Improving Message Delivery in UAV-based Delay Tolerant Networks

Cintia Almeida Barroca

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors

Prof. António Manuel Raminhos Cordeiro Grilo

Prof. Paulo Rogério Barreiros d'Almeida Pereira

Examination Committee

Chairperson: Prof. Horácio Cláudio De Campos Neto

Supervisor: Prof. António Manuel Raminhos Cordeiro Grilo

Members of the Committee: Prof. Miguel Nuno Dias Alves Pupo Correia

November 2017

Acknowledgements

First of all, I would like to thank my parents for supporting me in every moment of this journey and for all the effort they made to allow me to follow my dreams, even if it meant being far apart. Anything I say is not enough to express my sense of debt to you. To my brother, for always being supportive in his own way. At the same level, a special word to Tiago Ventosa, whose love and kindness made everything easier, being the calm to my storm and my number one supporter. To the Ventosa family that adopted me, gave their whole support and affection throughout this long journey. A special thanks to António Manuel Ventosa for lending me his personal computer without any hesitations. Without it this thesis would not be here today. Pardon me for invading your personal space, I'll be forever grateful for your fantastic CPU. I would also like to thank my MEFT friends for all the support, tips and tricks in making this document.

Last, but definitely not least, a special thanks to professor António Grilo for the availability and guidance throughout this work, and to professor Paulo Pereira for all the feedback given.

Abstract

Delay Tolerant Networks are sparse networks where complete direct end-to-end paths between source and destination can seldom be established. Routing mechanism in DTN usually rely on nodes' mobility to connect disconnected nodes, carrying messages around the network to overcome path disconnection. The proactive DTN approach consists of introducing dedicated nodes whose only purpose is to establish communication between ordinary nodes, and relieve them from energy-consuming work, such as message routing and forwarding. This master thesis introduces a proactive scheme called Deadline Triggering Pigeon with Travelling Salesman Problem with Deadlines – DTP-TSP-D. We envision a DTN where ground nodes can only communicate by flying UAVs with the capacity of carrying messages from one location to another. Each UAV either belongs to one node or to a cluster of nodes, and their role is to hover over their home-ground node (or ferrying around their home cluster) until they are triggered to deliver messages directed to other ground nodes. The triggering criterium is based on the deadlines of the messages present in the UAV's buffer, evaluating its ability to deliver all of them in time. It uses a developed TSP Genetic Algorithm to compute the route that achieves the most (timely) deliveries. The performance of DTP-TSP-D has been compared to dedicated node based protocols found in the literature, namely: SIRA, MRT-Grid and HoP DTN. The performance metrics used were delivery ratio and average delay. The results show that DTP-TSP-D achieves greater delivery rates than its competitors, while keeping a consistent average delay.

Keywords: DTN, UAV, Message Ferry, Homing Pigeon, TSP, Deadlines

Resumo

As Redes Tolerantes a Atraso são redes esparsas cujas ligações entre nós raramente podem ser estabelecidas. Os protocolos de encaminhamento em DTN dependem da mobilidade dos nós para estabelecer ligação entre nós desconectados. A abordagem pró-ativa envolve a introdução de nós dedicados cujo único objetivo é estabelecer comunicação entre nós comuns e aliviá-los de tarefas de encaminhamento dispendiosas de energia. Neste trabalho, apresentamos um esquema pró-ativo denominado Deadline Triggering Pigeon with Travelling Salesman Problem with Deadlines - DTP-TSP-D. Idealiza-se uma DTN onde nós comuns apenas conseguem comunicar através da utilização de UAVs com capacidade de carregar mensagens. Cada UAV pertence a um ou mais nós, sendo o seu papel manter-se em cima do seu nó (ou percorrer um conjunto de nós seguindo uma rota de *ferry*) até ser desencadeado para o modo *pigeon* que o leva a entregar mensagens para outros nós. O desencadeamento é feito analisando os prazos de validade das mensagens presentes no buffer do UAV, avaliando a sua capacidade de entregá-las todas a tempo. Foi desenvolvido um algoritmo genético que resolve o TSP determinando a rota que proporciona mais entregas dentro do prazo, o qual é usado tanto na fase de desencadeamento como na fase de entrega. O DTP-TSP-D foi comparado com protocolos que usam nós dedicados, nomeadamente: SIRA, MRT-Grid e HoP-DTN. As métricas de desempenho utilizadas foram a taxa de entrega e o atraso médio. Os resultados mostram que o DTP-TSP-D atinge taxas de entrega maiores do que os restantes algoritmos, mantendo um atraso médio consistente.

Palavras-chave: Redes Tolerantes a Atrasos, Veículo Aéreo não Tripulado, Ferry, Pombo Correio, Problema do Caixeiro Viajante, Prazos

Table of Contents

Index of Figures

Index of Tables

List of Acronyms

ACO	*Ant Colony Optimization*
CTP-TSP-D	*Capacity Triggered Pigeon with TSP Deadline*
DTN	*Delay Tolerant Network*
DTP-TSP-D	*Deadline Triggered Pigeon with TSP Deadline*
FCFS	*First Come, First Serve*
GA	*Genetic Algorithm*
MANET	*Mobile Ad Hoc Network*
MF	*Message Ferrying*
MRT	*Multiple Route*
SA	*Simulated Annealing*
SIRA	*Single Route Algorithm*
SRT	*Single Route*
TSP	*Travelling Salesman Problem*
UAV	*Unmanned Aerial Vehicle*
VANET	*Vehicular Ad Hoc Network*

Chapter 1

Introduction

This chapter serves as an introduction of the studied theme, provides the scope and objectives of this dissertation, as well as the motivation behind it. Finally, the thesis' content and organization are also outlined.

1.1 Motivation

Delay Tolerant Networks' (DTNs) [1] most important property is that complete end-to-end paths between source and destination do not always exist. Routing mechanisms in DTN usually rely on nodes' mobility to bridge gaps in space and time, carrying messages around the network to overcome path disconnection using a *store-carry-and-forward* scheme. A message originated from the source is first forwarded to an intermediate node that is supposed to be (or get) closer to the destination. Then the receiver stores the message locally and carries it for a while until a next contact is available.

Most solutions for routing in DTN are reactive approaches, where nodes rely on their inherent movements to disseminate data when they encounter each other, and their mobility is not controlled. Due to unpredictable mobility, this random behavior leads to low delivery rates and large delays. On the other hand, a proactive approach is to introduce extra nodes (messengers), which move around the network actively and visit ordinary nodes regularly, creating chances to re-connect disconnected nodes.

There are several reasons to choose such a proactive approach with extra auxiliary mobile nodes. Firstly, it can serve a variety of DTNs, especially in some challenging situations where it is impossible to deliver messages based on the movements of ordinary nodes only, like in disaster recovery, where mobile nodes (helicopters, Unmanned Aerial Vehicles (UAV), or personnel) equipped with communication devices capable of storing a large number of messages, can be commanded to follow a trajectory that interconnects disconnected user partitions. Secondly, these messengers can be controlled to provide predictable end-to-end quality of service such as maximum message delivery delay. Thirdly, these messengers are dedicated to message transmission tasks, which can relieve ordinary nodes from energy-consuming work such as message routing and forwarding.

The early work starts with using a single messenger as a ferry, for forwarding messages in DTNs [8]. However, these single ferry approaches are not scalable in traffic load, network size or geographic coverage, and thus multiple ferries are adopted [7], [9], [10], [11]. The most basic extension of the simple ferry scheme is to have multiple ferries treated as identical and not require them to cooperate with each other [7]. Zhao et al [9] began to investigate interaction between ferries and studied whether the multiple ferries should take the same (single) route or multiple routes, proposing four algorithms.

Zhang et al [10], explored the design of traveling routes for multiple ferries, proposing three new schemes. Later, Guo et al [12], [13] proposed an alternative usage of auxiliary nodes. Instead of having *shared messengers,* they propose an approach in which all the ordinary nodes own at least one *dedicated messenger*, called pigeon due to its similarity to the ancient messaging system using homing pigeons.

In this master thesis, a hybrid approach between shared and dedicated messengers is proposed. A new scheme for DTN was developed, based on proactive schemes like message ferrying and homing pigeon, named Deadline Triggered Pigeon with Travelling Salesman Problem with Deadlines (DTP-TSP-D). In this approach, the auxiliary node can act as a shared messenger by ferrying over a cluster of nodes, collecting and delivering messages. However, it can also act as a dedicated messenger, pigeon, whenever it is triggered to do so, going to deliver messages outside its assigned cluster. The goal is to simulate the use of UAVs as messengers in a DTN scenario, where they will act as both ferry and pigeon, and attempt to achieve better results in terms of average delay and delivery ratio than the schemes mention before.

1.2 Thesis Outline

This thesis is organized in six chapters, with this first one introducing the work in terms of context, motivation and main objectives.

- Chapter 2: state of the art – introduction of Delay Tolerant Networks, Unmanned Aerial Vehicles and Traveling Salesman Problem.
- Chapter 3: presents the analysis and implementation of existing algorithms in the literature.
- Chapter 4: presents the developed model.
- Chapter 5: presents the evaluation and comparison of the algorithms presented in the previous chapters.
- Chapter 6: concludes this thesis with a summary and suggestions for future work.

Chapter 2

State of the Art

In this chapter, several concepts related to the topic of this thesis are introduced and discussed in order to have the theoretical background needed to understand every aspect of this master thesis. The covered topics are Delay Tolerant Networks (DTN), Unmanned Aerial Vehicles (UAV) applications in DTNs and, finally, the Traveling Salesman Problem.

2.1 Delay Tolerant Networks

Internet usability is based on the assumption of existing end-to-end paths to establish a continuously available bidirectional connection between source and destination, having low packet loss rate and low propagation delay [2]. However, many communication environments do not meet these requirements, being characterized by intermittent connectivity, long or variable delay, asymmetric data rates and high error rates. DTN [1] overcome these problems using a store-carry-and-forward scheme, where messages are moved from a storage place in one node to a storage place in another node, along a path that eventually reaches the destination. Figure 1 illustrates this scheme.

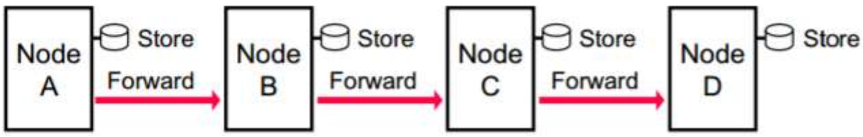


Figure 1 - Store-Carry-Forward Scheme in DTN.

A node will need to buffer the data until it gets an opportunity to forward it. Determining which node is best to forward each message is one of the difficulties of designing a protocol that is efficient and successful in delivering messages to their destinations in this kind of environment. If a message cannot be delivered immediately due to network partition, the best carriers for a message are those that have the highest chance of successful delivery, i.e., the highest delivery probabilities. Under the conditions of a DTN with sporadic contact opportunities, the main objective of routing is to maximize the probability of delivery at the destination, while minimizing the end-to-end delay.

Even though there is no standard taxonomy for DTN routing protocol classification, there are various interpretations on how to distinguish the protocols [3]. Protocols can be simply divided in those which use dedicated nodes with the assigned task of collecting and delivering messages from the source to the destination nodes, and those who don’t.

2.1.1 Routing protocols without dedicated nodes

In the category of schemes who do not single out special nodes for delivery purpose, messages delivery only relies on the inherent movement of nodes. Next follows some examples of this kind of protocols.

Epidemic:

Epidemic routing [4] implements flooding in a DTN, and was named after a technique for message forwarding that emulates how a disease spreads through direct contact in a population during an epidemic. In Epidemic protocol, the disease spreading is a message that must reach one or more destinations. Messages received at intermediate nodes are forwarded to all neighbors (except the one who sends the messages) without using any prediction of the link or path forwarding probability. Epidemic routing is a natural approach when no information can be determined about the movement patterns of nodes in the system. The epidemic routing protocol finds the optimal path for message delivery to destinations with the smallest delay, since it explores all available communication paths to deliver messages. The major drawback of this scheme is the waste of resources such as buffers, bandwidth and nodes power, due to forwarding of multiple copies of the same message. It causes contentions when resources are limited, leading to congestion and message dropping.

Spray and wait:

Spray and wait [5] is an alternative to reduce resource consumption from epidemic routing, using a spray and wait mechanism controlling the level of message spreading throughout the network. This protocol assumes no prior knowledge of network topology and node movement patterns, and simply forwards multiple copies of received messages using flooding technique. The difference from epidemic relies on the limited number of copies of each message. Spyropolous et al. [5] proved that the minimum number of copies to get the expected delay for message delivery depends only on the number of nodes in the network and is independent of the network size and the range of transmission.

PROPHET:

Probabilistic ROuting Protocol using History of Encounters and Transitivity (PROPHET) [6] assumes that nodes mobility is not purely random, but has several deterministic properties e.g. repeating behavior. The PROPHET scheme believes that mobile nodes tend to pass through some locations more than others, implying that passing through previously visited locations is highly probable. Thus, the nodes that met each other in the past are more likely to meet in the future. Similar to epidemic routing, whenever a node contacts with other nodes in the network, they exchange summary vectors containing the delivery predictability values for destinations known by each node. Then, they exchange only messages for which the neighbor has a higher delivery predictability.

Many more protocols could be presented in this section, [3] presents a great survey on dissemination protocols and their characteristics.

2.1.2 Dedicated node-based routing protocols

On this subsection, we have those protocols which use dedicated nodes to collect and deliver messages throughout the network. Next are presented some protocols that fall into this category.

Meet and Visit:

Meet and Visit [7] was suggested for forwarding messages in structures with mobile source and fixed destination nodes. This scheme actively explores information about meeting of peer nodes and their visiting locations. The knowledge regarding meetings and visiting places is stored at each node and used to estimate message delivery probabilities. Three important assumptions are introduced in the Meet and Visit protocol: (i) nodes have unlimited buffer space. (ii) there is infinite link capacity and (iii) destination nodes are fixed.

Message Ferrying:

Message Ferrying Routing [8] is a scheme that exploits controlled mobility to provide physical connectivity between otherwise disconnected nodes. It is a proactive scheme in which a set of special mobile nodes, called message ferries (or just ferries), move around the deployment area and are responsible for carrying data between nodes. Having ferry nodes with fixed moving paths, each ordinary node can save information about the ferry's mobility patterns and adapt its future trajectory to encounter the ferry and have messages sent or received. The main idea behind the Message Ferrying approach is to introduce non-randomness in the movement of nodes and exploit such non-randomness to help deliver data.

Using Multiple ferries [9] increase the system throughput (reducing message delay) and offers robustness to ferry failures. However, the route design is much more complicated than the single ferry case, considering the possibility of interaction between ferries. For the simple case of a single ferry, the authors adopted solutions for the well-studied traveling salesman problem (TSP). In the multiple ferries case, algorithms to assign nodes to specific ferries, synchronize among ferries, and assign ferries to specific routes are discussed. Four approaches were proposed, which are Single Route Algorithm (SIRA), Multi Route Algorithm (MURA), Node Relaying Algorithm (NRA) and Ferry Relaying Algorithm (FRA). Figure 2 illustrates these approaches.

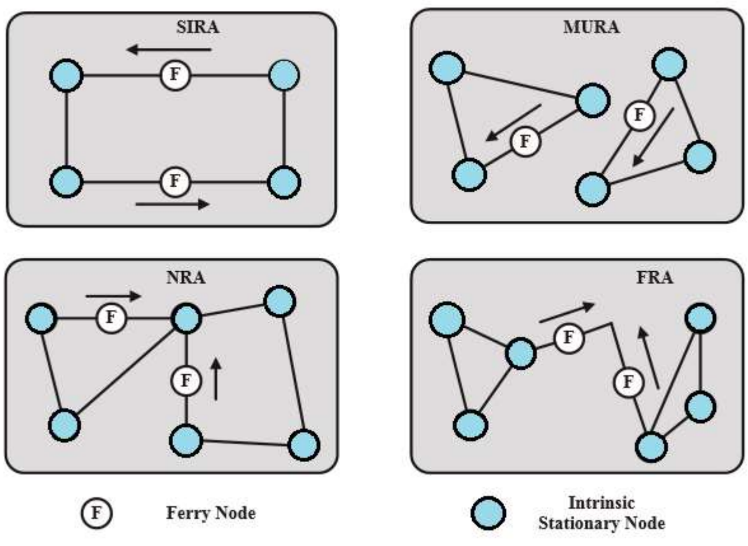


Figure 2 - illustration of the algorithms presented in [9].

In SIRA, all ferries follow the same route, as opposed to MURA. NRA utilizes nodes to relay messages between ferries. In contrast, FRA minimizes the waiting delay through direct interaction between ferries, unlike NRA that uses stationary nodes as the relay to minimizes the carrying delay in each ferry. The results obtained in the evaluation of the performance of route assignment algorithms, especially on the effect of the number of ferries on the average message delay, showed that when the traffic load is low, the improvement in delay due to the increased number of ferries is modest. This is because the delay is dominated by the distance between nodes. However, when the traffic load is high, an increase in the number of ferries can significantly reduce the delay.

Z. Zhang et al. [10] concentrated their work on the multi-ferry route design, especially how nodes are allocated to multiple ferries and how multiple ferry routes are connected. Figure 3 illustrate two different approaches of multiple ferries route design.

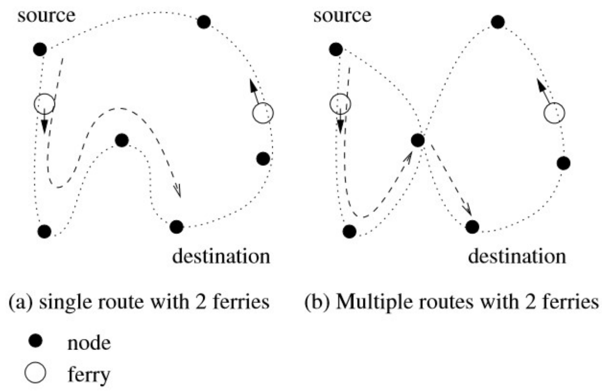


Figure 3 - Multiple ferries examples of route design: (a) SRT and (b) MRT [10].

Their first presented solution is the simpler case of having multiple ferries using the same route passing through all the nodes and optimized by a solution of the TSP, called single route approach (SRT), Figure 3 (a). For multiple ferries (MRT), Figure 3 (b), Zhang's first concern is the allocation of nodes for each ferry, having nodes partitioned into equal parts until each node partition has been served by one ferry only. This node allocation method naturally leads to a tree topology of ferry routes if a node is selected as a relay shared by both parts whenever a partition is performed. The advantage of the tree topology is that it uses the minimum number of relay nodes while keeping the ferry routes graph connected. This design is called MRT-Tree. By using more relay nodes, the connectivity of the ferry routes' graph can increase, which in turn will decrease the graph diameter and thus the average number of ferry hops each message takes. Another MRT design can be made if partitions adjacent to each other always have ferry routes connected. This design is called MRT-Grid, and has an average message delivery delay lower than MRT-Tree. Figure 4 shows the two MRT designs discussed above.

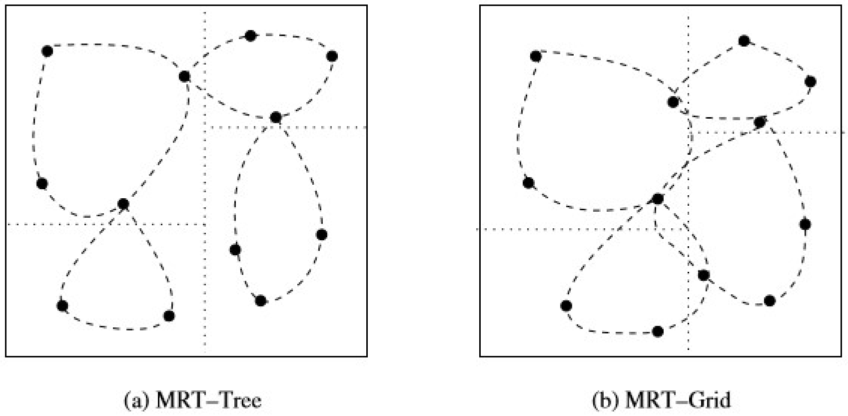


Figure 4 - MRT design: tree and grid approach [10].

In Xue et al. [11], the authors suggested a dynamic adaption of ordinary nodes into ferry nodes when there exists high network traffic. The idea of this adaptive ferry algorithm is having an ordinary node selected whenever the main ferry node issues a BUSY message, by having the energy level of each ordinary node compared, and the one with the highest energy level is promoted from ordinary node to selector-ferry. When the selector-ferry finds its own energy lower than a threshold value, it will switch to an ordinary node. The ferry will determine whether to elect another selector-ferry node according to the network traffic situation, and the process continues until the end of the simulation.

Homing Pigeon Based Routing

Homing Pigeon Based Routing [12] explores the idea of having extra auxiliary nodes (messengers) and takes a different approach, assuming that each node in the network owns a dedicated messenger, called pigeon. Guo et al. developed a homing-pigeon-based DTN (HoP-DTN), where messages are generated at the source (home) and delivered to corresponding destinations through the movement of pigeons. Figure 5 illustrates the HoP-DTN scheme.

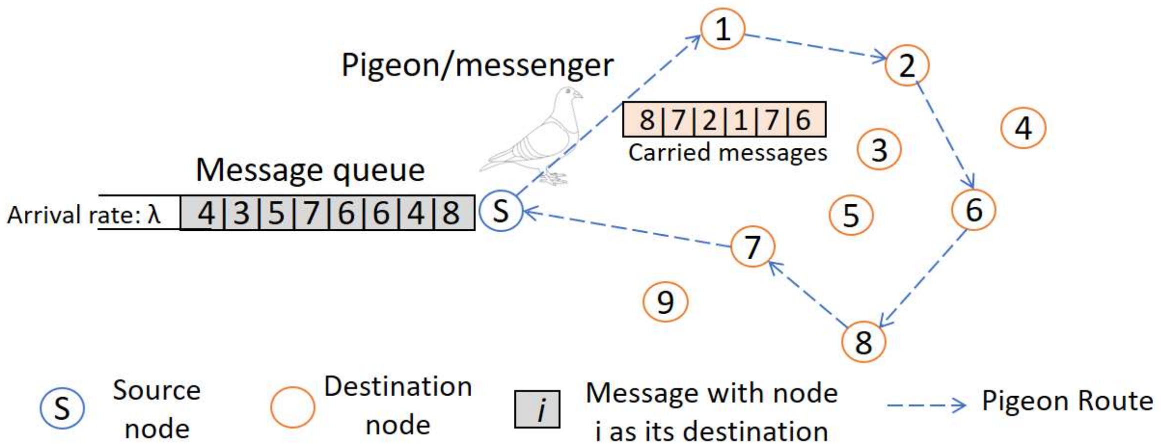


Figure 5 - Homing Pigeon based DTN scheme [12].

This dedicated messenger scheme, instead of shared messengers like MF, has the messengers only visiting the destination nodes of carried messages that adapt to the dynamic demand of traffic. A requirement of HoP-DTN is that pigeons know the location of other nodes. Visiting only the necessary nodes achieves faster message delivery and higher delivery ratio. The authors conclude that the HoP-DTN scheme achieves better performance because the special-purpose messengers (pigeons) proactively move to the destinations to deliver the messages, whereas in opportunistic contact-based (or reactive) schemes, the relay nodes passively wait for the connection opportunity to the next hop occurring at each hop of forwarding, which introduces a larger delay. The results also demonstrate that this HoP-DTN model outperforms the MF schemes.

In [13], the same authors evaluate the extension of their previous model using multiple pigeons per node, for message delivery. They also compare two different scheduling strategies to schedule multiple pigeons: on-demand and storage-based. In on-demand strategy, since pigeons start out immediately as soon as there are messages queueing at the source, they would travel more times for delivering the same number of messages and they return to the source faster. In storage-based scheduling, pigeons always carry a fixed number of messages per trip, having a distance travelling trip always higher than the on-demand strategy, but delivers the same number of messages with fewer trips.

2.2 Unmanned Aerial Vehicles – UAV

With their maneuverability and increasing affordability, unmanned aerial vehicles (UAVs) have many potential applications in wireless communication systems, like providing cost effective wireless connectivity for devices without infrastructure coverage due to e.g., severe shadowing by urban or mountainous terrain, or damage to the communication infrastructure caused by natural disasters [14].

Compared to terrestrial communications or those based on high-altitude platforms, on-demand wireless systems with low-altitude UAVs are in general faster to deploy, more flexibly re-configured, and are likely to have better communication channels due to the presence of short-range line-of-sight (LoS) links.

Many advances have been taking place in unmanned aerial vehicle technology lately. This is leading towards the design and development of UAVs with diverse sizes that possess increased on-board processing, memory, storage, and communication capabilities. Consequently, UAVs are increasingly being used in a vast amount of commercial, military, civilian, agricultural, and environmental applications. They are also used in many types of disasters including meteorological, geological, ecological ones. For example, they were used in the aftermath of hurricane Katrina (2005) [30], L'Aquila earthquake (2009), Typhoon Marakot (2009), Tohoku Earthquake (2011) [30], Haiti earthquake (2010) [32], Nepal Earthquake (2015) [31], Peru Floods (2017) [33] and Puerto Rico Hurricane [34]. To take full advantages of their services, these UAVs must be able to communicate efficiently with each other and with existing networking infrastructures.

2.2.1 The Evolution of UAVs

The earliest recorded use of an unmanned aerial vehicle was a balloon shaped UAV used to carry explosives into enemy's territory and intelligently triggered to accomplish the assigned mission from a remote end. This kind of strategy was used in warfighting in World War I. To mitigate limitations and enhance the level of intelligence of the vehicle, efforts were made in the post-world war I era to develop aerial torpedoes, which ultimately led to the modern form of cruise missiles used in World War II [15]. Since then, UAVs evolved from a military purpose to civilian use. Nowadays, the range of application of UAVs, or commonly known drones, goes from surveillance and air strikes, to pizza delivery and farming.

It is useful to distinguish drone types, since they go from small quadcopters to plane size UAV. Drones can be multi rotor (e.g. quadcopters), fixed wing, single rotor (e.g. helicopter) and fixed wing hybrid. Figure 6 illustrates different drones with man height for scale.

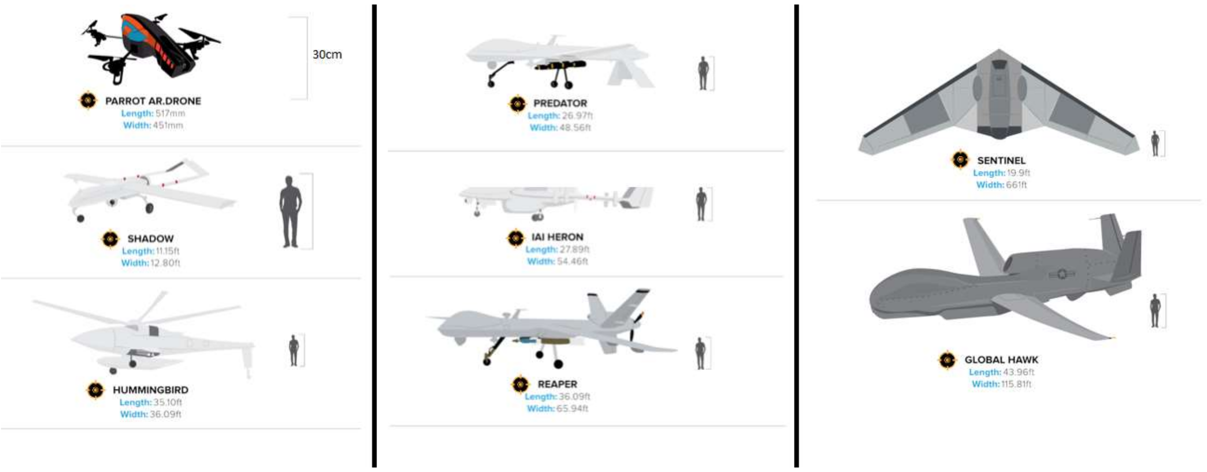


Figure 6 - Not all drones are equal [29].

UAVs are becoming increasingly popular in the commercial and private market, having as an example Amazon.com, the largest online retailer, which is using drones to deliver mail autonomously.

2.2.2 UAV Networks and applications

UAVs can be equipped with various antenna technologies, GPS, video cameras, infra-red cameras among others and can be deployed to incidents fast while flying at low altitudes. This motivates the use of UAVs as communication relays in scenarios such as cost-effective data harvesting in a widely-distributed sensor network (e.g., flood monitoring in jungles), as well as in the event of emergencies (e.g., telecom failure, disaster response and military operations), where parts of the communication infrastructure could be damaged, which is difficult to recover within a short period of time, and hence no direct communication between nodes is possible.

In those scenarios, UAVs can be used to extend the communication range of remaining systems employed by ground-based nodes to communicate with those that are not in direct contact, acting as relay nodes, connecting geographically separated Mobile Ad Hoc Network (MANET) clusters. Nodes belonging to different disconnected clusters can communicate with each other through an UAV, which can be placed in a strategic position between the two clusters. Figure 7 exemplify three cases of UAV-aided wireless communication.

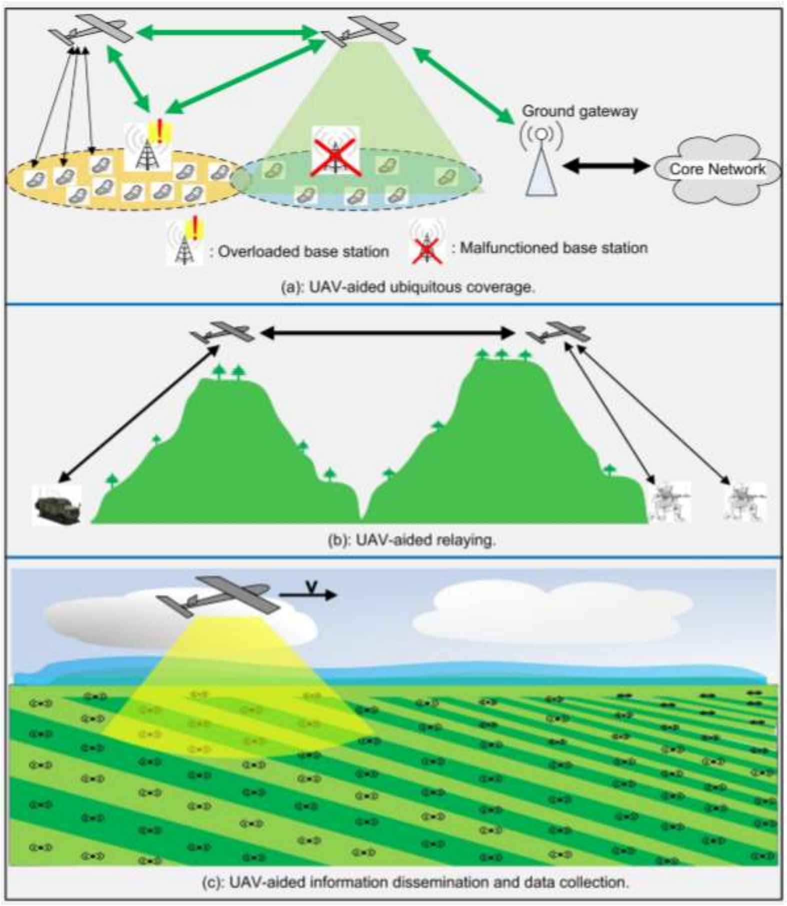


Figure 7 - Three Typical use cases of UAV-aided wireless communications [14].

Early uses of UAVs were characterized by employing a single large UAV as an aerial node, and one or more ground nodes, to perform a task. Nowadays, most public and civil applications can be carried out more efficiently with multi UAV systems, where the UAVs are smaller and less expensive and work in a coordinated manner. The degree of mobility of UAVs depends on the applications, which means that network configurations present significant variations. For example, UAVs can be used to form a static hovering bridge of access points to provide network access for wireless devices, behaving like an infrastructure-based network [19]. As opposed to this, in applications like agriculture or forest surveillance, UAVs could be highly mobile and communicate, cooperate and establish the network dynamically in an ad hoc manner. In both cases, malfunctions and battery drainage can occur, causing UAVs to go periodically out of service. New UAVs must be launched to take their places. Therefore, it is necessary that the links in the network automatically reconfigure themselves in order to achieve an optimized state.

All wireless mobile networks are prone to link disruptions with the UAV networks being no exception. The extent of disruption depends on how mobile the UAVs are, the power transmitted and inter-UAV distances. In the applications where UAVs provide communication coverage to an area, the UAVs are hovering and, consequently, the probability of disruptions is expected to be low. On the other hand, in applications requiring fast UAV mobility, there is a higher likelihood of disruptions. Delays in transmitting data could be due to poor link quality or having one or more UAV nodes storing the data because of end-to-end path not being available.

2.2.3 Routing with UAVs

As explained in the previous section, UAV networks can have many different configurations depending on node mobility, network partitioning, intermittent links, limited resources and varying quality of service (QoS) requirements, making routing in UAV networks a challenging research task. The choice of the appropriate networking protocol depends on the nature of the application which determines the following specifications [22]:

- Number of UAVs in the network: Some routing protocols provide good performance for small networks but do not perform well in large networks where other protocols can be more effective.
- On-board processing capabilities: This includes the microprocessor that may have the ability to perform complex calculations.
- On-board memory and storage capacity: Some of the routing protocols require large storage space, especially when the number of nodes is large. This is particularly true in UAV-based networks with small size UAV.
- Energy and power capability of the UAVs: some energy-aware routing protocols are more appropriate, especially for small size UAVs.
- GPS capability of the UAV: This allows designers to use geographic routing protocols, which provide good performance with large UAVs and high mobility networks. Many routing protocols have been developed for Vehicular Ad hoc Networks (VANETs), with similar characteristics.
- Handoff and roaming: As UAVs move in and out of communication range of various gateways, appropriate and timely handoff and roaming strategies must be used to ensure seamless switching between cells.
- Throughput: This is also an important parameter that needs to be considered in light of the data traffic that is required to be supported by the UAV-based network. High throughput and data rates are essential for good quality imagery and video, while lower data rates can be tolerated when the exchanged data is limited. In this case, while high data rates are not essential, low delay becomes critical for such real-time data traffic.
- Degree of UAV mobility: Some routing protocols provide good performance but suffer from a prohibitively large control message overhead when the nodes are highly mobile. This is true as discovered routes constantly break and new routes must be discovered. Other protocols, such as geographic-based ones are appropriate for such an environment since they have reduced overhead.

Some papers have considered the use of existing routing protocols for possible use in aerial networks. Although conventional ad hoc routing protocols are designed for mobile nodes, they are not necessarily suitable for aerial nodes due to varying requirements of dynamicity and link interruptions. Due to apparent similarity of UAV networks with Mobile Ad hoc Networks (MANET) and Vehicular Ad hoc Networks (VANETs), researchers have studied protocols used in those environments for possible application in aerial networks. However, multi-UAV networks may have several different requirements to be taken into account. Gupta et al. [21], discuss several networking protocols with a view to see their usefulness for UAV Networks. For the scope of this master thesis, we are only interested in those routing protocols that address DTN scenarios, which will be discussed in the next section.

2.2.4 UAVs in Delay Tolerant Networks

When disaster strikes and normal communications breaks down, lack of information flow could cause delay in rescue and recovery operations. In such situations, a UAV network with delay tolerant features may become an effective communication method [16].

Routing in networks with sporadic node contacts, consist of a sequence of time-dependent communication opportunities during which messages are transferred from the source towards the destination. In [21] Gupta et al. evaluate the suitability of numerous types of protocols for UAV networks, reaching the following conclusions.

UAV networks with deterministic protocols

Deterministic routing assumes that the future movements and links are completely known. In the context of UAV networks, this would be possible in applications where UAVs fly in controlled formations or in applications where they have to hover over an area.

UAV networks with Stochastic protocols

In UAV networks with intermittent links and opportunistic contacts, routing is challenging since the time when the nodes will come in contact and for how long are not known. When the contact does happen, it needs to be determined if the peer in contact is likely to take the message closer to the destination. The decision to hand over the message to the node in contact depends on available buffer space in the two nodes, relative priority to forward the message compared to previous messages that the node already holds, and, obviously, the probability of the contact taking the message closer to the destination. It should be kept in mind that the main goal of routing is to maximize the probability of delivery at the destination, while minimizing the end-to-end delay. Next are presented some DTN stochastic routing protocols and considerations for their application in UAV networks [21].

Epidemic based:	Requires large buffer space per node, bandwidth and power. In UAV networks, message delivery time depends on the buffer size. Delay is lower but with high-energy expenses.
Estimation/Probability/Statistical:	Random methods work well for small networks but for large networks estimation results in large overheads. Changes in topology affect convergence time. Delays are moderate at moderate energy consumption.
Node Movement Based:	UAVs can be made to follow a given trajectory that will connect source and destination in partitioned networks.
Message ferrying:	Location information is maintained. Large storage space required in ferries. Delays are high but energy expense is low. Can work with heterogeneous nodes.

Albuquerque et al. [17] investigate and evaluate the performance of most popular DTN routing protocols through simulations based on real flight traces of an UAV over two scenarios of natural disaster caused by heavy rains and landslides. In emergency scenarios, there are always rescue teams conducting search operations or transporting people and supplies. Thus, it is imperative to maintain communication contacts between teams and the command center. However, depending on factors such as topography or vegetation, groups may become isolated and unable to communicate. This is where UAVs can act, since the best way to reach the major points of interest affected by severe rainfalls is clearly through air. The authors collected data flying an UAV over a round trip route based on critical areas of each disaster and the command center placement on each occasion. The evaluated performance metrics were the ratio of delivered messages (probability of a message to reach the destination), the average delivery delay (average time a message takes to reach the destination) and the overhead (average number of copies generated for a message). Maxprop and Epidemic protocols scored lower latencies, which would make them favorites for faster message delivery. However, they have the cost of a higher overhead, which can pose a greater energy consumption for nodes. In emergency scenarios, where messages must be delivered urgently, an increase in power consumption to keep a shorter delivery delay might be acceptable.

Kwon et al. [18] provided a scheduling analysis framework to show if an UAV can cover a given set of distributed nodes. Each sensor node may have a different priority, required frequency of visits and communication range. Further, they discuss dynamic positioning of UAVs for maximizing communication efficiency by means of packet delivery rate and signal to noise ratio between the UAVs.

More recently, Reynaud et al. [20] presented a distributed controlled mobility strategy relying on virtual forces, which enables a flock of network nodes to move cooperatively and form multi-hop communication links where needed. They evaluate its ability to be jointly used with a dual packet-forwarding and epidemic routing protocol.

Summing up, the research in UAVs and DTNs is still widely open, with room for improvement. This master thesis intends to evaluate the use of UAVs as messengers in a delay tolerant network scenario, taking advantage of routing algorithms like *Message Ferrying* and *Homing-Pigeon-based routing* to do so. In any of these algorithms, including our, there is route optimization which is done by solving the Travelling Salesman Problem multiple times.

2.3 Traveling Salesman Problem

The TSP consists of a hypothetical salesman looking for the best path, the one minimizing the total length of the trip, to visit a set of cities, starting from a certain one, the hometown, stopping only once at each city, and ending up at the initial starting location.

The problem was first formulated in 1930 and is one of the most intensively studied problems in optimization. Even though the problem is computationally difficult, many heuristics and exact algorithms are known.

The TSP is defined as a permutation problem with the objective of finding the path of the shortest length (or the minimum cost). It can be modeled as an undirected weighted graph, such that cities are the graph's vertices, paths are the graph's edges, and a path's distance is the edge's length. Figure 8 illustrates an example.

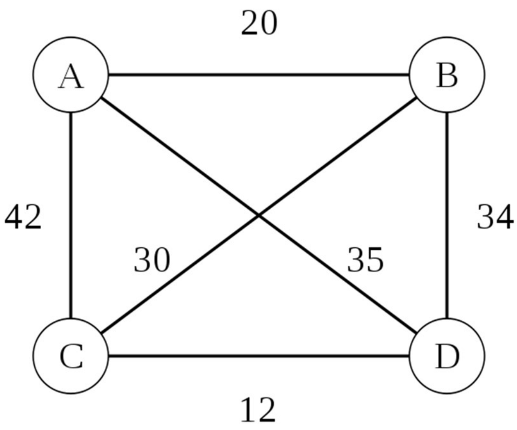


Figure 8 - example of an undirected weighted graph.

If *n* is the number of cities to be visited, then the number of possible routes is $(n-1)!$. Following this basic formulation, an exponential relationship exists between the number of cities and possible routes, for instance if there are 4 cities there are 6 possible routes, for 6 cities 120, for 10 cities 362880, and so on. As the number of cities increases, the amount of input data rises and the problem increases in complexity, thus the computational time needed makes brute force analysis impractical for all but a smaller number of cities.

We can immediately make a parallel between this and a DTN scenario, where the UAV is the salesman and the nodes to be visited are the cities. Every time a UAV has to visit several nodes it needs to know which is the best route that optimizes its delivery. Thus, the UAV has to make use of algorithms to compute this route. For the computation to be quicker, it can use heuristics to approximate the exact solution. Although they can sometimes not be the best solutions, they approximate the optimal answer, and their execution time is way more suitable than classical approaches.

Heuristic algorithms promise a feasible solution. They range from simple tour-construction methods like Nearest Neighbor, Clarke-Wright and Multiple Fragments to more complicated tour improving algorithms like Tabu Search and Lin-Kernighan. Finally, there is a group of fascinating algorithms which unfortunately tend to combine approximate solutions and large running-times. Here we find methods like Simulated Annealing, Genetic Algorithms, Ant Colony Algorithms and machine learning algorithms like Neural Networks. Next, some of these methods will be covered.

Simulated Annealing (SA)

Simulated annealing [27] is a well-known meta-heuristic search method that has been used successfully in solving many combinatorial optimization problems. It is a hill climbing algorithm with the added ability to escape from local optima in the search space. However, although it yields excellent solutions, it is very slow compared to a simple hill climbing procedure. The term simulated annealing is adopted from the annealing of solids, where we try to minimize the energy of the system using slow cooling until the atoms reach a stable state. The slow cooling technique allows atoms of the metal to line themselves up and to form a regular crystalline structure that has high density and low energy.

Ant Colony Optimization (ACO)

Ant Colony Optimization is a technique that is inspired by the behavior of real ants. Its principles were established by Dorigoet al. in 1991 [25]. Real ants cooperate to find food resources by laying a trail of a chemical substance called ‘pheromone’ along the path from the nest to the food source. Depending on the amount of pheromone available on a path, new ants are encouraged, with a high probability, to follow the same path, resulting in even more pheromone being placed on this path. Shorter routes to food sources have higher amounts of pheromone. Thus, over time, most ants are directed to use the shortest path. This type of indirect communication is called ‘stigmergy’ [26], in which the concept of positive feedback is exploited to find the best possible path, based on the experience of previous ants.

Genetic Algorithms (GAs)

The idea of simulation of biological evolution and the natural selection of organisms dates back to the 1950’s, while the theoretical foundation of GAs were established by John Holland in 1975 [35], after which GAs became popular as an intelligent optimization technique that may be adopted for solving many difficult problems.

The theme of a GA is to simulate the processes of biological evolution, natural selection and survival of the fittest in living organisms. In nature, individuals compete for the resources of the environment, and they also compete in selecting mates for reproduction. Individuals who are better or fitter in terms of their genetic traits survive to breed and produce offspring. Their offspring carry their parents’ basic genetic material, which leads to their survival and breeding. Over many generations, this favorable genetic material propagates to an increasing number of individuals. The combination of good characteristics from different ancestors can sometimes produce ‘super fit’ offspring which out-perform their parents. In this way, populations evolve to become better suited to their environment.

Genetic algorithm is started with a set of solutions (denoted by chromosomes) called population. Solutions from the original population are taken and used to form a new population. This is motivated by a hope, that the new generation will be better than the old one in its characteristics. Solutions, which are designated to form new solutions (offspring), are selected according to their fitness attributes; the more suitable they are, higher is their probability to replicate. This action is repeated until certain conditions are satisfied.

Chapter 3

Analysis and Implementation of Existing Algorithms

3.1 Introduction

To develop a new model, it is always useful to fully understand the existing ones. This chapter presents in more detail some of the proactive schemes presented in chapter 2, and how they were implemented in MATLAB [28]. The algorithms are Single Route Algorithm – SIRA, Multiple Route algorithm – MRT-Grid and Homing Pigeon based algorithm – HoP DTN.

The situation in which these algorithms will be evaluated corresponds to a large area, with distributed node in a sparse network. There are multiple UAVs available to allow ground nodes to communicate among them, and there is no alternative means of communication. The goal is to optimize the delivery ratio and the minimum delay of the messages exchanged between those ground nodes, which means optimizing the paths and timing decisions of the UAVs.

3.2 SIRA – Single Route Algorithm

Although this algorithm was developed over a decade ago by Zhao, it is still relevant as a basic message ferrying scheme, being the first comparison model for every new scheme in this area. As discussed before, SIRA is the algorithm in which all ferries follow the same route, being equally spaced. In SIRA, there is no node assignment, which is the same as saying all nodes are assigned to all ferries that share the responsibility of transporting data between nodes. Figure 9 gives an example of having ferries going to a single ferry route.

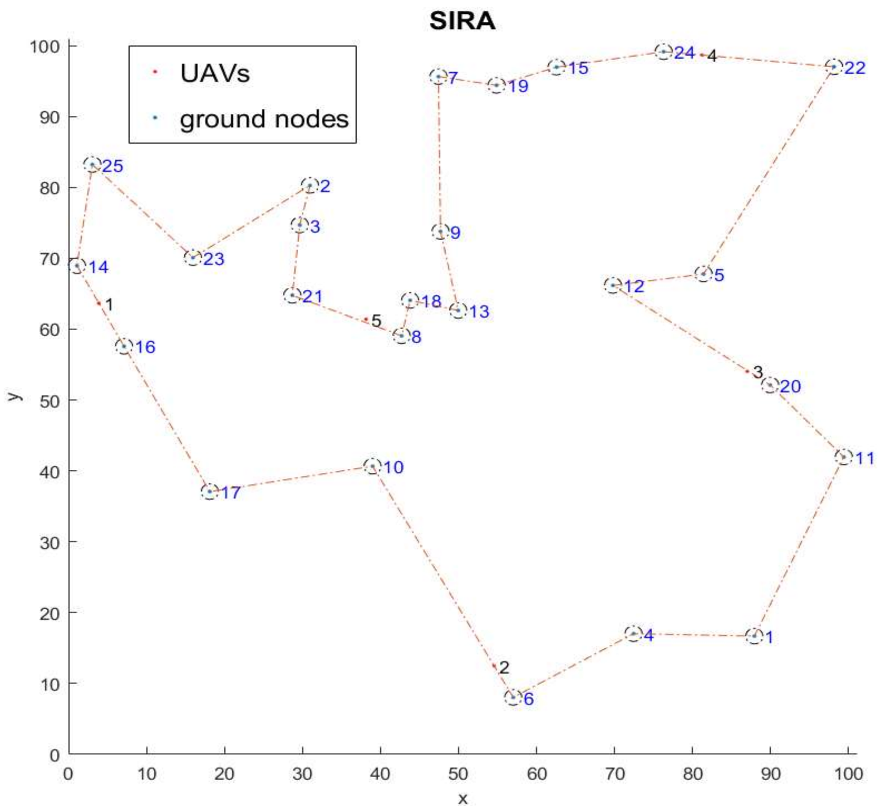


Figure 9 - SIRA's representation.

3.2.1 Route optimization

To compute the optimized route that minimizes the traveled distance, a genetic algorithm that solves the TSP problem was used [23]. First, the ground node positions are generated at random, the only restriction being ground nodes far enough so there is no possible interaction, creating a realistic DTN scenario. Then, the genetic algorithm function operates on those ground node positions, retrieving the best route possible for all ferries to follow. Algorithm 1 lists the pseudocode of the genetic algorithm.

Algorithm 1: Traveling Salesman Problem – Genetic Algorithm Pseudocode

1. Extract the configuration (ground nodes positions).
2. Compute the distance matrix.
3. Initialize the population: randomly sorts the ground nodes order in 100 possible ways.
4. **For** a certain number of iteration (in this case 10 000) **do**
   - 4.1. Evaluate each population member
   - 4.2. Find the best route – Retrieves the member (the order) that corresponds to the minimum distance traveled.
   - 4.3. Check if this best route is better than the best route obtained until this iteration.
   - 4.4. Save the best route.
   - 4.5. Mutation of population members occurs:
     - 4.5.1. Select 4 routes at a time.
     - 4.5.2. Calculate the distance traveled in those four routes and retrieve the best one.
     - 4.5.3. Mutate the best to get three new routes: gets 2 random numbers I and J

       Case 1: do nothing.

       Case 2: Flip – reverse the sequence order between I and J.

       Case 3: Swap – swaps the position of I and J.

       Case 4: Slide – slides all the sequence between I and J to the left.

The results of the TSP GA function contain a distance matrix, a record of the best solution history in terms of total distance, the best route obtained and the corresponding distance. Figure 10 presents those results using 25 ground nodes randomly placed in a 100x100 area.

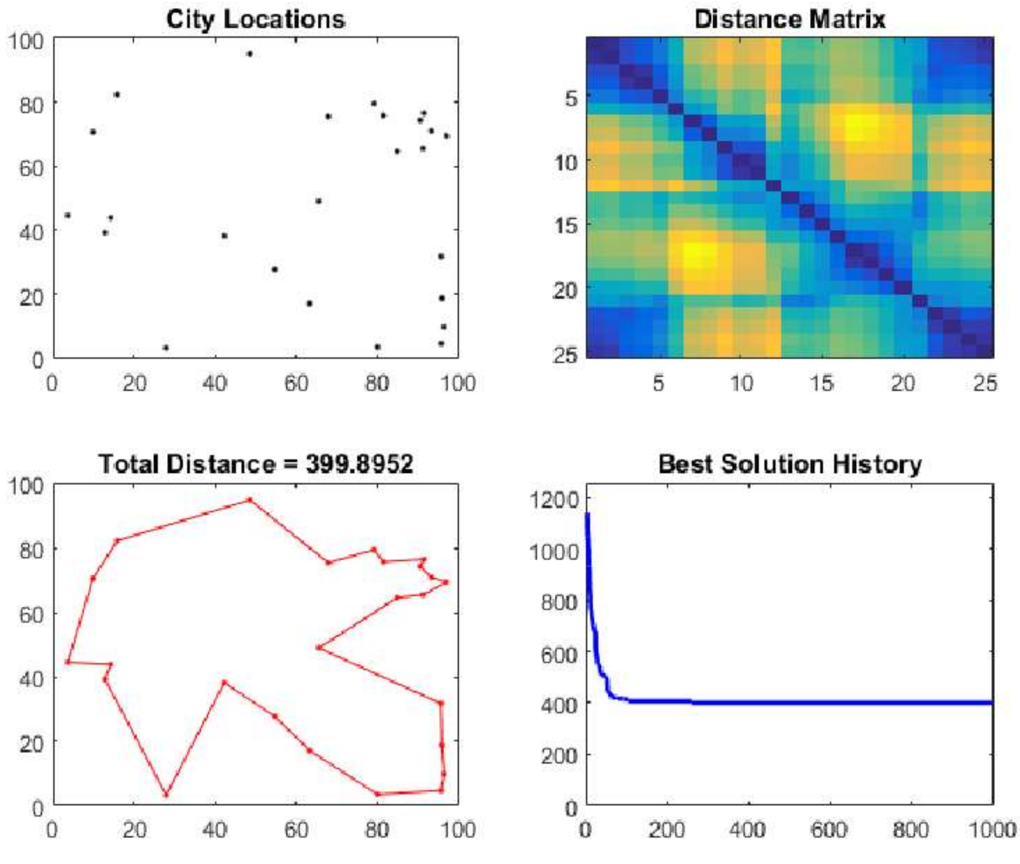


Figure 10 - TSP Genetic Algorithm Results.

The best route is the sequence of destinations (the order) that the UAV must take to maximize its resources. In the description of this SIRA algorithm, it was explained that the ferries should be equally spaced in the route.

Given the total route distance obtained with the TSP GA function, we can split it into the number of UAVs available, thus obtaining the distance by which they should be apart, the so-called *equidistance*. Then, a ferry is assigned to the first destination obtained in the best route. It doesn't really matter where to start – the first ferry could be assigned to any node. Using the distance matrix obtained before, we can find the last ground node whose distance to the first UAV, following the route, is less than the *equidistance*. Then, some trigonometry comes in handy. Figure 11 illustrates an example:

The ground nodes are represented by blue numbers, and the black 1 represents the first UAV.

With UAV-1 starting at ground node 22, see how we can determine the initial position of UAV-2:

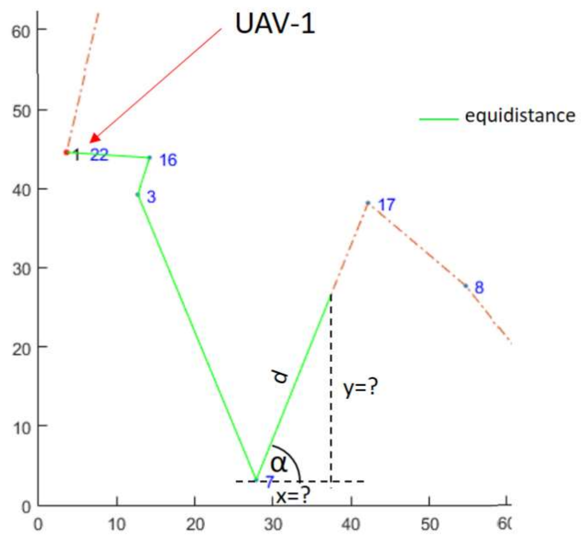


Figure 11 - trigonometry problem in hand.

As figure 11 shows, we have all we need to calculate the x and y coordinates of UAV-2.

$$d = equidistance - distance[22 \to 16 \to 3 \to 7] \quad (1)$$

$$\alpha = tan^{-1}\left(\frac{y_{17} - y_7}{x_{17} - x_7}\right) \quad (2)$$

$$UAV_2 = \begin{cases} x = x_7 + d\cos(\alpha) \\ y = y_7 + d\ sen\ (\alpha) \end{cases} \quad (3)$$

The same reasoning can be used to find the initial positions of all UAVs.

After having the route that the UAVs will follow and their initial positions, the simulation can finally start. After all UAVs are set to have the same constant speed, they start their movement and examine at each unit of time if they are near a ground node. When the UAV enters the transmission range, it first delivers the messages that have this ground node as destination and then collects as much messages as its buffer can handle at that time. After concluding this process, it continues in its route until reaching the next node, where it will repeat the process. This goes until the simulation time limit is reached.

3.2.2 Discussion

A point in favor of this algorithm, is the reduced computation power necessary in each UAV, since the route is only defined in the beginning. All the UAVs have to do is exchange messages and follow a pre-computed path.

The big disadvantage that comes to mind looking at this algorithm, is the constraint of following a single route. Several times, the destination node could be very close but, since the UAV has to strictly follow the route, the delay is much bigger than hypothetically going straight to it, and, in case of having deadlines involved, the messages might not even get there. For example, imagine a message having as source ground node 10, and destination ground node 8 in figure 9. This is where the idea of clustering emerges from, trying to deal with nodes in smaller clusters instead of one singular vast cluster. The next section will cover the multiple route algorithm.

3.3 Multiple Route Algorithm

In [10], the authors compared several multiple route algorithms, namely NRA (Node Relaying Algorithm), MRT-Tree and MRT-Grid. In these algorithms, the idea is always the same: data is relayed between ferries via nodes. That is, ferries forward data to a node and other ferries receive data from this node. The difference between the mentioned algorithms lies on the clustering.

In [9], Zhao clustered the nodes for NRA dividing the area into a grid of $r \times c$ cells. Ferries are assigned to cells, carry data for nodes within the cell and extend their route adding connectivity between ferry routes. This assigns nodes to ferries based on their geographic location. Figure 12 illustrates this kind of clustering.

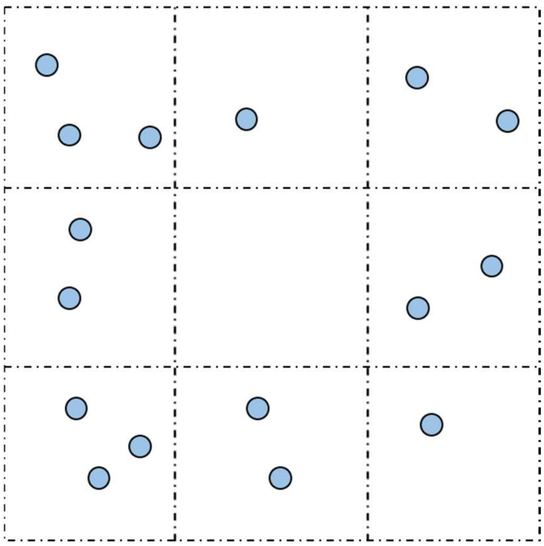


Figure 12 - NRA geographic clustering.

In case of irregular node distribution, there might be empty cells deprived of any node. This makes node relaying even more difficult, since there might not exist neighbors in the adjacent cell.

In [10], Zhang followed a different approach when trying to partition nodes into balanced parts, using the 2D-Tree partition algorithm. This algorithm performs by alternatively cutting horizontally and vertically, along the vertical and horizontal median, considering the node positions coordinates. Figure 13 illustrates the medians: vertical median corresponds to the median of the y coordinate; horizontal median corresponds to the median of the x coordinate.

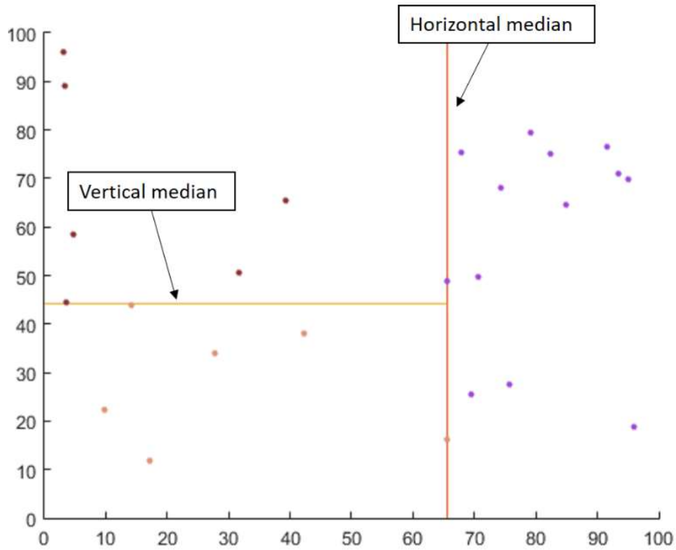


Figure 13 - Illustration of a horizontal and vertical median.

By cutting sequentially through the medians, this partition algorithm ensures there is no empty cells like in geographic clustering used in NRA. Figure 14 shows this kind of clustering for a similar distribution of nodes, with 5 partitions.

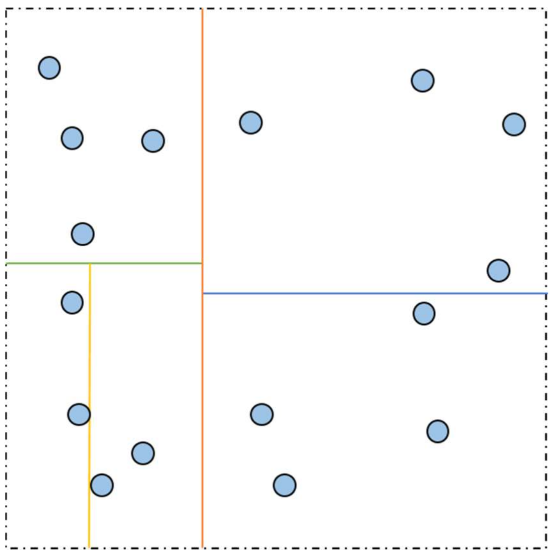


Figure 14 - MRT clustering.

Using this clustering, Zhang tested two possible designs: MRT-Tree and MRT-Grid. The former being the one using the minimum relay nodes possible, and the latter the case of connecting every adjacent partition using more relay nodes (Figure 4).

Since MRT-Grid was the one with better results in [10], we chose to implement only this one to further use as a comparison for our model. This Multiple Route algorithm consists of having ground nodes acting as relays to minimize the carrying delay in each ferry. The ferries forward messages with destination outside their cluster by delivering those messages to designated relay nodes, i.e. nodes that share ferry routes.

3.3.1 MRT-Grid Clustering

As explained before, MRT-Grid uses a 2D-Tree partition algorithm that acts by alternatively cutting horizontally and vertically in balanced parts, by always cutting through the horizontal and vertical medians. To implement this technique, a 2D-Tree algorithm was developed in MATLAB, using a recursive function that works for any number of partitions and not only a power of 2. Algorithm 2 lists the corresponding pseudocode:

Algorithm 2: 2D-Tree partition algorithm pseudocode

1. Receives configuration inputs (ground node positions, deployed area limits, number of UAVs available).
2. Computes the horizontal median of all nodes and does a vertical cut.
3. Each part obtained in the previous step is further sequentially divided into two sub-parts by a horizontal cut, which goes through the vertical median of all nodes in the corresponding parts.
4. Each part obtained in the previous step is sequentially divided into two sub-parts by a vertical cut, which goes through the horizontal median of all nodes in the corresponding parts.
5. The process stops whenever the number of desired partitions is reached.

Figure 15 illustrates this algorithm for 3 to 8 partitions.

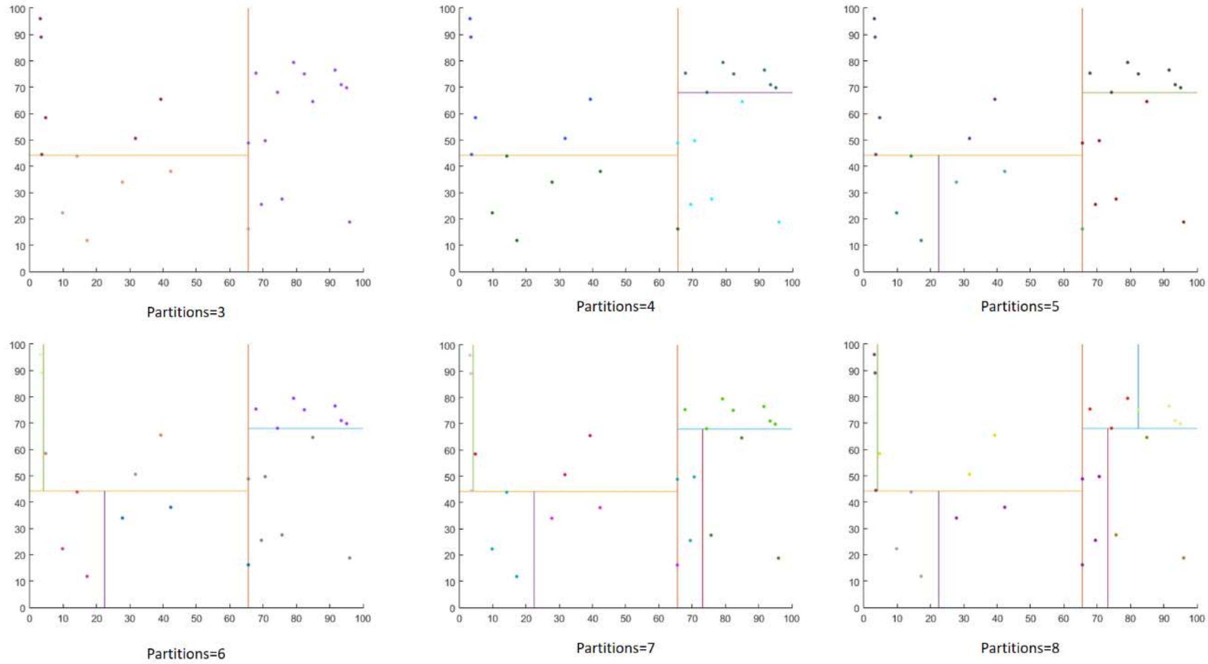


Figure 15 - 2D-Tree partition algorithm results for 3 to 8 partitions.

3.3.2 Ferry Routes Topology

To make any nodes reachable from any other nodes, the ferry routes must be connected in some way. Two ferry routes are connected if they share one or more nodes. To choose the nodes that will act as relays, the first step is finding the closest pair of nodes between two adjacent partitions, Figure 16. Then, the TSP genetic algorithm presented in section 3.2.1 evaluates the two possible extensions of ferry routes, and choses the one with minimum round-trip distance. Figure 17 shows the resulting relay nodes and routes.

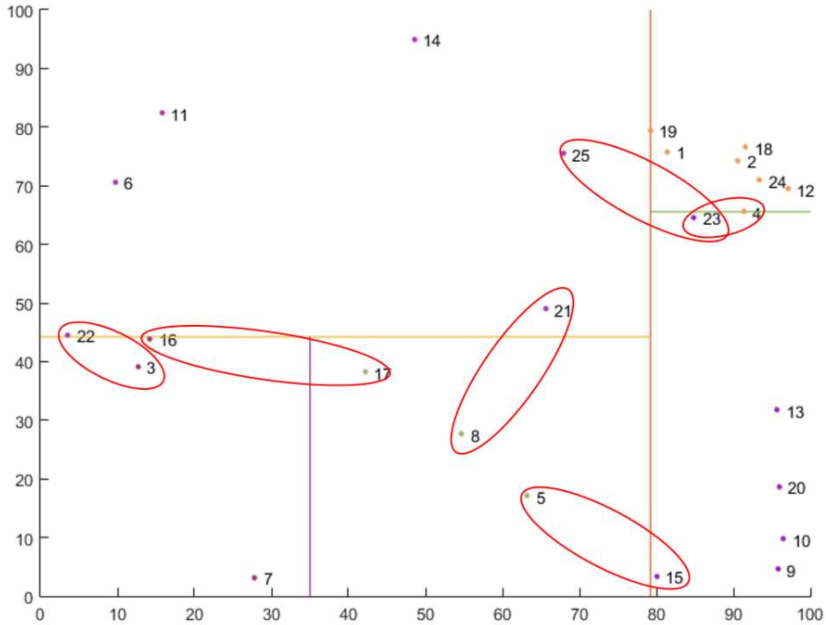


Figure 16 - Closest pair of nodes between partitions.

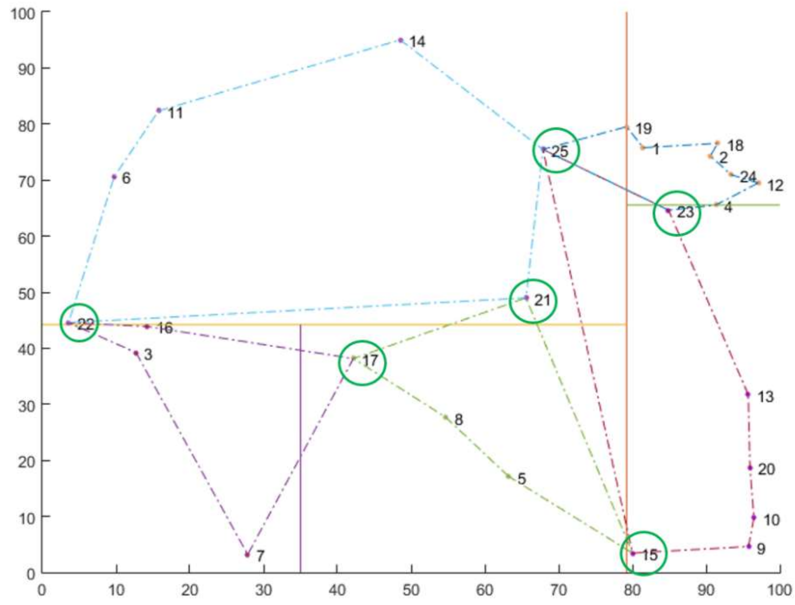


Figure 17 - Relay nodes and ferry routes obtained.

3.3.3 Routing in MRT-Grid

Zhang et al. [10] do not specify the routing used in this scheme, explicitly assuming that ferries have routing knowledge beforehand. In order to implement a decent algorithm, we decided to use directed graphs and shortest paths, using a source routing header with the full hop-by-hop route appended to each message.

After having the ferry routes computed, using MATLAB *digraph* function, it is possible to compute a directed graph representing available routes from any source to any target. Figure 18 illustrates the case of having 5 clusters.

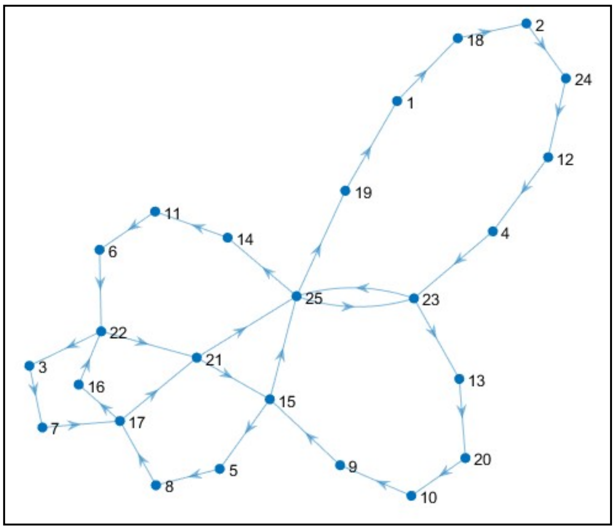


Figure 18 - graph with 5 clusters.

Using *shortestpath* function, stating the message source and target, it will return the sequence of hops that the message must sequentially take to arrive at the destination in the shortest way.

After having this pre-setting done, the simulation can finally start. Each UAV will start its journey at one of the ground nodes in its assigned cluster. Having all the same constant speed, each UAV will follow its own ferry route. Whenever a UAV is near a ground node, it firstly delivers the messages intended for that destination (if it exists) and then proceeds to collect messages from it. In case of the ground node being a designated relay node, the UAV will transfer all the messages that have the next hop out of its own cluster, and collect all those who have been left there with next hop belonging to its cluster.

3.3.4 Discussion

The bright side of this algorithm is the fact of using smaller clusters that allow UAVs to travel smaller distances, minimizing the carrying delay, making message transmission to nodes in the same cluster easier.

However, when a network becomes too partitioned, the delay of relaying messages from clusters that are far apart increases, since they have to be relayed several times.

3.4 Homing-Pigeon-Based DTN Algorithm

HoP DTN algorithm assumes that every node in the network owns a dedicated messenger, called pigeon due to the similarity to pigeon post used since over 2000 years ago. In each node, the pigeon waits till the buffer reaches full capacity, and then, instead of following a pre-computed route, it solves the TSP consisting of the destinations of the messages present in its buffer. At each destination, the pigeon unloads the corresponding messages to that destination and then immediately proceeds to the next one. When there's no more messages to deliver, the pigeon returns to its home-ground to collect a new batch of messages. Figure 19 shows each ground having one UAV for its personal use.

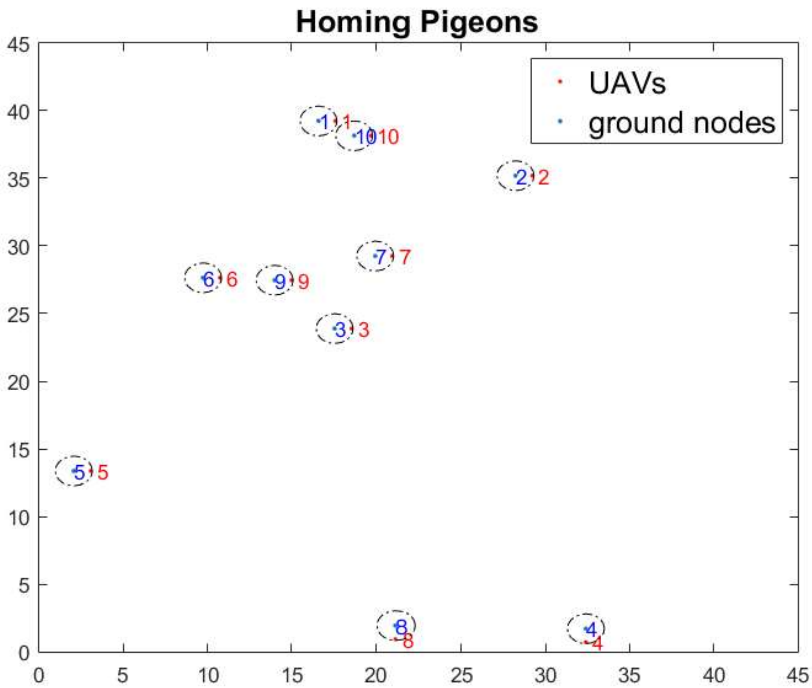


Figure 19 - HoP DTN algorithm illustration.

Once there is a certain number of messages to deliver, the pigeon computes the best route to go through all the message destinations, using the TSP GA presented earlier in 3.2.1. At each ground node, the pigeon delivers the corresponding messages, and immediately goes to the next ground node in the list. Once it reaches the final destination, the UAV goes back to its home ground node and waits till there are enough messages to start again.

3.4.1 Discussion

The major difference between this algorithm and the previous approaches is that each ground node owns a personal messenger, adaptable to its own demands, not having to visit all nodes along a predetermined route. Using personal messengers, there is no need for multi-hop relaying along multiple messenger, reducing message delivery delay.

A negative side of this algorithm is the capacity constraint imposed to start the delivery: each time a pigeon departs from the source, it must carry a fixed number of messages. Since it is possible that there are less than the amount of messages necessary in the source buffer, the pigeon is forced to wait until it reaches that number, which may compromise the delivery of those messages that were ready, in case of having deadlines to comply. Another detected flaw is that the pigeon at each destination only delivers the messages and does not use the now available space in the buffer to carry messages back to its source, which could improve the resource usage.

Chapter 4

Proposed algorithm

4.1 Introduction

This chapter will present the developed algorithm that will attempt to improve those presented earlier. After discussing all the advantages and shortcomings of each of them, we are now in position to present a scheme that uses those results to improve network performance. The model proposed here derives from both the MRT-Grid and the HoP DTN. In the new scheme, some new configurations to help achieve better results were added, like a TSP genetic algorithm that takes deadlines into account, and smart UAVs that will not load messages unable to reach their destination in time. The proposed approach is flexible, meaning it can act like a pigeon or ferry depending on the ratio of UAVs available and the ground nodes in the deployed area.

4.2 Deadline Triggered Pigeon with TSP Deadline

First of all, a k-means algorithm [24] for clustering the ground nodes and assigning UAVs was used, creating as many clusters as there are UAVs available. The reason to choose this kind of clustering algorithm lies on the need to group nodes that are near other nodes, in order to reduce the distance traveled by each UAV.

The UAVs act as ferries if their cluster has more than one ground node. If a cluster has more than three ground nodes, the TSP genetic algorithm presented in section 3.2.1 kicks in to obtain the best route – i.e. the one that represents less traveled distance for the UAV to act as a ferry.

Once the pre-setting is done, the simulation can finally start. Figure 20 presents the decision diagram by which each UAV goes through, when it reaches a ground node.

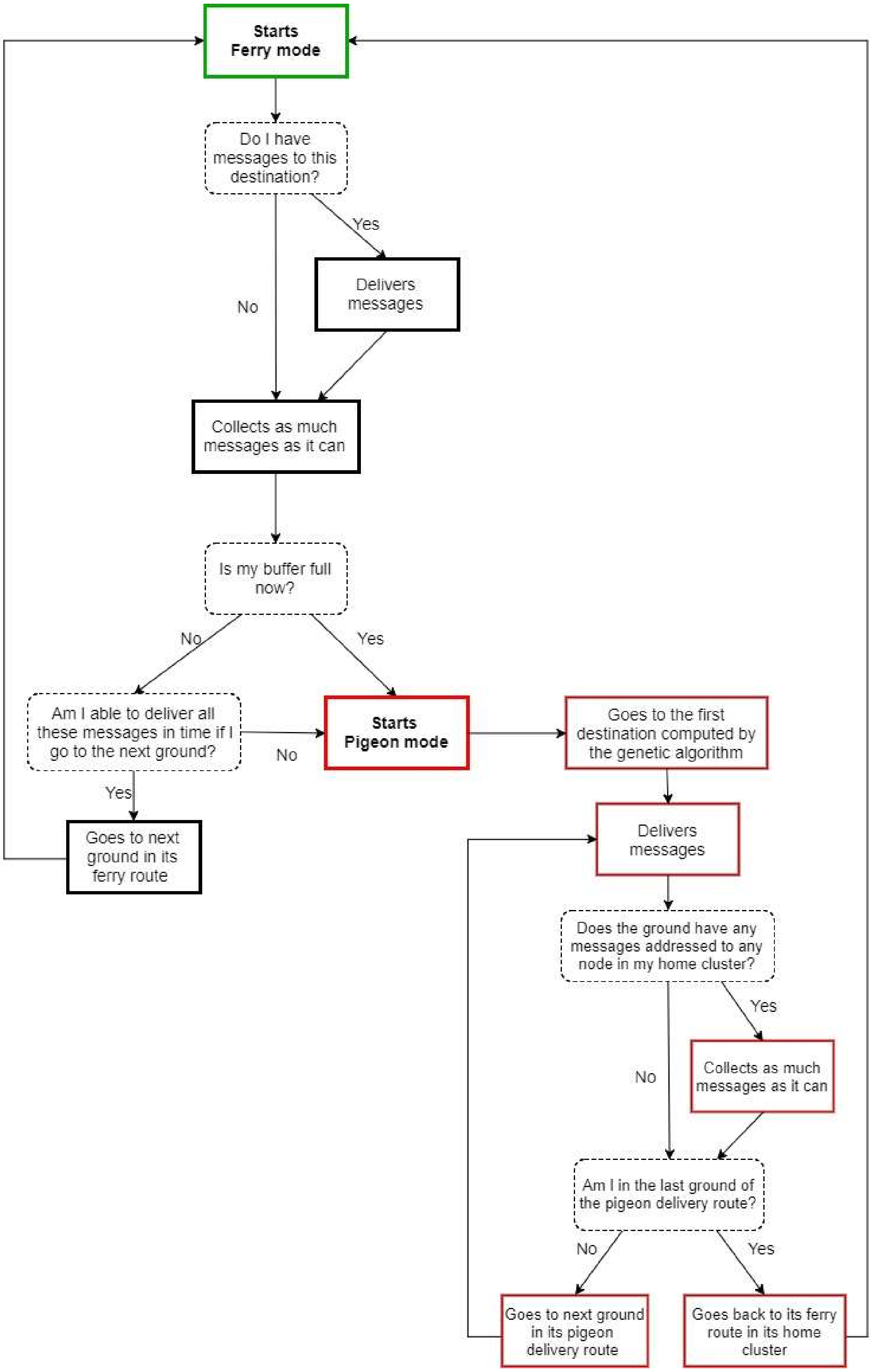


Figure 20 – UAV's decision diagram when it reaches a ground node.

Ferry mode is the case when the UAV is going through ground nodes in its own cluster following a ferry route. When it enters pigeon mode, it acts as described in HoP DTN algorithm, with the improvement of collecting messages in each destination it goes through, addressed to nodes in its own cluster.

As the name of the algorithm suggests, instead of capacity triggering as in the original HoP DTN algorithm, we use deadline triggering to convert the ferry into a pigeon. Meaning, the algorithm takes into account the deadlines of all messages in the buffer to compute the best route to deliver them, and checks if it can still deliver all of them if it goes to the next ground node on its ferry route, case negative the UAV computes the route starting from its current position, and transforms into a pigeon to deliver the messages through the new computed route.

To achieve this triggering, we had to modify the genetic algorithm used to solve the TSP. The best route is no longer the one that achieves minimum distance but the one which maximizes the number of timely delivered messages, in the minimum travelling distance.

From the code in [23], the fitness function was modified to accomplish what was desired. Algorithm 4 lists the pseudocode to this new TSP genetic algorithm.

Algorithm 4: TSP Genetic Algorithm with Deadlines Pseudocode

---

1. Configuration inputs (ground node positions, speed, messages deadlines)
2. Compute the distance matrix.
3. Initialize the population: randomly sorts the ground nodes order in 100 possible ways, creating the so-called population member.
4. **For** a certain number of iterations (in this case 10 000) **do**
   - 4.1. Evaluate each population member: computes the total distance and the time it will take to arrive at each ground node if it's done in that particular order. Computes how many messages will meet the deadline.
   - 4.2. Find the best route – the first criterion being the one whose route achieves more deliveries, and the second criterion being the minimum distance travelled (in case of a tie).
   - 4.3. Check if this best route is better than the best route obtained until this iteration.
   - 4.4. Save the best route.
   - 4.5. Mutation of population members occurs:
     - 4.5.1. Select 4 routes at a time.
     - 4.5.2. Retrieve the best one in terms of maximum deliveries.
     - 4.5.3. Mutate the best route to get three new routes: gets 2 random numbers I and J

       Case 1: do nothing.

       Case 2: Flip – reverse the sequence order between I and J.

       Case 3: Swap – swaps the position of I and J.

       Case 4: Slide – slides all the sequence between I and J to the left.1

The results of the TSP GA Deadlines function contain a distance matrix, a record of the best solution history, the best route obtained, as well as the corresponding distance and number of delivered messages-Figure 21 presents the results for 50 ground nodes randomly placed in a 10x10 area, having deliveries with deadline generated at random with value between 20 and 30 units of time, and UAV speed is 1 unit of distance per unit of time. Note that the best solution history now refers to most timely deliveries and not minimum distance like in figure 10.

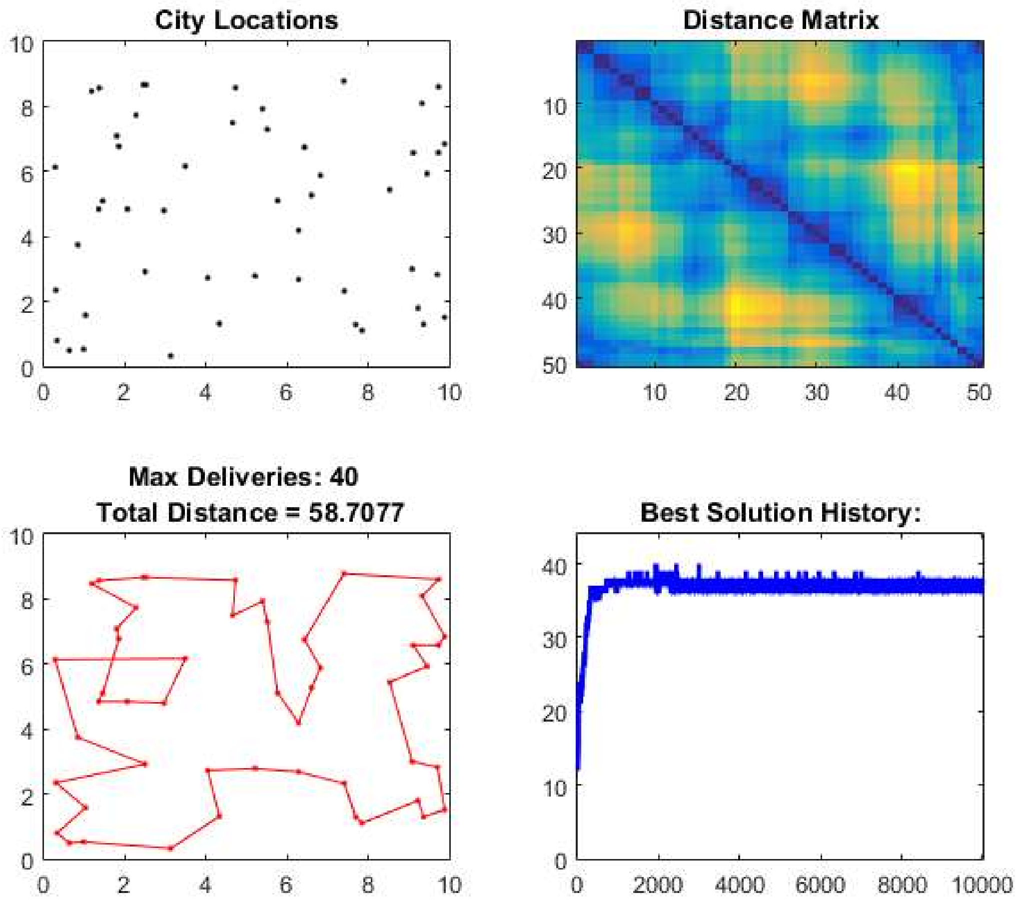


Figure 21 - TSP GA deadlines results.

Figure 22 compares the original TSP genetic algorithm and our modified version applied to the same configuration.

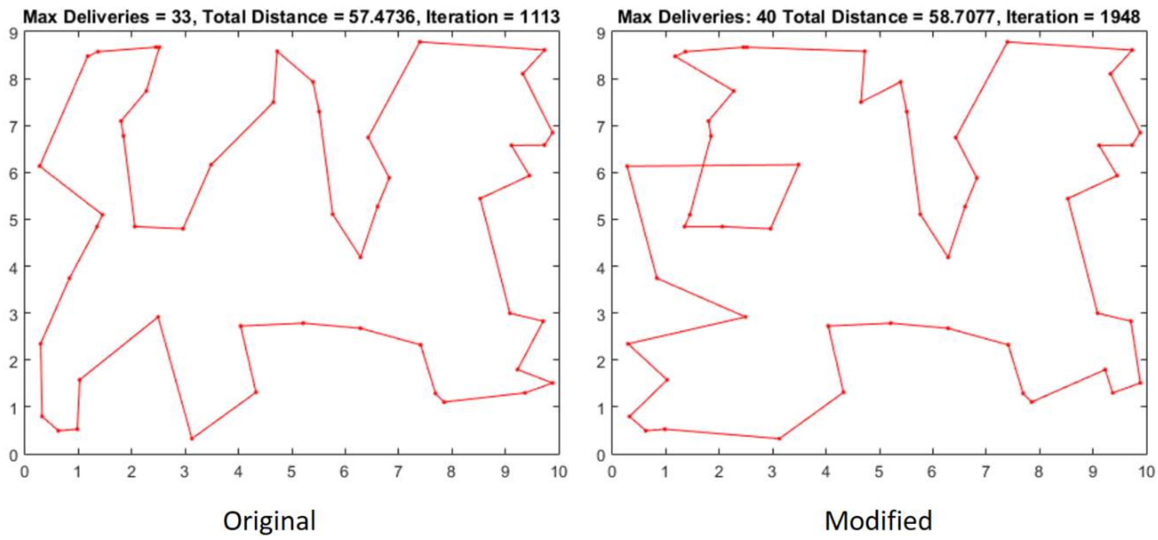


Figure 22 - TSP GA comparison.

As we can see in figure 22, with the maximum deliveries result shown above each resulting route, the modified version achieves better results for the purpose intended. Even though it does not take the smallest route, it is able to make more timely deliveries.

The TSP GA Deadlines algorithm is used as a trigger. Whenever a UAV reaches a ground node in its ferry route, it will collect messages, and right after, if it didn't fill up its deposit, it will run the TSP GA Deadlines algorithm having the next ground node position as the starting point and subtracting the time it takes to reach it. After finding how many timely deliveries it's able to achieve, if this number is less than the total number of messages in the buffer, then it will start pigeon mode and compute again the best route using TSP GA Deadline with its current position. Figure 23 present a segment of Figure 20 in more detail.

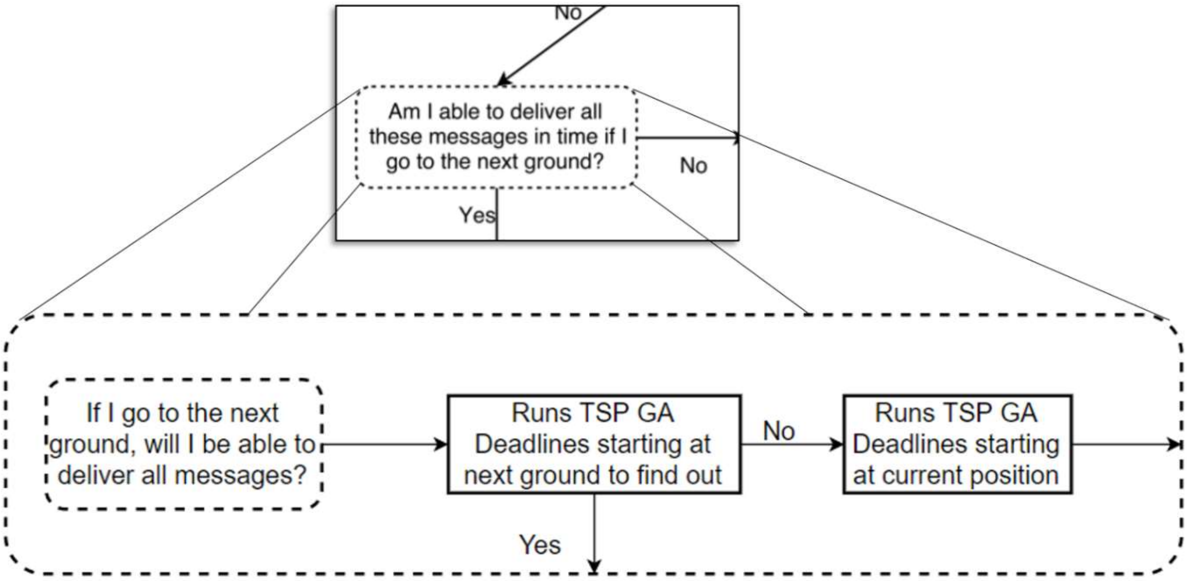


Figure 23 - Triggering step in more details.

Summing up, the main concept behind our model is to use the deadlines as a constraint to start the pigeon mode delivery when the UAV is working initially as a ferry. Compared to ferry models, the difference lies in the clustering, and in reaching nodes out of the cluster. Compared to the pigeon model, the main difference is this deadline constraint that makes the UAV start the delivery earlier than having its buffer full. In both scenarios, we seek to maximize the number of deliveries, using the modified genetic algorithm. Comparative performance evaluation is done in Chapter 5.

Chapter 5

Performance Evaluation

5.1 Introduction

In this chapter, we evaluate the performance of all the models presented earlier, using simulations results obtained with MATLAB. First are presented the simulation settings. Then, the results section is divided into two sub-section, presenting first the ferry algorithms, where there are less UAVs available than ground nodes, having DTP-TSP-D compared to SIRA and MRT-Grid. Finally, the DTP-TSP-D is compared with HoP-DTN. The performance metrics used are the delivery ratio and average delay of the delivered messages.

5.2 Simulation Settings

In our simulations, 25 nodes are randomly distributed in the deployment area. Ground nodes are stationary, and no direct communications are feasible between any pair of ground nodes. All messages have to be delivered by the UAVs. The generated messages are buffered in a First Come First Serve (FCFS) queue in the ground node, waiting for a UAV to enter its transmission range. The following assumptions were made:

1) The messages are generated at each node according to a Poisson process with a mean rate λ.
2) Each message picks one of the other ground nodes as its destination with an equal probability.
3) The buffer capacity of a ground node is unlimited.
4) All messages have the same expiration time (deadline).
5) The UAVs move around at a constant speed.
6) The ground nodes' positions are known beforehand.

The performance evaluation considers two of the most common performance metrics in protocol evaluation: Delivery Ratio and Average Delay. The definition of each performance metric is as follows:

Delivery Ratio – ratio of messages successfully received to the total generated.

Average Delay – It measures the average delay between the time a message is generated at a source and the time it reaches the destination. It captures how efficiently the path is chosen for forwarding.

To further analyze the behavior of the algorithms, the following independent variables were used: buffer size (UAV Capacity), deadline and message generation rate. Several setups were used, having the buffer size vary from 10 up to 150 messages, with increments of 10 messages, and the deadlines and generation rate λ are combined into nine different scenarios. Table 1 summarizes this setting.

Deadline (units of time)	Generation Rate (λ)	Buffer size (number of messages)
50	0.5	[10, 20, 30, … ,140,150]
	1	[10, 20, 30, … ,140,150]
	1.5	[10, 20, 30, … ,140,150]
100	0.5	[10, 20, 30, … ,140,150]
	1	[10, 20, 30, … ,140,150]
	1.5	[10, 20, 30, … ,140,150]
150	0.5	[10, 20, 30, … ,140,150]
	1	[10, 20, 30, … ,140,150]
	1.5	[10, 20, 30, … ,140,150]

Table 1 - Parameter combinations considered in the simulation scenarios.

The ground nodes are distributed in a 100x100 area, the simulation time is 1000 units and the UAV speed is 1, which means that, at each time increment, the UAV moves 1 distance unit. Table 2 sums this up.

Parameters	**Values**
Simulation duration	1000 units of time
Total number of ground nodes	25
Deployment area	100 x 100
UAV moving speed	1 unit of distance per unit of time

Table 2 - Simulation parameters.

5.2.1 Message Ferry Algorithms

When comparing DTP-TSP-D with other ferry models, we varied the UAV to ground node ratio, using 25 ground nodes together with 5 or 15 UAVS.

Using a desktop computer with a 4 GHz CPU, the average real simulation time of each algorithm is listed in the table 3.

Algorithm	**Number of UAVs**	**Time (h)**
DTP-TSP-D	5	40
	15	80
SIRA	5	12
	15	36
MRT-Grid	5	12
	15	36

Table 3 - Real simulation time of each ferry algorithm

5 UAVs

Figure 24 presents the results using 5 UAVs in the conditions explained in the previous section. Each line corresponds to a deadline value [50, 100, 150], and each column to a generation rate value [0.5, 1, 1.5], corresponding to the mean rate λ of the Poisson process. Table 2 shows the combinations corresponding to each scenario.

		Generation Rate (λ)		
		0.5	1	1.5
Deadline	50	(a)	(b)	(c)
	100	(d)	(e)	(f)
	150	(g)	(h)	(i)

Table 4 - Combinations of Deadline and Generation rate corresponding to each simulation scenario.

The error bars represent the 95% Confidence Interval.

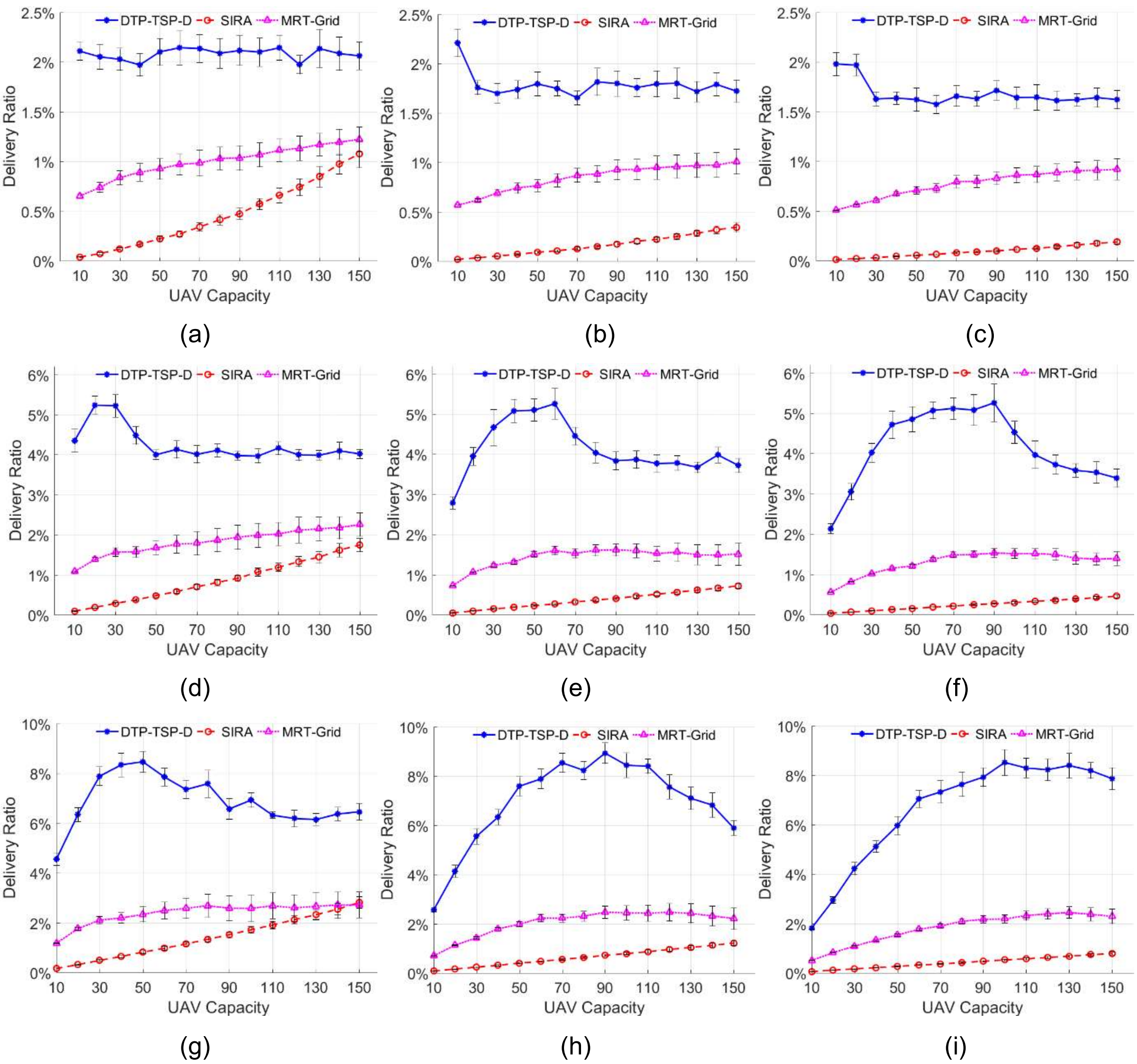
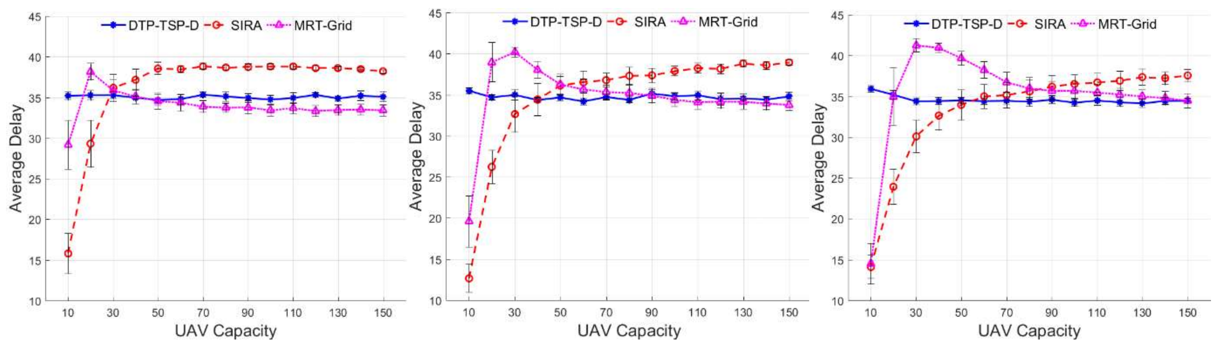


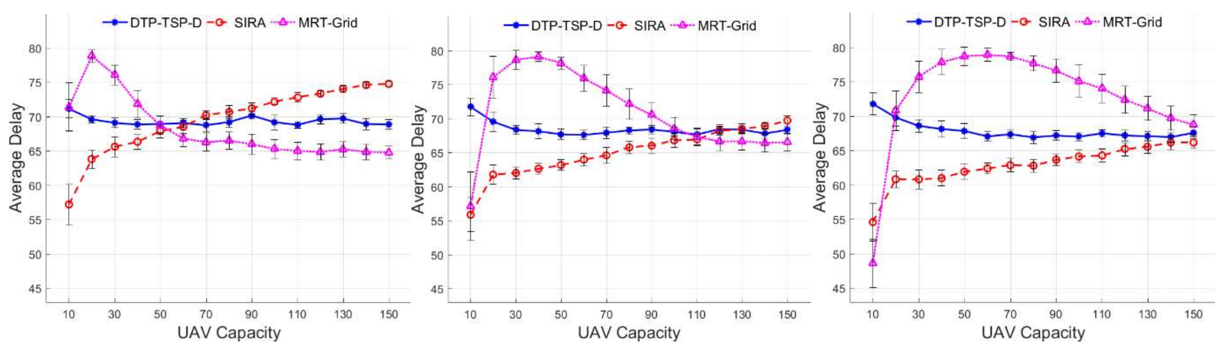
Figure 24 - Delivery Ratio results, for 5 UAVs:
(a) Generation Rate 0.5; Deadline 50; (b) Generation Rate 1; Deadline 50; (c) Generation Rate 1.5; Deadline 50;
(d) Generation Rate 0.5; Deadline 100; (e) Generation Rate 1; Deadline 100; (f) Generation Rate 1.5; Deadline 100;
(g) Generation Rate 0.5; Deadline 150; (h) Generation Rate 1; Deadline 150; (i) Generation Rate 1.5; Deadline 150;

Overall, our DTP-TSP-D obtains better results in terms of delivery ratio, with the corresponding curve being always higher than the others with the improvement ranging between 2.6 to 163.6 times better. With lower deadlines, [Figure 24 (a), (b) and (c)] the delivery ratio is practically constant with the increasing UAV capacity. However, for higher deadlines, we can see a different behavior, with the delivery ratio growing initially with the buffer capacity, reaching a peak and then decreasing a little. This behavior can be explained by a first phase where the lack of space in the buffer has a greater influence, having the pigeon mode triggered more frequently due to reaching full capacity. The highest value must correspond to an equilibrium between capacity triggered and deadline triggered. From that point on, the delivery ratio slightly decreases, meaning that the lack of capacity triggering results in lower delivery ratios. This is an expected result, since the UAV starts less trips in pigeon mode, thus not delivering so many messages in its way, and not bringing back messages to its cluster.

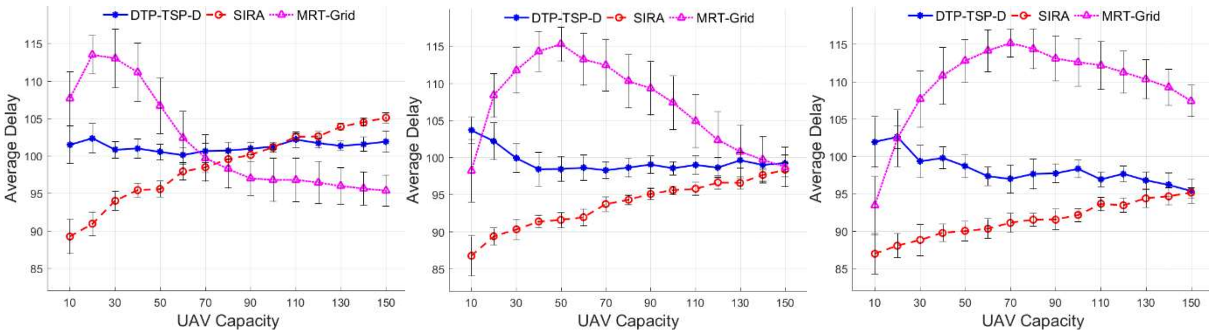
The behavior of SIRA and MRT-Grid is similar in all scenarios against the increase of buffer size, with the delivery ratio increasing as the buffer capacity increases. We can also observe that these algorithms are better suited for small generation rate, having a worse delivery ratio when the generation rate increases [Figure 24 from left to right], meaning that they are unable to drain all the traffic in the network, as the number of generated messages per unit of time increases. Overall, MRT-Grid is better than SIRA, proving that small clusters instead of one single route improve the delivery ratio, and thus the network performance. With only one route, for small deadlines, SIRA's valid transmissions are those made for nearby ground nodes next on the route, which are the only ones able to comply with deadlines. On the other hand, MRT-Grid intra-cluster transmissions are easily reachable, which can explain the difference obtained in delivery ratio. Figure 25 presents the average delay for each model in the nine scenarios listed in Table 2, with 5 UAVs.



(a) (b) (c)



(d) (e) (f)



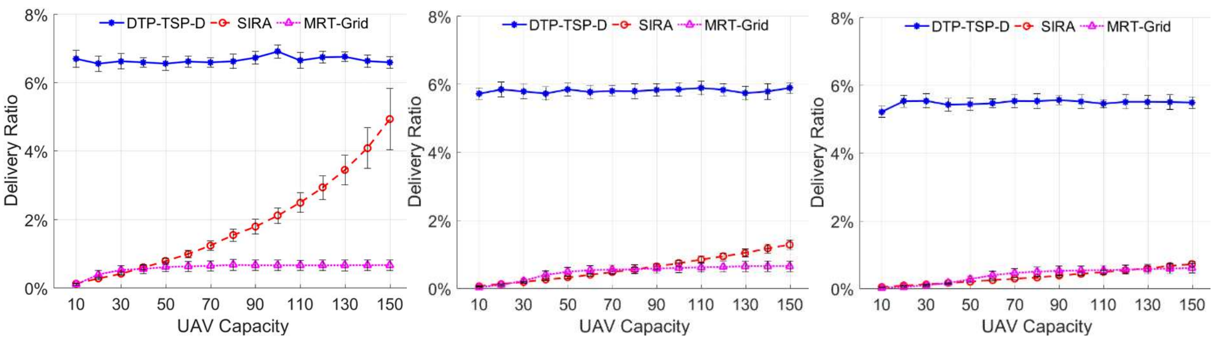
(g) (h) (i)

Figure 25 - Average Delay results, for 5 UAVs:
(a) Generation Rate 0.5; Deadline 50; (b) Generation Rate 1; Deadline 50; (c) Generation Rate 1.5; Deadline 50;
(d) Generation Rate 0.5; Deadline 100; (e) Generation Rate 1; Deadline 100; (f) Generation Rate 1.5; Deadline 100;
(g) Generation Rate 0.5; Deadline 150; (h) Generation Rate 1; Deadline 150; (i) Generation Rate 1.5; Deadline 150;

Here, the behaviors are quite diverse. There is no algorithm that is always better than the others. However, the average delay achieved by DTP-TSP-D is more constant than its competitors. Overall, SIRA is able to achieve smaller average delay, which was predictable due to the fact that its valid transmissions are only those with destinations close enough and next in the route. Thus, the deadlines impact both delivery ratio and average delay. If there was no time limit, SIRA would be able to deliver more messages, but would also attain worse average delay. On the other hand, MRT-Grid achieves overall greater average delays than the others, which may result from the amount of time lost at the relaying nodes, waiting for a UAV to take the messages to the next hop. Since DTP-TSP-D is deadline triggered, it was expected that the average delays would be closer to the deadlines. However, since every time the UAV is in pigeon mode it retrieves messages to its home cluster, those messages are delivered with smaller delay than those taken from the home cluster to the outside. Therefore, the average delay is quite smaller than the deadline.

15 UAVs

Figure 26 presents the results for 15 UAVs, with the conditions being the same as in the previous section.



(a) (b) (c)

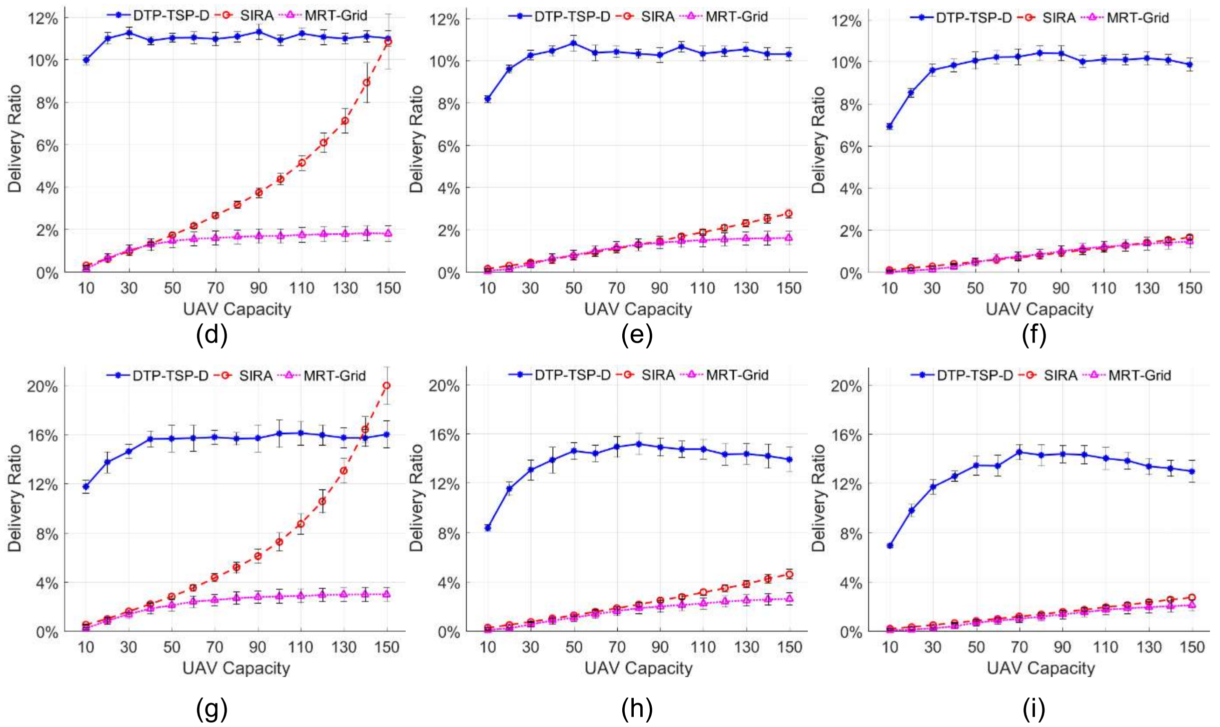


Figure 26 - Delivery Ratio results, for 15 UAVs:
(a) Generation Rate 0.5; Deadline 50; (b) Generation Rate 1; Deadline 50; (c) Generation Rate 1.5; Deadline 50;
(d) Generation Rate 0.5; Deadline 100; (e) Generation Rate 1; Deadline 100; (f) Generation Rate 1.5; Deadline 100;
(g) Generation Rate 0.5; Deadline 150; (h) Generation Rate 1; Deadline 150; (i) Generation Rate 1.5; Deadline 150;

DTP-TSP-D continues to perform better than the others in terms of delivery ratio. Although it appears that, with a higher number of UAVs, SIRA achieves comparable performance for a small generation rate [Figure 26 (a), (d), (g)]. Having more UAVs allows SIRA to deliver more traffic, with ground nodes visited more frequently. On the other hand, MRT-Grid seems to get worse as the number of UAVs increases. Figures 27, 28 and 29, allow to better compare the effect of adding more UAVs to the network, for each scheme.

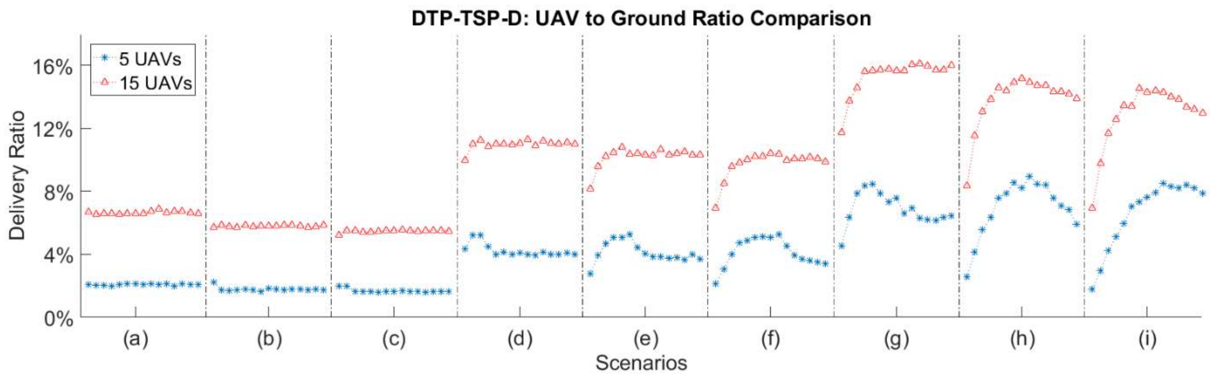


Figure 27 - DTP-TSP-D: Delivery Ratio results with 5 and 15 UAVs.

Figure 27 shows that DTP-TSP-D improves in terms of Delivery Ratio when more UAVs are employed. The behavior in each scenario mostly remains the same, achieving on average 2.12 times more timely deliveries.

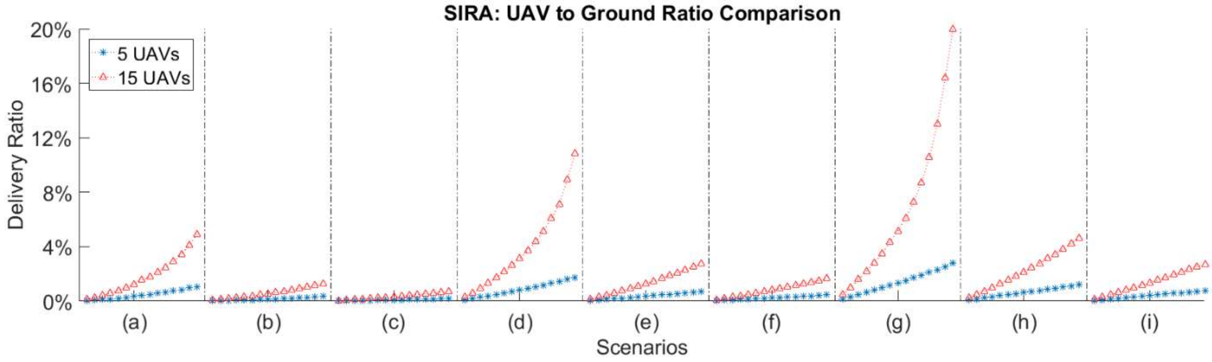


Figure 28 - SIRA: Delivery Ratio results with 5 and 15 UAVs.

Figure 28 shows that SIRA also improves when more UAVs are available, which is due to having ground nodes visited more often, as referred earlier. Here we can observe again that SIRA is more suitable for low generation rates, corresponding to scenarios (a), (d) and (g).

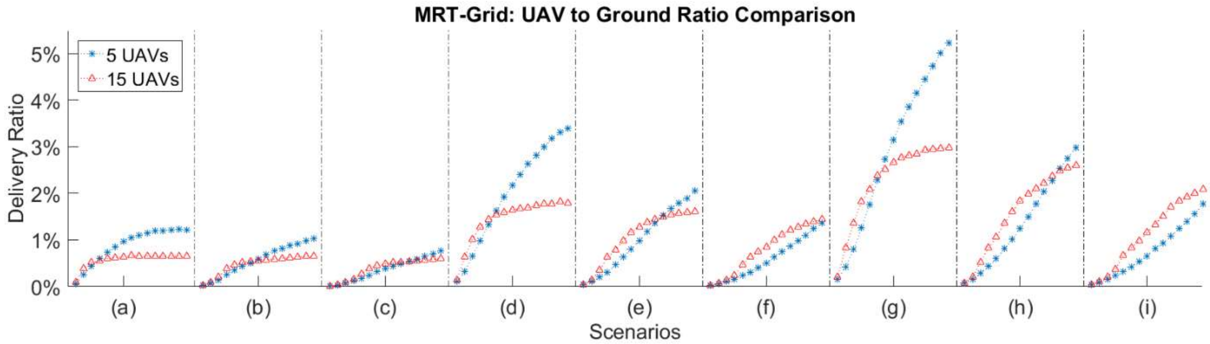


Figure 29 - MRT-Grid: Delivery Ratio results with 5 and 15 UAVs.

On the contrary, more UAVs does not affect MRT-Grid in a positive way. Results in Figure 29 show that the delivery ratio obtained with more UAVs is frequently smaller than the one obtained with fewer UAVs. The reason behind this result is, as portrayed in section 3.3.4, the fact that the network with 15 UAVs becomes too partitioned, having messages from ground nodes that are far apart, going through a high number of relay nodes to get to their destination. Figure 30 exemplifies a graph using 15 partitions. A simple example of MRT-Grid shortcoming, is picturing a message created in ground node 1 (bottom left of Figure 30) with ground node 12 as its destination (Upper right of Figure 30). It would have to go through a high number of relays, waiting for a carrier in each one and, due to the deadline, the message would probably not arrive in time to its destination.

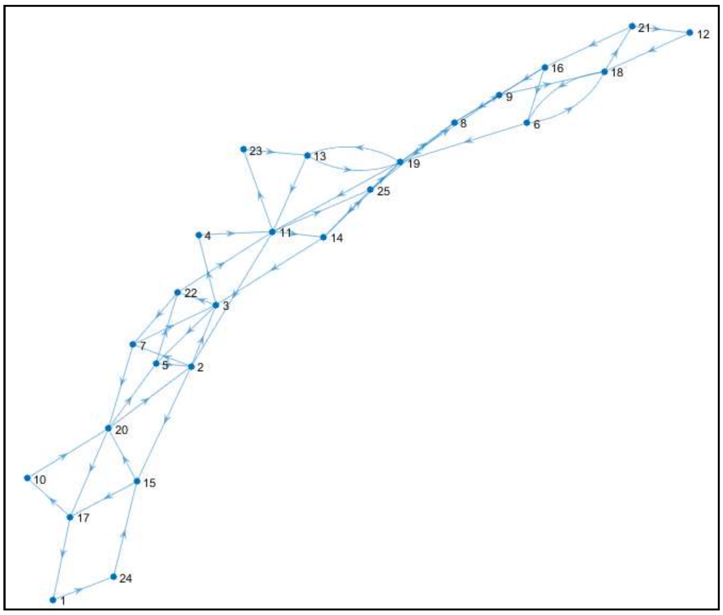
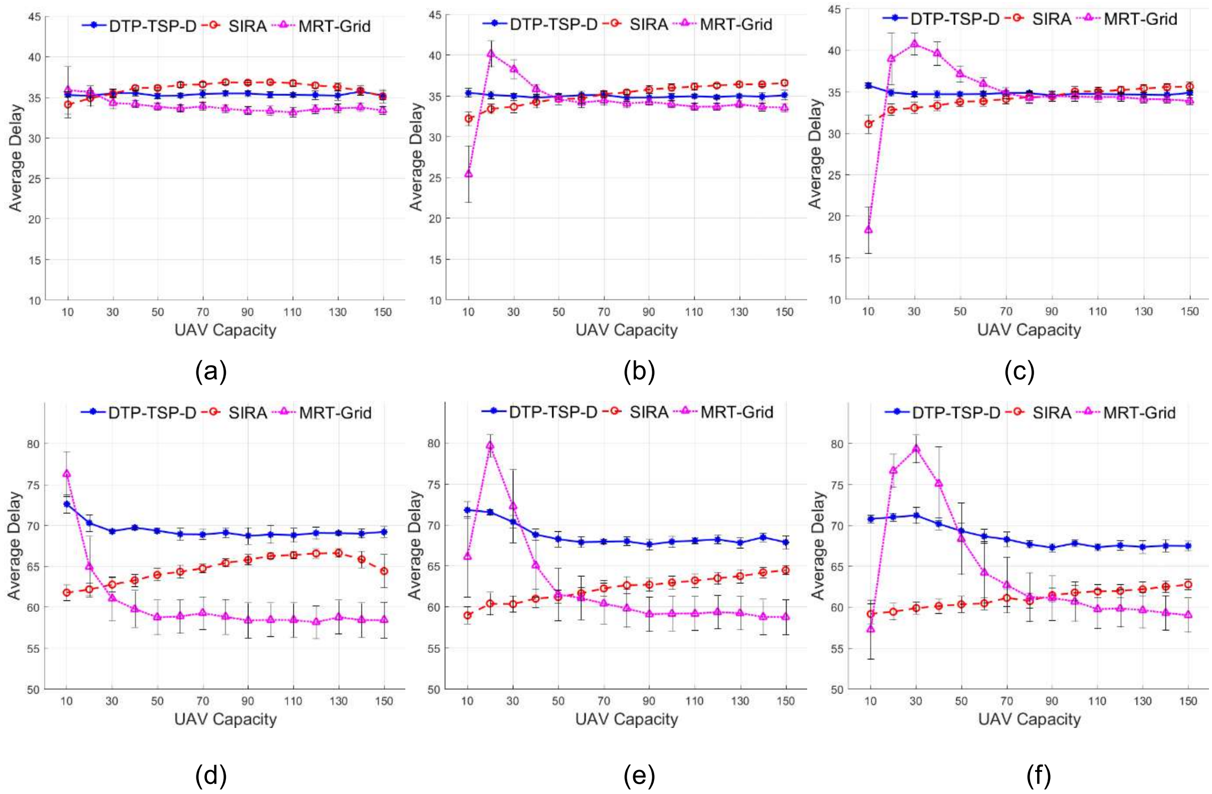
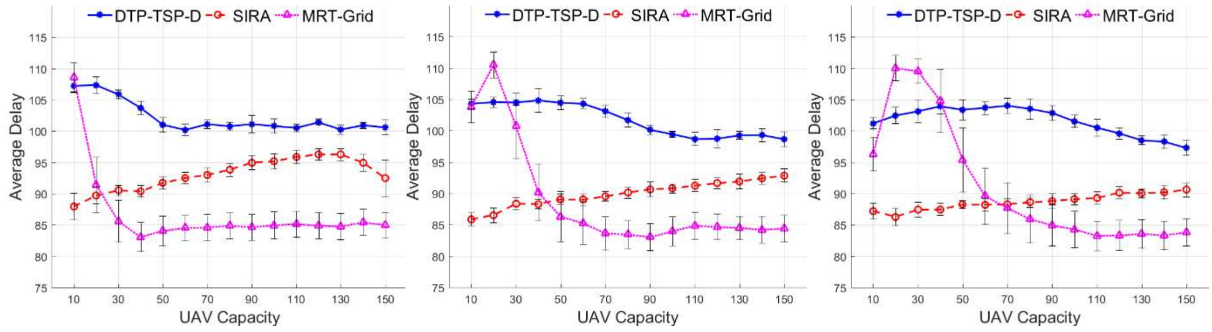


Figure 30 - Graph corresponding to clustering 25 ground nodes in 15 clusters with MRT-Grid.

Figure 31 presents the average delay in the nine scenarios with 15 UAVs, for each scheme.





(g) (h) (i)

Figure 31 - Average Delay results, for 15 UAVs:
(a) Generation Rate 0.5; Deadline 50; (b) Generation Rate 1; Deadline 50; (c) Generation Rate 1.5; Deadline 50;
(d) Generation Rate 0.5; Deadline 100; (e) Generation Rate 1; Deadline 100; (f) Generation Rate 1.5; Deadline 100;
(g) Generation Rate 0.5; Deadline 150; (h) Generation Rate 1; Deadline 150; (i) Generation Rate 1.5; Deadline 150;

As seen for 5 UAVs, DTP-TSP-D average delay seem to remain essentially constant in each scenario, and not be affected by the generation rate (the average delay mostly remains the same observing Figure 31 from left to right). However, its average delay is frequently the highest of the three algorithms. Contrary to what was observed with 5 UAVs, MRT-Grid with 15 UAVs often achieves lower average delay than its competitors. Figures 32, 33 and 34, allow to better compare the effect of adding more UAVs to the network for each scheme, in terms of average delay.

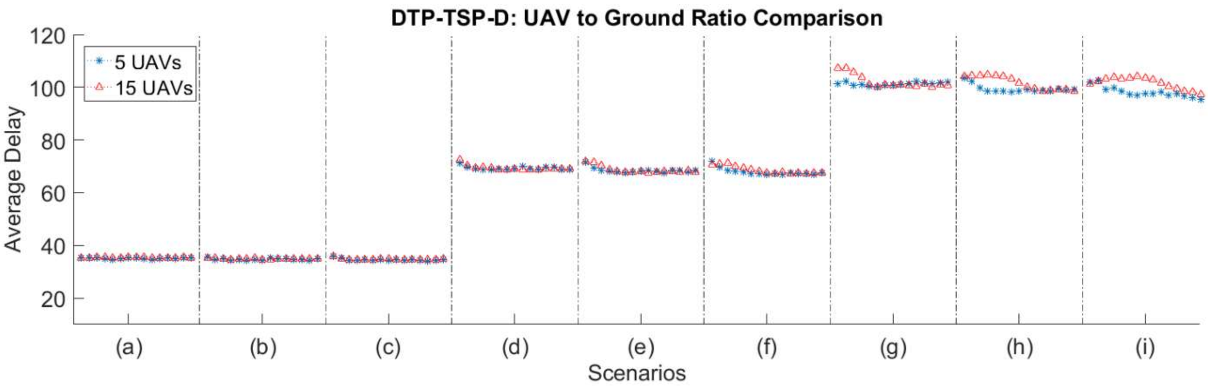


Figure 32 - DTP-TSP-D: Average Delay results with 5 and 15 UAVs.

Figure 32 shows that DTP-TSP-D average delay is not significantly affected by using more UAVs in the network.

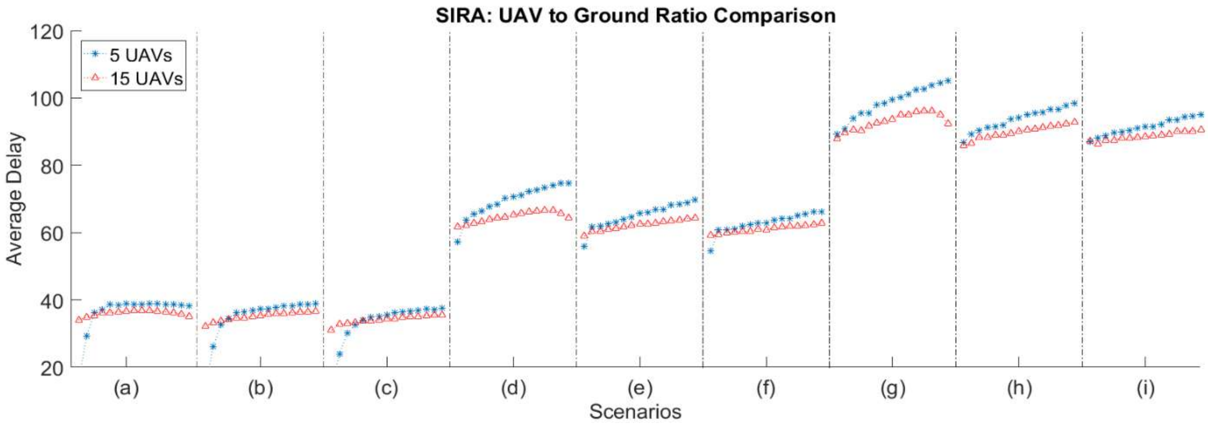


Figure 33 - SIRA: Average Delay results with 5 and 15 UAVs.

More UAVs seem to affect SIRA in a positive way, since globally it achieves lower average delay with 15 UAVs than with 5 UAVs. The reason behind these results, is the fact of having UAVs passing more frequently, meaning the messages will wait less time in the source than with 5 UAVs.

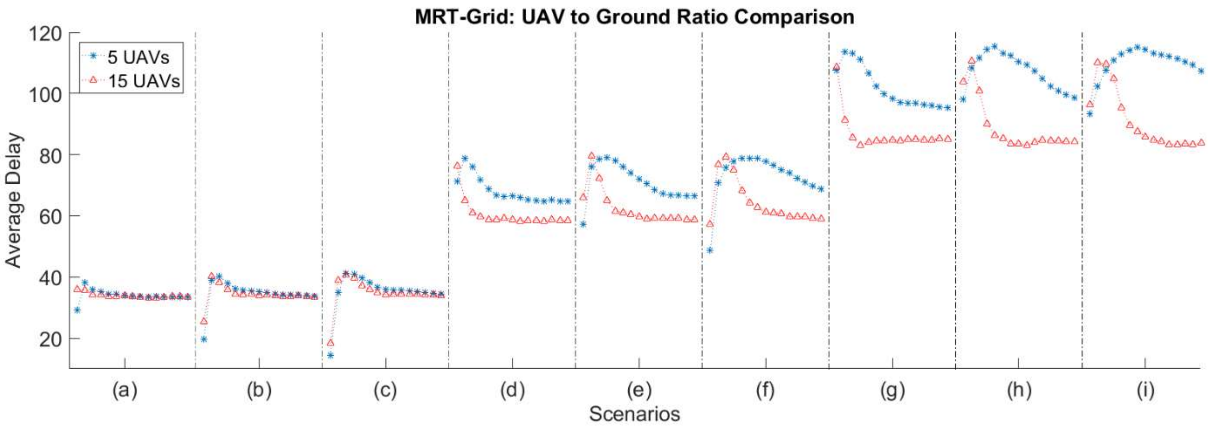


Figure 34 - MRT-Grid: Average Delay results with 5 and 15 UAVs.

MRT-Grid also presents lower average delay with the use of more UAVs. However, these results do not mean that having more UAVs is better, since the reason the average delay decreases is the fact that the range of valid transmissions also decreases. In other words, as it was shown in Figure 30, with more UAVs the graph gets more complicated, with messages being relayed through a higher number of relay nodes. This means that the valid transmissions, i.e., the number of messages arriving within bounds, are only those intended for close ground nodes that need fewer hops.

Discussion

The results obtained in this section demonstrates that the DTP-TSP-D achieves better delivery ratio than SIRA and MRT-Grid in every scenario analyzed, with the exception being the combination of low generation rate, high deadlines, large buffer size, and high UAV to ground node ratio, where SIRA has a better performance [Figure 26 (g)]. In terms of average delay, DTP-TSP-D does not score the lowest in any scenario. However, it is the only scheme that can deliver from any source to any destination. As was explained before, when dealing with deadlines, SIRA and MRT-Grid are often able to achieve lower delays due to the short range of valid transmissions in those schemes, which represents a substantial handicap. Regarding the use of more UAVs, MRT-Grid would benefit if the UAV assignment was done a different way. It would be interesting to see the results using the same number of UAVs but instead of each one having a cluster, assigning more than one per route, merging the advantage of more UAVs in SIRA with clusters in MRT-Grid. It is important to refer that the processing delay wasn't taken into account in any of these simulations, which would surely influence the results of the DTP-TSP-D.

5.2.2 Homing Pigeon Algorithms

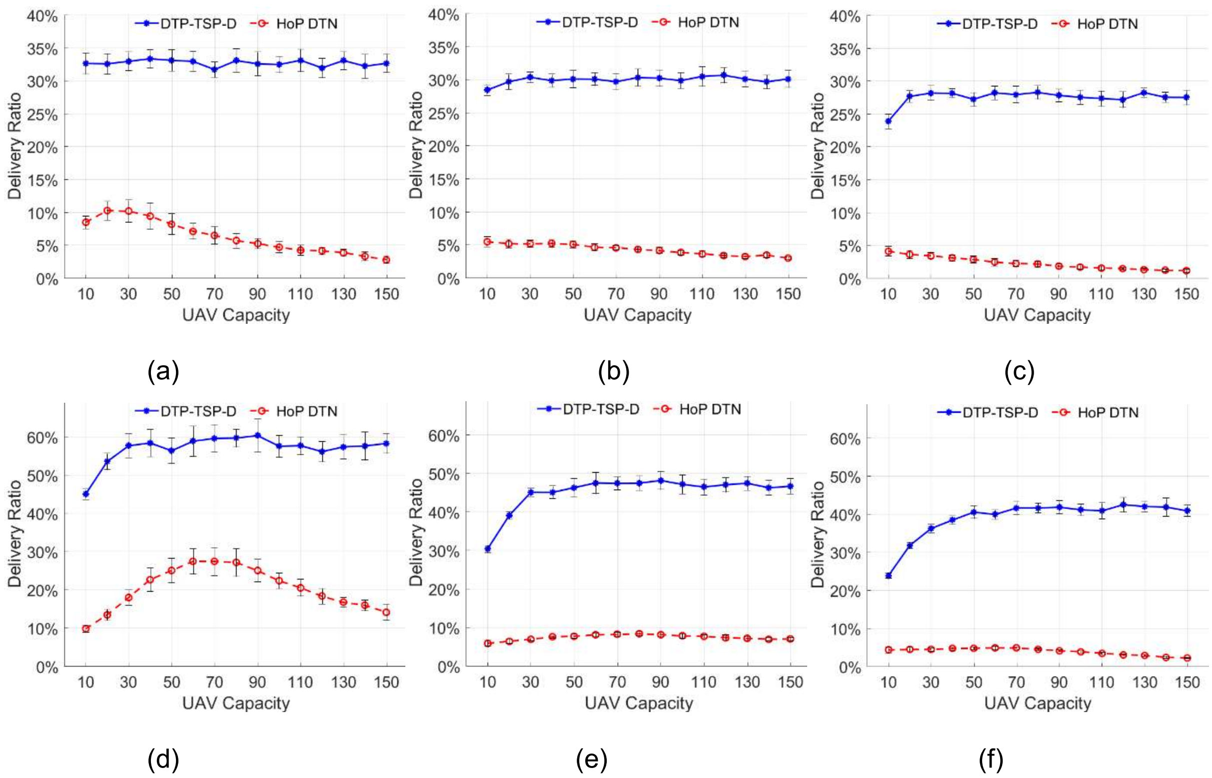
Due to the time frame of this master thesis, the pigeon models were only simulated with 5 ground nodes, leaving aside the 25 used with the previous ferry algorithms. In these simulation settings, there are 5 ground nodes, having each one its own dedicated messenger.

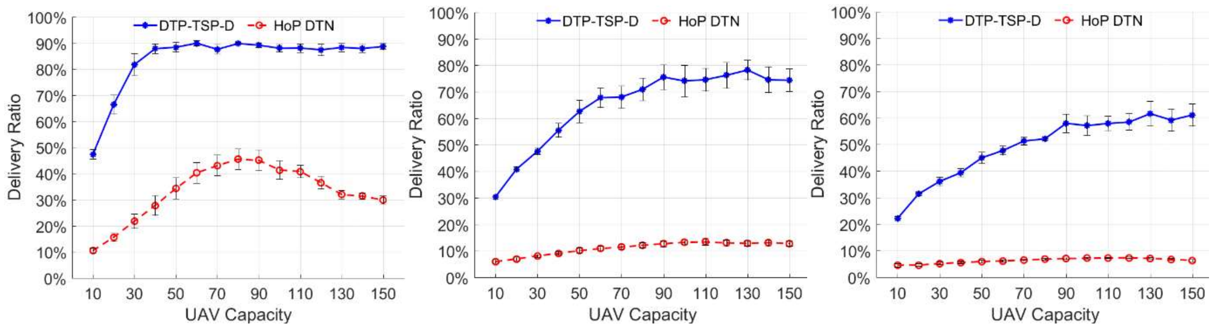
Using a desktop computer with a 4 GHz CPU, the average real simulation time of each algorithm, for each set of ground nodes, is listed in table 5.

Algorithm	Time (h)
DTP-TSP-D	40
HoP DTN	20

Table 5 - Real simulation time of each pigeon algorithm

Figure 35 shows the results obtained for HoP-DTN and DTP-TSP-D, in scenarios with the configurations specified in Table 2.





(g) (h) (i)

Figure 35 - Delivery Ratio results, for 5 UAVs and 5 Ground Nodes:
(a) Generation Rate 0.5; Deadline 50; (b) Generation Rate 1; Deadline 50; (c) Generation Rate 1.5; Deadline 50;
(d) Generation Rate 0.5; Deadline 100; (e) Generation Rate 1; Deadline 100; (f) Generation Rate 1.5; Deadline 100;
(g) Generation Rate 0.5; Deadline 150; (h) Generation Rate 1; Deadline 150; (i) Generation Rate 1.5; Deadline 150;

DTP-TSP-D clearly outperforms HoP-DTN in every scenario, in terms of delivery ratio. Looking at Figure 35 from left to right, we can observe that the delivery ratio decreases with the increasing generation rate. On the contrary, observing from up to bottom, it is clear that higher deadlines are favorable for DTP-TSP-D. HoP DTN also performs better under lower generation rate [Figure 35, from left to right], and achieves higher values of delivery ratio with higher messages deadlines [Figure 35, from up to bottom]. To figure out what is the best improvement in DTP-TSP-D, we developed two new algorithms that can be considered as middle steps between HoP-DTN and DTP-TSP-D. First, a Capacity Triggered Pigeon with TSP Deadline delivery (CTP-TSP-D), that fills up its buffer the same way HoP-DTN does, but delivers the messages according to the modified TSP GA deadlines instead of using the original TSP GA. This will allow us to understand the impact of taking deadlines into account when delivering messages. The second scheme has the same characteristics as the last one, with the advantage of every pigeon collecting messages that have its home ground node as destination, during the delivering phase. Figure 36 illustrates all four schemes for every scenario.

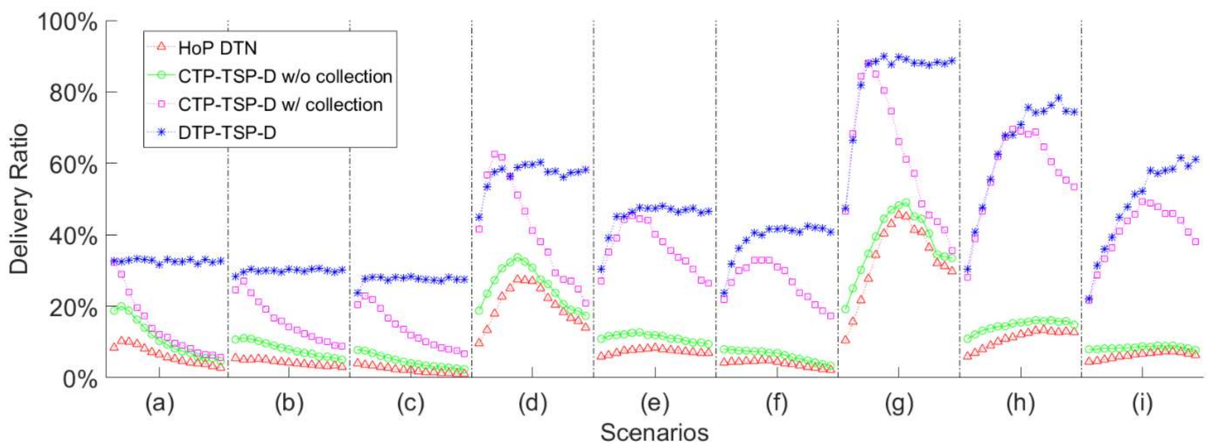


Figure 36 - Delivery Ratio comparison for all scenarios.

Comparing CTP-TSP-D w/o collection to HoP-DTN, it shows that deliveries taking deadlines into account are always better, with improvement ranging from 5% to 125%. However, it is not as significant as collecting messages from outside nodes to the home ground node, which improves up to 700% the delivery ratio. Figure 36 also allows to conclude that the deadline triggering is effective when the buffer size increases, having all other curves in the plot decreasing significantly and DTP-TSP-D remaining almost constant.

Regarding the average delay, Figure 37 shows the simulation results obtained with HoP-DTN and DTP-TSP-D.

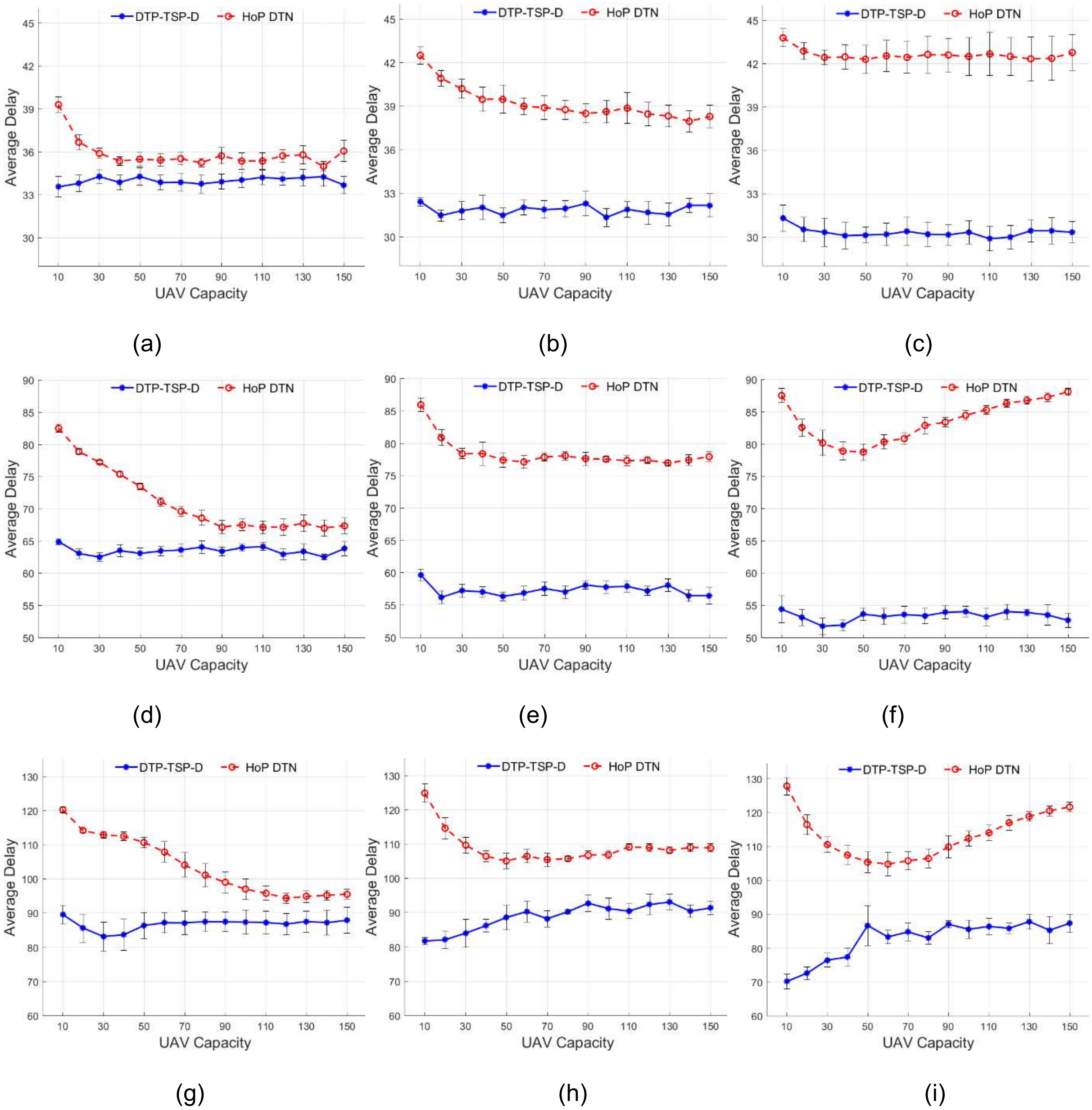


Figure 37 - Average Delay results, for 5 UAVs and 5 Ground Nodes:
(a) Generation Rate 0.5; Deadline 50; (b) Generation Rate 1; Deadline 50; (c) Generation Rate 1.5; Deadline 50;
(d) Generation Rate 0.5; Deadline 100; (e) Generation Rate 1; Deadline 100; (f) Generation Rate 1.5; Deadline 100;
(g) Generation Rate 0.5; Deadline 150; (h) Generation Rate 1; Deadline 150; (i) Generation Rate 1.5; Deadline 150;

As Figure 37 shows, DTP-TSP-D is also better in every scenario in terms of Average Delay. We can also observe that the average delay tends do decrease with the increasing generation rate [Figure 37, from left to right]. Figure 38 shows the results aggregated for these algorithms and those middle steps algorithm explained before.

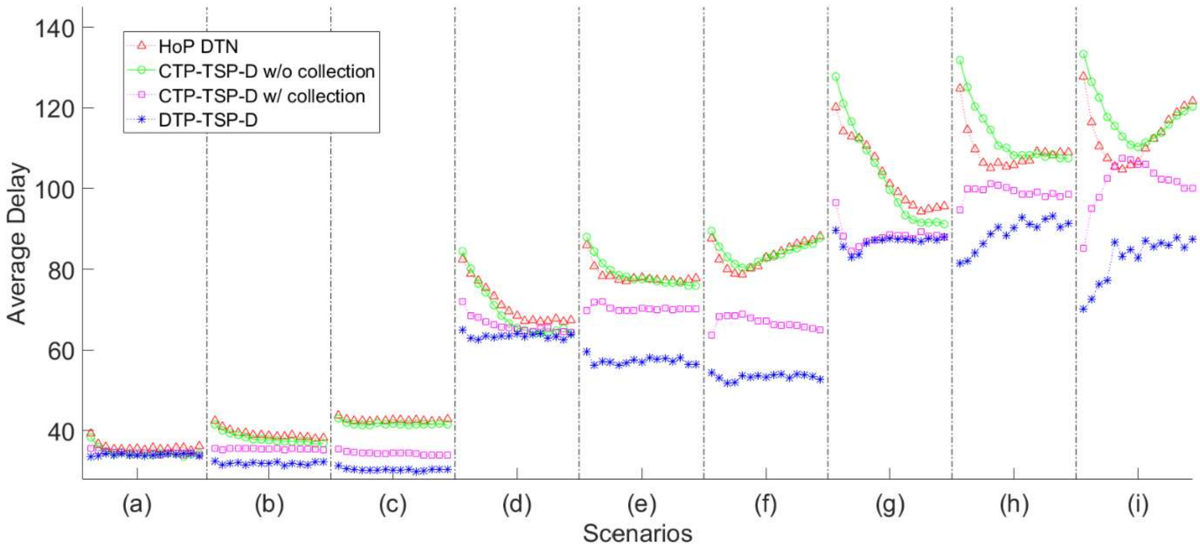


Figure 38 - Average Delay comparison for all scenarios.

From the results shown in Figure 38, we can conclude that the delivery with the modified TSP GA deadline does not significantly affect the average delay. On the other hand, bringing back messages to the home ground node decreases it by 11% on average, which was expected since those specific messages are delivered before the deadline, not having to wait till their own pigeon's buffer is full to get delivered. They thus contribute to significantly reduce the average delay. DTP-TSP-D continues to score lower average delay due to the triggering deadlines, since in this algorithm the pigeons do not have to wait till their buffer is full, and depart as soon as the timely delivery cannot wait any longer.

Chapter 6

Conclusions and Future Work

This final chapter presents the main conclusions drawn from the developed work, as well as some suggestions for future work.

6.1 Conclusions

This master thesis proposes a new dedicated node-based protocol, assuming a DTN where ground nodes are far apart and the only mean of communication is by flying UAVs from one location to another.

The literature review revealed two major approaches in this area: Message Ferrying and the Homing Pigeon schemes. Both present advantages, shortcomings and room for improvement. The extra auxiliary nodes (UAVs) could be shared by all the ordinary nodes (ground nodes) in the network, where multiple ferries could follow one single route (SIRA) or be divided in clusters as in MRT-Grid. Or, those special nodes could be owned by each ordinary node and follow solely their demands, as implemented in HoP DTN.

Based on those existing approaches, we developed a new scheme that uses the delivering format of HoP DTN, improved by a TSP genetic algorithm that computes the UAV routes, improving the (timely) delivery ratio, making use of a deadline trigger that allows the UAV to start its pigeon delivery route even before its buffer becomes full. The proposed scheme is designated Deadline Triggered Pigeon with Travelling Salesman Problem with Deadlines (DTP-TSP-D). When there are fewer UAVs available than ground nodes in the network, the UAVs may act as ferries, each being assigned to a cluster of ground nodes instead of one ground node only. The UAV will hover over the ground nodes in its cluster by following a ferry route obtained beforehand by a basic TSP GA. The UAV will follow this route, until it is triggered to become a pigeon, whereupon it delivers the messages targeted to ground nodes outside its cluster.

All algorithms were implemented in MATLAB. Simulations scenarios were created to evaluate the performance of DTP-TSP-D, SIRA, MRT-Grid and HoP DTN. The considered performance metrics were delivery ratio and average delay. Nine scenarios were used with different combinations of message generation rates and deadlines.

We started by comparing DTP-TSP-D to the MF schemes, namely: SIRA, MRT-Grid. The results demonstrate that the DTP-TSP-D outperforms the other schemes in terms of delivery ratio, although it does not always present the best average delay. Besides, by using different UAV to ground node ratios, first 5 UAVs to 25 ground nodes and later 15 UAVs to 25 ground nodes, we could observe that more UAVs means more deliveries, with the delivery ratio increasing on average approximately twice with the use of 15 UAVs versus using 5 UAVs. We also realized that adding more UAVs does not significantly affect the average delay.

When compared with HoP DTN, DTP-TSP-D attained better performance by a huge margin in both performance metrics. Using two middle step algorithms between HoP DTN and DTP-TSP-D, we concluded that the best improvement feature is collecting messages directed to the home ground node during pigeon mode (i.e., actual delivery to the destination). This feature allows to achieve better delivery ratios, up to 7 times higher than without this feature, and lower average delays, decreasing on average 11% when the feature is used. We also concluded that the deadline triggering method is useful when the buffer size of the UAV increases, meaning that the saved time by not filling up the buffer results in more successful deliveries.

6.2 Future Work

Due to the time frame of this master thesis, there was no time to test the DTP-TSP-D algorithm in a larger scale, which could be relevant. However, that also means more complex calculation of TSP solutions, which would be interesting to analyze in a real UAV, in terms of processing delay, physical transmission delay and power consumption.

It would also be interesting to test situations when there are more UAVs available than ground nodes. In fact, it should be defined how to employ the extra UAVs: e.g., should they be assigned to ground nodes as additional pigeons or be employed as message ferries in a hybrid scheme?

We considered the ground nodes to be stationary, future work may include applying this algorithm when partitioned clusters are mobile and dynamic, which may entail the development of ground node trajectory learning algorithms and collaboration between UAVs (e.g., by establishing intersecting routes or meeting points).

Bibliography

[1] K. Fall, ``A delay-tolerant network architecture for challenged internets", in *Proceedings of the 2003 Conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '03). ACM*, NY, USA, pp.27-34.

[2] C. Desai et al., “A Survey on Knowledge Based Classification of Different Routing Protocols in Delay Tolerant Networks”, *IJCSMC*, v. 2, no. 3, pp.83-88, March 2013.

[3] Sobin CC. et al, ``A survey of routing and data dissemination in Delay Tolerant Networks", in *JNCA*, vol.67, Suppl. C, pp. 128 - 146, London, UK, May 2016.

[4] A. Vahdat, and D. Becker, ``Epidemic Routing for Partially-Connected Ad Hoc Networks", in *Technical Report CS-200006*, Duke University, NC, April 2000.

[5] T. Spyropolous et al., ``Spray and wait: an efficient routing scheme for intermittently connected mobile networks", in *Proceedings of the ACM SIGCOMM Workshop on Delay Tolerant Networks (WDTN)*, Philadelphia, USA, August 2005.

[6] A. Lindgren et al., ``Probabilistic routing in intermittently connected networks", in *ACM SIGMOBILE Mobile Computing and Communications Review,* vol.7, no.3, pp. 19-20, New York, USA, July 2003.

[7] B. Burns et al., ``MV routing and capacity building in disruption tolerant networks", in *Proceedings of IEEE INFOCOM*, pp. 398-408, FL, USA, March 2005.

[8] W. Zhao et al., ``A message ferrying approach for data delivery in sparse mobile ad hoc networks", in *Proceedings of the 5th ACM MobiHoc*, Tokyo, Japan, May 2004, pp. 187-198.

[9] W. Zhao et al., ``Controlling the mobility of multiple data transport ferries in a delay-tolerant network*",* in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, FL, USA, March 2005.

[10] Z. Zhang, Z. Fei, ``Route Design for Multiple Ferries in Delay Tolerant Networks", in *Wireless Communications and Networking Conference (WCNC)*, Kowloon, China, March 2007.

[11] L. Xue et al., ``An Adaptive Message Ferry Routing Algorithm for Delay Tolerant Networks", in *ICCT- International Conference in Communication Technology*, Chengdu, China, November 2013.

[12] H. Guo et al., ``HoP-DTN: Modeling and Evaluation of Homing-Pigeon-Based Delay Tolerant Networks" in *IEEE Transactions on Vehicular Technology*, vol.59, n.2, pp. 857-868, February 2010

[13] H. Guo et al ``A practical routing strategy in Delay Tolerant Networks using multiple pigeons", in *MILCOM*, San Diego, CA, USA, November 2008.

[14] Y. Zeng et al. ``Wireless Communications with Unmanned Aerial Vehicles: Opportunities and Challenges", in *IEEE Communications Magazine*, vol.54, n.5, pp. 3-42, May 2016.

[15] STN Sarris, Zak. “Survey of UAV applications in civil markets”, in *The 9th IEEE Mediterranean Conference on Control and Automation (MED’01)*, 2001.

[16] N. Uchida et al. ``Proposal of delay tolerant network with cognitive wireless network for disaster information network system", in *Proceedings of the 27th Advanced Information Networking and Applications Workshops (WAINA),* pp. 249–254, Barcelona, Spain, March 2013.

[17] J. Albuquerque, S. Lucena and C. Campos, ``Evaluating data communications in natural disaster scenarios using opportunistic networks with Unmanned Aerial Vehicles", in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Rio de Janeiro, Brazil, November 2016.

[18] J. Kwon and S. Hailes, ``Scheduling UAVs to Bridge Communications in Delay-Tolerant Networks Using Real-Time Scheduling Analysis Techniques" in *Proceedings of the IEEE/SICE International Symposium on System Integration*, Tokyo, Japan, December 2014.

[19] S. Morgenthaler et al. ``UAVNet: A mobile wireless mesh network using Unmanned Aerial Vehicles", in *Globecom Workshops (GC Wkshps),* pp. 1603-1608, CA, USA, December 2012.

[20] Laurent Reynaud, Isabelle Guerin-Lassous, ``Improving the Performance of Challenged Networks with Controlled Mobility". In *8th International Conference ADHOCNETS*, Ottawa, Canada. Springer, 184, pp.205 - 217, September 2016.

[21] L. Gupta, R. Jain and G. Vaszkun ``Survey of Important Issues in UAV Communication Networks", in *IEEEE COMST*, vo.18, no.2, pp. 123-1152, November 2015.

[22] I. Jawhar et al., “Communication and networking of UAV-based systems: Classification and associated architectures”, in *JNCA*, vol.84, pp. 93-108, Apr. 2017.

[23] J. Kirk, Traveling Salesman Problem - Genetic Algorithm – File Exchange – MATLAB Central. [online] Available at: https://www.mathworks.com/matlabcentral/fileexchange/13680-traveling-salesman-problem-genetic-algorithm [Accessed 29 August 2017].

[24] Mathworks.com. (2017). *k-means clustering - MATLAB kmeans - MathWorks United Kingdom*. [online] Available at: https://www.mathworks.com/help/stats/kmeans.html [Accessed 2 Sep. 2017].

[25] M. Dorigo et al., “The ant system: Ant autocatalytic optimizing process” in *Technical Report TR91-016*, Politenico di Milano, 1991.

[26] M. Dorigo et al., “Ant algorithms for discrete optimization”, in *Artificial Life 5*, pp.137–172, April 1999.

[27] S. Kirkpatrick; C. D. Gelatt; M. P. Vecchi Science, “Optimization by Simulated Annealing”, Science, New Series, Vol. 220, No. 4598. pp. 671-680, May 1983.

[28] MATLAB and Statistics Toolbox Release 2016b, The MathWorks, Inc., Natick, Massachusetts, United States.

[29] Mineses, E. (2017). *Not all drones are created equal*. [online] Mashable. Available at: http://mashable.com/2014/11/21/drone-infographic/#8DPQ3S9TUkq4 [Accessed 6 Sep. 2017].

[30] Space.com (2017). *Drones Save Lives in Disasters, When They’re Allowed to Fly.* [online] Available at: https://www.space.com/30555-beginning-with-katrina-drones-save-lives-in-disasters.html [Accessed 13 Oct. 2017]

[31] HuffPost. (2017) How *Drones Are Helping Nepal Recover From The Earthquake.* [online] Available at: https://www.huffingtonpost.com/2015/05/07/nepal-earthquake-drones_n_7232764.html [Accessed 13 Oct. 2017]

[32] Murphy, D. (2017) *Unmanned Systems and Huricane Matthew: Lessons from 2010 Haiti Earthquake << Center for Robot-assisted Search and Rescue (CRASAR) at Texas A&M Univeristy.* [online] Crasar.org Available at: http://crasar.org/2016/10/04/unmanned-systems-and-hurricane-matthew-lessons-from-2010-haiti-earthquake/ [Accessed 13 Oct. 2017]

[33] The Team at X. (2017). *Helping out in Peru – The Team at X*. [online] Available at: https://blog.x.company/helping-out-in-peru-9e5a84839fd2 [Accessed 13 Oct. 2017].

[34] Hruska, J. (2017). *Project Loon Gets Green Light to Deploy Over Puerto Rico - ExtremeTech*. [online] ExtremeTech. Available at: https://www.extremetech.com/internet/257200-google-project-loon-gets-fcc-green-light-deploy-puerto-rico [Accessed 13 Oct. 2017].

[35] J. H. Holland. “Adaptation in Natural and Artificial Systems”, in *MIT Press*, 1975.