Wireless Personal Communications
https://doi.org/10.1007/s11277-023-10334-2

An Improved Particle Swarm Optimization Algorithm for UAV Base Station Placement

Faezeh Pasandideh[1,2,3] · Fabricio E. Rodriguez Cesen[5] · Pedro Henrique Morgan Pereira[4] · Christian Esteve Rothenberg[5] · Edison Pignaton de Freitas[1]

Accepted: 25 February 2023

Abstract
In cellular networks, a set of Base Stations (BSs) might be out of service and failed in the aftermath of natural disasters. One of the promising solutions to fix this situation is to send low altitude drones equipped with a small cellular BS (DBSs) to the target locations. This can provide cellular networks with vital communication links and make available temporary coverage for the users in unexpected circumstances. However, finding the minimum number of DBSs and their optimal locations are highly challenging issues. In this paper, a Mixed-Integer Non-Linear Programming formulation is provided, in which the DBSs' location and the proper number of DBSs are jointly determined. An improved PSO-based algorithm is proposed to jointly optimize DBSs' locations and find the minimum number of DBSs. As in the original PSO algorithm, the particles are randomly distributed in the initialization phase and a K-means-based clustering method is employed to generate the positions of the first-generation particles (DBSs). In addition, a custom communication protocol is presented for data exchange between the users' equipment (UE) and the network controller. The proposed approach is evaluated through four simulation experiments implemented using Mininet-Wifi integrated with CopelliaSim. The acquired results show that the proposed solution based on the integration of PSO and K-means algorithms provides a low packet loss and latency. Moreover, it indicates that most of the users in the considered scenarios are covered by the DBSs.

Keywords Drone base station · Particle swarm optimization · K-means algorithm · Deployment problem · Non-linear optimization

1 Introduction

Recently, drone-assisted cellular networks, where multiple drone-mounted BSs are integrated as relays to help the ground Base Stations (BSs), and to improve the ground users quality of service (QoS), have received increasing attention. In some situations, a set

✉ Faezeh Pasandideh
faezeh.pasandideh@inf.ufrgs.br; faezeh.pasandideh@hshl.de

Extended author information available on the last page of the article

Published online: 22 March 2023

of ground Base Stations (BSs) are temporary out of the service and failed in the aftermath of natural disasters, bad weather condition, and sudden congestion in places such as sports stadiums. When the BS failure occurs, re-designing the network can be extremely time-consuming. The packet traffic also tends to increase during such scenarios. On the other hand, reactivating the failed and inaccessible BSs makes the connectivity and topology control a challenging task. Therefore, some actions should be taken to reduce potential effects in existing network infrastructure after happening such unexpected events [1].

One of the promising solutions for the above described problem is to send low altitude Unmanned Aerial Vehicles (UAVs) or Drones equipped with a small cellular BS to the target locations [2]. The drone-mounted BSs (DBSs) provide cellular networks with vital communication links and make available temporary coverage for the users in circumstances of connection shortage. Many studies have investigated the use of UAV carried BSs (UAV-BSs) to improve wireless communications using UAV deployment and resource allocation schemes [3–18].

The above-mentioned literature have shown the network performance improvement is achievable by employing DBSs or UAV-BSs in cellular networks. However, some open challenges still remain when it comes to drone-cells, such as network resilience. Most of existing works are not able to provide an adequate level of network resiliency [19]. They manage the DBSs or UAV-BSs in an uncoordinated manner which suffers from lacked global network information and centralized decision making. The above-mentioned issues should be addressed in more intelligent ways to provide global view and control for deploying DBSs.

Recently, the software-defined network (SDN) has emerged as a promising method to design the future cellular network which provides a few new features to address the aforementioned challenges. By separating the control plane and data plane, a global view of the whole network is available for the SDN controllers. The information gathered at the control plane enables centralized decision making and control of the data plane through the flow-based procedure. Thus the network management is simplified since DBSs are directly programmable and abstracted to the data plane [20]. Several researchers have studied the placement of an SDN-based UAV network [15, 21–24].

However, there are still some open challenges and limitations when SDNs are implemented in UAV networks as they suffer frequent link disconnections. Moreover, it is difficult to identify the most proper route for the UAVs. In addition, in some use cases, UAV devices can be disrupted purposely and on-board resources might be not accessible [25, 26].

Observing this landscape, this paper proposes a method that addresses some of these above discussed concerns. The main contributions of this paper can be summarized as follows:

- The DBS Placement Problem is formulated as a Mixed-Integer Non-Linear Programming (MINLP) and then solved;
- A coverage modeling module and traffic modeling module consisting of a novel queuing modeling is developed within the controller;
- A PSO-based algorithm is proposed to jointly optimize DBSs' locations and to find the minimum number of DBSs;
- The unsupervised ML clustering technique called K-means is employed in the PSO algorithm that predicts the initial value of the number of DBSs more intelligently; and
- A custom communication protocol (CCP) developed for exchanging the UE's main data between UEs and the network controller.

The remainder of the paper is structured as follows. Section 2 discusses relevant related work. Section 3 describes the DBS placement problem formulation. The proposed solution is detailed presented in Sect. 4. Section 5 presents the proposal evaluation and explicitly discusses its validity through numerical results. Section 6 concludes the paper bring directions for future work.

2 Related work

Recently, drones have been used in wireless sensor networks and cellular networks thanks to their resilience and agility. The DBS deployment issue has been mostly influenced by the reliability of the air-to-ground link, which is the main purpose of optimizing the location, height, and number of the DBS in order to maximize the coverage on the network. In this section, traditional and ML-based DBS placement approaches and SDN-based DBS placement solutions are discussed.

Literature first formulate the 3D UAV-BS placement problem by taking into account the maximum allowed air-to-ground path loss (PL), data rate requirements of various users, limited available bandwidth, power constraint, backhaul, and access links resource management, with different goals, such as: to maximize the number of covered users [3–12]; to cope with the inter-cell interference [6]; to maximize the aggregated data of all users (data rate requirements) [13, 14]; to provide continuous connectivity and communication between various UAVs [15]; to maximize network profit [16]; to maximize the spectral efficiency of the whole system [17]; and to minimize the total energy consumption [18].

In [5], authors propose a method to solve the UAV-BS placement which includes a classical Branch and Bound algorithm with relaxation induced neighborhood search (RINS) and a heuristic algorithm to the UAV-BS placement and user association problem in linear time. In [14], authors take advantage of a Genetic Algorithm (GA) to solve the optimization problem in which the UAV-BS radius coverage and its horizontal position are considered as a gene in the GA model. In [16], authors use a low complexity heuristic algorithm called "Golden Section Search" (GSS) algorithm to find the optimal altitude of the UAV-BS. After finding the optimal location of the UAV-BS, the authors reduce the resource allocation problem to a Multidimensional Binary Knapsack Problem (MBKP) that can be solved in polynomial time. In [8], another meta-heuristic algorithm, called Social Spider (SSO) algorithm, is proposed in order to solve the problem and find the optimal location of the UAVs and their association with both GBSs and UEs. A PSO-based algorithm is also presented in [27] to jointly optimize UAVs' locations, their Signal to Interference plus Noise Ratio (SINR) transmit power, and find the minimum number of UAVs.

Recently, scholars take advantage of machine learning techniques for the possitiong issue of UAVs in the UAV-aided wireless networks. In [17], authors solve the optimization problem by proposing a deep reinforcement learning algorithm (DQN), which consists of both reinforcements learning Q-learning and deep learning CNN. DQN calculates the spectral efficiencies and tries to move the UAV-BS to a route or location that has the maximum average spectral efficiency. In [9], authors present a deep reinforcement learning (DRL) control strategy for UAV mobility by taking account to the movement of the users without considering their locations. A deep Q-learning algorithm is used to control the UAVs mobility, and a time-series CNN-based model to

estimate the quality of the link at each time slot. In [10], authors solve the deployment problem of multiple UAVs by adopting a centralized multi-agent Q-learning algorithm. The reward function used in the training process of Q-learning algorithm, is built based on corresponding power consumption and the optimal association scheme for given statistical user distribution and UAVs' state.

In theory the centralized architecture can obtain the best system performance. However, the time complexity of real-time modelling the dynamic network environment is high, since the network scale is expanding. In addition, the communication delay is also high, because the UEs data should be uploaded and processed centrally on the central controller. Therefore, authors in [11] propose a distributed resource allocation algorithm for the UAV networks based on multi-agent collaborative environment learning, and in [12] a federated multi-agent deep deterministic policy gradient (F-MADDPG) based trajectory optimization algorithm is presented. In [28] authors provide a UAV positioning ML-based algorithm to predict overloaded Macro cell based on the temporal and spatial flow of the users. After estimating the congested macro cells, a set of UAV-BSs should fly to the location in order to temporarily establish required links and communication.

Recently, the software-defined network (SDN) has emerged as a promising method to design the future cellular network, as well. Authors in [21–23], and [15] provide 3D multi-UAV SDN-enabled placement optimizations. The SDN architecture of the proposed models consist of a control plane, data plane, and a communication protocol. The OpenFlow v1.5 protocol is considered as the communication protocol between the data plane and the control plane. In the data plane, the authors investigate a high mobility urban topology where each DBS has dynamic resource allocation on demand.

The SDN controller which is the heart of control panel, consists of various modules. In [21] the controller consists of coverage area modeling, traffic modeling, and path planning modules for the DBSs' operations. The nearest neighbor weighted interpolation algorithm is employed in order to estimate the 3D position of the DBS. In [22], the control plane contains a distributed and parallel maximization (AM) iterative algorithm to solve the provided problem. Authors utilize the modified alternating direction method of multipliers (ADMM) and the successive convex optimization (SCO) algorithms to make the 3D UAV deployment problem and user association and scheduling blocks more tractable. In [23], the SDN controller includes two modules; the resource allocation (RA) and authentication and charging (AC) modules. The authors use the concave-convex procedure (CCCP) and bisection search algorithms to solve optimal resource allocation and optimal altitude-to-radius ratio sub-problems. In [15], the authors employ Grey Wolf Optimizer in controller to find the optimal placement of multiple UAVs and to calculate the distance between two UAVs. Authors in [29] present the SDN-based topology management for FANETs (STFANET) based on [30] which tries to maximize the number of interconnected users and minimize link distances. In that case, the network connectivity would still rely on individual nodes, in which their operational failure would compromise ongoing connections and transmissions between users. In other words, the restoration of active communications during failure occurrences might be merely performed by nodes' position replacement, which would demand reallocation time and energy consumption.

To enhance the UAV formation that outcomes from the construction phase of [29], we proposed a new approach in our previous paper [31] to construct a more resilient and manageable topology formation. We considered 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. Also, the spring virtual force method is applied by using attractive-repulsive forces among nodes.

Table 1 Summary of main characteristics of the most related studies

Reference	Addressed Problem	Centralized?	Proposal
[5]	To minimize the number of UAV-BSs deployed while maximizing the number of served served users	No	A classical Branch and Bound induced neighborhood search (RINS) algorithm with relaxation
[14]	To maximize the users' data-rate requirements	No	A Genetic algorithm (GA)
[16]	To maximize the network profit (throughput, latency, coverage and the network profit)	No	The Golden Section Search (GSS) algorithm
[8]	To maximize the users' coverage	No	The Social Spider x users' coverage
[17]	To maximize the spectral efficiency of the whole system	No	The deep reinforcement learning algorithm (DQN)
[9]	To maximize communication coverage and network connectivity for multiple real-time users	Yes	deep reinforcement learning (DRL) movement control strategy
[10]	To provide on-demand coverage with minimum network power consumption	Yes	A centralized multi-agent Q-learning algorithm
[11]	To optimize the coverage and utility	No	Multi-agent collaborative environment learning
[12]	To maximize the average spectrum efficiency	No	Federated multi-agent deep deterministic policy gradient (F-MADDPG) based trajectory optimization algorithm
[28]	To improve the quality of service in regions temporarily supported by UAV-BS	No	Prediction machine learning based algorithm
[21]	To maximize the covered users and communication quality	Yes	The nearest neighbour weighted interpolation method
[22]	To maximize the data rate utility among overall users	Yes	The successive convex optimization (SCO) and the modified alternating direction method of multipliers (ADMM) techniques
[23]	To maximize the number of covered users	Yes	The bisection search and concave-convex procedure (CCCP) methods
[15]	To provide continuous connectivity and communication between various UAVs	Yes	The Grey Wolf Optimizer (GWO)
Our paper	To provide maximum user coverage and minimum latency and packet loss	Yes	Improved K-means based PSO algorithm

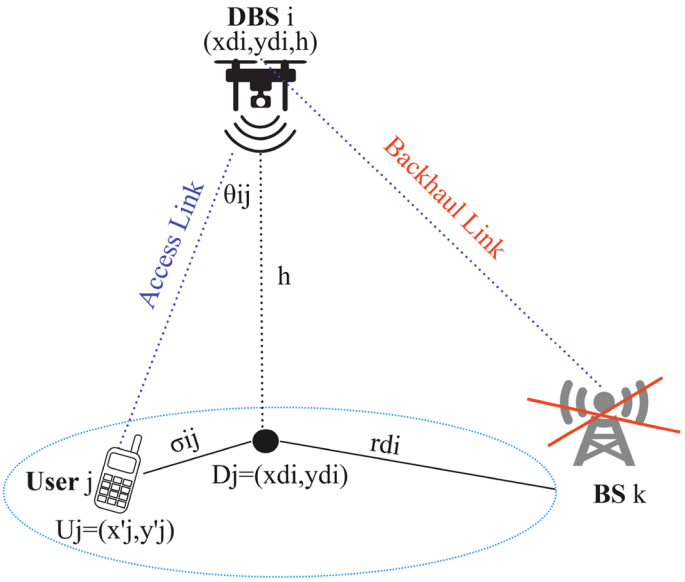

Fig. 1 The system overview

Table 1 summarizes the main aspects of the revised relevant related work.

3 System Model

Consider a very simple UAV-based communication system as represented in Fig. 1 which includes a mobile user ($user_j$), BS (BS_k) and DBS (DBS_i). In some use cases, terrestrial BS_k is out of service and cannot serve the $user_j$. In this case, DBS_i acts as a BS in order to provide temporary communication links for $user_j$. The users are located in an area of radius r_{di} served by a DBS, in this case DBS_i, located at 3D coordinates (x_{di}, y_{di}, h) where the altitude of DBS is h and can be between a minimum and maximum allowed altitude ($h_{min} \leq h \leq h_{max}$). Finding the optimal numbers and coordinates of the DBSs are still open issue which is known as placement problem. Each DBS can cover a maximum of M number of users, and each user for example $user_j$ is located at horizontal position $(x_j^{'}, y_j^{'})$ where $j = 1, 2, \ldots, M$. Each user j is capable of transmitting the data rate T_j and power p_{ij}. This paper only considers the data transmission between users and the DBSs (access link) for the sake of simplicity. Next, the data rate and path loss models in the access link are discussed.

3.1 Data Rate and Path Loss Models in the Access Link

DBS deployment affects the users coverage area as well as the reliability of the Air to Ground (AtG) communication links. Different types of channel models exist in the literature to model the AtG path loss. In this paper, the considered channel model is the one presented in [32] because of its well-proven results. Depending on the propagation environment, air-to ground communication links can be either LoS or non-line-of-sight (NloS).

Table 2 Summary of all notations

Symbols	Description
ζ_{ij}	The average pathloss between user j and DBS i
R_{ij}	The data rate between user j and DBS i
T_j	The data rate requirements of user j
B_{ij}	The sub-channel bandwidth allocated by the DBS i to user j
p_{ij}	The transmit power allocated by the DBS i to user j
f_c	The fronthaul carrier frequency
c	The speed of light
h	The altitude of the DBS
D_{ij}	3D distance between user j and DBS i
σ_{ij}	The horizontal distance between the user j and DBS i
ω	A noise figure
M	The maximum number of users covered by the DBS
$NDBS$	The number of DBSs
$E[w_{ij}]$	The waiting time of user j in the queue inside the DBS i
X_{ij}	A binary variable to determine if user j is covered by DBS i
r_{di}	The radius of the DBSs i
P_{max}	The maximum transmission power allocated by a DBS
B_{max}	The maximum bandwidth allocated by a DBS
ε^2	The noise power spectral density
U_j	The horizontal position of the user j which is $(x_j^{'}, y_j^{'})$
DBL_j	The battery level of user j
DTR_j	The traffic requested by user j
DT_j	The type of user (device) j, soft real-time or hard real-time

LoS connection probability between the receiver and transmitter is an important factor, as it affects the users' power utilization. The probability of having a LoS connection between user and DBS depends on different factors, such as building density, user's location, and DBS elevation angle between the user and DBS. The probability of having a LoS and NLOS connections between user j and the DBS i are calculated according to [32].

The data rate between user j and DBS i can be calculated as follows based on the path loss model provided in [32]:

$$R_{ij} = B_{ij} * \log_2\left(1 + \frac{p_{ij} * 10^{\frac{-\zeta_{ij}}{10}}}{B_{ij} * \varepsilon^2}\right) \tag{1}$$

where B_{ij} and p_{ij} are sub-channel bandwidth, transmit power allocated by the DBS i to user j and ε^2 is the noise power spectral density of the Zero-mean white Gaussian noise at the receiver. ζ_{ij} is the avarage pathloss between user j and DBS i which is calculated according [32].

The descriptions of all the notations are presented in Table 2.

3.2 Placement Problem Formulation

The initial number of required DBSs (*NDBS*) to start serving a set of users in an area can be defined as the total number of users in the network divided by the maximum number of users that each DBS covers. Then using intelligent algorithms, the optimal number of DBSs can be estimated. Considering the maximum number of users that a DBS can serve, M_i,

$$NDBS = \frac{TotalUsers}{M_i} \tag{2}$$

Considering the sum of data rate requirements of users, T and the capacity of the DBS, *Capacity* then:

$$M_i = \left\lfloor \frac{T}{Capacity} \right\rfloor \tag{3}$$

Therefore, the optimization problem is formulated as follows:

$$min_{x_{di}, y_{di}, h, X_{ij}} \sum_{i=1}^{NDBS} \sum_{j=1}^{M} E[w_{ij}] X_{ij} \tag{4}$$

Subject to

$$R_{ij} \geq T_j : \forall j \in M \tag{5}$$

$$\sum_{j \in M} R_{ij} \leq Capacity \tag{6}$$

$$x_{min} \leq x_{di} \leq x_{max} \tag{7}$$

$$y_{min} \leq y_{di} \leq y_{max} \tag{8}$$

$$h_{min} \leq h \leq h_{max} \tag{9}$$

$$\sum_{i=1}^{NDBS} X_{ij} \leq 1 : \forall \in \{1, 2, \ldots, M\} \tag{10}$$

$$\sum_{j=1}^{M} B_{ij} = B_{max} \tag{11}$$

$$\sum_{j=1}^{M} p_{ij} = P_{max} \tag{12}$$

$$r_{di} < r_{max} : i \in \{1, 2, \ldots, NDBS\} \tag{13}$$

The problem formulated by (4) aims to minimize the number of required DBSs (NDBS) by minimizing the waiting time of the users in the queue. $E[w_{ij}]$ is the waiting time of user j in

the queue inside the DBS i. X_{ij} is a binary variable ($X_{ij} \in \{0, 1\}$) that determines if user j is covered by DBS i or not. A user can be served by a DBS if horizontal euclidean distance between the user and DBS is less than or equal to DBS's coverage radius (r_{di}):

$$X_{ij} = \begin{cases} 1 & \text{if } \sigma_{ij} \leq r_{di} \\ 0 & \text{if } \sigma_{ij} > r_{di} \end{cases} \tag{14}$$

Constraint (5) indicates that QoS requirement of each user should be satisfied which R_j is the data rate between user j and DBS and T_j is the data requirement of user j. Constraint (6) shows that total data rate of all covered users served by one DBS cannot exceed the data rate capacity of that DBS. Constraints (7), (8), and (9) indicate the placement region of the 3D coordinates of DBSs which $x_{min}, x_{max}, y_{min}$ and y_{max} are limits of the area and h_{min} and h_{max} are the minimum and maximum altitude of a DBS allowed to reach. Constraint (10) ensures that each user should be served at most by one DBS. Constraint (11) and (12) show the resource limitation which B_{ij} is bandwidth allocated by DBS i for user j and p_{ij} is transmission power allocated by DBS i for user j. Constraint (13) indicates that radii of DBSs i (r_{di}) is no longer than the maximum radii.

3.3 Linearizing the optimization problem

The problem P, which is the constraint-based mixed-integer programming problem, can be reformulated as:

$$\begin{aligned} f(x_{di}, y_{di}, h, NDBS) = min_{x_{di}, y_{di}, h} \sum_{i=1}^{NDBS} \sum_{j=1}^{M} E[w_{ij}] X_{ij} \\ + Constraints. \end{aligned} \tag{15}$$

where $f(x_{di}, y_{di}, h, NDBS)$ is the unconstrained cost function and *Constraints* can be shown as:

$$\begin{aligned} Constraints = \Phi * \left(\sum_{k=1}^{9} \alpha_k + \left(P_{max} - \sum_{j=1}^{M} p_{ij} \right) \right. \\ \left. + \left(B_{max} - \sum_{j=1}^{M} b_{ij} \right) \right) \end{aligned} \tag{16}$$

According to problem P (4), the constraints (5–13) contain equals signs and comparison operators. Regarding the equals sign we need to multiply a large number (Φ) by the subtraction value of the expressions on the two sides of the equals sign and add the result to the cost function. In this case the $\Phi * (P_{max} - \sum_{j=1}^{M} p_{ij})$ and $\Phi * (B_{max} - \sum_{j=1}^{M} b_{ij})$ are added into the cost function. Regarding comparison operators, in this case greater than or equal to ($\geq$) and less than or equal to ($\leq$), the subtraction value of both sides of the equation is not highly important, but satisfying the equation is essential. Thus, a new binary variable α_k is defined to determine whether the given equation is satisfied:

$$\begin{cases} \alpha_k = 1 & \textit{if the equation is not satisfied} \\ \alpha_k = 0 & \textit{Otherwise} \end{cases} \tag{17}$$

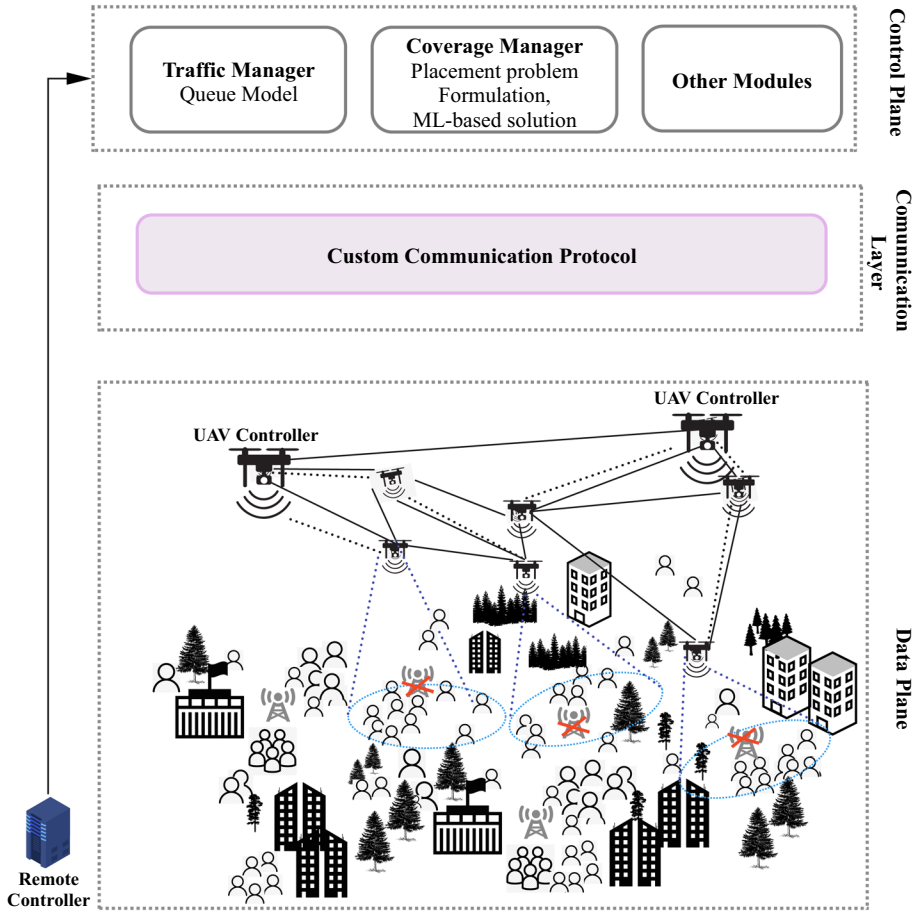

Fig. 2 Proposed network system architecture for DBSs placement

when the equation is not satisfied a penalty value should be imposed to the cost function. Therefore $(\alpha_k * \Phi)$ is added into the cost function.

The next section explains the provided solution to solve this formulated placement problem.

4 Proposed Solution

4.1 Proposal Overview

UAV deployment is an open issue in FANET-based scenarios. Due to regulatory issues, UAVs are not allowed to fly over all regions and have altitude constraints. Therefore, the UAV deployment in the vertical and horizontal dimensions is a challenging task. Motivated by these facts, this work proposes a conceptual design of a system for DBSs placement which is shown in Fig. 2. This proposal uses a SDN-based approach to manage the network resources so that the users can be properly served by the DBSs.

As can be observed in Fig. 2, in the *data plane* there is an urban topology including mobile users, BSs and DBSs. The terrestrial BSs are out of service and cannot serve these users. Then, the controller that has a global view of the whole network will enable programming the DBSs on the fly by managing network resources. The DBSs have a dedicated link with the controller to reduce backbone load and latency (*control plane*).

The system architecture is designed so that the controller runs multiple modules, which controls the traffic modeling and coverage area. Moreover, it runs a PSO-based algorithm to jointly optimize the DBSs' locations and to find the minimum number of DBSs. Finally, a custom communication protocol (CCP) is used for exchanging the UE's main data between UEs and the network controller. This section continues describing the details of these parts of the proposed solution.

4.2 System Architecture

This section describes the Traffic modeling module, the proposed communication protocol and PSO-based approach to solve the placement problem described in Sect. 3.2. According to Fig. 2, the proposed network architecture contains three main layers: the *Data layer* includes *NDBS* number of DBSs and *TotalUser* number of users, which are uniformly distributed in the topology with the locations given by $U_j = (x_j^{'}, y_j^{'})$; the *Control layer* that runs multiple modules, such as the traffic modeling module and coverage area modeling module; and the *Communication layer* which connects data layer to control layer. The elements that constitute these layers are described in the following.

4.2.1 Traffic Modeling Module

The network is composed of a number *NDBS* of DBSs each that can support a maximum number of users *M*. Both *NDBS* and *M* are known by every DBS and the controller, such as the traffic request (packet/s) for UE within the network. The network is composed of an aerial controller, multiple DBSs, and UEs (also known as nodes). The UEs are classified into two categories: Online or Disconnected.

Online devices (OD) are the ones that can communicate keeping a QoE above a threshold value, in which both values are related to the waiting time until a package is dropped. On the other hand, the Disconnected devices (DD) only exist when a device tries to connect to a DBS which capacity, *M*, is already full (exceed backhaul capacity). This device will be placed in a queue. If it remains in the queue for longer than a threshold value queue priority will be given to it.

Every UE must share at least four valuable data: local position, battery level, traffic request (number of packets per second), and the type of application (hard or soft real-time), which can be observed in Fig. 3. These data will be used to create the queues and to allocate the nodes to the respective queue's best position.

The U_j is horizontal position of the user j which is represented by $(x_j^{'}, y_j^{'})$. U_j is used to create the map with every DBS and user connected to the network.

The battery level of the user (DBL_j) is the percentage of the remaining energy of the user's battery. It is used to prioritize the communication for devices with values lower than a threshold, and place DDs in a privileged queue position due to the low battery levels. The traffic request (DTR_j) is the number of packets per second requested by user j. The maximum waiting time, and the mean queue length are calculated based on the DTR_j, such

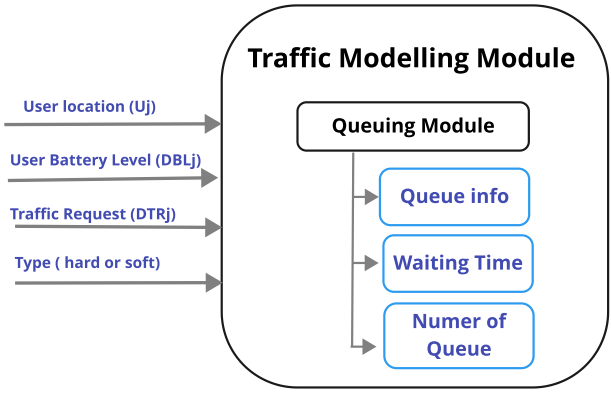

Fig. 3 Traffic Modelling Module Representation

calculation will be shown throughout this section. Each user or device j should have a type (DT_j) that can be soft real-time or hard real-time, depending on its application. There is one queue for each DT_j, therefore there are two sub-queues for each DBS, a hard real-time queue with a higher priority, and a soft real-time queue with lower priority.

Queuing module:

The queuing model plays a vital role in the choice of the users who can communicate with the controller. DBL_j and DTR_j are two main time-varying parameters in our queue model. The queuing module is proposed to control the OD's packages based on the delay tolerance for each DBS traffic demands. Therefore, in proposed queuing module, the mean queue length (MQL) for each DBS is calculated, which is the sum of the MQL for the soft real-time queue and the MQL for the hard real-time queue, in order to improve the Quality of Experience (QoE) for all users. For each DBS there are two possible queues, one specifically for hard real-time and the other one for soft real-time applications. The queue position of each user, for example user j, is defined based on its type (DT_j) and its battery level (BTL_j) If the BTL_j is higher than a threshold value, for example, 20 percent, it will be allocated to the last position of its respective queue. Otherwise, the battery level of the new user k (BTL_k) will be compared with the users that are already in the queue until the best position for this device is found.

4.2.2 Custom Communication Protocol

Already established communication protocols for similar applications were studied, such as the OpenFlow protocol. However, it was chosen to develop a Custom Communication protocol (CCP) in this work since it would be lighter and would allow for better network scalability than OpenFlow. In other words, OpenFlow provides a large set of functionalities that all of them are not necessary and useful for the proposes of this work. Therefore to skip those functionalities we use CPP which has a smaller set of functionalities compared with Openflow. CCP is developed for exchanging the UEs' main data between UEs and the network controller. Therefore, a tunneling protocol is used to encapsulates the CCP inside the UDP data. The CCP is divided into two sections, the header and data.

The CCP header section is composed of seven bytes, the first is the start byte that indicates the start of the CCP. The second one is the "Lenght" which is the number of bytes sent in the CCP's data section. The third byte is the sequence number, which represents the number of the message sent, to analyze packet loss. The fourth is the source address that is the name of the UE that sent the message. The "Dest ADD", the sixth byte, is the name of the device that

should receive the message. The sixth byte is the Mode of the UE, which can be defined as one or two. If the mode is set to one, the UE will send a message to the controller, which will save the UE's data, and it will wait for a response. However, if the UE's mode is set to 2, the message should be received by the UE which name is set in the "Dest ADD" byte. The seventh and last byte is the ENC, if it is set to one, it represents the end of the communication, the last message, between the UE and the rest of the network nodes.

The data section of the CCP is composed of at least six bytes, the first two respectively representing the UE's local position on the x-axis and y-axis. The third represents the UE's battery level, the fourth the UE's type of application. The type can be defined as one for hard real time application and zero for soft real time application. The fifth byte is the traffic request that represents the frequency of packets sent by the device (packets/s). And the sixth and last mandatory byte represents the max time that the UE can wait in the queue. If necessary more bytes can be sent within the data section, for example for more robust testing of the network.

4.2.3 Proposed PSO-Based Algorithm for Coverage Module Model

Particle Swarm Optimization is a computational method that optimizes a problem by iterative enhancing the candidate solution and discovering the global optimum. In this paper, PSO algorithm optimizes the placement problem of DBSs by using the sets of candidate DBS positions called particles and moving the swarm of particles, which represent potential solutions, around in the search-space according to simple mathematical formulae over the particle's position, velocity, and cost value. Each particle's movement is influenced by the best position the particle has experienced (local best position) and the best position that all the particles have experienced (global best position). Therefore, each particle (DBS) adjusts its flight according to its own flying experience and companion's flying experience. This is expected to move the swarm toward the best solutions [29, 31].

However, PSO suffers from trapping in the local minimum or finding the best global minimum in some problems. The particles are not successful to cover the entire search space, as they are distributed randomly in initialization phase using Gaussian or uniform distribution. It is expected that by improving the initialization phase, the final results of PSO would be more accurate. Therefore, it is important to improve the initial population generation phase to cover the feasible space properly. There are some investigations that take advantage of other optimization methods to reinforce the exploitation and exploration phases of the PSO algorithm, such as [33–35]. Authors in [36, 37] try to improve the population initialization step. In the following, the proposed PSO-Based algorithm is explained in detail.

The first step of the PSO algorithm is to initialize the PSO parameters, including, acceleration coefficients (c_1, c_2), random vectors (r_2, r_2) and inertia weight (w), and the number of population ($npop$) which is the number of DBSs($NDBS$). The basic PSO algorithm randomly determines these parameters. However, they can be initialized more accurately to enhance the PSO parameters. Regarding $npop$ (the number of DBSs) if the scheme starts with one DBS it is not desirable as the number of PSO iterations will increase. Thus a more accurate number of DBSs is needed to reduce the number of PSO iterations which is already calculated in Sect. 3.2. The next step is to initialize the positions of DBSs (particles). The initial generation is commonly randomly generated in the PSO algorithm. However, such random initialization suffers from the less satisfactory performance. More specifically, the K-means clustering-based algorithm can be employed to determine the initialized positions of the DBSs in the area.

The goal of the PSO algorithm is to locate the DBDs in a 2D plane and find (x_{di}, y_{di}) based on the linearized optimization problem in Sect. 3.3. After calculating the cost

function for each particle based on Sect. 3.3, then current PSO iteration time, particle's local best, and global best are updated based on this function. Next, the velocity of DBSs located at the new positions is updated and the velocity limits (v_{min} and v_{max}) are applied. To make sure that all the particles stay inside the search space, velocity mirror effects are avoided which means if a particle is outside the search space, it should be moved back inside. After updating the personal and global best of each DBS, the PSO algorithm will terminate if the swarm met the termination criteria. If it terminates, the optimal position of the DBS is obtained then check if the waiting time of the users in the queue is minimized or not. If it is not minimized, it is realized that the number of DBSs is not sufficient, then this number is increased by one and perform the steps from scratch.

(a) Improved population initialization phase

In the original PSO algorithm, the particles are distributed randomly in the initialization phase. The proper initialization of the first-generation particles can improve the performance of the PSO algorithm. A K-means based clustering method can be proposed as an initialization method to generate the positions of the first-generation particles (DBSs).

- **K-means:** More specifically, the K-means algorithm is an iterative algorithm that tries to partition data points (users) into K pre-defined distinct non-overlapping clusters where each user belongs to only one cluster. It assigns users to a cluster such that the sum of the squared distance between the users and the cluster's centroid is at the minimum [38]. The centroid is the arithmetic mean of all the users that belong to that cluster. The Cluster head or centroid, in this case refers to the DBS.

  The following problem has to be solved:

$$min_x \sum_{j=1}^{TotalUsers} \sum_{k=1}^{K} x_{jk} \left\| U_j - \mu_k \right\|^2 \tag{18}$$

  subject to

$$\sum_{k=1}^{K} x_{jk} = 1 \forall j \tag{19}$$

$$\mu_k = \frac{\sum_{j=1}^{TotalUsers} x_{jk} U_j}{\sum_{j=1}^{TotalUsers} x_{jk}} \tag{20}$$

$$x_{jk} \in \{0, 1\} \forall j, k \tag{21}$$

  where x_{jk} is a binary varible determines if user j belongs to cluster k ($x_{jk} = 1$) or not ($x_{jk} = 0$). U_j and μ_k are the coordinates of jth user and the centroid of user j's cluster, respectively. They are both located in $\mathbb{R}^d$, where d is the dimensional of users. Constraint (19) indicates that each user should be assigned to exactly one cluster. Constraint (20) shows that the coordinates of centroid of cluster k depend on values of x_{jk} and U_j variables. The problem (18) which contains these non-linear constraints can be rewritten as the following problem:

$$min_x \sum_{j=1}^{TotalUsers} \sum_{k=1}^{K} x_{jk} \left\| U_j - y_k \right\|^2 \tag{22}$$

subject to

$$\sum_{k=1}^{K} x_{jk} = 1 \forall j \tag{23}$$

$$\mu_k = \frac{\sum_{j=1}^{TotalUsers} x_{jk} U_j}{\sum_{j=1}^{TotalUsers} x_{jk}} \tag{24}$$

$$x_{jk} \in \{0, 1\} \forall j, k \tag{25}$$

$$y_k \in \mathbb{R}^d \forall k \tag{26}$$

The problem represented in (22) shows that instead of minimizing the distance to centroids (μ_k), the idea is to minimize the distance to just any set of points (y_k) that will give a better solution based on results. It turns out that these points are exactly the centroids. To solve the objective function, first the values for y_k variables are fixed and the optimal values for x_{jk} variables are found, then the values of x_{jk} variables are fixed, and the optimal values for y_k variables are found.

The proposed algorithm is shown in Algorithm 1. As can be observed in the algorithm, the positions of the users X_U is considered as an input.

Algorithm 1: Proposed algorithm.

Input: X_U.
Output: X_D and $NDBS^*$.
Calculate the initial value of $NDBS$ according to (5).
Generate first generation:$initPosDBS = k - means(NDBS)$
repeat
  for *each particle* (DBS) **do**
    Initialize PSO parameters: Initialize w, c_1 and c_2 according to [45], and r_1, r_2 and $velocity$
    Calculate the $Cost\ function$ according to (18)
    Update p_best and g_best.
  end
  repeat
    for *each particle DBS* **do**
      Update $velocity$
      Update $DBS\ velocity\ limit$.
      Update $Costfunction$.
      Update p_{best} and g_{best}.
    end
  until *The swarm met the termination criteria*;
  $X_D \leftarrow g_{best}$;
  $NDBS^* \leftarrow NDBS$;
  $++NDBS$
until *The* $E[w_{ij}]$ *is obtained*;

The output is the optimal position and optimal number of DBSs, respectively, X_D and $NDBS^*$ resulted by the proposed algorithm. After calculating the initial value of $NDBS$

according to (2), the initial positions of the particles (DBSs) are generated using K-means algorithm. The PSO parameters including w, c_1 and c_2 are initialized according to (XXXX),(XXXX) and (XXXX) respectively, the remaining parameters such as r_1, r_2 and *velocity* are initialized in line 5. Then the *Costfunction* is calculated according to (15) in line 6. Finally the personal best and global best are updated. Then the PSO algorithm iteratively runs updating the velocity and the velocity limit of the particles (line 11 and line 12). The cost function is obtained by the formulation provided in (15) and updated in line 13. Then PSO updates the individual local best solution of particles. The global solution is continuously updated (line 14), as well. The repetition finishes if the swarm met the termination criteria. Then the output of the algorithm X_D and $NDBS^*$ is obtained. Finally, if the $E[w_{ij}]$, the waiting time of $user_j$ in the queue inside the DBS_i, is satisfied, the the algorithm terminates, otherwise, the algorithm run from scratch with one more DBS ($NDBS++$).

5 Evaluation

5.1 Simulation Setup

The proposed network architecture was implemented using Mininet-Wifi wireless network emulator and CoppeliaSim.

The robot simulator CoppeliaSim represents an intuitive environment to create various virtual worlds by providing different types of robots, objects, structures, sensors, and actuators. Each object/model can be individually controlled via an embedded script, a plugin, a ROS or BlueZero node, a remote API client, or a custom solution.[1]

In this paper, the CoppeliaSim is used only for the representation of the DBSs and user-nodes. Some physical configurations for DBSs, such as the number of engines, the velocity, and the position, are defined inside the CappeliaSim. Instead of having all the actions and algorithms inside the CopelliaSim, a Remote API is provided which is inside the Mininet-Wifi, connecting to the CoppeliaSim and doing all the calculations and actions for the controlled nodes. In other words, the remote API acts as a controller, calculating the best positions for the DBSs, providing mobility and movement models, then sending all the information to the CoppeliaSim to verify if DBSs are located at the best places, the maximum number of users are covered and so forth. Mininet-Wifi was used to emulate the data plane (network scenario) that run virtual nodes, such as stations, hosts, access points, switches, and their links and characteristics. Figure 4 shows the implementation tools and their relations.

5.2 Simulated Scenarios

Consider a scenario where a set of ground base stations are out of the service and cannot serve all users due to temporary events such as natural disasters, happening of sudden

[1] https://www.coppeliarobotics.com/.

congestion in places such as sports stadiums, etc. Hence, multiple DBSs are immediately sent to the target location and establish the necessary communication links and cover that area.

The proposed approach is evaluated with three sets of users 15, 75 and 150 mobile users which are able to move in the environment with a square shape ($100 \times 100\ m^2$). The value of users speed is variable and it is calculated based on the users' position over time, ranging from 5 to 10 m/s. The simulation was performed for one hour long, which was considered to be enough in order to have the users sufficiently spread through the environment. Then, the number of DBSs and their optimal locations are achieved in each scenario using the proposed algorithm.

The full set of parameters referred to the simulated scenarios and presented algorithms that compose the proposed solution is presented in Table 3.

5.3 Evaluating the Proposed Algorithm

This section discusses the simulation results obtained from the characterization of the proposed algorithm including, the rate of packet loss, latency, bandwidth, and the number of covered and uncovered users. The results described in this section correspond to the simulation settings presented in Table 3.

A factorial experiment was considered to show the average, maximum and minimum rate of bandwidth, latency and packet loss, in which one factor (the number of end-users) varies from 15 to 75 and 150 and another factor (the number of DBSs) varies from 3 to 5 and 10, as shown in Table 4. According to Table 4, the average value of bandwidth for the scenarios with 15 users and 3 DBSs, 75 users and 5 DBSs, and 150 users and 10 DBSs, is 96.70, 42.74 and 22.18, respectively. In the scenario with 15 users a high bandwidth is available for users, while by increasing the number of users, the amount of bandwidth decreases, as expected, but the users are still being served. In addition, the average amount of latency is 12.34, 12.28 and 13.46 ms for the scenarios with (15 users, 3 DBSs), (75 users, 5 DBSs and (150 users, 10 DBSs), respectively. As Table 4 represents, the average rate of packet loss for the scenarios with (15 users, 3 DBSs), (75 users, 5 DBSs) and (150 users, 10 DBSs) is 4.29, 2.16 and 8.26 percentage, respectively.

From the results measured by these metrics, the following considerations can be done, as follows:

- *Connected and Disconnected users to the DBSs:*

  The proposed PSO-based algorithm calculated the allocation of 3, 5 and 10 DBSs for 15, 75 and 150 sets of users, according to Fig. 5(plots A, E and I), respectively, as well as their optimal locations. However, to provide a general view and confirm that these numbers of DBSs for each set of users are the optimal ones, Fig. 5 shows all the possible combinations, including (15 users, 3 DBSs), (15 users, 5 DBSs), (15 users, 10 DBSs), (75 users, 3 DBSs), (75 users, 5 DBSs), (75 users, 10 DBSs), (150 users, 3 DBSs), (150 users, 5 DBSs) and (150 users, 10 DBSs). Figure 5plot A shows the optimal location for 3 DBSs calculated by the PSO-based algorithm for a set of 15 users.

  Each DBS is represented with a different color and shape. The users are represented by small dots colored with the same color as the connected DBS. The grey color represents the disconnected users. The same explanation applies to the other figures. According to Fig. 5(plot A), all the users are covered by at least one of the 3 DBSs.

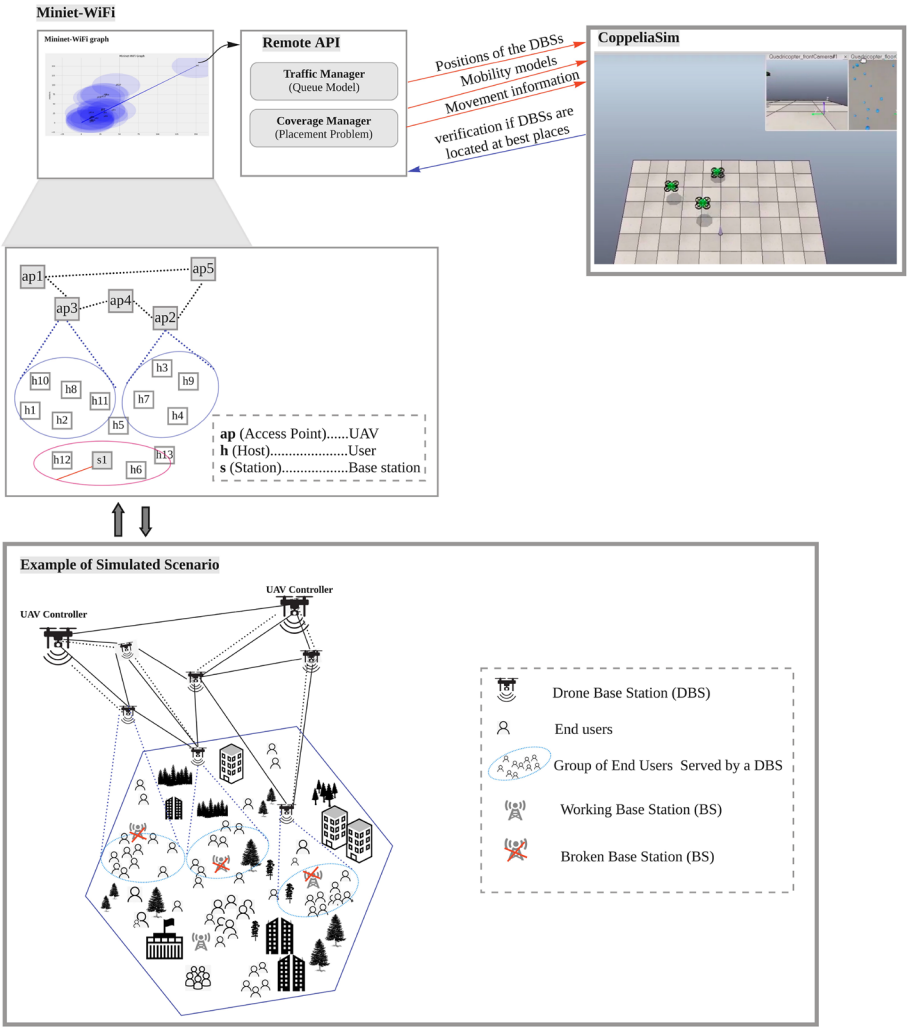

Fig. 4 Simulation tools and possible scenario

DBS0 shown in red covers users *UE3, UE5, UE6, and UE7*. *DBS1* shown in orange covers users *UE8, UE9, UE10, and UE12*. The majority of users including *UE0, UE1, UE2, UE4, UE11, UE13, and UE14* are connected to *DBS2* shown in blue. As can be seen from the Fig. 5(plot A), there is no uncovered users, and all users are covered by one of the DBSs. By increasing the number of users to 75 and 150, the number of disconnected users that are not served by any DBSs, increases. However, the number of disconnected and uncovered users are not remarkable, as shown in Fig. 5(plots E and I), respectively.

- *Packet loss for control and data plane:*

  In wireless communication such as UAV-based communication, decreasing the packet loss rate directly improves the performance of the network. When the DBSs start moving from their initial position, the different users connect to them. It depends

Table 3 Simulation parameters

Simulation Parameters	Value
Scenario	
Time	1 min
Dimension	100×100 m^2
Number of user nodes	15,75,150
User nodes' speed	[5..10]m/s
Maximum transmission power allocated by a DBS (P_{max})	0.5 mW
Maximum bandwidth allocated by a DBS (B_{max})	20 MHz
sub-channel bandwidth (B_{ij})	3 MHz
Transmission power (p_{ij})	30 dBm
Environment parameters a, b, μ_{LOS}, μ_{NLOS}	5.01888, 0.3511, 0.1, 21
Carrier Frequency (f_c)	20 GHz
Total backhaul capacity (*Capacity*)	800 MHz
noise power spectral density (ϵ)	− 147 dBm/Hz
PSO and K-means Parameters	
The initial number of Particles(cluster heads)	eq (2)
The inertia weight (w)	According to [39]
The position acceleration constant (c_1)	According to [39]
The position acceleration constant (c_2)	According to [39]
Termination criterion of topology construction algorithm	300

on the time that the DBSs reach the user's area to start having an actual measurement of the packet loss.

Fig. 6 represents the percentage of packet loss for different scenarios, including (15 users, 3 DBSs), (15 users, 5 DBSs), (15 users, 10 DBSs), (75 users, 3 DBSs), (75 users, 5 DBSs), (75 users, 10 DBSs), (150 users, 3 DBSs), (150 users, 5 DBSs) and (150 users, 10 DBSs). According to Fig. 6(plots A, B and C), that show the percentage of packet loss for the scenario with 15 mobile users, most of the users have an average of 1% packet loss. In the scenario with 15 users and 3 DBSs, as Fig. 6(plot A) illustrates, the 60% packet loss happens for the user *UE10* covered by *DBS1* shown in Fig. 5(plot A). This is because *UE10* is located far from the coverage area of *DBS1* so that the data packets of this user fail to reach *DBS1*. By repeating the test with 75 users, shown in to Fig. 6(plots D, E, and F), although most of the users experience a negligible packet loss rate, some peaks in packet loss occur. For example, in Fig. 6(plot E), users *UE10, UE22, UE41, UE60 and UE66* witness more than 60% packet loss rate in average. This is because the mentioned users are either in a dense area or far away from the DBSs, so that their data packets are dropped during their journey across the network.

As the number of users increases to 150, the network density, mobility of users and DBSs increase during the test, so that some connections are lost and re-established over the test. As Fig. 6(plots G and H) show, a huge amount of packet loss happens for the scenarios with (150 users, 3 DBSs) and (150 users, 5 DBSs). Since, 3 or 5 DBSs are not enough to serve 150 users, the proposed algorithm tries to divide the number of available DBSs, between all the users, which is not going to be able to fully cover all the users at the same time. Therefore, the users with lower priority are not covered and it can be observed a hug amount of packet loss. However, according Fig. 6(plot I),

Table 4 Summary of results, including the average, maximum and minimum rate of bandwidth, latency and packet loss, for the possible combinations of 15, 75 and 150 users with 3, 5 and 10 DBSs

	Users	3 DBSs			5 DBSs			10 DBSs		
		Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
Bandwidth	15	96.70844356	107	1.18	95.7416589	106	1.18	95.61068363	100	1.18
	75	42.16341996	58.21020408	15.2	42.74075711	61.46129032	34.00515625	47.77007301	70.37692308	16.8
	150	30.7051332	64.7	0.535	31.18818279	56.22	0.838	22.18762734	78.8	0.415
Latency	15	12.34953432	165.4781429	0	13.08329538	165.6259286	0	13.45752559	97.64142857	0
	75	11.91903918	171.3848462	0	12.2888868	97.346	0	14.52316102	341.4107143	0
	150	135.2504226	725.32	0	137.9083201	952.54	0	13.46193531	491.5492308	0
Packet Loss	15	4.249998087	60.5833	0	3.61666346	38.6944	0.0555556	0.2333333467	0.527778	0.0555556
	75	10.29111239	100	0.0833333	2.168147987	79.4722	0.305556	1.938887693	10.9722	0.833333
	150	89.190556	100	82.0278	90.452032	100	84.1389	8.265555333	19.5556	6.77778

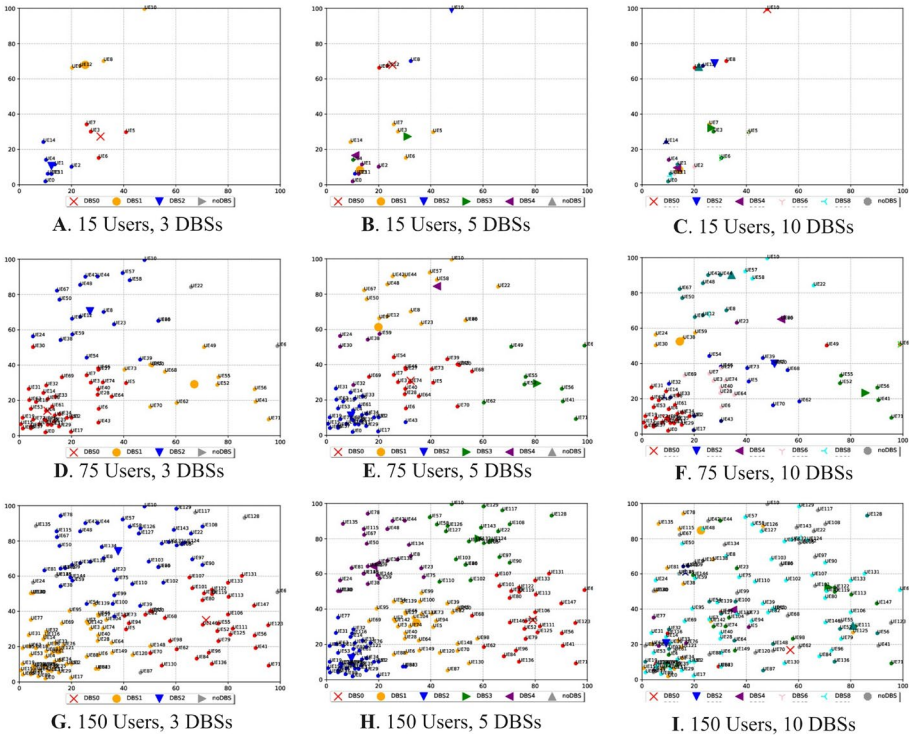

Fig. 5 The optimal location of 3, 5 and 10 DBSs for 15, 75 and 150 sets of users

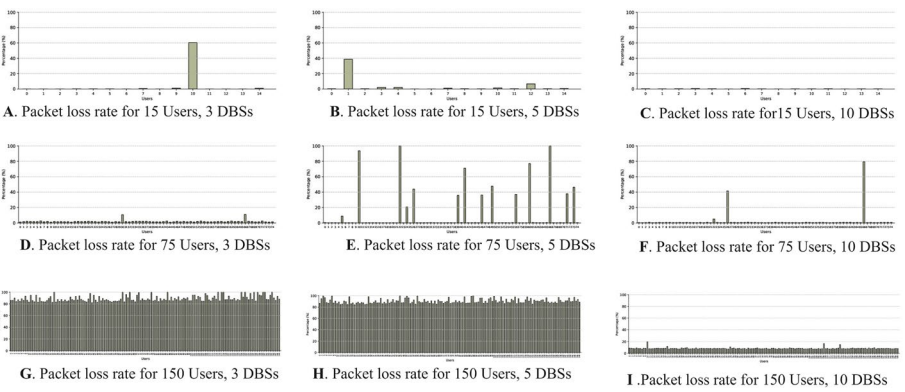

Fig. 6 The packet loss rate of 3, 5 and 10 DBSs for 15, 75 and 150 sets of users

the observed average packet loss rate is around 13% for data sent from users to the 10 DBSs, which is the best number of DBSs calculated by proposed algorithm.

- *Latency:* In FANETs delays accrued mostly because of the inadequate link quality and availability of End-to-end path. Figure 7 indicates the latency for all the possible combinations of 15, 75 and 150 users with 3, 5 and 10 DBSs. As Fig. 7 represents,

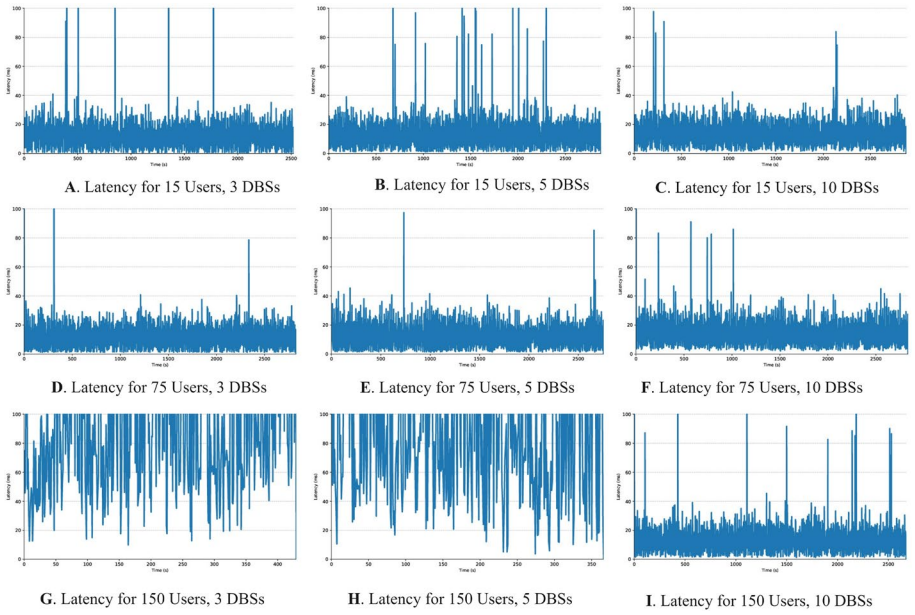

Fig. 7 The amount of Latency of 3, 5 and 10 DBSs for 15, 75 and 150 sets of users

the average latency for each set of users and DBSs is around 20 ms. However, in some stages of the simulation, the latency is extremely high. This occurs because the moving users have been not covered yet by any DBSs. In addition, latency is affected by the distance between users and DBSs.

The more distant users and DBSs are, the more latency is observed. Furthermore, before transmitting a lost packet through the network, the sender requires extra time to identify a dropped packet which leads to an increase in the latency. For example, for the scenario with 150 users, the Fig. 6(plot I), depicts around 10% packet loss rate which leads to 80 to 100 ms delay in some stages of the simulation (Fig. 7(plot I). However, according to Fig. 7(plots A, E and I), after a few seconds, the latency of the proposed approach starts stabilizing (at around 20 ms) and an acceptable latency is achieved because the multiple UAVs system with mesh topology starts covering the users. Figure 7(plots G and H), are different from the others, showing a high amount of latency, because 3 or 5 DBSs are not enough to cover the total number of users, which means many users are not able to connect to at least one DBS and they do not have any network connectivity.

- *Bandwidth* Fig. 8(plots A, B and C) illustrate the amount of bandwidth for different scenarios, including 15, 75 and 150 users and 3, 5 and 10 DBSs, in which the top line shows the max bandwidth (1 Mbps). Figure 8(plots A, B and C) show that the 15 users served by at least 3 DBSs, can take advantage of full bandwidth. However, in some stages the amount of bandwidth is being reduced because the users are disconnected from DBSs or they are associating to new DBSs. By repeating the test with 75 users, as depicted in Fig. 8(plots D, E, and F), the bandwidth is fluctuating mostly between around 40% and 60% of the maximum bandwidth of the network (1 Mbps). By increasing the number of users to 150 and repeating the simulation, the amount of bandwidth decreases, as the number of users increases. According to

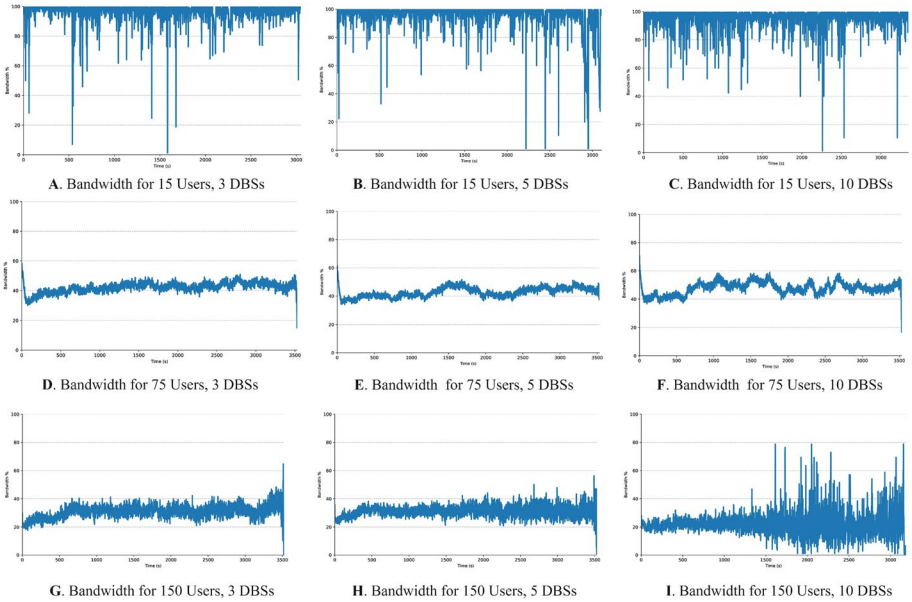

Fig. 8 The amount of bandwidth of 3, 5 and 10 DBSs for 15, 75 and 150 sets of users

Fig. 8(plots G, H and I), the bandwidth is low, between 20% and 40%. However, as Fig. 8(plot I) represents, after 1500s simulation time up to the end of the simulation (one hour), the bandwidth increases, which means that the packets are travelling quicker through the network. All in all, the variation in the bandwidth figures can be explained y the fact that the mobile users are moving at a variable speed in the area and, during the experiment, some of the users might not be covered by any DBSs, which, in turn, impacts the results of bandwidth and leads to the observed variations in the bandwidth figures.

All figures of the presented results are available in high resolution on the Github repository.[2]

6 Conclusion

This paper proposed an improved PSO-based placement algorithm to find the minimum number of DBSs and their optimal locations. An initialization approach that estimates the initial value of the number of DBSs is provided by employing a K-means clustering-based scheme in the PSO algorithm to improve the performance. A custom communication protocol (CCP) was also developed for exchanging the users' main data between users and the network controller. The simulation results show impressive performance of the proposed PSO-based scheme in which very low packet loss and latency. It also indicates that all the users in considered scenario are covered by the DBSs. To extend the work,

[2] https://github.com/ecwolf/SDN-DBS-p1/tree/PSO_continue/fig/select.

in the future we will propose more other existing evolutionary computation algorithms such as Ant colony optimization (ACO) and Genetic Algorithm (GA) combining with the K-means algorithm or other advanced clustering algorithms such as Density-based spatial clustering of applications with noise (DBSCAN) to solve the placement problem.

Funding This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior—Brasil (CAPES)—Finance Code 001 and in part by Conselho Nacional de Desenvolvimento Científico e Tecnológico—Brasil (CNPq) Projects 309505/2020-8 and 420109/2018-8, and partially funded by Grant #2021/00199-8, São Paulo Research Foundation (FAPESP).

Code or data availability Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Declarations

Conflicts of interest The authors declare that there is no conflict of interest.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent to publish Not applicable.

References

1. Pasandideh, F., da Costa, J. P. J., Kunst, R., Islam, N., Hardjawana, W., & de Freitas, E. P. (2022). A review of flying ad hoc networks: Key characteristics, applications, and wireless technologies. *Remote Sensing, 14*(18), 4459.
2. Deniz, F., Bagci, H., Korpeoglu, I., & Yazıcı, A. (2021). Energy-efficient and fault-tolerant drone-bs placement in heterogeneous wireless sensor networks. *Wireless Networks, 27*(1), 825–838. https://doi.org/10.1007/s11276-020-02494-x
3. Fahim, A. & Gadallah, Y. (2020). Optimized 3d drone placement and resource allocation for lte-based m2m communications. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)* (pp. 1–5).
4. Zahedi, M. H., Sobouti, M. J., Mohajerzadeh, A. H., Rezaee, A. A., & Hosseini Seno, S. A. (2020). Fuzzy based efficient drone base stations (DBSS) placement in the 5g cellular network. *Iranian Journal of Fuzzy Systems, 17*(2), 29–38.
5. Akram, T., Awais, M., Naqvi, R., Ahmed, A., & Naeem, M. (2020). Multicriteria UAV base stations placement for disaster management. *IEEE Systems Journal, 14*(3), 3475–3482.
6. Shakoor, S., Kaleem, Z., Do, D., Dobre, O. A., & Jamalipour, A. (2020). Joint optimization of UAV 3D placement and path loss factor for energy efficient maximal coverage. *IEEE Internet of Things Journal*. https://doi.org/10.1109/JIOT.2020.3019065
7. Cherif, N., Jaafar, W., Yanikomeroglu, H. & Yongacoglu, A. (2020). On the optimal 3d placement of a UAV base station for maximal coverage of UAV users.
8. Chaalal, E., Reynaud, L., & Senouci, S. M. (2020). A social spider optimisation algorithm for 3d unmanned aerial base stations placement. In *IFIP Networking Conference (Networking)* (pp. 544–548).
9. Tarekegn, G. B., Juang, R.-T., Lin, H.-P., Munaye, Y. Y., Wang, L.-C., & Bitew, M. A. (2022). Deep-reinforcement-learning-based drone base station deployment for wireless communication services. *IEEE Internet of Things Journal, 9*(21), 21899–21915.
10. Wang, L., Zhang, H., Guo, S., & Yuan, D. (2022). Deployment and association of multiple UAVs in UAV-assisted cellular networks with the knowledge of statistical user position. *IEEE Transactions on Wireless Communications, 21*(8), 6553–6567.
11. Dai, Z., Zhang, Y., Zhang, W., Luo, X., & He, Z. (2022). A multi-agent collaborative environment learning method for UAV deployment and resource allocation. *IEEE Transactions on Signal and Information Processing over Networks, 8*, 120–130.

12. Wu, S., Xu, W., Wang, F., Li, G., & Pan, M. (2022). Distributed federated deep reinforcement learning based trajectory optimization for air-ground cooperative emergency networks. *IEEE Transactions on Vehicular Technology, 71*(8), 9107–9112.
13. Zhang, S., & Ansari, N. (2020). 3d drone base station placement and resource allocation with FSO-based backhaul in hotspots. *IEEE Transactions on Vehicular Technology, 69*(3), 3322–3329.
14. Zhong, X., Huo, Y., Dong, X., & Liang, Z. (2020). Qos-compliant 3-d deployment optimization strategy for UAV base stations. *IEEE Systems Journal, 15*, 1795–1803.
15. Vashisht, S., Jain, S., & Mann, R. S. (2019). Software defined UAV-based location aware deployment scheme for optimal wireless coverage. In *IEEE International Conference on Dependable, Autonomic and Secure Computing, International Conference on Pervasive Intelligence and Computing, International Conference on Cloud and Big Data Computing, International Conference on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)* (pp. 907–912).
16. Cicek, C. T., Gultekin, H., Tavli, B., & Yanikomeroglu, H. (2020). Backhaul-aware optimization of UAV base station location and bandwidth allocation for profit maximization. *IEEE Access, 8*, 154573–154588.
17. Guo, J., Huo, Y., Shi, X., Wu, J., Yu, P., Feng, L. & Li, W. (2019). 3d aerial vehicle base station (UAV-BS) position planning based on deep q-learning for capacity enhancement of users with different QoS requirements. In *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)* (pp. 1508–1512).
18. You, J., Jung, S., Seo, J., & Kang, J. (2020). Energy-efficient 3-d placement of an unmanned aerial vehicle base station with antenna tilting. *IEEE Communications Letters, 24*(6), 1323–1327.
19. Sterbenz, J. P., Hutchison, D., Çetinkaya, E. K., Jabbar, A., Rohrer, J. P., Schöller, M., & Smith, P. (2010). Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines. *Computer Networks, 54*(8), 1245–1265.
20. ur Rahman, S., Kim, G., Cho, Y. & Khan, A. (2017). Deployment of an SDN-based UAV network: Controller placement and tradeoff between control overhead and delay. In *2017 International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 1290–1292).
21. Bozkaya, E., & Canberk, B. (2020). SDN-enabled deployment and path planning of aerial base stations. *Computer Networks*. https://doi.org/10.1016/j.comnet.2020.107125
22. Pan, C., Yi, J., Yin, C., Yu, J., & Li, X. (2019). Joint 3d UAV placement and resource allocation in software-defined cellular networks with wireless backhaul. *IEEE Access, 7*, 104279–104293.
23. Pan, C., Yin, C., Yu, J., & Kiran, N. (2018). 3D UAV placement and resource allocation in software defined cellular networks. In *IEEE/CIC International Conference on Communications in China (ICCC Workshops), 2018* (pp. 136–141).
24. Pan, C., Yin, C., Beaulieu, N. C., & Yu, J. (2019). 3d UAV placement and user association in software-defined cellular networks. *Wireless Networks, 25*(7), 3883–3897. https://doi.org/10.1007/s11276-018-01925-0
25. McCoy, J., & Rawat, D. B. (2019). Software-defined networking for unmanned aerial vehicular networking and security: A survey. *Electronics, 8*(12), 1468. https://doi.org/10.3390/electronics8121468
26. Secinti, G., Darian, P. B., Canberk, B., & Chowdhury, K. R. (2018). SDNs in the sky: Robust end-to-end connectivity for aerial vehicular networks. *IEEE Communications Magazine, 56*(1), 16–21.
27. Liu, W., Niu, G., Cao, Q., Pun, M. O., & Chen, J. (2020). Particle swarm optimization for interference-limited unmanned aerial vehicle-assisted networks. *IEEE Access, 8*, 174 342-174 352.
28. Alfaia, R. D., de Freitas, Souto A. V., Cardoso, E. H. S., Araújo, J. P. L. D., & Francês, C. R. L. (2022). Resource management in 5G networks assisted by UAV base stations: Machine learning for overloaded Macrocell prediction based on users' temporal and spatial flow. *Drones, 6*(6), 145.
29. Dapper e Silva, T., Emygdio de Melo, C. F., Cumino, P., Rosário, D., Cerqueira, E., & Pignaton de Freitas, E. (2019). STFANET: SDN-based topology management for flying ad hoc network. *IEEE Access, 7*, 173 499-173 514.
30. Kim, D., & Lee, J. (2018). Integrated topology management in flying ad hoc networks: Topology construction and adjustment. *IEEE Access, 6*, 61 196-61 211.
31. Pasandideh, F., Dapper-e-Silva, T., Santos da Silva, A. A., & de Freitas, E. P. (2021). Topology management for flying ad hoc networks based on particle swarm optimization and software-defined networking. *Wireless Networks*. https://doi.org/10.1007/s11276-021-02835-4
32. Qiu, C., Wei, Z., Yuan, X., Feng, Z., & Zhang, P. (2020). Multiple UAV-mounted base station placement and user association with joint fronthaul and backhaul optimization. *IEEE Transactions on Communications, 68*(9), 5864–5877.
33. Garg, H. (2016). A hybrid PSO-GA algorithm for constrained optimization problems. *Applied Mathematics and Computation, 274*, 292–305.

34. Sree Ranjini, K. S., & Murugan, S. (2017). Memory based hybrid dragonfly algorithm for numerical optimization problems. *Expert Systems with Applications, 83*, 63–78.
35. Chegini, S. N., Bagheri, A., & Najafi, F. (2018). PSOSCALF: A new hybrid PSO based on sine cosine algorithm and levy flight for solving optimization problems. *Applied Soft Computing, 73*, 697–726.
36. Dong, N., Wu, C.-H., Ip, W.-H., Chen, Z.-Q., Chan, C.-Y., & Yung, K.-L. (2012). An opposition-based chaotic GA/PSO hybrid algorithm and its application in circle detection. *Computers & Mathematics with Applications, 64*(6), 1886–1902.
37. Xiang, T., Liao, X., & Wong, K. (2007). An improved particle swarm optimization algorithm combined with piecewise linear chaotic map. *Applied Mathematics and Computation, 190*(2), 1637–1645.
38. Krishna, K., & Narasimha Murty, M. (1999). Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 29*(3), 433–439.
39. Paul, S., De, S. & Dey, S. (2020). A novel approach of data clustering using an improved particle swarm optimization based k-means clustering algorithm. In *2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)* (pp. 1–6).

Faezeh Pasandideh received her B.S. in computer engineering from Quchan University Of Advanced Technologies Engineering, Iran, in 2012, and her M.S. in computer engineering from Islamic Azad University (IAU), Science and Research Branch, Tehran, in 2015. She is currently pursuing a Ph.D. degree in Graduate Program in Computer Science (PPGC)—Federal University Rio Grande Do Sul (UFRGS), BRAZIL and Hamm-Lippstadt University of Applied Sciences, Germany. She is also a doctoral member in the "Technology and Systems" department, PK NRW, Germany. Her research interests are in UAV networks, software-defined networks, Routing, and Resource Management In Wireless Sensor Networks, Quality Of Service (QOS), Underwater Wireless Sensor Network, Traffic and Congestion Control, IoT, 5 G and 6 G networks.

Fabricio E. Rodriguez Cesen is a Ph.D. student at the University of Campinas (UNICAMP), Brazil, and a Researcher of Information and Networking Technologies Research and Innovation Group (IN-TRIG). He holds the Networks and Data Communication Electronic Engineering degree from the Ar-my University—ESPE of Ecuador, and the M.Sc. degree in Electrical Engineering in the area of Computer Engineering from the University of Campinas (UNICAMP), Brazil, 2018. During his master thesis at UNICAMP, he worked on the Design, Implementation, and Evaluation of IPv4/IPv6 Longest Prefix Match support in Multi-Architecture Programmable Dataplanes. He collaborated with Ericsson and RNP in different projects, and he is the INTRIG's network administrator.

Pedro Henrique Morgan Pereira is currently pursuing his master's degree in the Graduate Program in Electrical Engineering at the Federal University of Rio Grande do Sul (UFRGS), Brazil. He is a bachelor's in Electrical Engineering from the Federal University of Rio Grande do Sul (UFRGS), 2020. His main research interests include the following topics: telecommunication protocols; Internet of Things; IIoT; Unmanned Aerial Vehicles; Image processing and artificial intelligence.

Christian Esteve Rothenberg is an Associate Professor at University of Campinas (UNICAMP), where he received his Ph.D. in Electrical and Computer Engineering in 2010. From 2010 to 2013, he worked as Senior Research Scientist in the areas of IP systems and networking at Fundação Centro de Pesquisa e Desenvolvimento (CPqD), Campinas, Brazil. Christian has been an OpenFlow® enthusiast since late 2008, when he worked with protocol version 0.8.9 to proof the concept of data center forwarding based on probabilistic data structures as part of his PhD thesis. These days, he devotes his efforts to bridge the gap between academia and industry, contributing to operational deployments of OpenFlow/RouteFlow and value-added SDN use cases in the Brazilian network technologies ecosystem. He received his Ph.D. in Electrical and Computer Engineering from the University of Campinas (UNICAMP), Brazil. He also holds the Telecommunication Engineering degree from the Technical University of Madrid, Spain, and the MSc degree in Electrical Engineering and Information Technology from the Darmstadt University of Technology (TUD), Germany. Christian has two US patents and over 200 scientific publications, including book chapters, journals, and top-tier networking conferences such as SIGCOMM and INFOCOM

Edison Pignaton de Freitas Computer Engineer by the Military Institute of Engineering in Brazil (2003), he received the M.Sc. in Computer Science by Federal University of Rio Grande do Sul (UFRGS) in Brazil (2007), and the Ph.D. degree in Computer Science and Engineering from Halmstad University in Sweden in 2011. Currently, he holds an associate professor position at UFRGS, being also Secretary for International Relations, developing research in the Graduate Program on Computer Science and in the Graduate Program in Electrical Engineering. His research interests are mainly in the following areas: Computer Networks, Internet of Things, Real-Time Systems, Multiagents Systems and Unmanned Aerial Vehicles.

Authors and Affiliations

Faezeh Pasandideh[1,2,3] · Fabricio E. Rodriguez Cesen[5] · Pedro Henrique Morgan Pereira[4] · Christian Esteve Rothenberg[5] · Edison Pignaton de Freitas[1]

Fabricio E. Rodriguez Cesen
frodri@dca.fee.unicamp.br

Pedro Henrique Morgan Pereira
morgan.pereira@ufrgs.br

Christian Esteve Rothenberg
chesteve@dca.fee.unicamp.br

Edison Pignaton de Freitas
edison.pignaton@inf.ufrgs.br

1 Department of Computer Science, Federal University Rio Grande do Sul (UFRGS), Porto Alegre, Brazil

2 Department 2 Lippstadt, Hamm-Lippstadt, University of Applied Sciences, Hamm, Germany

3 Department of "Technology and Systems", Pk Nrw, Germany

4 Department of Electrical Engineering, Federal University Rio Grande do Sul (UFRGS), Porto Alegre, Brazil

5 School of Electrical and Computer Engineering (FEEC), University of UNICAMP, Campinas, Brazil