IoT

MDPI

Article

Trajectory Planing for Cooperating Unmanned Aerial Vehicles in the IoT †

Emmanuel Tuyishimire [1,*], Antoine Bagula [2], Slim Rekhis [3] and Noureddine Boudriga [3]

[1] Department of Knowledge and Information Stewardship, University of Cape Town, Cape Town 7701, South Africa
[2] Computer Science Department, University of the Western Cape, Cape Town 7535, South Africa; bbagula@uwc.ac.za
[3] Communication Networks and Security Research Laboratory, Cartage University, Tunis 1054, Tunisia; slim.rekhis@gmail.com (S.R.); noure.boudriga2@gmail.com (N.B.)
* Correspondence: emmanuel.tuyishimire@uct.ac.za

† This paper is an extended version of our paper published in 2017 Cooperative data muling from ground sensors to base stations using UAVs. In Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Heraklion, Greece, 3–6 July 2017; pp. 35–41.

Abstract: The use of Unmanned Aerial Vehicles (UAVs) in data transport has attracted a lot of attention and applications, as a modern traffic engineering technique used in data sensing, transport, and delivery to where infrastructure is available for its interpretation. Due to UAVs' constraints such as limited power lifetime, it has been necessary to assist them with ground sensors to gather local data, which has to be transferred to UAVs upon visiting the sensors. The management of such ground sensor communication together with a team of flying UAVs constitutes an interesting data muling problem, which still deserves to be addressed and investigated. This paper revisits the issue of traffic engineering in Internet-of-Things (IoT) settings, to assess the relevance of using UAVs for the persistent collection of sensor readings from the sensor nodes located in an environment and their delivery to base stations where further processing is performed. We propose a persistent path planning and UAV allocation model, where a team of heterogeneous UAVs coming from various base stations are used to collect data from ground sensors and deliver the collected information to their closest base stations. This problem is mathematically formalised as a real-time constrained optimisation model, and proven to be NP-hard. The paper proposes a heuristic solution to the problem and evaluates its relative efficiency through performing experiments on both artificial and real sensors networks, using various scenarios of UAVs settings.

Keywords: real-time visitation; cooperative UAVs; path planning; clustered network

Citation: Tuyishimire, E.; Bagula, A.; Rekhis, S.; Boudriga, N. Trajectory Planing for Cooperating Unmanned Aerial Vehicles in the IoT. *IoT* **2022**, *3*, 147–168. https://doi.org/10.3390/iot3010010

Academic Editors: Enrique Moguel and Jaime Galán-Jiménez

Received: 22 January 2022
Accepted: 15 February 2022
Published: 24 February 2022

1. Introduction

The use of UAVs has emerged as a flexible and cost-efficient alternative to traditional traffic engineering techniques that have been used in IoT settings, in order to transport sensor readings from their points of collection to their processing places. However, the joint path finding and resource allocation for a team when tasked to achieve collaborative data muling is still an issue that require further investigations. On the other hand, accurate solutions to data muling problems are still scarce, especially when considering the limited flying capacity of the battery-powered UAVs. The issues related to the efficient task allocation to a team of UAVs under stringent data collection requirements, such as real time data collection, still need to be addressed. This would benefit many task assignment models. especially when UAVs have different specifications (speeds, battery, lifetime, memory, functionalities, etc.) and only fresh and complete information need to be collected. Furthermore, persistent collection requirement needs to be addressed and this requires the data muling system to deal with outdated or premature sensor readings.

Potential applications of a such real-time data muling model include (i) city surveillance in order to evaluate risks and respond with appropriate actions by having a team of UAVs persistently visiting locations of interests in a smart city for public safety, parking spots localization [1], and pollution monitoring [2]; (ii) drought mitigation to support small scale farming in rural areas [3,4] by using a team of UAVs to collect farmland image collection and processing these images to achieve situation recognition for precision irrigation; (iii) periodic surveillance of buildings and cities' infrastructures for structural health monitoring and maintenance; and (iv) extension of the reach of community mesh networks in rural settings for healthcare [5,6] by using a team of UAVs (such as drones) as wireless access points.

Sensors visitation under the fuel consumption constraints was addressed in [7], the visitation under the revisit deadline constraint was proposed in [8], and a path planning model has been proposed in [9,10]. Both works assumed a single moving agent (UAV), which optimally visits various targets. Ref. [11] proposes a cooperative UAVs model where many targets are visited by a team of UAVs for persistent surveillance and pursuit. In this work, the UAVs do not communicate with each other but rather rely on the information from the static underground sensors, which are optimally placed as proposed in [12]. However, all these models do not consider the persistent data delivery and heterogeneity of UAVs, which might have different fabrics and characteristics. Furthermore, neither the energy/battery consumption while the UAVs are waiting for the updated information from the terrestrial sensor network nor the penalty associated with stale information due to late visitation by the UAV to the sensor nodes have been accounted for.

While models were proposed in [13–16] for the periodic and persistent UAVs visitation of a single target from different positions, the models do not consider the path planning issues, which are as necessary as the path planning, especially for restricted environments. After deploying ground networks to assist UAV visitations [17–19], multiple UAV models have also been employed to visit many assisting sensors (target) [20,21]. Even if models to assist UAVs with ground sensor network have been recommended (see [18,22–24]), the efficient targets visitation and the assignment issue have not been addressed. However when heterogeneous UAVs have to visit multiple sensors, an UAV assignment model is required to complement path planning models.

This paper assumes a restricted and complex network, where sensors are not only connected in terms of their ability to forward data to each other, but also in terms of the possible paths UAVs may use to visit each sensor from any base station or any other sensor in a region of interest (there could be two links between two nodes). We propose a persistent and real-time path planning model together with a task allocation model, where a team of heterogeneous UAVs coming from various base stations are used to collect sensor readings from ground sensors and deliver the collected information to their closest base stations. The underlying data muling problem is (i) mathematically formalised as a real-time constrained optimisation problem, (ii) proven to be intractable, and (iii) solved using a novel heuristic solution, whose relative efficiency is proven through running experiments on both artificial and real sensor network, with various UAV settings. This paper is an extension of the work in [25]. An extension has been done by detailing the work and the proposed model has been transformed to make the model a real-time respondent. Furthermore, analysis results corresponding to added features have been provided and at this stage a real IoT network has been considered and compared with a random one.

The rest of this paper is organised as follows. The cooperative data muling model is presented in Section 2 and its algorithmic solution is provided in the same section. Simulated results are provided and discussed in Section 3 while the conclusion is drawn in Section 4.

2. The Cooperative Data Muling Model

In this paper, we consider an "Internet of Things in Motion" model as shown in Figure 1. We assume that UAVs are assisted by special ground-based sensors, which

locally collect data from other sensors. That is, sensors are grouped into separated clusters, each with its own sink node (the cluster head), where the information is to be collected from other sensor nodes (cluster members) and relayed to UAVs, which deliver the sensor readings to base stations. Note that only cluster heads can communicate with UAVs, and the optimal clustering scheme is not covered in this paper. Furthermore, the inner cluster communication technology is not covered here (it has been discussed in [26–28]).

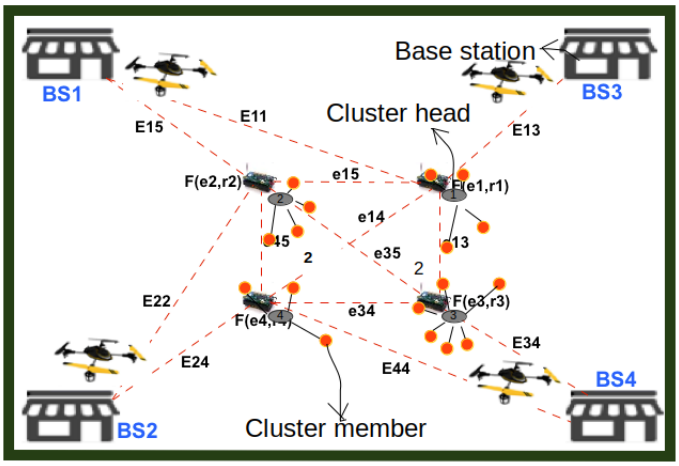


Figure 1. Cooperative data muling.

The cooperative data muling model considered in this paper is illustrated by Figure 1. The figure reveals four base stations (B1, B2, B3 and B4) from where UAVs take off to collect data from sinks located in a region of interest and later comes back for data delivery. In this illustrative scenario, all possible collection paths that can be taken by each UAV from the base stations to access data collected by cluster heads (1, 2, 3, and 4) and deliver the collected data to the closest base stations, are represented in dotted and orange lines, whereas the continuous black lines show the communication links from sensors to their cluster heads. Here, we assume that UAVs are assisted by special nodes (cluster heads/sinks) to collect information to reduce the loss due to the UAVs capabilities (fuelling, timing, etc.). Furthermore, it is assumed that the UAV paths topology is known (this means that all possible communication paths between all sensors are known) and it is represented by small black lines in the figure. The total energy required for data collection at a node/sensor, say i, is a function $F(r_i, e_i)$ of its revisit deadline r_i (the maximum amount of time required to (re)visit the node) and the energy e_i required to transfer data from node i to an arriving UAV. The travel cost from a base station B_j to a sink i is translated in an energy metric denoted by E_{ij}, and the transportation cost between two sinks i and k is also translated into an energy metric denoted by e_{ik}.

2.1. The Data Muling Problem

In this subsection, the problem is modelled as a constrained optimisation problem. We start by defining/denoting all cost-related terms and later we combine them to form a cost function.

2.1.1. UAVs Waiting Time on Sink Nodes

Let t_i be the entire time spent by an UAV to arrive at the sink i since it took off from a base station and r_i be the expected time for an UAV to arrive at the sink i. It is also referred to as revisit time at the collection point i. The sink visitation-based cost may be expressed in terms of penalties for both early and late visits on the sink nodes, the collection of information, and a risk associated with the autonomy of the UAVs. These costs are described as follows.

- Early visit penalty: an early visit penalty will be assigned to an UAV if $t_i < r_i$ to express the case where the visiting UAV arrive premature data collection. In this case, the UAV will wait for a period of time $w_i = r_i - t_i$ needed by the sink node to capture mature information from the field and transmit it to the waiting UAV;
- Late visit penalty: a late visit penalty will be applied to the UAV if $t_i > r_i$ to express the fact that the visiting UAV is late by a period of time $l_i = t_{iv} - r_i$ wasted by the UAV to arrive late to a collection point where data was ready for collection.This penalty can also be expressed using a piece-wise function;
- Data collection cost: a data collection cost will be applied to any UAV to consider the fact that the UAV has to use the energy e_i to collect information from the visited node i. Note that while the costs w_i and l_i depend on how the terrestrial and airborne sink networks have been traffic engineered, the data collection cost e_i may depend on different engineering parameters and functions, which may be bound to the communication interfaces of the equipment used by both the ground sink nodes and the UAVs and the protocol used for such communication.

For each UAV, the total cost $F(r, e_i)$ of visiting the sink i, without taking into account the travelling cost is expressed by the following equation.

$$F(r_i, e_i) = \alpha w_i u(w_i) + \beta l_i u(l_i) + \gamma e_i \tag{1}$$

where $u(w_i)$ and $u(l_i)$ are the values of a unit step function applied to w_i and wl, respectively. The coefficients α, *beta*, and *gamma* are associated with the setting-based importance/weighting allocated to the early and late arrival penalties and the data transfer penalty, respectively.

2.1.2. The Assumed Network

We consider a hybrid sensor network (a network with multiple types of links) represented by a bi-directed graph $\mathcal{G}(\mathcal{S}, \mathcal{N}, \mathcal{L}, \mathcal{B}, \mathcal{P})$ where $\mathcal{S}$ is the set of all sensor nodes, $\mathcal{N}$ is the set of all sinks, $\mathcal{L}$ is the set of wireless communication links between the sensor nodes, $\mathcal{B}$ is the set of UAV base stations, while $\mathcal{P}$ is the set links showing the possible moves of the UAVs. Here, a move expresses one of the two following kinds of connections.

- Base station-sink: these are bidirectional UAV paths connecting sinks and base stations. The cost of moving from a base station b to a sink i is denoted E_{bi} and its opposite is $E - ib$, with $E_{ib} = E_{bi}$;
- Sink-sink: these are the UAV paths connecting sinks amongst themselves and the cost to move from one sink i to j is denoted by e_{ij}, with $e_{ij} = e_{ji}$.

2.1.3. Initial Conditions

- Each UAV is assumed to start its journey from a base station;
- The waiting times at all sensor nodes. That is $l_i = w_j = 0, \forall\, i, j \in N$.

Here, it is assumed that the maximum number of UAVs at each base station is equal to the degree/capacity of the base station.

2.1.4. The Data Muling Modelling

The data muling is performed in two steps

Data collection: During data collection, an UAV is to move from base station a to collect data from $k > 0$ sinks labelled by a set of indices $p^* = [1, 2, \ldots, k]$. In this case, we represent the path used for data collection by $p = [a, 1, 2, \ldots, k]$. The energy required for this step is expressed by the following equation.

$$C(p) = E_{a1} + \sum_{i \in p^*} F(r_i, e_i) + \sum_{\substack{i,\, j \in P \\ j = i+1}} e_{ij}. \tag{2}$$

Data delivery: During data delivery, an UAV may or may not pass by some already-visited sink to deliver information to the closest base station b. However, the end point of the collection path p is the starting point of the delivery path. In this case, the corresponding energy is expressed as a function $E(p)$ of the data collection path.

Therefore, the total energy $E_T(p)$ required for data collection and delivery is given by the equation

$$E_T(p) = C(p) + E(p). \tag{3}$$

The data muling problem consists of finding for each UAV an optimal path so that the total energy spent by all the UAVs to collect and deliver the sensor readings/data without colliding is minimized. Mathematically, we represent the set of UAVs by $U = \{1, 2, 3, \ldots m\}$, where each UAV, u, departing from base station a_u will follow path p_u to collect data at locations of interest and another path (perhaps different from pu) to deliver the data to its closet base station. Let us consider 1_u, the first sink to be visited by the UAV u. The data muling problem is formulated as follows.

$$MinZ = \sum_{u=1}^{m} \left(\sum E_{a_u 1_u} + \sum_{i \in p_u^*} F(r_i, e_i) + \sum_{\substack{i,\, j \in p_u^* \\ j=i+1}} e_{ij} + E(p_u) \right). \tag{4}$$

subject to

$$\forall\, v, w \in U, p_v^* \cap p_w^* = \emptyset = d(p_v^*) \cap d(p_w^*) \tag{4a}$$

$$\bigcup_{u \in U} P_u^* = S \tag{4b}$$

$$e_i, e_{ij}, E(p), r, E_{a_u 1_u} \geq 0, \forall\, i,\ j, p,\ a_u,\ 1_u \tag{4c}$$

Here, the first constraint states that any two collection or delivery paths have no sink in common. This guarantees collision avoidance for the UAVs. On the other hand, the second constraint expresses the fact that all sinks are to be visited, and the last constraints expresses the positivity of all considered variables.

2.2. Real-Time Visitation

We consider Equation (1). In the scenarios where the variables have very strict conditions, instead of being part of the cost function, they need to be part of the problem constraints. Table 1 shows all possible models. Here, "1" shows the case where a corresponding variable is restricted (part of the constraints) and "0" shows the other way.

Table 1. Data collection scenarios.

Waiting Penalty (w_i)	Late Penalty (l_i)	Explanation
0	0	None of the two variables is bounded
0	1	Only the late penalty is bounded
1	0	Only the waiting penalty is bounded
1	1	Both variables are bounded

The optimisation problem changes its constraints so as to become the following:

$$MinZ = \sum_{u=1}^{m} \left(\sum E_{a_u 1_u} + \sum_{i \in p_u^*} F(r_i, e_i) + \sum_{\substack{i,\, j \in p_u^* \\ j=i+1}} e_{ij} + E(p_u) \right). \tag{5}$$

subject to

$$\forall v, w \in U, p_v^* \cap p_w^* = \emptyset = d(p_v^*) \cap d(p_w^*) \tag{5a}$$

$$\bigcup_{u \in U} P_u^* = S \tag{5b}$$

$$0 \leq w_i \leq W_i \tag{5c}$$

$$0 \leq l_i \leq L_i \tag{5d}$$

$$e_i, e_{ij}, E(p), r, E_{a_u 1_u} \geq 0, \textit{for all } i,\ j, p,\ a_u,\ 1_u \tag{5e}$$

where W_i and L_i are the predetermined thresholds, which may take any non negative value.

2.3. Related Problems and Solutions

The data muling problem considered in this paper is closely related to the file recovery problem in [29] (NP-hard problem) solved by curving techniques, including those using the Parallel Unique Path (PUP) algorithm. This problem considers a case of many fragmented files, which need to be reassembled, starting from their headers, which are assumed to be known initially. The PUP algorithm is a variation of Dijkstra's routing algorithm [30], where, starting from the headers, clusters are successively added based on their best matches.

This is done with the aim of building paths from headers having a cluster added to an existing path if and only if the link to it has the least weight. On the other hand the Vehicle Routing Problem (VRP) [31,32] consists of finding the optimal road from a depot, to be taken for delivering resources to customers and return to the depot. Exact and heuristic algorithms for its solution have been surveyed in [31]. In the survey, all stated algorithms assume a single distance matrix (the cost matrix) and hence could fail to be a good fit for our persistent visitation scenario since in our case the weighting of nodes matters and it is not a fixed value.

Furthermore, for the VRP, vehicles end their trips at the depots where they started from. This would limit the number of topologies where the data muling problem is solvable and could also impose a data muling scheme, which is not necessarily optimal. Note that in our case, we are interested in the case where the late and stale visitation are taken care of, and this depends on the dynamic position of UAVs (see the Equation (1)). Furthermore, UAVs deliver the collected information to optimal base stations (which are not necessarily their starting points).

2.4. The Data Muling Problem Intractability

To prove its intractability, we provide a polynomial reduction of one-depot VRP (which is known to be an NP-hard problem), into a special case of the data muling problem: the case where each sink's weight is zero. The transformation consists of a two-step process, which transforms the graph $\mathcal{G}$ as follows.

a. Group all base stations in in one cluster/group and consider this cluster/group as a special node for the graph, this gives the VRP's topology $\mathcal{G}'$ $(\mathcal{S},\ \mathcal{N},\ \mathcal{L},\ \mathcal{B}'\ , \mathcal{P})$, where $\#\ \mathcal{B}' = 1$;
b. For every link of $\mathcal{G}'$, make the link weight in the new graph (found in a.) the inverse of the weight in graph $\mathcal{G}'$.

This will reduce the VRP's into the data muling problem's solution. Clearly, the time complexity of the transformation process is polynomial, since Step a has complexity (#) and Step b has complexity (#R). The time complexity for the whole graph transformation/reduction process is therefore (#R) + (#), which is polynomial. This shows that the problem of interest in this paper is NP-hard and hence a heuristic solution is important.

2.5. The Data Muling Algorithm

In this work, we adapt Dijkstra's algorithm in the same way it is done in [29] to solve the data muling problem. While many rounds are considered by our algorithm, we consider only the case where each node is visited only once per round. It is assumed that each UAV is capable of collecting and delivering data to a base station, where it can be recharged before going for another data collection round.

Algorithm 1 has two major steps: the first step consists of using Equation (2) to select the best node to visit for every UAV (Steps 6–7); the second step consists of adjusting the UAV's paths by choosing the cheapest UAV for every best node (Steps 8–23). Once the visitation is done, the collection paths are captured in Cpaths and the two steps are repeated to select the nearest base station for every UAV data delivery following the delivery paths recorded by Dpaths.

Algorithm 1: Cooperative algorithm.

```
1  Assume a network G(N,L,B,P) as specified in Section 2.1.2 ;
2  Choices ← all sink to be visited ;
3  Initialise the path to the initial hosting base stations;
4  done ← choices ∪ B;
5  while choices ≠ ∅ do
6      for u ∈ U do
7          Select the next destination of least cost, using Equation (2) ;
8      end
9      Assign = ∅;
10     for u ∈ U do
11         Let c be the choice of u;
12         for v ∈ U \ Assign do
13             if u and v selected the same choice c then
14                 Include c in the path of a UAV of least cost (2);
15                 Include the UAV in Assign;
16                 Include the c in done;
17                 Choices ← Choices \ {c};
18                 Break;
19             end
20         end
21         if c ∈ Choices then
22             Include c in the path of u;
23             Include the c in done;
24             Choices ← Choices \ {c};
25             Include u in Assign;
26             Break;
27         end
28     end
29     if Choices = ∅ or #done = #U then
30         Choices ← B;
31         done ← ∅;
32         B ← ∅;
33         if #done ≠ #U then
34             Cpaths ← all UAVs' paths;
35         end
36 end
37 Dpaths ← all current UAVs' paths ;
38 Return Cpaths and Dpaths
```

Proposition 1 (Polynomial termination). *Algorithm 1 terminates in polynomial time when all sink nodes have been visited.*

Proof. Note that each time a sink is included in a path of one of the UAVs, it gets excluded from the list choice (Lines 16 and 21). Since all UAV paths consist of a connected graph, whenever $choice = \emptyset$, there is at least one UAV that makes a new selection of a next sink to visit on Line 6. So, all sink are visited.

Once $choice = \emptyset$, the next destinations become the base stations and the set done = (Lines 25 and 26, consecutively). The next step is to make $\#done = \#U$ true by assigning to every UAV a base station. In this case, the statement at Line 24 is true and the set $choice = B$, which it had been updated to . This makes Algorithm 1 stop. On the other hand, the time complexity of the algorithm is clearly $\mathcal{O}((\#U)^2)$, which is a polynomial. Hence, the result follows. □

Remark 1. *Persistent visitation model. Since each UAV's computed path is ended by a base station and the UAV paths form a static network, Algorithm 1 can be used repetitively to make a persistent visitation.*

2.6. Illustration of the Algorithm

Algorithm 1 uses an example in Figure 2. We run one round of the algorithm step by step. In this example, we consider a case were each sensor node weighting is constantly zero. That is, $\alpha = \beta = 0$ (see the constants in Equation (1)).

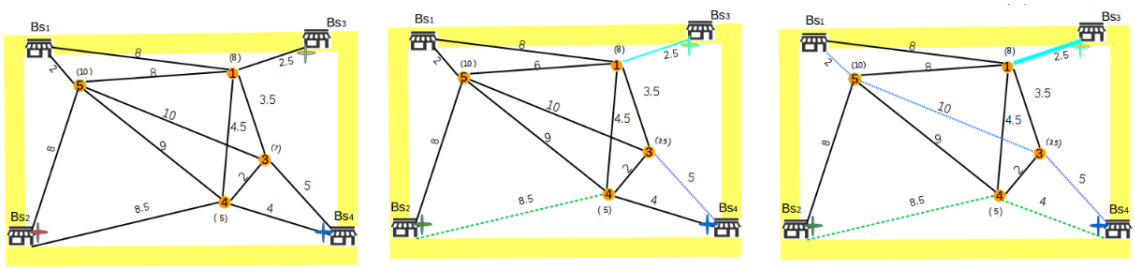


Figure 2. Illustrative example. (**a**) Initial step; (**b**) The best neighbour choice; (**c**) Illustrative example.

Figure 2a shows the initial conditions of the considered system. The system has four sensors, four base stations, and three UAVs positioned at all of them except for Base station Bs1. The links and sensors are weighted as discussed in Section 2. Let u2, u3, and u4 be the name of the UAVs staying at Base Stations Bs2, Bs3, and Bs4, respectively.

Figure 2b reveals how the UAVs make the choices for their first moves. The best choice is the one corresponding to the cheapest move, which is evaluated using the weight of the road to be used together with the delay at the sensor to be visited. This is why UAVs u2, u3, and u4 move to Sensors 4, 1, and 3, respectively. At this stage, only node 5 is the only node not yet visited.

Figure 2c shows the next moves up to the end of the algorithm. It shows that UAV u4 moves to Sensor 5, because it is the one corresponding to the cheapest move. On the other hand, all other UAVs do not have any other choice of sensor to visit. They then need to visit their closest base station. Once UAV u4 arrives at Node 5, it visits the closest base station, which is Bs1.

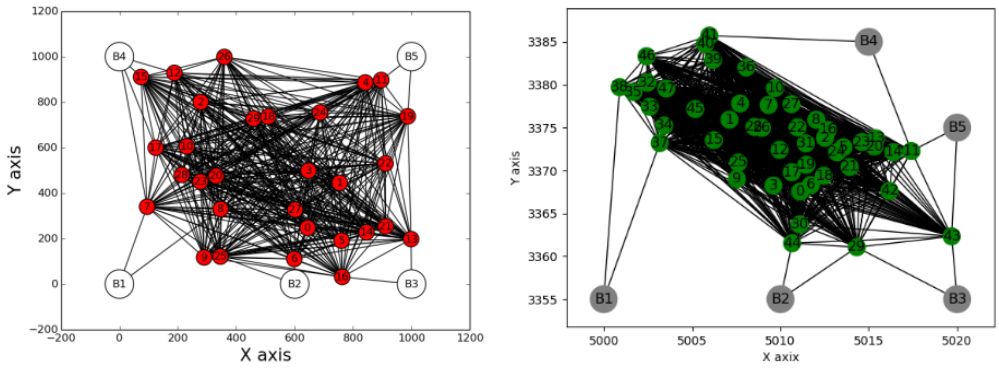
2.7. Limitation of the Proposed Model

UAVs are assumed to always have enough resources to execute the assigned tasks. It is assumed that, before departing, UAVs are fully charged (or have enough fuel) and can pass through any path which may be assigned to it. On the other hand, ground sensors are assumed to always work in perfect conditions. Furthermore, several communication models may be used to deliver data from nodes to sinks (see [26,33] for example). The task

assignment model in this work has not included the communication (from nodes to sinks) cost and constrains. This has been achieved by assuming that data to be delivered at every sink would eventually be available.

3. Experimental Results

Python was used to run Algorithm 1 on two more complex networks (Figure 3a,b). The performance of the algorithm is studied and the behaviours of considered parameters are investigated. The first network (Figure 3a) consists of five base stations: B1, B2, B3, B4, and B5, as shown by bigger nodes in Figure 3a. In the figure, smaller nodes represent the sink to be visited and the links in the network show the possible paths the UAVs may take to visit the targets. Note here that the sink's network (network without base stations) is a complete graph (each UAV is able to move from one sink to any other one in the network, but not to any base station) where nodes are randomly deployed on a 1 km^2 area. On the other hand, we consider a real network (Figure 3b) consisting of the Cape Town complete network, whose nodes are police stations and their Cartesian coordinates have been extracted from GPS positions. The names corresponding to each node label are described in Appendix A (see Figure A1a,b). Note here that in these experiments the considered UAV are drones.



(**a**) (**b**)

Figure 3. The considered networks. (**a**) Artificial network; (**b**) Real Cape Town network.

As shown in Figure 3a, the considered network consists of nodes randomly placed in an area of size 1 km^2 and sinks are labelled in terms of the energy required to collect information from them. The coordinates of nodes in both networks are in metres and could be seen or approximated using Figure 3a,b. Positions (of sinks or base stations) consist of triples but for simplicity of plotting/viewing them, they have been projected on X-Y coordinates and hence, they are presented in the 2D Cartesian coordinate system.

3.1. Impact of Speed Distribution on Path Planning

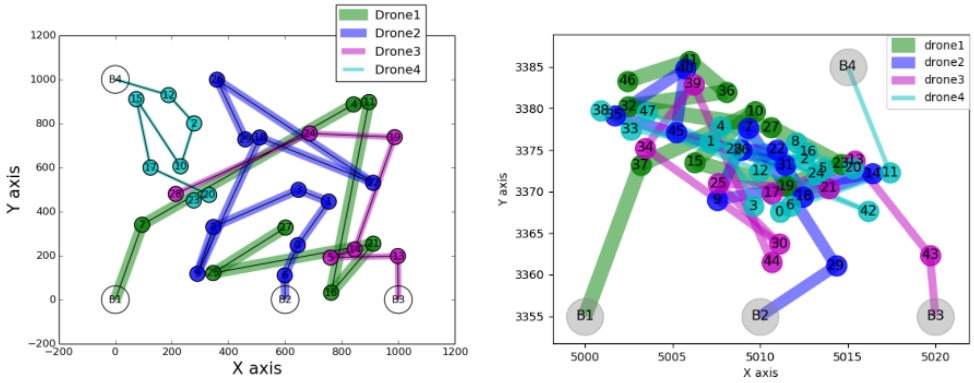
In two steps, we present the paths taken by UAVs using Algorithm 1.

Step 1. Data collection: it consists of visiting all sinks using the first three steps of Algorithm 1. The corresponding path for each UAV is shown in Figure 4a,b;

Step 2. Data delivery: it consists of visiting base stations using the last step of the same algorithm. The results are shown in tables where the speed distribution of the UAVs is also presented.

The assumed cost function parameters are set to $\alpha = \beta = 0.5$ and $\gamma = 1$. Figure 4a,b reveals that UAVs do not visit the same number of sinks in the case of both considered networks. For example in Figure 4a, Drone1, Drone2, Drone3, and Drone4 visit sinks 7, 10, 6, and 7, respectively. Table 2 shows that when the speeds are the same for all UAVs, the delivery requires some UAVs to pass by some of the visited nodes to arrive at the base stations. This is because each sink does not need to be connected at a base station. Table 2a shows that in the case of random network, Drone3 and Drone4 deliver the collected data

at the same base station (B1), and for the real network, Table 2b reveals that Drone1 and Drone4 deliver the data to B4.



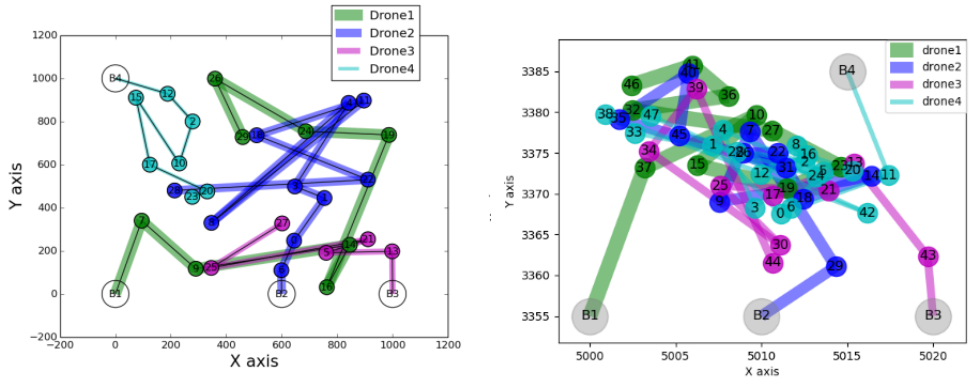
(**a**) (**b**)

Figure 4. Path generation when the UAVs have the same speed. (**a**) Path generation on a random network; (**b**) Path generation on a real network.

Table 2. Delivery when all speeds are the same.

(a) Delivery in the Random Network			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	500	B1	[27, 6, B2]
Drone2	500	B2	[29, 12, B4]
Drone3	500	B3	[28, 7, B1]
Drone4	500	B4	[23, 8, B1]
(b) Delivery in the Real Network			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	500	B1	[46, 41, B4]
Drone2	500	B2	[45, 11, B5]
Drone3	500	B3	[44, B2]
Drone4	500	B4	[47, 12, B4]

Figure 5 shows that when speeds are different, some UAVs may not change their paths but other UAVs will change their path. For example Drone4 does not change its path in both Figure 5a,b, whereas all other drones do. On the other hand, Figure 5b shows that for the real network, all drones keep their paths.



(**a**) (**b**)

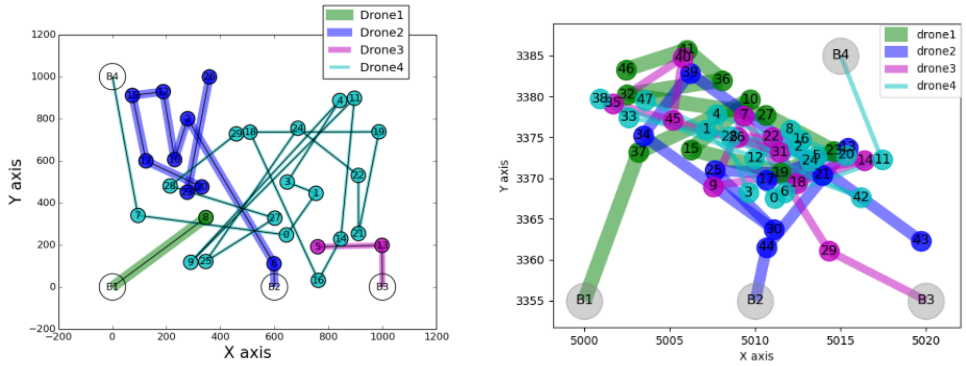
Figure 5. Paths generation when the UAVs have the different speeds. (**a**) Path generation on a random network; (**b**) Path generation on a real network.

Table 3 shows that when the speeds are different, the delivery also requires some UAVs to pass by some of the already visited nodes in order to arrive at the closest base stations. Table 3a shows that Drone2 and Drone4 deliver the collected data at the same base station (B1), and Table 3b shows that Drone1 and Drone4 deliver the collected data at the same base station (B4).

Table 3. Data delivery when all drones have a different speed.

(a) Delivery in the Random Network			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	800	B1	[29, 12, B4]
Drone2	700	B2	[28, 7, B1]
Drone3	600	B3	[27, 6, B2]
Drone4	500	B4	[23, 8, B1]
(b) Delivery in the Real Network			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	800	B1	[46, 41, B4]
Drone2	700	B2	[45, 11, B5]
Drone3	600	B3	[44, B2]
Drone4	500	B4	[47, 12, B4]

It is shown in Figure 6 that the distribution of the UAVs' speeds have an impact on paths generation. For example Figure 6a shows that Drone1, Drone2,



(**a**) (**b**)

Figure 6. Paths generation when speeds distribution is changed. (**a**) Path generation on a random network; (**b**) Path generation on a real network.

Drone3, and Drone4 visit sinks 1, 9, 2, and 18, respectively. This shows a big difference due to the fact that the choice of target depends on the current and not the previous visitation costs, together with the change of speed distribution in the base stations.

Table 4 shows that when the speeds are differently distributed, the paths are changed and thus the delivery paths also change. Table 4a shows that for the random network, Drone2 and Drone4 deliver the collected data at the same base station (B4); and for the real network, Table 4b shows that Drone1 and Drone4 deliver the collected data at the same base station (B4). Since the path generation for both networks essentially behave the same, we consider (the artificial) random network for the next analysis.

Table 4. Data delivery when speed distribution changes.

(a) Delivery in the Random Network			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	500	B1	[8, 6, B2]
Drone2	600	B2	[26, 12, B4]
Drone3	700	B3	[5, 16, B3]
Drone4	800	B4	[29, 15, B4]
(b) Delivery in the Real Network			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	500	B1	[46, 41, B4]
Drone2	600	B2	[43, B3]
Drone3	700	B3	[45, 11, B5]
Drone4	800	B4	[47, 12, B4]

3.2. Impact of the Speed Distribution on the Cost (Total Energy)

We now study the impact of speed distribution on the cost, considering many runs of the algorithm. We perform 20 runs of Algorithm 1 in 3 cases of speed distribution, as shown by the second columns, in the tables. Figure 7a shows a case where each UAV's speed equals 500 m/min, and on the other side Figure 7b corresponds to the case where UAVs have different speeds.

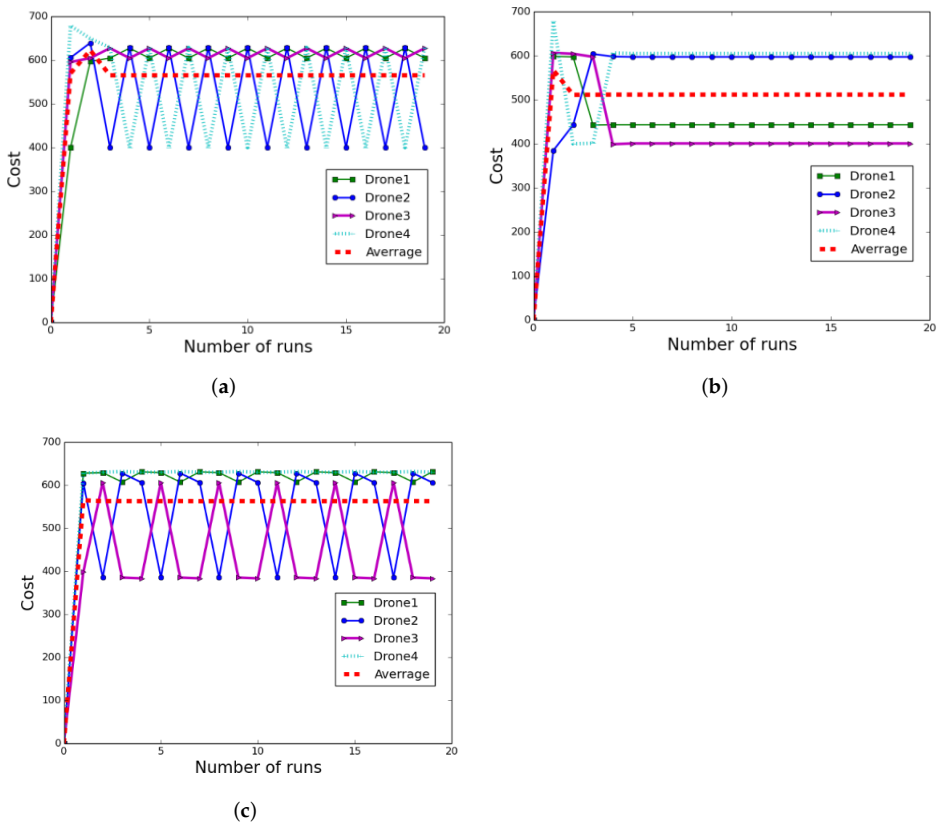


Figure 7. Impact of speed distribution. (**a**) Cost at the same speed; (**b**) cost at the same speed; (**c**) cost related to a different speed distribution.

Figure 7a reveals that the cost for each drone lies in one of the few fixed values. Drone4 takes four values and all others take three and succeed each other to take the minimum and maximum values. The average cost is constantly close to 560.

Figure 7b shows that after the first four runs, all UAVs correspond to constant costs where the average cost is constantly close to 500.

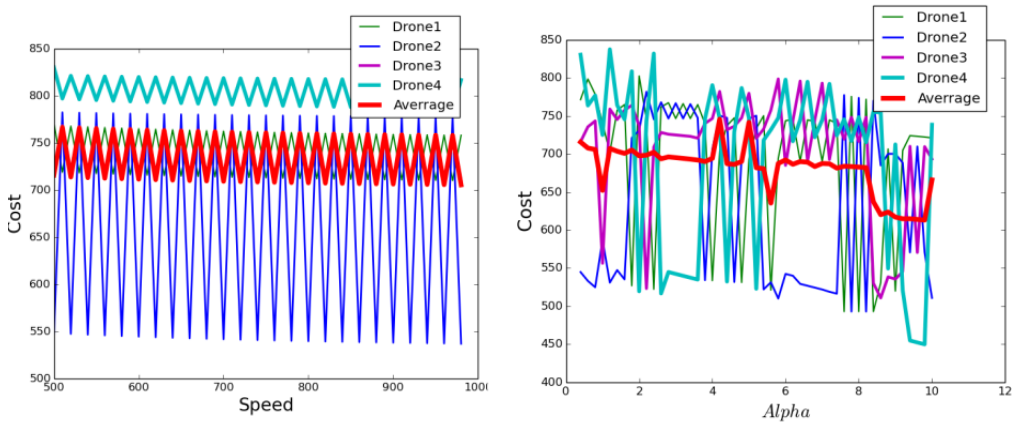
Figure 7c shows that the change in speed distribution may change the average cost and also the trends of the cost function. In this case, the amended speed distribution corresponds to the one in Table 4, and shows that for each UAV, the cost changes periodically and can only take one of a few fixed values. This is why the average cost also takes one of the fixed values on a periodic basis.

3.3. Effect of Parameters Variation

In this subsection, we study the effect of four parameters on the cost variation as the number of runs varies. The four considered parameters are described as follows.

- Speed value.Keeping the speed the same for all UAVs, we aim to study the impact of its increase on the coverage cost;
- Overdue time (α). We study the impact of overdue time on the cost. Great attention is placed on this time by incrementing the corresponding coefficient (penalty) by 0.05, for each new run;
- Delay (β). While all the other parameters are constant, we vary the parameter corresponding to the delay penalty and study how it changes the value of the coverage cost;
- Data collection rate (γ). Data collection rate is incremented by 0.05 for its value $\gamma = 1$, of 20 runs of the algorithm.

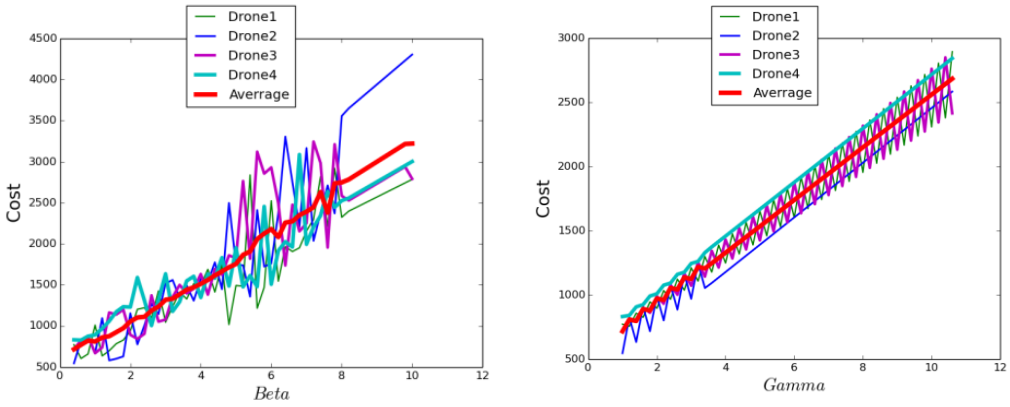
Figure 8a shows a case where the speed has been incremented 20 times for all UAVs. It shows that from the first run, each UAV periodically changes its cost between two cost values. Drone4 corresponds to the highest cost while notably Drone3 corresponds exactly to the average cost. Figure 8b shows a case where the overdue (α) is incremented by 0.5, at each of 20 consecutive runs. The values of cost corresponding to each UAV is stochastic and the average is stochastically decreasing. We consider a case where the latency is the factor that changes in Figure 9a. The figure shows stochastic values as well but however the average cost is mostly increasing.



(**a**) (**b**)

Figure 8. Impact of Speed and overdue time on the total cost (**a**) Variation of the cost with respect to the speed; (**b**) Variation of the cost with respect to the overdue time (α).

Figure 9b shows the impact of γ on the variation of the cost over 20 runs. It shows that Drone4 mostly corresponds to the highest cost, and once $\gamma > 3.8$, Drone4 and Drone2 are constantly increasing their corresponding costs, whereas the others are periodically increasing and decreasing.



(**a**) (**b**)

Figure 9. Impact of the delay and data collection rate on the total cost. (**a**) Variation of the cost with respect to the delay (β); (**b**) variation of the cost with respect to the data collection rate.

3.4. Prioritisation Analysis

In this subsection, we discuss the effect of the delay and overdue time boundaries. We assume the same network as shown by Figure 3a, where the UAVs Drone1, Drone2, and Drone3 are initially positioned at Base Stations B1, B2, and B3, respectively; and all the UAVs are assumed to have the same speed v = 800 m/min.

3.4.1. Effect of Delay Constraints on Path Design

Figure 10 and Table 5 represent the case where each sensor node is visited if the arrival of a drone is late for no more than 30 min. Figure 10 shows that 14 sensor node could not be visited (see the red hexagon shaped nodes). The UAV Drone3 visited most of the nodes, whereas other UAVs could visit only three sensors each. Table 5 shows that Drone1 delivers to Base Station B4 via node 12, Drone2 to Base Station B3 via node 14 and then 13, and Drone3 to Base station B3 via node 16.

Table 6 shows the data delivery paths when UAVs may be late by no more than 60 min, and Figure 11 shows the data collection path with this setting. When changing the late threshold to 60 min, Table 6 shows that the delivery paths changed, and the delivery is done as the last column of the table shows.

Compared to Figure 10, Figure 11 reveals that Drone3 does not change the path, but the other UAVs extend their path to two more sensor nodes each. This results in 10 sensor nodes to be missed, as shown by the figure.

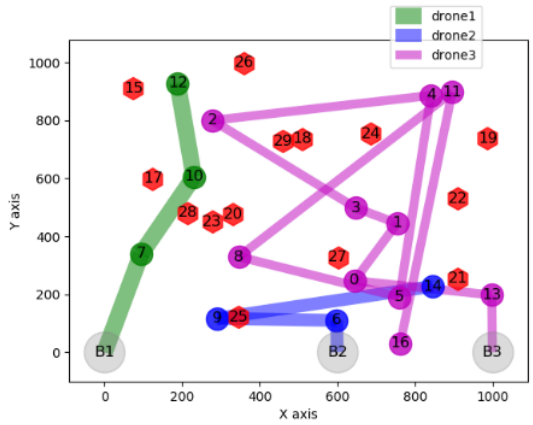


Figure 10. Path details (waiting = 30).

Table 5. Data delivery when UAVs may be late for no more than 30 min.

UAV	Speed (m/min)	Source	Returning Path
Drone1	800	B1	[12, B4]
Drone2	800	B2	[14, 13, B3]
Drone3	800	B3	[16, B3]

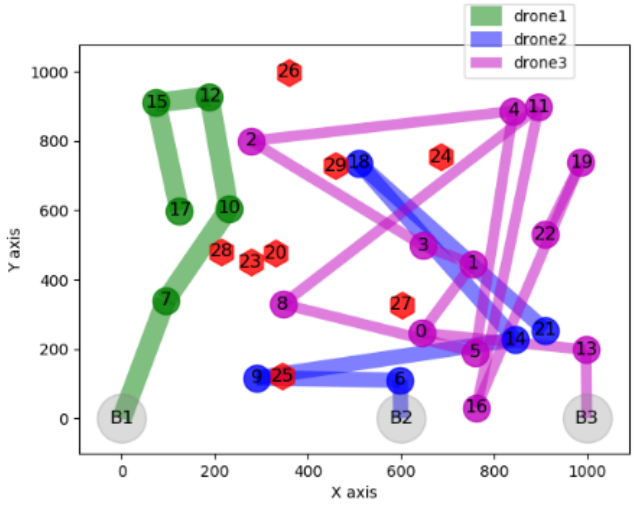


Figure 11. Path details (waiting = 60).

Table 6. Data delivery when UAVs may be late for no more than 60 min.

UAV	Speed (m/min)	Source	Returning Path
Drone1	800	B1	[17, B4]
Drone2	800	B2	[21, 13, B3]
Drone3	800	B3	[22, 19, B5]

Table 7 and Figure 12, respectively, show the data delivery and collection paths when UAVs may be late for a very long time.

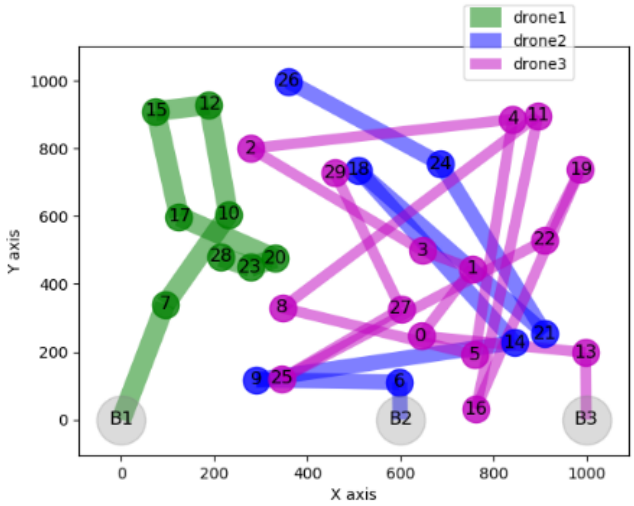


Figure 12. Path details (waiting = ∞).

Table 7. Data delivery when UAVs may be late for more than 60 min.

UAV	Speed (m/min)	Source	Returning Path
Drone1	800	B1	[28, 7, B1]
Drone2	800	B2	[26, 12, B4]
Drone3	800	B3	[29, 15, B4]

Compared with the previous two cases, Table 7 shows that the delivery paths keep on changing. Figure 12 shows that all nodes have been visited and the path of each drone has been extended to cover more nodes.

Figure 13 reveals the changes in the number of missed nodes while the late threshold evolves. The figure shows that when the delay threshold increases, the number of unvisited nodes remains constant or decreases (it never increases), until it converges to zero. This is justified by the fact that allowing a longer delay increases the chance for a node to be visited.

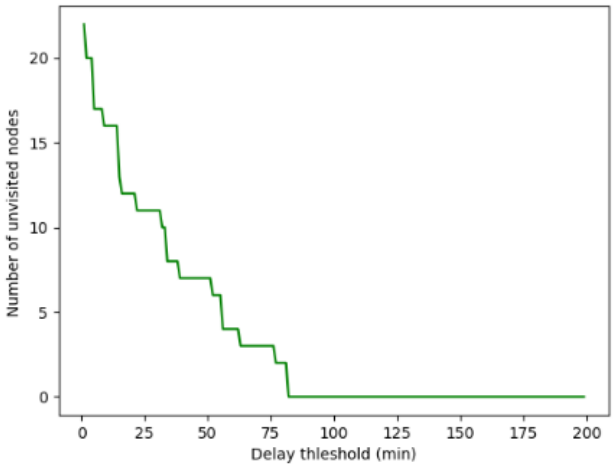


Figure 13. Variation of the number of unvisited nodes with respect to the delay threshold.

3.4.2. Effect of Overdue Constraints on Path Design

In this section, we study the effect of the waiting constraints. Here, sensors can only be visited after some fixed time, called the waiting threshold.

Figure 14 shows how paths corresponding to different thresholds are generated. Figure 14a considers a case where there is no waiting limitation (the waiting time is zero) and clearly all sensors are visited. When the waiting time is set to 0.3 min (18 s) in Figure 14b, three nodes are not visited and Drone 3 could not visit any sensor. Setting the waiting threshold to 0.5 min, only UAV Drone 1 could visit, and five sensors were missed (see Figure 14c). Figure 14d shows that when setting the waiting time to 0.7 min, no node could be visited. This is because there is a specific time for UAVs to arrive at each sensor, which depends on the distance and speed. If the distance is small and the UAV can arrive earlier than the waiting threshold, the visitation is impossible.

Tables 8 and 9, show deliveries corresponding to visitations shown in Figure 14. Note that if an UAV does not visit any sensor, it remains at its initial position. By changing the waiting threshold, Figure 15 shows the corresponding number of missed sensors.

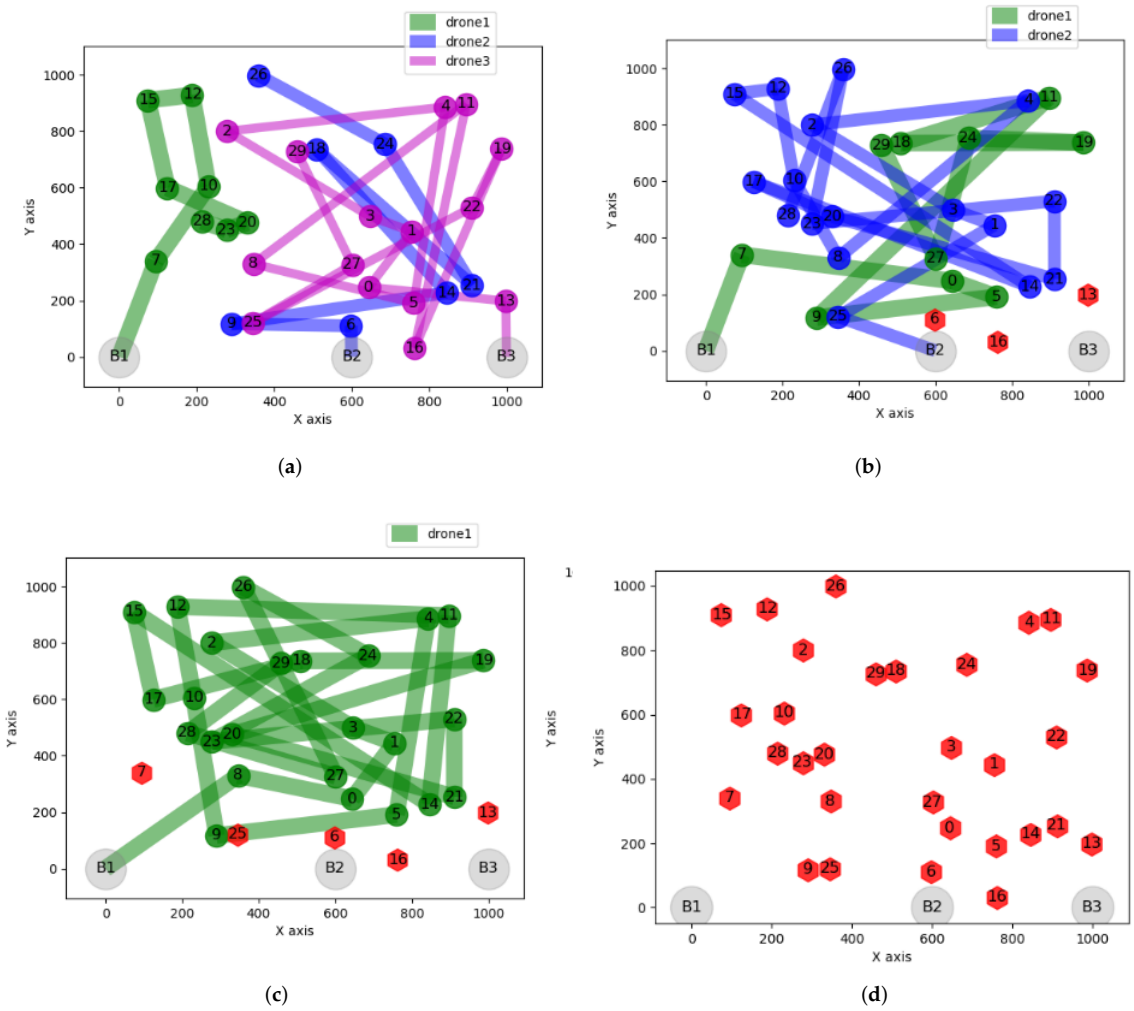


Figure 14. Visitation constrained by the waiting time. (**a**) Waiting for 0.0 min; (**b**) Waiting for 0.3 min; (**c**) Waiting for 0.5 min; (**d**) Waiting for 0.7 min.

Table 8. Data delivery when the waiting threshold is small.

(a) Data Delivery When the Waiting Threshold Is 0 min			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	800	B1	[28, 7, B1]
Drone2	800	B2	[26, 12, B4]
Drone3	800	B3	[29, 15, B4]
(b) Data Delivery When the Waiting Threshold Is 0.3 min			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	800	B1	[29, 12, B4]
Drone2	800	B2	[28, 7, B1]
Drone3	800	B3	[B3]

Figure 15 reveals that the number of missed nodes increases as the waiting threshold increases and converges to the total number of sensors to be visited.

Table 9. Data delivery when the waiting threshold is high.

(a) Data Delivery When the Waiting Threshold Is 0.5 min			
UAV	**Speed (m/min)**	**Source**	**Returning Path**
Drone1	800	B1	[28, 7, B1]
Drone2	800	B2	[26, 12, B4]
Drone3	800	B3	[29, 15, B4]
(b) Data Delivery When the Waiting Threshold Is 0.7 min			
UAV	**Speed (m/min)**	**Source**	**Returning path**
Drone1	800	B1	[29, 12, B4]
Drone2	800	B2	[28, 7, B1]
Drone3	800	B3	[B3]

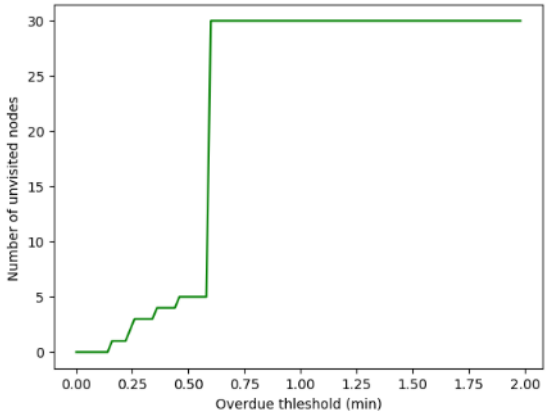
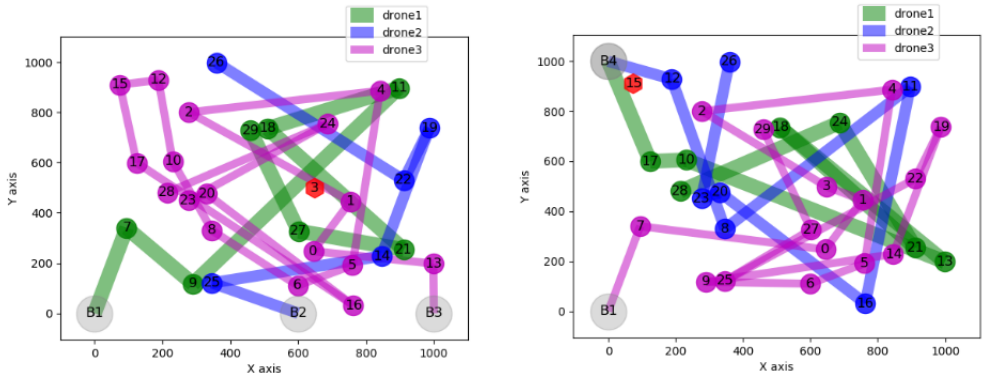


Figure 15. Variation of the number of unvisited nodes with respect to the overdue threshold.

3.4.3. Persistent Visitation Analysis

In this subsection, we study the trend of paths when UAVs persistently visit sensors. Persistent visitation is done by resetting the initial base stations for the next visit, to the destination of the previous visitation.

Figures 16 and 17 show four consecutive visitations and deliveries when the waiting threshold is set to 12 s. Figure 16a shows that only node 3 has not been visited. Figure 16b shows that a new node (node 15) has not been visited and all visitation paths change.

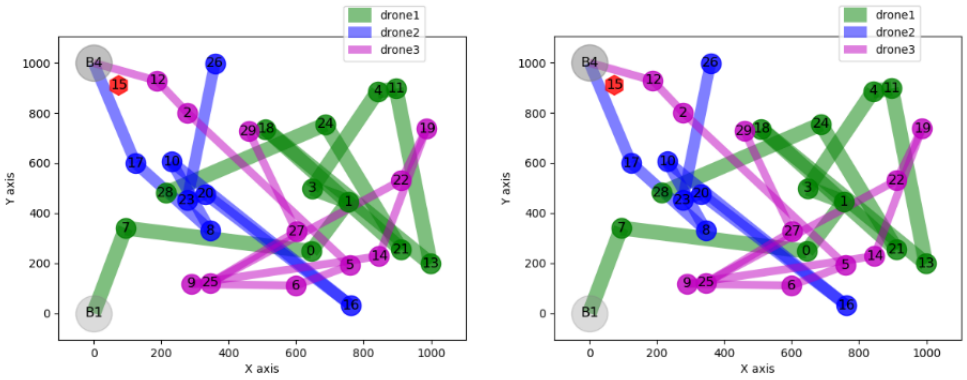


(**a**) (**b**)

Figure 16. First two visitations. (**a**) First visitation; (**b**) Second visitation.

Figure 17a shows that paths keep on changing and non-visited nodes remain the same as the previous visitation. Note that Figure 17b is exactly the same as Figure 17a. This shows that all the following paths will be the same as Figure 17a and hence paths

generation may converge to specific paths. This is a special case where the UAVs' initial positions become the same as their optimal destination.

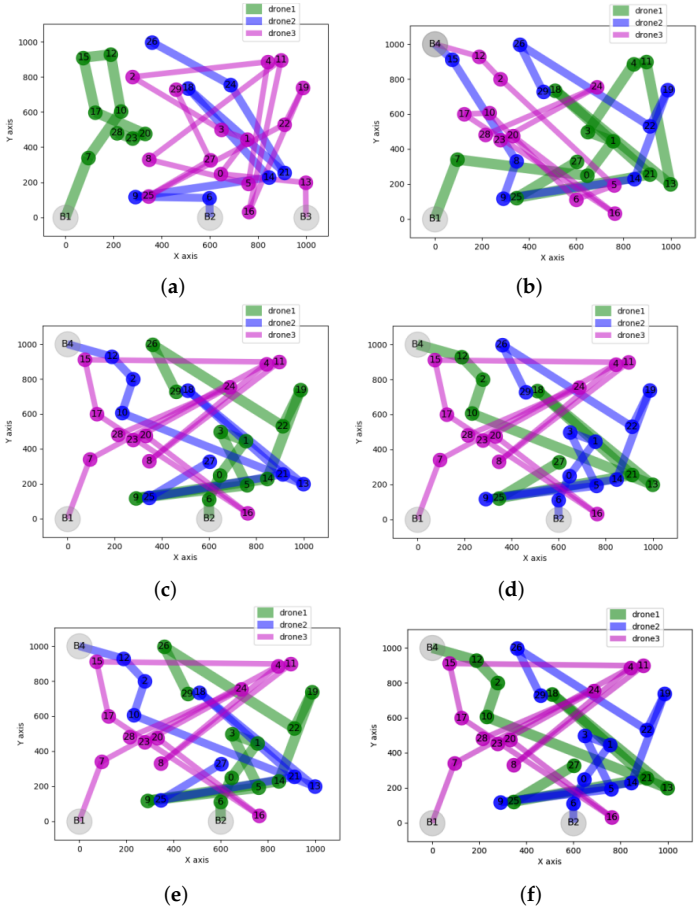


(**a**) (**b**)

Figure 17. Next two visitations. (**a**) Third visitation; (**b**) Fourth visitation.

We now consider the constraint free visitations and study the pattern of generated paths.

Figures 18 shows the first six consecutive unconstrained visitations. The path changes, and from the third visitation the path generation becomes periodic: the third visitation is the same as the fifth, and the fourth visitation is the same as the sixth. This shows that after each visitation, the path generation remains the same. This is justified by the fact that from each base station, each UAV has a single optimal destination.



(**a**) (**b**)

(**c**) (**d**)

(**e**) (**f**)

Figure 18. (**a**) First visitation; (**b**) second visitation; (**c**) third visitation; (**d**) fourth visitation; (**e**) fifth visitation; (**f**) sixth visitation.

4. Conclusions and Future Work

In this paper, a model for data muling and targeting revisit costs minimization for a team of UAVs has been provided. A mathematical formulation of the model has been presented and the underlying problem has been polynomially reduced to another NP-hard problem, and hence proved to be an NP-hard problem. A heuristic solution to the problem has been provided and its performance was evaluated through simulation experiments. Simulation results have revealed different path distribution patterns under different experimental settings and the impact of these settings on path length fairness and related energy cost. Furthermore, simulations show that consecutive path generation becomes periodic after some number of visitations. Calibrating constraint parameters has shown to lead to issues of not using some UAVs, and it could reach a point where it becomes impossible to visit some or all of the nodes.

While this paper has presented the basis of a data muling model aiming at supplementing traditional traffic engineering techniques used in sink networks, several network and traffic aspects related to the proposed data muling model still need to be investigated. These include the design of efficient communication models that consider the outdoor characteristics of drone-to-sink communication, as suggested in [34]. Taking advantage of the emerging white space frequency bands as discussed in [35] to achieve drone-to-sink and drone-to-drone communication is another direction for further work. The management of ground sensors has been done in [36–38], but these models could be more efficient when UAVs are introduced. This will enhance many modern cyber security models such as the one proposed in [39].

Author Contributions: Conceptualization, E.T. and S.R.; methodology, E.T., S.R. and N.B.; software, E.T.; validation, E.T., A.B., S.R. and N.B.; formal analysis, E.T., S.R. and N.B.; resources, E.T. and A.B.; data curation, E.T.; writing—original draft preparation, E.T.; writing—review and editing, S.R.; visualization, E.T.; supervision, A.B. and N.B.; project administration, A.B. and N.B.; funding acquisition, A.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable

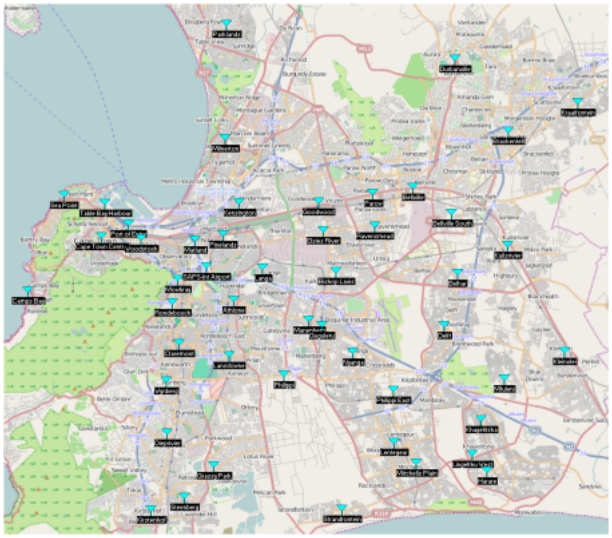
Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

We assume the public safety network consisting of Cape Town (South Africa) police stations as collection points of a ground sensor network used, for example for city safety or traffic control. Cape Town police stations are labelled in terms of integers in interval [1, 49] and their GPS coordinates are used as their positions (see Figure A1a). The corresponding positions on a map are shown in Figure A1b.

Label	Station	Longitude	Latitude
1.	Bellville	33°54'06"S	018°37'12"E
2.	Nyanga	33°59'20"S	018°34'54"E
3.	SAPS-Int Airport	33°56'37"S	018°29'18"E
4.	Bellville South	33°54'55"S	018°38'40"E
5.	Philippi	34°00'02"S	018°32'16"E
6.	Matland	33°55'46"S	018°28'52"E
7.	Parow	33°54'16"S	018°35'39"E
8.	Lansdowne	33°59'26"S	018°30'10"E
9.	Rondebosch	33°57'46"S	018°27'59"E
10.	Kuilsrivier	33°55'56"S	018°40'50"E
11.	Wynberg	34°00'15"S	018°27'46"E
12.	Sea Point	33°54'20"S	018°23'51"E
13.	Bishop Lavis	33°56'46"S	018°34'15"E
14.	Cape Town Central	33°55'40"S	018°25'16"E
15.	Table Bay Harbour	33°54'36"S	018°25'24"E
16.	Delft	33°58'29"S	018°38'24"E
17.	Mowbray	33°57'00"S	018°28'12"E
18.	Ravensmead	33°55'19"S	018°35'45"E
19.	Goodwood	33°54'33"S	018°33'35"E
20.	Elsies River	33°55'28"S	018°33'46"E
21.	Port of Entry	33°55'15"S	018°26'18"E
22.	Kensington	33°54'34"S	018°30'33"E
23.	Athlone	33°57'41"S	018°30'20"E
24.	Woodstock	33°55'43"S	018°26'48"E
25.	Pinelands	33°55'36"S	018°29'56"E
26.	Belhar	33°56'49"S	018°38'55"E
27.	Manenberg	33°58'18"S	018°33'13"E
28.	Claremont	33°59'04"S	018°28'15"E
29.	Guguletu	33°58'29"S	018°33'43"E
30.	Durbanville	33°50'01"S	018°38'49"E
31.	Brackenfell	33°52'18"S	018°40'51"E
32.	Langa	33°56'39"S	018°31'27"E
33.	Mitchells Plain	34°02'51"S	018°37'19"E
34.	Khayelitsha	34°01'29"S	018°39'48"E
35.	Mfuleni	34°00'11"S	018°40'43"E
36.	Lingelthu West	34°02'35"S	018°39'28"E
37.	Dieprivier	34°01'54"S	018°27'47"E
38.	Kleinvlei	33°59'18"S	018°43'00"E
39.	Harare	34°03'06"S	018°40'01"E
40.	Grassy Park	34°02'55"S	018°29'35"E
41.	Steenberg	34°03'56"S	018°28'28"E
42.	Kirstenhof	34°04'19"S	018°27'11"E
43.	Camps Bay	33°57'22"S	018°22'29"E
44.	Milnerton	33°52'32"S	018°30'01"E
45.	Parklands	33°48'55"S	018°30'04"E
46.	Kraaifontein	33°51'24"S	018°43'30"E
47.	Philippi East	34°00'32"S	018°36'27"E
48.	Strandfontein	34°04'19"S	018°34'29"E
49.	Lentegeur	34°02'11"S	018°36'29"E

(**a**)



(**b**)

Figure A1. Raw real network study. (**a**) GPS positions of Cape Town police stations; (**b**) positions on the globe's map.

References

1. Bagula, A.; Castelli, L.; Zennaro, M. On the design of smart parking networks in the smart cities: An optimal sensor placement model. *Sensors* **2015**, *15*, 15443–15467. [CrossRef] [PubMed]
2. Bagula, A.; Zennaro, M.; Inggs, G.; Scott, S.; Gascon, D. Ubiquitous sensor networking for development (usn4d): An application to pollution monitoring. *Sensors* **2012**, *12*, 391–414. [CrossRef] [PubMed]
3. Masinde, M.; Bagula, A. A framework for predicting droughts in developing countries using sensor networks and mobile phones. In Proceedings of the 2010 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists, Bela Bela, South Africa, 11–13 October 2010; pp. 390–393.
4. Masinde, M.; Bagula, A.; Muthama, N.J. The role of ICTs in downscaling and up-scaling integrated weather forecasts for farmers in sub-Saharan Africa. In Proceedings of the Fifth International Conference on Information and Communication Technologies and Development, Atlanta, GA, USA, 12–15 March 2012; pp. 122–129.
5. Mandava, M.; Lubamba, C.; Ismail, A.; Bagula, A.; Bagula, H. Cyber-healthcare for public healthcare in the developing world. In Proceedings of the 2016 IEEE Symposium on Computers and Communication (ISCC), Messina, Italy, 27–30 June 2016; pp. 14–19.
6. Bagula, A.; Lubamba, C.; Mandava, M.; Bagula, H.; Zennaro, M.; Pietrosemoli, E. Cloud based patient prioritization as service in public health care. In Proceedings of the 2016 ITU Kaleidoscope: ICTs for a Sustainable World (ITU WT), Bangkok, Thailand, 4–16 November 2016; pp. 1–8.
7. Las Fargeas, J.; Hyun, B.; Kabamba, P.; Girard, A. Persistent visitation with fuel constraints. *Procedia-Soc. Behav. Sci.* **2012**, *54*, 1037–1046. [CrossRef]
8. Las Fargeas, J.; Hyun, B.; Kabamba, P.; Girard, A. Persistent visitation under revisit constraints. In Proceedings of the 2013 International Conference on Unmanned Aircraft Systems (ICUAS), Atlanta, GA, USA, 28–31 May 2013; pp. 952–957.
9. Gao, H.; Qin, Y.; Hu, C.; Liu, Y.; Li, K. An interacting multiple model for trajectory prediction of intelligent vehicles in typical road traffic scenario. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, 1–12. [CrossRef] [PubMed]
10. Gao, H.; Kan, Z.; Li, K. Robust lateral trajectory following control of unmanned vehicle based on model predictive control. *IEEE/ASME Trans. Mechatron.* **2021**. [CrossRef]
11. Las Fargeas, J.; Kabamba, P.; Girard, A. Cooperative surveillance and pursuit using unmanned aerial vehicles and unattended ground sensors. *Sensors* **2015**, *15*, 1365–1388. [CrossRef] [PubMed]
12. Las Fargeas, J.; Kabamba, P.; Girard, A. Optimal configuration of alarm sensors for monitoring mobile ergodic markov phenomena on arbitrary graphs. *IEEE Sens. J.* **2015**, *15*, 3622–3634. [CrossRef]
13. Tuyishimire, E.; Adiel, I.; Rekhis, S.; Bagula, B.A.; Boudriga, N. Internet of Things in Motion: A Cooperative Data Muling Model Under Revisit Constraints. In Proceedings of the Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences, Toulouse, France, 18–21 June 2016; pp. 1123–1130.

14. Bagula, A.; Tuyishimire, E.; Wadepoel, J.; Boudriga, N.; Rekhis, S. Internet-of-Things in Motion: A Cooperative Data Muling Model for Public Safety. In Proceedings of the Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences, Toulouse, France, 18–21 June 2016; pp. 17–24.
15. Bender, M.A.; Fekete, S.P.; Kröller, A.; Liberatore, V.; Mitchell, J.S.; Polishchuk, V.; Suomela, J. The minimum backlog problem. *Theor. Comput. Sci.* **2015**, *605*, 51–61. [CrossRef]
16. Citovsky, G.; Gao, J.; Mitchell, J.S.; Zeng, J. Exact and approximation algorithms for data mule scheduling in a sensor network. In *International Symposium on Algorithms and Experiments for Wireless Sensor Networks*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 57–70.
17. Tuyishimire, E.; Bagula, B.A.; Ismail, A. Optimal clustering for efficient data muling in the internet-of-things in motion. In *International Symposium on Ubiquitous Networking*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 359–371.
18. Gupta, A.; Gupta, S.K.; Rashid, M.; Khan, A.; Manjul, M. Unmanned aerial vehicles integrated HetNet for smart dense urban area. In *Transactions on Emerging Telecommunications Technologies*; IEEE: Piscataway, NJ, USA, 2020.
19. Tuyishimire, E.; Bagula, A.; Ismail, A. Clustered data muling in the internet of things in motion. *Sensors* **2019**, *19*, 484. [CrossRef]
20. Ismail, A.; Bagula, B.A.; Tuyishimire, E. Internet-of-things in motion: A uav coalition model for remote sensing in smart cities. *Sensors* **2018**, *18*, 2184. [CrossRef]
21. Ismail, A.; Tuyishimire, E.; Bagula, A. Generating dubins path for fixed wing uavs in search missions. In *International Symposium on Ubiquitous Networking*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 347–358.
22. Gupta, A.; Sundhan, S.; Gupta, S.K.; Alsamhi, S.; Rashid, M. Collaboration of UAV and HetNet for better QoS: A comparative study. *Int. J. Veh. Inf. Commun. Syst.* **2020**, *5*, 309–333. [CrossRef]
23. Gupta, A.; Sundhan, S.; Alsamhi, S.; Gupta, S.K. Review for capacity and coverage improvement in aerially controlled heterogeneous network. In *Optical and Wireless Technologies*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 365–376.
24. Syed, F.; Gupta, S.K.; Hamood Alsamhi, S.; Rashid, M.; Liu, X. A survey on recent optimal techniques for securing unmanned aerial vehicles applications. *Trans. Emerg. Telecommun. Technol.* **2021**, *32*, e4133. [CrossRef]
25. Tuyishimire, E.; Bagula, A.; Rekhis, S.; Boudriga, N. *Cooperative Data Muling from Ground Sensors to Base Stations Using UAVs*; IEEE: Heraklion, Greece, 2017.
26. Tuyishimire, E. Internet of Things: Least Interference Beaconing Algorithms. Master's Thesis, University of Cape Town, Cape Town, South Africa, 2014.
27. Mauwa, H.; Bagula, A.; Tuyishimire, E.; Ngqondi, T. Community healthcare mesh network engineering in white space frequencies. In *2019 ITU Kaleidoscope: ICT for Health: Networks, Standards and Innovation (ITU K)*; IEEE: Piscataway, NJ, USA, 2019; pp. 1–8.
28. Mauwa, H.; Bagula, A.; Tuyishimire, E.; Ngqondi, T. An optimal spectrum allocation strategy for dynamic spectrum markets. In Proceedings of the 2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Barcelona, Spain, 21–23 October 2019; pp. 1–8.
29. Pal, A.; Memon, N. The evolution of file carving. *IEEE Signal Process. Mag.* **2009**, *26*, 59–71. [CrossRef]
30. Dijkstra, E.W. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. [CrossRef]
31. Laporte, G. The vehicle routing problem: An overview of exact and approximate algorithms. *Eur. J. Oper. Res.* **1992**, *59*, 345–358. [CrossRef]
32. Toth, P.; Vigo, D. An overview of vehicle routing problems. In *The Vehicle Routing Problem*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2002; pp. 1–26.
33. Tuyishimire, E. Routing in Mobile Networks. Master's Thsis, University of Cape Town, Cape Town, South Africa, 2013.
34. Zennaro, M.; Ntareme, H.; Bagula, A. Experimental evaluation of temporal and energy characteristics of an outdoor sensor network. In Proceedings of the International Conference on Mobile Technology, Applications, and Systems, Yilan, Taiwan, 10–12 September 2008; pp. 1–5.
35. Brown, T.X.; Pietrosemoli, E.; Zennaro, M.; Bagula, A.; Mauwa, H.; Nleya, S.M. A survey of TV white space measurements. In *International Conference on e-Infrastructure and e-Services for Developing Countries*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 164–172.
36. Tuyishimire, E.; Bagula, B.A. A Formal and Efficient Routing Model for Persistent Traffics in the Internet of Things. In Proceedings of the 2020 Conference on Information Communications Technology and Society (ICTAS), Durban, South Africa, 11–12 March 2020; pp. 1–6.
37. Tuyishimire, E.; Bagula, B.A. Modelling and analysis of interference diffusion in the internet of things: An epidemic model. In Proceedings of the 2020 Conference on Information Communications Technology and Society (ICTAS), Durban, South Africa, 11–12 March 2020; pp. 1–6.
38. Tuyishimire, E.; Bagula, B.A. A novel management model for dynamic sensor networks using diffusion sets. In Proceedings of the 2020 Conference on Information Communications Technology and Society (ICTAS), Durban, South Africa, 11–12 March 2020; pp. 1–6.
39. Antoine, B.; Emmanuel, T.; Olasupo, A. Cyber physical systems (cps) surveillance using an epidemic model. *arXiv* **2019**, arXiv:1912.07479.