UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

TÚLIO DAPPER E SILVA

FORMATION COORDINATION AND NETWORK MANAGEMENT OF UAV NETWORKS USING PARTICLE SWARM OPTIMIZATION AND SOFTWARE-DEFINED NETWORKING

Porto Alegre
2020

TÚLIO DAPPER E SILVA

FORMATION COORDINATION AND NETWORK MANAGEMENT OF UAV NETWORKS USING PARTICLE SWARM OPTIMIZATION AND SOFTWARE-DEFINED NETWORKING

Thesis presented to Programa de Pós-Graduação em Engenharia Elétrica of Universidade Federal do Rio Grande do Sul in partial fulfillment of the requirements for the degree of Master in Electrical Engineering.
Area: Control and Automation

ADVISOR: Prof. PhD Edison Pignaton de Freitas

Porto Alegre
2020

TÚLIO DAPPER E SILVA

FORMATION COORDINATION AND NETWORK MANAGEMENT OF UAV NETWORKS USING PARTICLE SWARM OPTIMIZATION AND SOFTWARE-DEFINED NETWORKING

This thesis was considered adequate for obtaining the degree of Master in Electrical Engineering and approved in its final form by the Advisor and the Examination Committee.

Porto Alegre, February 2020.

ACKNOWLEDGMENTS

My deep gratitude goes first to my laboratory colleagues. Antonio Silva's mentoring and encouragement have been especially valuable, and his insights launched a great part of this dissertation. Thanks also to Maik Basso, Tiago Alves, and Carlos Emygdio for their endless support, kind, and whatever the circumstances, being keen to help. My appreciation extends also to my professors, including my coordinator Edison Pignaton, to my family and friends.

ABSTRACT

In recent years, with the growth in the use of Unmanned Aerial Vehicles (UAVs), UAV-based systems have become popular in both military and civil applications. The lack of reliable communication infrastructure in these scenarios has motivated the use of UAVs to establish a network as flying nodes, also known as UAV networks. However, the high mobility degree of flying and terrestrial users may be responsible for constant changes in nodes' positioning, which makes it more challenging to guarantee their communication during the operational time. In this context, this work presents a framework solution for formation coordination and network management of UAVs, which aims to establish and maintain a set of relays units in order to provide a constant, reliable and efficient communication link among user nodes - which are performing individual or collaborative missions on its turn. Such a framework relies on a set of formation coordination algorithms - including the Particle Swarm Optimization (PSO) evolutionary algorithm -, and also considers the use of Software-defined Networking-based (SDN) communication protocol for network management. For coordination proposes, a novel particle selection criteria is proposed, which aims to guarantee network manageability of UAV formations, therefore being able to guarantee service persistence in case of nodes' failure occurrence, as well as to provide required network performance, as a consequence. Simulations performed in OMNeT++ show the efficiency of the proposed solution and prove a promising direction of the solution for accomplishing its purposes.

Keywords: UAV Network, Formation Coordination, Particle Swarm Optimization, Network Management, Software Defined Networking.

RESUMO

Em regiões de confrontos militares, em cenários pós-catástrofes naturais e, inclusive, em grandes áreas de cultivo agrícola, é comum a ausência de uma infra-estrutura pré-estabelecida de comunicação entre usuários durante a execução de uma ou mais operações eventuais. Nestes casos, Veículos Aéreos Não Tripulados (VANTs) podem ser vistos como uma alternativa para o estabelecimento de uma rede temporária durante essas missões. Para algumas aplicações, a alta mobilidade destes usuários podem trazem grandes desafios para o gerenciamento autônomo de uma estrutura de comunicação aérea, como a organização espacial dos nós roteadores e as políticas de encaminhamento de pacotes adotadas durante a operação. Tendo isso em vista, esse trabalho apresenta o estudo de uma solução que visa o estabelecimento e manutenção das conexões entre os usuários - nos quais executam tarefas individuais ou colaborativas -, através do uso de algoritmos de coordenação de formação - no qual inclui o algoritmo evolucionário Otimização por Enxame de Partículas -, e, também, de conceitos relacionados a Rede Definidas por Software para o gerenciamento da rede. Ainda, é proposto um novo critério de seleção das partículas do algoritmo evolucionário, visando garantir gerenciabilidade das topologias formadas e, consequentemente, a persistência do serviço em caso de falha dos nós roteadores, assim como o cumprimento de especificações desejadas para o desempenho da rede. Simulações em OMNeT++ mostraram a eficácia da proposta e sustentam o modelo proposto a fim de atingir seus objetivos.

Palavras-chave: Rede de VANTs, Coordenação de Formação, Otimização por Enxame de Partículas, Gerenciamento de Rede, Redes Definidas por Software.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACO	Ant Colony Optimization
AODV	Ad hoc On-Demand Distance Vector Routing
CN	Controller Node
DRL	Deep Reinforcement Learning
DRNS	Dynamical Relay Node Placement Solution
ESO	Emergent Self-Organization
FANET	Flying Ad-hoc Network
INT	In-band Network Telemetry
IWSN	Industrial Wireless Sensor Networks
LMA	Lawnmower Algorithm
LOO	Leave-One-Out
MAC	Media Access Control
MANET	Mobile Ad Hoc Networks
MOPSO	Multi-Objective Particle Swarm Optimization
MPC	Model Predictive Control
PF	Pareto Front
PS	Pareto Set
PSO	Particle Swarm Optimization
QoE	Quality of Experience
QoS	Quality of Service
RN	Relay Node
SDN	Software Defined Networking
SO	Self-Organization
TCP	Transmission Control Protocol
UAV	Unmanned Aerial Vehicle
UN	User Node
WSN	Wireless Sensor Network

CONTENTS

1 INTRODUCTION

The lack of reliable communication infrastructure in collaborative missions performed by multiple users, such as in post-disaster and warfare scenarios, is a notorious problem since decisions and actions might require shared information and a short time to be taken. In some cases, reliable and fast wireless communication among them through the network is critical, as time is precious when rescuing victims in a post-natural disaster scenario or soldiers on the battlefield support are under concern (PÓLKA; PTAK; KUZIORA, 2017; KARACA *et al.*, 2018). In the past few years, collaborative applications of Unmanned Aerial Vehicles (UAVs) have been proposed to address such demands, by the so-called UAV networks (VASHIST; JAIN, 2019; DE RANGO *et al.*, 2019; SHARMA; KUMAR, 2017; GUPTA; JAIN; VASZKUN, 2016; FADLULLAH *et al.*, 2016), due to their capabilities of sensing, processing, moving and communicating (KIM; LEE, 2018a; BEKMEZCI; SAHINGOZ; TEMEL, 2013). Particularly, nowadays warfare requires an adaptive, flexible and autonomous networks to rapidly adjust to different situations and missions (KOTT; SWAMI; WEST, 2016). In fact, UAV networks have been already considered in very concrete military scenarios to support Command and Control (C2) systems in the battlefield (CHOUDHARY; SHARMA; YOU, 2019; KOTT; ALBERTS, 2017; ORFANUS; FREITAS; ELIASSEN, 2016; BASU; REDI; SHURBANOV, 2004), and teams of UAVs have been considered to search for victims in post-disaster situations (SÁNCHEZ-GARCÍA; REINA; TORAL, 2019; REINA; TORAL; TAWFIK, 2016; SCHERER *et al.*, 2015; HAUERT *et al.*, 2008).

In both civilian and military applications, multiple and independent users - which may be terrestrial or flying nodes - perform their own or collaborative mission by exploring, sensing, and disseminating information. As they may require fast data dissemination, a well-established UAV network connection is a valuable asset in order to guarantee communication among them with high availability and reliability (ORFANUS; FREITAS; ELIASSEN, 2016). As a consequence, flying relay nodes may be considered being best allocated in order to provide desired connectivity among user nodes as long and as stable as possible. In the case of node failure leading a disconnection, the UAV network must be able to adapt itself and reestablish the connection as swiftly as possible. In addition, certain applications, such as video dissemination, require well-defined network performance which should be provided by the current network topology - *e.g.*, latency, jitter, and packet loss. In order to provide efficient communication links, the network should be able to monitor its current performance, and therefore adjust its formation in order to meet required application demands by considering nodes' redistribution, routing policy changes or even packet prioritization.

In such a scenario, finding the optimal location for placing the relay nodes in an interconnected UAV network is known to be one of the most challenging problems (KIM;

LEE, 2018b), as fundamental goals of network design should be achieved, such as scalability, availability, security and manageability (STEWART *et al.*, 2008). Firstly, scalable networks are said as those able to grow and support new applications as new users are included without impacting ongoing services already being delivered to existing users. In order to provide availability, the network should deliver consistent performance and reliable service, even in the occurrence of communication link failures. In addition, safeguarding network resources relies on planning filters and firewall features by advance and also adjusting them throughout an operation by monitoring data traffic flow. Finally, independently of its initial network design, the network manager should be able to manage and support the network. In other words, a network that does not provide functionalities - *e.g.*, communication protocol for controlling proposes - and resources, such as alternative paths and replacement nodes, may not function effectively and efficiently, considering that the network manager may not be able to correct failures occurrence or improve network performance.

Observing the described landscape, a communication protocol must sense and coordinate the behavior of multiple UAVs in order to maximize the benefits of the UAV network (CUMINO *et al.*, 2018). In this context, Software-Defined Networking (SDN) is a promising approach, since it introduces complete network programmability by separating the control plane and the data plane (WICKBOLDT *et al.*, 2015), and consequently provides flexibility towards many aspects of the network management, such as data traffic monitoring and dynamic configuration. In this context, In-band Network Telemetry (INT) is a complement solution enabled by the SDN-concept that allows a centralized entity to verify network performance using in-band telemetry. As a consequence, the SDN controller can adjust network rules and policies aiming to enhance network behavior, while not degrading its performance by using exclusively dedicated control packets. Even though these concepts were first proposed to be used in traditional networks, they having been already considered in wireless networks, including those composed by UAVs (KIRICHEK *et al.*, 2017; ZHAO *et al.*, 2018; KARAAGAC; DE POORTER; HOEBEKE, 2019; YAO *et al.*, 2018).

Although many works (ERDELJ *et al.*, 2017; KIM *et al.*, 2016; ORFANUS; FREITAS; ELIASSEN, 2016; MORAES; FREITAS, 2017; ROSATI *et al.*, 2016) have proposed the use of UAV networks to establish the connection among terrestrial and flying nodes, their proposals left room for further improvements, especially in availability and manageability (WANG *et al.*, 2017). Moreover, a careful review in the available literature reveals a lack of approaches that completely address the problem in setting up and maintaining the network connectivity of dynamic networks, such as UAV networks, without relying on either anchor nodes (*e.g.*, a base station) or less flexible network topologies, which are set up offline. ZHAO *et al.* (2018) present a realistic simulation, however with restricted network topology. On its turn, KIM; LEE (2018b) present a solution for formation coordination in a dynamic UAV network topology, however without realistic simulations. Therefore, considering the works found in the literature, there is a lack of an implementation containing both formation coordination algorithms - which considers the allocation of a controller and a set of relay nodes -, as well as a network management solution - for monitoring and controlling interests -, in order to sustain a UAV network aimed to minimize network disruptions and meet application demands of mobile users.

1.1 Objectives

This work introduces an integrated coordination and management strategy for UAV networks, which includes formation coordination algorithms and an SDN-based UAV communication protocol. The network is composed of three different types of nodes: a controller node, a set of relay nodes, and a set of user nodes. Relay nodes aim to sustain communication links among user nodes for as long as possible. Hence, the controller node is in charge of finding and setting up the best location for the relay nodes through formation coordination algorithms as part of his coordination functionalities. In addition, the controller is also responsible for monitoring network performance and configuring it throughout an operation by setting up routing policies and data packets priorities, for instance. From a SDN point of view, the maintenance of the network is performed through the exchange of both dedicated control packets and in-band telemetry. To execute this process, the controller node periodically collects contextual information from the others, as well as data packet telemetry which has been originated from the relay nodes. Having collected that information, the controller is able to determine relay nodes' best location, as well as configure routing rules or packet prioritization policies.

Firstly, this work combines formation coordination algorithms based on KIM; LEE (2018b) with a UAV communication protocol based on ZHAO *et al.* (2018) to present realistic wireless network measures in establishing communication among mobile users nodes. In order to run the simulation, while considering modeled physical constraints, the proposed coordination protocol was implemented at the network layer using the OMNeT++ simulator. The performed simulation considered a large area, such as a city, where user nodes move through the scenario. As most of the operations limit the number of relay nodes - as they introduce financial cost -, this work firstly presents the UAV network performance evaluation categorized by the number of relay nodes. Finally, this work also addresses the formation coordination problem by improving the particle selection criteria in Particle Swarm Optimization (PSO) algorithm. Instead of aiming to minimize communication link distances among user nodes, this proposal considers a multi-objective optimization in order to enhance availability and manageability by evaluating a topology solution through graph theory metrics - such as the number of components, bridges, and degree -, and imposing the virtual spring force concept - firstly introduced by (TROTTA *et al.*, 2018). To validate this proposal, simulations were also performed in OMNeT++ simulator. Throughout video transmissions among users, relay nodes were imposed to failure, therefore the controller should be able to manage the network for retrieving those connections.

1.2 Contributions

The main contributions of this work are:

- A modular framework design for UAV networks - considering the use of both formation coordination algorithms and SDN concepts for network management - aimed to establish and maintain communication links among mobile and independent nodes, also referred as users;

- A network evaluation of a complete solution, which combines a set of formation coordination algorithms (KIM; LEE, 2018b) and an SDN-based communication protocol (ZHAO *et al.*, 2018), through the use of the OMNeT++ simulator to assess network performance in terms of packet loss, latency, and connectivity.

- A novel particle selection criteria of the PSO algorithm - included in the formation construction phase -, aimed to guarantee manageability and availability by considering multiple objectives evaluated through the use of graph theory metrics and the virtual spring force concept.

1.3 Organization

The rest of this work is organized as follows. Firstly, formation coordination and network management for UAV networks, along with the PSO algorithm and the SDN concepts are explored in Chapter 2. On its turn, Chapter 3 presents related works of formation coordination algorithms which uses PSO, the SDN implementation in UAV networks, as well as the use of INT in wireless networks. In addition, the reviewed works are yet compared to this present work. In the following, Chapter 4 presents an overview of the complete solution, including the proposed architecture, system model and framework. Framework and implementation details are presented in Chapter 5, along with formation coordination algorithms and the proposed SDN-based communication protocol. Simulation results are described and discussed in Chapter 6. Finally, concluding remarks and possible future work are presented in Chapter 7.

2 BACKGROUND

The deployment and further maintenance of UAV networks rely on determining optimal relay nodes' position - also referred as formation coordination -, and configuring network policies, such as routing paths and packet prioritization - *i.e.*, network management. Firstly, the evolutionary algorithm Particle Swarm Optimization (PSO) has been already considered for solving the coordination problem in literature. In addition, Software-Defined Networking (SDN) and In-band Network Telemetry (INT) are promising tools to network management. In this chapter, basic concepts of UAV networks, PSO, SDN, and INT are presented.

2.1 UAV Networks

Over the past decade, the use of Unmanned Air Vehicles (UAVs) has been increasingly considered in a wide range of civil and military applications, mainly in those considered inappropriate or potentially hazardous to humans (e.g., surveillance, search and rescue). Moreover, a single UAV may not meet the demanded requirements of a complex mission, such as providing network infrastructure to natural-disaster survivors while searching for other victims. In this context, the use of multiple[1] UAVs are seen as necessary. Having the main objective of coordinating such flying swarm, collective and collaborative intelligence brings potential benefits (LEE; MIN; KIM, 2007; LI; SUN; XU, 2006; SAADAOUI; EL BOUANANI, 2018).

The proper operation of a team of UAVs requires a robust inter-UAV network with UAVs cooperating in keeping the network organized in the event of link or node failure (GUPTA; JAIN; VASZKUN, 2016). Most of the literature treats UAV networks as ad hoc networks - also referred to as Flying Ad Hoc Networks (FANETs). The decentralized nature of an ad hoc network implies that there is no pre-existing infrastructure, such as routers in wired networks or access points in managed wireless networks. In this context, nodes participate in the routing scheme by forwarding data, in which the determination of which nodes forward data is dynamically made based on network connectivity and the routing algorithm in use. However, some applications may require some kind of infrastructure, which may be terrestrial or aerial. In this section, features and functionalities of the UAV networks are discussed to understand its nature, constraints, and possibilities.

In the review presented by GUPTA; JAIN; VASZKUN (2016) on UAV communications, some concerns related to these applications are pointed out. Firstly, the fluid topology leads to node vanishing and link instability. Users' sessions should be maintained by transferring them seamlessly from an out of service UAV to an active path. In addition,

[1]In this Master Thesis, multiples and swarms are used as synonyms.

routing protocol scheme may not be a simple implementation of a proactive or a reactive scheme. In this context, the inter-UAV backbone has to reorganize itself when a UAV fails. Finally, the battery dependent device leads to the need of having ways of conserving energy resulting in an increased life of the network. In this way, the UAV network may manage itself having policies that reduce the consumption of energy. Comparing traditional Mobile Ad Hoc Networks (MANETs), the authors pointed out that UAV networks require changes at the Media Access Control (MAC) and network layers in order to contain self-organizing capabilities, be delay tolerant, present a more flexible and automated control through the use of SDN, and also employ energy-saving mechanisms.

2.1.1 Formation Coordination

The controller and relay nodes' deployment, including initialization and further maintenance, is referred to as formation coordination. Such a task is divided in two fundamental subproblems - *location* and *allocation*, which needs to be solved simultaneously (MIN *et al.*, 2016). The *location* problem consists of finding optimal locations for positioning networked robots while mainly maintains communication ranges and safety distances. However, many other criteria may be considered, such as minimizing the number of hops for an end-to-end transmission or enabling multi-routing possibilities for providing a manageable solution and securing from link outages. As soon as a nodes' positioning is defined, nodes should be assigned, which brings node *allocation* problem. In this task, energy consumption and travel distance may be considered.

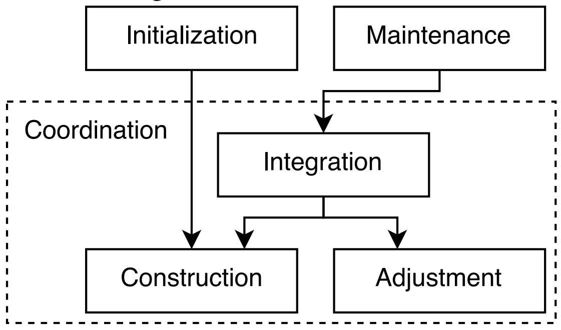
Even though recent technological advances have improved UAV capabilities, such as internal memory, processing, and energy capacity, they still present operational constrains due to the limitations of their resources. As a consequence, their use should require lighter use of computational resources, leading to consuming as fewer resources as possible (MAHMOUD *et al.*, 2016). In this context, the development of efficient algorithms for the cooperative work of UAVs becomes an important field of research (SAADAOUI; BOUANANI, 2017). The initial nodes distribution is performed during the *construction* phase. Such an algorithm is generally computationally expensive to achieve a feasible result in general. After having nodes' already deployed, there is a need to monitor the network conditions and change its positioning for its ongoing maintenance. In such a task, further maintenance should be performed more lightly by *adjustment* algorithms. At some point, the adjustment solutions may not be particularly enough to maintain a satisfying network topology performance. Aiming to evaluate the network and then determine whether to execute the construction or the adjustment method, an *integration* algorithm is generally implemented, which is capable of evaluating the network current topology and deciding which method to consider (either the *construction* or the *adjustment*).

Figure 1 depicts an overview of the coordination functionality of a UAV system. As presented, the formation coordination is composed by *initialization* and *maintenance* phases. Firstly, a *construction* algorithm is performed during startup phase. Further maintenance is performed by both *construction* and *adjustment* algorithms, which are coordinated by an *integration* algorithm.

2.1.2 Network Management

A particularity of UAV networks is the high mobility of nodes, therefore the topology formation and the routing policies of the network need to be continuously adjusted to maintain nodes' connectivity. As a consequence, communication became one of the most design issues for UAV networks, as the entire fleet relies on a reliable communica-

Figure 1 – Coordination.



Source: The author.

tion protocol for sharing information monitoring, commanding, and other tasks. Network management in wireless networks, such as for UAV networks, presents additional problems to those found in wired networks due to the unreliable nature of the transmission medium, especially wireless networks when not properly coordinated, they can interfere with the other signals, degrading the user experience.

Specifically, in ad hoc networks, in which nodes communicate between them without the need of a central infrastructure, UAVs are required to collaborate and organize themselves to relay information. In order to control a fleet of UAVs in a decentralized manner, several routing protocols were proposed in the literature, such as *dynamic source routing*, *pre-computed routing*, *on demand routing*, *flooding*, *cluster based routing*, and others. These proposals are mainly organized in three categories: *static*, *proactive* and *reactive routing protocols*. Having the main objective in standardizing functions, constructions and protocols, which were applied to ad hoc networks, the MANET WG (Working Group MANET)[3] have focused their work in developing a more robust and efficient network, such as the Ad hoc On-Demand Distance Vector Routing (AODV). The main objective of the AODV routing protocol is to be adaptive to mobile nodes, which act as routers in the network, whilst reducing bandwidth and nodes' computational use. Even though it requires the use of traditional routing tables, AODV is a reactive protocol - i.e, whenever there is a need to send data packets to an unknown destination node, the process of route discovery is initiated (BELDING-ROYER; PERKINS, 2003).

The use of a centralized entity for managing an entire network formed by UAVs has been already considered in literature (ZHAO *et al.*, 2018; ORFANUS; FREITAS; ELIASSEN, 2016). Due to the use of a unique controller as the network manager, any decision is based on a complete state view of the network, leading to optimized solutions as a consequence. In addition, the use of ad hoc protocols - such as AODV - for managing a network in a decentralized manner would require a high amount of data being transmitted among collaborative nodes for self-organizing, self-healing and determining their own routing policies. However, a centralized solution presents a point of failure - the controller -, in which its failure would compromise the network functionality. In this context, a replacement strategy might be necessary. Moreover, the adoption of a single controller compromises the network scalability, as the growing of the network may be limited due to some restrictions, such as the latency for control packet deliveries, computing capability limitations, high concentration of transmissions around the controller, which leads to energy exhaustion, and others. In this context, the use of multiple controllers in a flat or

[3]Available in: https://ietf.org/wg/manet/.

hierarchical architecture would be an alternative.

Although being initially applied to terrestrial and fixed network infrastructures, SDN technology is also enabled by using a centralized entity for controlling a fleet of UAVs. The implementation of SDN services and applications guarantees the delivery of its best performance by controlling, monitoring, configuring and tuning the network layer and switching functionality (SHARMA *et al.*, 2017). Having a global network view for management decisions - such as routing policies and packet prioritization (ZHAO *et al.*, 2018; SECINTI *et al.*, 2018) -, SDN also provides a software abstraction layer for control plane services and applications. That abstraction enables unified management of modern networks whose topology continuously evolves. As a consequence, the use of SDN might be considered in the path and channel selection, therefore contributing to reducing wireless interference and improving the usage of communication resources (GUPTA; JAIN; VASZKUN, 2016).

2.2 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a branch of Artificial Intelligence (AI) and Evolutionary Computing (EC), which was developed by Kennedy and Eberhart in 1995 (KENNEDY; EBERHART, 1995). The formulation of this algorithm was inspired by the behavior of social organisms in groups - such as representing the movement of each individual within a flock of birds or a school of fish - by emulating the interaction between members to share information (WILSON; MANTOOTH, 2013). Due to its effectiveness and rapid convergence, PSO has been applied in many areas, such as function optimizations, artificial neural network training, and fuzzy systems control (CHRIKI *et al.*, 2019).

2.2.1 Optimization Problem

The formulation of formal optimization problems in general requires the specification of three main components, which are: *decision variables*, *constrains* and *objective functions* (MAIER *et al.*, 2018). *Decision variables* are those values that can be manipulated in order to find the optimal solution for a problem. Such solution is evaluated through the use of *objective functions* - also referred as *cost functions* -, that should be maximized (or minimized). Generally, these values are restricted to a set of *constrains* previously defined. Considering $x_i = (x_1, x_2, \ldots, N)$ as *decision variables* to a problem in N-dimensional space and $f_m(x) = (f_1(x), f_2(x), f_3(x), \ldots, M)$ as a set of *objective functions*, a formal mathematical formulation is presented in (1). The minimization formulation considers constrains related to its boundaries - x_i^L and x_i^U - and also related to application trade-offs - g and h.

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f_m(x), && m = 1, 2, \ldots, M. \\ \text{subject to} \quad & g_j(x) \leq 0, && j = 1, 2, \ldots, J. \\ & h_k(x) = 0, && k = 1, 2, \ldots, K. \\ & x_i^L \leq x_i \leq x_i^U \end{aligned} \tag{1}$$

The space definition is also relevant, which are: *search*, *decision*, *feasible* and *objective* space. Firstly, the *search* space considers the range of all possible solutions in N-dimensions. The *decision* space considers the set of solutions where x are possible by applying boundaries constrains. By satisfying g and h, a *feasible* space gathers all the feasible solutions. By calculating the cost functions using the feasible solutions, the *objective* space is generated.

2.2.2 Traditional Particle Swarm Optimization Algorithm

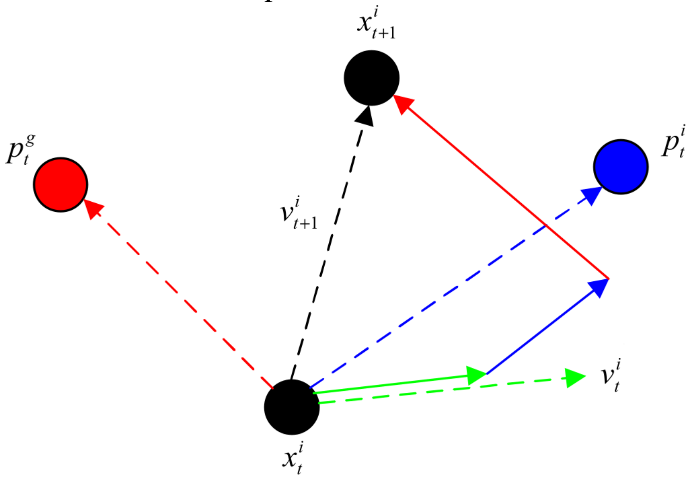
In the PSO algorithm, an iterative search is performed in order to obtain the optimal solution employing agents, called particles, whose trajectories are adjusted by a deterministic and a stochastic component. Firstly, the deterministic component is partly influenced by the particle inertial trend, by the particle's best-known solution - referred to as local best solution - and by the group's best-shared solution - known as the global best solution. As part of the stochastic component, random weights are applied in the influence of the best-known solutions. The stochastic properties of the algorithm enable solution variability, therefore guaranteeing the search space exploitation (MARTÍNEZ; CAO, 2019).

Throughout the iterative process, each particle i at iteration t is positioned in x_t^i belonged to a multidimensional space. In addition, each particle knows its best solution so far p_t^g - *i.e.*, the best local solution -, which represents its personal experience. Moreover, each particle knows the best value so far achieved by the group p_t^i - *i.e.*, the global best solution -, which represents the knowledge of how other particles have performed. Both the best local and best global solutions are selected through an evaluation of the particles' position based on already-defined evaluation functions. Such evaluation results in a fitness value of the current position - named f_t^i -, which is considered for particles' selection, as maximize or minimize optimization. At each stage of the iterative process, the position of each particle is updated based on its velocity v_t^i. As a result, the basic structure of a particle i, which is part of a selected population, is presented in the following (SHARAF; EL-GAMMAL, 2011):

- x_t^i: current solution (at iteration t) in the feasible space. The size of this vector is defined by the number of variables involved in the problem;
- v_t^i: velocity (at iteration t) for each dimension of x_t^i. This variable represents the step size of the current solution and, as a consequence, how rapidly it changes over the iterations. Having that, it is used to limit the range and resolution of the search;
- f_t^i: fitness value of x_t^i by computing problem-specific evaluation functions;
- p_t^i: local best solution, which is the best solution of particle i yet evaluated until iteration t;
- p_t^g: global best solution, which is the best solution yet evaluated by the population of particles until iteration t.

Particles' position is determined based on its current position and velocity values, as presented in (2). On its turn, the velocity is determined based on three main parts: inertial, cognitive and social components (WANG; TAN; LIU, 2018). Firstly, the inertial component is influenced by the particle's previous velocity, which means that the particle maintains its current state and conducts an inertial movement based on its velocity. The magnitude of the inertial movement is defined by w, which is the inertial weight factor. The second part depends on the distance between the current position x_t^i and its optimal position - *i.e.*, the local best solution p_t^g -, called the cognitive component, which refers to the particle's thinking, resulting from its own experience. Therefore, the parameter k_1 is called cognitive learning factor, as it is applied to the cognitive component. Finally, the third component relies on the distance between the current position x_t^i and the global

Figure 2 – Two-dimensional representation of the iterative PSO algorithm.



Source: WANG; TAN; LIU (2018).

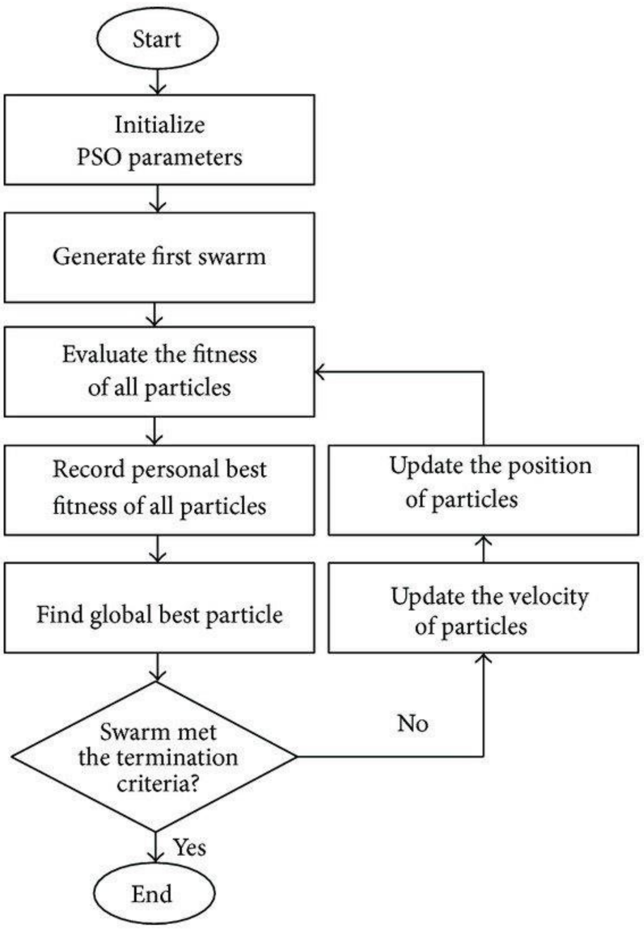
optimal position of the swarm - *i.e.*, the global best solution p_t^i -, referred to as the social component. As a consequence, information and knowledge are shared among particles and their movements are also influenced by other particles' experiences. The use of the social component stimulates the influence of high-evaluated particles and it is weighted by k_2, called the social learning factor. In order to illustrate the composition of the particles' position, Figure 2 presents a two-dimensional representation of a particle i, in which its solution is shown throughout two consecutive iterations - *i.e.*, x_t^i and x_{t+1}^i. As can be seen, the velocity v_{t+1}^i is a composition of the previous velocity v_t^i, the local best solution p_t^g and the global best solution p_t^i. Equation (3) presents how the current velocity v_t^i is determined, having that r_1 and r_2 are random values on the range [0;1].

$$x_{t+1}^i = x_t^i + v_{t+1}^i \tag{2}$$

$$v_{t+1}^i = w \cdot v_t^i + r_1 \cdot k_1 \cdot (p_t^g - x_t^i) + r_2 \cdot k_2 \cdot (p_t^i - x_t^i) \tag{3}$$

A particle is an independent agent that navigates through the search space according to the experience of its own and its companions, in which the former is the cognitive part of the particles' update formula, and the latter is the social part (WANG; TAN; LIU, 2018). Both parts play an important role in guiding the particles' search. In a single-objective PSO algorithm, choosing the appropriate cognitive and social guide (p_t^g and p_t^i) is performed through an evaluation function cost that results in a fitness cost. That value can then be used for particles' comparison and selection. In the traditional use of the PSO algorithm, Figure 3 presents a step-by-step flowchart. Firstly, PSO parameters - *e.g.*, weights, number of particles, number of iterations - are initialized, as well as the particles are generated in random positions and their velocities are set to zero. Throughout the iterations, particles are evaluated through a cost function, therefore their personal - *i.e.*, local - best solution is selected, as well as the unique global best solution. Depending on the termination criteria, which can be the number of iterations or based on the cost function values, the algorithm may end, therefore the global best solution is set as the PSO output. Otherwise, particles' velocity and position are updated - according to (3) and (2), respectively -, and the local and global best solution are selected once again.

Figure 3 – PSO flowchart.


Start
Initialize PSO parameters
Generate first swarm
Evaluate the fitness of all particles
Record personal best fitness of all particles
Find global best particle
Swarm met the termination criteria?
No
Yes
End
Update the velocity of particles
Update the position of particles

Source: OUDIRA; DJOUANE; GARAH (2019).

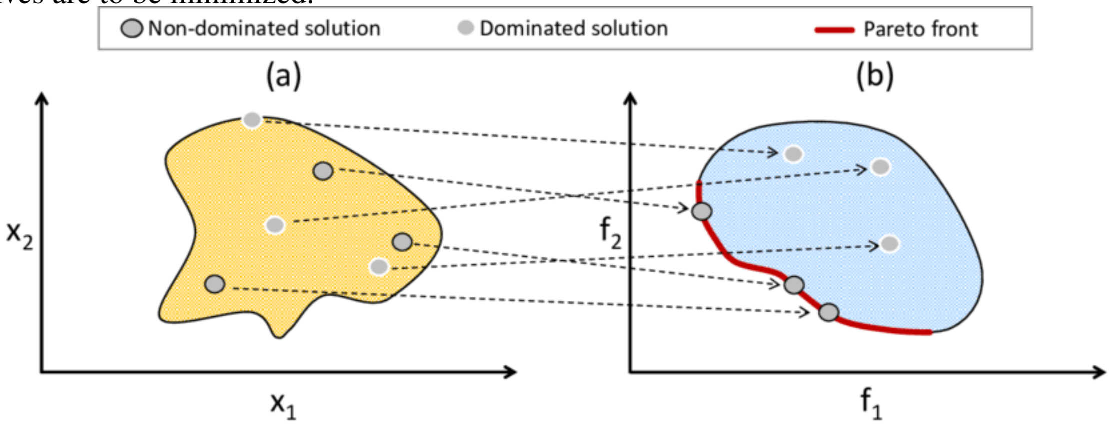
2.2.3 Multi-objective Optimization

The use of more than one assessment leads to the problem of multi-objective optimization and, as a consequence, to *non-dominant* and *dominant* solutions. Having two vectors belonged to the feasible space x_A and x_B - which comprehends to feasible solutions -, x_A is said to dominate x_B, if: every objective variable of x_A is less or equal than x_B (considering a minimize optimization) and at least one of them overcomes. Therefore, the following conditions are satisfied (LUO *et al.*, 2020):

$$\begin{aligned} f_m(x_A) &\leq f_m(x_B), \quad m = 1, 2, \ldots, M \\ f_j(x_A) &< f_j(x_B), \quad\ \ j \in \{1, 2, \ldots, M\} \end{aligned} \tag{4}$$

A vector x' belonged to the feasible space is said to be *non-dominant* - or a *Pareto optimal* solution -, if there is no other known solution that dominates x'. In addition, the set of all *Pareto optimal* solutions is called the *Pareto set* (PS). As a consequence, the solutions included in the PS are *non-dominant*. Having the PS plotted in the objective space, these solutions are collectively known as the *Pareto front* (PF). In Figure 4, the conversion from the feasible space (at the left side) to the objective space (at the right side) is presented. As can be seen in the figure, both sets of *non-dominant* and *dominant* solutions are included in the objective space, as well as the set of *dominant* solutions forms the PF.

Figure 4 – Mapping of (a) a feasible space onto (b) an objective space, where both objectives are to be minimized.



Source: MAIER *et al.* (2018).

Multi-objective optimization is composed of multiple target functions. Unfortunately, due to conflicting objectives, it is generally impossible to achieve a perfect solution by optimizing all objectives (a dominant solution). Therefore, only Pareto optimal solution can be found. In PSO algorithms, a multiple-target implementation is generally applied by giving a different weight to each objective function, then combining them. Although it is possible to obtain a group optimal solutions, a more convenient method would be to simultaneously obtain a group of Pareto optimal solutions. In the Multi-Objective PSO (MOPSO), the selection of the cognitive guide conforms to the same rule as the traditional PSO algorithm, despite the guide should be determined under Pareto dominance. On the other hand, the selection of the social guide is performed through the use of a candidate pool, which stores more Pareto optimal solutions. The selection of the guide is then performed in such a way that it provides effective guidance for the particles to obtain a better convergence speed and provides balanced search along the Pareto frontier, to maintain the diversity of the population.

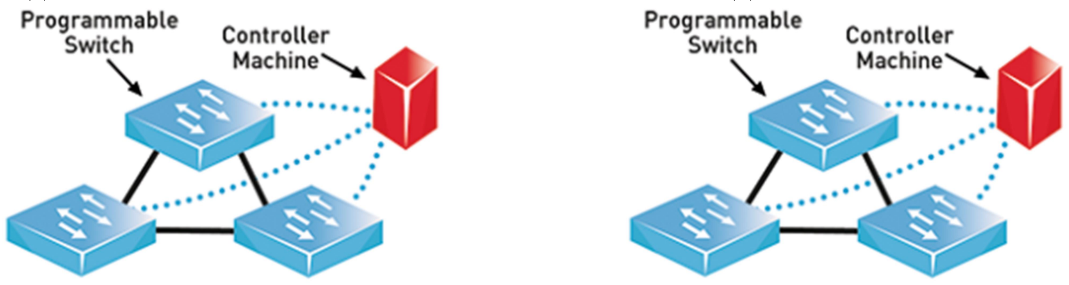
2.2.4 Discrete Optimization

In general, continuous optimization problems with smooth objective functions are easier to solve (SCALES, 2006), as the optimizer can be driven using the differential or derivative information of the objective function. One way of dealing with discrete objective functions is to preferably search for best solutions in terms of one singular goal (look for a specific flat region), and when several solutions are non-dominant in terms of the preferable discrete objective evaluation, a continuous metric is optimized.

However, a discrete optimization divides the search space into flat regions, wherein differential information remains null. Therefore, a direct consequence of using several objective functions to drive the global process is that the resulting optimization surface presents discontinuities in regions of search, which makes the optimization harder (DENGIZ; KONAK; SMITH, 2011). Among different flat regions, several solutions obtain the same value of the objective function, while they may not be equally adequate.

Figure 5 – The architecture of traditional network and SDN.

(a) Traditional Network Architecture. (b) SDN Architecture.



Source: TANG (2019).

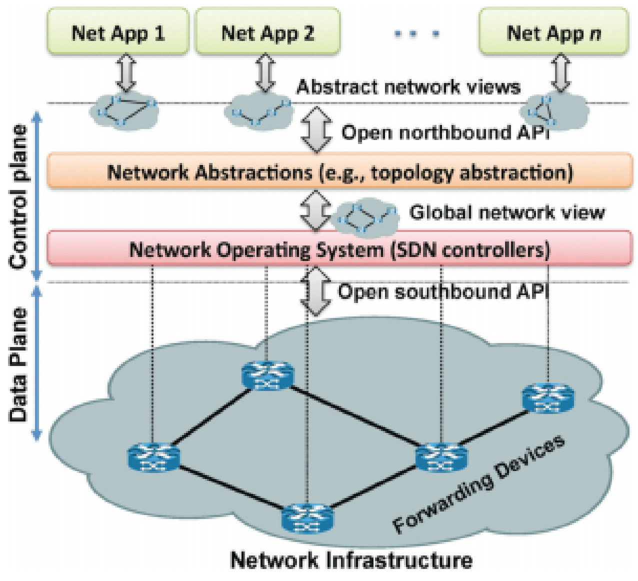
2.3 Software-Defined Networking

One of the main requirements for the modern use of UAV networks is to support tactical operations in areas without an already-built infrastructure (POULARAKIS; IOSIFIDIS; TASSIULAS, 2018). In addition, an efficient autonomous system is commonly high dependent on a communication architecture that facilitates traffic control, which means the ability to set up links to be utilized by selected data flow at a chosen time (LANDMARK; LARSEN; KURE, 2018). Otherwise, available resources might not be utilized appropriately. Mobile Ad Hoc Networks (MANETs) has been long considered as the ultimate solution to achieve such objective, as envisioned as a fully decentralized system presenting self-organizing capabilities. Although MANETs have been currently deployed at the edge, it suffers from complex configuration requirements and protocol overhead due to constant network formation change (POULARAKIS; IOSIFIDIS; TASSIULAS, 2018).

In this context, Software-Defined Networks (SDN) is a new paradigm that facilitates traffic control through the benefits of decoupling the control and the forwarding plane (LANDMARK; LARSEN; KURE, 2018). The control plane (which contains the functionalities of handling network traffic) and the data plane (which contains the functionalities of forwarding traffic according to the rules that the control plane assigned) are traditionally part of network devices (routers and switches), as presented in Figure 5(a). As a consequence, the network configuration is an exhaustive and complex process, as the network administrator has to manually configure individual network devices (TANG, 2019). SDN has become one of the most popular ways of deploying network applications at a faster rate and reduced overall cost (KEARY, 2018). In contrast to the traditional networks that rely on physical infrastructures - such as switches and routers to manage their own connections -, SDN is an approach to networking that uses open protocols - *e.g.*, OpenFlow - to control, manage and provision services from a centralized entity at the edge of the network. As a consequence, SDN physically separates the network control plane from the data plane, therefore a programmable controller can control several devices (TANG, 2019), which is shown in Figure 5(b).

Decoupling the control plane logic from routers and switches - which are traditionally in charge of defining their own data traffic policies -, simplify those devices, while the control logic is implemented in a centralized controller (Kreutz *et al.*, 2015). The separation can be accomplished through a well-defined programming interface between the switches and the SDN controller. Figure 6 illustrates the SDN architecture. The described separation between data and control plane can be observed. As can be seen, the data plane is composed basically by forwarding devices merely in charge of forwarding

Figure 6 – SDN architecture.



Source: Kreutz *et al.* (2015).

incoming data packets guided by a given set of instructions. On its turn, the Network Operating System (performed in the SDN controller) is able to dictate data flow through an open southbound API (*e.g.*, the OpenFlow protocol), enabling data flow monitoring, receiving devices' requests and setting devices' forwarding rules. Having collected data flow information from the forwarding devices, the SDN controller contains a global network view. That view - or part of it - is then shared among a diverse set of network applications through a northbound API, in order to facilitate maintenance and insertion of new applications or functionalities.

As mentioned, each switch maintains a set of entries in a table (flow table, or routing table) that specifies how to handle incoming or outgoing packets. In addition, those entries are filled by an external entity - the SDN controller. Each entry contains two main information, which are: action and rules. Firstly, the action specifies how to deal with a packet - *e.g.*, forward to a specific port -, and the rule specifies a set of criteria that a packet must meet for the relative action to be executed - *e.g.*, the destination of the packet. A group of packets that match the same rules is called a stream.

The SDN has four main pillars, in which its architecture is defined:

- Control and data plane decoupling, as control plane functionalities are removed from network devices, which became simple traffic routing devices.
- Forwarding policies are based on incoming packet information, and instructions are simply followed. As a consequence, every incoming packet is treated equally.
- Maintenance and insertion of programmable network applications are facilitated in the SDN controller, which is able to interact with devices' data plane on its turn.
- Control logic becomes SDN controllers' responsibility, as an external entity.

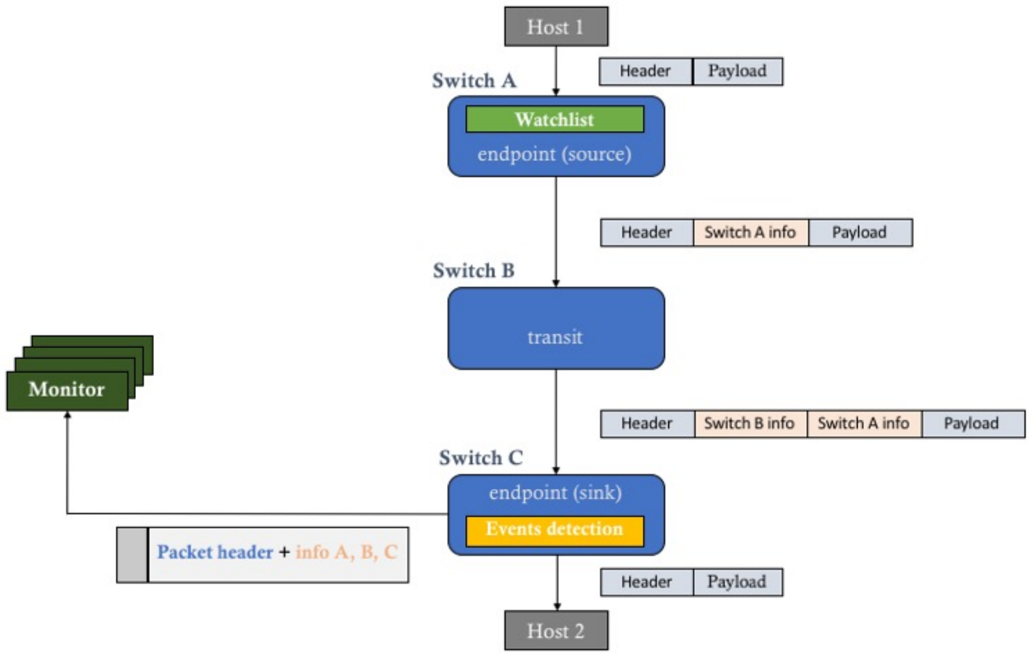
In order to standardize communication between the controller and the network devices, the OpenFlow communication protocol emerged as the primary means of communication between the controller and switches (MCKEOWN *et al.*, 2008). OpenFlow enables to control the flow of each network traffic - determining the routes that such packets should follow and specifying any sort of processing that should also be performed. In this way, it becomes possible to adjust network behavior dynamically without affecting ongoing traffic. As a consequence, the OpenFlow protocol interface is the standardization of the messages which are sent by the controller to the switch over a secure channel to define packet forwarding policies. Therefore, the SDN controller is responsible for configuring and managing the forwarding devices by monitoring data flow and receiving their requests through this interface, as well as adding, updating, or deleting flow entries from the forwarding devices' routing table. In other words, it is a set of possible instructions used to control the data plane generically; therefore, managing any network hardware which is enabled by the OpenFlow standard. That can be operated reactively (in response to a packet arrival) or proactively. Considering the case in which a forwarding device loses contact with the SDN controller, it should attempt to contact one or more backup controllers. After a certain number of failed attempts, the forwarding device should enter an emergency mode and immediately reset any current connection.

The OpenFlow protocol defines three groups of messages: Controller-to-switch, Symmetric, and Asynchronous. Firstly, the Controller-to-switch messages are initiated by the controller and used to directly manage or inspect the state of a forwarding device. On its turn, Symmetric messages are periodic and initiated by either a forwarding device or the controller, being sent without a request, such as Hello messages - which are exchanged between a forwarding device and the controller upon connection setup. Finally, Asynchronous messages are initiated by a forwarding device and used to update the controller about network events and switch state changes. Therefore, forwarding devices send asynchronous messages to the controller to denote a packet arrival, switch state changes, or an error. For instance, for each incoming packet that does not match any rule contained in the routing table, a Packet-In message is sent to the SDN controller. If the forwarding device has sufficient memory to buffer those incoming packets - while a request is being sent to the SDN controller -, the Packet-In message would merely contain a fraction of the incoming packet header. Otherwise, forwarding devices that do not support internal buffering should transmit the whole incoming packet to the SDN controller as part of the Packet-In message. As soon as the related instruction is received, the forwarding device is then able to forward the original packet following demanded actions.

2.3.1 In-band Network Telemetry

In recent years, network applications have required a deterministic and real-time network service with latency and reliability guaranteed (GROSSMAN, 2018), such as video dissemination (ZHAO *et al.*, 2018). In order to accomplish a persistent and problem-free operation, it is critical to have visibility and awareness of what is happening on the network at any moment (KARAAGAC; DE POORTER; HOEBEKE, 2019). As a consequence, high-level detailed network monitoring (*e.g.*, per-packet hop-by-hop delays, instantaneous queue size) has become crucial to correctly identify, characterize and manage faults, performance and security issues (MARQUES *et al.*, 2019). Traditional network monitoring relies on tools, such as *ping* and *zing*, that are used to determine traffic latency and end-to-end reliability. These approaches introduce additional control traffic that can occupy extensive network resources, impact network behavior and interfere with the scheduled data flow. As a consequence, such approaches have limited applicability in constrained and dynamic networks due to their static and inefficient design (KARAAGAC; DE POORTER; HOEBEKE, 2019).

Figure 7 – INT workflow.



Source: PROGRAMMERSOUGHT (2019).

In-band Network Telemetry (INT) is a new approach that offers low overhead monitoring possibilities as an innovative service that is supported by the programmability of data plane (HANDIGOL *et al.*, 2014). In other words, it takes the opportunity of defining custom headers for telemetry proposes - *e.g.*, queue occupancy, processing time, policy rules - in data traffic packets (KIM *et al.*, 2015). Such information, usually referred to as metadata, is encapsulated and accumulated in a data packet along its path. Having metadata carried within data packets - rather than being sent in dedicated packets -, is considered an *in-band* monitoring. At some point, however, *out-of-band* or active telemetry is necessary, as the metadata should be extracted and dedicated packets would be used to report a monitoring control entity, while the original data package is delivered to the recipient (THOMAS; LAUPKHOV, 2016; MARQUES *et al.*, 2019).

In order to illustrate the INT workflow, Figure 7 demonstrates both *in-band* and *out-of-band* telemetry in use. The figure presents a set of two hosts (Host 1 and Host 2), three switches (Switch A, Switch B, and Switch C) and one monitoring control entity (Monitor). From Host 1 to Host 2, there is a packet stream relayed by the three switches (Switches A, B, and C), in which accumulates their telemetry information along with the relayed data packet - *i.e.*, *in-band* - through metadata fields, such as "Switch A Info" and "Switch B Info". Arriving at the last switch (Switch C), the metadata is extracted from the incoming data packet, and transmitted to the Monitor through a dedicated packet - *i.e.*, *out-of-band* - along with the data packet header. Finally, the incoming data packet, in which the metadata has been removed, is forwarded to the destination.

3 RELATED WORK

3.1 PSO-based UAV Formation Coordination

In the military domain, ground nodes may be partitioned due to the mission requirements. A fixed placement strategy of the communication structure is not wise as it would be probably be attacked and yet would not be able to serve marching troops. BASU; REDI; SHURBANOV (2004) proposed the use of UAVs which obey local flocking rules (like birds and insects) to adapt themselves to the motion of ground nodes and therefore maintain high connectivity among them. Hence, the authors focused their work to achieve a network connection among moving ground nodes in a distributed manner through a FANET. In their work, the authors aim to minimize the number of disconnected ground nodes, as well as minimize the standard deviation in the number of connected nodes by UAV. By the end, they have shown efficacy in using this approach rather than a static structure in a scenario which troops are marching. On its turn, ORFANUS; FREITAS; ELIASSEN (2016) proposed the use of Self-Organization (SO) paradigm to design a robust and efficient UAV relay network in order to support military operations, such as reconnaissance and area coverage. For robustness, they used Emergent Self-Organization (ESO), which is a decentralized type of SO. Simultaneously with ESO, they implemented a feedback-based system which applied positive feedback to expand area coverage and negative feedback to maintain nodes' connectivity. Their work considered the scenario where area coverage is the main goal, instead of keeping connectivity among nodes performing unknown missions.

Focused on the Wireless Sensor Networks (WSN) context formed by static nodes, MAGÁN-CARRIÓN *et al.* (2016) proposed a three-stage relay node placement strategy designed to maximize connectivity by measuring throughput and inter-node reachability. The three steps are: firstly, a set of potential locations for the relay nodes are identified (which may be higher in number than actually available). Then, the best locations are selected based on the Leave-One-Out (LOO) approach, which focuses on optimizing reachability. Finally, the solution is optimized in terms of throughput by using a PSO-based algorithm. In the simulation analyses, they demonstrated to have proposed an efficient method, although demanding long execution time. Regarding (MAGÁN-CARRIÓN *et al.*, 2016), MAGÁN-CARRIÓN *et al.* (2017) continued focusing on the network deployment - *i.e.*, distribution of relay nodes -, however considering they capable of self-relocating. Having included such a feature, they proposed a multi-stage Dynamical RN placement Solution (DRNS), based on PSO algorithms and Model Predictive Control (MPC) techniques. Following a bi-objective optimization procedure, both network connectivity and throughput were jointly maximized.

Applications, such as surveillance operations, might require that information should be transmitted from a mobile device to the base station throughout the execution of the task. Having constraints related to the wireless transmission ranges and physical obstacles in the scene, relay chains formed by intermediate UAVs may be necessary. BURDAKOV *et al.* (2010) focused their work on presenting a solution by maximizing transmission quality given a known target position and minimizing the number of relay nodes which are required. They presented two new algorithms: one uses label-correcting graph search to efficiently generate a set of optimal relay chains solutions - in which reveals a trade-off between the number of UAVs and the quality of the chain -, allowing ground operators to choose among them; the second uses a dual ascent technique to generate high-quality relay chain given a limited number of UAVs.

SÁNCHEZ-GARCÍA; REINA; TORAL (2019) aimed their work to use a team of UAVs in post-disaster scenarios, where there is a need to effectively search for victims and provide them further assistance with data link communication. They proposed a distributed Particle Swarm Optimization (PSO) algorithm to perform the exploration mission by having the UAVs sharing their best candidate solutions with their neighbor nodes. Moreover, they presented a characterization of the proposed algorithm by using a different set of parameters' values. In comparison with a trajectory planning algorithm that sweeps the entire area - the Lawnmower Algorithm (LMA) -, their solution was able to find 25%, 50% and 75% of the victims faster than the LMA. In addition, they pointed out that their proposed algorithm presented a higher number of connections and a smaller amount of elapsed time among them in comparison to the LMA.

SHAKHATREH *et al.* (2017) proposed a PSO-based algorithm for efficient UAV positioning, thus minimizing total transmission power. In the solution proposed by the authors, only one UAV was considered, and two distinct cases were also considered for utility function analysis. In the first case, only the minimum transmission power was considered. In the second case, the asymmetry of the dimensions of each floor was considered, and a gradient algorithm was used to determine the efficient positioning of the UAV. In this solution, the authors concluded that the PSO-based algorithm would converge to the efficient positioning of the 3D UAV when the maximum number of iterations is equal to 50. On the other hand, it was also found that the gradient descent algorithm will converge to the efficient positioning when the maximum number of iterations is equal to 100. This study showed that an optimization of the PSO algorithm is necessary for an application in a multiple UAV scenario.

KIM; LEE (2018b) proposed a formation coordination methodology in order to sustain a UAV formation and therefore establish communication links between Mobile Units (MU) and their ground station, by using a set of Relay Units (RU). Through the execution of three algorithms - construction, adjustment, and integration -, they presented the efficacy of their approach in terms of the resultant distances among nodes. The strategy begins with the construction algorithm to determine a starting point for each node, which is computed based on Particle Swarm Optimization (PSO) approach. Consequently, this procedure is computationally overwhelming and moreover the result may differ significantly from the current nodes' location. In order to keep the connectivity of the network after its construction, a Gradient function algorithm is used to adjust the current formation. On its turn, this procedure is much lighter in terms of computational effort; however, at some point, the controller must be able to detect the lack of its efficiency. In this case, the formation is completely redesigned applying the construction algorithm, which may completely differ from the current one. Finally, the authors implemented a method to integrate both construction and adjustment procedures.

3.2 SDN-based UAV Network Management

ZHAO *et al.* (2018) and KIRICHEK *et al.* (2017) presented SDN-like solutions for UAV networks. Firstly, ZHAO *et al.* (2018) considered the allocation of relay nodes aiming to establish a communication link between a source and a destination node for video transmission. To achieve this goal, the controller node periodically gathers relevant information from each node, such as the location and the remaining energy. As soon as an event is identified, the controller selects and places relay and source nodes in strategical locations. After settling the first formation by managing their positions and filling their routing tables, the SDN controller keeps monitoring the condition of the nodes and may replace them if they inform low residual energy. Similarly, KIRICHEK *et al.* (2017) also proposed an SDN-based solution for UAV networks in order to manage a group of UAVs. Flying nodes are used simply as switches in order to settle communication links between terrestrial segments and a flying data collector. An additional node is considered as the SDN controller to update the nodes' routing tables. Their work showed the efficiency of the SDN-based solution through simulation.

SECINTI *et al.* (2018) presented a management strategy for UAV networks based on SDNs. In the proposed solution, the Openflow v1.5 protocol is used for the communication of the control plan. During operation, the controller acquires the location and channel availability information, generates the routing table with the help of the routing algorithm, and forwards the routing table to the relay nodes. In the proposal, the authors do not address issues of instantaneous updates. Therefore, the solution cannot be employed in a real environment where the relay node moves to ensure greater coverage of the area. Moreover, it does not provide replacement of the relay in operation time, which usually occurs, due to the need for battery charge, failure of operation or maintenance.

3.2.1 In-band Network Telemetry in Wireless Networks

The separation of the control plane and the forward plane conceived by the SDN approach have simplified the management of large and dynamic networks. However, the manual configuration of forwarding strategies remains a challenge operation to be performed. In their work, YAO *et al.* (2018) proposed an autonomous control-loop architecture, named NetworkAI, which aims to solve real-time large scale network control problems with any assumption of the underlying network. In doing so, their proposal englobes the use of In-band Network Telemetry (INT) for network monitoring and Deep Reinforcement Learning (DRL) algorithms for learning and applying decisions with no prior knowledge of the network task. The authors considered the collection of both network and traffic information. Firstly, the network information, which is contained in data packets using INT and is related to the status of the network device, such as network physical topology, hop latency and queue occupancy. On its turn, traffic information includes service-level data - e.g., Quality of Service (QoS) and Quality of Experience (QoE) - and anomaly traffic detection information. In order to transmit traffic information, some nodes are in charge of transferring raw traffic data to the network analytical plane. To demonstrate the feasibility of their work, they have presented a QoS-routing use case, as well as preliminary experimental evidence. In the end, the authors have presented some opening challenges, such as the communication overhead for retrieving and issuing data in an SDN architecture, a large amount of training cost, which weakens the flexibility of NetworkIA, and the need for implementing a test-bed for real network environment experimentation.

Industry 4.0 has emerged the need for flexible and reconfigurable network solutions which enables powerful monitoring and management functionalities. To accomplish it, the Industrial Wireless Sensor Networks (IWSN) requires high visibility of what is happening at any time. However, traditional probing or polling-based network monitoring approaches for IP networks present static and inefficient design, as to introduce control traffic which occupies network resources, impacting network behavior and interfering scheduled application traffic flow. In their work, KARAAGAC; DE POORTER; HOEBEKE (2019) proposed a novel efficient network monitoring and telemetry solution for IWSN by developing a flexible and powerful INT with minimal resource consumption and communication overhead. By adapting the recent use of INT to wired networks, INT enabled the collection and reporting of the network state through data plane, without depending on extra control packets. In addition, the network state is objected at the exact point that real user traffic is being transmitted. Unlike existing in-band monitoring technologies, the authors proposed both uplink and downlink telemetry in INT design, allowing continuous, periodic, event-driven or query-driven INT initialization.

3.3 Work Comparison

The main characteristics of existing works related to formation coordination and management problem are presented in Table 1. Based on the analysis of the state-of-the-art, it is possible to conclude that there is a lack in the literature for realistic simulations of Centralized Self-Organizing (CSO) UAV networks for accomplishing undetermined missions - carried by multiple and mobile independent nodes -, and which also include the controller in the network as a node. Therefore, the controller node must communicate with the others through a multi-hop wireless link. Moreover, there is a lack of network performance evaluations, such as packet loss and latency, in works that also propose solutions for the relay node placement problem. The coordination protocol, which considers both the control plane protocol and the formation management algorithms, must coordinate a fleet of UAVs aimed to avoid connection losses in a dynamic scenario. Very few works describe the communication protocol - including packet types and parameters - required over the control plane in order to perform such a task. Thus, so far, not all of these key features have been provided in a unified cooperative UAV scheme for enhancing the military and civil applications.

Besides, considering recent literature, several authors proposed an enhanced PSO algorithm combined with an adaptive decision-maker for UAV formation coordination, and the results of the comparison illustrate the advantages of the proposed strategy. Moreover, some researchers also focused on PSO analysis and technical improvements. However, few studies show robustness in their solutions, capable of actually considering the UAV constraints and network management capabilities. Although this work also presents a PSO particle selection approach, it is intended to provide a persistent network service tolerant to failures and enabling network performance improvements by evaluating manageability over the topology solutions.

Table 1 – Summary of formation coordination and network management proposals.

Reference	Addressed Problem	Proposal	Centralized vs. Distributed	Mobile User Nodes	One-hop vs. Multi-hop to Controller	Network Performance Evaluation	Control Plane Communication Protocol
SÁNCHEZ-GARCÍA; REINA; TORAL (2019)	Discover and provide assistance to victims in a post-disaster scenario	UAV team follows a distributed PSO-based exploration algorithm	Distributed	No	Not applicable	No	No
BASU; REDI; SHURBANOV (2004)	UAV placement and navigation strategies to provide connection among marching ground nodes	Adaption to the motion of ground nodes using local flocking rules	Distributed	Yes	Not applicable	No	No
MAGÁN-CARRIÓN *et al.* (2016)	Relay node placement for static scenarios	Three-stage placement procedure based on PSO and LOO algorithms	Centralized	No	One-hop	No	No
BURDAKOV *et al.* (2010)	Relay positioning for providing connection between UAV and base station with given obstacles	Label-correcting and dual ascent algorithms for relay chain generations	Centralized	No	One-hop	No	No
KIM; LEE (2018b)	FANET topology management for adapting to frequent and rapid fluctuations	Topology management based on construction and adjustment algorithms	Centralized	Yes	Not applicable	No	No
MAGÁN-CARRIÓN *et al.* (2017)	Relay node placement for dynamic scenarios	Multi-stage placement solution based on PSO and MPC techniques	Centralized	Yes	One-hop	No	Partial
ZHAO *et al.* (2018)	Management of UAV network to guarantee satisfactory video quality	Software-defined UAV networking architecture (SD-UAVNet)	Centralized	No	Two-hop	Yes	Yes
STFANET This Proposal	FANET formation coordination for providing connection among mobile and independent nodes	SDN UAV controller performing formation coordination algorithms	Centralized	Yes	Multi-hop	Yes	Yes

4 PROPOSED APPROACH OVERVIEW

In this chapter, the architecture is presented by introducing and describing nodes functionalities and capabilities. In order to properly address the problem, a System Model is described. That introduces the proposed approach of this work by presenting network functionalities and their integration, therefore being basis to a modular Framework also proposed in the following. On its turn, the framework depicts the way that further implementation is structured.

4.1 Architecture

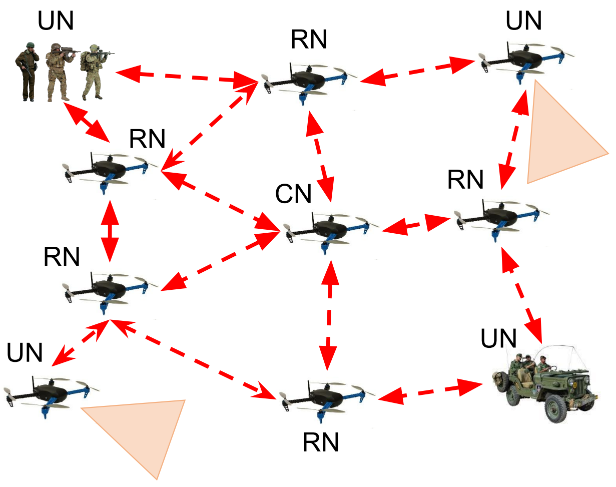
The architecture is composed of three types of nodes, namely, one Controller Node (CN), a set of Relay Nodes (RN) and a set of User Nodes (UN), as shown in Figure 8. The UNs perform an arbitrary mission - such as the search for points of interest (MORAES; FREITAS, 2017) - and thus their positions are defined exclusively by themselves. On its turn, the RNs must be allocated in order to provide the best possible link availability among the UNs. The coordination and formation management are centralized and performed by a flying node referred to as the CN. It periodically receives information from all the other nodes of the network, which contains contextual information, such as their position, trajectory, and speed. In this way, the CN is responsible for positioning the RNs and itself, and also for setting the routing tables of each node. By doing this, the CN aims to provide the connectivity among the UNs, as well as between itself and all the other nodes. In the figure, the arrows represents the routes set by the controller node, which defines both data and control packet routing tables.

In this work, some constrains were applied. Firstly, both user and relay nodes were considered to dislocate in a 2D dimension space without obstacles due to the restrictions imposed by the network simulator. In addition, no battery life restriction were imposed to the UAV nodes. The descriptions of these three types of nodes, as well as their internal functionalities, are presented in the following:

- The RN is responsible for forwarding packets and, as a consequence, establishing the connection among UNs (for data packet transmissions) and between all the nodes and the CN (for control packet transmissions on its turn). Every RN must periodically send its current state to the CN, so it is able to determine the positioning and routing scheme to extend or, at least, preserve the network connection;

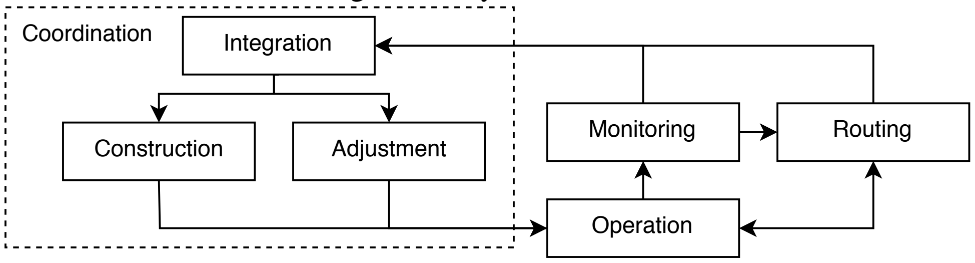
- The CN is primarily responsible for monitoring nodes' locations, as well as setting up and maintaining the entire network. Hence, the CN updates the routing tables of every node (UNs and RNs) and also determines the position of the RNs. Although

Figure 8 – The use of the proposed architecture in a military context.



Source: The author.

Figure 9 – System model.



Source: The author.

the CN could serve as a relay node, it acts exclusively in collecting and transmitting control packets in order to focus its processing and radio capabilities in the coordination functionality;

- The UN is performing a mission which is unknown from the perspective of the network control. Hence, its location is not determined by the CN, however by the application itself. Similarly to the RNs, every UN must periodically provide its current state to the CN.

4.2 System Model

As already mentioned, the CN is responsible for setting up and maintaining a reliable and persistent network infrastructure having the UNs as dynamic users and using a limited amount of RNs as mobile switches. In this context, formation coordination and network monitoring become crucial assets. A complete overview of this proposal is presented in Figure 9, referred to as the system model.

An initialization procedure provides a first UAV formation by using a Construction algorithm. To perform such a task, the fundamental goals of the network design should be addressed properly. In the context of this work, manageability criteria should be achieved by having multiple paths between any source and destination node. Therefore, availability is also pursued as the network manager (*i.e.*, the CN) would be capable of reestablishing ongoing transmission - in case of nodes' failure or link congestion - without relying on nodes' reallocation, in which would require long recuperation time.

Throughout the operation, current topology formation is also being constantly monitored by dedicated control messages in a Monitoring application. Therefore, the mobility of the UNs are logged and their current position is estimated, in order to be used by the CN to compute RNs' location aiming to maintain nodes' connectivity. In addition, frequent changes in routing policies are not desired, as requires dedicated control messages transmitted by the CN to the nodes. As a consequence, the less amount of changes is desired in a new topological formation, therefore current routing policies could be also considered in the Integration algorithm. Finally, network performance would also be considered in order to prioritize certain transmissions that may not be attending properly. In other words, having network performance data along with current topology formation information and routing policies, the CN in charge might decide whether to change UAV formation - applying an Integration of construction and adjustment algorithm -, therefore coordinating the fleet.

In the reason of adopting the use of SDN-concept along with in-band network monitoring, routing policies are updated in order to improve network performance and meet desired application requirements. As a consequence, a Monitoring application would also evaluate such network metrics - *i.e.*, packet losses and latency for end-to-end assessment, along with jitter, queue size and packet drops for hop-by-hop evaluation - in order to provide appropriate video quality transmission, for example. Moreover, security issues would also be accomplished, as data packet flow would be constantly monitored and security policies could be adjusted as needed.

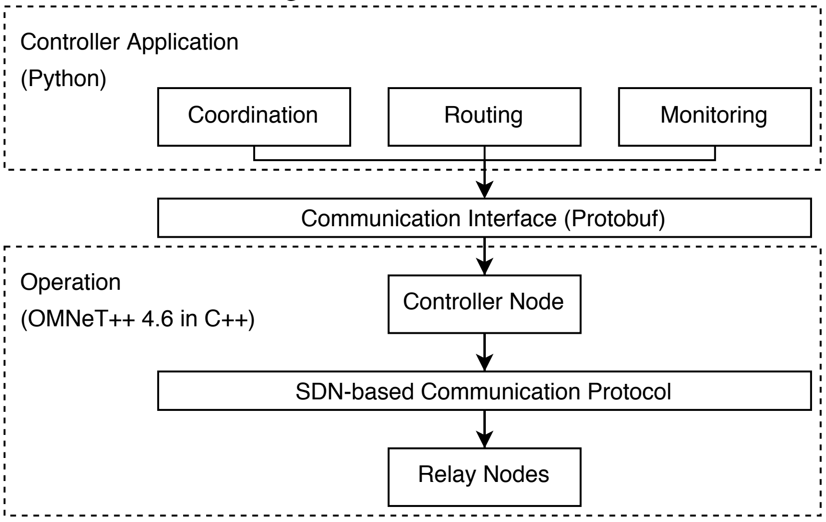
As routing policies are exclusively defined by the CN, the RNs should be able to request the CN an instruction related to relying on a data packet, such as the next hop or whether to drop it. On its turn, the CN should also be able to respond to that specific node with a routing table update, as well as the other nodes that are currently affected. As a consequence, a Routing application receives a nodes' request and transmit routing table updates by computing routing table rules based on a complete view of the network. In addition, the current routing policies are also considered in the Integration algorithm, due to a need of minimizing the number of routing changes - therefore, reducing the need for routing table updates - in the case of changing nodes' formation.

In this work, an Integration algorithm - including Construction and Adjustment - along with Routing and Monitoring are firstly addressed by combining proposals found in the literature. In the following, a novel evaluation of the topology formation is presented by using graph theory definitions - *e.g.*, the number of connected components and bridges - as network evaluation, as well as the virtual spring force concept. In addition, multiple goals are aimed to be achieved - and some are discrete metrics -, such as maximizing the number of alternative routes, minimizing the number of points of failure, maintaining safety distances among nodes, as well as proper link distances for link quality. As a consequence, such a problem deals with multi-objective and discrete optimization.

4.3 Framework

In order to implement the system model for further simulation, a framework is proposed, as presented in Figure 10. In this framework, two developing environments were considered, which are the Controller Application - in this work, implemented in Python language - and the Operation - on its turn, C++. Firstly, the Controller Application holds the set of coordination and management functionalities considered as part of the CN. On

Figure 10 – Framework.



Source: The author.

its turn, the Operation implements the communication among wireless nodes, considering realistic imperfections, such as delay, wireless interference, and others.

As mentioned earlier, one of the advantages of using the SDN concept is to separate the control plane functionalities from the switches and introduce them to a central unity. Some of these functionalities are then part of applications that runs at the CN, as defining routing policies, for instance. In order to implement all the functionalities that the CN application, the Controller Application module contains a set of SDN applications that are performed by the CN, such as Coordination, Monitoring, and Routing. Together, those sets of applications continuously identify, classify and command topology formation changes as needed. In addition, some of those modules share valuable information in order to best perform their tasks, as shown in the system model.

A northbound protocol (Communication Interface) was implemented by using Protobuf library[1]. Through the use of such a library, the Controller Application and the Operation are completely apart from each other. In addition, a southbound protocol (SDN-based Communication Protocol) was developed by considering the use of a set of SDN control messages types in order to manage and monitor network behavior.

In order to perform network evaluation of the SDN-based proposed architecture and system model, an operational data flow containing a set of nodes (CN, RNs and UNs) needs to be implemented, which has been named Operation. This implementation takes advantage of the Mobile Multimedia Wireless Sensor Network (M3WSN), which is an OMNeT++ framework developed over Castalia (ROSÁRIO *et al.*, 2013). OMNeT++ is a network simulator that has been widely used for implementing and testing novel solutions. Therefore, by using this framework, it is possible to gather valuable results considering the already modeled physical layer constraints. The CN is in charge of receiving and transmitting control packets through a wireless media from CNs and RNs. The control packet flow follows an SDN-based communication protocol in order to monitor and manage UAV formation and routing policies. Also, UNs are capable of transmitting data packets (*e.g.*, video dissemination) among each other relying on RNs as switches, in which contain a set of routing tables that are constantly updated throughout the operation.

[1]https://developers.google.com/protocol-buffers

5 FRAMEWORK AND IMPLEMENTATION DETAILS

In this chapter, a coordination strategy based on a set of algorithms - such as construction, particle swarm optimization, adjustment, initialization, integration, and others - are presented along with their mathematical definition and logical description. That is combined with an SDN-based communication protocol, in which the message types considered in this work are also detailed and explained. The combination of both strategies is named STFANET - SDN-based Topology management and coordination of Flying Ad Hoc Networks - in this work. Finally, a novel approach for the particle swarm optimization selection criteria - as part of the construction algorithm - is proposed and described.

5.1 Definitions

The architecture described in Section 4.1 considers: a set of UNs $U = \{u_k\}_{k=1}^{|U|}$, a set of UAVs as RNs $R = \{r_k\}_{k=1}^{|R|}$, and one UAV as the CN c [1]. One of the main functionalities of the CN is to set the routing table rules of each node based on the graph of the network as a whole. Two distinct routing tables are considered. Firstly, there is one for the control packet transmissions P_c, in which the CN is either the source or the destination of the packets. Moreover, there is another specifically for the data packet transmissions P_d, in which the CN is not included in the graph; for these routes, the source and the destination are UNs of the network. As a consequence, the CN does not participate as a router, therefore saving its computational and energy resources.

In order to manage the network topology formation, the CN needs to store the location and the routing table rules of each node. Firstly, its own location is defined by x_c. The set of RNs' location and the set of UNs' location are defined by $X_R = \{x_v\}_{v \in R}$ and $X_U = \{x_v\}_{v \in U}$, respectively. Aiming to simplify further mathematical formulations, X is defined as the set containing all nodes' location, as shown in (5). In addition, the routing tables rules for data and control packet transmissions of each node are also stored at the CN. Firstly, (6) presents P_d as the set of routes p stored at the CN for data packet transmissions. On its turn, as shown in (7), P_c contains the routes p for control packet transmissions. The union of these set of routes P is shown in (8).

$$X = x_c \cup X_R \cup X_U \tag{5}$$

$$P_d = \{p_u^v\}_{u,v \in U, u \neq v} \tag{6}$$

$$P_c = \{p_u^v\}_{u=c, v \in R \cup U} \cup \{p_u^v\}_{u \in R \cup U, v=c} \tag{7}$$

[1] In this work, having x as a vector, the term $|x|$ refers to the size of the vector x.

$$P = \{p_k\}_{k=1}^{|P|} = P_d \cup P_c \tag{8}$$

In order to best place the RNs and itself, the CN takes into account many aspects. The first criteria to be considered is the minimum distance of d_s among the nodes to avoid their collision. In addition, an ideal link length d_r^* should also be aimed in order to have proper communication through the wireless channel. Moreover, there is a maximum link length of d_r^{max} that two nodes may still be able to communicate among each other. The CN aims to have the nodes allocated between the safe distance and ideal link length - whenever there is an established route. Hence, the constant values d_s, d_r^* and d_r^{max} must obey (9):

$$d_s < d_r^* < d_r^{max} \tag{9}$$

Aiming to compute routing paths for data and control transmissions, the Dijkstra algorithm - proposed in (DIJKSTRA, 1959) - was adopted considering the length of each link as its cost. The algorithm is represented by ρ, as shown in (10). This function is responsible for returning the set of routes P by having X, which contains the locations of the CN, the RNs and the UNs, as well as the maximum link length d_r.

$$P = \rho(X, d_r) \tag{10}$$

Considering that one route among each pair of UNs is desired, (11) defines n_d^* as the ideal number of data packet routes. On its turn, (12) presents the ideal number of control packet routes n_c^*, having that one route from the CN to all other nodes, and vice-versa, is expected. Finally, (13) defines n^* as the total amount of routes in an ideal scenario.

$$n_d^* = \frac{2|U|!}{2!(|U|-2)!} = \frac{|U|!}{(|U|-2)!} = |U|(|U|-1) = |U|^2 - |U| \tag{11}$$

$$n_c^* = 2(|U| + |R|) \tag{12}$$

$$n^* = n_d^* + n_c^* \tag{13}$$

5.2 STFANET

5.2.1 SDN-based Communication Protocol

The CN is responsible for topology coordination and management through the exchange of control packets received from and transmitted to every node included in the network. Table 2 presents the types of messages that are considered in this solution, which are described in the following:

- Notify_Relay_Status and Notify_User_Status: information about nodes' mobility need to be periodically transmitted from each RN and UN to the CN informing their position, speed, and trajectory. This set of information is encapsulated and transmitted to the CN through Notify_Relay_Status (RNs) and Notify_User_Status (UNs) messages;

Table 2 – Control message types exchanged among the CN, RNs and UNs.

Message	Source	Destination
Acknowledge	CN, RN and UN	CN, RN and UN
Notify_Relay_Status	RN	CN
Notify_User_Status	UN	CN
Packet_In	RN	CN
Update_Position	CN	RN
Update_Routing_Table	CN	RN and UN

- Packet_In: in the case of an RN or an UN receives a packet with a destination that is not included in the routing table current rules, the node will notify the occurrence to the CN. Such notification is made through Packet_In messages. The CN will then be able to transmit the routing table rules modifications to the affected nodes.
- Update_Routing_Table: as response to Packet_In requests, the CN communicates the change of routing policies to each affected node (RNs and UNs) through Update_Routing_Table messages;
- Update_Position: the CN processes formation coordination algorithms based on nodes' status, and communicates the change of position to each RN that is affected through Update_Position messages;
- Acknowledge: for each Packet_In, Update_Routing_Table and Update_Position being transmitted, the destination responds with an Acknowledge message to the source node. Having that the source does not receive such confirmation packet, it re-transmits the original command in order to ensure network control.

5.2.2 Formation Coordination

Based on the collected mobility information from RNs and UNs, the CN is able to compute formation coordination algorithms to best allocate the available RNs in order to sustain the desired connectivity among the UNs. The periodic execution of these algorithms results in topology configurations that are set through the use of control packets.

In this sense, the main algorithms for formation coordination are explained, which aim to solve the construction, adjustment, and integration of the topology formation, as well as the node allocation problem. As will be presented in this section, a PSO-based strategy is used in the construction algorithm. On its turn, a Gradient function is used towards the adjustment algorithm. Throughout the entire operation, both construction and adjustment algorithms are used independently to update nodes' positions, in a procedure referred to as the integration algorithm. Finally, an algorithm for solving the node allocation problem based on distance vectors is also presented.

5.2.2.1 Construction

STFANET considers a construction algorithm in order to build the topology first formation and redesign it whenever is needed. In this sense, STFANET takes into account the Particle Swarm Optimization (PSO) algorithm to find an optimal solution. Specifically, the aim is to define the best solution for node placement after a fixed number of random solutions have been interactively improved. Having it shortly, for each iteration and for each solution (a particle), the current one is compared to its best (known as the local best) and to the best of all other particles by the moment (known as the global best solution). The local and the global best solution are updated through the iterations. In the end, the final result is the best global solution that has been achieved.

In the PSO algorithm, particles - or solutions - need to be compared with the local and the global best solution at each iteration. Rather than using a single cost function, a particle will be selected rather than another if the first contains more active routes than the second. If there is no difference in this metric, the objective is to minimize a cost function that considers the sum of the link length, and the maximum violations of the link length and safe distance among nodes. As a result, the proposed evaluation of the topology solutions takes into account the number of routes, the links' length and the distance among nodes. The cost function is presented in (14), which is the sum of (15), (16) and (17). In the following, $\delta(u, v)$ is considered to be the distance between nodes u and v. Firstly, (15) determines the sum of the link length being part of the topology, having σ as weight. Equation (16) considers the restriction of having link lengths greater than the ideal value (d_r^*). Therefore, this component is the sum of the distances between the nodes which do not respect the communication range distance and takes λ as weight. Finally, (17) considers the restriction of having the nodes closer than the admitted distance value. On its turn, this metric is defined as the square of the closest link that does not obey such restriction and takes μ as weight.

$$f(X, P) = f_1(X, P) + f_2(X, P) + f_3(X, P) \tag{14}$$

$$f_1(X, P) = \sigma \sum_{p \in P} \sum_{k=1,\dots,|p|-1} \delta(p_k, p_{k+1}) \tag{15}$$

$$f_2(X, P) = \lambda \sum_{p \in P} \Big(\max \big\{0, \max_{k=1,\dots,|p|-1} \delta(p_k, p_{k+1}) - d_r^* \big\} \Big)^2 \tag{16}$$

$$f_3(X, P) = \mu \Big(\max \big\{0, d_s - \min_{u,v \in c \cup R \cup I, u \neq v} \delta(u, v) \big\} \Big)^2 \tag{17}$$

In the following, the population Ψ is a set of particles ψ, which its size is $N_\Psi = |\Psi|$. Moreover, r_1 and r_2 are random variables uniformly distributed on vectors $[0, 1]^2$, and $\circ$ stands for the Hadamard product [2]. During the iterations of the PSO algorithm, (18), (19), and (20) are used to determine the new particles' proprieties - *i.e.*, velocities and positions. Firstly, (18) determines the new velocity value V_ψ^{k+1}, which depends on the current value V_ψ^k and the weight w. In addition, the velocity value also depends on the current local best position $X_{\psi^*}^k$ weighted by $k_1 \cdot r_1$ and the current global best solution $X_{g^*}^k$ weighted by $k_2 \cdot r_2$. In (19), after having determined the new velocity value V^{k+1} - based on the current state -, each element of $V_\psi^k = \{v_{\psi,j}^k\}_{j=1}^{|V|}$ is analysed and, if necessary, set to its limit V_{max}. Finally, the next position value is the sum of the current value and the velocity value recently determined, as seen in (20).

[2]The Hadamard product takes two matrices of the same dimensions and produces another matrix of the same dimension as the operands where each element i, j is the product of elements i, j of the original two matrices.

$$\begin{aligned} V^{k+1}_{c\cup R,\psi} &= wV^{k}_{c\cup R,\psi} \\ &\quad + k_1 r_1 \circ (X^{k}_{c\cup R,\psi^*} - X^{k}_{c\cup R,\psi}) \\ &\quad + k_2 r_2 \circ (X^{k}_{c\cup R,g} - X^{k}_{c\cup R,\psi}), \quad \forall \psi \in \Psi \end{aligned} \tag{18}$$

$$v^{k+1}_{\psi,j} = \begin{cases} v^{k+1}_{\psi,j}, & \text{if } v^{k+1}_{\psi,j} \in [-V_{max}, V_{max}]. \\ -V_{max}, & \text{if } v^{k+1}_{\psi,j} < -V_{max}. \\ V_{max}, & \text{otherwise.} \end{cases} \tag{19}$$

$$X^{k+1}_{c\cup R,\psi} = X^{k}_{c\cup R,\psi} + V^{k+1}_{c\cup R,\psi}, \quad \forall \psi \in \Psi \tag{20}$$

The construction algorithm is presented in (17). As shown, the entries are the positions of the UNs X_U and the number of available RNs $|R|$. The output is the CN and RNs' formation - respectively, x^*_c and X^*_R - resulted by the PSO computation. Firstly, the particles ψ are initialized with uniformly random RNs' positions within the UNs' perimeter (line 2) and routes are formed (line 3). While the particles are initialized, their individual local best solution ψ^* are set to their initial values (line 4), as well as the global solution g is continuously updated among the initializations (line 5). Following that, the PSO algorithm is iteratively computed by altering the particle values (line 9 and 10), computing new routing policies (line 11), as well as choosing the best solution among the derived solutions (line 12 and 13). The iterations run until one of the two following conditions is satisfied: the global best solution does not suffer any change during a fixed number of iterations N^t_k or the maximum number of iterations N_k is achieved. In the end, the algorithm's output (x^*_c and X^*_R) is derived from the best global solution computed so far (line 15 and 16).

Equation (21) defines the construction algorithm. As can be seen, α represents the construction procedure that returns an unsorted RNs' location $X^{'}_R$ and the CN's location x_c, having the UNs' location X_U and the number of RNs $|R|$ as entries.

$$\{x_c, X^{'}_R\} = \alpha(X_U, |R|) \tag{21}$$

5.2.2.2 *Adjustment*

The incremental adjustment of the nodes' location is performed by monitoring the built topology formation. Basically, by measuring the distance between each node and its neighbors, a gradient function aims to evenly distribute them. In this way, the adjustment algorithm does not compute any other routing alternative in order to consider a better routing solution. Instead, it only alters nodes' position in order to best allocate them facing the current UAV formation.

Both (22) and (23) are in charge of performing the formation adjustment. In (22), the resultant position x^*_n referred to the node n is derived from the current state x_n and gradient value Δ_{x_n}, which is on its turn determined based on its neighbors as shown in (23). In addition, there is a threshold value in order to consider a speed limit in the adjustment strength, where γ scales the gradient and γ_r defines the maximum adjustment value (speed limit). In order to determine the gradient value Δ_{x_n} referred to the node n, (23) takes into consideration each neighbour node v which belongs to one of the routes contained in P; $N_n(p)$ is the set of two neighbors of the node n belonged the route p.

```
   Input: $X_U$ and $|R|$.
   Output: $x_c^*$ and $X_R^*$.
1  for each particle $\psi \in \Psi$ do
2      Initialize $x_{c,\psi}$, $X_{R,\psi}$, $v_{c,\psi}$, and $V_{R,\psi}$.
3      $P_\psi \leftarrow \rho(x_{c,\psi}, X_{R,\psi}, X_U, d_r^*)$;
4      $\psi^* \leftarrow \psi$;
5      Update $g$ according to $f(x_{c,\psi^*}, X_{R,\psi^*}, X_U, P_{\psi^*})$.
6  end
7  repeat
8      for each particle $\psi \in \Psi$ do
9          Update $v_{c,\psi}$ and $V_{R,\psi}$ according to (18) and (19).
10         Update $x_{c,\psi}$ and $X_{R,\psi}$ according to (20).
11         $P_\psi \leftarrow \rho(x_{c,\psi}, X_{R,\psi}, X_M, d_r^*)$;
12         Update $\psi^*$ according to $f(x_{c,\psi}, X_{R,\psi}, X_U, P_\psi)$.
13         Update $g$ according to $f(x_{c,\psi^*}, X_{R,\psi^*}, X_U, P_{\psi^*})$.
14     end
15     $x_c^* \leftarrow x_{c,g}$;
16     $X_R^* \leftarrow X_{R,g}$;
17 until termination conditions are satisfied;
```

Algorithm 1: Construction algorithm.

Finally, α defines the degree of agility (or response) for reacting to topology formation changes.

$$\begin{cases} x_n - \gamma \cdot \Delta_{x_n}, & \text{if } ||\gamma \Delta_{x_n}|| \leq \gamma_r. \\ x_n - \gamma_r \frac{\Delta_{x_n}}{||\Delta_{x_n}||}, & \text{otherwise.} \end{cases} \tag{22}$$

$$\Delta_{x_n}(X, P) = \sum_{p \in P} \sum_{v \in N_n(p)} \left[\alpha ||x_n - x_v||^{\alpha-2}(x_n - x_v)\right] \tag{23}$$

The algorithm (3) describes the execution of the adjustment method. The input parameters are the nodes' positioning (x_c, X_R and X_U), and the current routing policy P. On its turn, the output of this algorithm is the desired positions of the CN x_c and the RNs X_R. As shown, the method will compute each position - considering the CN and the RNs - individually according to (22) and (23).

```
   Input: $x_c$,$X_R$,$X_U$, and $P$.
   Output: $x_c^*$ and $X_R^*$.
1  for each node $n \in c \cup R$ do
2      Find $x_n^*$ according to (22) and (23).
3  end
```

Algorithm 2: Adjustment algorithm.

In this work, β represents the adjustment procedure, as shown in (24). The function returns the ideal position of the CN x_c^* and the RNs' location X_R^*, having the current nodes' location X and the active routes P as entries.

$$\{x_c^*, X_R^*\} = \beta(X, P) \tag{24}$$

5.2.2.3 Nodes allocation

Towards the formation coordination, the first step for the CN is to set the UAV formation. As seen earlier, after that, the topology formation is then periodically adapted by using the adjustment methodology. However, at some point, the CN may need to alter the topology formation by performing the construction algorithm once again. In both cases, after having constructed the topology formation, the current UAV formation needs to be remodeled to the newest proposed formation. In order to perform that task, the RNs need to be assigned.

In this work, the selection policy takes into account the current nodes' location and the future formation, in a way that the closest node will be allocated. Aiming to perform the assignment, the strategy is to consider that $D_{|R| \times |R|}$ is a matrix that contains all combinations of distances from the current nodes to the target locations. Considering that $||x_u - x_v||$ represents the distance from node v to node u, (25) depicts the content of matrix D.

$$D_{u,v} = \{||x_u - x_v||\}_{u,v \in R} \tag{25}$$

The idea is to iteractively select the minimum values of the matrix D, in order to provide an ordered list of most proximal nodes and their future position. After finding it, both the column and the line of this element need to be discarded. For the next iterations, the algorithm will continue by searching for the minimum value of those elements that are still valid. As a consequence, the algorithm is able to identify the closest relation between current nodes' location and the desired formation.

Equation (26) presents λ as the function responsible for returning the ideal and sorted nodes' location X_R^* based on the current location X_R and the desired location $X_R^{'}$ - given in a unsorted sequence.

$$\{X_R^*\} = \lambda(X_R, X_R^{'}) \tag{26}$$

Moreover, after the node allocation algorithm has been processed, the CN is able to estimate how long should the RNs take in order to completely recompose the topology formation. The prediction is taken by the longest distance and the mobility model of the RN. The time needed in order to have all the RNs at the desired position is given by δ.

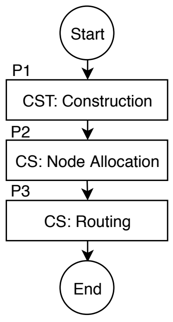
5.2.2.4 Cleaning Routes

In order to evaluate the current routing strategy having the nodes' location updates, the CN should be able to check whether the topology formation changes are not causing any damage for the network performance. Thus, the cleaning procedure aims to discard routes that might not be operating properly. The result of this method is not directly informed to the other nodes. This outcome is used to detect whether the CN should act in order to reestablish connectivity among the nodes.

Having a set of routes, the algorithm will filter those which contain link distances greater than a threshold value given as parameter. Equation (27) represents this method considering X as the current position of all the nodes, P as the set of the routes, and d_r as the limit link distance (threshold) for cleaning. The response will be a set of cleaned routes P'.

$$P' = \gamma(X, P, d_r) \tag{27}$$

Figure 11 – Flowchart presenting the initialization phase of the coordination algorithm including the descriptions of each step. CS and CST stand for "Compute and Set" and "Compute and Store", respectively.



Source: The author.

5.2.2.5 Coordination

The coordination algorithm is a combination of two main stages, which are: initialization and integration. The initialization is performed once, as soon as the controller node has just started by basically computing the construction algorithm. This stage is responsible for the initial topology formation, as well as defining its routing policies for forwarding both control and data packets. The CN is also responsible for combining both construction and adjustment algorithms in order to provide a communication link among the UNs for as long as possible. As a consequence, an integration algorithm must regulate the network as fast as possible aiming to react quickly to the mobility of the UNs. Such task is a while-true operation, which is performed throughout the entire operation.

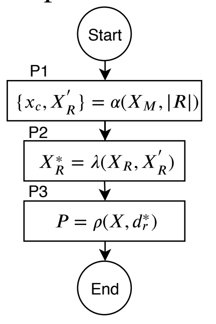
Before performing either the construction or adjustment algorithm, the current nodes' positioning is estimated based on their last contextual information that was successfully received by the CN. Therefore, for each node, the CN uses the trajectory contained in the last received message along with the timestamp, in order to predict its current location. By doing that, even though some control packets have not been received, the CN is able to cover such missing information and the network is not significantly affected by that.

Figures 11 and 12 present the flowchart which depicts the initialization algorithm. The first (Figure 11) describes the process, while the later (Figure 12) shows the equations regarding each step of the flowchart. Similarly, both Figures 13 and 14 present the flowchart of the integration algorithm - a while-true operation - including the description and the equations, respectively. The initialization and the integration are better described individually in the following.

- Initialization

The algorithm begins considering that every node is located in the range of the CN; even that they might not be in the range of each other. Firstly, the CN aims to plan the RNs' and its own position through the construction algorithm (P1). The RNs are then better allocated using the node allocation algorithm (P2). As soon as it is finished, the routing is computed using the routing strategy (P3). From this moment, the CN will keep monitoring the network in order to best update the routing strategies and RNs' location based on the collected information.

Figure 12 – Flowchart presenting the initialization phase of the coordination algorithm including the equations used in each step.


Start
P1
$\{x_c, X'_R\} = \alpha(X_M, |R|)$
P2
$X^*_R = \lambda(X_R, X'_R)$
P3
$P = \rho(X, d^*_r)$
End

Source: The author.

- Integration

Once the topology formation is built and set, the CN is responsible for adjusting it according to the UNs' movements. Having that the construction algorithm should be avoided due to its high computational cost, the adjustment algorithm is able to continuously adapt the network until there is a need to change the current topology formation.

The first aspect addressed in the integration is: has been passed the needed time δ since the last construction instant t_{last} until the present moment t_{now}? This question will guide the definition of which parameter should be used in order to accommodate the topology formation through the adjustment algorithm. In other words, this question will determine whether the adjustment algorithm will consider the current routing strategy (P5) or the one that is aimed when the nodes achieve at the desired position of the construction algorithm (P5 and P6).

After having just computed the adjustment of the UAV formation, a checking procedure for evaluating the need for changing the network configuration is applied. The procedure starts at P7. At this point, the CN contains the current routing strategy P and the updated nodes' location X. The cleaning method is performed in order to internally discard the routes that have a link length greater than the ideal value of d^*_r. Having the cleaned routes P', the algorithm is able to evaluate whether the current nodes' locations have led the current solution to not reach the ideal number of routes n^* (Q2). In the case that the current solution is adequate, the UAV formation is not modified. However, if the current solution is not satisfactory, the algorithm attempts to solve the inefficacy by computing a routing strategy considering the current formation X (P8).

The proposed routing solution needs to be evaluated. Thus, Q3 performs the same question as Q2, examining the current routing strategy P - which has been recently computed - with the ideal number of routes n^*. If the newest routing solution has solved the inefficacy, nothing else is either computed or modified.

The previous question Q3 may indicate that the current routing strategy may also not have been sufficient for having the desired connectivity. In this case, the algorithm should consider that the nodes may not have achieved their ideal location X^* by the moment. In order to evaluate the routing strategy considering that all nodes are hypothetically at their ideal locations, the routing algorithm is performed once again (P9). The resulting routing strategy P' is temporary and it is considered just to check its efficiency through question Q4. Having concluded that nodes' location targets lead to the desired connectivity, a temporary routing strategy is performed (P15). At this time, it is considered the current nodes' position X and the maximum link length d_r^{max} in order to have a temporary establishment of the network until the nodes do not reach their target position. On the other hand, the question Q5 may indicate that achieving the nodes' location targets does not lead to the desired network connectivity. In this case, the algorithm will start computing the construction algorithm (P10).

Similar to the initialization process, the computation algorithm is performed and the resulted topology formation is stored (P10). As soon as it is finished, having the current RNs' location X_R and the unsorted targets X_R^*, the algorithm needs to determine which RN is assigned to each desired location by setting a temporary solution X_R'' at P11. After completing this procedure, the CN needs to evaluate whether the recent computed result will overtake the current formation. As a consequence, the CN computes and stores the predicted routes having the desired positions at P12. In addition, the CN cleans the routes that the link length is greater than the ideal value of d_r^* at P13. The comparison between both solutions (Q5) will lead the CN to choose between keeping the current solution or setting the recent computed solution. Having the decision made to adopt the computed solution, the CN sets the RNs' target positions at P14, as well as set a temporary routing scheme until the RNs achieve their ideal positions (P15).

5.2.2.6 Time complexity analysis

The computational performance of both construction and adjustment algorithms can be evaluated through their time complexity analysis, which is presented in Table 3. In the following table, $|R|$ and $|U|$ stands for the number of relay and user nodes, N_Ψ and N_k stands for the number of particles and iterations of the PSO algorithm, and $|P|$ is the number of routes previously defined in (8).

Table 3 – Time complexity analysis of construction and adjustment algorithms.

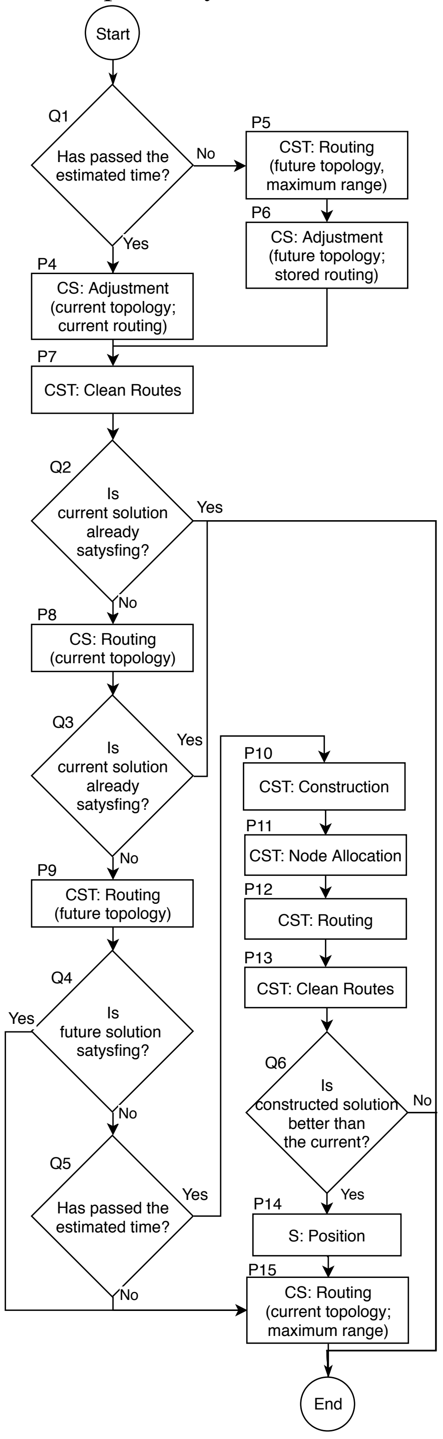
Algorithm	O(.)
Construction (PSO algorithm)	$O(N_k * N_\Psi * (\|U\| + \|R\|\|)^2)$
Adjustment (Gradient function)	$O(\|R\| * \|P\|)$

In O notation terms, the construction algorithm presented in Section 5.2.2.1 can be expressed as $O(N_k * N_\Psi * (|U| + |R||)^2)$, which becomes $O(N^4)$. As it is computationally intensive and implies a high computation cost if performed frequently, a lighter algorithm is generally preferred. The adjustment algorithm, presented in Section 5.2.2.2, can be defined as $O(|R| * |P|)$ in O notation, which becomes $O(N^2)$.

5.3 Particle Selection Criteria in Particle Swarm Optimization

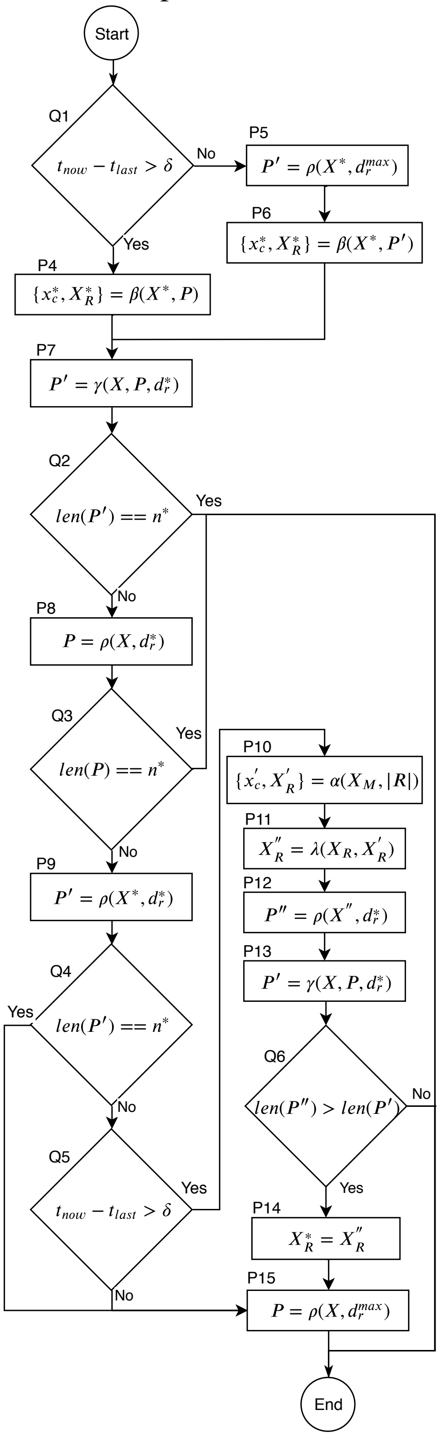
In Section 5.2.2.1, the discussed approach based on KIM; LEE (2018b) for constructing a topology formation aims basically to maximize the number of interconnected users and minimize link distances while maintaining safe distances among nodes. In that case, the network connectivity would still rely on individual nodes, in which their operational failure would compromise ongoing connections and transmissions among users. In other words, the restoration of active transmissions during failure occurrences might be merely performed by nodes' position replacement, which would demand reallocation time and energy consumption. Moreover, user applications usually require specific network performance in order to guarantee their functionality and user satisfaction, such as packet loss, latency, and jitter. To take advantage of software-defined monitoring and controlling

Figure 13 – Flowchart presenting the while-true operation of the integration algorithm including the descriptions of each step. S, CS, and CST stand for "Set", "Compute and Set", and "Compute and Store", respectively.


Start
Q1
Has passed the estimated time?
No
Yes
P5
CST: Routing (future topology, maximum range)
P6
CS: Adjustment (future topology; stored routing)
P4
CS: Adjustment (current topology; current routing)
P7
CST: Clean Routes
Q2
Is current solution already satysfing?
Yes
No
P8
CS: Routing (current topology)
Q3
Is current solution already satysfing?
Yes
No
P9
CST: Routing (future topology)
Q4
Is future solution satysfing?
Yes
No
Q5
Has passed the estimated time?
Yes
No
P10
CST: Construction
P11
CST: Node Allocation
P12
CST: Routing
P13
CST: Clean Routes
Q6
Is constructed solution better than the current?
No
Yes
P14
S: Position
P15
CS: Routing (current topology; maximum range)
End

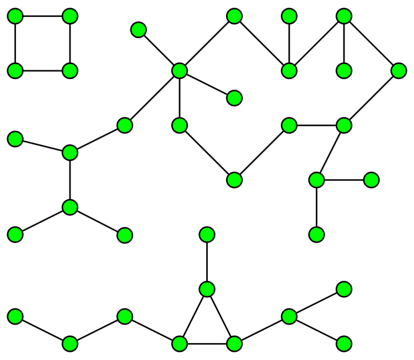
Source: The author.

Figure 14 – Flowchart presenting the while-true operation of the integration algorithm including the equations used in each step.


Start
Q1
$t_{now} - t_{last} > \delta$
No
Yes
P5
$P' = \rho(X^*, d_r^{max})$
P6
$\{x_c^*, X_R^*\} = \beta(X^*, P')$
P4
$\{x_c^*, X_R^*\} = \beta(X^*, P)$
P7
$P' = \gamma(X, P, d_r^*)$
Q2
$len(P') == n^*$
Yes
No
P8
$P = \rho(X, d_r^*)$
Q3
$len(P) == n^*$
Yes
No
P9
$P' = \rho(X^*, d_r^*)$
Q4
$len(P') == n^*$
Yes
No
Q5
$t_{now} - t_{last} > \delta$
Yes
No
P10
$\{x_c', X_R'\} = \alpha(X_M, |R|)$
P11
$X_R'' = \lambda(X_R, X_R')$
P12
$P'' = \rho(X'', d_r^*)$
P13
$P' = \gamma(X, P, d_r^*)$
Q6
$len(P'') > len(P')$
No
Yes
P14
$X_R^* = X_R''$
P15
$P = \rho(X, d_r^{max})$
End

Source: The author.

Figure 15 – Connected components in a graph.



Source: The author.

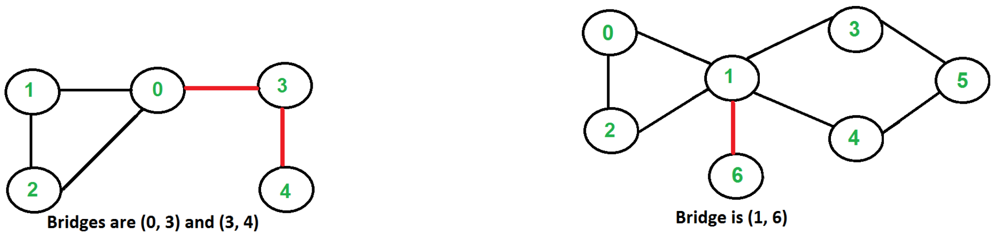
functionalities (Section 2.3), the network design should provide manageability resources by providing alternative and redundant routing paths. In this case, the network manager should be able to configure nodes' network policies keen to achieve desired network performance requirements, or even restore transmission failures occurred after nodes' failure. Finally, user mobility is an important concern that should be addressed in formation construction. By maximizing area coverage, the nodes can move while still maintaining a connection to the network.

In order to enhance the UAV formation that outcomes from the construction phase, a new approach is proposed aimed to construct a more resilient and manageable topology formation. For that, this novel proposal considers a set of graph theory concepts for network evaluation to guarantee user connectivity, alternative transmission paths and less possible amount of nodes being points of failure, as a consequence. In addition, the spring virtual force method - firstly introduced by TROTTA *et al.* (2018) - is applied by using attractive-repulsive forces among nodes in order to accomplish the following objectives: impose safety distance gaps for avoiding collision, provide sufficient communication link distance for proper link quality, and maximize area coverage for enabling user mobility.

In graph theory concept, a component - also referred to as connected component - of a graph is a sub-graph in which any two vertices are connected by paths, and which is connected to no additional vertices in the super-graph. For instance, Figure 15 illustrates a graph that has three components. As a consequence, a graph that is itself connected has exactly one component, consisting of the whole graph. Furthermore, an edge is referred to as a bridge if removing it disconnects the graph. In other words, the removal of a bridge increases the number of connected components in a graph. In the following, Figure 16 presents two graphs having bridges highlighted in red color, as an example. As already mentioned, one of the objectives of this novel approach remains in maximizing the number of connected users - *e.g.*, the number of components contained in the graph. Moreover, aiming to provide alternative transmission paths, as well as minimizing the number of points of failure, this novel approach also aims in minimizing the number of bridges. Such metrics related to the number of connected components and the number of bridges are discrete, therefore leading to discrete optimization described in Section 2.2.4.

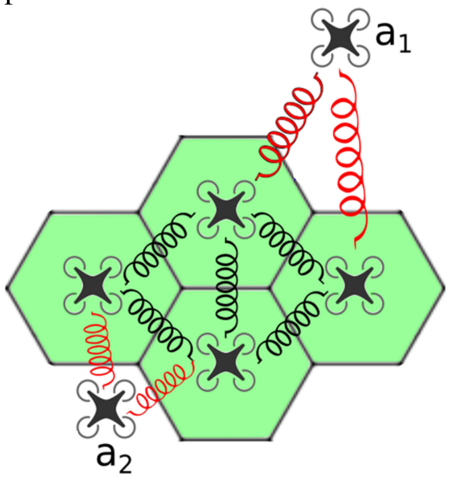
A virtual spring force $\vec{F}(u,v)$ - described in (TROTTA *et al.*, 2018) - acts between each pair of nodes u and v for each edge of the graph, and its intensity is computed according to the Hooke's law, as presented in (28). The spring displacement is given by the current distance from u to v and the length of the spring equilibrium d_{EQ}. In this case, the distance d_{EQ} among the nodes should be appropriate for having them properly

Figure 16 – Bridges in a graph.



Source: The author.

Figure 17 – Representation of the virtual spring method, where a_1 is being attracted by their neighbours, and a_2, repulsed.



Source: TROTTA *et al.* (2018) - modified by the author.

positioned according to hexagonal patterns for optimal scenario coverage. The force is attractive when the distance among nodes is greater than d_{EQ}, and repulse otherwise. The stiffness of the spring k_{ST} is assumed to be a constant value. In this way, the proposed method balances the "push" and "pull" forces, as depicted in Figure 17.

$$\vec{F}(u, v) = -k_{ST}(|\vec{x}_u - \vec{x}_v| - d_{EQ}) \tag{28}$$

Finally, the particle selection procedure of the PSO algorithm is based on the following evaluation: firstly, the most important criteria is to minimize the number of connected components of the graph, therefore maintaining the maximum amount of nodes interconnected. As a consequence, for each pair of particles - or solutions -, the one that presents less number of connected components is considered to be dominant. By having an equal number of connected components, the number of bridges is minimized: a particle that presents less number of bridges, in this case, is a dominant solution. Having an equal number of bridges, the objective is to minimize the cost function presented in (29), which is a representation of the virtual spring method.

$$f(X_R) = \sum_{u,v \in R, u \neq v} |\vec{F}(u, v)| \tag{29}$$

6 EVALUATION AND ANALYSIS

6.1 STFANET

Simulations of the proposed STFANET coordination protocol were performed using the Mobile Multi-Media Wireless Sensor Network (M3WSN) OMNeT++ framework (ROSÁRIO *et al.*, 2013). OMNeT++ is a network simulator for implementing and testing network solution architectures and setups. This work focus on the implementation of the STFANET in the network layer. Therefore, by using this framework, it is possible to gather valuable results considering the already modeled physical layer constraints. In addition, by being a discrete event simulator, there is no need to present the processing capabilities of the computers used for running the simulations. This means that the simulation processing times do not affect the results. A campaign of 70 independent simulation runs was conducted, varying the INs' movements and transmissions. As evaluation metrics, packet loss, latency, and connectivity index were assessed, presenting the mean and the confidence interval of 95%. These metrics were evaluated by comparing the variation of the number of RNs contained in the UAV network. Therefore, the following metrics are considered to evaluate the network reliability:

- Packet loss (%): rate of non-delivered packets to the final destination over the number of sent packets from the source.
- Latency (ms): period between the transmission of a packet from its source to its delivery in the final destination.
- Connectivity (%): rate of time that all nodes are capable of transmitting a message to any other node throughout the simulation time.

6.1.1 Scenario

This work considers a scenario with sets of 3, 6, 9, 12 and 15 RNs in order to provide connectivity among 5 UNs moving randomly through a flat terrain of 10km x 10km. Each UN transmits a data packet to a random destination in intervals that varies from 100ms to 200ms. The UNs are able to move from 5 to 10m/s, while the RNs and the CN are considered to be able to move at a maximum speed of 20m/s. The simulation was performed for 10 minutes long, which was considered to be enough in order to have the nodes sufficiently spread through the environment, as they start at the center of the scenario. All nodes can transmit data in a communication range of around 1000m. The coordination protocol also needs to be configured. The nodes are set to transmit their contextual information to the controller in intervals of 250ms. The CN adjusts or reconstructs the topology formation in intervals of 500ms. The simulation parameters were also set to

Table 4 – Simulation parameters.

Simulation Parameters	Value
Scenario	
Time	10min
Dimension	10 x 10km
Number of user nodes	5
Number of relay nodes	[3, 6, 9, 12, 15]
User nodes' speed	[5..10]m/s
Relay nodes' speed	20m/s
Protocol	
Beacon interval	250ms
Formation interval	500ms
Distances	
Minimum distance (d_s)	50m
Ideal link length (d_r^*)	700m
Maximum link length (d_r^{max})	1000m
Construction	
Number of particles (N_Ψ)	30
Number of iterations (N_k)	300
Stop threshold (N_k^t)	10
Inertia weight (w)	0.7
Cognitive parameter (k_1)	1.5
Social parameter (k_2)	1.5
Threshold for velocity clamping (V_{max})	20%
Sum of link length weight (σ)	0.5
Ideal link distance weight (λ)	0.3
Safety distance weight (μ)	0.3
Adjustment	
Positive step size (γ)	0.05
Maximum travel distance (γ_r)	2
Power of link distance (α)	5

allow wireless channel temporal variations, link asymmetry, and irregular radio ranges, as expected in a real scenario. UAVs rely on the CSMA/CA MAC protocol, without using RTS/CTS messages and retransmissions. In case of buffer overflow, the UAVs consider a drop tail mechanism to drop packets.

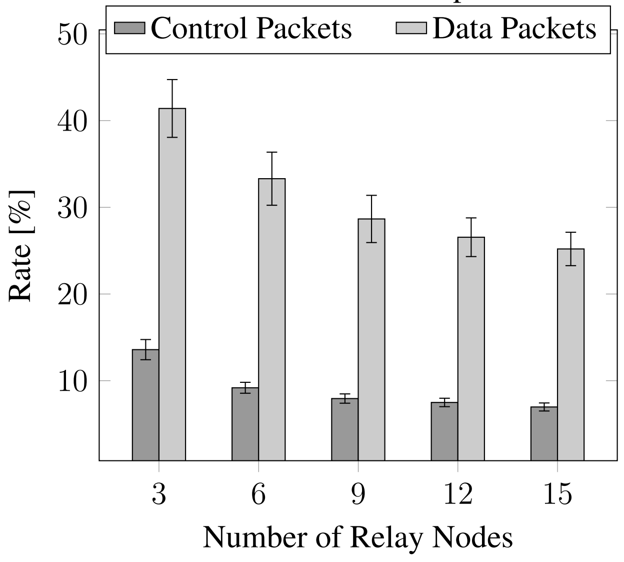
The parameters referred to both scenario and STFANET coordination protocol are presented in Table 4. Those which are manipulated in mathematical expressions included in this work are presented along with their respective symbols. Parameters related to the PSO algorithm, such as the number of particles and the number of iterations were based on KIM; LEE (2018b).

In the following, the control plane includes messages being transmitted between each node and the CN for network management proposes. On its turn, the data plane includes exclusively messages being transmitted among INs and originated from their application layer.

6.1.2 Packet Loss

Figure 18 presents the mean and the confidence interval of the packet loss rate for the control and data plane. As expected, the increase of the number of RNs in the UAV network resulted in a decrease of the packet loss in both planes. This is due to the greater number of connections and the smaller link length needed as more RNs are available. The rate dropped 7.7% (from 14% to 7%) in the control plane and 27.7% (from 42% to 25%) in the data plane. In addition, the overall best performance of the control plane against the data plane is due to the ability of the CN to address the RNs' positions, and therefore keep them connected even that the UNs are not reachable.

Figure 18 – Packet loss for control and data packet transmissions.



Source: The author.

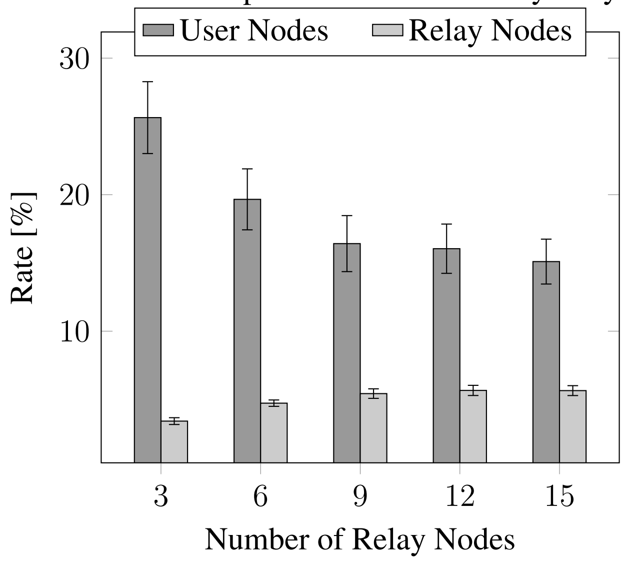
Specifically evaluating the control plane, Figure 19 categorizes the packet loss rate of the control plane by the CN is communicating either with an UN or an RN. The graph shows that the packet loss rate for RNs is consistently lower than for UNs. As can be seen, there was a fall of 11% (from 26% to 15%) in the packet loss rate for the UNs. As greater was the number of RNs, they were able to contribute better to form links between the UNs and the CN. On the other hand, having control packets being delivered through one or more hops as increases the number of RNs in the UAV network, there was a slight rise of 3% (from 3% to 6%) in the packet loss considering the RNs, exclusively.

In both Figure 18 and Figure 19, the interval confidence decreases with the increase in the number of RNs in the UAV network. For instance, considering the messages being transmitted in the data plane, the confidence interval decreased by 42.2%. On its turn, in the control plane, it decreased by 60.2%. Hence, the results show that the increase in the number of RNs implies not only in the packet loss rate rise but also less variability among the simulations.

6.1.3 Latency

Figures 20 and 21 present the latency of delivered messages being transmitted through the data plane and the control plane, respectively. In addition, the figures categorize the measure according to the number of hops used for the delivery of the messages. For each number of RNs used in the simulation, it is presented the latency on average for the

Figure 19 – Packet loss for control packet transmissions by relay and user nodes.



Source: The author.

messages that used from 1 to 4 hops in order to be successfully transmitted. The following numbers of hops are not shown here, as there was no significant change in the behavior of the graphs.

As can be seen in Figures 20 and 21, there was a decrease of the latency for delivered packets as larger was the number of RNs in the UAV network - either for the control plane or for the data plane. This behavior can be explained by the possibility of having the nodes even closer to each other as there was an increase in its quantity. As a consequence, the latency decreases significantly.

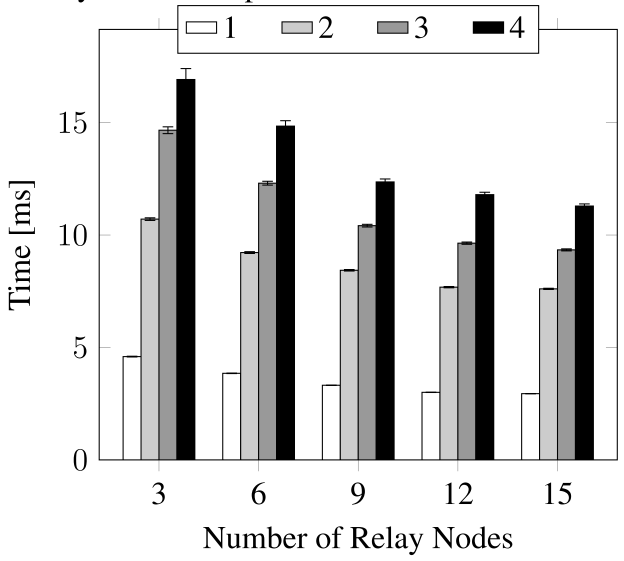
6.1.4 Connectivity

The connectivity can be measured as the ratio of time that all the UNs are able to communicate with each other. Having traced the routing tables' rules and nodes' positions through the simulation, it was possible to measure the period of time that all the UNs were virtually connected - even that they were not in fact communicating. By doing this, as presented in Figure 22, the simulation showed that the connectivity rose 44.6% as it was from 49% with 3 RNs to 71% having 15 RNs available in the UAV network. As expected, as greater was the number of available RNs, the network could keep the UNs connected for a longer period of time.

6.2 A Novel Particle Selection Criteria in PSO

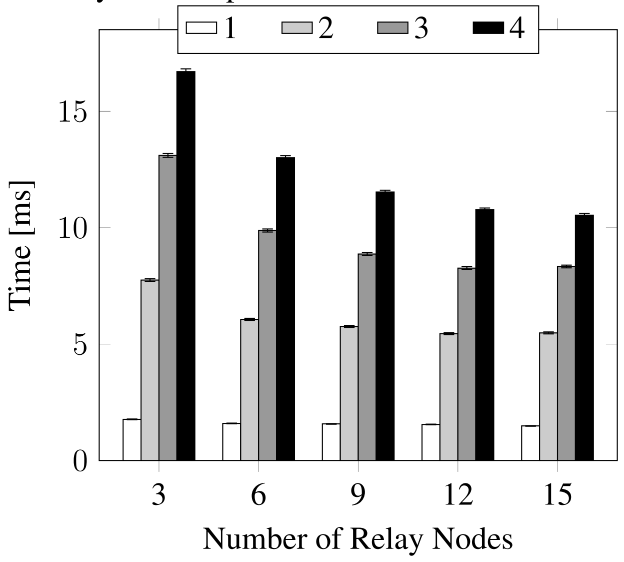
In Section 5.3, a novel approach was proposed for evaluating and selecting particles throughout the PSO iterations during the formation construction phase. Such evaluation is performed following graph theory metrics computation, instead of purely link distance optimization as proposed by KIM; LEE (2018b). By applying those metrics, it is expected that the novel approach overcomes the other in relay node failure occurrences, due to its manageable topology formation that can be explored by the SDN controller. As a consequence, this new approach promises to provide service persistence and less time needed for recovering, attending both availability and manageability, as network fundamental de-

Figure 20 – Latency for control packet transmissions and number of hops.



Source: The author.

Figure 21 – Latency for data packet transmissions and number of hops.

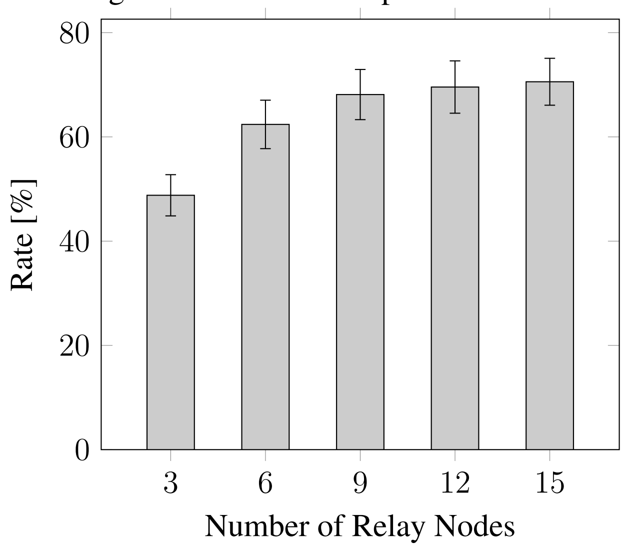


Source: The author.

sign goals. In this section, this new approach will be compared to the one presented by KIM; LEE (2018b) in terms of graph metrics. In the following, the considered metrics are described:

- Number of connected components: it is the number of sub-graphs in which any two vertices are connected by paths. In this case, a vertex with no incident edges is itself a component. A graph that is itself connected has exactly one component, consisting of the whole graph.
- Degree: it is the number of edges adjacent to a node.
- Node connectivity: is equal to the minimum number of nodes that must be removed to disconnect a graph and increase the number of components, consequently.

Figure 22 – Connected period ratio.



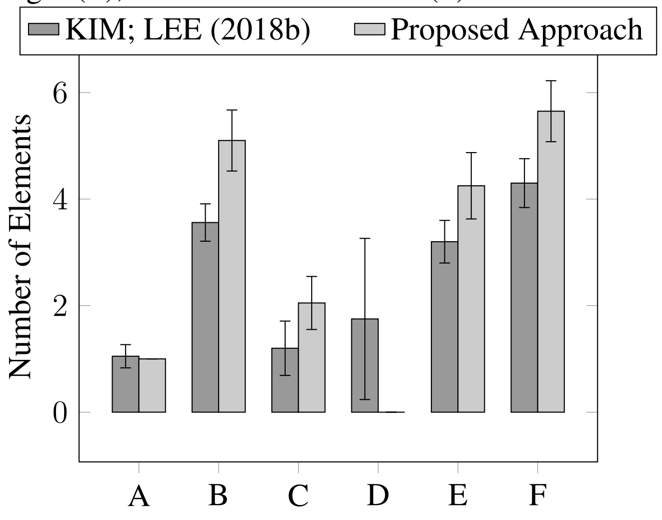
Source: The author.

- Number of bridges: is the number of edges whose removal causes the number of connected components of the graph to increase.
- Maximum clique length: a clique is a subset of the vertex set of an undirected graph, such that for every two vertices, there exists an edge connecting the two. The maximum clique length is the size of the clique of the largest possible size in a given graph.
- Number of colors: vertex coloring assigns a color to each vertex such that no two adjacent vertices are of the same color. The vertex coloring presents how nodes interfere with each other. As a consequence, as less number of colors, less the number of nodes with interference among each other.

A comparison of the formation construction formed between the PSO selection presented by KIM; LEE (2018b) and the one proposed in this work (Section 5.3) is presented in Figures 23 and 24. Firstly, Figure 23 presents the graph theory metrics presented above (node connectivity, number of connected components, number of bridges, maximum clique length, degree mean and number of colors). On its turn, Figure 24 presents the area coverage evaluation of both solutions, measured by the ratio of the area covered by the relay over the maximum reachable area coverage having the available nodes.

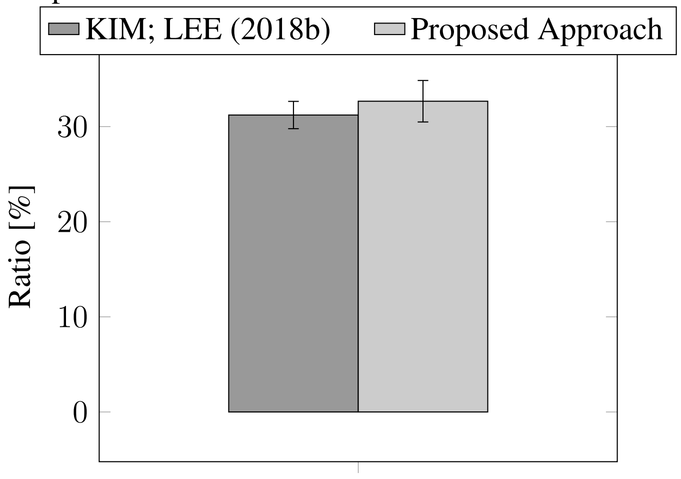
The number of components being equal to one means that the topology formation is completely inter-connected without containing isolated sub-graphs. As can be seen in Figure 23, the PSO solution based on KIM; LEE (2018b) did not guarantee such requirement in all simulation runs, as the mean is slightly greater than 1. On the other hand, this novel approach presented one component in every run, mainly due to the prioritization of this metric in the PSO criteria selection. In addition, the degree has raised from approximately 3.5 up to 5 in the mean. In other words, on average, each node is then able to communicate to 5 other nodes (*i.e.*, neighbors). As a consequence, the minimum number of nodes needed to disconnect the graph (node connectivity) has raised, approximately, from 1 to 2, therefore the number of bridges is reduced from nearly 2 to zero in all runs. As a conclusion, the novel approach provided typology formations that a link failure would not compromise network connection by raising the number of connected components. However, the interference among nodes is compromised. As presented by the number of colors and the maximum number of cliques, the proximity of the nodes raised both metrics - respectively, from approximately 4 to above 5 and from nearly 3 up to 4. Finally, Figure 24 presents the area coverage of both topology formation solutions. By using the spring-force concept in the PSO criteria selection, there is also a concern of spreading the nodes' through the scenario. As a consequence, the area covered by the nodes in the novel proposal is slightly greater than the one proposed by KIM; LEE (2018b).

Figure 23 – A comparison of the formation construction in terms of graph theory metrics: connected components (A), degree (B), node connectivity (C), number of bridges (D), maximum clique length (E), and number of colours (F).



Source: The author.

Figure 24 – A comparison of the formation construction in terms of area coverage.



Source: The author.

6.3 Network Evaluation in Failure Situations

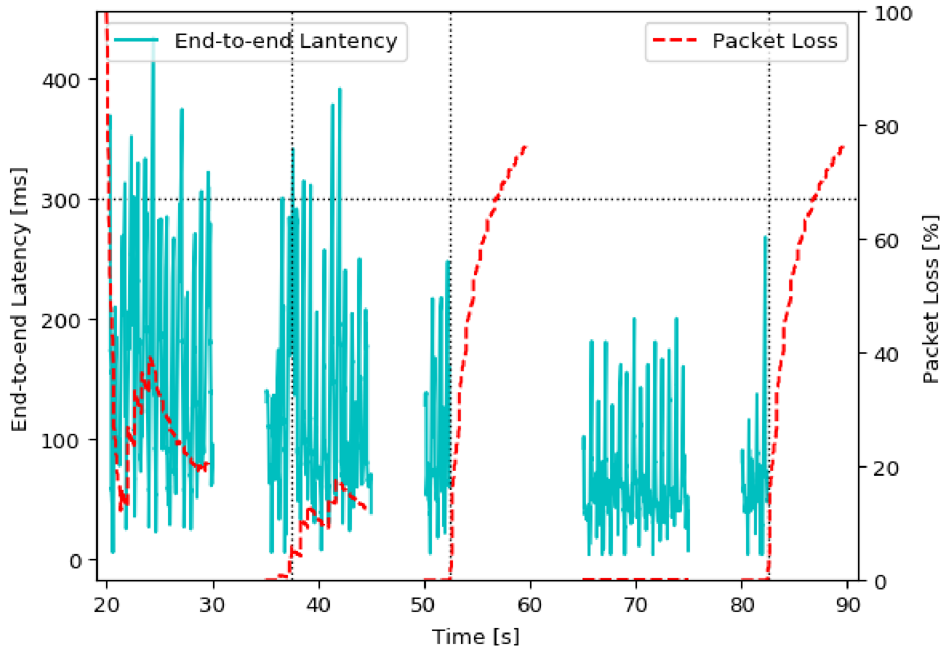
In order to compare and evaluate the PSO outcomes from the particle selection criteria presented by KIM; LEE (2018b) and the one presented in this work (Section 5.3) in terms of realistic wireless communication metrics, simulation runs were performed in the OMNeT++ simulator by having users nodes allocated in random positions and relay nodes assigned to the computed locations. Throughout the simulation, nodes were forced to fail in order to evaluate the network manageability of both topology formation configurations. Aiming to network control, the controller node alters routing policies as soon as a node failure is introduced. It is expected that the capacity and ability to manage a topology is related to the topology formation configuration and its available resources.

An analysis of the latency and packet loss is presented in Figure 25 - respectively, in continuous and dashed lines - for five video transmissions among user nodes. The three first transmissions (from $20s$ to $60s$) are related to the same pair of source and destination nodes, as well as the two last transmissions (from $65s$ to $90s$). The first transmission of each pair happens without any node failure. In the consequent transmissions of each pair, one of the nodes responsible for forwarding that transmission fails having passed $2s$. The vertical lines depict the instant that nodes fail, to better visualize their impact in the network metrics. On its turn, the horizontal lines represent the end-to-end latency tolerance - or delay limit - presented by ZHAO *et al.* (2018), in which a packet is considered to be lost if the delay is over that threshold.

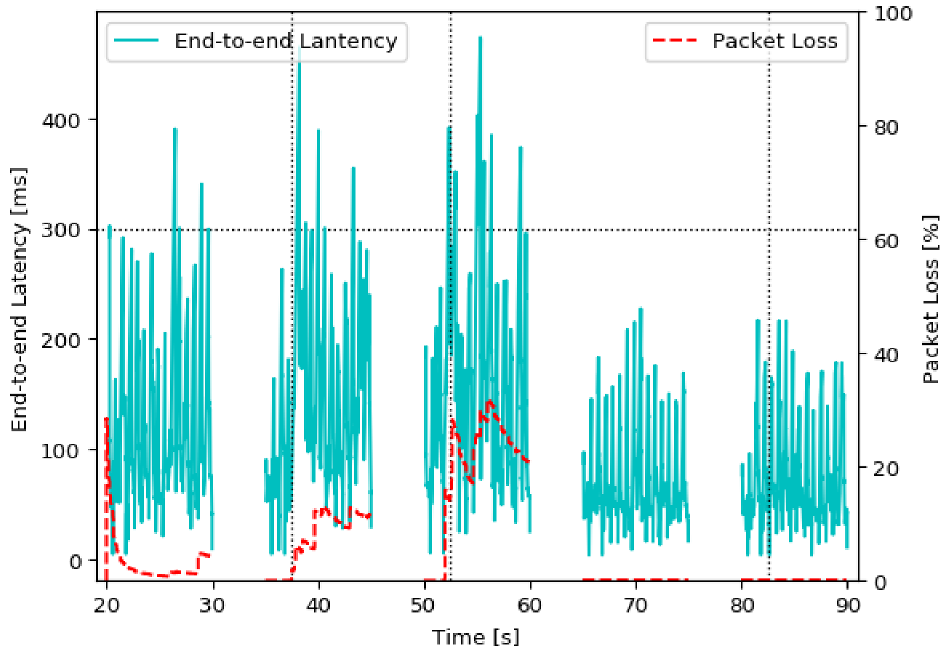
Firstly, Figure 25(a) presents the transmission metrics having the topology formation formed based on KIM; LEE (2018b), and Figure 25(b), having the UAV formation constructed by the novel approach. In Figures 25(a) and 25(b), the first transmission occurs without having any node failure. As can be noticed, Figure 25(a) presents higher packet loss due to the end-to-end delay being over the tolerated limit. In the consequent transmission, disseminated from a same pair of source and destination, one node fails (at $37s$), which is marked by a vertical dotted line in both Figures 25(a) and 25(b). As can be noticed, the controller node was able to manage both the topology formation configurations by redefining routing policies. However, in the third transmission, a second consecutive failure (at $52s$) led the transmission to cease in Figure 25(a). On the other hand, the controller was still able to manage the routing policies in Figure 25(b). In addition, as soon as a node fails, higher becomes the number of nodes that the new path relies on - *i.e.*, the number of hops -, therefore the packet loss increases throughout the transmission, as well as the latency. Finally, the last two transmissions are related to a different pair of source and destination nodes. Similarly, the first one occurs without any node failure. However, in the last transmission, a node failure interrupts the transmission in Figure 25(a), as the controller is not able to manage such failure. On its turn, the controller node manages the routing policies in Figure 25(b) presenting no data packet loss.

Figure 25 – A comparison of the formation construction in terms of latency and packet loss.

(a) (KIM; LEE, 2018b)



(b) Novel Approach.



Source: The author.

7 CONCLUSION AND FUTURE WORK

This work presented an integrated strategy for formation coordination and network management of UAV networks, considering the use of a dedicated node over the control of relay nodes aimed to provide and sustain connections among mobile user nodes. The formation coordination is given by a combination of algorithms, such as the construction and adjustment, which are orchestrated by the integration algorithm. The network management is enabled through an SDN-based communication protocol, responsible for adjusting network policies throughout the operation. Firstly, this work presents a solution by combining an SDN-based communication protocol (ZHAO *et al.*, 2018) with a set of formation coordination algorithms (KIM; LEE, 2018b), which has been named as STFANET. In the following, a novel approach for determining nodes' formation has been proposed by replacing particle selection criteria in the PSO algorithm.

The STFANET passed through a comprehensive set of tests on OMNeT++ to evaluate its efficiency. Without considering re-transmission as delivered packets, the data packet loss rate is limited to 25% and the control packet loss rate is up to 10% over the adopted experimental scenario. The performance gap is due to the capability of the controller to set the relay nodes' position in order to avoid being out of range, whilst the user nodes are not over control of the controller node. Although some control packets would not be successfully transmitted, the controller is able to predict nodes' locations by taking their most recent transmission data. As a result, the proposed protocol was capable of maintaining the network connection for at least 70% of the time. The results also show that the latency is considerably reduced as a higher number of nodes is introduced in the network. Varying from 3 to 15 nodes, the latency can be reduced from 15% to 40%, depending on the number of hops needed for a single transmission. Such outcomes prove a promising direction of the solution for accomplishing its purposes. As expected, the UAV network has shown to provide higher network performance indicators - considering packet loss, latency, and connectivity -, as the number of relay nodes available is increased. Nevertheless, that raise in performance halts as soon as the number of relay nodes has passed a certain value. The aforementioned evaluation would be necessary for determining a sufficient and minimum quantity of relay nodes for providing the desired network performance in a specific scenario.

Focusing on the coordination problem - specifically, the construction procedure - a novel particle selection criteria in the PSO algorithm was proposed. In order to provide persistent network service in the occurrence of node failures and also be able to deliver demanded network performance, network manageability should be guaranteed by mainly increasing the number of alternative paths. Instead of optimizing distances among UAVs, this work proposes the use of graph theory metrics - such as node connectivity, number of bridges and node degree -, and the virtual spring force concept. As a result, a higher level of availability and manageability was obtained and shown through an evaluation in terms of graph theory and network performance metrics - such as latency and packet loss.

Imposing to similar conditions, the novel approach for determining topology formation required that at least one additional node needs to fail in order to increase the number of components when compared to KIM; LEE (2018b). In addition, the number of neighbors (*i.e.*, degree) has increased in two per node in mean, therefore presenting a higher number of alternative paths. Moreover, the area coverage has lightly increased either - although that objective has not been prioritized - enabling a higher degree of node mobility. As a drawback, the topology formation presented an increase of communication interference among nodes. The maximum group length of nodes that interfere with each other (*i.e.*, maximum clique length) has increased in one node in the mean. However, due to the media access control protocol in use, that has not presented a significant impact on network performance. An evaluation of the novel particle selection criteria was also performed in the OMNeT++ simulator. In this case, a set of video transmissions being disseminated among user nodes was been forwarded by relay nodes, which were forced to failure. In these simulations, the proposed formation construction has proved to reach manageable solutions as, after nodes' failure, the proposed solution was able to correct routing policies and reestablish connection in every failure occurrence; on the other hand, a solution based on KIM; LEE (2018b) was able to correct in one case only. The results also have shown that the considered packet loss was lower in general, as a higher number of packets were delivered within the required delay limit.

As future work, it is planned to conclude the system model firstly proposed by also establishing routing policies based on network performance acquired through the use of INT. In addition, such information would also be considered in the integration algorithm, therefore nodes could be reallocated based on their current performance and the ongoing transmissions. Moreover, the strategy used by the controller node to determine the relay nodes positioning and the nodes' routing policies could be enhanced, by using a more detailed set of information - for instance, energy resources and their individual capabilities. Another possible direction for future work is to mix decentralized strategies of network connectivity control, which can be beneficial for some specific types of missions.

REFERENCES

BASU, P.; REDI, J.; SHURBANOV, V. Coordinated flocking of UAVs for improved connectivity of mobile ground nodes. *In:* MILITARY COMMUNICATIONS CONFERENCE, 2004, Monterey. **Proceedings[. . .]** IEEE, 2004. v.3, p.1628–1634.

BEKMEZCI, I.; SAHINGOZ, O. K.; TEMEL Şamil. Flying Ad-hoc Networks (FANETs): a survey. **Ad Hoc Networks**, Amsterdam, v.11, n.3, p.1254 – 1270, 2013.

BELDING-ROYER, E. M.; PERKINS, C. E. Evolution and future directions of the ad hoc on-demand distance-vector routing protocol. **Ad Hoc Networks**, Amsterdam, v.1, n.1, p.125 – 150, 2003.

BURDAKOV, O. *et al.* Relay positioning for unmanned aerial vehicle surveillance. **The International Journal of Robotics Research**, Los Angeles, v.29, n.8, p.1069–1087, 2010.

CHOUDHARY, G.; SHARMA, V.; YOU, I. Sustainable and secure trajectories for the military Internet of Drones (IoD) through an efficient Medium Access Control (MAC) protocol. **Computers & Electrical Engineering**, Amsterdam, v.74, p.59–73, 2019.

CHRIKI, A. *et al.* FANET: communication, mobility models and security issues. **Computer Networks**, Amsterdam, v.163, p.106877, 2019.

CUMINO, P. *et al.* Cooperative UAV scheme for enhancing video transmission and global network energy efficiency. **Sensors**, Basel, v.18, n.12, p.4155, 2018.

DE RANGO, F. *et al.* Scalable and ligthway bio-inspired coordination protocol for FANET in precision agriculture applications. **Computers & Electrical Engineering**, Amsterdam, v.74, p.305–318, 2019.

DENGIZ, O.; KONAK, A.; SMITH, A. E. Connectivity management in mobile ad hoc networks using particle swarm optimization. **Ad Hoc Networks**, Amsterdam, v.9, n.7, p.1312–1326, 2011.

DIJKSTRA, E. W. A note on two problems in connexion with graphs. **Numerische mathematik**, Berlin, v.1, n.1, p.269–271, 1959.

ERDELJ, M. *et al.* Help from the sky: leveraging uavs for disaster management. **IEEE Pervasive Computing**, Washington, v.16, n.1, p.24–32, 2017.

FADLULLAH, Z. M. *et al.* A dynamic trajectory control algorithm for improving the communication throughput and delay in UAV-aided networks. **IEEE Network**, New York, v.30, n.1, p.100–105, 2016.

GROSSMAN, E. **Deterministic networking use cases**. Wilmington: IETF, 2018.

GUPTA, L.; JAIN, R.; VASZKUN, G. Survey of important issues in UAV communication networks. **IEEE Communications Surveys Tutorials**, New York, v.18, n.2, p.1123–1152, 2016.

HANDIGOL, N. *et al.* I know what your packet did last hop: using packet histories to troubleshoot networks. *In:* USENIX SYMPOSIUM ON NETWORKED SYSTEMS DESIGN AND IMPLEMENTATION (NSDI 14), 11., 2014, Seattle. **Proceedings[. . .]** USENIX Association, 2014. p.71–85.

HAUERT, S. *et al.* Ant-based swarming with positionless micro air vehicles for communication relay. **Swarm Intelligence**, Amsterdam, v.2, n.2-4, p.167–188, 2008.

KARAAGAC, A.; DE POORTER, E.; HOEBEKE, J. In-band network telemetry in industrial wireless sensor networks. **IEEE Transactions on Network and Service Management**, New York, v.PP, n.X, p.1–1, 2019.

KARACA, Y. *et al.* The potential use of unmanned aircraft systems (drones) in mountain search and rescue operations. **The American Journal of Emergency Medicine**, New York, v.36, n.4, p.583 – 588, 2018.

KEARY, T. **What is Software Defined Networking (SDN) and why is it important?** 2018.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. *In:* INTERNATIONAL CONFERENCE ON NEURAL NETWORKS, 1995, Washington. **Proceedings[. . .]** IEEE, 1995. p.1942–1948.

KIM, C. *et al.* In-band network telemetry via programmable dataplanes. *In:* ACM SIGCOMM, 2015, London. **Proceedings[. . .]** ACM, 2015.

KIM, D.; LEE, J. Integrated topology management in flying ad hoc networks: topology construction and adjustment. **IEEE Access**, New York, v.6, p.61196–61211, 2018.

KIM, D.-Y.; LEE, J.-W. Joint mission assignment and location management for UAVs in mission-critical flying ad hoc networks. *In:* INTERNATIONAL CONFERENCE ON INFORMATION AND COMMUNICATION TECHNOLOGY CONVERGENCE (ICTC), 8., 2018, Jeju City. **Proceedings[. . .]** IEEE, 2018. p.323–328.

KIM, G.-H. *et al.* Multi-drone control and network self-recovery for flying Ad Hoc Networks. *In:* INTERNATIONAL CONFERENCE ON UBIQUITOUS AND FUTURE NETWORKS (ICUFN), 2016, Vienna. **Proceedings[. . .]** IEEE, 2016. p.148–150.

KIRICHEK, R. *et al.* Software-defined architecture for flying ubiquitous sensor networking. *In:* INTERNATIONAL CONFERENCE ON ADVANCED COMMUNICATION TECHNOLOGY (ICACT), 2017, Pyeongchang. **Proceedings[. . .]** IEEE, 2017. p.158–162.

KOTT, A.; ALBERTS, D. S. How do you command an army of intelligent things? **Computer**, Washington, v.50, n.12, p.96–100, 2017.

KOTT, A.; SWAMI, A.; WEST, B. J. The internet of battle things. **Computer**, Washington, v.49, n.12, p.70–75, 2016.

Kreutz, D. *et al.* Software-defined networking: a comprehensive survey. **Proceedings of the IEEE**, New York, v.103, n.1, p.14–76, 2015.

LANDMARK, L.; LARSEN, E.; KURE, O. **Traffic control in a heterogeneous mobile tactical network with autonomous platforms**. Kjeller: Norwegian Defence Research Establishment, 2018.

LEE, J.-H.; MIN, B.-M.; KIM, E.-T. Autopilot design of tilt-rotor UAV using particle swarm optimization method. *In:* INTERNATIONAL CONFERENCE ON CONTROL, AUTOMATION AND SYSTEMS, 2007, Seoul. **Proceedings[...]** IEEE, 2007. p.1629–1633.

LI, S.; SUN, X.; XU, Y. Particle swarm optimization for route planning of unmanned aerial vehicles. *In:* INTERNATIONAL CONFERENCE ON INFORMATION ACQUISITION, 2006, Weihai. **Proceedings[...]** IEEE, 2006. p.1213–1218.

LUO, J. *et al.* A many-objective particle swarm optimizer based on indicator and direction vectors for many-objective optimization. **Information Sciences**, Amsterdam, v.514, p.166–202, 2020.

MAGÁN-CARRIÓN, R. *et al.* Optimal relay placement in multi-hop wireless networks. **Ad Hoc Networks**, Amsterdam, v.46, p.23–36, 2016.

MAGÁN-CARRIÓN, R. *et al.* A dynamical relay node placement solution for MANETs. **Computer Communications**, Amsterdam, v.114, n.October, p.36–50, 2017.

MAHMOUD, S. *et al.* UAV and WSN softwarization and collaboration using cloud computing. *In:* SMART CLOUD NETWORKS SYSTEMS (SCNS), 2016, Dubai. **Proceedings[...]** IEEE, 2016. p.1–8.

MAIER, H. *et al.* Introductory overview: optimization using evolutionary algorithms and other metaheuristics. **Environmental Modelling & Software**, Amsterdam, v.114, 2018.

MARQUES, J. A. *et al.* An optimization-based approach for efficient network monitoring using in-band network telemetry. **Journal of Internet Services and Applications**, New York, v.10, n.1, 2019.

MARTÍNEZ, C. M.; CAO, D. Integrated energy management for electrified vehicles. *In:* **Ihorizon-enabled energy management for electrified vehicles**. Amsterdam: Elsevier, 2019. p.15–75.

MCKEOWN, N. *et al.* OpenFlow: enabling innovation in campus networks. **SIGCOMM Comput. Commun. Rev.**, New York, v.38, n.2, p.69–74, 2008.

MIN, B. C. *et al.* Finding the optimal location and allocation of relay robots for building a rapid end-to-end wireless communication. **Ad Hoc Networks**, Amsterdam, v.39, p.23–44, 2016.

MORAES, R. S.; FREITAS, E. P. Distributed control for groups of unmanned aerial vehicles performing surveillance missions and providing relay communication network services. **Journal of Intelligent & Robotic Systems**, New York, 2017.

ORFANUS, D.; FREITAS, E. P. de; ELIASSEN, F. Self-organization as a supporting paradigm for military UAV relay networks. **IEEE Communications Letters**, New York, v.20, n.4, p.804–807, 2016.

OUDIRA, H.; DJOUANE, L.; GARAH, M. Optimization of suitable propagation model for mobile communication in different area. **International Journal of Information Science and Technology**, Singapore, 2019.

PÓLKA, M.; PTAK, S.; KUZIORA, L. The use of UAVs for search and rescue operations. **Procedia Engineering**, Amsterdam, v.192, p.748 – 752, 2017.

POULARAKIS, K.; IOSIFIDIS, G.; TASSIULAS, L. SDN-enabled tactical ad hoc networks: extending programmable control to the edge. **IEEE Communications Magazine**, New York, v.56, n.7, p.132–138, 2018.

PROGRAMMERSOUGHT. **P4 and programmable network**. Available in: <https://www.programmersought.com/article/9347794046>, Accessed in: 2019 jan 21.

REINA, D. G.; TORAL, S. L.; TAWFIK, H. UAVs deployment in disaster scenarios based on global and local search optimization algorithms. *In:* INTERNATIONAL CONFERENCE ON DEVELOPMENTS IN ESYSTEMS ENGINEERING (DESE), 2016, Liverpool. **Proceedings[. . .]** IEEE, 2016. p.197–202.

ROSÁRIO, D. *et al.* An OMNeT++ framework to evaluate video transmission in mobile wireless multimedia sensor networks. *In:* INTERNATIONAL CONFERENCE ON SIMULATION TOOLS AND TECHNIQUES, 2013, Brussels. **Proceedings[. . .]** ICST, 2013. p.277–284.

ROSATI, S. *et al.* Dynamic routing for flying ad hoc networks. **IEEE Transactions on Vehicular Technology**, New York, v.65, n.3, p.1690–1700, 2016.

SAADAOUI, H.; BOUANANI, F. E. Information sharing in UAVs cooperative search based on calculating the minimum time. *In:* INTERNATIONAL CONFERENCE ON SMART DIGITAL ENVIRONMENT, 2017, New York. **Proceedings[. . .]** ACM, 2017. p.168–173.

SAADAOUI, H.; EL BOUANANI, F. Information sharing based on local PSO for UAVs cooperative search of unmoved targets. *In:* INTERNATIONAL CONFERENCE ON ADVANCED COMMUNICATION TECHNOLOGIES AND NETWORKING (COMMNET), 2018, Marrakech. **Proceedings[. . .]** IEEE, 2018. p.1–6.

SÁNCHEZ-GARCÍA, J.; REINA, D.; TORAL, S. A distributed PSO-based exploration algorithm for a UAV network assisting a disaster scenario. **Future Generation Computer Systems**, Amsterdam, v.90, p.129–148, 2019.

SCALES, L. E. Fundamentals of unconstrained optimization. *In:* **Numerical Optimization**. New York: Springer New York, 2006. p.10–29.

SCHERER, J. *et al.* An autonomous multi-UAV system for search and rescue. *In:* WORKSHOP ON MICRO AERIAL VEHICLE NETWORKS, SYSTEMS, AND APPLICATIONS FOR CIVILIAN USE, 1., 2015, New York. **Proceedings[. . .]** ACM, 2015. p.33–38.

SECINTI, G. *et al.* SDNs in the sky: robust end-to-end connectivity for aerial vehicular networks. **IEEE Communications Magazine**, New York, v.56, n.1, p.16–21, 2018.

SHAKHATREH, H. *et al.* Efficient 3D placement of a UAV using particle swarm optimization. *In:* INTERNATIONAL CONFERENCE ON INFORMATION AND COMMUNICATION SYSTEMS (ICICS), 2017, Irbid. **Proceedings[. . .]** IEEE, 2017. p.258–263.

SHARAF, A. M.; EL-GAMMAL, A. A. Novel AI-based soft computing applications in motor drives. *In:* **Power electronics handbook**. 3rd ed. Amsterdam: Elsevier, 2011. p.993–1034.

SHARMA, V.; KUMAR, R. G-FANET: an ambient network formation between ground and flying ad hoc networks. **Telecommunication Systems**, New York, v.65, n.1, p.31–54, 2017.

SHARMA, V. *et al.* Efficient management and fast handovers in software defined wireless networks using UAVs. **IEEE Network**, New York, v.31, n.6, p.78–85, 2017.

STEWART, K. *et al.* **Introducing network design concepts**. Indianapolis: Cisco Press, 2008.

TANG, F. **A low-cost and resilient flow monitoring framework in SDN**. 2019. Tese (Doutorado em Engenharia Elétrica) — Dalhousie University, 2019.

THOMAS, J.; LAUPKHOV, P. Tracking packets' paths and latency via int (in-band network telemetry). *In:* P4 WORKSHOP, 2016, Stanford. **Proceedings[. . .]** P4, 2016.

TROTTA, A. *et al.* Joint coverage, connectivity, and charging strategies for distributed UAV networks. **IEEE Transactions on Robotics**, Piscataway, v.34, n.4, p.883–900, 2018.

VASHIST, S.; JAIN, S. Location-aware network of drones for consumer applications: supporting efficient management between multiple drones. **IEEE Consumer Electronics Magazine**, New York, v.8, n.3, p.68–73, 2019.

WANG, D.; TAN, D.; LIU, L. Particle swarm optimization algorithm: an overview. **Soft Computing**, Berlin, v.22, n.2, p.387–408, 2018.

WANG, J. *et al.* Taking drones to the next level: cooperative distributed unmanned-aerial-vehicular networks for small and mini drones. **IEEE Vehicular Technology Magazine**, New York, v.12, n.3, p.73–82, 2017.

WICKBOLDT, J. A. *et al.* Software-defined networking: management requirements and challenges. **IEEE Communications Magazine**, New York, v.53, n.1, p.278–285, 2015.

WILSON, P.; MANTOOTH, H. A. Chapter 10 - model-based optimization techniques. *In:* WILSON, P.; MANTOOTH, H. A. (ed.). **Model-based engineering for complex electronic systems**. Oxford: Newnes, 2013. p.347 – 367.

YAO, H. *et al.* NetworkAI: an intelligent network architecture for self-learning control strategies in software defined networks. **IEEE Internet of Things Journal**, New York, v.5, n.6, p.4319–4327, 2018.

ZHAO, Z. *et al.* Software-defined unmanned aerial vehicles networking for video dissemination services. **Ad Hoc Networks**, Amsterdam, 2018.