pp. 1–9.

VITA

Amirahmad Chapnevis received a B.S. degree in University of Isfahan (Iran) in 2015 and an M.S. degree in Amirkabir University of Technology (Iran) in 2019. He completed a Ph.D. degree in the Computer Science Department of Virginia Commonwealth University under the supervision of Dr. Eyuphan Bulut in May 2024. His current research interests include efficient communication modeling for massive IoT, and path planning in UAV networks.

Publications

1. **Chapnevis, Amirahmad**, and Eyuphan Bulut. "UAV Mesh Network Trajectory Planning for Age Optimal Data Collection in Infrastructureless Areas", accepted to appear in Proc. of IEEE International Conference on Communications (ICC) , Denver, June, 2024.

2. Vilela, Brandon Dominic, Kshitij Kokkera, **Amirahmad Chapnevis**, and Eyuphan Bulut. "UAV Control Using Eye Gestures: Exploring the Skies Through Your Eyes." Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing. 2023.

3. Farley, Jack, **Amirahmad Chapnevis**, and Eyuphan Bulut. "Generalized Path Planning for Collaborative UAVs using Reinforcement and Imitation Learning." Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing. 2023.

4. **Chapnevis, Amirahmad**, and Eyuphan Bulut. "AoI-Optimal Cellular-Connected

UAV Trajectory Planning for IoT Data Collection." IEEE 48th Conference on Local Computer Networks (LCN), 2023.

5. **Chapnevis, Amirahmad**, and Eyuphan Bulut. "Delay Optimal UAV Trajectory Planning for Secure Data Collection from Mobile IoT Networks." IEEE International Conference on Industrial Technology (ICIT). IEEE, 2023.

6. **Chapnevis, Amirahmad**, and Eyuphan Bulut. "Share-Not-To-Waste: Scalable IoT Networking through Subscriber Identity Sharing." IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS). IEEE.

7. **Chapnevis, Amirahmad**, Ismail Güvenç, and Eyuphan Bulut. "IMSI sharing-based dynamic and flexible traffic aggregation for massive IoT networks." IEEE Internet of Things Journal 9.19 (2022): 18221-18237.

8. **Chapnevis, Amirahmad**, et al. "Collaborative trajectory optimization for outage-aware cellular-enabled UAVs." IEEE 93rd Vehicular Technology Conference (VTC2021-Spring). IEEE, 2021.

9. **Chapnevis, Amirahmad**, Ismail Güvenç, and Eyuphan Bulut. "Traffic shifting based resource optimization in aggregated IoT communication." IEEE 45th Conference on Local Computer Networks (LCN). IEEE, 2020.