Distributed Connectivity Maintenance in Swarm of Drones During Post-Disaster Transportation Applications

Ahmet Kurt, Nico Saputro, Kemal Akkaya, *Senior Member, IEEE*, and A. Selcuk Uluagac

***Abstract*— Considering post-disaster scenarios for intelligent traffic management and damage assessment where communication infrastructure may not be available, we advocate a swarm-of-drones mesh communication architecture that can sustain in-network connectivity among drones. The connectivity sustenance requirement stems from the fact that drones may move to various locations in response to service requests but they still need to cooperate for data collection and transmissions. To address this need, we propose a fully distributed connectivity maintenance heuristic which enables the swarm to quickly adapt its formation in response to the service requests. To select the moving drone(s) that would bring minimal overhead in terms of time and moving distance, the connected dominating set (CDS) concept from graph theory is utilized. Specifically, a variation of CDS, namely E-CDS, is introduced to address the needs of 3-D mobile swarm-of-drones. We then show that E-CDS is NP-Complete and propose a new distributed heuristic to solve it. Once the E-CDS is determined in advance, drones not part of this E-CDS set are picked for movement tasks. When the movement is to cause any disconnection with the rest of the swarm, other drones are also relocated to restore the connectivity. The proposed heuristics are implemented in ns-3 network simulator as part of the existing IEEE 802.11s mesh standard and the effectiveness is tested in terms of providing undisturbed services under different conditions. The results indicate that the proposed distributed heuristic almost matches the performance of a centralized solution and suits perfectly the needs of post-disaster traffic management.**

***Index Terms*— Drones, wireless mesh network, connectivity maintenance, post-disaster transportation management, IEEE 802.11s, connected dominating set, NP-completeness.**

I. Introduction

CONTINUOUS surveillance of traffic is crucial in metropolitan areas to provide updates and situational awareness. This becomes a challenge in the aftermath of disasters, which cause damages to power systems and road infrastructure [1]. This is very relevant in disaster prone areas such as South Florida where traffic congestion is already a major problem [2]. Traffic suffers even further when traffic lights are out of power, some road are inaccessible and cellular communication is intermittent.

In such cases, the transportation authority needs to perform the damage assessment and notification actions by utilizing the available Intelligent Transportation System (ITS) applications such as the dynamic message boards located over the roads or Roadside Units (RSUs) which can communicate with vehicles [3]. However, collection of information will be limited due to lack of power and communications.

Therefore, recently, Unmanned Aerial Vehicles (UAVs) or drones, have been envisioned as one of the tools that may provide the aforementioned services due to their ability for easy deployment, hovering capability, and bird's view from the sky features [4]–[6]. In particular, a swarm-of-drones as used in many other civil applications such as for archaeological site mapping [7], and natural disaster management [5] can also be applied to ITS settings.

In such a setting, the drones may be requested to move to certain spots temporarily to capture videos or act as relays for any incoming traffic from the ground. However, at the same time, these drones need to be maintaining their group ties so that they can enable communication not just among themselves but also with any remote control center(s). This communication will also be needed to manage drone re-charging activities and find their locations in case GPS is not accessible. Therefore, there needs to be continuous connectivity (i.e., they can reach each other anytime) among the drones to sustain these services in a seamless manner using meshing capabilities.

In this paper, we first layout an architectural design for enabling a swarm-of-drones network using wireless mesh capability which will assist various parties (e.g., transportation authority personnel, emergency medical teams, fire rescue, police, etc.) in a post-disaster transportation scenario. We rely on a mix of wireless mesh routing and cellular/wide area communications through which these drones could be managed. We adopt the swarm communication architecture in [8], which utilizes three communications technologies: (1) IEEE 802.11p [9] is proposed for the communications between vehicles and drones; (2) IEEE 802.11s [10] wireless mesh network is used to support the internal swarm communications; and (3) Wide-area communication (i.e., cellular or

Manuscript received March 15, 2020; revised November 15, 2020 and January 31, 2021; accepted March 11, 2021. Date of publication March 24, 2021; date of current version September 1, 2021. This work was supported in part by the National Priorities Research Program (NPRP) through the Qatar National Research Fund under Grant #NPRP9-257-1-056 and in part by the Ministry of Research and Technology/National Research and Innovation Agency (Kemenristek/BRIN) of the Republic of Indonesia under the World Class Research Scheme 2020. This article was presented in part at the Proceeding of IEEE 43rd Conference on Local Computer Networks Workshops (LCN Workshops), 2018. The Associate Editor for this article was Y. Wan. *(Corresponding author: Ahmet Kurt.)*

Ahmet Kurt, Kemal Akkaya, and A. Selcuk Uluagac are with the Department of Electrical and Computer Engineering, Florida International University, Miami, FL 33174 USA (e-mail: akurt005@fiu.edu; kakkaya@fiu.edu; suluagac@fiu.edu).

Nico Saputro is with the Department of Electrical Engineering, Parahyangan Catholic University, Bandung 40141, Indonesia (e-mail: nico@unpar.ac.id).

Digital Object Identifier 10.1109/TITS.2021.3066843

LoRaWAN [11]) is used for the long-range communications between a gateway drone (i.e., root in the mesh network) in the swarm to a control center.

Once the communication architecture is fixed, we propose a heuristic that will enable seamless connectivity among the drones for service sustainability. Specifically, when there is a need to move certain drones for different tasks, our heuristic will select the most suitable drone that will minimally impact the connectivity among the existing drones. In case a disconnection would occur, then our proposed heuristic will enable connectivity restoration without disrupting existing data traffic with minimized drone movement costs.

Specifically, our heuristic is based on Connected Dominating Set (CDS) concept [12] which ensures that a core set of nodes are always reachable through the lifetime of the swarm. In our case, since the swarm will act semi-autonomously, there is a need for a distributed heuristic which will not bring major overhead due to risk of disrupting existing services. In addition, certain nodes will not be moveable (i.e., gateway drone) and should be part of the CDS, which further restricts the solution space. Therefore, we introduce a variation of CDS problem to incorporate these needs which we call E-CDS. We first prove that finding the E-CDS of a network is an NP-Complete problem just like the well-known CDS problem and then we adopt a distributed heuristic to solve this problem. Our solution enables localized E-CDS computation in a fast and efficient manner by embedding it at the MAC layer of the protocol stack.

Once the E-CDS is determined, in case of service needs for drone movements, we propose a distributed protocol that will identify the closest drone(s) to undertake the mission which are not part of the E-CDS. When the identified drone(s) move to new locations, this approach also re-stores connectivity in case any disconnections occur. Basically, either the nodes stretch or additional nodes are relocated to fill the gaps.

We implemented the proposed swarm communications architecture and E-CDS based connectivity maintenance protocol under ns-3 network simulator [13] by utilizing the IEEE 802.11s mesh standard [10]. To the best of our knowledge, the E-CDS computation is the first to be added to IEEE 802.11s WiFi mesh standard. We relied on a new MAC-layer beacon and implemented the heuristic at the MAC layer to minimize the communication overhead. The simulation results in 3-D topologies of drones demonstrated the feasibility of the proposed heuristic and its ability to function with negligible overhead, almost matching the performance of a centralized approach.

The paper is organized as follows. In Section II, we discuss the relevant work while Section III describes the drone communications architectures along with some preliminaries. The proposed heuristic is explained in Section IV. Detailed performance evaluation of the proposed model is given in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

A. Swarm of Drones in Its

While single drone applications have been prevalent, swarm of drones has just started to be considered for many ITS and disaster management applications. For instance, Menouar *et al.* [6] discusses how UAVs can be used for ITSs in smart cities. They investigate numerous use cases which include but not limited to utilizing UAVs as flying RSUs to capture video recordings of an incident scene, as flying accident report agent or as flying police eye etc. They offer a vision rather than getting into the details of specific problems. In [8], the authors tackle a specific problem when drones act as a mobile RSU in a secure hybrid communication infrastructure. Specifically, they propose a novel tunneling protocol to integrate LTE with IEEE 802.11s. In [14], the authors look at a slightly different problem which is about placement of drone docking stations for ITS. They strive to determine the best locations to install the docking stations in a large geographical area. According to their findings, drones have to reach the incident location in a reasonable time and they should not have a risk of battery failure during a mission. Chen *et al.* [15], proposed a long-range and broadband aerial communication system for UAVs to extend communication range, increase throughput and reduce interference. A slightly different work [16] which is a combination of IoT and ITS settings focuses on an (UAV)-enabled wireless powered mobile edge computing (MEC) system. Authors propose an optimization problem to minimize some metrics such as: total energy required for UAV operations (through CPU frequency optimization), the offloading amount, the transmitted power and the trajectories of the UAVs. To solve the optimization problem authors propose two algorithms namely a successive convex approximation based algorithm and a decomposition and iteration (DAI) based algorithm. And their results indicate that both algorithms converge quickly while DAI algorithm achieves a more optimized trajectory. Also very recently, a deep learning based drone detection algorithm in ITS settings were proposed in [17].

Our work differs from the above studies, as none of them tackles the problem of connectivity maintenance within a swarm of drones, particularly within ITS settings.

B. Connected Dominating Set Heuristics

Finding the CDS is an NP-complete problem [18]. Hence, there has been a lot of proposed algorithms in the literature based on approximation. Many of these algorithms are motivated by the greedy heuristics proposed for the general graphs by Guha and Khuller [12]. However, given the nature of our application, centralized solutions may not be practical all times. There are also several distributed heuristics to address CDS problem. For instance, the authors in [19] presented an energy efficient distributed CDS algorithm based on coordinated reconstruction mechanism. Their algorithm's time complexity is $O(n)$. Wan *et al.* also proposed a distributed CDS construction algorithm with a same time complexity of $O(n)$ [20]. Their algorithm consists of two phases. In the first phase, a maximal independent set is constructed and in the second phase a dominating tree is constructed. They compare their algorithm with existing CDS algorithms and show that their proposed algorithm is more efficient in terms of message complexity. Dubhashi *et al.* in [21] presented a fast distributed CDS algorithm for a given distributed network.

Their algorithm finds the CDS in $O(\log n)$ time. The most efficient algorithm in terms of message complexity was presented in [22]. In this approach, the authors follow a two-phase algorithm where they first mark the potential nodes in the first phase and prune some of them in the second. The algorithm only relies on two broadcasts messages from each node and thus has a message complexity of $O(1)$.

While our problem is a more specific version of the CDS problem where there are mandatory nodes within the CDS (i.e., E-CDS), our distributed heuristic to solve it was motivated from the approach in [22] due to its message efficiency. This is crucial in a distributed protocol for minimizing the problems in wireless 3-D environments where drones will be deployed. We revised this approach in [22] to come up with a distributed solution.

We would like to also note that although various distributed approaches were offered for CDS in the literature, none of them was implemented in a realistic simulator or environment as a complete protocol. Our work's unique contribution includes a complete implementation as part of the IEEE 802.11s mesh standard.

C. CDS and Connectivity Restoration in Mesh Networks

Utilizing CDS for numerous purposes and connectivity restoration are two research areas that have been intensively studied in the context of specific types of mesh networks such as mobile ad hoc networks (MANETs) and wireless sensor networks (WSN), for years. Particularly, CDS has been used to form an underlying virtual network architecture that can be used for various applications such as media access coordination; unicast, multicast/broadcast, and location-based routing; energy conservation, and topology control [23]. Recently, a CDS-based topology control algorithm was proposed by Qi *et al.* [24] for flying ad hoc networks (FANETs). In this work, authors approach the topology control problem by transforming it into three sub-problems and propose an algorithm for each sub-problem. First algorithm minimizes global transmission power, second algorithm constructs an CDS for virtual backbone creation and the last algorithm adjusts node positions. Their evaluation results show better performance on nodes' network overhead and power usage compared to other typical topology control methods.

Our goal in this work is different as we focus on connectivity maintenance. Regarding connectivity maintenance and restorations, approaches appeared within WSNs to address the node failure problem, which may cause disjoint segments in the network and hinder the WSN operations. The proposed approaches in WSNs can be classified as the proactive and reactive approaches [25]. Proactive ones typically strives to provision resources at the setup by introducing redundant nodes as spares while reactive solutions strive to provide the real-time restoration of the lost connectivity. The reactive approaches can be divided into three categories [25] by utilizing: (1) nodes that are part of the network by re-positioning them (e.g., [26]); (2) additional stationary relay nodes; and (3) additional mobile relay nodes. Our proposed heuristic connectivity restoration can be viewed as a version of the first category. It basically strives to utilize the existing nodes that are part of the swarm by re-positioning them, either by re-positioning one or more non backbone nodes as relay node(s) in between the node in the new location and the closest backbone node to this node; or by stretching some of the backbone nodes' locations toward the new location node. Different from these approaches, however, we have additional challenges such as the 3-D environment, constant mobility, and restricted roles of some drones. In addition, none of the existing WSN connectivity restoration approaches were implemented within a realistic environment to deal with routing and packet losses due to mobility.

We would like to note that this work is an extension to our previous study [27] and the additions are as follows: 1) The conference version only proposed a centralized connectivity maintenance heuristic. In this paper, we propose a completely new distributed connectivity maintenance heuristic. This new distributed approach is compared with the prior centralized approach as well as two other distributed approaches; 2) Instead of CDS, a completely new enforced-CDS (E-CDS) problem is introduced. In E-CDS, certain nodes are enforced to be in CDS. We prove that E-CDS problem is NP-complete; 3) A distributed moving node selection algorithm is presented for relocating nodes upon request from control center. In the conference version, no pseudocode was given for this algorithm and there was no actual implementation on ns-3. In this paper, we provide a complete implementation in ns-3 by introducing two new network packets which are called *beacon-like* packet and *claim* packet; and 4) The evaluation section has been extended significantly. We performed 6 different experiments while in the prior work, only end-to-end delay results were presented. In a separate work, we have detailed how this approach was implemented in ns-3 as an 802.11s MAC layer extension [28].

III. PRELIMINARIES

In this section, we first describe the overview of the system model. Then, we present the problem formulation.

A. System Model

The proposed swarm of drones is used to assist in transportation services as well as the first responder/police personnel for damage assessment in a post-disaster scenario. We assume an urban area of interest where the LTE public cellular network coverage may not be available in every spot (e.g., the LTE base station is damaged). Drones are deployed in this area of interest to form an Aerial Wireless Mesh Network (AWMN) to cover this blank spot area. Each member of the AWMN is called a *flying mesh*. We assume that such mesh communications are done through the existing IEEE 802.11s mesh standard [10]. Basically, based on the transmission range of the drones, we assume a unit disc graph $G(V, E)$ to form the mesh network where V represents the set of drones and E represents the wireless links among these drones. For the rest of the paper, we will use AWMN and swarm of drones interchangeably.

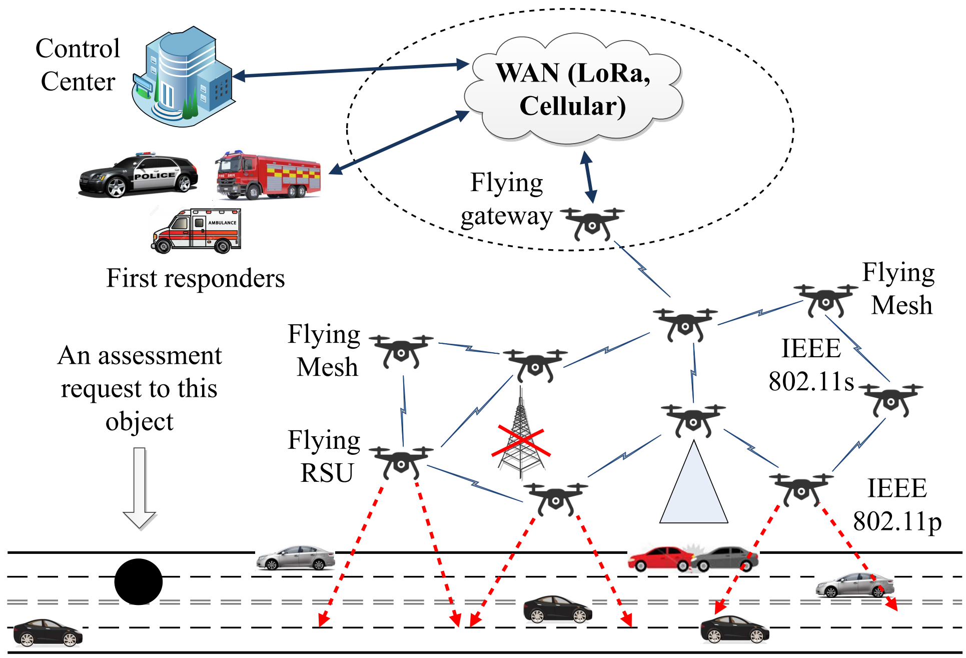


Fig. 1. The proposed network architecture for swarm of drones in a post-disaster transportation management scenario.

The reason we picked 802.11s mesh is twofold: First, it is a standard which is already used in practice (e.g., with Google routers [29]); and 2) its routing protocol, namely, Hybrid Wireless Mesh Protocol (HWMP) [30] performs the best for 3-D AWMN. Specifically, A. Nayyar [31] performed a comparison of MANET routing protocols (AODV, DSDV, DSR, AOMDV, OLSR and HWMP) on UAV swarms. The results for packet delivery ratio and throughput indicate that HWMP outperforms the other routing protocols. For end-to-end delay, the best performance is provided by both HWMP and DSDV for the majority of the data points. Therefore, HWMP is best suited for 3-D UAV swarms.

The drone deployment must be done such that at least one of the flying meshes has a wide area communication capability (such as LoRa standard) to act as a gateway for remote connections to perform command and control. In this case, this flying mesh uses two communications protocols (i.e., IEEE 802.11s and LoRa) and becomes the *flying gateway* for the AWMN. The default path selection mechanism in IEEE 802.11s, HWMP, has two path selection modes: (1) on-demand that enables peer-to-peer communications between drones; and (2) proactive tree that requires the presence of a root node in order to build a tree that connects all drones in the mesh to the root node. In this case, the flying gateway also acts as the *root node* in the AWMN.

Two other types of drones that use more than one communication protocols (i.e., IEEE 802.11s and IEEE 802.11p) are the *flying road-side unit (RSU)* and the *flying investigator*. While flying RSU acts as a temporary RSU for drone-to-vehicle communications, flying investigator can sense the area or inspect infrastructure using one or more on board sensors (e.g., visual sensors, ultrasound sensors, high definition video, etc).

We envision that each drone is battery operated and can sustain their operations for temporary needs. However, if there is a longer duration need, then the drones can leave and get re-charged without impacting the services. We also assume that drones have on-board GPS. Note that the GPS coordinate information in terms of Geographic latitude and longitude will be converted to the Cartesian coordinates (x,y,z). Additionally, each on-board sensor and communication technology in the drone can be activated/deactivated on demand. The overall system architecture is provided in Fig. 1.

B. Swarm Control Assumptions

In our proposed system architecture, the *flying gateway* is the interface for the swarm-of-drones (i.e., AWMN) to ITS and any ground emergency personnel/policemen. It receives certain task requests (i.e., video reports, RSU function, etc.) from the control center or emergency personnel. These requests are fulfilled among the swarm accordingly. During such task execution, the swarm self-determines which drone(s) should be moved, and then these drone(s) move to the desired position(s) to undertake the needed task at the new location. Fig. 2 presents our assumed control protocol, showing the message exchanges between the *flying gateway* and the rest of the drones in the swarm. In the disaster assessment request, the requestor (i.e., control center/first responders) must specify the expected role (e.g., *flying investigator* or *flying RSU*) and the target location.

C. Problem Definition

During task execution, it is important to maintain connectivity among all drones within an AWMN so that each node not only transmits their data without interruption but also receives any updates. Given the mobility of drones, the connectivity

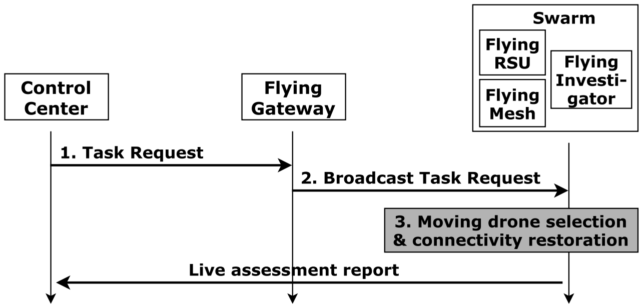


Fig. 2. Proposed swarm control protocol.

maintenance need to be coordinated among the drones. This requires a connectivity maintenance protocol.

More formally, we define the problem as follows: We assume a swarm of N autonomous drones that form an AWMN G. Each drone in G is assigned to one of the following roles R = *flying gateway, flying investigator, flying RSU, flying mesh*. Let $g(t)$, $i(t)$, $r(t)$, and $m(t)$ represent the set of drones in the swarm at any given time t that has the flying gateway, flying investigator, flying RSU, and flying mesh roles respectively, then

$$|g(t)| + |i(t)| + |r(t)| + |m(t)| = |N| \tag{1}$$

and

$$|g(t)| = 1, \quad \forall t \tag{2}$$

since only a single drone can have the non-transferable flying gateway role all the time. On receiving k tasks at time $(t + 1)$ where $k \ll (N_r(t) + N_m(t))$, then the problem can be formulated as follows:

$$\min \quad (C_K + C_S) \tag{3}$$

$$\text{subject to} \quad Connected(G) \tag{4}$$

$$(K \cup S) \cap (g(t) \cup i(t)) = \emptyset \tag{5}$$

where K represents the set of drones picked for movement, S represents the set of drones that moved for connectivity restoration rather than service needs, C is the travel distance cost for a set of drones and $Connected()$ is a function representing whether a graph is connected or not.

IV. Connectivity-Aware Drone Movement

A. Overview

To tackle the problem of connectivity restoration with minimum movement, we first need to determine the right node(s) to move. Then, we consider an efficient movement algorithm to move those nodes. For the first challenge, we propose a CDS-based heuristic while second approach considers a cascading approach to fill in the locations as the nodes are moved.

For CDS heuristic, the motivation is to maintain a core for the AWMN that keeps everyone reachable to every other node in the network. Specifically, when the CDS is determined, each node will know whether it is a dominator (i.e., an element of the E-CDS) or a dominatee (i.e., can reach to a dominator within one-hop) as shown in Figure 3. Any movement in the network will be from these dominatee nodes preferably. An example CDS for a graph is shown in Fig. 3. Selection of the nodes among dominatees and their movement will be done in the restoration heuristic.

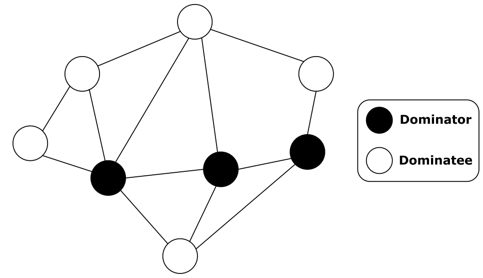


Fig. 3. An example CDS for a graph; black nodes are dominators, white nodes are dominatees.

B. A More Restricted CDS Need: E-CDS

In this section, we provide the details of the connectivity maintenance approach through tackling a special version of the CDS problem.

As mentioned, CDS maintains a set of core drones in the network that are reachable by any other node. Ideally, our goal is not to use these core nodes for any movement related tasks. For such tasks, we prefer to pick among nodes that are not part of the CDS. While any node can be picked to be in the core, in our case there is a special consideration. Specifically, some of the nodes should not be moved. For instance, the gateway drone should not be moving as it acts as the data collector for all other drones. Similarly, we can define a specific flying RSU node to be fixed so that it can serve the vehicles within its vicinity.

This creates a more specific CDS. The problem than evolves to a more restricted case. Basically, the selected CDS should guarantee the existence of certain nodes within it. In regular CDS, any node is qualified for this set and thus there is no guarantee that certain nodes would be in it eventually. This creates a new problem, which we name as *CDS with certain Enforced Nodes (E-CDS)*.

More formally, *E-CDS* is defined as follows: Let $G = (V, E)$ and V' be a set of vertices where $V' \subseteq V$. *E-CDS* is defined as a subset G' of V which must meet the following conditions: 1) Every vertex not in G' is adjacent to at least one member of G'; 2) $V' \subseteq G'$ and all the vertices in G' are connected to each other through edges in E. The E-CDS decision problem is the following:

INSTANCE: Given a graph G and an integer k.

QUESTION: Does G have an E-CDS of size at most k?

Theorem: Finding E-CDS is NP-Complete.

Proof: We provide a proof by making a reduction from the CDS problem which is already NP-Complete [18]. Let us assume that G' is a CDS for graph $G = (V, E)$ where $|G'| = k'$. Let us consider an enforced set V' which is a subset of V as shown in Fig. 4. By definition of CDS, any member of V' should have an edge to any of the members of G'. If we merge G' with V' and the edges connecting them to G', then we will obtain a new subgraph, say, H with size

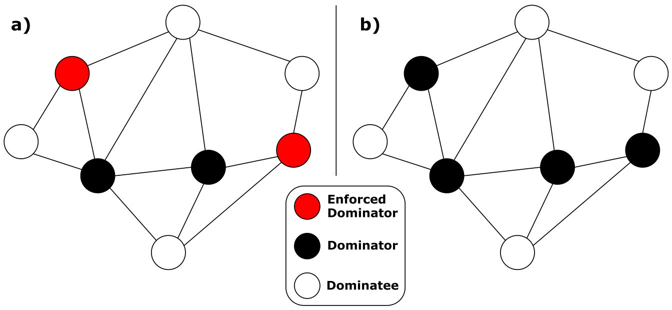


Fig. 4. a) A graph G where red nodes are members of V' (i.e., with higher priorities) and black nodes are members of G' (i.e., dominator nodes). b) New CDS is shown (i.e., $H = G' \cup V'$) where members of V' are included in this new CDS.

$|H| = k$ which will still be connected and a CDS for G. This way we were able to obtain another version of CDS problem which is NP-Complete. Therefore, E-CDS should also be NP-Complete.

To the best of our knowledge, this is a new CDS problem that has not been defined and solved before in the literature. Therefore, as will be shown in the next subsection, we modified the Algorithm 1 and 2 of [22] to implement the new E-CDS.

C. Distributed CDS-Enforced Heuristic and Protocol

Since the *E-CDS* problem is NP-Complete, the solution to this problem will be either centralized or distributed approximation. There are many centralized heuristics for the original CDS [23] but these are not suitable for large mesh networks that are dynamic. In our case since AWMN will be mostly autonomous, centralized control might not be suitable at all times. Therefore, a distributed maintenance protocol is considered. The localized or distributed heuristics for the original CDS strive to minimize the overhead as it will require coordination and communication among the nodes. Therefore, in this work, we picked one of the most efficient of these CDS algorithms (i.e., [22] due to its $O(1)$ message complexity) and adapted it for *E-CDS* in AWMN.

1) E-CDS Heuristic: In our approach, E-CDS is calculated distributively meaning there is no central drone that knows every other drone's location and calculates the E-CDS using that information. Rather, E-CDS is calculated locally by each drone to determine whether it is part of the E-CDS or not. The proposed heuristic consists of two algorithms: First algorithm is called *marking process* which marks some of the drones in the network. If a drone is marked, this means it is potentially part of the E-CDS. In the same way, if a drone is not marked, that means it is not part of the E-CDS. In the algorithm, we force *flying investigators* and *flying gateway* to be in the E-CDS by marking them regardless of the result of the marking process since they should not be moving. Marking process is presented in Algorithm 1. The algorithm starts by checking (line 2-8) whether there are two neighbors of the drone that are not neighbors of each other (i.e., not in the transmission range of each other). If this is the case, the drone is marked (line 5). To do this, the algorithm uses the neighboring information collected by the drone. Lastly, the algorithm checks (line 9) whether the drone is a flying investigator or a flying gateway. If the drone is either of them, it is marked as *True* (line 10). This is the enforcing step of the algorithm that forces the drone to be in the E-CDS based on its role. This step is also a modification to the original marking process algorithm that is initially proposed by Dai *et al.* in [22]. Marking process is an asynchronous (each drone runs the algorithm simultaneously) event-based (neighboring information is dynamic) algorithm.

Algorithm 1: Marking Process

```
Input  : Selected drone and its neighbors
/* neighbors is collected by beaconLike
   packets in a function called
   receiveBeaconLike()                          */
Output: Processed marker
1  initialize marker = False;
2  for neighbor_A in neighbors do
3    for neighbor_B in neighbors do
4      if edgeExists(neighbor_A, neighbor_B) == False
        then
5        marker = True;
6      end
7    end
8  end
9  if drone.role == flying gateway or
    drone.role == flying investigator then
10   marker = True;
11 end
12 return marker;
```

Second algorithm is called *restricted k-dominant pruning* and only the drones that were marked *True* by the marking process run this algorithm. We provide the pseudo-code in the Algorithm 2. The purpose of running this algorithm is to prune the E-CDS further by removing the redundant drones by unmarking them (i.e. by converting their markers to *False* again). After this step, the number of marked drones decreases making the E-CDS closer to the optimal solution. Similar to the marking process, this algorithm uses the neighboring information collected by the drone and has 4 main steps. It starts by checking (line 2) whether the drone is a flying investigator or flying gateway. If the drone has either of the roles, it does not run the algorithm and stays in the E-CDS. This step is also a modification to the original restricted k-pruning algorithm proposed in [22]. In the second step, the algorithm creates a subgraph (line 3) consisting of the marked neighbors of the drone with higher IDs. An ID is assigned to each drone in the network which serves as a priority and prevents simultaneous unmarking of two drones that are neighbors of each other. Drones can have equal IDs but this will result in having less number of pruned drones from the E-CDS. In the third step, strongly connected components of the previously computed subgraph is calculated (line 4). Lastly, a set subtraction is done which checks if the marked higher ID neighboring drones are eligible for pruning or not (line 5-6). If the condition is satisfied, the drone is marked *False* and removed from the E-CDS (line 7). An example of how pruning is performed on a directed graph is shown in Fig. 5.

Algorithm 2: Restricted k-Dominant Pruning

```
Input  : Selected drone and its neighbors
Output: Processed marker
initialize marker = True;
if drone.role ≠ flying gateway and
  drone.role ≠ flying investigator then
    G_V+ = buildSubgraph();
    SCC = computeSCC(G_V+);
    /* buildSubgraph and computeSCC
       subroutines are not shown for
       simplicity                                  */
    for strongly connected component V_i in SCC do
        if neighbors − V_i ⊆ neighbor set of V_i then
            marker = False;
        end
    end
end
return marker;
```

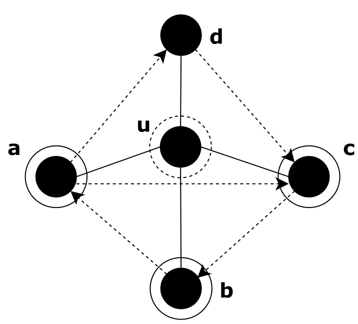


Fig. 5. The nodes with a circle around them, i.e., a, b, c, are marked already and part of the E-CDS. Arrows show the direction of the links between the nodes. Host u will unmark itself via the restricted k-dominant pruning since it has 2 marked neighbors covering it. Note that this is not the case for node b because there is no incoming link from node a to node b.

2) Distributed E-CDS Protocol Design: Both of the above algorithms require drones to know about their neighbors and their neighbors' neighbors (2-hop neighbors). This means there is a need for a protocol to collect this information by only talking to 1-hop neighbors of a node (i.e., everyone needs to share their 1-hop neighbor list with its own neighbors). As mentioned, IEEE 802.11s (HWMP) is the best performing mesh standard for 3-D mobile environments. Therefore, we opt to incorporate the protocol within this mesh standard.

In our distributed E-CDS protocol development, in order for drones to learn about their neighbors, we created a new packet that we called *beacon-like packet*. This packet mimics the behaviour of the IEEE 802.11s *beacon* that are exchanged between the nodes in a mesh network in regular intervals. Our *beacon-like packet* works exactly the same as IEEE 802.11s *beacon* except the information they carry is different than beacons. *Beacon-like packet* carries the drone information such as location, marker, ID etc. as well as information about drone's neighbors such as their markers, IDs, positions etc. as shown in Fig. 6.

We created this new packet at the MAC layer instead of application layer which aims to improve the performance. Basically, the E-CDS protocol is based on the broadcast of

Beacon-like packet

node ID	marker	role	position	neighbor's information
1 byte	1 byte	1 byte	24 bytes	varying byte > 24 bytes

Claim packet

position	isClaimer	doIMove	isCloser	isFurther
24 bytes	1 byte	1 byte	1 byte	1 byte

Fig. 6. Contents of the beacon-like and claim packets.

Algorithm 3: Moving Drone Selection

```
Input  : Selected drone, its neighbors and
         newLocation
Output: Calculated movingDrone
if drone.marker ≠ True then
    neighborDistances = []; /* an empty list */
    myDst = calcDist(drone, newLocation);
    for neighbor in neighbors do
        dst = calcDist(neighbor, newLocation);
        neighborDistances.append(dst);
    end
    if myDst < min(neighborDistances) then
        drone.SetIsClaimer();
        broadcastClaim();
    end
    otherClaimersDst = []; /* an empty list */
    if drone.isClaimer == True and drone receives
     claimPacket then
        sender = claimPacket.getSender();
        senderDst = calcDst(sender, newLocation);
        otherClaimersDst.append(senderDst);
        if senderDst < myDst then
            broadcastClaim(isFurther = True);
        else
            broadcastClaim(isCloser = True);
        end
    end
    if myDst < min(otherClaimersDst) then
        broadcastClaim(doIMove = True);
        broadcastMovingPacket();
    end
end
```

these *beacon-like packets* and the processing of the information with Algorithm 1 and 2. Packets will not be traveling through all layers of TCP/IP protocol stack to reduce processing delays. Eventually, each node will know whether it is in E-CDS or not. This information is then used for moving tasks as described next.

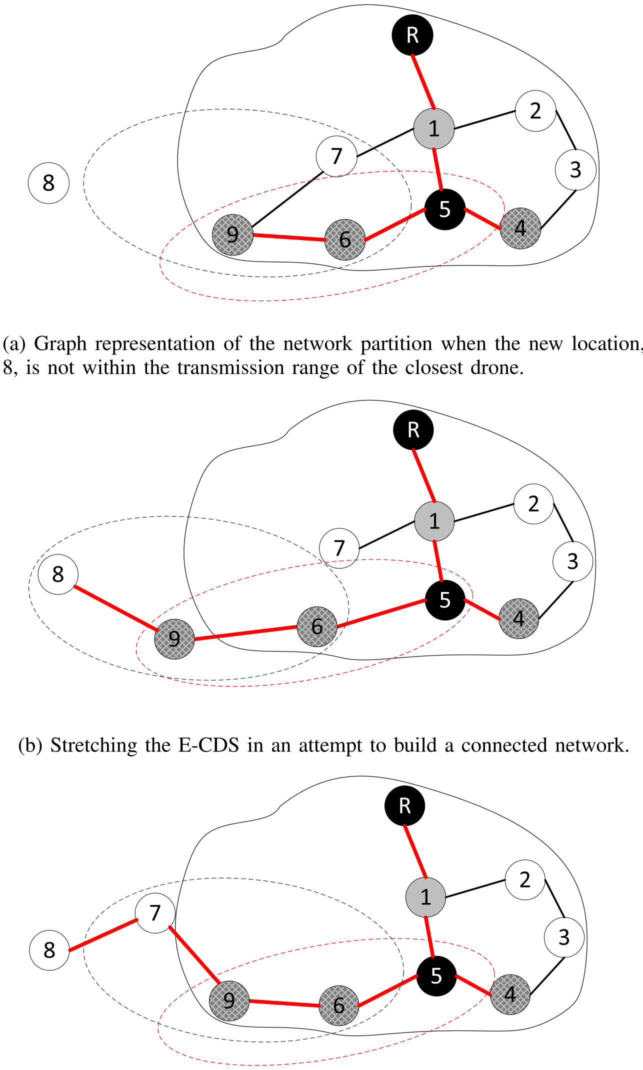
D. Moving Drone Selection and Movements

As mentioned, through the control center, there can be certain task requests which require some drones to move to certain locations. Once the E-CDS of the AWMN is found, our next objective is to determine the moving drones to perform the required task(s) from the control center in a distributed fashion. We would like to note that there has already been some proposed ideas in 2-D WMNs for such node movement process [25] but none of them was implemented in a realistic and distributed setting to understand the impact on the overall network operations. Our goal here is to pick the most suitable of these ideas to adopt for AWMN settings and more importantly implement them in a realistic setting (i.e., within 802.11s standard) to observe the overhead for drone services in an ITS application scenario.

Back to our approach, optimally, the closest dominatee drone to the new location should be moved to the new location. However, after the move, the connectivity of this drone to the rest of the AWMN should be maintained. We achieve this in two ways: 1) The new location still keeps the moved node within the transmission range of any of the drones in the AWMN by stretching their positions if needed; 2) If the moved node will be out of transmission range, then the other closest dominatees are picked to fill in the space. The number of such dominatees is computed by the moving drone and broadcast to its neighbors.

More specifically, our moving drone selection protocol is presented at Algorithm 3 and is as follows: Each dominatee drone (i.e., drones that are not part of the CDS) calculates whether it is the closest drone to the new location in its 2-hop neighborhood (line 2-8). Drones do not know about other drones except for the ones that are in their 2-hop neighborhood. If a drone calculates that it is the closest drone, it marks itself as a *claiming drone* and broadcasts a claim packet (line 9-10). There can be multiple claiming drones depending on the topology and the roles/IDs of the drones. If there are more than one claiming drones, each claiming drone needs to learn about the existence of other claiming drones. Additionally, they all need to know which claiming node is actually the closest to the new location. To achieve this, we introduced a new packet called *claim packet* which is a broadcast packet at the MAC layer and is only processed by the claiming drones upon reception. Other drones who receive these packets simply ignore them. When a claiming drone receives a claim packet, it compares its location with the location embedded into the packet which belongs to the location of the claiming drone that sent the packet (line 13-17). If the sender is closer to the new location, the receiver drone marks itself as not-moving and sends a claim packet to the sender to let it know that it (the sender) is closer to the new location (line 18). In this way, the sender of the claim packet can learn that it is closer to the new location than other claiming drones. If the receiver of a claim packet is closer, then similar procedure follows. The receiver sends back a claim packet to the sender that it (the receiver) is closer to the new location so that the sender can mark itself as not-moving (line 20). After the exchange of claim packets is completed and once the closest claiming drone is determined, closest claiming drone broadcasts a claim packet to let other claiming drones know that it will be the moving node (line 23-24).

Once a node knows that it will be the moving node, it computes its distance to the other nodes to find out whether it will be out of range with its current neighbors. If this is the case, then it computes if any of its neighbors 1) should stretch to be within its transmission range; or 2) should move to fill the gap to enable a multi-hop connection. The node then



(a) Graph representation of the network partition when the new location, 8, is not within the transmission range of the closest drone.

(b) Stretching the E-CDS in an attempt to build a connected network.

(c) Moving one of the dominatees to fill the gap to enable the connectivity.

Fig. 7. The illustration of the proposed heuristic for connectivity restoration. Note that for the presentation clarity, we show the transmission ranges and node positions in 2-D while in the algorithm we assumed that the transmission ranges are spherical.

broadcasts a *moving packet* (line 25) to its neighbors which is further broadcast to its 2-hop neighbors and moves to the new location. The overall connectivity maintenance process is shown in Fig. 7.

V. Performance Evaluation

A. Experiment Setup

We implemented and simulated the proposed AWMN communication architecture and connectivity maintenance protocol in ns-3 network simulator version 3.29 [32]. In our simulations, we chose the new location such that, moving one drone to the new location would be enough. Multiple simulation configurations were used to perform the different measurements. All simulations results are an average of results from 30 simulations with different topologies. In these topologies, drones are randomly placed within a rectangular prism having the dimensions $400 \times 250 \times 250$ such that the minimum Euclidean distance between drones is 80m, the maximum Euclidean distance between a drone and the drone closest to it is 100m. The coordinates of the new location is (200, 125, 275) in all simulations. UDP protocol is used for data transmissions as we assume most of the data transmission will be video traffic. We used ns-3 network performance tool v2 [33] for measurements related to the UDP transmissions. Data rate of the UDP transmissions are 50kbps, UDP packet size is 128bytes. Simulation time is 103 seconds and UDP transmissions starts at 3rd second of the simulation. The transmission of beacon-like packets and the calculations for E-CDS and moving node starts at 50th second of the simulations. The IEEE 802.11s' transmission range is set to 100m.

B. Experiment Metrics and Baselines

While the main focus of the paper is to investigate the performance and overhead of the distributed approach, there needs to be a baseline to compare the distributed results to. Our baseline is the centralized approach to the connectivity maintenance problem. In the centralized approach, E-CDS is calculated by the flying gateway in a centralized manner. Similarly, moving node calculation is also performed by the flying gateway without needing to exchange any external packets in addition to those used in IEEE 802.11s by standard. We used the centralized CDS calculation method in [34]. Therefore, throughout the experiment results, we compare the distributed version to the centralized version. Furthermore, in order to have more comprehensive comparison, we included two more existing distributed approaches in the literature and compared these with the main distributed and centralized approaches throughout the experiments. First of these distributed approaches is taken from [35] and it is called the *least degree & closest node* approach. It is *best candidate selection* part of the *DARA-1C* algorithm. As the name suggests, this approach restores the connectivity of a partitioned network by choosing the closest node to the new location among the least degree nodes. In other words, first drones with least number of neighbors are determined then among these drones, closest one to the new location is sent. Second of these distributed approaches is from [25] and it is called *block movement* approach. In this approach, all drones in the network move towards the new location together. As both of these approaches are distributed methods, we used our existing beacon-like packet for drones to get neighboring information.

We used the following performance metrics for the evaluation:

- The *end-to-end delay* of the data travelling from drones to the flying gateway. This includes queuing and transmission delays for all links.
- The *packet delivery ratio* (PDR) is the percentage of successfully delivered packets from a source to flying gateway in the transmissions.
- *Total time* it takes to decide which drone(s) will be sent to the new location plus the moving time of the moving drone.

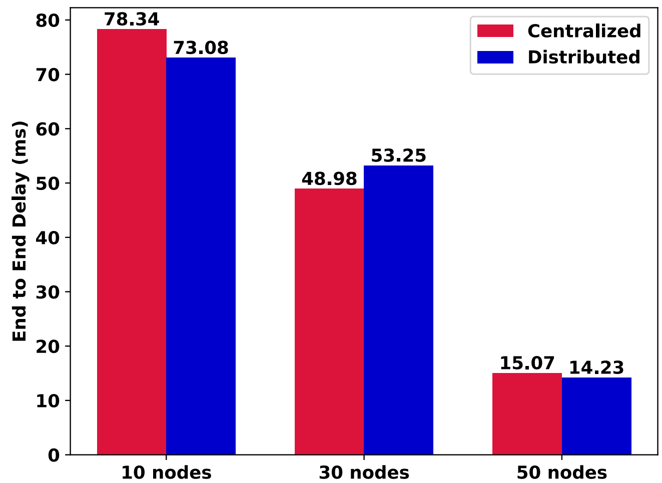


Fig. 8. The impact of connectivity maintenance on end-to-end delay under varying number of drones in the network.

- *Total distance* taken by the moving drone as the result of moving to a new location given by the control center.

C. Experiment Results and Discussions

We present experiment results for 6 scenarios and compare each experiment result for centralized and distributed versions.

1) Impact on the End-to-End Delay Performance: In this experiment, assuming there is a moving node and its moving is handled by our protocol, we investigated the effect of number of drones on the end-to-end delay of data traveling from transmitting drones to the flying gateway. In this scenario, 5 of the drones are selected randomly to transmit UDP data packets to the flying gateway simultaneously. Possible packet collisions at the receiver is prevented by slightly randomizing the start time of the transmissions of the transmitting drones. This experiment simulates the real life scenario where some drones transmit video to the control center through the flying gateway.

As can be seen in Fig. 8, the difference in end-to-end delay between centralized and distributed version is minimal which shows that additional packet transmissions due to connectivity maintenance (i.e., beacon-like packet, claim packet) in the distributed version does not hinder the performance of transmissions of the transmitting nodes. Indeed, in case of 10 nodes, our approach performs better than the centralized as the contention overhead to access the channel will be even less with reduced number of nodes. In addition, we observe that the average end-to-end delay for data transmissions from 5 of the drones to the flying gateway decreases with increasing number of drones. Average end-to-end delay for 10 drones was measured as 70ms, 50ms for 30 nodes and 20ms for 50 drones. This is because when number of drones in the simulation increases, transmitting drones would have more route options and thus shorter paths to the flying gateway. Consequently, shorter paths result in lower end-to-end delay.

2) Impact on the Packet Delivery Ratio Performance: In this experiment, we assessed the PDR for the UDP transmissions when there is ongoing connectivity maintenance process within the AWMN. Fig. 9 presents the results for

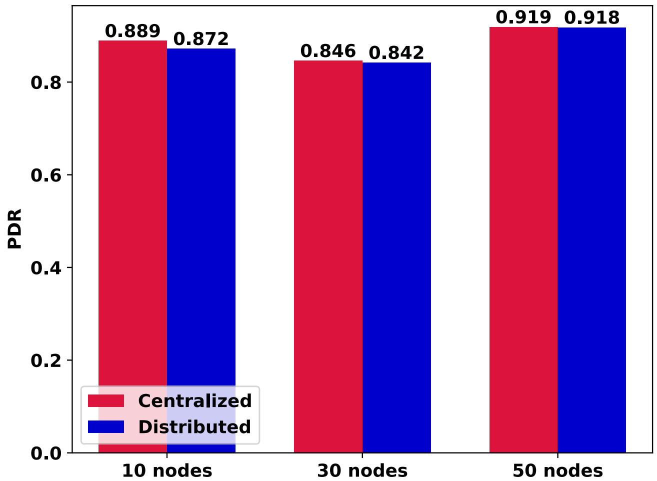


Fig. 9. The impact of connectivity maintenance on PDR under varying number of drones in the network.

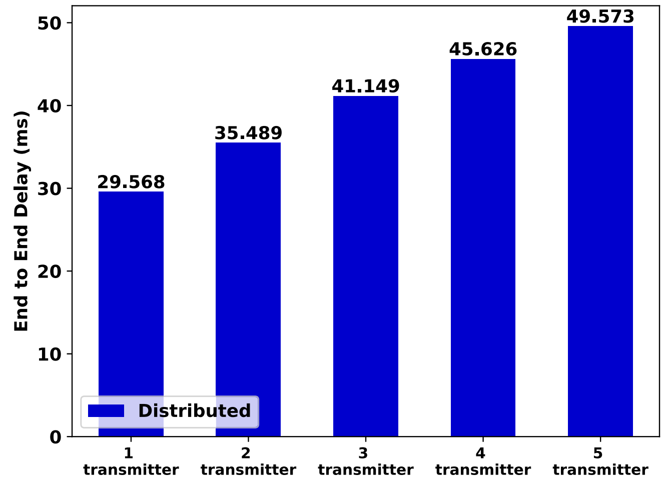


Fig. 10. The impact of varying the number of transmitting drones on end-to-end delay.

this experiment. We observed around 90% PDR on average for our approach which is typically high considering that the transmissions are UDP. The PDR for the centralized approach is almost similar to that of our approach, which again indicates that our approach brings almost negligible overhead due to the way it was designed and implemented. We also observe that there is not really a clear correlation between the number of drones in the simulation and the PDR. This might be due to the fact that the number of transmitting drones is kept constant during these experiments and thus the produced data amount does not change. However, in case of more nodes, there may be more options for data routes which would be expected to help slightly improve PDR (i.e., shorter route or less congested route). We see that this makes little impact when we look at the results for 50 nodes case. This is because for other cases, the available paths are stable and are not impacted by any E-CDS activities whether it is centralized or distributed.

3) Varying Number of Transmitting Drones: In the previous experiments, we kept the number of transmitting drones fixed. In this experiment, we wanted to investigate if changing the number of transmitting drones would make an impact on our approach in terms of PDR and end-to-end delay. The number of transmitting nodes are varied from 1 to 5 and the results are depicted in Fig. 10 and Fig. 11. The total number of drones in the simulations for this experiment is 30.

Results show a clear correlation between the number of transmitting drones and end-to-end delay and PDR. The end-to-end delay increases as the number of transmitting drones increases as seen in Fig. 10. This is due to the fact that more packets are to be transmitted and thus there will be more contention in the network. We also see that the increase is pretty linear and thus the impact of connectivity maintenance in the network is independent of the number of transmitting sources which hints about the scalability of our approach.

This scalability feature is also confirmed when we looked at the PDR results in Fig. 11. The PDR does not decrease like the end-to-end delay. Its value never goes below %85 on average. This means that the PDR can keep up with the

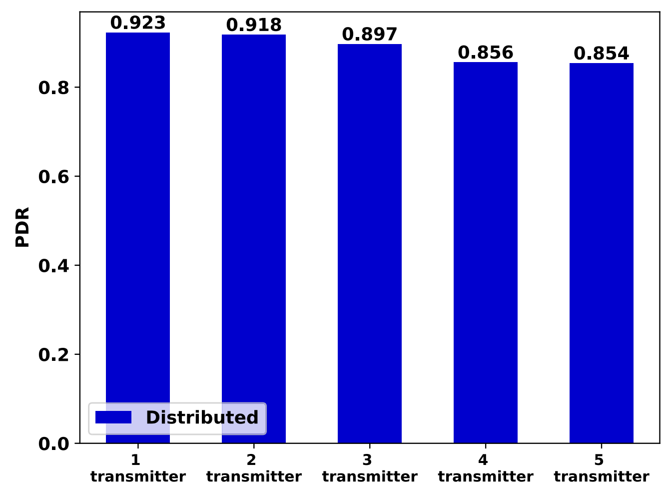


Fig. 11. The impact of varying the number of transmitting drones on PDR.

increased number of source nodes generating traffic thanks to the minimal overhead of our approach.

4) Total Time: In this experiment, we measured the total time it takes to calculate the E-CDS, find the claiming drones, decide on which drone to move to the new location, moving time of the drone to the new location. We varied the number of drones involved. We assumed drones move with a speed of 15m/s [36]. In this experiment, we compare the *least degree & closest drone* and *block movement* methods along with the centralized and distributed methods.

Fig. 12 shows the total time under varying number of drones for all 4 approaches. First, the centralized approach computes the E-CDS much faster since there is no packet exchange. Second and more importantly, moving time of the drone to the new location is less in centralized approach since a much less E-CDS (i.e., minimum E-CDS) could be calculated and therefore there will be more dominatee drone options (i.e., closer) to pick for sending to the new location. This may not be the case in the distributed approach since the E-CDS computed in the distributed approach is not necessarily the minimum

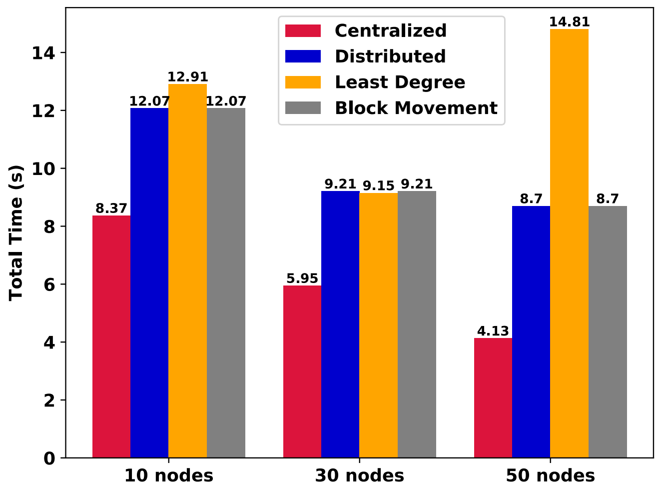


Fig. 12. Total time including; computing the E-CDS, finding the claiming nodes, deciding which node to send, and moving time of that node to the new location for all 4 approaches under varying number of drones in the network.

CDS that will lead to best dominatee selection. This will lead to a longer travel as will be also seen in Table II and thus more time. Least degree & closest drone approach on the other hand, has a total time very close to that of distributed approach when there are 10 and 30 drones in the network. When there are 50 nodes in the network, total time spikes almost 15 seconds which is nearly two times of the total time of the distributed approach. This is because least degree approach could not pick good drones to send to the new location. Picked drones are far away from the new location causing longer travel times to the destination. Block movement approach has the exact same total time of the distributed approach since moving one drone or all drones simultaneously is the same in terms of their associated moving time. However, as will be seen in Table II, distinguishing factor between two approaches will be the total traveling distance of the drones.

We also solely looked at the E-CDS computation and moving node selection time for centralized and distributed approaches. In other words, it is the total time excluding the moving time of the moving drone. The results are shown in Table I. As can be seen for the centralized approach, the calculation of the E-CDS and node selection for movement takes less than 1ms. Distributed approach on the other hand takes around 1.5s on average to decide on which node to send to the new location. Least degree & closest drone and block movement approaches are not shown in the Table I since they share the same results with the distributed approach. The performance of the distributed approach can be further increased by decreasing the interval of the beacon-like packets since nodes learn about their neighbors more quickly when beacon-like packets are sent more frequently. However, the major delay is still due to the actual movement time of the moving drone as predicted. Nevertheless, overall, there is around 3 seconds difference on total time between the centralized and distributed approaches which could be tolerable unless there is any real-time requirement for the ITS applications.

TABLE I

TOTAL TIME EXCLUDING THE MOVING TIME OF THE MOVING DRONE TO THE NEW LOCATION UNDER VARYING NUMBER OF DRONES IN THE NETWORK

Number of Drones	Centralized	Distributed
10	<1 ms	2.625 seconds
30	<1 ms	1.239 seconds
50	<1 ms	1.116 seconds

TABLE II

TOTAL DISTANCE TRAVELED BY THE MOVING DRONE(S) FOR ALL 4 APPROACHES UNDER VARYING NUMBER OF DRONES IN THE NETWORK

Number of Drones	Centralized	Distributed	Least Degree	Block Movement
10	125.6 m	141.7 m	154.24 m	1417 m
30	89.31 m	119.56 m	118.7 m	3586.8 m
50	61.98 m	112.51 m	205.37 m	5625.5 m

5) Total Distance: This experiment examines the total distance the moving node takes when moving to the new location. This experiment is particularly important because drones consume from their battery while moving from one location to another. It is important to keep battery consumption of the drones low for more reliable service. Same as in the previous experiment, *least degree & closest drone* and *block movement* approaches are compared with the centralized and distributed approaches in this experiment. The experiment results are given in Table II.

Clearly, total distance is the least in the centralized approach. As explained before, this is because centralized version can calculate an E-CDS which is closer to the optimal solution in size and thus there will be more dominatee nodes available to move. Consequently, the dominatee node picked in the centralized approach will more likely to be closer to the destination than the one in the distributed approach which results in lesser total distance. As seen in Table II, as the number of drones in the simulation increases, the distance gap between the centralized and distributed approaches increases. This can be attributed to the fact that with increased network size, the E-CDS size computed for the distributed approach will grow faster than the centralized one. As a result, the number of dominatee nodes available to move will be diminishing. Nevertheless, our approach seems to be performing very close to the centralized one when the number of drones in the swarm is smaller. For increased network size, there is a trade-off between the size of the E-CDS and the convergence time. If the network size will be larger, the distributed approach can be tuned to work with k neighboring nodes where $k > 2$. When least degree & closest drone approach is examined, similar behaviour like in the total time experiment is observed. While total distance is very close to that of distributed approach for 10 and 30 nodes case, total distance of least degree approach for 50 nodes is almost two times of the total distance of distributed approach. This is because further nodes are picked by the algorithm to move to the new location which are not as optimal as distributed case. For block movement method, total distance is way worse compared to all other methods since all

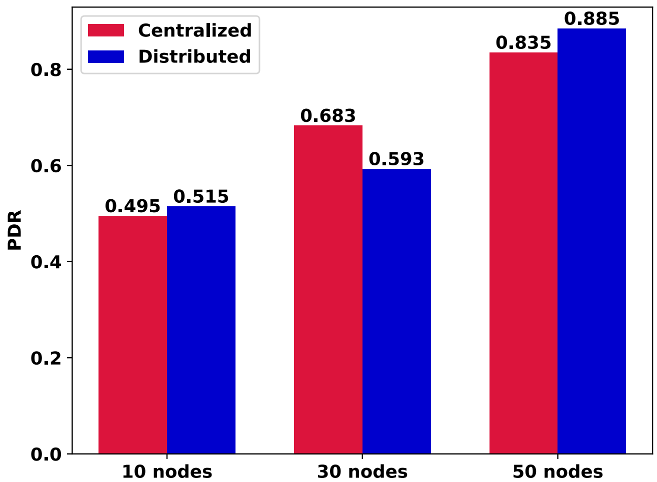


Fig. 13. PDR of the moving drone under varying number of drones in the network.

of the drones are moved together towards the new location. For example, for 10 nodes case, while total distance of the distributed approach is 141.7 meters, it is 10 times of that value for block movement approach which is 1417 meters. The results overall show that our approach brings the best travel distance among all the distributed approaches.

6) Moving Drone PDR: In the last experiment, we looked at the PDR of the moving drone. The aim of this experiment is to understand if a moving drone can still reliably transmit data to the flying gateway as there may be changes to its routes during movement. Experiment results are presented in Fig. 13. The results indicate that PDR of the moving drone increases for both approaches as the total number of nodes in the network increases. This is because when the network is denser, underlying mesh protocol can find more effective/reliable paths to the gateway node to route the data packets. The important observation from these results is the behavior of our distributed approach compared to centralized one. It is understandable that with our approach, there is the overhead of the E-CDS message broadcasts and thus accessing the channel could be delayed regardless of the network size. This will cause some reduction in the PDR. However, we see that this reduction is very minor and in some cases, distributed approach's PDR is even a bit higher. We speculate that this may be due to the changes to the existing routes. When nodes move, some routes may need to be updated. In some cases, such updates may benefit distributed approach better (i.e., there can be shorter paths compared to centralized approach) as the topologies are purely random. As a result, we can say that the distributed approach performs almost as well as the centralized approach.

VI. Conclusion

In this paper, we presented a distributed approach for connectivity maintenance problem among a swarm of drones in post-disaster transportation applications. We introduced a more specific version of the CDS problem that we call E-CDS where there are a set of nodes that have to be put into the CDS because of their higher priority in the network. We first proved that computing an E-CDS is also NP-complete. Then we adopted a distributed CDS heuristic for our case that is used to determine which node(s) to move in case of moving requests from the control center. For implementation of the distributed protocol, we included two additional new packets that are called beacon-like packet and claim packet. Using these new packets, nodes could learn their neighboring information and use it to compute an E-CDS and decide on which node to move to a given incident location.

We analyzed the performance of the distributed approach in detail in the performance evaluation section through an implementation as part of IEEE 802.11s in ns-3 simulator. Specifically, the overhead of the algorithms, end-to-end delay, packet delivery ratio were examined in a series of simulations. The results show that the overhead of the distributed approach is negligible and the distributed approach performs as good as the centralized approach especially for smaller network sizes.

In the future, we plan to look into the same problem without GPS availability assumption. Additionally, the case where there are more than one flying gateway among the swarm of drones is left as a future work too.

References

[1] S. Saha, S. Nandi, P. S. Paul, V. K. Shah, A. Roy, and S. K. Das, "Designing delay constrained hybrid ad hoc network infrastructure for post-disaster communication," *Ad Hoc Netw.*, vol. 25, pp. 406–429, Feb. 2015.

[2] M. A. Hadi, D. Quigley, P. Sinha, and L. Hsia, "Benefit and cost parameters of intelligent transportation systems: Use in evaluations of deployment analysis systems in Florida," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1910, no. 1, pp. 57–63, Jan. 2005.

[3] M. Alam, J. Ferreira, and J. Fonseca, *Introduction to Intelligent Transportation Systems*. Cham, Switzerland: Springer, 2016, pp. 1–17.

[4] E. N. Barmpounakis, E. I. Vlahogianni, and J. C. Golias, "Unmanned aerial aircraft systems for transportation engineering: Current practice and future challenges," *Int. J. Transp. Sci. Technol.*, vol. 5, no. 3, pp. 111–122, Oct. 2016.

[5] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging UAVs for disaster management," *IEEE Pervas. Comput.*, vol. 16, no. 1, pp. 24–32, Jan. 2017.

[6] H. Menouar, I. Guvenc, K. Akkaya, A. S. Uluagac, A. Kadri, and A. Tuncer, "UAV-enabled intelligent transportation systems for the smart city: Applications and challenges," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 22–28, Mar. 2017.

[7] R. Saleri *et al.*, "UAV photogrammetry for archaeological survey: The theaters area of Pompeii," in *Proc. Digit. Heritage Int. Congr. (DigitalHeritage)*, vol. 2, Oct. 2013, pp. 497–502.

[8] N. Saputro, K. Akkaya, R. Algin, and S. Uluagac, "Drone-assisted multi-purpose roadside units for intelligent transportation systems," in *Proc. IEEE 88th Veh. Technol. Conf. (VTC-Fall)*, Aug. 2018, pp. 1–5.

[9] *IEEE Standard for Information Technology—Local and Metropolitan Area Networks—Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments*, IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11n-2009, IEEE Std 802.11w-2009), Jul. 2010, pp. 1–51. [Online]. Available: https://ieeexplore.ieee.org/document/5514475

[10] *IEEE Standard for Information Technology–Telecommunications and Information Exchange Between Systems–Local and Metropolitan Area Networks–Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 10: Mesh Networking*, IEEE Std 802.11s-2011 (Amendment to IEEE Std 802.11-2007 as amended by IEEE 802.11k-2008, IEEE 802.11r-2008, IEEE 802.11y-2008, IEEE 802.11w-2009, IEEE 802.11n-2009, IEEE 802.11p-2010, IEEE 802.11z-2010, IEEE 802.11v-2011, IEEE 802.11u-2011), Sep. 2011, pp. 1–372. [Online]. Available: https://ieeexplore.ieee.org/document/6018236

[11] L. Vangelista, A. Zanella, and M. Zorzi, "Long-range IoT technologies: The dawn of LoRa," in *Future Access Enablers for Ubiquitous and Intelligent Infrastructures*. Cham, Switzerland: Springer, 2015, pp. 51–58. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-27072-2_7

[12] S. Guha and S. Khuller, "Approximation algorithms for connected dominating sets," *Algorithmica*, vol. 20, no. 4, pp. 374–387, Apr. 1998.

[13] (2019). *A Discrete-Event Network Simulator for Internet Systems, Ns-3 Version 3.29*. [Online]. Available: https://www.nsnam.org/releases/ns-3-29/

[14] H. Ghazzai, H. Menouar, and A. Kadri, "On the placement of UAV docking stations for future intelligent transportation systems," in *Proc. IEEE 85th Veh. Technol. Conf. (VTC Spring)*, Jun. 2017, pp. 1–6.

[15] J. Chen *et al.*, "Long-range and broadband aerial communication using directional antennas (ACDA): Design and implementation," *IEEE Trans. Veh. Technol.*, vol. 66, no. 12, pp. 10793–10805, Dec. 2017.

[16] Y. Liu, K. Xiong, Q. Ni, P. Fan, and K. B. Letaief, "UAV-assisted wireless powered cooperative mobile edge computing: Joint offloading, CPU control, and trajectory optimization," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2777–2790, Apr. 2020.

[17] D.-H. Lee, "CNN-based single object detection and tracking in videos and its application to drone detection," *Multimedia Tools Appl.*, pp. 1–12, Oct. 2020. [Online]. Available: https://link.springer.com/article/10.1007/s11042-020-09924-0

[18] M. R. Garey and D. S. Johnson, *Computers and Intractability*. New York, NY, USA: W. H. Freeman and Company, 1979.

[19] Z. Yuanyuan, X. Jia, and H. Yanxiang, "Energy efficient distributed connected dominating sets construction in wireless sensor networks," in *Proc. Conf. Commun. Mobile Comput. (IWCMC)*, 2006, pp. 797–802.

[20] P.-J. Wan, K. M. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," in *Proc. 21st Annu. Joint Conf. IEEE Comput. Commun. Societies*, vol. 3, Jun. 2002, pp. 1597–1604.

[21] D. Dubhashi, A. Mei, A. Panconesi, J. Radhakrishnan, and A. Srinivasan, "Fast distributed algorithms for (weakly) connected dominating sets and linear-size skeletons," *J. Comput. Syst. Sci.*, vol. 71, no. 4, pp. 467–479, Nov. 2005.

[22] F. Dai and J. Wu, "An extended localized algorithm for connected dominating set formation in ad hoc wireless networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, no. 10, pp. 908–920, Oct. 2004.

[23] J. Yu, N. Wang, G. Wang, and D. Yu, "Connected dominating sets in wireless ad hoc and sensor networks—A comprehensive survey," *Comput. Commun.*, vol. 36, no. 2, pp. 121–134, Jan. 2013.

[24] X. Qi, P. Yuan, Q. Zhang, and Z. Yang, "CDS-based topology control in FANETs via power and position optimization," *IEEE Wireless Commun. Lett.*, vol. 9, no. 12, pp. 2015–2019, Dec. 2020.

[25] M. Younis, I. F. Senturk, K. Akkaya, S. Lee, and F. Senel, "Topology management techniques for tolerating node failures in wireless sensor networks: A survey," *Comput. Netw.*, vol. 58, pp. 254–283, Jan. 2014.

[26] K. Akkaya and F. Senel, "Detecting and connecting disjoint sub-networks in wireless sensor and actor networks," *Ad Hoc Netw.*, vol. 7, no. 7, pp. 1330–1346, Sep. 2009.

[27] N. Saputro, K. Akkaya, and S. Uluagac, "Supporting seamless connectivity in drone-assisted intelligent transportation systems," in *Proc. IEEE 43rd Conf. Local Comput. Netw. Workshops (LCN Workshops)*, Oct. 2018, pp. 110–116.

[28] A. Kurt and K. Akkaya, "Connectivity maintenance extensions to IEEE 802.11s MAC layer in ns-3," in *Proc. Workshop Ns-3*, Jun. 2020, pp. 17–24.

[29] Google. (2020). *Google WiFi Mesh Router*. [Online]. Available: https://store.google.com/us/product/nest_wifi

[30] S. M. S. Bari, F. Anwar, and M. H. Masud, "Performance study of hybrid wireless mesh protocol (HWMP) for IEEE 802.11s WLAN mesh networks," in *Proc. Int. Conf. Comput. Commun. Eng. (ICCCE)*, Jul. 2012, pp. 712–716.

[31] A. Nayyar, "Flying adhoc network (FANETs): Simulation based performance comparison of routing protocols: AODV, DSDV, DSR, OLSR, AOMDV and HWMP," in *Proc. Int. Conf. Adv. Big Data, Comput. Data Commun. Syst. (icABCD)*, Aug. 2018, pp. 1–9.

[32] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, "Network simulations with the ns-3 simulator," *SIGCOMM Demonstration*, vol. 14, no. 14, p. 527, 2008.

[33] N. Jevtic. (2019). *Ns-3 Network Performance Analysis Tool on Application Layer*. [Online]. Available: https://github.com/neje/ns3-network-performance-tool-v2

[34] S. Butenko, C. Oliveira, and P. M. Pardalos, "A new algorithm for the minimum connected dominating set problem on ad hoc wireless networks," in *Proc. CCCT*, 2003, pp. 39–44.

[35] A. A. Abbasi, M. Younis, and K. Akkaya, "Movement-assisted connectivity restoration in wireless sensor and actor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 9, pp. 1366–1379, Sep. 2009.

[36] Dji. (2020). *Dji Phantom 3 Pro Max Speed*. [Online]. Available: https://www.dji.com/phantom-3-pro/info

Ahmet Kurt received the joint B.S. degrees from Antalya Bilim University, Antalya, Turkey, in 2018. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the Florida International University, Miami, USA. His current research interests include wireless networks, payment channel networks, and Bitcoin's Lightning Network.

Nico Saputro received the Ph.D. degree in electrical engineering from Florida International University in 2016. From 2016 to 2020, he joined the Department of Electrical and Computer Engineering, Florida International University as a Postdoctoral Associate. He is an Associate Professor and the Head of Center for Network, Intelligence and Cyber Security (CeNICS) with the Department of Electrical Engineering, Parahyangan Catholic University, Bandung, Indonesia. His research interests include security and privacy in a variety of wireless and cellular networks, communication protocols for Smart Grid, moving target defense, SDN, precision agriculture/smart farming, and the Internet of Things. He was a recipient of Fulbright Presidential Scholarship in 2009 to 2012.

Kemal Akkaya (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Maryland Baltimore County in 2005. He is a full Professor with the Department of Electrical and Computer Engineering, Florida International University. He leads the Advanced Wireless and Security Lab (ADWISE), Department of Electrical and Computer Engineering. His current research interests include security and privacy, the Internet of Things, and cyber-physical systems. He has received Top Cited Article Award from Elsevier in 2010. He was the General Chair of IEEE LCN 2018 and TPC Chair for IEEE ICC Smart Grid Communications. He is the Area Editor of *Ad Hoc Network Journal* and serves on the editorial board of IEEE COMMUNICATION SURVEYS AND TUTORIALS.

A. Selcuk Uluagac received the M.S. and Ph.D. degrees from the Georgia Institute of Technology, and the M.S. degree from Carnegie Mellon University. He is currently a member of the faculty with the Department of Electrical and Computer Engineering, Florida International University, as an Eminent Scholar Chaired Associate Professor, where he directs the Cyber-Physical Systems Security Lab (CSL), focusing on security and privacy of the Internet of Things and cyber-physical systems. Before joining Florida International University, he was a Senior Research Engineer with the School of Electrical and Computer Engineering, Georgia Institute of Technology. He received the U.S. National Science Foundation CAREER Award and U.S. Air Force Office of Sponsored Research's Summer Faculty Fellowship in 2015. He serves on the editorial boards of the IEEE TRANSACTIONS ON MOBILE COMPUTING, *Computer Networks*, and the IEEE COMMUNICATIONS AND SURVEYS AND TUTORIALS.