ESTA: An Efficient Spatial-Temporal Range Aggregation Query Processing Algorithm for UAV Networks

Wenbin Zhai[a], Xin Li[a], Liang Liu[a,*], Youwei Ding[b], Wanying Lu[a]

[a]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China*
[b]*School of Artificial Intelligence and Information Technology, Nanjing University of Chinese Medicine, Nanjing, China*

Abstract

Unmanned Aerial Vehicle (UAV) networks have been widely used in both military and civilian scenarios. When users are interested in the statistical information of the historical sensory data in a certain region during a certain time period, they will send an aggregation query request with a spatial-temporal constraint to target UAVs which store the qualified data. Then, the target UAVs will return the query results to users. Meanwhile, the query results can be aggregated within the network during transmission to save energy and bandwidth resources, which are typically scarce in UAV networks. However, due to the unique characteristics of UAV networks, it is difficult to perform efficient in-network aggregation of query results without the sacrifice of the user query delay. To the best of our knowledge, there is no research on spatial-temporal range aggregation query in UAV networks. In this paper, we propose an Efficient Spatial-Temporal range Aggregation query processing (ESTA) algorithm for UAV networks. First, a topology change graph is constructed based on the pre-planned trajectory information. Meanwhile, an efficient shortest path algorithm is proposed to obtain the user query delay. Then, on the basis of ensuring the user query delay, ESTA transforms the aggregation processing of query results into recursively solving the set cover problem, thereby constructing a spatial-temporal aggregation tree (STAT), based on which an efficient in-network aggregation routing path for query results can be found. Through extensive simulation, we demonstrate that ESTA can save more than 50% of the energy consumption compared with the baseline algorithm.

Keywords:
UAV networks, query processing, spatial-temporal query, spatial-temporal aggregation tree.

1. Introduction

Over the past few decades, with the rapid development of sensors, navigation systems and wireless communication technologies, Unmanned Aerial Vehicles (UAVs) achieve significant performance improvements. Moreover, due to the low cost, flexible deployment and simple operation, they have been widely used in both military and civilian scenarios, such as military reconnaissance, border patrol, disaster response, farmland monitoring, etc [1].

In the most UAV application scenarios, numerous UAVs often form a multi-hop UAV network to cooperate with each other in order to efficiently collect data and complete missions. The multi-hop UAV network can be regarded as a distributed database, in which each UAV stores and carries a large amount of sensory data with spatial and temporal labels [2, 3]. When users are interested in the sensory data in a certain region during a certain time period, they will send a query request with the specific spatial-temporal constraint to each target UAV which stores the qualified data. When target UAVs receive the query request, they will search their respective locally stored sensory data and then return the data satisfying the spatial-temporal constraint to the ground station.

However, UAVs are energy-constraint and the bandwidth resources of UAV networks are typically scarce [4, 5], so the cost of sending the raw data of all query results back to the ground station is very expensive. Moreover, in many practical situations, users are only interested in the statistical information of the data, rather than all the specific information [6]. For instance, in the forest fire monitoring, users want to know the maximum temperature in the region r_1 during the time period between t_1 and t_2. In this case, it is not necessary to return all qualified data to the ground station, which will waste a lot of energy and bandwidth resources.

Data aggregation [7] is a promising technology to solve this problem. It is a form of in-network processing that utilizes various aggregation techniques to combine, abstract or compress raw data at intermediate nodes within the network [8]. For example, the raw data can be abstracted as mean, maximum, minimum, median and summary as needed. Through data aggregation, the ground station does not need to receive all the query results from target UAVs. The query results can be first aggregated at the intermediate nodes as needed, and then the aggregated data will be routed to the ground station. Therefore, data aggregation can effectively eliminate redundancy, extract useful information, and reduce the number and data size of transmissions, thereby reducing energy and bandwidth consumption.

Due to its high efficiency, data aggregation has attracted great attention from researchers in recent years [9–11]. However,

---

*Corresponding author, Email: liangliu@nuaa.edu.cn

most of them [12–14] focus on data aggregation in static wireless sensor networks (WSNs), rather than that in dynamic UAV networks. To the best of our knowledge, there is no research on spatial-temporal range aggregation query in UAV networks. Compared with conventional static WSNs, UAV networks have many unique characteristics, such as high mobility, sparse distribution, intermittent connectivity, unstable link quality and store-carry-forward (SCF) mechanism, which make the spatial-temporal range aggregation query in UAV networks faces the following challenges. First, due to the high dynamic of topology and intermittent connectivity of communications, it is difficult to find an efficient in-network aggregation routing for query results. In addition, what users most care about is the user query delay (i.e., the delay from sending the query request to receiving the query result), rather than how data is transmitted within the network [15]. Therefore, how to aggregate data effectively to reduce data communication and energy consumption without sacrificing the user query delay is a great challenge.

To overcome the above issues, in this paper, we propose an Efficient Spatial-Temporal range Aggregation query processing (ESTA) algorithm for UAV networks. The main idea of this paper is to aggregate query results during transmission as soon as possible to reduce energy and bandwidth consumption on the basis of ensuring the user query delay. First, ESTA utilizes the pre-planned trajectory information of UAVs to determine target UAVs. Meanwhile, the topology change graph (TCG) is constructed in order to reflect the communication windows between UAVs. According to the constructed TCG, an efficient shortest path algorithm is proposed, based on which the user query delay can be obtained. Then, we transform the aggregation processing of query results into recursively solving the set cover problem, thus constructing a spatial-temporal aggregation tree (STAT), based on which an efficient in-network aggregation routing path for query results can be found. In STAT, the root node is the ground station, and except for the leaf nodes, each node will aggregate query results of all its child nodes, and then forward the aggregated results to its parent node. The main contributions of this paper are as follows:

- We propose an Efficient Spatial-Temporal range Aggregation query processing (ESTA) algorithm for UAV networks, which can find an efficient in-network aggregation path for target UAVs without the sacrifice of the user query delay. As we all know, we are the first to study the spatial-temporal range aggregation query in UAV networks.

- We conduct extensive simulations on the Opportunistic Network Environment (ONE) simulator [16]. The experimental results show the superior performance of ESTA compared with the baseline spatial-temporal range aggregation query processing algorithm in terms of the query delay and energy consumption.

The remainder of the paper is organized as follows. Section 2 summarizes the state-of-the-art in spatial-temporal range aggregation query processing and routing protocols for UAV networks. The system model is introduced in Section 3. In Section 4, our proposed ESTA is described in detail. We provide the performance evaluation of ESTA through extensive simulation in Section 5, and conclude this paper in Section 6.

2. Related Work

In this section, we review related works in literature. First, we introduce the latest progress of data aggregation processing in static networks, since there is no such research in dynamic UAV networks. Then, since the efficiency of spatial-temporal range aggregation query processing relies on the design of the routing protocol, we summarize the state-of-the-art routing protocols in UAV networks.

2.1. Data aggregation processing in static networks

Due to the practicability and efficiency of data aggregation processing in energy and bandwidth conservation, it has attracted great attention from researchers in recent years. However, the existing researches on data aggregation processing mainly focus on static networks, which can be divided into tree-based, cluster-based and machine learning-based data aggregation [17].

In tree-based data aggregation, all nodes are arranged in the form of tree nodes. Except for leaf nodes, each node can act as an aggregator to aggregate sensory data from its child nodes and then transmit the aggregated data to its parent node up to the root node [18]. The authors in [19] study the complex query in Internet of Things (IoT). They combine the pruning ability and aggregation cost to define aggregation gain, based on which a data aggregation tree with minimum communication cost can be constructed. Two new data aggregation models are proposed in [9], based on which mixed-integer programming is exploited to jointly optimize the energy consumption of data aggregation and dissemination. The authors in [20] use integer linear programming to define the trade-off between the energy consumption and latency. Then based on heuristics, an aggregation tree whose root is the destination can be constructed to aggregate data, thereby reducing energy consumption and latency.

In cluster-based data aggregation, the network is divided into multiple clusters. Each cluster has a head node, which is responsible for aggregating sensory data from nodes in the cluster, and then delivering the aggregated data to the ground station. The authors in [12] propose a two-layer cluster-based distributed data aggregation algorithm which takes clustering and routing into consideration. The cluster-heads (CHs) are elected based on the relative connectivity, distance to the ground station and residual energy. Meanwhile, the fuzzy logic is utilized to optimize the routing paths from the CHs to the ground station. A multi-objective multi-attribute data aggregation scheme is proposed in [14], which consists of five phases, that is, CHs election, reliability measurement, data aggregation, scheduling policy and routing selection. The authors in [21] propose a cluster-based data aggregation scheme which utilizes prioritized channel access at CHs to minimize latency. Meanwhile, a novel queuing model is used to consider the joint effects of data scheduling and aggregation. The authors in [22] utilize compressed sensing to construct a coefficient measurement matrix from the network, which is used to assist in data aggregation.

In addition, Machining Learning (ML) has been widely used in data aggregation processing in WSNs. The authors in [23] consider a novel data aggregation scenario with duty cycle. The Markov decision process is used to co-optimize the duty cycle and data aggregation to achieve a good trade-off between energy consumption, throughput and data consistency. The authors in [24] utilize Q-learning, a reinforcement learning technique, to adaptively find the optimal aggregation routing. An optimal routing aggregation tree is constructed, which considers residual energy, distance between nodes and link quality. Moreover, the routing hole problem is also considered and then solved.

However, compared with static networks, UAV networks have many unique characteristics, such as high mobility, sparse distribution, intermittent connectivity, unstable link quality and store-carry-forward (SCF) mechanism. These characteristics make the above-mentioned approaches that rely on stable topology and constant connectivity inapplicable in UAV networks. As far as we know, there is no research on spatial-temporal range aggregation query processing in UAV networks.

2.2. Routing protocols in UAV networks

The spatial-temporal range aggregation query processing for UAV networks depends on the design of routing protocols. Therefore, we introduce the latest progress of energy optimization in routing protocols in UAV networks, which can be divided into topology-based and geographic routing protocols.

Topology-based routing protocols utilize network topology-related information for message forwarding and delivery. The authors in [25] transform the multicast energy minimization problem in delay-constraint UAV networks into the problem of solving the directed Steiner tree, thereby finding an efficient multicast routing. Moreover, the transmission and receiving energy are both considered when optimizing the transmission scheme. In [26], the redundancy offered by the network topology is exploited to make efficient routing decisions for energy efficiency and balance. Furthermore, a novel and unique routing diagnostic and recovery mechanism is provided based on the network topology redundancy.

Geographic routing protocols exploit local position information instead of global topology information for routing decisions. The pure idea is to forward packets to the neighbor node which is nearest to the destination. The authors in [27] adaptively utilize the position information, residual energy, and the characteristics of energy consumption to make routing decisions for better route recovery from routing holes. An improved vision of geographic routing is proposed in [28], which takes the position information, energy consumption and delivery delay into consideration. Meanwhile, an energy-aware routing tree is constructed which connects nodes on the shortest paths. A destination-driven energy-efficient multicast tree is provided in [29], which makes use of position information and energy characteristic to assist multicast routing decisions. However, due to the sparse distribution of nodes and intermittent connectivity of communications, pure geographic routing protocols sometimes are not efficient enough to cope with UAV networks.

In order to address the above challenges, a promising approach called store-carry-forward mechanism is proposed and widely used due to its simplicity and effectiveness. A novel Q-learning based multi-objective optimization routing protocol is proposed in [30] which can dynamically adjust the Q-learning parameters to adapt to the high dynamics of UAV networks. In addition, the acquired knowledge can also be used to explore and discover efficient routing paths to reduce energy consumption and delay. Based on the position information, the authors in [31] further take the prediction of trajectories and the load of UAVs into consideration to optimize packet forwarding.

However, none of the existing energy-efficient routing protocols for UAV networks consider the data aggregation in the aggregation query processing, which makes them inefficient in spatial-temporal range aggregation query processing. Furthermore, they do not fully utilize the trajectory information and topology change information in UAV networks to optimize routing.

3. System Model

3.1. Network Model

We consider the UAV network which consists of multiple UAVs and a ground station. Without loss of generality, we abstract the UAV network from the three-dimensional space into a Euclidean space, ignoring the vertical space [32]. Furthermore, in most application scenarios like military missions, the flight trajectories of UAVs are pre-planned and can be obtained in advance through mission planning and path planning [3, 33, 34]. Even if UAVs re-plan the trajectories during the mission, their trajectories can also be obtained by the ground station in advance through the out-of-band channel [35–37].

Each UAV flies along its respective pre-planned trajectory, collects spatial-temporal sensory data and then stores it locally. The ground station may send an aggregation query request with a specific spatial-temporal constraint to target UAVs as needed, denoted as $Q = Request(T, R, D, A)$, where T is the time period of the query, R is the target query region, D is the data type of the query and A is the aggregation operation. For example, if the ground station queries the maximum temperature in the region r_1 between t_1 and t_2, the query request can be expressed as $Q_1 = Request([t_1, t_2], r_1, TEMP, MAX)$. After receiving the query request, each target UAV will search its local sensory data, perform the aggregation operation and return the qualified data as its query result to the ground station. For instance, each target UAV first searches its local sensory data on temperature in r_1 between t_1 and t_2. Then, it performs "MAX" aggregation operation and uses the maximum temperature as its aggregation query result.

Meanwhile, the computing resources of UAVs are relatively abundant, however, the energy and bandwidth resources are relatively scarce [4]. Moreover, in UAV networks, the cost of the communication is multiple times that of the computation [5]. Therefore, in the spatial-temporal range aggregation query processing, each UAV in the network can aggregate multiple received query results, and then forward the aggregated result to

the ground station. For instance, after a UAV receives query results that contain the maximum temperature from multiple UAVs, it will aggregate these query results and select the highest maximum temperature as the aggregated query result. For convenience, in this paper, we assume that the packet size of the query result of each target UAV and the aggregated result is the same.

3.2. Topology Change Graph

After receiving the query requests, target UAVs will search their respective local sensory data and then send the aggregation query results back to the ground station. Meanwhile, these query results should be further aggregated within the network during transmission to reduce the energy and bandwidth consumption. However, due to the high dynamic of topology and intermittent connectivity of communications in UAV networks, it is difficult to find an efficient in-network aggregation routing for query results. To solve the above problem, in this paper, we build the topology change graph (TCG) based on the pre-planned trajectories of UAVs in order to accurately reflect communication windows between UAVs and topology changes of the network.

The changes of the UAV network topology can be abstracted into a topology change graph, which can be formalized as $TCG =< V, E >$, where V is the set of UAVs in the network and E represents communication links between UAVs. Based on the pre-planned trajectories of UAVs, we can calculate the communication windows between UAVs. It is worth noting that due to the high dynamic of topology, there may be multiple intermittent connections and communications between two UAVs. Without loss of generality, in this paper, we abstract them as a continuous time period [3]. In other words, for each $e_{ij} \in E$, it can be formalized as a quadruple $e_{ij} =< u_i, u_j, t_{begin}^{ij}, t_{end}^{ij} >$, which represents that u_i and u_j can communicate with each other between t_{begin}^{ij} and t_{end}^{ij}. For example, as Fig. 1 shows, there are ten UAVs $u_1 \sim u_{10}$ and a ground station g_0. The edges represent communication windows between UAVs, such as UAV u_1 and u_4 can communicate with each other between $0s$ and $2s$, namely $e_{14} =< u_1, u_4, 0s, 2s >$.

4. ESTA

4.1. Architecture

Fig. 2 shows the architecture of ESTA. It consists of three phases: target UAVs determination, query requests distribution and query results aggregation.

4.1.1. Target UAVs Determination (Section 4.2)

When users are interested in the sensory data in a certain region during a certain time period, the ground station first uses the pre-planned trajectories of UAVs to determine target UAVs which store and carry the qualified data.

4.1.2. Query Requests Distribution (Section 4.3)

After obtaining target UAVs, the ground station distributes aggregation query requests to target UAVs based on a certain query distribution routing protocol.
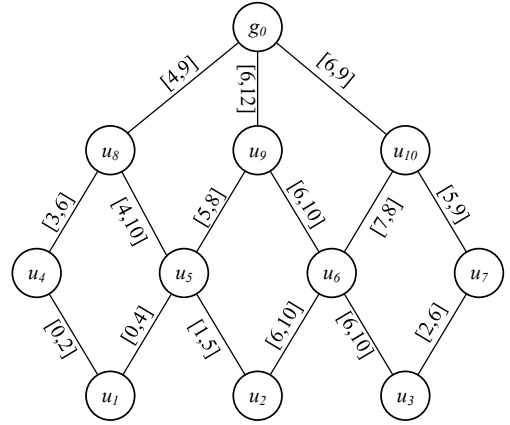

Figure 1: The topology change graph.

4.1.3. Query Results Aggregation (Section 4.4)

When target UAVs receive the query request, they search their respective locally stored data, perform the aggregation operation on sensory data satisfying the spatial-temporal constraint and then return the aggregation query results to the ground station. Meanwhile, in order to reduce energy and bandwidth consumption without sacrificing the user query delay, an efficient spatial-temporal aggregation tree is constructed, based on which the query results from different target UAVs can be further aggregated at the intermediate nodes during transmission.
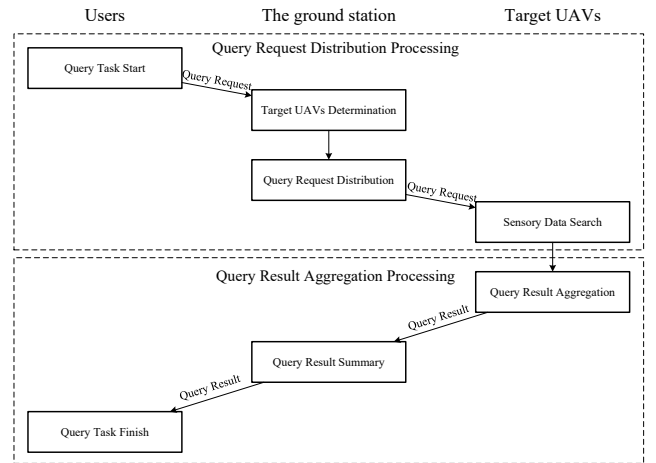

Figure 2: The spatial-temporal range aggregation query processing architecture.

4.2. Target UAVs Determination

Most of the existing researches [12] on spatial-temporal range aggregation query processing focus on static WSNs. In static WSNs, since the positions of nodes are fixed and usually do not change, it is very easy to determine target nodes that store and carry the qualified data, that is, nodes in the target query region.

4

However, in UAV networks, due to the high mobility of nodes and high dynamic of topology, when the ground station sends out the spatial-temporal range aggregation query request, the qualified target UAVs may have left the target query region. For instance, as shown in Fig. 3, UAVs fly in the mission coverage region and collect sensory data, $u_i^{t_j}$ represents the position of u_i at t_j. At t_2, the ground station g_0 requests to query the maximum temperature in the target region r_1 between t_1 and t_2, denoted as $Q_1 = Request([t_1, t_2], r_1, TEMP, MAX)$. We can find that at t_2, UAV u_3 and u_5 have left the target query region r_1 due to the flight movement. In this case, if we collect and aggregate data in the same way as static WSNs, the stored data in u_3 and u_5 will be missed.
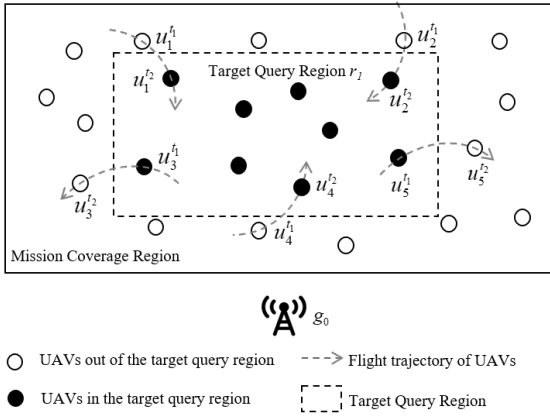

Figure 3: The UAV network status.

Another feasible method is to use flooding to distribute the spatial-temporal range aggregation query request to all nodes in the network without distinguishing whether they store the query data or not[38]. This approach ensures the accuracy and correctness of the query, however, it consumes a huge amount of energy and bandwidth resources, which are typically scarce in UAV networks.

Therefore, in order to make up for these shortcomings, in this paper, we utilize the pre-planned trajectory information of UAVs to assist in determining target UAVs. First, the time period of the query T is abstracted into n discrete time points, denoted as $T =< t_1, \ldots, t_j, t_{j+1}, \ldots, t_n >$. Then, since the trajectory of each UAV is pre-planned, for each UAV u_i, we can calculate its position at t_j, denoted as $u_i^{t_j} = (x_{ij}, y_{ij}), 1 \le j \le n$. Based on the obtained n position points, the trajectory of UAV u_i in the time period T can be approximately divided into $n-1$ line segments, denoted as $FT_i =< a_1, \ldots, a_j, \ldots, a_{n-1} >$, where $a_j = (u_i^{t_j}, u_i^{t_{j+1}})$. For UAV u_i, if there is a line segment a_j that interacts with the target query region R, u_i is considered as a target UAV. Finally, we will obtain the set of target UAVs, denoted as U_{target}.

4.3. Query Requests Distribution

After obtaining target UAVs, the ground station needs to distribute query requests to target UAVs. However, due to the unique characteristics of UAV networks, e.g., high mobility and intermittent connectivity, traditional routing protocols for well-connected networks, such as Ad hoc On-Demand Distance Vector (AODV) based [39] or Optimized Link State Routing (OLSR) based [40] routing protocols, are not suitable for dynamic UAV networks. Therefore, in order to successfully distribute query requests to each target UAV, in this paper, multi-hop routing protocols for UAV networks [25, 27, 31] are utilized. Especially, multicast routing protocols, whose target is to deliver messages from the source to a group of destinations, are well suited for query requests distribution, and there is a considerable research effort for the development of multicast routing protocols for UAV networks [3, 25, 35].

Based on the above method, the ground station can efficiently distribute the query requests to target UAVs. In addition, it is worth noting that no matter what routing protocol is used, there is no guarantee that the query requests reach target UAVs and target UAVs send out the query results at the same time, which makes the assumption of synchronous transmission in most existing methods [7, 12, 23] untenable and unrealistic. However, our query result aggregation scheme (see in Section 4.4) does not rely on this assumption and can cope well with the asynchrony of the transmissions.

4.4. Query Results Aggregation
4.4.1. Basic idea

In the spatial-temporal range aggregation query processing, the ground station needs to receive query results from multiple target UAVs, and the query task becomes successful only when all query results are delivered to the ground station. Therefore, the user query delay is the time when all query results arrive at the ground station. Meanwhile, for the spatial-temporal range aggregation query task, query results from different target UAVs can be aggregated at the intermediate nodes of the network during the transmission process. For example, in Fig. 1, u_5 can aggregate the query results from u_1 and u_2, then deliver the aggregated result to g_0.

Based on the above observations and analyses, the main idea of this paper is to aggregate query results as soon as possible on the basis of ensuring the user query delay, thereby reducing data communication and energy consumption in the network. As depicted in Fig. 1, u_1, u_2 and u_3 are target UAVs and need to send query results back to the ground station. For convenience, in this paper, we assume that the duration to transmit a query result from one node to its neighbor node is $0.1s$, denoted as $t_{trans} = 0.1s$, and the corresponding energy consumption is E_m due to the same packet size. Their transmission paths with the earliest delivery time to the ground station are

1. $pa_1 : u_1 \xrightarrow{[0,2]} u_4 \xrightarrow{[3,6]} u_8 \xrightarrow{[4,9]} g_0$. The earliest delivery time of the query result of u_1 is $4.1s$, and the corresponding energy consumption is $3E_m$.
2. $pa_2 : u_2 \xrightarrow{[1,5]} u_5 \xrightarrow{[0,4]} u_1 \xrightarrow{[0,2]} u_4 \xrightarrow{[3,6]} u_8 \xrightarrow{[4,9]} g_0$. The earliest delivery time of the query result of u_2 is $4.1s$, and the corresponding energy consumption is $5E_m$.
3. $pa_3 : u_3 \xrightarrow{[2,6]} u_7 \xrightarrow{[5,9]} u_{10} \xrightarrow{[6,9]} g_0$. The earliest delivery time of the query result of u_3 is $6.1s$, and the corresponding energy consumption is $3E_m$.

5

If each query result is delivered along its shortest path, the total energy consumption is $3E_m + 5E_m + 3E_m = 11E_m$. Meanwhile, if we take in-network aggregation into consideration, the query results of u_1 and u_2 can be aggregated at UAV u_4. As shown in Fig. 4(a), UAV u_4 will receive query results from u_1 and u_2 at $0.1s$ and $1.3s$, u_4 can aggregate them and then forward the aggregated result to u_8 at $3s$. In this case, the total energy consumption becomes $3E_m + 1E_m + 2E_m + 3E_m = 9E_m$.

However, the above schemes are not efficient enough. Since the earliest delivery time of the last arrival packet (i.e., the query result of u_3) is $6.1s$, we take $6.1s$ as the delay constraint of all query results in order to ensure the user query delay. Then, we aggregate query results of all target UAVs in the network as soon as possible, so we can obtain a better transmission scheme, as shown in Fig. 4(b). The query results of u_1 and u_2 will be aggregated at u_5 and then the aggregated result will be forwarded to g_0. The total energy consumption of this scheme is $1E_m + 1E_m + 2E_m + 3E_m = 7E_m$. In this case, the user query delay is the same as the above two schemes (i.e., $6.1s$), meanwhile, the energy consumption is further reduced.
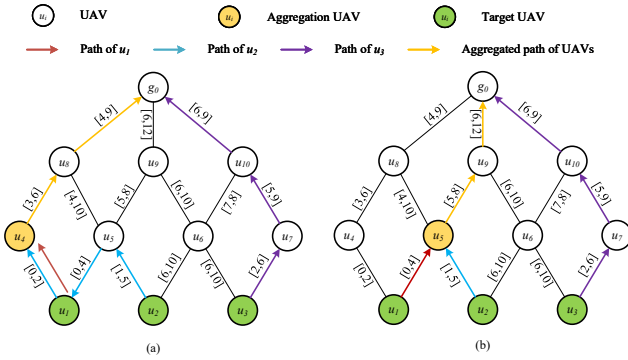

Figure 4: The basic idea of ESTA.

4.4.2. Determine the user query delay

The main idea of this paper is to aggregate query results as soon as possible on the basis of ensuring the user query delay, thereby reducing data communication and energy consumption in the network. Therefore, we need to study and determine the user query delay first.

In the aggregation query processing, the query task becomes successful only when all query results return to the ground station, that is, the time when the last query result delivered to the ground station is the user query delay. We assume that there are a total of k target UAVs, denoted as $U_{target} = \{u_1, u_2, \ldots, u_i, \ldots, u_k\}$. Without loss of generality, we take the minimum delay required for a query task as the user query delay. In other words, for the query result of each target UAV u_i, its earliest delivery time sent back to the ground station is t_i, so the user query delay is

$$t_d = \max\{t_1, t_2, \ldots, t_i, \ldots, t_k\} \tag{1}$$

Therefore, in order to obtain the user query delay, we need to calculate the earliest delivery time for each query result.

In this paper, based on the constructed TCG, the problem of the earliest delivery time can be transformed into the shortest path problem. However, it is worth noting that, different from edges in traditional topology graph in static networks, edges in TCG have lifetime, which makes the shortest path problem unique and complex in UAV networks. Specifically, for $e_{ij} = <u_i, u_j, t_{begin}^{ij}, t_{end}^{ij}>$ and $e_{jk} = <u_j, u_k, t_{begin}^{jk}, t_{end}^{jk}>$, when u_j receives m from u_i at t_{rx}^{ij}, and $t_{begin}^{jk} + t_{trans} \le t_{end}^{jk}$, where t_{trans} is the duration to transmit a query result m from u_j to u_k, there will be three cases:

- If $t_{rx}^{ij} < t_{begin}^{jk}$, the communication e_{jk} between u_j and u_k has not started yet, u_j has to store and carry m until t_{begin}^{jk}, and then u_j can forward m to u_k through e_{jk}.

- If $t_{begin}^{jk} \le t_{rx}^{ij} \le t_{end}^{jk} - t_{trans}$, u_j can immediately forward m to u_k through e_{jk}.

- If $t_{rx}^{ij} > t_{end}^{jk} - t_{trans}$, the communication e_{jk} between u_j and u_k is not enough to transmit m (i.e., $t_{end}^{jk} - t_{trans} < t_{rx}^{ij} \le t_{end}^{jk}$) or even has finished (i.e., $t_{rx}^{ij} > t_{end}^{jk}$), u_j can not forward m to u_k through e_{jk}.

In brief, the query result m can be forwarded from u_j to u_k through e_{jk} if and only if

$$\max\{t_{rx}^{ij}, t_{begin}^{jk}\} + t_{trans} \le t_{end}^{jk} \tag{2}$$

Based on the above observations and analyses, in this section, we propose the Shortest Path Algorithm (SPA) for TCG. For each target UAV u_s in U_{target}, we use A to denote the set of nodes that have found the shortest path which have the earliest delivery time from the source node (i.e., u_s) to themselves, and B to represent the set of nodes that have not yet found the shortest path, namely $B = V - A$. Meanwhile, t_{gen}^s is the time when the query result of u_s is generated, and $T(u_s, u_i)$ and $H(u_s, u_i)$ are the earliest delivery time and required hop count along the shortest path from u_s to u_i, where $u_i \in V$. To start with, set $A = \emptyset$ and $B = V$. Besides, $T(u_s, u_s) = t_{gen}^s, H(u_s, u_s) = 0$ and $T(u_s, u) = \infty, H(u_s, u) = \infty, u \in V - \{u_s\}$. The algorithm is expressed as follows:

1. The node that has the earliest delivery time in set B, denoted as u_i, is removed from this set and added into set A.

2. If u_i is the ground station g_0, return to Step 1; otherwise, consider each node which is adjacent to u_i and still in set B, denoted as $u_j \in U_{neighbor}^i - A$. If u_i can forward the query result of u_s to u_j through the edge e_{ij} (i.e., $\max\{T(u_s, u_i), t_{begin}^{ij}\} + t_{trans} \le t_{end}^{ij}$), we judge whether $\max\{T(u_s, u_i), t_{begin}^{ij}\} + t_{trans}$ is smaller than $T(u_s, u_j)$, and if so, we update $T(u_s, u_j)$ as $\max\{T(u_s, u_i), t_{begin}^{ij}\} + t_{trans}$. Meanwhile, $H(u_s, u_j)$ is updated to $H(u_s, u_i) + 1$. This means through u_i, the query result of u_s can be delivered from u_s to u_j earlier.

3. If $B = \emptyset$, the process is done, which means the shortest paths, corresponding earliest delivery times and required

hop counts from the target UAV u_s to each node in the network are all found; otherwise, return to Step 1.

Algorithm 1 SPA: The Shortest Path Algorithm for TCG

Input:
    U_{target}, TCG
Output:
    $T(u_s, u_i), prev(u_s, u_i), H(u_s, u_i), u_s \in U_{target}, u_i \in V$
1: **for** each $u_s \in U_{target}$ **do**
2:     $A = \emptyset, B = V, T(u_s, u_s) = t_{gen}^s, H(u_s, u_s) = 0$
3:     **for** each $u \in V - u_s$ **do**
4:         $T(u_s, u) = \infty$
5:         $prev(u_s, u) = null$
6:         $H(u_s, u) = \infty$
7:     **end for**
8:     **while** $B \neq \emptyset$ **do**
9:         $u_i \Leftarrow \underset{u \in B}{\arg\min}\ T(u_s, u)$
10:        $A = A \cup \{u_i\}$
11:        $B = B - \{u_i\}$
12:        **for** each $u_j \in U_{neighbor}^i - A$ **do**
13:           $T_i(u_s, u_j) = \max\{T(u_s, u_i), t_{begin}^{ij}\} + t_{trans}$
14:           **if** $T_i(u_s, u_j) \leq t_{end}^{ij}$ && $T_i(u_s, u_j) < T(u_s, u_j)$ **then**
15:             $T(u_s, u_j) = T_i(u_s, u_j)$
16:             $prev(u_s, u_j) = u_i$
17:             $H(u_s, u_j) = H(u_s, u_i) + 1$
18:           **end if**
19:        **end for**
20:     **end while**
21: **end for**

After describing the main stages of the shortest path algorithm for TCG, we summarize it in Algorithm 1. If there is a path that can successfully deliver the query result from the target UAV to the ground station, SPA can always find the shortest one which has the earliest delivery time. Through SPA, we can obtain the earliest delivery time for the query result of each target UAV to the ground station, namely $t_1, t_2, \ldots, t_i, \ldots, t_k$. Then, as mentioned above, we take the maximum value as the user query delay, namely $t_d = \max\{t_1, t_2, \ldots, t_i, \ldots, t_k\}$.

Taking Fig. 1 as an example, target UAVs are u_1, u_2, u_3. For convenience, in this paper, we assume that $t_{trans} = 0.1s$ and $t_{gen}^s = 0s, u_s \in U_{target}$. According to Algorithm 1, we can calculate the earliest delivery time and required hop count from each target UAV to all nodes in the network, as depicted in Table 1 and 2. The earliest delivery times from u_1, u_2, u_3 to g_0 are $4.1s, 4.1s, 6.1s$ respectively, that is, $t_1 = 4.1s, t_2 = 4.1s, t_3 = 6.1s$. Therefore, the user query delay is $t_d = \max\{4.1s, 4.1s, 6.1s\} = 6.1s$.

4.4.3. Minimum Forwarding Set and Set Cover Problem

After obtaining the user query delay (i.e., t_d), we use it as the delay constraint of all query results, because the query task becomes successful only when the last query result reaches the ground station, and the early arrival of other query results is meaningless. Moreover, on the basis of ensuring the user query

Table 1: The Earliest Delivery Time

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_{10}	g_0
u_1	0	1.1	6.2	0.1	0.1	6.1	7.2	3.1	5.1	7.1	4.1
u_2	1.2	0	6.2	1.3	1.1	6.1	7.2	3.1	5.1	7.1	4.1
u_3	∞	6.2	0	∞	6.3	6.1	2.1	6.4	6.2	5.1	6.1

Table 2: The Hop Count of The Shortest Path

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_{10}	g_0
u_1	0	2	4	1	1	3	5	2	2	4	3
u_2	2	0	2	3	1	1	3	4	2	2	5
u_3	∞	2	0	∞	3	1	1	4	2	2	3

delay (i.e., ensuring all query result packets can be delivered to the ground station before t_d), aggregating query results in the network can reduce data communication and energy consumption without the sacrifice of the query delay.

Therefore, in order to enable the query results of different target UAVs to be aggregated in the network as soon as possible, in this section, we first propose the concept of minimum forwarding set for TCG.

Definition 4.1 (Minimum Forwarding Set). *u_i is a node in TCG, and needs to receive the query results from U_{target}^i before t_d^i, where U_{target}^i is the subset of U_{target} and t_d^i is the delay constraint of query results of U_{target}^i to u_i. Moreover, $U_{neighbor}^i$ is the set of neighbor nodes of u_i, if there exists a set $N_i \subseteq U_{neighbor}^i$, such that*

1. *The query results of all target UAVs in U_{target}^i can be delivered to u_i through the nodes in N_i before t_d^i;*
2. *If any element in N_i is removed, the query result of at least one target UAV in U_{target}^i can not be delivered to u_i through N_i before t_d^i,*

then N_i is called the minimum forwarding set of u_i.

Thus, we formulate the aggregation processing of query results as the minimum forwarding set problem, namely recursively finding the minimum forwarding set from the ground station to all target UAVs. In order to solve the above problem, for any u_j which is a neighbor node of u_i, we define the deliverable target UAVs of u_j as the subset of U_{target}^i whose elements are the target UAVs that can be delivered to u_i through u_j before t_d^i, denoted as $U_{target}^j \subseteq U_{target}^i$. Then, based on the shortest path algorithm for TCG, for each neighbor node u_j of u_i, we can further calculate the deliverable target UAVs of u_j.

The process can be expressed as follows: First, for any $u_j \in U_{neighbor}^i$, we can obtain the earliest delivery time from $u_s \in$

U^i_{target} to u_j (i.e., $T(u_s, u_j), u_s \in U^i_{target}$) according to the shortest path algorithm. Then, since u_j is the neighbor node of u_i, the edge between u_i and u_j is denoted as $e_{ij} = < u_i, u_j, t^{ij}_{begin}, t^{ij}_{end} >$. Therefore, for each $u_s \in U^i_{target}$, if $\max\{T(u_s, u_j), t^{ij}_{begin}\} + t_{trans} \leq \min\{t^{ij}_{end}, t^i_d\}$, the query result of u_s can be delivered to u_i through u_j before t^i_d and we add u_s into U^j_{target}. Meanwhile, in order to ensure that all query results of U^i_{target} can be delivered to u_i through e_{ij} before t^i_d, all query results of U^j_{target} should be delivered to u_j before t^j_d. In other words, t^j_d is the delay constraint of all query results from U^j_{target} to u_j, and $t^j_d = \min\{t^i_d, t^{ij}_{end}\} - t_{trans}$. Algorithm 2 summarizes this process in detail.

For example, in Fig. 1, UAV u_1, u_2, u_3 are target UAVs of g_0 (i.e., $U^0_{target} = \{u_1, u_2, u_3\}$) and the delay constraint of g_0 is $6.1s$ (i.e., $t^0_d = 6.1s$), which means the query results of all target UAVs need to be delivered to g_0 before $6.1s$. UAV u_8, u_9, u_{10} are the neighbor nodes of g_0. For UAV u_8, the adjacent edge is $e_{80} = < u_8, g_0, 4s, 9s >$, and according to Table 1, $T(u_1, u_8) = 3.1s$, $T(u_2, u_8) = 3.1s$, $T(u_3, u_8) = 6.4s$. Then for target UAV u_1, since $\max\{3.1s, 4s\} + 0.1s < \min\{9s, 6.1s\}$, the query result of u_1 can be delivered to g_0 through u_8 before $6.1s$. Similarly, the query result of u_2 can be delivered to g_0 through u_8 before $6.1s$. However, for target UAV u_3, since $\max\{6.4s, 4s\} + 0.1s > \min\{9s, 6.1s\}$, the query result of u_3 can not be delivered to g_0 through u_8 before $6.1s$. Therefore, the deliverable target UAVs of u_8 are u_1, u_2, denoted as $U^8_{target} = \{u_1, u_2\}$. Similarly, $U^9_{target} = \{u_1, u_2\}$ and $U^{10}_{target} = \{u_3\}$. Meanwhile, $t^8_d = \min\{t^0_d, t^{80}_{end}\} - t_{trans} = \min\{6.1s, 9s\} - 0.1s = 6s$. Similarly, $t^9_d = 6s$ and $t^{10}_d = 6s$.

Algorithm 2 DTU: The Deliverable Target UAVs algorithm

Input:
   $u_i, U^i_{target}, t^i_d, U^i_{neighbor}, TCG$
Output:
   $U^j_{target}, t^j_d, u_j \in U^i_{neighbor}$
1: **for** each $u_j \in U^i_{neighbor}$ **do**
2:    $U^j_{target} = \emptyset$
3:    **for** each $u_s \in U^i_{target}$ **do**
4:       **if** $\max\{T(u_s, u_j), t^{ij}_{begin}\} + t_{trans} \leq \min\{t^{ij}_{end}, t^i_d\}$ **then**
5:          $U^j_{target} = U^j_{target} \cup \{u_s\}$
6:       **end if**
7:    **end for**
8:    $t^j_d = \min\{t^i_d, t^{ij}_{end}\} - t_{trans}$
9: **end for**

After obtaining the deliverable target UAVs of each neighbor nodes of u_i (i.e., $U^j_{target}, u_j \in U^i_{neighbor}$), the above minimum forwarding set problem can be transformed into a set cover problem.

Definition 4.2 (Set Cover Problem). u_i *is a node in TCG and its deliverable target UAVs are U^i_{target}. The neighbor nodes of u_i are $U^i_{neighbor} = \{u_1, u_2, \ldots, u_j\}$, and their corresponding deliverable target UAVs are $U = \{U^1_{target}, U^2_{target}, \ldots, U^j_{target}\}$. More-*

over, $\bigcup_{k=1}^{j} U^k_{target} = U^i_{target}$. *The set cover problem is to identify the smallest subset of U whose union equals U^i_{target}.*

As we all know, the optimization version of set cover is NP-hard [41] and there are considerable efforts to solve this problem [42]. For convenience, in this paper, we use the greedy algorithm to solve the above set cover problem. The main idea is to choose the set that contains the largest number of uncovered elements at each stage. When there are multiple largest subsets, we choose the one with the smallest maximum hop count from their respective deliverable target UAVs to themselves. Moreover, in each round, we remove the covered elements from the unselected subsets to ensure that each element appears in only one of the smallest subsets in the end. This is because the query result of each target UAV only need to be aggregated at most once. The repeated appearance of elements will cause the query results of target UAVs to be aggregated and delivered multiple times, which will lead to unnecessary consumption of energy and bandwidth resources. Algorithm 3 summarizes this process in detail.

For instance, $U^0_{target} = \{u_1, u_2, u_3\} = U$ and UAV u_8, u_9, u_{10} are neighbor nodes of g_0 where $U^8_{target} = \{u_1, u_2\}$, $U^9_{target} = \{u_1, u_2\}$, $U^{10}_{target} = \{u_3\}$. First, $N_0 = \emptyset$, since $|U^8_{target}| = |U^9_{target}| > |U^{10}_{target}|$, and according to Table 2, $\max\{H(u_1, u_8), H(u_2, u_8)\} = \max\{2, 4\} = 4$, $\max\{H(u_1, u_9), H(u_2, u_9)\} = \max\{2, 2\} = 2 < 4$, we choose U^9_{target} and add it into the smallest subset, $N_0 = \{U^9_{target} = \{u_1, u_2\}\}$. Then, we delete the corresponding covered elements from U, U^8_{target} and U^{10}_{target}, namely $U = U - U^9_{target} = \{u_3\}$, $U^8_{target} = U^8_{target} - U^9_{target} = \emptyset$ and $U^{10}_{target} = U^{10}_{target} - U^9_{target} = \{u_3\}$. Next, since $|U^{10}_{target}| > |U^8_{target}|$, we choose U^{10}_{target} and add it into the smallest subset, $N_0 = \{U^9_{target} = \{u_1, u_2\}, U^{10}_{target} = \{u_3\}\}$. Meanwhile, $U = U - U^{10}_{target} = \emptyset$ and $U^8_{target} = U^8_{target} - U^{10}_{target} = \emptyset$. Since $U = \emptyset$, the process is done and the smallest subset of U^0_{target} is $\{U^9_{target} = \{u_1, u_2\}, U^{10}_{target} = \{u_3\}\}$.

Algorithm 3 MSC: Minimum Set Cover algorithm

Input:
   $u_i, U^i_{target}, U^i_{neighbor}, DTU$
Output:
   N_i
1: $U = U^i_{target}$
2: $N_i = \emptyset$
3: **while** $U \neq \emptyset$ **do**
4:    **for** each $u_j \in U^i_{neighbor}$ **do**
5:       $M \Leftarrow \underset{u_j \in U^i_{neighbor}}{\arg\max} |U^j_{target}|$
6:       $u_k \Leftarrow \underset{u_l \in M}{\arg\min} \max\{H(u_s, u_l), u_s \in U^l_{target}\}$
7:       $U = U - U^k_{target}$
8:       $N_i = N_i \cup \{U^k_{target}\}$
9:       **for** each $u_r \in U^i_{neighbor} - \{u_k\}$ **do**
10:          $U^r_{target} = U^r_{target} - U^k_{target}$
11:       **end for**
12:    **end for**
13: **end while**

8

4.4.4. Build the spatial-temporal aggregation tree

To ensure query results to be aggregated within the network as soon as possible, we propose the concept of minimum forwarding set and transform it into a set cover problem. In addition, the aggregation processing of query results can be transformed into recursively finding the minimum forwarding set from the ground station g_0 to all target UAVs U_{target}, thus constructing a Spatial-Temporal Aggregation Tree (STAT). In this section, we describe the construction procedure of the spatial-temporal aggregation tree. We use a queue Q to assist the insertion of nodes in STAT. Meanwhile, we use a set F to keep track of the nodes that have been added into STAT. The algorithm is expressed as follows:

1. Add the ground station g_0 into Q and insert g_0 into STAT as the root node. Meanwhile, we initialize set $F = \{g_0\}$.

2. Take the first element in the queue Q, denoted as u_i, and obtain its neighbor nodes $U_{neighbor}^i$. Note that $U_{neighbor}^i$ does not contain elements in set F, that is, nodes that have been added into STAT will not be considered as neighbor nodes of any UAV.

3. For each neighbor node of u_i, denoted as u_j, if $u_j \in U_{target}^i$, add u_j into STAT as the child node of u_i and add it into set F. Meanwhile, we delete u_j from U_{target}^i and $U_{neighbor}^i$. This means the aggregation path from the target UAV u_j to the ground station g_0 is found.

4. Calculate the deliverable target UAVs of each $u_j \in U_{neighbor}^i$, and then find the minimum forwarding set of u_i, that is, the minimum subset of U_{target}^i, denoted as N_i.

5. For each u_k whose $U_{target}^k \in N_i$, add u_k into the queue Q and insert it into STAT as a child node of u_i. Meanwhile, add u_k into set F, which means u_k has been added into STAT.

6. If the queue Q is empty or all target UAVs have been added into STAT (i.e., $U_{target} \subseteq F$), the construction process of STAT is done, which means the aggregation paths of all target UAVs have been found; otherwise, go to Step 2.

After describing the main stages of the construction of the spatial-temporal aggregation tree, we summarize it in Algorithm 4.

In Fig. 5, we build a spatial-temporal aggregation tree for the example in Fig. 1 to illustrate the construction process. First, g_0 is added into Q and inserted into STAT as the root node. $F = \{g_0\}$. We take out g_0 from Q, as mentioned above, since $U_{target} = \{u_1, u_2, u_3\}$ and the minimum forwarding set of g_0 is $N_0 = \{U_{target}^9 = \{u_1, u_2\}, U_{target}^{10} = \{u_3\}\}$. We add u_9 and u_{10} into Q and insert them into STAT as the child nodes of g_0. Meanwhile, $F = \{g_0, u_9, u_{10}\}$.

Then, as shown in Fig. 5(c), we take out u_9 from Q. Since $g_0 \in F$, the neighbor nodes of u_9 are u_5 and u_6. For UAV u_5, $\max\{T(u_1, u_5), t_{begin}^{59}\} + t_{trans} = \max\{0.1s, 5s\} + 0.1s = 5.1s$, $\max\{T(u_2, u_5), t_{begin}^{59}\} + t_{trans} = \max\{1.1s, 5s\} + 0.1s = 5.1s$ and $\min\{t_{end}^{59}, t_d^9\} = \min\{8s, 6s\} = 6s$. Since $5.1s < 6s$, $U_{target}^5 = \{u_1, u_2\}$. Similarly, for UAV u_6, $\max\{T(u_1, u_6), t_{begin}^{69}\} + t_{trans} = \max\{6.1s, 6s\} + 0.1s = 6.2s$, $\max\{T(u_2, u_6), t_{begin}^{69}\} + t_{trans} =$

Algorithm 4 STA-Tree

Input:
    $g_0, U_{target}, t_d, TCG$

Output:
    $STAT$

1: Q.push(g_0)
2: STA.add($g_0, null$)
3: $F = \{g_0\}$
4: **while** !Q.isEmpty() & $U_{target} \not\subseteq F$ **do**
5:     $u_i \Leftarrow Q$.poll()
6:     $U_{neighbor}^i = U_{neighbor}^i - F$
7:     **for** each $u_j \in U_{neighbor}^i$ **do**
8:         **if** $u_j \in U_{target}^i$ **then**
9:           STA.add(u_j, u_i)
10:          $F = F \cup \{u_j\}$
11:          $U_{target}^i = U_{target}^i - \{u_j\}$
12:          $U_{neighbor}^i = U_{neighbor}^i - \{u_j\}$
13:         **end if**
14:     **end for**
15:     $N_i \Leftarrow MSC(u_i, U_{target}^i, U_{neighbor}^i, DTU(u_i, U_{target}^i, t_d^i, U_{neighbor}^i, TCG))$
16:     **for** each $u_k, U_{target}^k \in N_i$ **do**
17:         Q.push(u_k)
18:         STA.add(u_k, u_i)
19:         $F = F \cup \{u_k\}$
20:     **end for**
21: **end while**

$\max\{6.1s, 6s\} + 0.1s = 6.2s$ and $\min\{t_{end}^{69}, t_d^9\} = \min\{10s, 6s\} = 6s$. Since $6.2s > 6s$, $U_{target}^6 = \emptyset$. Therefore, the minimum forwarding set of u_9 is $N_9 = \{U_{target}^5 = \{u_1, u_2\}\}$. We add u_5 into STAT as the child node of u_9 and add it into Q. Meanwhile, $F = \{g_0, u_9, u_{10}, u_5\}$.

Next, as Fig. 5(d) shows, we take out u_{10} from Q. The neighbor nodes of u_{10} are u_6 and u_7. For UAV u_6, $\max\{T(u_3, u_6), t_{begin}^{610}\} + t_{trans} = \max\{6.1s, 7s\} + 0.1s = 7.1s$ and $\min\{t_{end}^{610}, t_d^{10}\} = \min\{8s, 6s\} = 6s$. Since $7.1s > 6s$, $U_{target}^6 = \emptyset$. For UAV u_7, $\max\{T(u_3, u_7), t_{begin}^{710}\} + t_{trans} = \max\{2.1s, 5s\} + 0.1s = 5.1s$ and $\min\{t_{end}^{710}, t_d^{10}\} = \min\{9s, 6s\} = 6s$. Since $5.1s < 6s$, $U_{target}^7 = \{u_3\}$. Therefore, the minimum forwarding set of u_{10} is $N_{10} = \{U_{target}^7 = \{u_3\}\}$. We add u_7 into STAT as the child node of u_{10} and add it into Q. Meanwhile, $F = \{g_0, u_9, u_{10}, u_5, u_7\}$.

Then, as depicted in Fig. 5(e), we take out u_5 from Q. The neighbor nodes of u_5 are u_1, u_2 and u_8. Since u_1 and u_2 are both target UAVs, we directly insert them into STAT as the child nodes of u_5. Meanwhile, we update $U_{target}^5 = U_{target}^5 - \{u_1, u_2\} = \emptyset$ and $F = \{g_0, u_9, u_{10}, u_5, u_7, u_1, u_2\}$. Finally, we take out u_7 from Q. Since u_3 is the neighbor nodes of u_7, we insert it as the child node of u_7. Meanwhile, we update $U_{target}^7 = U_{target}^7 - \{u_3\} = \emptyset$ and $F = \{g_0, u_9, u_{10}, u_5, u_7, u_1, u_2, u_3\}$. Since $U_{target} \subseteq F$, the construction procedure of STAT is done.

Based on the constructed spatial-temporal aggregation tree, we can obtain the aggregated routing paths of all target UAVs.
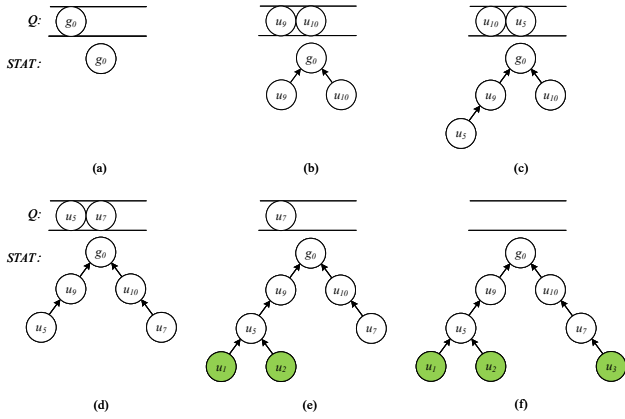
Figure 5: The construction procedure of the spatial-temporal aggregation tree.
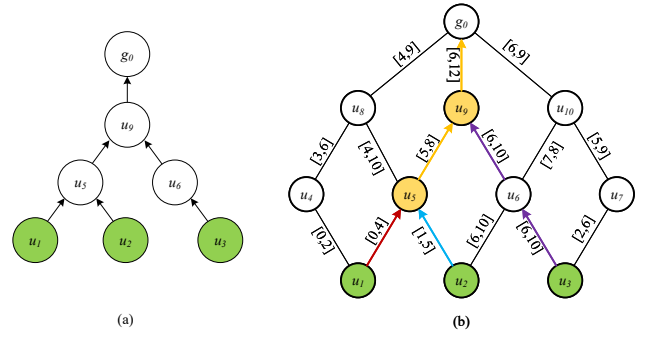

Figure 6: The generalized scenario expansion when $t_{nd} = 7.1s$. (a) The spatial-temporal aggregation tree. (b) The corresponding transmission scheme.

In STAT, the root node is the ground station, and except for the leaf nodes, each node will aggregate query results of all its child nodes, and then forward the aggregated result to its parent node. For example, as shown in Fig. 5(f), u_1 and u_2 send their respective query results to u_5, then u_5 aggregates these query results and forwards the aggregated result to u_9, and finally u_9 delivers it to g_0.

4.4.5. Generalized Scenario Expansion

In the previous section, we take the minimum delay required to complete the query task (i.e., the earliest delivery time of the last arrival packet) as the user query delay. As shown in the above example, $t_d = \max\{t_1, t_2, t_3\} = \max\{4.1s, 4.1s, 6.1s\} = 6.1s$. However, in many practical application scenarios, the requirement of the user query delay is not so tight, that is, the query delay constraint given by the user is typically greater than the minimum delay required for a query task. Therefore, in order to extend to more general scenarios, in this section, we introduce the slack variable $\zeta \geq 0$ to relax the user query delay, namely

$$t_{nd} = t_d + \zeta \qquad (3)$$

For example, let $\zeta = 1s$, then $t_{nd} = 6.1s + 1s = 7.1s$. According to Table 1, at this time, $U^8_{target} = \{u_1, u_2, u_3\}$, $U^9_{target} = \{u_1, u_2, u_3\}$ and $U^{10}_{target} = \{u_3\}$. Since $|U^8_{target}| = |U^9_{target}| = 3$, and $\max\{H(u_1, u_8), H(u_2, u_8), H(u_3, u_8)\} = \max\{2, 4, 4\} = 4$, $\max\{H(u_1, u_9), H(u_2, u_9), H(u_3, u_9)\} = \max\{2, 2, 2\} = 2$, the minimum forwarding set of g_0 is $N_0 = \{U^9_{target} = \{u_1, u_2, u_3\}\}$. Similarly, according to Algorithm 4, we can obtain the final spatial-temporal aggregation tree, as shown in Fig. 6(a), and the corresponding transmission scheme is shown in Fig. 6(b). The query results of u_1 and u_2 will be aggregated at u_5 first, and then further aggregated at u_9 with the query result of u_3. Finally, the aggregated query result will be delivered to g_0. The total energy consumption is $6E_m$, which is further reduced. By introducing the slack variable, we demonstrate that ESTA can work well with generalized scenarios.

5. Performance Evaluation

In this section, we provide a comprehensive experimental design and performance analysis of ESTA on the Opportunistic Network Environment (ONE) simulator [16].

5.1. Simulation Setup and Scenarios

Referring to the simulation scenarios in [3, 31, 37], we design a simulation scenario inspired by the forest monitoring missions. One stationary ground station is placed together with 21 search UAVs and 4 ferry UAVs. Each search UAV is responsible for a $200m \times 200m$ region and uses a typical zigzag movement pattern to efficiently cover the region, and each ferry UAV flies forth and back along the specified trajectory to assist search UAVs with message delivery. Note that the trajectories of all UAVs are pre-planned. Moreover, we use the energy model proposed in [43]. The experimental screenshot is shown in Fig. 7 and Table 3 summarizes the detailed experimental parameters.
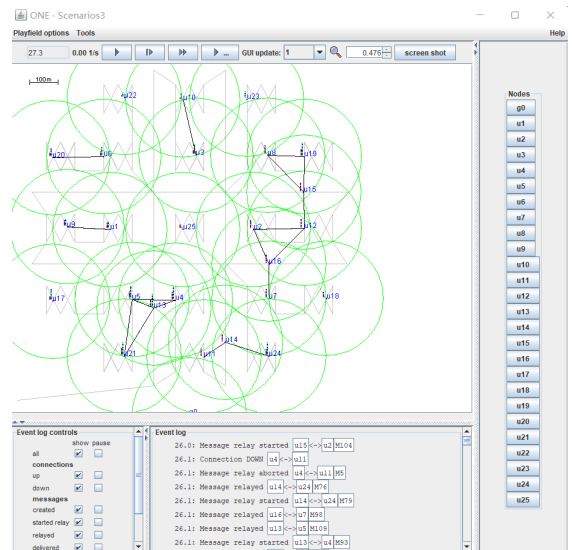

Figure 7: An experimental screenshot of the ONE.

10

Table 3: Simulation Settings

Parameter	Default value
Simulation Area (m^2)	1200×1300
Simulation Time (s)	1000
Number of UAVs	25
Mobility Model	MapRouteMovement
UAV Speed (m/s)	10
Communication Range (m)	200
Query Result Packet Size (KB)	10
Link Throughput (KB/s)	125
Query Time Period (s)	30
Query Region Ratio	10%
Query Number	1000

In this paper, we use the following metrics to evaluate the performance of the spatial-temporal range aggregation query algorithms.

- *Query delay.* The query delay represents the time it takes to complete a query task, that is, the time for all query results to be successfully delivered to the ground station.

- *Query Energy consumption.* The query energy consumption is the energy consumed by all query result packets to be successfully delivered to the ground station in a query task.

In addition, as far as we know, we are the first to research spatial-temporal range aggregation query processing in UAV networks. Therefore, we propose a Baseline Spatial-Temporal range Aggregation query processing (BSTA) algorithm and then compare it with ESTA. The BSTA consists of two main stages:

1. According to the constructed TCG and proposed SPA, we can obtain the shortest paths for query results of all target UAVs. Then, each target UAV delivers its query result along the shortest path to the ground station.

2. In the process of query result delivery, if a UAV stores and carries the query results of multiple target UAVs at the same time, it will aggregate these query results. Then the aggregated result will be forwarded to the ground station along the remaining shortest path.

For example, as shown in Fig. 1, target UAVs are u_1, u_2, u_3, and as mentioned above, the corresponding shortest paths are pa_1, pa_2, pa_3. According to the shortest paths, as shown in Fig. 4(a), UAV u_4 will receive the query results of u_1 and u_2 at $0.1s$ and $1.3s$. Then, u_4 will store and carry them until $3s$, and forward them to u_8 at $3s$. Therefore, u_4 aggregates the query results of u_1 and u_2, and then forwards the aggregated result to u_8 at $3s$. Through the way of piggybacking in-network aggregation, BSTA reduces energy and bandwidth consumption without sacrificing the query delay.
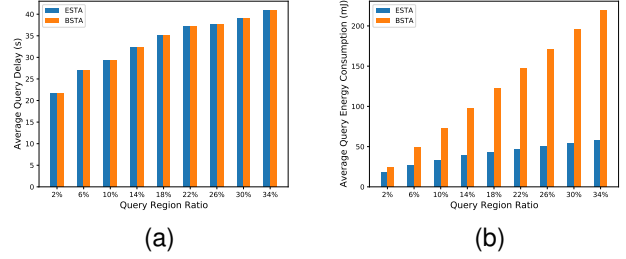

Figure 8: The impact of the query region ratio. (a) Average query delay. (b) Average query energy consumption.

5.2. The Impact of Different Variables

5.2.1. Impact of the query region ratio

As Fig. 8(a) shows, as the query region ratio increases, the query delay increases. This is because the query delay is the time it takes for all query results to be delivered to the ground station, which is essentially determined by the arrival time of the last query result reaching the ground station. When the query region ratio becomes larger, the expectation of the delivery delay of each query result remains unchanged, but the variance becomes larger, resulting in a larger query delay. On the other hand, it is obvious that the query delays of ESTA and BSTA are the same. In BSTA, the query result of each target UAV is delivered along its own shortest path, while in ESTA, the earliest delivery time of the last arrival query result is the delay constraint, so the delivery delays of the two algorithms are the same.

Furthermore, with the increase of the query region, the query energy consumption of both BSTA and ESTA increases, however, the increase rate of ESTA is much smaller than that of BSTA, which means that ESTA can save more energy than BSTA with the increase of the query region. As depicted in Fig. 8(b), when the query region ratio becomes 34%, the query energy consumption of ESTA is 26.2% of that of BSTA. This is because by constructing a spatio-temporal aggregation tree, ESTA makes an overall plan for the query results of all target UAVs and performs in-network aggregation as early as possible, which can effectively reduce the query energy consumption.

5.2.2. Impact of the query time period

As shown in Fig. 9(a), with the increase of the query time period, the average query delay of both ESTA and BSTA shows an overall upward trend. The reason is that the larger the query time period, the more target UAVs which store and carry the qualified data. Therefore, the increase of the query time period leads to the increase of the query delay, which is similar to the impact of the query region. However, the trajectories of UAVs are pre-planned, and UAVs cover the mission coverage region in an efficient pattern, rather than flying in a disorder way. Therefore, with the increase of the query time period, the number of target UAVs that satisfy the spatial-temporal constraint tends to be stable, and the query delay also tends to be stable.
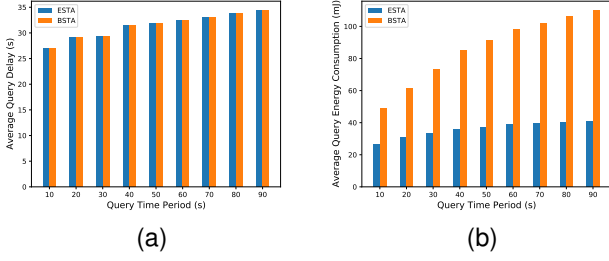
(a)　　　　　　(b)

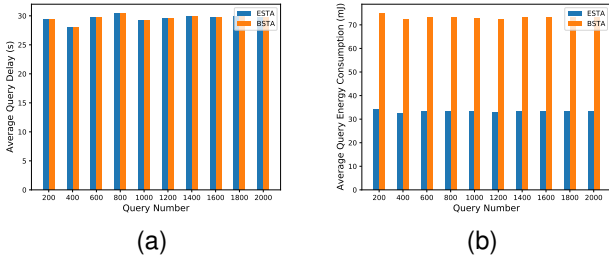Figure 9: The impact of the query time period. (a) Average query delay. (b) Average query energy consumption.

(a)　　　　　　(b)

Figure 10: The impact of the query number. (a) Average query delay. (b) Average query energy consumption.

(a)　　　　　　(b)

Figure 11: The impact of the UAV speed. (a) Average query delay. (b) Average query energy consumption.

(a)　　　　　　(b)

Figure 12: The impact of the communication range. (a) Average query delay. (b) Average query energy consumption.

Moreover, the query energy consumptions of both ESTA and BSTA increase with the query time period due to more query results which need to be delivered to the ground station. ESTA can efficiently reduce the energy consumption on the basis of ensuring query delay through the in-network aggregation, especially when the query time period is long. When the query time period is 90*s*, the query energy consumption of ESTA is only 37.2% of that of BSTA.

5.2.3. Impact of the query number

In this experiment, different numbers of query tasks are sent out at a constant speed within 500 seconds. As depicted in Fig. 10(a), the query delays of both ESTA and BSTA do not fluctuate significantly with the query number. This is because both ESTA and BSTA use pre-planned trajectory information to calculate routing paths in advance, so they can adapt to various UAV network environments.

At the same time, we can find that ESTA can effectively reduce the query energy consumption through in-network aggregation. As Fig. 10(b) shows, the query energy consumption of ESTA is stable at 45.2% of that of BSTA, demonstrating the efficiency and superiority of ESTA.

5.2.4. Impact of the UAV speed

As shown in Fig. 11(a), the average query delay of both ESTA and BSTA generally decreases as the UAV speed increases. The reason is that as the UAV speed increases, there are more communication opportunities between UAVs, and less storage-and-carry time required by UAVs, so that UAVs can deliver query results to the ground station with less delivery delay.
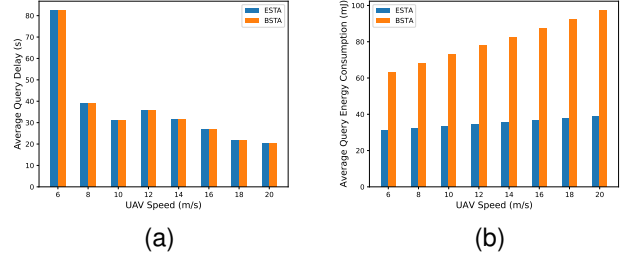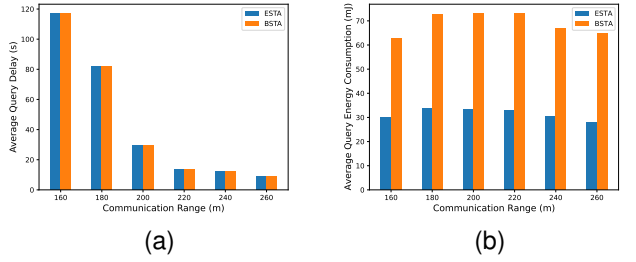
As shown in Fig. 11(b), the average query energy consumptions of both ESTA and BSTA increase slowly as the UAV speed increases. This is because under the same spatial-temporal constraint, due to the increase of the UAV speed, the number of target UAVs increases, resulting in an increase of the query energy consumption.

Meanwhile, it is worth noting that the average query energy consumption of ESTA is only 39.8% of that of BSTA. This is because ESTA further performs in-network aggregation on the basis of BSTA, thereby further reducing the query energy consumption.

5.2.5. Impact of the communication range

In this experiment, we investigate the effect of the communication range, which determines the sparsity and connectivity of the UAV network. As shown in Fig. 12(a), as the communication range increases, the query delay decreases rapidly. The reason is that with the increase of the communication range, both ESTA and BSTA can find the transmission paths which have earlier delivery time for query results, thereby reducing the query delay. Meanwhile, the query energy consumption of ESTA is about 45% of that of BSTA. Furthermore, the better the connectivity of the UAV network, the better the effect of the in-network aggregation.

5.2.6. Impact of the slack variable

As shown in Fig. 13(a), with the increase of the slack variable, the query delay of ESTA also increase gradually, because the introduction of the slack variable makes ESTA relax the requirement of the query delay constraint in order to further ag-
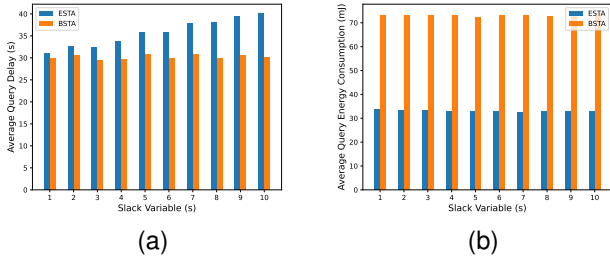
Figure 13: The impact of the slack variable. (a) Average query delay. (b) Average query energy consumption.

gregate query results within the network. Furthermore, in order to more intuitively reflect the impact of the slack variable on the query energy consumption, Fig. 14 shows the ratio of the energy consumption of ESTA to that of BSTA. It can be seen that with the increase of the slack variable, the ratio decreases, which indicates that the introduction of the slack variable can better optimize the spatial-temporal aggregation tree, so that ESTA can further reduce the energy consumption.
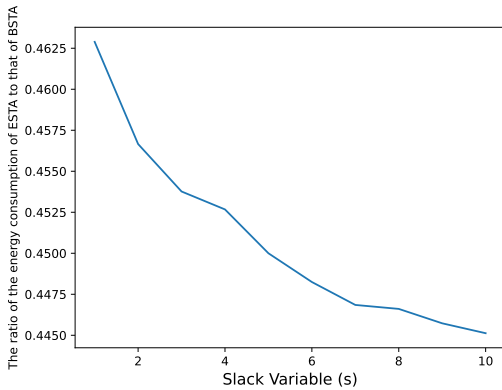

Figure 14: The impact of the slack variable on the ratio of the energy consumption of ESTA to that of BSTA.

6. Conclusion

In this paper, we study the spatial-temporal range aggregation query in UAV networks, and then propose an Efficient Spatial-Temporal range Aggregation query processing (ESTA) algorithm. First, ESTA utilizes the pre-planned trajectory information to construct a topology change graph (TCG) in order to reflect the communication windows between UAVs. Then, an efficient shortest path algorithm is proposed based on which the user query delay can be obtained. Next, ESTA transforms the aggregation processing of query results into recursively solving the set cover problem, thus constructing a spatial-temporal aggregation tree (STAT). With the constructed STAT, an efficient in-network aggregation routing path for query results without the sacrifice of the user query delay can be found. Extensive experimental results show that compared with the baseline spatial-temporal range aggregation query processing algorithm, ESTA

performs well in terms of the query delay and query energy consumption.

References

[1] D. Mishra, E. Natalizio, A survey on cellular-connected uavs: Design challenges, enabling 5g/b5g innovations, and experimental advancements, Computer Networks 182 (2020) 107451.

[2] A. Belfkih, C. Duvallet, B. Sadeg, A survey on wireless sensor network databases, Wireless Networks 25 (8) (2019) 4921–4946.

[3] X. Li, L. Liu, L. Wang, J. Xi, J. Peng, J. Meng, Trajectory-aware spatio-temporal range query processing for unmanned aerial vehicle networks, Computer Communications 178 (2021) 271–285.

[4] A. Chriki, H. Touati, H. Snoussi, F. Kamoun, Fanet: Communication, mobility models and security issues, Computer Networks 163 (2019) 106877.

[5] X. Cao, P. Yang, M. Alzenad, X. Xi, D. Wu, H. Yanikomeroglu, Airborne communication networks: A survey, IEEE Journal on Selected Areas in Communications 36 (9) (2018) 1907–1926.

[6] H. Dai, Y. Ji, F. Xiao, G. Yang, X. Yi, L. Chen, Privacy-preserving max/min query processing for wsn-as-a-service, in: 2019 IFIP Networking Conference (IFIP Networking), IEEE, 2019, pp. 1–9.

[7] Q. Chen, H. Gao, Z. Cai, L. Cheng, J. Li, Energy-collision aware data aggregation scheduling for energy harvesting sensor networks, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, IEEE, 2018, pp. 117–125.

[8] B. Pourghebleh, N. J. Navimipour, Data aggregation mechanisms in the internet of things: A systematic review of the literature and recommendations for future research, Journal of Network and Computer Applications 97 (2017) 23–34.

[9] E. Fitzgerald, M. Pióro, A. Tomaszwski, Energy-optimal data aggregation and dissemination for the internet of things, IEEE Internet of Things Journal 5 (2) (2018) 955–969.

[10] V.-T. Pham, T. N. Nguyen, B.-H. Liu, T. Lin, Minimizing latency for multiple-type data aggregation in wireless sensor networks, in: 2021 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2021, pp. 1–6.

[11] Y. Gao, X. Li, J. Li, Y. Gao, Distributed and efficient minimum-latency data aggregation scheduling for multichannel wireless sensor networks, IEEE Internet of Things Journal 6 (5) (2019) 8482–8495.

[12] S. A. Sert, A. Alchihabi, A. Yazici, A two-tier distributed fuzzy logic based protocol for efficient data aggregation in multihop wireless sensor networks, IEEE Transactions on Fuzzy Systems 26 (6) (2018) 3615–3629.

[13] E. Fitzgerald, M. Pióro, A. Tomaszwski, Energy-optimal data aggregation and dissemination for the internet of things, IEEE Internet of Things Journal 5 (2) (2018) 955–969.

[14] I. S. Amiri, J. Prakash, M. Balasaraswathi, V. Sivasankaran, T. Sundararajan, M. Hindia, V. Tilwari, K. Dimyati, O. Henry, Dabpr: a large-scale internet of things-based data aggregation back pressure routing for disaster management, Wireless Networks 26 (4) (2020) 2353–2374.

[15] Q. Xia, Z. Xu, W. Liang, S. Yu, S. Guo, A. Y. Zomaya, Efficient data placement and replication for qos-aware approximate query evaluation of big data analytics, IEEE Transactions on Parallel and Distributed Systems 30 (12) (2019) 2677–2691.

[16] A. Keränen, J. Ott, T. Kärkkäinen, The one simulator for dtn protocol evaluation, in: Proceedings of the 2nd international conference on simulation tools and techniques, 2009, pp. 1–10.

[17] S. Abbasian Dehkordi, K. Farajzadeh, J. Rezazadeh, R. Farahbakhsh, K. Sandrasegaran, M. Abbasian Dehkordi, A survey on data aggregation techniques in iot sensor networks, Wireless Networks 26 (2) (2020) 1243–1263.

[18] T.-D. Nguyen, D.-T. Le, V.-V. Vo, M. Kim, H. Choo, Fast sensory data aggregation in iot networks: Collision-resistant dynamic approach, IEEE Internet of Things Journal 8 (2) (2020) 766–777.

[19] B. Yin, X. Wei, Communication-efficient data aggregation tree construction for complex queries in iot applications, IEEE Internet of Things Journal 6 (2) (2018) 3352–3363.

[20] Z. Zhao, W. Yang, B. Wu, Flow aggregation through dynamic routing overlaps in software defined networks, Computer Networks 176 (2020) 107293.

[21] S. Bhandari, S. K. Sharma, X. Wang, Latency minimization in wireless iot using prioritized channel access and data aggregation, in: GLOBECOM 2017-2017 IEEE Global Communications Conference, IEEE, 2017, pp. 1–6.

[22] M. Amarlingam, P. K. Mishra, P. Rajalakshmi, S. S. Channappayya, C. S. Sastry, Novel light weight compressed data aggregation using sparse measurements for iot networks, Journal of Network and Computer Applications 121 (2018) 119–134.

[23] H. Ko, J. Lee, S. Pack, Cg-e2s2: Consistency-guaranteed and energy-efficient sleep scheduling algorithm with data aggregation for iot, Future Generation Computer Systems 92 (2019) 1093–1102.

[24] R. Maivizhi, P. Yogesh, Q-learning based routing for in-network aggregation in wireless sensor networks, Wireless Networks 27 (3) (2021) 2231–2250.

[25] L. Fu, X. Fu, Z. Zhang, Z. Xu, X. Wu, X. Wang, S. Lu, Joint optimization of multicast energy in delay-constrained mobile wireless networks, IEEE/ACM Transactions on Networking 26 (1) (2018) 633–646.

[26] M. Navarro, Y. Liang, X. Zhong, Energy-efficient and balanced routing in low-power wireless sensor networks for data collection, Ad Hoc Networks 127 (2022) 102766.

[27] H. Huang, H. Yin, G. Min, J. Zhang, Y. Wu, X. Zhang, Energy-aware dual-path geographic routing to bypass routing holes in wireless sensor networks, IEEE Transactions on Mobile Computing 17 (6) (2017) 1339–1352.

[28] R. Yarinezhad, S. Azizi, An energy-efficient routing protocol for the internet of things networks based on geographical location and link quality, Computer Networks 193 (2021) 108116.

[29] H. Huang, J. Zhang, X. Zhang, B. Yi, Q. Fan, F. Li, Emgr: Energy-efficient multicast geographic routing in wireless sensor networks, Computer Networks 129 (2017) 51–63.

[30] J. Liu, Q. Wang, C. He, K. Jaffrès-Runser, Y. Xu, Z. Li, Y. Xu, Qmr: Q-learning based multi-objective optimization routing protocol for flying ad hoc networks, Computer Communications 150 (2020) 304–316.

[31] M. Asadpour, K. A. Hummel, D. Giustiniano, S. Draskovic, Route or carry: Motion-driven packet forwarding in micro aerial vehicle networks, IEEE Transactions on Mobile Computing 16 (3) (2016) 843–856.

[32] K. Schneider, B. Zhang, L. Benmohamed, Hop-by-hop multipath routing: Choosing the right nexthop set, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications, IEEE, 2020, pp. 2273–2282.

[33] Z. Zhou, J. Feng, B. Gu, B. Ai, S. Mumtaz, J. Rodriguez, M. Guizani, When mobile crowd sensing meets uav: Energy-efficient task assignment and route planning, IEEE Transactions on Communications 66 (11) (2018) 5526–5538.

[34] G. Zhang, Q. Wu, M. Cui, R. Zhang, Securing uav communications via joint trajectory and power control, IEEE Transactions on Wireless Communications 18 (2) (2019) 1376–1389.

[35] J. Peng, H. Gao, L. Liu, N. Li, X. Xu, Tbm: An efficient trajectory-based multicast routing protocol for sparse uav networks, in: 2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), IEEE, 2020, pp. 867–872.

[36] J. P. Jeong, J. Kim, T. Hwang, F. Xu, S. Guo, Y. J. Gu, Q. Cao, M. Liu, T. He, Tpd: travel prediction-based data forwarding for light-traffic vehicular networks, Computer Networks 93 (2015) 166–182.

[37] J. Peng, H. Gao, L. Liu, Y. Wu, X. Xu, Fntar: A future network topology-aware routing protocol in uav networks, in: 2020 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2020, pp. 1–6.

[38] H. Song, L. Liu, B. Shang, S. Pudlewski, E. S. Bentley, Enhanced flooding-based routing protocol for swarm uav networks: Random network coding meets clustering, in: IEEE INFOCOM 2021-IEEE Conference on Computer Communications, IEEE, 2021, pp. 1–10.

[39] M. Anand, T. Sasikala, Efficient energy optimization in mobile ad hoc network (manet) using better-quality aodv protocol, Cluster Computing 22 (5) (2019) 12681–12687.

[40] M. A. Kachooei, F. Hendessi, B. S. Ghahfarokhi, M. Nozari, An olsr-based geocast routing protocol for vehicular ad hoc networks, Peer-to-Peer Networking and Applications 15 (1) (2022) 246–266.

[41] A. Abboud, R. Addanki, F. Grandoni, D. Panigrahi, B. Saha, Dynamic set cover: improved algorithms and lower bounds, in: Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, 2019, pp. 114–125.

[42] S. Bhattacharya, M. Henzinger, D. Nanongkai, A new deterministic algorithm for dynamic set cover, in: 2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS), IEEE, 2019, pp. 406–423.

[43] M. Zuniga, B. Krishnamachari, Analyzing the transitional region in low power wireless links, in: 2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004., IEEE, 2004, pp. 517–526.