future internet

MDPI

Article

Cluster-Based Data Aggregation in Flying Sensor Networks Enabled Internet of Things

Abdu Salam [1], Qaisar Javaid [2], Masood Ahmad [1], Ishtiaq Wahid [1] and Muhammad Yeasir Arafat [3,*]

[1] Department of Computer Science, Abdul Wali Khan University, Mardan 23200, Pakistan; abdusalam@awkum.edu.pk (A.S.); masood@awkum.edu.pk (M.A.); ishtiaqwahid@awkum.edu.pk (I.W.)
[2] Department of Computer Science and Software Engineering, International Islamic University, Islamabad 44000, Pakistan; qaisar@iiu.edu.pk
[3] IT Research Institute, Chosun University, 309 Pilmun-daero, Dong-gu, Gwangju 61452, Republic of Korea
* Correspondence: myarafat@chosun.ac.kr

Abstract: Multiple unmanned aerial vehicles (UAVs) are organized into clusters in a flying sensor network (FSNet) to achieve scalability and prolong the network lifetime. There are a variety of optimization schemes that can be adapted to determine the cluster head (CH) and to form stable and balanced clusters. Similarly, in FSNet, duplicated data may be transmitted to the CHs when multiple UAVs monitor activities in the vicinity where an event of interest occurs. The communication of duplicate data may consume more energy and bandwidth than computation for data aggregation. This paper proposes a honey-bee algorithm (HBA) to select the optimal CH set and form stable and balanced clusters. The modified HBA determines CHs based on the residual energy, UAV degree, and relative mobility. To transmit data, the UAV joins the nearest CH. The re-affiliation rate decreases with the proposed stable clustering procedure. Once the cluster is formed, ordinary UAVs transmit data to their UAVs-CH. An aggregation method based on dynamic programming is proposed to save energy consumption and bandwidth. The data aggregation procedure is applied at the cluster level to minimize communication and save bandwidth and energy. Simulation experiments validated the proposed scheme. The simulation results are compared with recent cluster-based data aggregation schemes. The results show that our proposed scheme outperforms state-of-the-art cluster-based data aggregation schemes in FSNet.

Keywords: clustering; data aggregation; dynamic programming; flying sensor network; internet of things

Citation: Salam, A.; Javaid, Q.; Ahmad, M.; Wahid, I.; Arafat, M.Y. Cluster-Based Data Aggregation in Flying Sensor Networks Enabled Internet of Things. *Future Internet* **2023**, *15*, 279. https://doi.org/10.3390/fi15080279

Academic Editor: Claude Chaudet

Received: 31 July 2023
Revised: 16 August 2023
Accepted: 18 August 2023
Published: 20 August 2023

1. Introduction

A multi-unmanned aerial vehicle (UAV)-aided flying sensor network (FSNet) is constrained by various energy factors, such as limited energy, computation, memory, and communication [1,2]. The energy consumption for sensing and computation is less than the energy used for communication among the UAVs or to the ground station (GS) [3,4]. The available energy resources are sometimes insufficient for transmission and computation during the mission. However, the collected data need to be communicated to the destination for further processing. The performance of the network lifetime depends on efficient energy utilization [5]. The researchers tried to minimize energy utilization in other wireless networks, but energy utilization still exists. Due to the flying speed of UAVs, the rapid variation in topology, terrain structure, and diverse directions make it difficult to collect and route information [6]. The researchers proposed energy-efficient schemes by considering different parameters such as reducing the communication distance, computation cost, mobility, and degree. However, data collection and minimization of communication load go unnoticed to save bandwidth and energy [7,8].

A data aggregation approach reduces the energy consumption of UAVs and increases their lifespan. The data aggregation approach is different in wireless sensor networks

(WSNs) and vehicular ad hoc networks (VANETs) from UAV networks [9]. In WSN, the uses of the data aggregation approach are for decreasing energy consumption rather than minimizing network capacity usage [10]. In VANET, due to the high variation in the topology, data aggregation is performed by many vehicles. In addition to the degree, mobility, density, and other parameters, the multi-UAV system is constrained by energy factors, which is why it is compatible with WSN and VANET requirements [11]. UAVs also consume energy by processing and storing more data, just like flight and communication [12].

UAV-based data aggregation has emerged as a promising solution for collecting data from remote and hard-to-reach areas. One of the main motivations for UAV-based data aggregation is the ability to obtain data more efficiently and accurately. UAVs can cover a larger area in a shorter amount of time compared to traditional data collection methods. This means that data can be collected faster, allowing for a quicker analysis and quicker decision-making. Additionally, UAV-based data aggregation can also reduce data collection costs. Traditional methods often require expensive equipment and personnel, whereas UAVs can be operated by a single person and require minimal equipment.

The fundamental operation in FSNet is data aggregation, which aims to transmit data among UAVs or to the GS. The data aggregation approach reduces communication costs and bandwidth utilization while obtaining aggregated data. Data aggregation utilizes the concept of many-to-few. A data aggregation protocol describes how the data are gathered, how they are routed to the destination, and when they should be transmitted.

Therefore, in this study, we developed a cluster-based data aggregation scheme for UAV-based FSNet. This research contributes the following:

- An effective mechanism designed based on a honey-bee algorithm (HBA) to select optimal unmanned aerial vehicles–cluster head (UAVs-CH).
- The formation of balanced and stable clusters reduces re-affiliation rates.
- Data aggregation algorithm proposed to limit duplicated data communication to the base station (BS).
- Avoids the transmission of unwanted packets to the BS and save FSNet bandwidth.
- Mathematical techniques measure the accuracy and correctness of the proposed scheme.

The remaining article is structured as follows: Section 2 elaborates on the existing literature. Section 3 discusses the proposed cluster formation and data aggregation schemes. In Section 4, experiment evaluation and simulation outcomes are outlined. In the Section 6, the paper concludes and states future directions.

2. Background

This section reviews the existing work on energy-efficient clustering and UAV-based data aggregation approaches. A UAV-based data aggregation network architecture differs significantly from a traditional wireless sensor network. For these reasons, existing algorithms for stationary or low-mobility WSNs are not feasible [13]. A data aggregation algorithm based on UAVs needs to be able to adapt to networks with high mobility, sudden changes in topology, and sporadic communication links. WSNs require different levels of quality of service, including delay, packet loss, and reliability on the underlying networks. Furthermore, WSNs are limited in terms of energy, computation power, and network resources. An appropriate data aggregation technique is essential for meeting requirements while respecting WSN limitations. WSNs require time synchronization to coordinate data, energy, and localization. To address the sensor time synchronization problem, the authors of [14] proposed a pairwise broadcast synchronization (PBS) protocol for multi-cluster sensor networks that reduces overall energy consumption while maintaining synchronization accuracy. Reference [15] proposed a distributed heuristic algorithm for selecting appropriate sensors in a multi-hop sensor network to reduce the number of message exchanges needed for network-wide synchronization using PBS.

2.1. Energy Efficient Clustering

Energy-efficient clustering strategies for FSNet are still in their infancy [16]. This section presents an overview of some existing energy-efficient solutions based on different parameters, criteria, and approaches. FSNet has many military and civil applications. Nevertheless, the main issues of UAVs in FSNet are limited energy, high mobility, frequent topology changes, and terrain structure, due to the limited flight time of UAVs and lack of routing efficiency [17]. Aadil et al. [18] addressed these issues and proposed a clustering model with energy-aware link-based clustering called EALC. For efficient clustering, parameters such as energy, degree, cluster formation, and cluster head lifetime are taken into account. Link quality and transmission range are considered first to reduce the energy consumption of flying nodes. The K-means density-clustering model used for high mobility considers distance and energy. This model selects cluster heads (CHs) with the least computation and maximum throughput and minimizes routing overhead. The lifetime of UAVs increased with the use of an energy-efficient selection of CHs. A simple clustering approach is used to reduce computation overhead. Nevertheless, EALC ignored the communication load, data aggregation, and bandwidth utilization factors.

Arafat and Moh [19] proposed an energy-efficient clustering scheme based on particle swarm optimization for an emergency mission. First, implement swarm-intelligence-based localization and clustering (SIL and SIC) schemes that define the search space with a boundary box to reduce computation power. UAVs are placed randomly in the search space. SIL uses a grouping scheme to calculate the distance between the target UAVs. The proposed model has an estimation model to measure UAV distance from CH. The distance between nodes and CH is considered for reducing energy consumption. The Euclidean distance is utilized for locating UAVs and balancing cluster size. In SIC, cluster formation and CH selection are performed with a fitness function that considers remaining energy and distance. The performance metrics used in this paper are energy consumption, communication load, and delay. Node degree and residual energy are considered to balance inter-cluster and intra-cluster energy utilization.

Yang et al. [20] proposed a probabilistic energy-aware clustering scheme to find the most efficient path for UAVs using ant colony optimization (ACO). WSN data-gathering efficiency is achieved with UAVs. The network is divided into clusters, each with a CH, cluster members (CMs), and UAV. The nodes are stationary with a position-aware CH; the CH receives information from the UAVs flying around the cluster heads. The proposed model has three stages. First, a UAV senses data about the farmland event via the ground segment. Second, the data gathered by UAVs is transmitted to the data center through the CH. Finally, the data center contains a database and management information system.

To overcome packet loss, long delay, and increased routing overhead, Yu et al. [21] proposed a clustering protocol based on ACO to enhance network performance. A reliable link supervision method was established due to UAV mobility and topology variation. The nonlinear processing scheme increases buffer size and link load to control and avoid congestion on the link. Based on the density of UAVs, two routing strategies are proposed, namely sparse formation and concentrated formation. ACO-based polymorphism-aware routing integrates dynamic source routing and ACO.

2.2. UAVs-Based Data Aggregation

In FSNets, the collection and transmission of information through multiple hops increase energy utilization [22]. The data aggregation approach reduces energy utilization and increases network lifetime by minimizing UAVs load. The data aggregation approaches reduce the communication cost and energy consumption [23]. The researchers developed aggregation approaches for FSNet without redundant data elimination [24].

Wang et al. [25] introduced a UAV-assisted topology-aware data aggregation protocol in WSN (TA-UAV-DA). The data aggregation approach was inspired by the compressive sensing approach to reduce the errors rate in the data reconstruction process, extra overhead, and energy consumption. Balanced tree-based topology construction is performed to

minimize the scope and update the matrix measurement. The CMs send data to CH, and UAV gathers data from the CHs. The simulation results show that the approach performs better when reconstructing data, and it is more efficient in data aggregation and storage constraints than a random walk and intelligent compressive sensing.

Wu et al. [26] developed an energy-efficient UAV-based data aggregation protocol (EE-UAV-DA) for WSNs. In EE-UAV-DA scheme, UAVs gather data as data mules. The proposed method calculates the optimum link for the data mule through all CHs, using a genetic algorithm that achieves high-energy efficiency. By balancing system throughput, the proposed approach reduces the delay between the sensors and the sink node. The optimization scheme provided by heuristic search identifies optimal solutions for joint CH selection and optimal routes for the data mule to decrease energy consumption. The objective function calculates and measures each solution's quality for the optimum path.

Thammawichai [27] proposed optimizing communication and computation for multi-UAV information-gathering applications called OC-mUAV. Multi-hop clustering incorporates data aggregation by using a mixed-integer optimization formulation with mixed-integer nonlinear programming. In order to determine the roles of UAVs, the optimal control problem was formulated. The system framework tries to ensure that the optimal number of UAVs communicate with BS. To maintain minimum energy consumption during routing, each UAV acts as an aggregator. An adaptive energy consumption model minimizes energy consumption by considering sensing energy, aggregation energy, transmitting energy, and receiving multi-UAV energy. To reduce communication and communication energy, area mapping and target tracking were addressed during testing. Target and sensor models are used to select the sensor UAV based on a subset of UAVs. The distance between UAVs is used for mapping. As a result of the data aggregation framework, the network is flexible and reliable since it is a self-organized network, which prolongs the lifetime of the network due to multi-hop networks and provides better performance due to clustering of heterogeneous UAVs.

Dong et al. [28] proposed an algorithm to collect and process data in WSN, using UAVs and mobile agents (MAs) to search for victims at disaster sites. MAs move around the area to collect data from sensor nodes and share information with UAVs. The UAV assigns MAs to group leaders. The density of sensor nodes in a group known by the group leader has high residual energy and is an optimum link to a UAV. MAs' routing is based on information-driven static and dynamic mobile agent planning algorithms. The proposed scheme is efficient in energy and time for any dense network using MAs and UAVs.

To decrease energy utilization in a UAV-aided WSN, Liu and Zhu [29] proposed an energy-efficient data collection method. Sensor nodes are placed randomly in the environment. The proposed approach uses three transmission modes to solve the short buffer size of sensor nodes to transmit data within the allotted time slots. The sensor node selects the modes, i.e., waiting, transmission to a sink node, and uploading to UAV in each discrete time slot. The sensor node is selected in waiting mode to sleep and not transmit the status of the node. The sensor node uploads data to the sink node in the second mode. In the third mode, the sensor node delivers data to the UAV based on the threshold value and distance condition during the UAV preplanned trajectory visit. The UAV of a fixed-wing aircraft is deployed at a constant velocity. This article uses a finite-horizon sequential Markov process and dynamic programming algorithm for the optimized transmission policy. Secondly, the proposed method optimizes the preplanned trajectory for UAVs using a recursive random search algorithm.

The authors of [30] analyzed the performance of an energy-constrained Internet of Things (IoT) system that uses a power beacon and UAV for data collection. The study examined how different system and channel parameters affect outage probability, outage capacity, and ergodic capacity. In [31], the authors explored the challenges and possible solutions for implementing a fully immersive and interactive industrial metaverse, which is a virtual space that interacts with the physical world in real time. In the paper, the authors focused on improving key performance indicators, such as the Age of Information, latency,

and reliability through optimizing short-packet structures in 6G URLLC communication. A cooperative strategy involving an unmanned ground vehicle and UAV is proposed in [32] to collect data from sensor nodes (SNs) in UAV-enabled data collection systems when SNs may not be able to upload their data because of factors such as insufficient energy and low flight altitude. A collaborative strategy selection algorithm that combines multistage-based SN association and UAV-UGV path optimization algorithms was used to determine trajectories for mobile data collection nodes on the ground and in the air to minimize mission completion time.

3. Flying Sensor Network Cluster Optimization

In cluster-based flying networks, the selection of CHs and cluster formation requires special attention to decrease the re-affiliation rate and save FSNet resources [33]. To select CHs, essential parameters such as the remaining energy, mobility, and flying nodes degree are considered to obtain optimal clusters [34–36]. These parameters are optimized to distribute load among clusters. We use the clustering approach to balance and select CHs in accordance with [37], as shown in Figure 1. Data aggregation is initiated once clusters are formed.

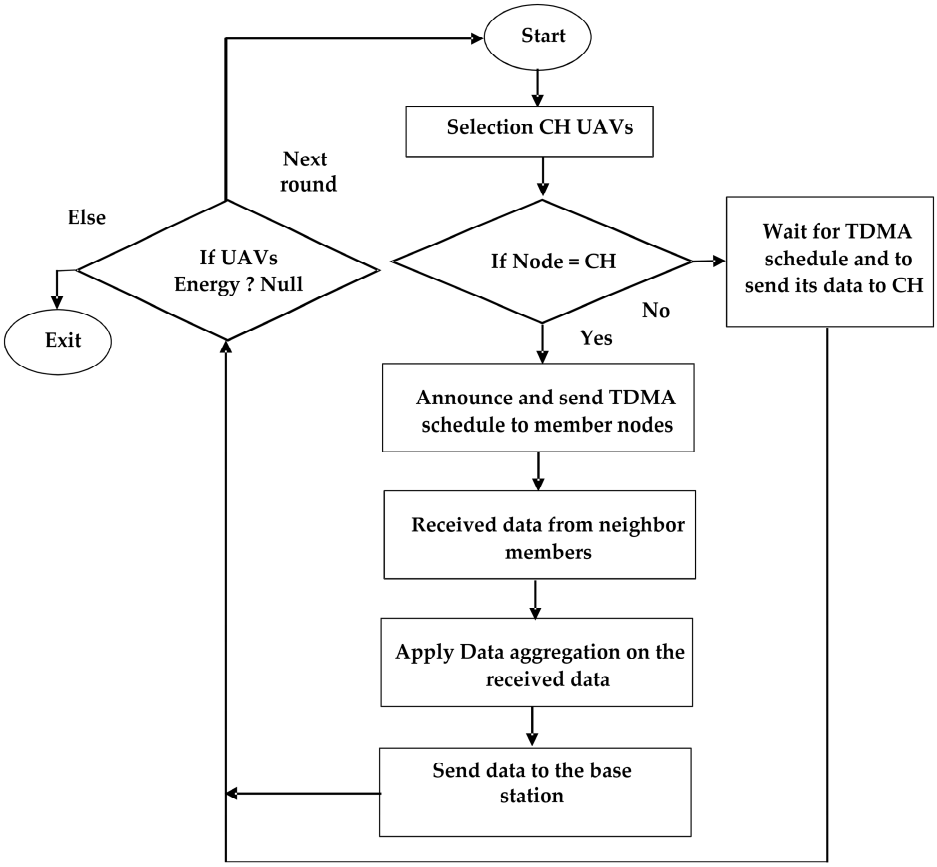

Figure 1. Flowchart of proposed system.

As shown in Figure 1, the proposed approach begins with the selection of a CH-UAV. The selection process determines which UAV will act as the CH. If the node is the CH, it proceeds to the next step. Then, the CH-UAV broadcasts the time division multiple access (TDMA) schedules to all member nodes within its cluster. TDMA is a channel access method that allows multiple nodes to share the same communication channel by dividing it into different time slots. The CH-UAV receives data from all neighboring nodes within its cluster. Data aggregation is the process of combining or summarizing data from multiple sources. The CH-UAV performs data aggregation on the received data, reducing

redundancy and improving efficiency. After data aggregation, the CH-UAV sends the aggregated data to the BS for further processing. If the current node is not CH, it proceeds to the next step. The non-CH node waits for the TDMA schedule broadcast by the CH-UAV. This schedule determines when the node can transmit or receive data. The algorithm checks the energy level of nodes. If the energy is still available, it returns to the CH selection step, indicating that the next round of the process will begin. When the node's energy reaches zero or depletes, the flowchart terminates, and the process ends.

In cluster-based routing for flying sensor networks, CHs may receive multiple copies of the same dataset from different sensors located in the vicinity. The communication of data requires more resources as compared to computation. Hence, a method is required to discard duplicated packets when sending data to the base station at the cluster level. Thus, network resources such as batteries and bandwidth can be utilized for other purposes. The network lifetime will increase. Our proposed algorithm works in two phases: cluster setup and data aggregation.

Clustering Setup

The HBA is applied in the cluster setup phase to determine the optimal CHs [37]. The CH selection is based on the HBA to form a balanced cluster. When selecting CHs, the UAV mobility and neighbor criteria are considered to minimize re-clustering. In FSNet, once CHs are selected, they broadcast a message containing ID, position, and status. All UAVs in CH range will receive broadcast messages and join the cluster. Once UAVs join a cluster, they become CMs and share information with CH. If a UAV receives membership messages from multiple CH, joining will be based on the distance between the UAV and CH. If the distance is the same, the random UAVS-CH selection mechanism will take place. The working of the cluster setup phase is shown in Algorithm 1 below. Once the cluster is formed, the data-aggregation-and-communication phase is initiated to transmit the data to the BS.

Algorithm 1: Pseudo Code of UAVs Enabled CH Selection.

Procedure CH-Selection-Multi-UAVs (MUAVs)()
Input : Swarm of UAVs SW_{UAV}, UAV nectar $[n_{UAV}]$, and cluster C_{FSNet}.
Output: UAVs-CH
call function calculate-UAVs-Nectar (n_{UAV})
// v1 represents the number of nodes (UAVs) when there are n total UAVs in the network
for$(v = 1; v \leq C_{FSNet}; v++)$ do
// selection of UAVs-CH in a random way
UAVs-CH[v] = functionRand (SW_{UAV})
end for
while (highest-value! = yes) do
for$(v1 = 1; v1 \leq n_{UAV}; v1 = v1 + 1)$ do
// the suitability of current selection is computed
if ($v1$ in UAVs-CH) then
Fitness Value $FValue_{UAV} = FValue_{UAV} + 1(SW_{UAV}[u] + AFV_{UAV})$
// Average fitness value AFV_{UAV}
end if
end for
if $(FValue_{UAV} < PFValue_{UAV})$ then
// $PFValue_{UAV}$ is the suitable value in the existing solution
swap $FValue_{UAV}$
end if
if (UAVs-CH-optimum! = yes) then
while$(empb! = 0)$ do // Employed bee (empb)
// visiting of bees employed till empty, where α_{ij} is UAV affiliation with the current round while y is the neighborhood size
$UAV_j(x+1) = UAV_j(x) + \alpha_{ij} * y$
// selection of different UAVs from fellow citizen

Algorithm 1: *Cont.*

end while
$Pr_i = \frac{W_{UAV_i}}{\sum_{j=1}^{k} W_{UAV_j}}$
// the new UAVs probability Pr_i will be calculated based on Weight of UAV (W_{UAV})
while (the Obees $\neq \in$) do //Onlooker bees (Obees)
Selection of another set of UAVs-CH will be carried out subject to the probability Pr_i
end while
Else
return UAVs-CH
end while
end procedure

4. Data Aggregation and Communication

We collect data from flying UAVs and match the data to discard similar data sent by multiple sensors in the data aggregation process. Data aggregation is divided into two levels. In level 1, the data are collected using a TDMA schedule when multiple sensors simultaneously communicate data.

A near-linear time algorithm is proposed in this paper for the data aggregation level 2 problem in FSNet. The data coming from sensors is converted into a long string. According to our knowledge, this is the leading work to assume nontrivial alignment among the strings and the patterns. Specifically, we demonstrate the data aggregation problem in two ways: First, to sanction approximating $D[i]$'s, and second, an additional procedure sanction named partial data move, for the movement of partial data from one position to another in a data.

This similarity among the X and Y data is known as data match with moves (DMM) and designated as $d(X, Y)$ [38]. DMM is a powerful data-matching tool that can greatly benefit FSNet data aggregation. By using DMM, network operators can easily match data from different sources, allowing for a more accurate and comprehensive analysis of the data. The DMM algorithm can take advantage of UAV mobility by assigning them to different regions and optimizing their movement patterns to efficiently collect data. Moreover, DMM can provide a wide range of benefits, including reducing the amount of data, reducing bandwidth usage, increasing energy efficiency, and supporting proximity-based data aggregation. There are various applications in computational biology where partial data matches are considered a primeval in multiple situations. Moving a larger subsequence is similar to the insert or delete operation; during text processing, moving a large array together might be assumed, like reordering to deleting or inserting typescripts. Keep in mind that the nontrivial placements are still a challenge for DMM. Hence, $d(X, Y)$ is the size of the small structure of edit procedures that convert Y to X; the allowable procedure affects the data stated. The deletion of a character at location, loc, transforms X to $X[1] \dots X[loc-1], X[loc+1] \dots X[n]$.

- The insertion of a character, "c" at a location, "loc", gives $X[1] \dots X[loc-1], c, X[loc] \dots X[n]$.
- The substitution of a character at the location, "loc", with character, "c", results in $X[1] \dots X[loc-1], c, X[loc+1] \dots X[n]$.
- The partial data movement with factors $1 \leq loc \leq loc_2 \leq k \leq n$ converts $X[1] \dots X[n]$ into $X[1] \dots X[loc-1], X[loc_2] \dots X[loc_3-1], X[loc] \dots X[loc_2-1], X[loc_3] \dots X[n]$.

The data are identical when the edit distance between two data is "0". The metric represents its measure. The transformation is performed in several operations, and each operation's cost is equal even in the inverse case. Hence, $d(X, Y) = d(Y, X)$; then, every distance resulting from transforming one datum to another must follow the triangular inequity. The restrictions of the interaction of edit operations are none. These restrictions may be as follows: it is relatively conceivable for a fractional data move to take a fractional

datum to a different position and then for a successive data move to function on a fractional datum that overlays the relocated fractional datum and its neighboring typescripts.

The deterministic algorithm results in the DMM problem, and the running time complexity is $O(nlogn)$. An algorithm returns the equivalent array, "Ar", where every $Ar[loc]$ is estimated to be close to the $O(lognlog^*n)$ factor. The proposed methodology depends on inserting data to a vector of arrays below $L1$ metric. The $L1$ size among these arrays is $O(lognlog^*n)$, the estimate of the DMM between the two original data.

The proposed method can further solve several problems beyond the primary data aggregation Level 2 problem. These contain data-similarity search problems. A study presented in [39] showed that calculating the distance among two datasets is NP-Complete. An approximation algorithm with complexity $O(logn)$ was presented to find the distance among datasets. However, the approximation may not resolve the data-match problem.

The proposed scheme focuses on the vital components. Firstly, we parsed data into a hierarchy of partial data. We use a simple hierarchical mechanism for parsing called edit-sensitive parsing (ESP), which generates a tree with three degrees [40]. ESP may not be an innovative parsing method; however, it is an effort to make straightforward the procedural details of relating predefined coin throwing to obtain classified data fragmentation. It is expected that the ease of ESP assists in/exposes more uses of classified data decays. The next module of this research is the approximate distance preservative data inserting to array spaces based on hierarchical parsing.

4.1. Data Embedding

We demonstrate a data-embedding scheme, which embeds data into a multidimensional matrix. Assume some data, X, over an alphabet, $\sum$. The data, X, will be embedded as $Em(X)$, an array with multi-magnitudes, $O(|\sum||X|)$, but the number of magnitudes of the nonzero array will be relatively minimum, indeed $O(|X|)$.

The time complexity of the embedding Em process is linear. The proposed scheme will parse X into different fractional data and reflect the multi-set $T(X)$ of these fractional data. We confirm that the size of $T(X)$ is as a maximum $2|X|$. Hence, $Em(X)$ is the distinguishing array for $T(X)$. The process through which the parse tree $T(X)$ will generate is known as ESP. The following subsection explores the ESP.

4.1.1. EPS

The parse tree, $EST(X)$, that is formed for data $X : X$ is the breakdown structure of partial data analogous to the nodes of $EST(X)$. The aim is to limit the data-editing operations. A clear EST contains all dynamic data of length 2^{loc}, namely $X\left[loc_2 2^{loc} \ldots \left((loc_2+1)2^{loc}-1\right)\right] \in loc$ and loc_2; it results in a complete binary tree. However, if X is updated using addition or removal to obtain X', X and X' will be construed using the same technique to two dissimilar hierarchical partial datasets; thus, the resultant embedding will not preserve approximation.

Suppose we have data X; the next step is to form an ESP tree in a hierarchal fashion with $P_i(X)$ repetitions. Every repetition produces a different level of EST. At every repetition, i, the process initiates with data, Xi, and divides them into chunks of size two or three. We substitute every chunk analogous to $X[loc_3 \ldots loc_2]$ by name and refer to the pair $(loc, h(X[loc_3 \ldots loc_2]))$. Moreover, h corresponds to a 1-1 hash function on partial data X.

Suppose $X0 = X$, and the repetitions until the length of data left become 1. The EST tree of X contains levels, and for every string of $Xloc-1$, a node at level i must exist, and the children are the nodes in level $loc-1$. Here, a leaf node is each data unit of $X0 = X$. More precisely, we can make the partitions of Si data into dissimilar non-overlapping units. The data can be divided into non-overlapping units in three ways:

(a) Maximum adjacent partial data of Xi that comprise a repetitive sign ($X0$ shows in the form a^l for $a \in \Sigma_i$ where $l > 1$).
(b) Length of partial data (Long) at least $log^*|\sum loc-1|$ not of type 1 above.
(c) Length of partial data (short) less than $log^*|\sum loc-1|$ not of type 1.

The general term "meta block" (mb) is used for all such partial data. To produce next-level parsing, we process each mb as demonstrated in the following subsections.

4.1.2. Type 2: Long Data without Duplications

We assume a dataset where two consecutive symbols are duplicates and represent an mb of type 2. Assuming a structure, X, without duplications (i.e., $X[loc]6 = X[loc + 1]$ for $loc = 1 \ldots |X| - 1$), we select as a maximum $|X|/2$ and minimum $|X|/3$ partial data of X as a node. We obtain X upon concatenating these nodes. The first phase comprises repeating the reduction process of an alphabet.

Reduction of alphabet ($\sum$): For every character $C[loc]$, calculate a new tag. $C[loc - 1]$ is the symbol located to the left of $C[loc]$, and assume $C[loc]$ and $C[loc - 1]$ are denoted as binary numbers. The least-significant bit (LSB) key where $C[loc]$ differs from $C[loc - 1]$ is represented by Le, and assume bit $(Le, C[i])$ is the value of $C[i]$ at the Le^{th} bit position. For example, the location of bit Le is next to the character at the former index, i.e., form label $(C[loc])$ as $Le + bit(Le, C[loc])$.

Lemma 1. *For some loc, if* $C[loc] \neq C[loc + 1]$, *then* $C[loc] \neq label(C[loc + 1])$.

Proof. Supposing the LSB location, where $C[loc]$ differs from $C[loc + 1]$, is similar in a way where $C[loc]$ also differs from $C[loc - 1]$ (else, $labelC[loc] \neq label(C[loc + 1])$). However, the bit character at this position in every symbol must be different; therefore, $labelC[loc] \neq label(C[loc + 1])$.

Adopting this method, we create an innovative series. If the existing alphabets have length, η, then the extracted alphabets have size, $2log|\eta|$. We currently repeat (repetition is orthogonal to the duplication that produces an EST tree of X; repeating on C that is a subseries with no matching contiguous characters) and make the character decrease until the length of the alphabet is unable to shrink. This will take $log^*|\eta|$ repetitions. Note that the labels for the first $log^*|\eta|$ symbols will not exist.

The lemma states that if a symbol at position $C[loc]$ is not equal to the symbol at position $C[loc + 1]$, then the label assigned to $C[loc]$ is not equal to the label assigned to $C[loc + 1]$. Lemma 1's goal is to establish a connection between symbols and their corresponding labels based on how they differ from one another. Lemma 1's proof demonstrates that if there is a difference between two symbols, ($C[loc]$ and $C[loc + 1]$), their labels will also be different. The subsequent actions and procedures in the text are theoretically justified by this lemma. □

Lemma 2. *After the last reiteration of the* $\sum$*reduction, the size of* $\sum$*is 6.*η.

Proof. Upon every repetition of cataloguing, the size of the alphabet is reduced from $|\sum|$ to $2[log|\sum|]$. If $|\sum| > 6$, then $2[log|\sum|]$ is firmly smaller than 6. A has no duplicate symbols contiguously, nor do the final order of tags on A by Lemma 1 iteratively.

Lemma 2 plays a crucial role in establishing the size of the alphabet ($\sum$) after the last iteration of cataloguing in the given text. It states that the size of $\sum$ is 6 times the original alphabet size (η) after the last repetition of cataloguing. The proof of Lemma 2 demonstrates that if the initial alphabet size is greater than 6, then the reduction process reduces the alphabet size to a value that is strictly smaller than 6.

Lastly, three passes over the order of labels were accomplished to decrease the alphabet from {0, 1, 2}: initially, we substitute every 3 with the minimum item from {0, 1, 2} that is not in the neighborhood of 3, and then we perform the same operation for every 4 and 5. This produces a series of symbols extracted from the $\sum\{0, 1, 2\}$, where no contiguous labels are equal. We designate this series A'.

We currently choose distinct positions, known as landmarks, from the structures that are closely related to each other. We initially chose some location, I, and a local maximum as a landmark, such as $A'[loc - 1] < A'[loc] > A'[loc + 1]$. □

Two local maximums might have four overriding symbols. Moreover, we chose some i as a landmark, i.e., local minima, such as $A'[loc-1] < A'[loc] < A'[loc+1]$, that was not contiguous to a previously selected landmark. In Figures 2 and 3, the process is depicted graphically.

I	**Text**	G	a	b	D	a	e	h	e	g	A	B	a	c
ii	**in binary**	110	000	001	011	000	100	111	100	110	000	001	000	010
iii	**labels**	-	011	000	010	001	100	000	001	010	011	000	001	010
iv	**labels as integers**	-	3	0	2	1	4	0	1	2	3	0	1	2
v	**final labels**	-	1	0	2	1	2	0	1	2	1	0	1	2

Figure 2. Landmark finding and alphabet reduction process.

Figure 3. Nodes' formation based on landmark symbols.

Lemma 3. *For some two consecutive landmark locations, loc and* $loc_2, 2 \leq |loc - loc_2| \leq 3$.

Proof. Using our tagging mechanism, we claim that no contiguous pair of symbols is tagged—subsequently, we may not have two contiguous local maximums and specifically inhibit tagging local minima next to a local minimum. A modest case investigation demonstrates that the parting of landmark locations is two overriding labels. □

Lemma 4. *Defining the nearby landmark to location, loc, subject to only 5 adjacent locations to the right and* $log^*|\eta| + 5$ *to the left.*

Proof. Once a reiteration of alphabet reduction is performed, every symbol depends merely on its left label. We iterate this $log^*|\eta|$ time; therefore, the symbol at a location, i, is based on $log^*|\eta|$ for the left labels. As soon as we accomplish the last step of reducing the $\sum$ form $\sum$ of six (6) to $\sum$ of three (3), the last label at a location, i, is based on a max of three extra labels to its right and left. We must tag every local maximum location and, formerly, every local minimum position not contiguous to a local maximum; therefore, we must inspect max two symbols to the right of i and two labels to the left, which, in order, depends on three labels to the right and $log^*|\eta| + 3$ labels to the left. The aggregate dependence is, hence, as indicated. We currently demonstrate how to divide A into units of size 3 or 2 nearby landmarks. □

4.1.3. Type 1 (Repeating mbs) and Type 3 (Short mbs)

We look for landmarks discovered without any difficulty in the local neighborhood. Hence, we assume data blocks containing a distinct repetitive symbol as big landmarks. Type 3 and Type 1 blocks are parsed in a usual way; the details are given for completeness. The mbs having size one are attached to the left or the right of the repeating mb. The attachment will preferably be to the left if both are possible. The mbs of size two or three are remembered as blocks without further splitting, whereas an mb of length four is allocated into two blocks of size two. In each mb with a size of five or greater, the parsing may be carried out on the leftmost three labels as a block, and then reiterate the rest.

4.1.4. Constructing ET (X)

While partitioning $Xloc$ into 2 or 3 labels, $Xloc + 1$ is constructed by substituting every block, bl, by $hf(bl)$, where hf is a 1-1 naming hash function. It is essential to mention that

the units of distinct levels will call hash functions onto distinct domains to compute names. To this point, the focus is on any given level i. Using the random property, the computation of $hf()$ for a specific unit of size 2 or 3 will take $O(1)$ time. This produces the series $Xloc + 1$; this process is reiterated till the series is of size one or till the tree root. Let $P_i(X)$ represent total nodes at level i in $ET(X)$. The original data, $M_0(X) = |X|$, are used to design the first (leaf) level from the symbols. We have $P_i(X)/3 \leq P_{i+1}(X) \leq [P_i(X)/2]$. Hence, $\frac{3}{2}|X| \leq P_i|X_i| \leq 2|X|$. Therefore, for every i, $|\sum_i| \leq |X|$ and, thus, $log^*|\sum_i| \leq log^*|X|$ are shown in Figure 4.

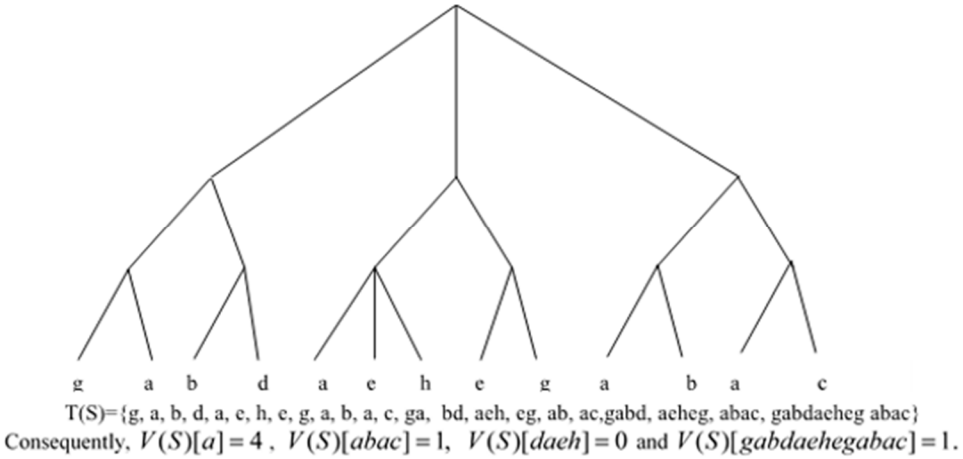

Figure 4. Representation of node structure hierarchically in parse tree based on string S.

Theorem 1. *Given the data, X, the $ET(X)$ can be calculated in $O(|X|\ log^*|X|)$ time.*

4.1.5. Properties of ESP

How to calculate the $ET(X)$ for some data X is defined above. Every node n in $ET(X)$ denotes the partial data of X obtained from the leaf nodes on the merging in the subtree at the root node n.

Definition 1. *Describe the multi-set $T(X)$ such that all partial data of X are designated by the nodes of $ET(X)$ (for each level). The characteristic array $A(X)$ of $T(X)$, i.e., $A(X)|n|$, can be defined as the sum of occurrences a partial data n appears in $T(X)$. In conclusion, $A_i(X)$ the features array limited to those nodes appear at a level i in $ET(X)$.*

$T(X)$ contains as a maximum $2|X|$ data having size $|X|$ An (X) is $O(|\sum| + |X|)$ dimensional array as its dominion is some data that exist in $T(X)$.

The typical L_1 distance between two arrays a_1 & a_2 by $||a_1 - a_2||1$ is specified. $|A(X) - A(Y)|1 = P_{X \in T(X) \cup T(Y)}|A(X)[x] - A(Y)[x]|$. Recall that $d(X, Y)$ indicates the match with moves among data Y and X.

Theorem 2. *For data Y & X, suppose n be maximum $(|Y|, |X|)$. Then $|A(Y) - A(X)|1 = O(lognlog^*n)d(X, Y)$.*

4.2. Upper Bound Proof $|A(Y) - A(X)|1 = O(lognlog^*n)d(X, Y)$

Proof. Express this bound on $L1$ space, assume the impact of the edit procedures, and show that all adds a role to the $L1$ distance limited by $O(lognlog^*n)$. Editing processes are permitted to overlay on blocks. We present a Lemma applicable to all data, except with no contiguous repetitive alphabets. □

Lemma 5. *The nearby landmark to any character of X_{loc}* is computed by at most five successive characters of X_{loc} *to the right and at most $log^*|\Sigma_{loc}| + 5$ repeated characters of X_i to the left.*

Proof. Assume that a character of X_{loc}, i.e., $X_{loc}[loc_2]$, indicates the mechanism to identify the nearby landmark.

Type 1 Iterating mb: A lengthy duplication of a character, c, is considered a distinct and larger landmark. $X_{loc}[loc_2]$ is contained within such an mb if $X_{loc}[loc_2] = X_{loc}[loc_2 + 1]$ or if $X_{loc}[loc_2] = X_{loc}[loc_2 - 1]$. We also assume $X_{loc}[loc_2]$ to be part of a reiterating partial data if $X_{loc}[loc_2 - 1] = X_{loc}[loc_2 - 2]$; $X_{loc}[loc_2 + 1] = X_{loc}[loc_2 + 2]$; and $X_{loc}[loc_2]6 = X_{loc}[loc_2 + 1]$ and $X_{loc}[loc_2]6 = X_{loc}[loc_2 - 1]$.

Type 2 and type 3 Nonrepeating mbs: When $X_{loc}[loc_2]$ is not a character in re-counting mb, formerly choose it either in a long or short mb. We investigate the partial data $X_{loc}[loc_2 - log*|\Sigma_{loc}| - 3 \dots loc_2 - 1]$. When there is a k such that $X_{loc}[loc_3] = X_{loc}[loc_3 - 1]$ and loc_3 is more significant than all, a repeating mb ending at location loc_3 exists. In the landmark, thus, we analyze $X_{loc}[loc_2]$ as a short mb part, beginning at $X[loc_3 + 1]$. Inspecting the partial data $X[loc_2 + 1 \dots loc_2 + 5]$ permits us to find if there is an additional repeating mb nearby location, loc_2, and we can decide the formation of a node comprising $X_{loc}[loc_2]$. When no repeating mb is marked in $X_{loc}[loc_2 - log^*|\Sigma_{loc}| - 3 \dots loc_2 - 1]$, it is possible to use the alphabet decrease procedure to find a landmark. The ability to find a nearby landmark to a character through observing only a limited number of successive nearby codes demonstrates that if a modification occurs outside this area, a similar landmark will be found; hence, a similar node will be shaped comprising that symbol. This permits the verification of the subsequent lemma. □

Lemma 6. *Inserting* $loc_3 \leq log^*n + 10$ *contiguous symbols into* X *to obtain* X' *means* $|A_i(X) - A_i(X')|1 \leq 2(log^*n + 10) \in$ *levels.*

Proof. We have a contribution after the insertion itself into the $L1$ distance and its effect on the nearby vicinity. Assume that the aggregate of characters at the level i break down to distinct nodes next to addition equated to earlier nodes. Consider the total characters at a level i break down in a different way as a result of the insertion P_i. Lemma 4.5 proves that in a non-overlapping mb, any character $log^*|\Sigma_{loc}| + 5$ indexes to the right or above five indexes to the left of each character can be altered and will identify the identical nearby landmark; hence, it will be shaped according to similar nodes. Hence, it will not pay for P_i. In the same way, for an overlapping mb, any character within the block will be parsed into a similar node (i.e., into a tripartite of that character), excluding the last four characters and depending on the block size.

Thus, for an overriding mb, $P_i \leq 4$. The total characters that are lower-level parsed in a different way into nodes as a result of the insertion are max $P_i - \frac{1}{2}$, and there is an area of 5 characters at maximum to the left, plus $log^*|\Sigma_i| + 5$ characters on the right side are parsed differently at a level i. As noted earlier, $|\Sigma_i| \leq |X| \leq n$, and the recurrence $P_i \leq P_i - \frac{1}{2} + log^*n + 10$ can consequently form. If $P_i - 1 \leq 2(log^*n + 10)$, then $P_i \leq 2(log^*n + 10)$. Insertion point, $P_0 \leq log^*n + 10$. To conclude, $|A_i(X) - A_i(X')|1 \leq 2\left(P_i - \frac{1}{2}\right)$, so then we might miss $P_i - \frac{1}{2}$ ancient nodes and obtain these various new nodes. □

Lemma 7. *Removing* $k < log^*n + 10$ *contiguous characters from* X *to obtain* X' *means* $|A_i(X) - A_i(X')|1 \leq 2(log^*n + 10)$.

Proof. Notice that the removal of a series of symbols is exactly the double to addition of that series at the similar index. Assume that a series of symbols pop in and then are removed; the resulting data are the same as the original data. Hence, the modified nodes number must be essentially limited by its equal quantity to the insert, as proved in Lemma 6. We merge both lemmas to confirm that modifying processes limited the parse-tree influences. □

Lemma 8. *When a particular allowed operation of edit distance alters data* X *to* X'*, then* $|A(X) - A(X')|1 \leq 8logn(log^*n + 10)$.

Proof. Each permissible procedure is assumed.

Character edit processes

The scenario for inserting follows straightaway from Lemma 6, as the character insertion operation will influence the breakdown of max $2(log^*n + 10)$ characters at all levels and the maximum levels $log2n$. Overall, $|A(X) - A(X')|1 \leq 2logn(log^*n + 10)$. Likewise, the scenario for deletion follows instantaneously from Lemma 7. To conclude, the case for substitution is presented by noticing that a symbol substitution assumed that can be used for removing instantly contiguous to an addition.

Partial data Moves

If the size of partial data being moved is $log^*n + 10$ at maximum, then a move can be said to be a deletion of the partial data following its reinsertion somewhere else. From Lemma 6 and Lemma 7, we see that $|A(X) - A(X')|1 \leq 4logn(log^*n + 10)$. Otherwise, we assume parsing of partial data by ESP. Assume a symbol in a non-iterating mb greater than $log^* + 5$ symbols from the beginning of partial data and greater than 5 symbols from the tail. As per Lemma 5, only symbols inside the partial data being moved decide how that symbol is parsed. Therefore, the parsing of these symbols and, hence, the contribution to $A(X)$ are free of the position of partial data in original data. Only the $log^*n + 5$ symbols at the start and 5 symbols of the partial data at the end will influence the data-parsing procedure. We can consider these to be the deletion of two partial data of the size $k \leq log^*n + 10$ and their reinsertion somewhere else.

Lemma 8 demonstrates that each permissible action influences the $L1$ conversion distance by $8logn(log^*n + 10)$ maximum. Assume we initiate with Y and accomplish a sequence of d editing processes, generating $Y_1, Y2, \ldots Y_d$ as a result. To conclude, $Y_d = X$, so $|A(Y_d) - A(X)|1 = 0$. We start with a number, $|A(Y) - A(X)|1$; in addition, we are aware that $\left|A\left(Y_{loc_2}\right) - A\left(Y_{loc_2+1}\right)\right| \leq 8logn(log^*n + 10)$ was discussed above. Henceforth, as $d(X, Y)$ steps, convert Y to X, and then $|A(Y) - A(X)|\frac{1}{8}logn(log^*n + 10) \leq d(X, Y)$, giving around $d(X, Y).8logn(log^*n + 10)$. □

4.3. Data Aggregation Level 2 Problem Solution

In this subsection, an algorithm is proposed that solves the problem of DMM. For any data X, we adopt that $A(X)$ needs to be placed in the space of $O(|X|)$ via registering nonzero parts only of $|X|$. Furthermore, appropriately, we save $A(X)[x]$ as an array indexed by $h(x)$ if it is nonzero, and we hold x as a reference to X, along with $|x|$.

The following outcome on pairwise data association is tracked directly from Theorems 1 and 2 composed with the surveillance that is given: $A(Y)$ and $A\ (X)|A(Y) - A(X)|1$ found in $O(|Y| + |X|)$ time.

Theorem 3. *Given data X and Y with* $n = max(|X|, |Y|)$*, to approximate a deterministic algorithm used for* $d(X, Y)$ *to precise up to an* $O(lognlog^*n)$ *factor in* $O(nlog^*n)$ *time with* $O(n)$ *space.*

4.3.1. Pruning Lemma

Before solving the data-match problem, pattern p of size m associate with $t[loc \ldots n]$ for each i, then there will be $O(n)$ comparisons. Moreover, we have to calculate the distance among p and $t[loc \ldots loc_3]$ for all possible $loc_3 \geq loc$ to calculate the optimal placement beginning at the location i, that shows sub-problems generally $O(mn)$. In the worst case, the algorithm of classical dynamic programming accomplishes the whole evaluations in $O(mn)$ time at max by using the dependency between sub-problems. In the algorithm, a different methodology is proposed. Initially, the below crucial observations were made:

Lemma 9. *(Pruning Lemma) Given an arrangement, t, and string, s,* $\forall loc, loc_2 : 1 \leq loc \leq loc_2 \leq n$, $d(t, s[loc \ldots loc + m - 1] \leq 2d(t, s[loc \ldots loc_2])$.

Proof. Note that for all j values in lemma, $d(t, s[loc \dots loc_2]) \geq |(Y - loc + 1) - m|$, as these multiple symbols must be deleted or inserted. By triangle inequality of match with moves, for all, $loc_2 d(t, s[loc \dots loc + m - 1]$.

$$\begin{aligned} &\leq d(t, s[loc \dots loc_2]) + d(s[loc \dots loc_2], s[loc \dots loc + m - 1] \\ &\quad = d(t, s[loc \dots loc_2]) + |(Y - loc + 1) - m| \\ &\leq 2d(t, s[loc \dots loc_2]) \end{aligned}$$

The lengthiest common prefix followed to assume $s[loc \dots loc_2]$ and $s[loc \dots loc + m - 1]$.

The above Lemma importance is that it serves to estimate only $O(n)$ distances, such as $d(t, s\,loc \dots loc + m - 1)$ for all i values, to solve the data-match problem DMM, equal to a factor 2 approximation. Therefore, it trims away candidates from the "quadratic" number of distance calculations that an honest process would require. □

4.3.2. ESP Sub-Trees

We estimate distance among partial data s and the pattern t by matching the parsing ESP of two patterns. On the other hand, it would be costly and useless to parse the partial data s. In this subsection, we indicate that an ESP tree is assigned to a datum, and being inspired by a partial datum means that the subtree will have similar editing sensitivity features as the complete one.

Definition 2. *Let $ET_{loc}(X)_{loc_2}$ at loc_2, such that loc represents the level of UAV in the breakdown of X.*

The set of values $X[a_1 \dots a_n]$ to the tags on last level of the sub-tree, where (root) $ET_{loc}(X)_{loc_2}$ is defined as range, $\left(ET_{loc}(X)_{loc_2}\right)$, and relates to the partial data, $X[a_1 \dots a_n]$.

The ESP subtree of data X can be defined $EST(X, l, r)$ as the sub-tree $ET(X)$ that comprises all $X[loc]$, such that $l \leq\ loc\ \leq r$, plus all parent UAVs. Formally, we search and discover UAVs of $ET_{loc}(X)_{loc_2}$, where $[l \dots r] \cap range(ET_{loc}(X)loc_2)6 = \emptyset$. A name node is derived from $ET_{loc}(X)_{loc_2}$, which is $loc, hXrange(ET_{loc}(X)loc_2) \cap [a \dots b))$.

This results in an appropriate sub-tree of $ET(X)$, as a node is part of the sub-tree if one of its children is counted in as a minimum. As stated earlier, we can define an array that represents this ET.

Definition 3. *Define $AX(X,\ l,\ r)$ as the characteristic array of $EST(X)$ by similarity with $A(X)$; namely $AX(X,\ l,\ r)[x]$ represents the the total times the partial data x are denoted as a node in $EST(X,\ l,\ r)$. In this regard, $EST(X,\ 1,\ |X|) = ET(X)$; however, if not, generally $EST(X,\ l,\ r) = ET(X[l \dots r])$. Through $EST(X,\ l,\ r)$, it send the features of the edit-sensitive parsing. Similar to Theorem 2, Theorem 4 is stated as follows.*

Theorem 4. *Let d be $d(Y[l_p \dots r_p],\ X\ [l_q \dots r_q])$. Then, $d \leq 2||A.XY,\ l_p,\ r_p) - A\ X(X,\ l_q,\ r_q])||1 = O(logn\ log^* n)d$.*

Proof. Definitely, as Lemma 9 does not assume anything regarding the tree structure, the lower bound exists, meaning that the edit distance is simply double the length of the difference amongst the ESP subtrees.

For the higher limit, assume implementing the required editing processes to the partial data of X and Y. The impact on the ordinary ESP trees is observed, $ET(Y)$ and $ET(X)$. Theorem 2 verified that all editing processes could create a divergence of $O(lognlog^* n)$ maximum among $A(X)$ and $A(X')$. Following this, the variance in $AS(X, l, r)$ should be limited as a result of similar volume: it seems impossible to remove additional UAVs, as the UAVs of $EST(X, l, r)$ is the subset of UAVs of $ET(X)$. Hence, as demonstrated in Theorem 2, the aggregate divergence $\left|\left|A.X\left(Y, l_p, r_p\right) - A.X\left(X, l_q, r_q\right)\right|\right|1 = d \cdot O(lognlog^* n)$. □

Lemma 10. *$A\ X(X,\ l + 1,\ r + 1)$ can be calculated from $A\ X(X,\ l,\ r)$ in time $O(log\ |X|)$.*

Proof. Remember that a UAV is contained within $EST(X, l, r)$ once any dependent is last-level analogous to $X[i]$ contained when $i \in [l \dots r]$. It results in an easy method for discovering $EST(X, l+1, r+1)$ from $EST(X, l, r)$ and $AX(X, l+1, r+1)$. We need to eliminate X and some parents that do not cover $X[l]$ from $EST(X, l, r)$ to generate $AX(X, l+1, r+1)$.

Each parent X is adjusted to guarantee that its tag is accurate. The UAV located at i(level) matching to the partial data $X[loc_2 \dots loc_3]$, which is a parent of X, was earlier denoted in the sub-tree via the $(loc, h(X[l \dots loc_3]))$; it is required to be substituted with $(loc, h(X[l+1 \dots loc_3]))$.

Suppose y is a *UAV* conforming to $X[r+1]$ in $ET(X)$ at the right. We add y to $EST(X, l+1, r)$ to generate $EST(X, l+1, r+1)$, and we give the ancestor of y and designate her ancestor in $ET(X)$; give her the parent addition of your choice when it is absent. Alter each parent of y to guarantee that their name is accurate: a parent of y corresponding to the data $X[loc_2 \dots loc_3]$ will set $(loc, h(X[loc_2 \dots r+1]))$. As these circumstances, we simply assume parents of a child UAV since the depth of tree is $O(log|X|)$, and this shows that the process complexity with respect to time is $O(log|X|)$. □

4.4. Data Aggregation Level 2 Algorithm

Theorem 5. *Given a text, t, and string s, to resolve the data-matching issue with moves by calculating an $O(log\ n\ log^* n)$ approximation to $D[loc] = min\_loc \leq loc_3 \leq n\ d(s,\ t[loc \dots loc_3])$ for each loc in time $O(n\ log\ n)$.*

Proof. Our algorithm is as follows:

Given a dataset X of size m and dataset Y of size n, we calculate $ET(s)$ and $ET(t)$ in time $O(mlog^* n)$ as per Theorem 1.

Measure $EST(t, 1, m)$. This can be performed in the $O(n)$ worst case, as a pre-order traversal of $ET(t)$ will be performed to determine which nodes are in $EST(t, 1, m)$. From this, we can calculate $\hat{D}[1]||AX(t, 1, m) - AX(p, 1, m)||1$. We then recursively calculate $||AX(t, loc+1, loc+m) - AX(s, 1, m)||1$ from $||AX(t, loc, loc+m-1) - AX(s, 1, m)||1$ via Lemma 7 to discover which nodes we need to add to or remove from $EST(t, loc, loc+m-1)$ and regulate the total of the difference properly. □

This requires n comparisons and will take $O(logn)$ time for each. By Theorem 4 and Lemma 9, $D[loc] \leq \hat{D}[loc] \leq O(lognlog^* n)D[loc]$. If $logn$ is $O(logm)$, as one would expect for some rational size string and text, then a tighter investigation demonstrates the running time to be $O(nlogm)$. This is for the reason that we merely want to assume the lower $logm$ levels of the parse trees; above, this $EST(t, loc, loc+m-1)$ has only one node in each level. Figure 5 shows the flow of data aggregation Algorithm 2 to find the duplicated data in two datasets, X and Y.

Algorithm 2: Data Aggregation Algorithm to Find the Duplicated Data in Two Datasets, X and Y

Procedure Data_Match_with_Moves (DMM)
Input: Datasets, X and Y
Output: True/False // Duplication Found or Not Found
Initializations

1. i. $X_{len} \leftarrow X.length()$
   ii. $Y_{len} \leftarrow Y.length\ ()$
2. Allocate vector space $V[0{:}m, 0{:}n]$ ▷ $V[i, j]$ will contain the length of $X[1{:}i]$ and $Y[1{:}j]$.
3. $V[0, j] \leftarrow 0$ for all $0 \leq j \leq n$ and $V[i, 0] \leftarrow 0$ for all $0 \leq i \leq m$. ▷ Base Cases
4. for ($i \leftarrow 1$ to $Xlen$) then
5. for ($j \leftarrow 0$ to $Ylen$) then // matching the symbols from X with Y symbols
6. if ($j = 0$) then // if Y is blank then eliminate all X symbols
   // if symbol from both dataset is matching then no operation is required

Algorithm 2: *Cont.*

```
                    V[i % 2][j] ← i;
                      else if (X[j − 1] = Y[i − 1]) then
                      V[i % 2][j] ← V[(i − 1) % 2][j − 1];
                    end else if
    // if symbols from both datasets do not match, then we take the smallest from 3
    operations.
    // i.e., insert, delete and substitute
    else then
                    V[i % 2][j] ← 1 + min(V[(i − 1) % 2][j],
                    min(V[i % 2][j − 1],
                    V[(i − 1) % 2][j − 1]));
                    end if
    end for
      end for
    // after filling the V vector, if the size of Xlen is even, then
    //we end up in the 0th row else
    //we end up in the ith row, so we take Xlen % 2 to get row
        P←V[Xlen % 2][Ylen] // the final value after matching two datasets
        L←max (X_len, Y_len)
    if (P < L/2), then
        return 1 // true value will return, i.e., duplication found
        else then
    return 0 // false value will return, i.e., duplication not found
    end if
    end procedure DMM
```

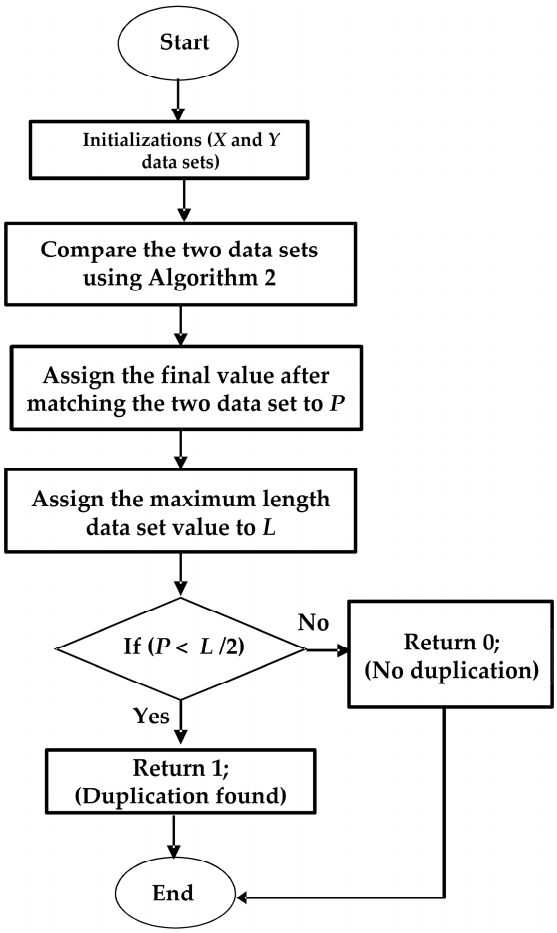

Figure 5. Data aggregation model.

5. Performance Evaluation and Simulation Study

The performance metrics for evaluating the proposed scheme were measured with the mean packet delivery ratio (PDR), mean energy consumption, end-to-end delay, packets drop ratio, communication overhead, and bandwidth utilization [41–43]. The PDR is the ratio of packets received by the receiver UAVs versus the packet sent by the sender UAVs. The higher ratio means that the performance of the proposed scheme is better. The energy consumption shows the mean amount of energy consumed by the UAVs for data transmission. The end-to-end delay means the time taken by the UAVs for packets' sending and receiving. It also measures the delay caused during route discovery and waiting in a queue. The packet drop ratio means that the packets may be dropped during the transmission, and it counts the ratio of the total number of packets received and packets sent. Sometimes, the same packets or the additional information communicated to the UAVs-CH reduces communication speed and consumes energy. The details of the simulation parameters are given in Table 1.

Table 1. Simulation parameters.

Parameter	Value
Network Simulator	MATLAB
Covered Area	2 km × 2 km
MAC Protocol	IEEE 802.11 and IEEE 802.16
Antenna Type	Omni directional
Propagation Model	Two-ray ground reflection model (intra-cluster) Long-distance propagation loss model (inter-cluster)
Radio Frequency	2.4 GHz, 5 GHz
Number of UAVs	10 to 60
UAV Altitude	40–50 m
UAV Transmission Range	200 to 300 m
UAV Mobility Model	Random waypoint model
Transport Protocol	Stream control transmission protocol (SCTP)
Traffic Model	Poisson traffic model
Application Packet Size	1000 Bytes
Initial Energy	2 to 5 J
Channel Model	Multi-Propagation Channel (MPC) Model

The performance of the proposed redundant data elimination aggregation approach was compared with non-redundant data elimination aggregation approaches, i.e., EE-UAV-DA, OC-mUAV, and TA-UAV-DA. The data rate was fixed in the simulation, i.e., 250 Kbps, and the number of UAVs varied from 10 to 60.

Figure 6 shows the number of UAVs and the mean end-to-end delay. It is observed that with the increase in UAVs, the proposed scheme has less of a mean delay as compared to the EE-UAV-DA, OC-mUAV, and TA-UAV-DA. The FSNet-OC-DA uses HBA to form clusters and data aggregation. Selecting optimized cluster heads has a strong impact on end-to-end delay. The results show that our proposed scheme (FSNet-OC-DA) performs better than other schemes under consideration.

Figure 7 presents the number of UAVs vs. PDR. The simulation results show that, initially, the PDR decreased with an increase in UAVs. The PDR of the proposed scheme falls, while the EE-UAV-DA, OC-mUAV, and TA-UAV-DA have more decreases than our proposed algorithm. The impact of optimum cluster formation using HBA reflects that once optimized cluster heads are selected, this will increase the PDR. This is because the cluster heads belong to optimal zones with high energy, relative mobility, and high density.

As the packets are routed via cluster heads, the proposed scheme with optimal cluster heads performs better than others.

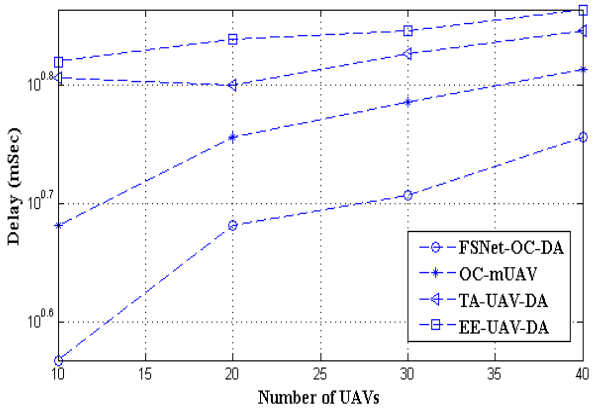

Figure 6. Number of UAVs vs. end-to-end delay.

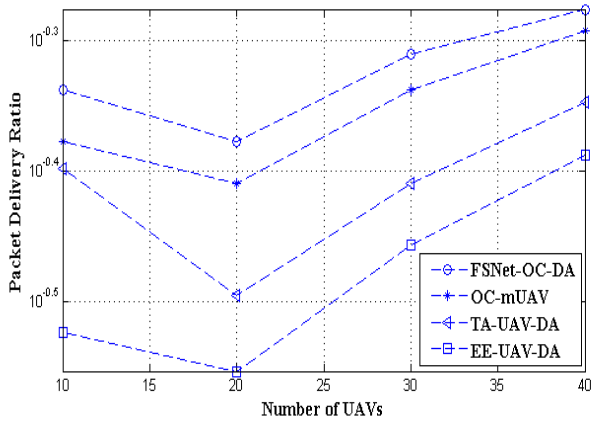

Figure 7. Number of UAVs vs. PDR.

Figure 8 illustrates the number of UAVs and packet drop ratio of the proposed scheme and other existing approaches. The simulation result shows that the drop ratio increased in the existing methods and our proposed scheme with the increase in UAVs. The packet drop ratio increased in all schemes because, in dense networks, the number of packets increases, and the load on other nodes also increases. This results in packet drops because the nodes are overloaded. The packet drop in our proposed scheme is less than others because we optimized cluster formation using HBA.

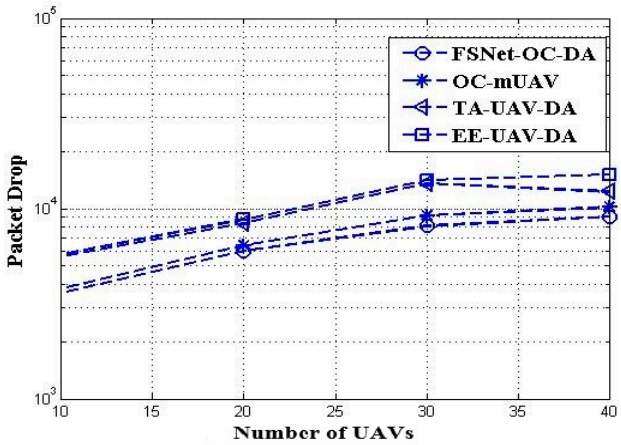

Figure 8. Number of UAVs vs. packet drop ratio.

The increase in the number of UAVs has a direct impact on residual energy. Figure 9 shows that the proposed FSNet-OC-DA residual energy (recorded in round 10) is better than all existing approaches. The proposed scheme eliminates duplicate data transmission to the UAVS-CH by using a near-linear time algorithm. Eliminating data reduces long-distance communication. As we already know, communication consumes more energy than computation. Our model focuses on optimization and reduces communication costs. The efficient optimization and data aggregation enable nodes to consume less energy. Hence, FSNet-OC-DA energy consumption is better than the other schemes.

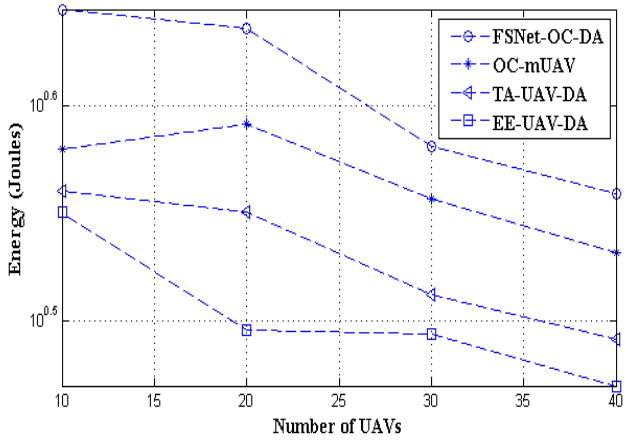

Figure 9. Number of UAVs vs. residual energy.

Figure 10 shows that the communication overhead increases with the increase in UAVs, but the FSNet-OC-DA has significantly less communication overhead. The number of UAVs in each cluster is fixed in the simulation, i.e., 40, and the data rate varies from 50 to 250 Kbps. Figure 11 represents the UAVs' data rate and end-to-end delay of the FSNet-OC-DA, EE-UAV-DA, OC-mUAV, and TA-UAV-DA. The simulation result in Figure 10 shows that the end-to-end delay increases with the increase in data rates from 50 to 250 Kbps. It is observed that the FSNet-OC-DA has less end-to-end delay than the existing approaches. The PDR for different data rates (50 to 250 Kbps) is represented in Figure 12. Initially, when the data rate is 50 Kbps, the PDR is very high, but the delivery ratio decreases gradually with the increase in the data rate.

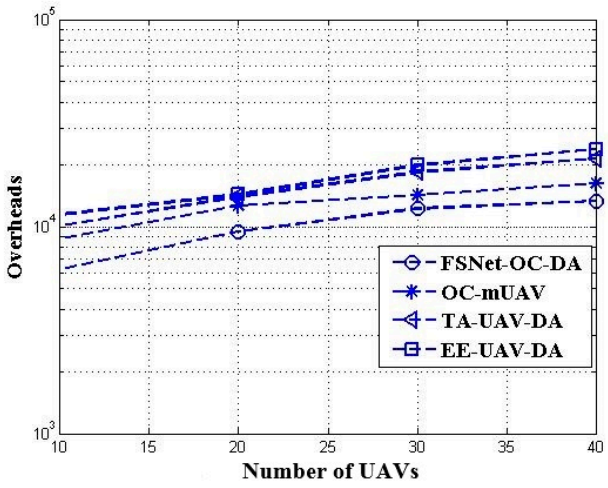

Figure 10. Number of UAVs vs. message overhead.

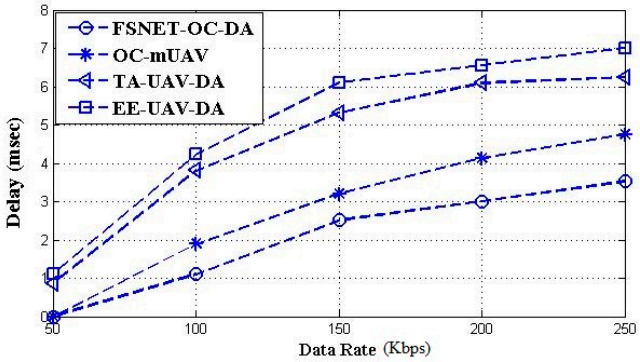

Figure 11. UAVs' data rate vs. end-to-end delay.

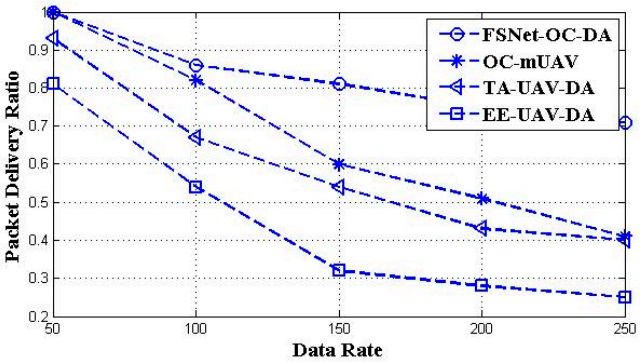

Figure 12. UAVs' data rate vs. PDR.

Figure 13 shows that, initially, the UAVs' data rate is 50 Kbps, and the packet drop ratio is very minimal. Still, with the increase in the UAVs' data rate, the packet drop ratio increases up to the highest level. The FSNet-OC-DA packet drop ratio is always lower when the data rate varies from 50 to 250 Kbps.

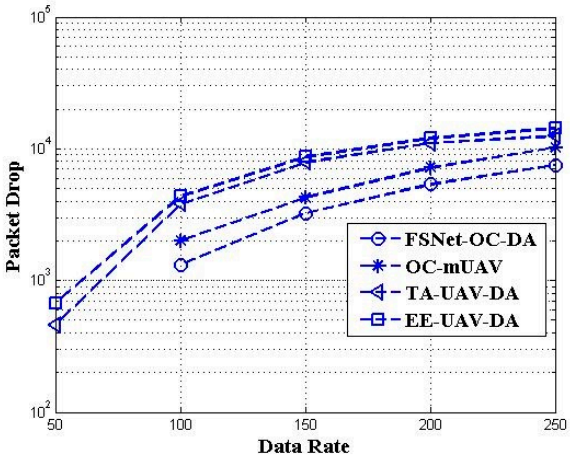

Figure 13. UAVs' data rate vs. packet drop ratio.

Figure 14 represents the data rate and residual energy of UAVs. The residual energy decreases with the increase in the data rate of UAVs. The remaining residual energy of the FSNet-OC-DA is still better among other existing approaches.

Figure 15 represents the UAVs' data rate and the ratio of the message overhead. The non-redundant data elimination approaches' communication overhead is higher than the proposed redundant data elimination schemes. Bandwidth is the capacity of communication channels among the UAVs. The bandwidth occupancy rate is the ratio of the

bandwidth used by the redundant data elimination aggregation approach and without redundant aggregation approaches.

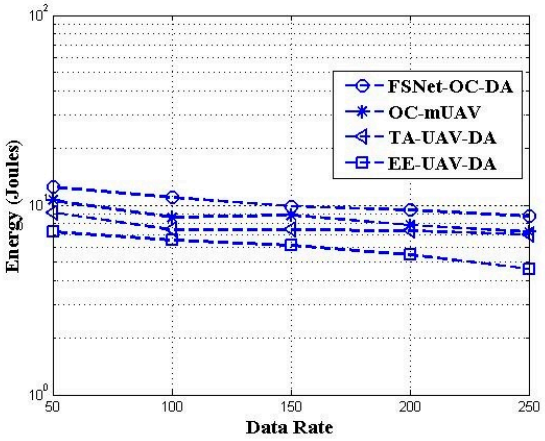

Figure 14. UAVs' data rate vs. residual energy.

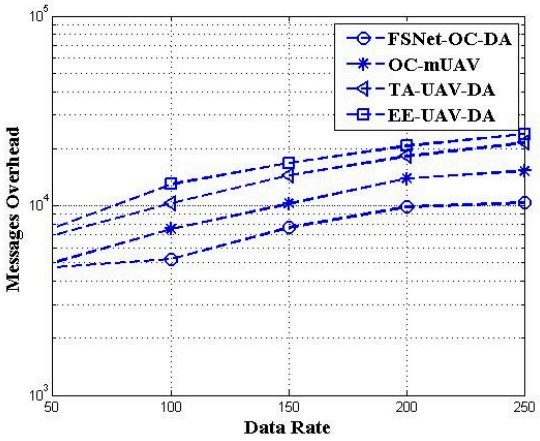

Figure 15. UAVs' data rate vs. message overhead.

Figure 16 represents the bandwidth occupancy and redundancy rate of each UAV. The bandwidth occupancy of the FSNet-OC-DA is moving towards close to zero compared to the existing approaches. In FSNet-OC-DA, the UAVs avoid redundant data and transmit only actual data to a UAVS-CH. At the same time, the conventional data aggregation approaches utilize (50) percent bandwidth of the total available bandwidth.

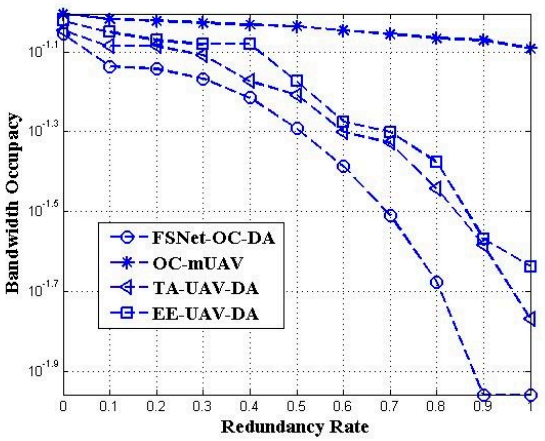

Figure 16. Redundancy rate vs. bandwidth occupancy.

6. Conclusions and Future Work

Sensors and UAVs have made it easier to monitor, observe, and share information about an area of interest remotely. The researchers proposed energy-efficient schemes by considering different parameters, such as reducing communication distance, computation cost, mobility, and degree. However, data collection minimizes the communication load to save bandwidth and energy.

In this study, the honeybee foraging property was first used to select the optimal CH set and form stable and balanced clusters. The modified HBA selects UAVs-CH based on residual energy, UAV degree, and relative mobility. To transmit data, the UAV connects to the nearest CH. Ordinary UAVs choose CH randomly if they have the same distance with more than one CH. The reaffiliation rate will decrease with the proposed stable clustering procedure. Secondly, ordinary UAVs transmit data to their CH once clusters are formed. An aggregation method based on dynamic programming is proposed to save energy consumption and bandwidth. The data aggregation procedure is applied at the cluster level to minimize communication and save bandwidth and energy. Simulation experiments validate the proposed FSNet-OC-DA. FSNet-OC-DA is compared with non-redundant data elimination and aggregation approaches, such as EE-UAV-DA, OC-mUAV, and TA-UAV-DA, in terms of the end-to-end delay, PDR, packet drop ratio, residual energy, communication overhead, bandwidth occupancy with varying numbers of UAVs, data rate, and redundancy rate. The simulation results show that our proposed FSNet-OC-DA outperforms state-of-the-art cluster-based data aggregation schemes.

Author Contributions: Conceptualization, M.A.; methodology A.S. and Q.J.; software, A.S. and I.W.; validation, A.S. and M.Y.A.; draft preparation, A.S., M.Y.A. and I.W.; review and editing, M.A. and I.W.; visualization, A.S.; supervision, Q.J.; funding acquisition, M.Y.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data will be made available on request.

Acknowledgments: The authors wish to thank the editor and anonymous referees for their helpful comments on improving the quality of this paper.

Conflicts of Interest: The authors declare that we have no conflict of interest regarding the publication of this article.

References

1. Alam, M.M.; Arafat, M.Y.; Moh, S.; Shen, J. Topology control algorithms in multi-unmanned aerial vehicle networks: An extensive survey. *J. Netw. Comput. Appl.* **2022**, *207*, 103495. [CrossRef]
2. Sarkar, N.I.; Gul, S. Artificial Intelligence-Based Autonomous UAV Networks: A Survey. *Drones* **2023**, *7*, 322. [CrossRef]
3. Abu-Baker, A.; Shakhatreh, H.; Sawalmeh, A.; Alenezi, A.H. Efficient data collection in UAV-assisted cluster-based wireless sensor networks for 3D Environment: Optimization Study. *J. Sens.* **2023**, *2023*, 9513868. [CrossRef]
4. Luo, X.; Chen, C.; Zeng, C.; Li, C.; Xu, J.; Gong, S. Deep Reinforcement Learning for Joint Trajectory Planning, Transmission Scheduling, and Access Control in UAV-Assisted Wireless Sensor Networks. *Sensors* **2023**, *23*, 4691. [CrossRef] [PubMed]
5. Ahmad, S.; Zhang, J.; Khan, A.; Khan, U.A.; Hayat, B. JO-TADP: Learning-Based Cooperative Dynamic Resource Allocation for MEC–UAV-Enabled Wireless Network. *Drones* **2023**, *7*, 303. [CrossRef]
6. Zhou, R.; Zhang, X.; Song, D.; Qin, K.; Xu, L. Topology Duration Optimization for UAV Swarm Network under the System Performance Constraint. *Appl. Sci.* **2023**, *13*, 5602. [CrossRef]
7. Salam, A.; Javaid, Q.; Ali, G.; Ahmad, F.; Ahmad, M.; Wahid, I. Flying Sensor Network optimization using Bee Intelligence for internet of things. In *Advances in Intelligent Systems and Computing, Proceedings of the International Conference on Computer Science and Information Technologies, Zbarazh, Ukraine, 23–26 September 2020*; Springer: Cham, Switzerland, 2020; Volume 1252, pp. 331–339.
8. Chen, T.; Dong, F.; Ye, H.; Wang, Y.; Wu, B. Data Collection Mechanism for UAV-Assisted Cellular Network Based on PPO. *Electronics* **2023**, *12*, 1376. [CrossRef]
9. Amodu, O.A.; Nordin, R.; Jarray, C.; Bukar, U.A.; Raja Mahmood, R.A.; Othman, M. A Survey on the Design Aspects and Opportunities in Age-Aware UAV-Aided Data Collection for Sensor Networks and Internet of Things Applications. *Drones* **2023**, *7*, 260. [CrossRef]
10. Xiong, J.; Li, Z.; Li, H.; Tang, L.; Zhong, S. Energy-Constrained UAV Data Acquisition in Wireless Sensor Networks with the Age of Information. *Electronics* **2023**, *12*, 1739. [CrossRef]

11. Kim, T.; Lee, S.; Kim, K.H.; Jo, Y.-I. FANET Routing Protocol Analysis for Multi-UAV-Based Reconnaissance Mobility Models. *Drones* **2023**, *7*, 161. [CrossRef]
12. Arafat, M.Y.; Moh, S. JRCS: Joint Routing and charging strategy for logistics drones. *IEEE Int. Things J.* **2022**, *9*, 21751–21764. [CrossRef]
13. Arafat, M.Y.; Habib, M.A.; Moh, S. Routing Protocols for UAV-Aided Wireless Sensor Networks. *Appl. Sci.* **2020**, *10*, 4077. [CrossRef]
14. Noh, K.-L.; Wu, Y.-C.; Qaraqe, K.; Suter, B.W. Extension of pairwise broadcast clock synchronization for Multicluster sensor networks. *EURASIP J. Adv. Signal Process.* **2008**, *2007*, 286168. [CrossRef]
15. Cheng, K.-Y.; Lui, K.-S.; Wu, Y.-C.; Tam, V. A distributed Multihop Time Synchronization Protocol for wireless sensor networks using pairwise broadcast synchronization. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 1764–1772. [CrossRef]
16. Alam, M.M.; Moh, S. Survey on Q-Learning-Based Position-Aware Routing Protocols in Flying Ad Hoc Networks. *Electronics* **2022**, *11*, 1099. [CrossRef]
17. Xiong, F.; Zheng, H.; Ruan, L.; Wang, H.; Tang, L.; Dong, X.; Li, A. Energy-saving data aggregation for Multi-UAV system. *IEEE Trans. Veh. Technol.* **2020**, *69*, 9002–9016. [CrossRef]
18. Aadil, F.; Raza, A.; Khan, M.F.; Maqsood, M.; Mehmood, I.; Rho, S. Energy Aware Cluster-Based Routing in Flying Ad-Hoc Networks. *Sensors* **2018**, *18*, 1413. [CrossRef]
19. Arafat, M.Y.; Moh, S. Localization and clustering based on swarm intelligence in UAV Networks for Emergency Communications. *IEEE Internet Things J.* **2019**, *6*, 8958–8976. [CrossRef]
20. Yang, J.; Wang, X.; Li, Z.; Yang, P.; Luo, X.; Zhang, K.; Zhang, S.; Chen, L. Path planning of unmanned aerial vehicles for farmland information monitoring based on WSN. In Proceedings of the 2016 12th World Congress on Intelligent Control and Automation (WCICA) 2016, Guilin, China, 12–15 June 2016.
21. Yu, Y.; Ru, L.; Chi, W.; Liu, Y.; Yu, Q.; Fang, K. Ant colony optimization based polymorphism-aware routing algorithm for ad hoc UAV network. *Multimed. Tools Appl.* **2016**, *75*, 14451–14476. [CrossRef]
22. Holtorf, L.; Titov, I.; Daschner, F.; Gerken, M. UAV-Based Wireless Data Collection from Underground Sensor Nodes for Precision Agriculture. *AgriEngineering* **2023**, *5*, 338–354. [CrossRef]
23. Zhang, X.; Cao, Y. Memetic Algorithm with Isomorphic Transcoding for UAV Deployment Optimization in Energy-Efficient AIoT Data Collection. *Mathematics* **2022**, *10*, 4668. [CrossRef]
24. Bharany, S.; Sharma, S.; Frnda, J.; Shuaib, M.; Khalid, M.I.; Hussain, S.; Iqbal, J.; Ullah, S.S. Wildfire Monitoring Based on Energy Efficient Clustering Approach for FANETS. *Drones* **2022**, *6*, 193. [CrossRef]
25. Wang, X.; Zhou, Q.; Cheng, C.-T. A UAV-assisted topology-aware data aggregation protocol in WSN. *Phys. Commun.* **2019**, *34*, 48–57. [CrossRef]
26. Wu, Q.; Sun, P.; Boukerche, A. An energy-efficient UAV-based data aggregation protocol in Wireless Sensor Networks. In Proceedings of the 8th ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications 2018, Montreal, QC, Canada, 28 October—2 November 2018; pp. 34–40.
27. Thammawichai, M.; Baliyarasimhuni, S.P.; Kerrigan, E.C.; Sousa, J.B. Optimizing Communication and computation for Multi-UAV information gathering applications. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *54*, 601–615. [CrossRef]
28. Dong, M.; Ota, K.; Lin, M.; Tang, Z.; Du, S.; Zhu, H. UAV-Assisted Data Gathering in wireless sensor networks. *J. Supercomput.* **2014**, *70*, 1142–1155. [CrossRef]
29. Liu, B.; Zhu, H. Energy-Effective Data Gathering for UAV-Aided Wireless Sensor Networks. *Sensors* **2019**, *19*, 2506. [CrossRef] [PubMed]
30. Cvetković, A.; Blagojević, V.; Manojlović, J. Capacity Analysis of Power Beacon-Assisted Industrial IoT System with UAV Data Collector. *Drones* **2023**, *7*, 146. [CrossRef]
31. Cao, J.; Zhu, X.; Sun, S.; Wei, Z.; Jiang, Y.; Wang, J.; Lau, V.K.N. Toward industrial metaverse: Age of information, latency and reliability of short-packet transmission in 6G. *IEEE Wirel. Commun.* **2023**, *30*, 40–47. [CrossRef]
32. Li, Z.; Zhao, W.; Liu, C. Completion Time Minimization for UAV-UGV-Enabled Data Collection. *Sensors* **2022**, *22*, 5839. [CrossRef]
33. Nie, M.; Huang, P.; Zeng, J.; Lu, Y.; Zhang, T.; Lv, T. A Novel Dynamic Transmission Power of Cluster Heads Based Clustering Scheme. *Electronics* **2023**, *12*, 619. [CrossRef]
34. Zhang, M.; Li, J.; Wu, X.; Wang, X. Coalition Game Based Distributed Clustering Approach for Group Oriented Unmanned Aerial Vehicle Networks. *Drones* **2023**, *7*, 91. [CrossRef]
35. Mehmood, A.; Iqbal, Z.; Shah, A.A.; Maple, C.; Lloret, J. An Intelligent Cluster-Based Communication System for Multi-Unmanned Aerial Vehicles for Searching and Rescuing. *Electronics* **2023**, *12*, 607. [CrossRef]
36. Chen, G.; Chen, G. A Method of Relay Node Selection for UAV Cluster Networks Based on Distance and Energy Constraints. *Sustainability* **2022**, *14*, 16089. [CrossRef]
37. Salam, A.; Javaid, Q.; Ahmad, M. Bioinspired mobility-aware clustering optimization in flying ad hoc sensor network for internet of things: Bimac-FASNET. *Complexity* **2020**, *2020*, 9797650. [CrossRef]
38. Cormode, G.; Muthukrishnan, S. The string edit distance matching problem with moves. *ACM Trans. Algorithms* **2007**, *3*, 1–19. [CrossRef]

39. Shapira, D.; Storer, J.A. Edit Distance with Move Operations. In *Combinatorial Pattern Matching, Proceedings of the CPM 2002, Fukuoka, Japan, 3–5 July 2002*; Lecture Notes in Computer Science; Apostolico, A., Takeda, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2373, p. 2373. [CrossRef]
40. Maruyama, S.; Nakahara, M.; Kishiue, N.; Sakamoto, H. ESP-index: A compressed index based on edit-sensitive parsing. *J. Discrete Algorithms* **2013**, *18*, 100–112. [CrossRef]
41. Wheeb, A.H.; Nordin, R.; Samah, A.A.; Kanellopoulos, D. Performance Evaluation of Standard and Modified OLSR Protocols for Uncoordinated UAV Ad-Hoc Networks in Search and Rescue Environments. *Electronics* **2023**, *12*, 1334. [CrossRef]
42. Liu, P.; Wang, X.; Hawbani, A.; Busaileh, O.; Zhao, L.; Al-Dubai, A. FRCA: A novel flexible routing computing approach for wireless sensor networks. *IEEE Trans. Mob. Comput.* **2020**, *19*, 2623–2639. [CrossRef]
43. Hu, Y.; Liu, Y.; Kaushik, A.; Masouros, C.; Thompson, J. Timely data collection for UAV-based IOT Networks: A deep reinforcement learning approach. *IEEE Sens. J.* **2023**, *23*, 12295–12308. [CrossRef]