Latency-and-Coverage Aware Data Aggregation Scheduling for Multihop Battery-Free Wireless Networks

Zhipeng Cai, *Senior Member, IEEE*, and Quan Chen

***Abstract*—Battery-Free Wireless Sensor Networks (BF-WSNs) have been attracting increasing interests in the recent years. To reduce the latency in BF-WSNs, the Minimum Latency Aggregation Scheduling (MLAS) problem with coverage requirement q is proposed recently, which tries to choose q percent of nodes for communication and aggregation. In the existing method, the authors try to select nodes adaptively according to their energy status and schedule these nodes to achieve the minimum latency. Unfortunately, it cannot guarantee the distribution of the aggregated nodes and may result in these nodes being squeezed in a small area and a poor aggregation quality. Thus, we re-investigate the q-coverage MLAS problem in this paper, which can guarantee that the aggregated nodes are distributed evenly. Firstly, the 1-coverage MLAS problem, in which each node can be covered by at least one aggregated node, is studied. To reduce the latency, we intertwine the selection of aggregated nodes and the computation of a collision-free communication schedule simultaneously. Two algorithms are proposed by scheduling the communication tasks in the bottom-up and top-down manner respectively. Secondly, to satisfy the arbitrary coverage requirement q, three algorithms are proposed to guarantee the aggregated nodes are evenly distributed in the network with a low latency. Additionally, the method to extend the proposed algorithms for the BF-WSNs with multiple channels is also studied. The theoretical analysis and simulation results verify that the proposed algorithms have high performance in terms of latency.**

***Index Terms*—Data aggregation, latency-and-coverage aware, battery-free, multiple channels, wireless networks.**

I. Introduction

Thanks to the emerging energy harvesting technology, Battery-Free Wireless Sensor Networks (BF-WSNs) have drawn extensive attentions from researchers, where the rechargeable devices can be charged with a wireless charger through the air [1], [2]. Through harvesting energy from ambient environment, BF-WSNs can operate in a more autonomous fashion, which increases the applicability of BF-WSNs and Internet of Things (IoTs), making IoT a major source of big data. However, the harvested energy at each node remains scarce and differs greatly due to their limited capability. Thus, how to exploit the harvested energy at each node for sensing and networking algorithm design, has drawn a growing research interest recently [3], [4].

This work was supported by the the National Science Foundation of U.S. (1741277, 1704287, 1829674, 1912753, and 2011845). (*Corresponding author: Zhipeng Cai.*)

Zhipeng Cai is with the Department of Computer Science, Georgia State University, Atlanta, GA 30303 (e-mail: zcai@gsu.edu).

Quan Chen is with the School of Computers, Guangdong University of Technology, Guangzhou 510006, China (e-mail: quan.c@gdut.edu.cn).

Data aggregation is an essential operation in wireless sensor networks, where the sink node tries to obtain the summary information of the whole network [5]–[9]. Due to the limited communication range of sensor nodes, the problem of Minimum Latency Aggregation Scheduling (MLAS) is proved to be NP-hard and has attracted extensive attentions from researchers. In traditional battery-powered WSNs, the MLAS problem has been studied by [10]–[17], where the authors try to minimize the interference between wireless communication links to reduce the latency. Furthermore, when the sensor nodes exploit the duty-cycled working scheme for energy conserving, the MLAS problem has been studied by [18]–[22]. In their algorithms, the sleep latency between nodes is considered to further reduce the latency. However, all these methods are not suitable for BF-WSNs due to the dynamic energy status of sensor nodes.

For BF-WSNs, a sensor node can only transmit and receive messages after it has been charged enough energy, and the captured energy remains scarce and differs greatly among sensors. For example, according to [4], the harvested energy ranges from $1.44mJ/s$ to $20uJ/s$ at different distances, which means the charging time at each node are highly different. In such networks, the MLAS problem has been firstly studied by [4], where the energy-collision problem between communication tasks is introduced for BF-WSNs and three centralized algorithms are proposed to construct the aggregation tree and schedule BF-nodes adaptively according to their current energy status. Since the latency is mainly caused by the charging time in BF-WSNs, the authors in [1] proposed the MLAS problem with coverage requirement q recently, which tries to pick up q percent of nodes to participate in the aggregation process and schedule these nodes to achieve the minimum latency. However, their algorithm has the following two problems:

i) Problem 1. The nodes participated in the aggregation process are not evenly distributed in the network, which may cause the aggregated nodes being squeezed in a small area incurring a poor aggregation quality. This is because they only consider the latency during the selection of the aggregated nodes.
ii) Problem 2. Even only choosing q percent of nodes to participate in the aggregation process, their algorithm can be improved greatly. This is because the aggregation latency is not only related to the charging delay, but also the structure of the aggregation tree.

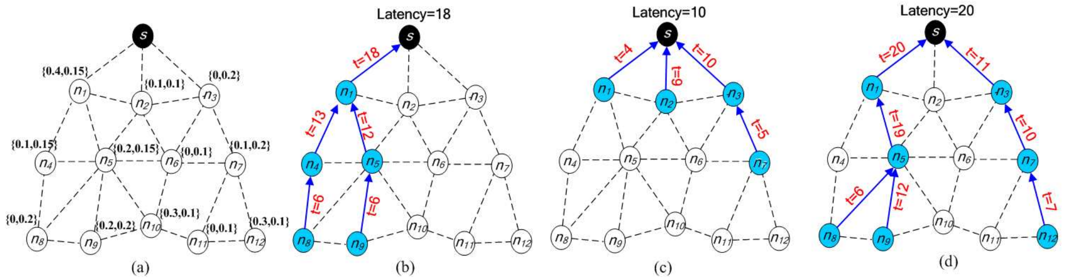


Fig. 1. The example of q-coverage data aggregation scheduling in BF-WSNs, where (a) denotes a BF-WSN, (b), (c) and (d) denote the resulted aggregation tree and communication schedule by three methods. The aggregated nodes are marked blue.

For example, Fig.1(a) gives the example of a BF-WSN, where the numbers in the brackets above the nodes denote their initial energy and charging rate, and the energy for transmitting and receiving a packet are assumed to be 1 for simplicity. If we set $q = 30$, then by choosing 30% of the nodes at each level, the algorithms in [1] may generate the aggregation tree and schedule the communication tasks as in Fig.1(b). However, there are a large number of nodes that cannot be covered by the aggregated nodes, which may greatly reduce the accuracy of the aggregation result [6]. In addition, if we just consider the coverage requirement q and generate the aggregation tree as in Fig.1(c), the aggregation latency will be much less, but the aggregated nodes also cannot cover the whole network.

To address the above problems, we study the coverage aware MLAS problem in multi-hop BF-WSNs in this paper. To guarantee all the nodes can be covered, we first investigate the 1-coverage MLAS problem, in which each node can be covered by at least one node which participates in the aggregation process. On the basis of the 1-coverage MLAS problem, we then propose the q-coverage MLAS problem to satisfy the arbitrary coverage requirement q (q percent of nodes to be aggregated) while the aggregated nodes are evenly distributed in the network. For these two problems, we propose the first latency-and-coverage aware scheduling algorithm by intertwining the selection of aggregated nodes with coverage requirement and the computation of a collision-free communication schedule simultaneously. For example, for the network shown in Fig.1(a), the resulted aggregation tree and communication schedule when $q = 30$ is shown in Fig.1(d), in which not only all the nodes are covered, but also the aggregation latency is comparable with that of [1]. Our main contributions are as follows.

1) The 1-coverage MLAS problem is firstly defined in BF-WSNs, in which each node can be covered by at least one aggregated node. On the basis of the 1-coverage MLAS problem, the q-coverage MLAS problem is also defined and proved to be NP-hard, in which the arbitrary coverage requirement q can be ensured while the aggregated nodes are distributed evenly.

2) For the 1-coverage MLAS problem, a parent choosing algorithm is first proposed to intertwine the selection of the aggregated nodes and the computation of a collision-free communication schedule simultaneously. Then, two latency aware algorithms are proposed to schedule the communication tasks in the bottom-up and top-down manner respectively. The theoretical latency bound of the proposed algorithm is also analyzed.

3) For the q-coverage MLAS problem, three latency-and-coverage aware algorithms are proposed to satisfy the arbitrary coverage requirement q while the aggregated nodes are evenly distributed in the networks with a low latency. Additionally, we also study how to extend the proposed methods when there are multiple channels in BF-WSNs.

4) Through extensive simulations, it is shown that the proposed algorithms have high performance in terms of both aggregation latency and coverage quality.

The rest of the paper is organized as follows. Section II introduces the related works. Section III presents the wireless network model and problem definition. Section IV and Section V describe the detailed algorithm design for the 1-coverage and q-coverage MLAS problem in BF-WSNs respectively. Section VI explains the theoretical analysis for the proposed algorithms. The simulation results are shown in Sections VII. Finally, Section VIII concludes the paper.

II. Related Works

The MLAS problem in traditional wireless sensor networks has been studied by [10]–[17], where a sensor node is assumed to be battery-powered and always awake. With this assumption, the MLAS problem is firstly proved to be NP-hard in [10], where a Shortest Path Tree (SPT) is constructed for data aggregation, and the latency bound is $(\Delta - 1) * R$, where R denotes the radius of the wireless network and Δ is the maximum node degree. By making use of a balanced SPT for data aggregation, Malhotra *et. al.* further reduced the latency in [11]. Huang *et al.* improved the latency bound to $23R+\Delta-18$ in [12] by exploiting a CDS-based (Connected Dominating Set) tree for aggregation. The latency bound is further reduced to $16R + \Delta - 11$ by [13] with an improved First-Fit scheduling algorithm. Wan *et al.* proposed a pipeline scheduling algorithm with a latency bound of $(1 + O(log(R)/\sqrt[3]{R}))R + \Delta$ in [14]. Besides the above centralized solutions, the distributed aggregation algorithm with a latency bound of $48R + 6\Delta + 16$ was proposed by Yu *et al.* [15]. The latency bound was reduced to $16R+\Delta-14$ by Xu *et al.* [16] through setting the center node as the aggregation source. After that, Bagaa *et. al.* proposed a distributed algorithm which outperforms all the above methods in [17] by choosing the parent from neighbors smartly.

When the sensor nodes work cyclically to save energy in duty-cycled wireless networks, the MLAS problem is studied by [18]–[22]. The MLAS problem in duty-cycled wireless networks was firstly proved to be NP-hard in [18], and a centralized algorithm with latency bound of $(15R+\Delta-3)*|W|$ was proposed based on a CDS-based tree for data aggregation, where $|W|$ is the length of a working cycle. The data aggregation scheduling algorithm based on a SPT tree was proposed in [19]. The latency bound was further improved to $(3R+6\Delta+O(logR))|W|$ in [20] by using a pipelined scheduling method. Recently, Chen *et al.* [21] proposed the first distributed data aggregation scheduling algorithm for duty-cycled networks, where they try to exploit all the active time slots among the neighbors to reduce the latency. The delay efficient algorithm for duty-cycled networks with multiple channels was studied by [22]. However, all the above methods are not suitable for the battery capacity constrained networks.

For BF-WSNs, the MLAS problem was firstly studied by [4], where three centralized algorithms are proposed to construct the aggregation tree and schedule BF-nodes adaptively according to their current energy conditions. But it is aimed for the whole network. To the best of our knowledge, the q-coverage MLAS problem in BF-WSNs has been only studied by [1] recently, in which they try to choose q percent of nodes for aggregation and communication. However, they only consider the latency during data aggregation. It may result in the aggregated nodes being squeezed in a small area, which is not beneficial to obtain an approximate result [6]. Except the data aggregation scheduling problem, the minimum latency broadcast scheduling and CDS construction problem in BF-WSNs are studied by [2] and [3] recently.

III. Network Model and Problem Definition

A. Network Model

Considering a multi-hop wireless network $G = (V, E)$, where $V = \{1, 2, ..., n\}$ is the set of battery-free sensor nodes deployed in a target area, and $E = \{(v, u) \mid 1 \le v, u \le n \ \& \ v \ne u\}$ denotes the one-hop neighborhood relationships among nodes (For any two nodes u and v, they are neighbors if only if $(v, u) \in E$, and thus, they can communicate with each other directly).

Similar as in [1]–[4], we assume the network time is divided into a series of time slots with fixed length τ (which is enough for one transmission, and can be set according to CC2420 [23], *i.e.*, τ = 50ms), and each node can be charged by wireless power transfer. Let δ_v be the charging rate of node v (Note that, the charging rate can also be varying with time, which will be introduced in our future work). For node $v \in V$, let $B_v[k]$ $(k \ge 0)$ denote the residual battery level of v at the beginning of k-th time slot, and $B_v[0]$ denotes the initial energy at node v and can be just set as 0. As in [2]–[4], we assume the charing rate of each node can be obtained by the sink to calculate the communication schedule for each node.

As for data aggregation operation, the energy is mainly consumed by receiving or transmitting a packet [1] (we omit the energy consumption of data processing and sensing here, which can also be added in the proposed model, *i.e.*, reserve the energy for sensing at the beginning and the energy for data processing before transmitting its own packet). Let E_r/E_t denote the consumed energy for receiving/transmitting a packet. Additionally, the battery level at each node is bounded, then we can have $B_{min} \le B_v[k] \le B_{max}$, where B_{min} and B_{max} denote the minimum residual energy and the battery capacity respectively. For simplicity, we just set B_{min} as 0 in this paper. Then, the residual battery level of v, *i.e.*, $B_v[k]$ $(k \ge 1)$, can be updated with

$$B_v[k] = \begin{cases} min\{B_v[k-1] + \delta_v, B_{max}\}, & if\ v\ is\ charging \\ min\{B_v[k-1] + \delta_v - E_r, B_{max}\}, & if\ v\ is\ receiving \\ min\{B_v[k-1] + \delta_v - E_t, B_{max}\}, & if\ v\ is\ transmitting \end{cases}$$

B. Wireless Channel and Interference Model

Data aggregation scheduling problem in BF-WSN expects a data aggregation tree routed at the sink and a collision-free transmission plan for each node in the tree. Assume $NB(v)$ denotes the set of v's neighbors. Let $Child(v)$ be the set of v's children. Node v's parent and transmitting time slot are denoted by $p(v)$ and $t(v)$, respectively. Then, we can have the following definition:

Definition 1. **(Transmission Plan)** Given node $v \in V$, let $sch(v) = [v, p(v), t(v)]$ denote the transmission plan of v, where $(v, p(v)) \in E$ and $B_v[t(v)] \ge B_{min}+E_t$ and $B_{p(v)}[t(v)] \ge B_{min}+E_r$.

Actually, it means node v is scheduled to transmit its data to its parent $p(v)$ at time $t(v)$ for data aggregation, where node v and v's parent satisfies the battery level constraint.

For wireless communication, we first consider the MLAS problem with single channel under protocol interference model (the collision occurs when the receiver hears two and more messages simultaneously) as in [1], [4] in this paper. Note that, the proposed algorithm can also be suitable for the physical interference model (the collision occurs when the receiver's SINR value is less than a certain threshold) through a little modification, which will be shown in our future work. Given two transmission plans $[u_1, p(u_1), t(u_1)]$ and $[u_2, p(u_2), t(u_2)]$, these two communication tasks are conflicted under protocol interference model if the following two conditions are satisfied: 1) $t(u_1) = t(u_2)$; 2) $p(u_1) \in INF(u_2)\ or\ p(u_2) \in INF(u_1)$, where $INF(u)$ denotes the set of nodes lying in u's interference range. This is called **Time-Collision**, since the communication tasks are conflicted by transmitting in the same time slot.

Different from traditional WSN, there exists a new kind of collision between communication tasks in BF-WSNs, which is called **Energy-Collision** [4]. As the example shown in Fig.2(a), assume the candidate transmission plan of node u and w are calculated as $[u, v, 0]$ and $[w, v, 3]$ respectively. We can find these two transmission plans are time-collision free and satisfy the battery level constraint. However, they cannot be scheduled simultaneously. If $[w, v, 3]$ is scheduled, then $[u, v, 0]$ cannot be scheduled. This is due to node u and w share a same parent v. The common parent v won't have enough energy to receive w's packet at time 3 ($B_v[3] = 0.3 \le E_r$) after receiving u's packet at time 0. Thus, one must be very careful when scheduling in BF-WSNs. Additionally, energy-collision also occurs in the two cases as in Fig.2(b) and 2(c). Generally, the transmission plan $[u_1, p(u_1), t(u_1)]$ is energy-conflicted by $[u_2, p(u_2), t(u_2)]$ if the following two conditions are satisfied: 1) $p(u_1) = p(u_2)\ or\ p(u_1) = u_2\ or\ p(u_2) = u_1$; 2) $[u_1, p(u_1), t(u_1)]$ cannot be scheduled due to the battery level constraint when $[u_2, p(u_2), t(u_2)]$ is scheduled.

C. Problem Definition

In [1], the MLAS problem with coverage requirement q in BF-WSNs is proposed to further reduce the aggregation

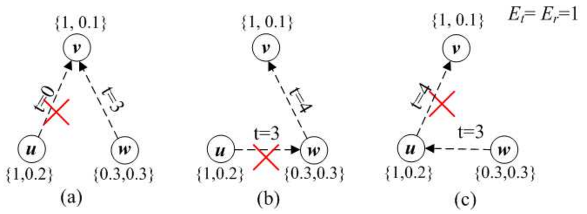


Fig. 2. Energy-collision between wireless communication tasks.

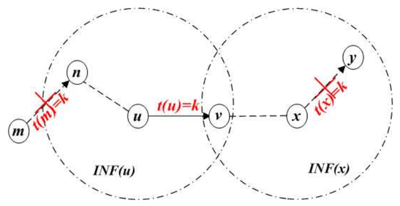


Fig. 3. The example of avoiding time-collisions.

latency, in which only a part of nodes are chosen to participate in the aggregation process. In their proposed problem, the number of aggregated nodes is required to be no less than $|V| * q$, *i.e.*, $|T| \geq |V| * q/100$, where T denotes the constructed aggregation tree, and q is the coverage requirement. However, it may result in the aggregated nodes being squeezed in a small area, which is not beneficial to obtain an approximate result [6]. Thus, to improve the aggregation quality, we consider the following q-coverage problem, which has the following two coverage requirements:

1) For $\forall u \in V$, we have $T \cap \{NB(u) \cup \{u\}\} \neq \emptyset$;
2) $|T|/|V| \geq q/100$.

Actually, the Requirement 1 requires that for any node $\forall u \in V$, either u or one of its neighbors has been included in the aggregation tree. In this case, we can guarantee all the nodes in the network can be covered by the aggregated nodes. This can greatly improve the aggregation quality by utilizing the spatial relationship between nodes. Note that, we call Requirement 1 as the 1-coverage requirement, and the MLAS problem with only Requirement 1 as the 1-coverage MLAS problem in this paper. The Requirement 2 can be used to further improve the aggregation quality by adjusting the parameter q. Obviously, when $q = 100$, it is equivalent to the traditional MLAS problem in wireless networks.

Now, we will introduce the formal definition of the q-coverage MLAS problem in BF-WSNs.

Input:

1) A BF-WSN $G = (V, E)$;
2) A sink node s, and the coverage quality requirement q $(0 < q \leq 100)$.

Output: An aggregation tree T and a collision-free (including time-collision and energy-collision) aggregation schedule $\mathcal{S} = \{\ [u, p(u), t(u)]|, \forall u \in T - \{s\}\}$, which satisfies :

1) The aggregation tree T satisfies the above two coverage requirements.
2) The data aggregation schedule has the minimum aggregation latency.

It is proved to be NP-hard in Theorem 1 since the traditional MLAS problem in BF-WSNs is NP-hard [4], which is a special case of the q-coverage MLAS problem.

Theorem 1. *The q-coverage MLAS problem in BF-WSNs is NP-hard.*

IV. Data Aggregation Scheduling for The 1-Coverage MLAS Problem

Before introducing the algorithms for the q-coverage MLAS problem in BF-WSNs, we first introduce two efficient algorithms for the 1-coverage MLAS problem in BF-WSNs, which are: 1) Bottom-Up and coverage aware data Aggregation Scheduling (BUAS) algorithm; 2) Top-Down and coverage aware data Aggregation Scheduling (TDAS) Algorithm. Note that, these two algorithms can construct the aggregation tree and schedule the communication tasks in a bottom-up and top-down manner respectively.

A. Bottom-Up and Coverage Aware Data Aggregation Scheduling Algorithm

In this subsection, we will first introduce the BUAS algorithm for the 1-coverage MLAS problem in BF-WSNs. Note that, BUAS can select the aggregated nodes and compute a collision-free communication schedule simultaneously in a bottom-up manner.

Before introducing the method, we first introduce four bit vectors [4] for collision-avoiding (including time-collision and energy-collision). Then, with these special structures, we would introduce a Candidate Transmission Plan Calculation algorithm to choose the nodes participating in the aggregation tree and determine the transmission plans of these nodes. Last, we introduce the BUAS scheduling algorithm to construct a latency and coverage aware aggregation tree for the 1-coverage MLAS problem in BF-WSNs.

1) Special Structures to Avoid Collisions between Wireless Communications Links: For any node $u \in V$, we maintain the following four bit vectors in the proposed algorithm:

i. **Forbidden-to-Receive (F2R)** $F2R_u$. If $F2R_u[k] = 1$ $(k \geq 0)$, it means u cannot receive messages at time k for time-collision avoiding.
ii. **Forbidden-to-Transmit (F2T)** $F2T_u$. If $F2T_u[k] = 1$ $(k \geq 0)$, it means u cannot transmit messages at time k for time-collision avoiding.
iii. **Slot-to-Receive (S2R)** $S2R_u$. If $S2R_u[k] = 1 (k \geq 0)$, it means u receives a packet at time k.
iv. **Forbidden-to-Use (F2U)** $F2U_u$. If $F2U_u[k] = 1$ $(k \geq 0)$, it means u cannot receive messages at time k for energy-collision avoiding.

The F2R/F2T vectors denote whether a node can receive/transmit in time slot k for time-collision avoiding, while the S2R and F2U vectors are used for energy-collision avoiding. In the following, we will introduce how to set these vectors during data aggregation.

First, to avoid time-collision during wireless communication, the F2R/F2T vectors are used to forbid the interfered nodes to receive/transmit in time slot k respectively. For example in Fig.3, when the transmission plan $sch(u) = [u, v, k]$ is scheduled, then we do as follows:

$$F2R_n[k] = 1,\ if\, n \in INF(u)\ \&\ sch(u) \in \mathcal{S} \tag{1}$$

$$F2T_x[k] = 1,\ if\, v \in INF(x)\ \&\ sch(u) \in \mathcal{S} \tag{2}$$

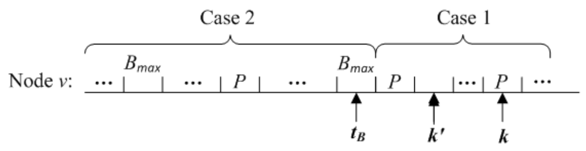


Fig. 4. The example of F2U calculation, where 'P' denotes that the time slot is scheduled to receive a packet.

Actually, it means when the transmission plan $sch(u) = [u, v, k]$ is scheduled, the following nodes cannot receive/transmit in time slot k : 1) The nodes in u's interference range, *i.e.*, node n in Fig.3, they cannot receive a message in time slot k to avoid themselves being conflicted; 2) The nodes whose interference range contains the receiving node v, *i.e.*, node x in Fig.3, they cannot transmit a message in time slot k to avoid v being conflicted.

Next, we introduce how to avoid energy-collision with the S2R and F2U vectors.

Assume node v has been scheduled to receive a message from node w in time slot k, *i.e.*, $sch(w) = [w, v, k]$. At this time, another node u also wants to choose v as its parent in time slot k' ($k' < k$) as in Fig.2(a), where the time slot k' satisfies $F2T_u[k'] = 0 \,\&\, F2R_v[k'] = 0$ and $B_u[k'] \geq E_t + B_{min} \,\&\, B_v[k'] \geq E_r + B_{min}$. This does not mean $sch(u) = [u, v, k']$ can be just scheduled since it may result in the parent v doesn't have enough energy for receiving w's message at time k, and then the existing transmission plan $sch(w) = [w, v, k]$ being failed.

To handle this case, we design the F2U vector to reflect whether time slot k' ($k' < k$) can still be used by node v to receive some other node's message. When the transmission plan $sch(w) = [w, v, k]$ is scheduled, the S2R vector of the receiver v can be just updated as :

$$S2R_v[k] = 1, if\ p(w) = v\ \&\ sch(w) \in \mathcal{S} \tag{3}$$

In addition, since node v cannot receive messages in time slot k any more, we can also have:

$$F2U_v[k] = 1, if\ p(w) = v\ \&\ sch(w) \in \mathcal{S} \tag{4}$$

As for any time slot k' ($k' < k\ \&\ B_v[k'] \geq E_r + B_{min}$) (Note that, if $B_v[k'] < E_r + B_{min}$, we can just set $F2U_v[k'] = 1$), do:

$$F2U_v[k'] = 1, if\ \exists k \in R_v(k')\ \&\ B_v[k] - E_r - \gamma_k^{k'} < B_{min} \tag{5}$$

where $R_v(k') = \{t | t > k'\ \&\ S2R_v[t] = 1\}$ and $\gamma_k^{k'}$ denotes the reduced battery level in time slot k if node v wants to receive a message at time k', which can be calculated as follows [4]:

- Case 1: $\gamma_k^{k'} = E_r$ if $t_B \leq k' < k$, where t_B denotes the last time slot that the battery level is full as in Fig. 4.
- Case 2: $\gamma_k^{k'} = max\{E_r - \sum_{t \in S_{k'}} (\delta_v + B_v[t] - B_{max}), 0\}$ if $0 \leq k' < t_B$, where $S_{k'} = \{t | S2R_v[t] = 0\ \&\ B_v[t] \geq B_{max} - \delta_v \,\&\, k' \leq t < k\}$.

Similarly, for the sender w as in Fig.2(b), if it is scheduled to transmit a message in time slot k, we can also set the F2U vector of w as follows for time slot k' ($k' < k\ \&\ B_w[k'] \geq E_r + B_{min}$):

$$F2U_w[k'] = 1, if\ B_w[k] - E_t - \gamma_k^{k'} < B_{min} \tag{6}$$

Note that, as for the scenario as in Fig.2(c), one can just set $F2U_u[k] = 1$ if $B_u[k] < E_t + B_{min}$ after it has been scheduled to receive a message in time slot k' ($k' < k$).

2) Candidate Transmission Plan Calculation: Before introducing the BUAS method, we first introduce a concept of *Candidate Transmission Plan* (CTP), which is used to choose the nodes participating in the aggregation tree and determine the transmission plan of these nodes with the above bit vectors. Let $\boldsymbol{CTP(u)} = [u, p(u), t(u), w(u)]$ be node u's CTP, where $p(u)$ and $t(u)$ denote its candidate parent and transmitting time slot respectively. The weight $w(u)$ is used for choosing the nodes participating in the aggregation tree, which will be introduced later. Let $L(u)$ denote the level of u, which can be obtained by a breadth-first search from the sink node s. Assume $NB_{up}(u) = \{v \mid L(v) < L(u), \forall v \in NB(u)\}$ denote the set of neighbors whose level is less than u. For node u, we try to choose an upper level neighbor who can receive u's message earliest as its parent. This parent and chosen time slot will be set as node u's candidate parent and transmitting time slot. It mainly works as follows.

First, for data-freshness guaranteed, node u's children cannot be chosen as the parent of u, and node u can only transmit its packet after it has received all the packets of its children. Thus, we set the Earliest Transmitting Time (ETT) of u, *i.e.*, ETT_u, as $ETT_u = min\{k | B_u[k] \geq E_t + B_{min}\ \&\ k > t_{ch}(u)\}$, where $\boldsymbol{t_{ch}(u)} = max\{t \mid t = sch(v).t(v), v \in Child(u)\}$ denotes the largest transmitting time slot of its children. Note that, the condition $B_u[k] \geq E_t + B_{min}$ is used to make sure node u has enough energy at time k.

Second, for each neighbor v ($v \in NB_{up}(u)$), we try to calculate its smallest time to receive u's packet. Note that, such time slot must be time-collision and energy-collision free from other transmission plans. Let such smallest time slot from neighbor v be $\boldsymbol{t_v(u)}$. Note that, the neighbor v may have been scheduled. In this case, the time slot $t_v(u)$ must satisfy that $t_v(u) < sch(v).t(v)$, where $sch(v).t(v)$ denote v's scheduled transmitting time slot. This is because node u must transmit the packet to v before v transmitted its message. With the four bit vectors in above subsection, $t_v(u)$ can be calculated as:

- If node v has not been scheduled, then $t_v(u) = min\{k\ | F2R_v[k] = 0 \& F2T_u[k] = 0 \& F2U_v[k] = 0 \& k \geq ETT_u\}$. Note that, the condition $F2R_v[k] = 0 \& F2T_u[k] = 0$ is used for avoiding time-collision from other transmission plans, and the condition $F2U_v[j] = 0 \& j \geq ETT_u$ is used for energy-collision avoiding and data-freshness guaranteed.
- Otherwise, we can have $t_v(u) = min\{k\ | F2R_v[k] = 0 \& F2T_u[k] = 0 \& F2U_v[k] = 0 \& sch(v).t(v) \geq k \geq ETT_u\}$. Note that, if there does not exist such a time slot, then we can just set $t_v(u) \leftarrow \infty$.

Actually, for each neighbor v ($v \in NB_{up}(u)$), we can calculate a CTP for node u. Let $CTP_v(u)$ denote node u's CTP from neighbor v. Then, we can have $CTP_v(u) \leftarrow [u, v, t_v(u), 0]$. To reduce the latency, the neighbor which can provide the smallest time slot $t_v(u)$ will be picked as the candidate parent, *i.e.*, $CTP(u) = argmin_{t_v(u)}\{CTP_v(u), \forall v \in NB_{up}(u)\}$.

For example in Fig.1(d), as for node n_{10}, we can first calculate its earliest transmitting time slot as $ETT_{n_{10}} = 7$ since $B_{n_{10}}[7] = 1.0 \geq E_t$ and $t_{ch}(n_{10}) = 0$. In addition, we can find it has two upper level neighbors, *i.e.*, n_5 and n_6. Then, node n_{10}'s smallest transmitting time slot to n_5, *i.e.*, $t_{n_5}(n_{10})$, can be calculated as $t_{n_5}(n_{10}) = 7$ according to the above analysis. Similarly, as for neighbor n_6, we can have $t_{n_6}(n_{10}) = 10$. Since n_5 can provide a smaller time slot, then it will be chosen as n_{10}'s candidate parent, *i.e.*, $CTP(n_{10}) \leftarrow [n_{10}, n_5, 7, 0]$.

Now, the CTP for node u in BF-WSNs, *i.e.*, $CTP(u)$, can be obtained. For simplicity, we call this algorithm as CTP Computation Algorithm in this paper.

3) The BUAS Scheduling Algorithm: Here, to satisfy the 1-coverage requirement, we utilize the concept of Connected Dominating Set (CDS) to construct the aggregation tree. In a CDS, a maximal independent set is first chosen to cover all the nodes in the network G (called dominators), and then a minimum number of intermediate nodes are chosen as the connecting nodes between two dominators (called connectors). The left nodes which are dominated by a dominator are called dominatees. However, one cannot just exploit the existing methods to generate the CDS here, which may result in a huge latency in BF-WSNs.

In this subsection, we will introduce the method to construct a CDS with low latency in BF-WSNs by intertwining the construction of CDS and the computation of a collision-free aggregation schedule simultaneously.

In the proposed algorithm, each node has four states: Dominator, Connector, Dominatee, and White. Initially, all the nodes in the network G are marked White. Let V^l denote the set of nodes at level l, and let V^l_{con}, V^l_{dom} and V^l_{white} denote the set of Connector, Dominator, and White nodes at level l in the network G respectively.

The proposed BUAS algorithm is executed level by level in a bottom-up manner, where the following two steps are executed intermediately from level R to 1, where $R = max\{L(u), \forall u \in V\}$ denotes the maximum level in the network.

First, we try to schedule all the Connector nodes at level l (*i.e.*, the parent node of the chosen dominators). For each node $u \in V^l_{con}$, we try to choose a Dominator or White neighbor with minimum latency from its upper level as its parent. Note that, the White neighbors can also be picked as the parent of the connectors. This is because the White neighbors have not been dominated by any dominators yet and can still be chosen as the dominator. It works as follows:

i. For each node $u \in V^l_{con}$, we first calculate a CTP with the algorithm introduced in the above subsection. To find the parent for the connectors, we make a little modification to the CTP Computation Algorithm, where we only choose the upper level neighbors who are marked Dominator or White as its candidate parent.
ii. Let the connector with maximum transmitting time be x (The node with larger latency will be scheduled early). Schedule x according to its CTP, *i.e.*, $\mathcal{S} \leftarrow [x, CTP(x).p(x), CTP(x).t(x)]$.
iii. If $CTP(x).p(x)$ is marked White, marked it Dominator (which will be scheduled lately). For all the white neighbors of $CTP(x).p(x)$ in the wireless network G, marked them Dominatee.
iv. Remove x from V^l_{con}, *i.e.*, $V^l_{con} = V^l_{con} - \{x\}$. Update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors as introduced in the above subsection.

The above four steps repeat until V^l_{con} is empty. Note that, the connector u may cannot find such a dominator as its parent in some cases, and we will introduce the method to handle it laterly.

Second, we try to schedule all the Dominator nodes at level l, *i.e.*, the nodes in $V^l_{dom} \cup V^l_{white}$. Note that, V^l_{dom} denotes the set of dominators which has been chosen by the above step, and V^l_{white} denotes the set of white nodes which can be picked as the dominators. It works as follows:

i. For each node $u \in V^l_{dom} \cup V^l_{white}$, calculate a CTP with the algorithm introduced in the above subsection. Note that, we can choose any node at its upper level as its parent here. This is because we want to choose a connector parent for the dominators. If u is a dominator node, its upper level neighbors will be either connector or dominatee nodes. And if u is a white node, its upper level neighbors will be dominatee, connector, white nodes. Note that, the dominatee or white neighbors can also be picked as a connector.
ii. To reduce the number of dominators, we try to choose the node with more uncovered neighbors as the dominator. Let $NB_{White}(u)$ denote the set of u's neighbors which are uncovered by any dominators. Thus, we set the weight of the calculated CTP, *i.e.*, $w(u)$, as $w(u) = |NB_{White}(u)|$.
iii. Let the node with the maximum weight be x (The node with larger weight will be scheduled early). Note that, the White node will be scheduled prior, since it has more uncovered neighbors, while the dominator node has none uncovered neighbors, *i.e.*, $|NB_{White}(u)| = 0$. Schedule x according to its CTP, *i.e.*, $\mathcal{S} \leftarrow [x, CTP(x).p(x), CTP(x).t(x)]$.
iv. If x is a white node, mark it dominator and all the white neighbors of x Dominatee. If $CTP(x).p(x)$ is marked dominatee or white, mark it connector (which then can be scheduled by the above step).
v. Remove x and x's white neighbors from $V^l_{dom} \cup V^l_{white}$. Update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors as introduced in the above subsection.

The above steps repeat until all the nodes in $V^l_{dom} \cup V^l_{white}$ are either scheduled or marked in the wireless network G. Note that, for any node $u \in V^l_{dom} \cup V^l_{white}$, we can always find such a neighbor as its parent, which is shown in Lemma 1, and the proof is omitted here.

Lemma 1. *For any node $u \in V^l_{dom} \cup V^l_{white}$, it can always find a Connector node as its parent.*

After we scheduling all the connectors and dominators at level 1, the aggregation tree from level R to level 1 is constructed. As for the dominators and connectors at level 1 in the wireless network G, we can just pick the sink node s as their parent. That is, for each node $u \in V^1_{con} \cup V^1_{dom}$, we can just use the above CTP Computation Algorithm to calculate a CTP from the sink s and schedule them one by one according to their CTPs.

As we can see, if every dominator and connector node can find a parent with the above algorithm, then the constructed aggregation tree forms a CDS of the wireless network G.

However, in some cases, the connector node at level l may cannot find a dominator node as its parent in the wireless

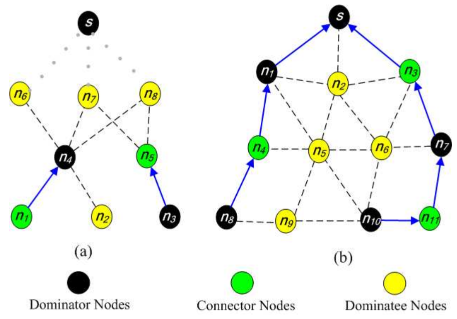

s
n_1 n_2 n_3 n_4 n_5 n_6 n_7 n_8
(a)
s
n_1 n_2 n_3 n_4 n_5 n_6 n_7 n_8 n_9 n_{10} n_{11}
(b)
Dominator Nodes
Connector Nodes
Dominatee Nodes

Fig. 5. Two special cases of BUAS algorithm.

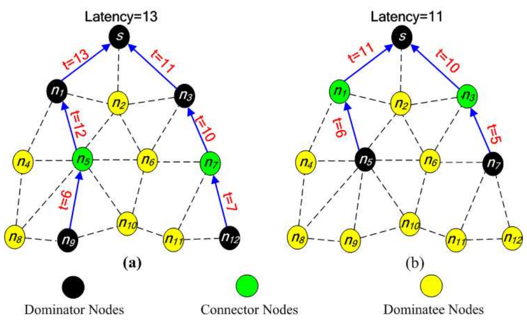


Fig. 6. The aggregation result of BUAS and TDAS algorithm.

network G. For example in Fig. 5(a), when the connector node n_1 chooses n_4 from its upper level neighbors as its dominator parent, and the dominator node n_3 choose n_5 as its connector parent. In this case, when we try to schedule the connector node n_5, we may find, all its upper level neighbors has been dominated. As a result, there is no White node or Dominator node can act as its parent. In this case, we choose a dominatee neighbor which has the smallest latency as its parent. Note that, its parent will still be marked Connector in this case. In addition, we mark the sink s as a dominator in general. This may result in two dominators being connected if some node at level 1 is also picked as the dominator. For example as in Fig. 5(b), node n_1 which is at level 1 is also picked as the dominator by the algorithm.

Although the BUAS algorithm can not return a true CDS of network G with low latency, it can guarantee that the 1-coverage requirement is satisfied (which will be shown later).

As for the BF-WSN shown in Fig. 1(a), the generated aggregation tree by BUAS is shown in Fig. 6(a). We can see only 13 time slots are needed for the wireless network G. But it also results in more nodes participated in the aggregation tree, which may result in a larger latency.

B. Top-down Coverage Aware Data Aggregation Scheduling Algorithm

In this subsection, we will introduce a Top-Down and coverage aware data Aggregation Scheduling algorithm (TDAS), which can construct a true, latency-aware and collision-free CDS for the 1-coverage MLAS problem in BF-WSNs.

Different from the BUAS method, there are two steps in TDAS: 1) The selection of dominators are conducted in a top-down manner level by level; 2)Then, a TDAS scheduling algorithm is executed to construct a latency and coverage aware CDS for the 1-coverage MLAS problem.

1) The Selection of Dominators: Since the failure of constructing a CDS in BUAS is mainly due to the selection of dominators. Thus, in this subsection, we introduce a method to choose the latency-aware dominators for the wireless network G in a top-down manner at first.

Similar as in BUAS, each node has four states in TDAS: Dominator, Connector, Dominatee and White. And all the nodes in the network are marked white initially. Note that, since the latency is mainly caused by the charging delay in BF-WSNs, we try to choose the node with more charging rate as the dominators. It mainly works follows.

Initially, the sink node s is marked as a Dominator, and all its neighbors at level 1 will be marked Dominatee.

Then, for each level l ($2 \le l \le R$), do:

i. Let x be the white node with the smallest charging rate in V^l_{white}. Here, we try to avoid choosing x as the dominator in the wireless network G.
ii. Let w be the node with the largest charging rate in $\{u \mid u \in NB(x) \& u \in V^l_{white}\}$. Note that, if there are two nodes that have the same largest charging rate, the one with more unscheduled neighbors will be chosen.
iii. Mark w Dominator, and mark all the white neighbors of w Dominatee;
iv. Remove w and w's neighbors from V^l_{white};

The above steps repeat until there are no white nodes at level l. Then, we will go to the next level, and do the above steps again.

Finally, when all the nodes at level R are marked, the whole dominators are chosen for network G. In the following, we will introduce the method to construct a latency aware CDS with these chosen Dominators for the wireless network G.

2) The TDAS Scheduling Algorithm: Different from BUAS, the dominators have been established in TDAS. Thus, we only need to find the connectors between two dominators in this case. Note that, the selection of connectors and the computation of a collision-free aggregation schedule is also conducted simultaneously here, which mainly works as follows.

Note that, the CDS is also constructed in a bottom-up manner level by level in TDAS. We execute the following two steps intermediately from level R to 2 for the wireless network G.

First, for each dominator node $u \in V^l_{dom}$, we try to choose a node between u and its upper level dominators with low latency as its connector parent. To choose a connector with low latency for dominator u, we exploit a two-step searching technique here. It works as follows:

i. For dominator u, calculate a CTP from each neighbor $v \in NB_{up}(u)$ as in the CTP Computation Algorithm, *i.e.*, $CTP_v(u)$. Then, set $ETT_v = max\{ETT_v, CTP(u).t(u) + 1\}$. Then, with this, node v can calculate its own CTP in BF-WSNs.
ii. For each node $v \in NB_{up}(u)$, calculate a CTP from its dominator neighbors in the wireless network G. Then, node v's smallest time to reach a dominator node is $CTP(v).t(v)$.
iii. Let the neighbor of u with minimum $CTP(v).t(v)$ be x, *i.e.*, $x = argmin_v\{CTP(v).t(v), \forall v \in NB_{up}(u)\}$. Then node x will be chosen as u's connector. Schedule u according to its CTP from node x, *i.e.*, $\mathcal{S} \leftarrow [u, x, CTP_x(u).t(u)]$. Note that, we cannot schedule the connector x here since

x may be chosen as the connector by other nodes again.

iv. Update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors as introduced in the above subsection.

The above four steps repeat until all the nodes in V^l_{dom} are scheduled. Note that, the dominator u can always find such a connector neighbor as its parent, which will be shown later.

Second, for each node $u \in V^{l-1}_{con}$, we try to choose a dominator neighbor at its upper level in the wireless network G as its parent. It mainly works as follows:

i. For each node $u \in V^{l-1}_{con}$ ($R \geq l \leq 2$), calculate a CTP from its dominator neighbors in the wireless network G. Since we need to choose a dominator neighbor as its parent, we make a little modification to the CTP Computation Algorithm, where we only choose the upper level neighbors who are marked Dominator as its candidate parent.
ii. Schedule u according to its CTP, *i.e.*, $\mathcal{S} \leftarrow [u, CTP(u).p(u), CTP(u).t(u)]$.
iii. Update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors as introduced in the above subsection.

The above three steps repeat until all the nodes in V^{l-1}_{con} are scheduled. Similarly, the connector u can always find such a dominator neighbor as its parent, which will be shown later.

Finally, when we schedule all the dominators at level 2 and the connectors at level 1, the latency-aware and collision-free CDS is constructed. The correctness of TDAS will be analyzed laterly.

For the BF-WSNs shown in Fig.1, the resulted aggregation tree and communication schedule after executing TDAS is shown in Fig.6(b). Compared to BUAS, it forms a true CDS of the wireless network G with fewer nodes and less latency.

V. Data Aggregation Scheduling for the q-Coverage MLAS Problem

To further improve the aggregation quality, we propose the the q-Coverage MLAS problem in BF-WSNs, which can not only satisfy the arbitrary coverage requirement q, but also guarantee the aggregated nodes are evenly distributed in the wireless network G.

In this section, we will introduce three algorithms for the q-coverage MLAS problem in BF-WSNs. First, we will introduce two algorithms based on the proposed BUAS and TDAS algorithm, which are called BUAS-q and TDAS-q in this paper. Then, we will introduce a more efficient Latency and Coverage aware Aggregation Scheduling (LCAS) algorithm without any waiting time for the q-coverage MLAS problem in BF-WSNs.

A. The BUAS-q and TDAS-q Scheduling Algorithm

Note that, the above BUAS and TDAS algorithms have constructed a backbone of the aggregation tree. Thus, we just need to add more nodes into the aggregation tree to satisfy the q-coverage requirement. In addition, to improve the coverage quality, we try to make the new adding nodes evenly distributed in the wireless network G.

Since the set of dominators generated by the above BUAS and TDAS algorithm can cover the whole network, here, we try to assign these new adding nodes to each dominator. For each dominator u, let N_u denote the number of u's neighbors who are required to participate in the aggregation process. Then the following theorem can guarantee the q-coverage requirement.

Theorem 2. *For any dominator node u in the constructed backbone, if $N_u = \lceil |D(u) \cup \{u\}| * q \rceil - 1$, then the q-coverage requirement can be guaranteed, where $D(u) \subseteq NB(u)$ denotes the set of neighbors which are dominated by u.*

Proof: Let $V_{dom} = \{u | u \in V^l_{dom}, 0 \leq l \leq R\}$ denote the set of dominators in the wireless network G. Since these dominators can cover all the nodes in the wireless network G, then we can have $\sum_{u \in V_{dom}} |\{u\} \cup D(u)| = |V|$ (Note that, if node w is dominated by several dominator nodes, we choose the one which first dominate w as its dominator). Since the aggregation tree T is constructed by the dominators and its dominated neighbors, then we can have $|T| \geq \sum_{u \in V_{dom}} 1 + N_u \geq \sum_{u \in V_{dom}} \lceil |D(u) \cup \{u\}| * q \rceil \geq \sum_{u \in V_{dom}} |D(u) \cup \{u\}| * q \geq q * \sum_{u \in V_{dom}} |D(u) \cup \{u\}|$. In addition, since $\sum_{u \in V_{dom}} |D(u) \cup \{u\}| = |V|$, thus, we can have $|T|/|V| \geq q$, which means the q-coverage requirement can be guaranteed. The theorem is proved. ■

According to Theorem 2, to satisfy the q-coverage requirement, we only need to re-find $N_u - |NB(u) \cap T|$ neighbors for each dominator to be added into the aggregation tree, where $NB(u) \cap T$ denote the set of u's neighbors who have been included in the aggregation tree. For simplicity, we call these new adding nodes as Leaf nodes in this paper, although they may not be the true leaf nodes in the aggregation tree (they may also have child nodes).

To reduce the aggregation latency, we try to avoid only choose the dominator nodes as the parent of Leaf nodes. This is because if all these neighbors choose a dominator node as their parent, it will result in the dominator node needs a large charging time to receive all the packets from these neighbors. On the contrary, we choose the parent for the leaf nodes adaptively. For a leaf node v and its neighbor $u \in NB(v)$, u can be picked as the parent of v if it satisfies one of the following conditions:

- Node u has been included in the aggregation tree (marked as Dominator, Connector, or Leaf). Note that, a leaf node can also be chosen as the parent in the proposed method.
- Node u is marked dominatee but it has an upper level dominator neighbor w and w's N_w has not been satisfied yet. This condition is used to avoid adding too many nodes into the aggregation tree. Note that, if u chooses a dominatee node as its parent, its dominatee parent will also be marked Leaf in the wireless network G.

In the following, we will introduce the method to choose and add the leaf nodes into the aggregation tree level by level from R to 0.

For each dominator u at level l ($R \geq l \geq 0$), *i.e.*, $u \in V^l_{dom}$, update the variable N_u by $N_u = N_u - |NB(u) \cap T|$. Let $NB_{Leaf}(u)$ denote the set of neighbors of u who are marked leaf and have not been scheduled (the dominatee nodes which are picked as the parent by some leaf node). Then, we execute the following two steps intermittently to add and schedule u's leaf neighbors.

1) If $|NB_{Leaf}(u)| > 0$, assume x be the first node in $NB_{Leaf}(u)$, calculate a CTP from its neighbors which

Algorithm 1: The Leaf Nodes Choosing and Scheduling Algorithm

```
Input: A Dominator node $u$ in the wireless network $G$;
Output: The Transmission Plans of its Leaf neighbors in the wireless network $G$;
1  $N_u \leftarrow \lceil |D(u) \cup \{u\}| * q \rceil - 1 - |NB(u) \cap T|$;
2  $NB_{Leaf}(u) \leftarrow$ the set of neighbors of $u$ who are marked Leaf and have not been scheduled;
3  while $(|NB_{Leaf}(u)| > 0)$ || $(N_u > 0)$ do
4      while $|NB_{Leaf}(u)| > 0$ do
5          Let $v$ be the first node in $NB_{Leaf}(u)$;
6          Calculate a CTP from its neighbors for leaf $u$ with the CTP Computation Algorithm;
7          $\mathcal{S} \leftarrow \mathcal{S} \cup [v, CTP(v).p(v), CTP(v).t(v)]$;$N_u \leftarrow N_u - 1$;
8          Update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors as introduced in the above subsection;
9          Mark its parent Leaf if $CTP(x).p(x)$ is a Dominatee node, and update the $NB_{Leaf}(u)$ set;
10     if $N_u > 0$ then
11         For each unscheduled dominatee neighbor in $NB(u)$, calculate a CTP with the CTP Computation Algorithm;
12         Sorting these nodes according to its transmitting time slot;
13         Let $x$ be the node with $N_u$-th large latency;
14         $\mathcal{S} \leftarrow \mathcal{S} \cup [x, CTP(x).p(x), CTP(x).t(x)]$;$N_u \leftarrow N_u - 1$;
15         Update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors as introduced in the above subsection;
16         Mark its parent Leaf if $CTP(x).p(x)$ is a dominatee, and update the $NB_{Leaf}(u)$ set;
17 return The Transmission Plans of $u$'s Leaf neighbors in the wireless network $G$;
```

satisfies one of the above conditions with the CTP Computation Algorithm. Schedule x according to its CTP, *i.e.*, $\mathcal{S} \leftarrow [x, CTP(x).p(x), CTP(x).t(x)]$, and update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors. Mark its parent leaf if $CTP(x).p(x)$ is a dominatee, and update the $NB_{Leaf}(u)$ set. Set $N_u = N_u - 1$. The above steps repeats until $|NB_{Leaf}(u)| = 0$.

2) Else if $N_u > 0$, for each unscheduled dominatee neighbor v in $NB(u)$, calculate a CTP from its neighbors which satisfies one of the above conditions with the CTP Computation Algorithm. Sorting these nodes according to its transmitting time slot. Let x be the node with N_u-th large latency. Schedule x according to its CTP, and update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors. Mark its parent Leaf if $CTP(x).p(x)$ is a dominatee and update the $NB_{Leaf}(u)$ set. Set N_u as $N_u = N_u - 1$.

The above steps repeat until $|NB_{leaf}(u)| = 0$ and $N_u \le 0$. The detail is shown in Algorithm 1.

Finally, when all the leaf nodes are scheduled and added into the aggregation tree, we just need to schedule the backbone of the aggregation tree. Here, we exploit a simple trick. Let $\mathcal{L}_{Leaf} = max\{sch(x).t(x), \forall x \textit{ is marked Leaf}\}$ be the last transmitting time slot of leaf nodes. Then, for any node u in the backbone, one can just set $sch(u).t(u) = sch(u).t(u) + \mathcal{L}_{Leaf}$ as the new transmitting time slot of u, where $sch(u)$ denotes the transmission plan calculated by BUAS and TDAS in the above section. It can be easily verified that the obtained aggregation schedule is still conflict-free and battery level constraint guaranteed.

B. The LCAS Scheduling Algorithm with Non-Waiting time

Although the above two algorithms can return a feasible solution for the q-coverage MLAS problem in BF-WSNs. However, all the dominators and connectors can be scheduled only when all the leaf nodes have been scheduled. This may result in much waiting time for the dominators and connectors. In this subsection, we will introduce a more efficient algorithm, *i.e.*, LCAS, in which the nodes in the backbone do not need to wait for any period of time.

Its main idea is that we schedule the dominators and connectors immediately if all of its neighboring leaf nodes have been scheduled in the wireless network G. Note that, the dominator and connector nodes calculated by TDAS are used in the proposed LCAS algorithm. Specially, for each level l $(R \ge l \ge 2)$, the following three steps are executed.

First, schedule the leaf nodes at level l. For each dominator u at level l, *i.e.*, $u \in V^l_{dom}$, exploit Algorithm 1 in the above subsection to choose and schedule the leaf neighbors of u.

Second, schedule the dominator nodes at level l. For each dominator node u at level l, *i.e.*, $u \in V^l_{dom}$, calculate a CTP from its connector parent as in TDAS. Let the dominator node with the maximum transmitting time slot be x (The node with larger latency will be scheduled early). Schedule x according to its CTP, *i.e.*, $\mathcal{S} \leftarrow [x, CTP(x).p(x), CTP(x).t(x)]$, and update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors as introduced before. This process repeats until all the dominator nodes at level l are scheduled in the wireless network G.

Third, schedule the connector nodes at level $l-1$. After all the dominator nodes at level l have been scheduled, calculate a CTP for each connector node u at level $l-1$ from its dominator parent as in TDAS. Let the node with the maximum transmitting time slot be x. Schedule x according to its CTP, *i.e.*, $\mathcal{S} \leftarrow [x, CTP(x).p(x), CTP(x).t(x)]$, and update the $F2R$, $F2T$, $S2R$ and $F2U$ vectors. This process repeats until all the connector nodes at level $l-1$ are scheduled in the wireless network G.

The above three steps repeat until all the dominator and connector nodes expect the sink have been scheduled in the wireless network G. As for the sink node, we only need to exploit Algorithm 1 in the aforementioned section to choose and schedule its leaf neighbors.

Finally, the whole aggregation tree and the aggregation schedule is established. For the BF-WSNs shown in Fig.1, the resulted aggregation tree and aggregation schedule after executing LCAS is shown in Fig.1(d). Compared to [1], the aggregated nodes are evenly distributed and can cover all the nodes in the wireless network G with a comparable little latency.

C. Discussion of Scheduling with Multiple Channels

In some scenarios, each sensor node may be assigned with multiple channels [22] in BF-WSNs to reduce the interference among wireless communications. Fortunately, the proposed algorithms can still work under this scenario with a little modification. The main idea works as follows.

Different from the scenario with single channel, the aggregated nodes need to decide which channel it will use for transmitting when there are multiple channels. In this case, we can modify the definition of the Transmission Plan of node v in Section III to $sch(v) = [v, p(v), t(v), c(v)]$, where $1 \leq c(v) \leq m$ denotes the used channel of v and m denotes the number of channels in the network. In addition, the time-collision is also different here, where two transmission plans are time-conflicted only when they are on the same channel. Thus, to avoid time-collision, we need to redesign the F2R and F2T vectors.

Here, we use $F2R_{v,i}[k]/F2T_{v,i}[k]$ to denote whether node v can use channel i to receive/transmit a message in time slot k. Similar as in Section IV.A, when a transmission plan $sch(u) = [u, v, k, i]$ is scheduled as in Fig.3, then we can do : 1) $F2R_{n,i}[k] = 1,\ if n \in INF(u)\ \&\ sch(u) \in \mathcal{S}$; 2) $F2T_{x,i}[k] = 1,\ if v \in INF(x)\ \&\ sch(u) \in \mathcal{S}$. In addition, since each node can only transmit/receive one message per time slot, then if node v has transmitted/received a message on channel i, then it cannot transmit/receive messages on other available channels at the same time slot. To avoid this, one just use the F2U vector to forbid node v transmit/receive messages after it has transmitted/received a message at time k, *i.e.*, $F2U_v[k] = 1, if sch(v) \in \mathcal{S}\ \&\ t(v) = k\ or\ sch(w) \in \mathcal{S}\ \&\ t(w) = k\ \&\ p(w) = v$. With the above structures, when u wants to transmit a message to its neighbor v time-collisions free, one can choose a time slot k and channel i which satisfy $F2T_{u,i}[k] = 0\ \&\ F2R_{v,i}[k] = 0\ \&\ F2U_u[k] = 0\ \&\ F2U_v[k] = 0$. Note that, to avoid energy-collision, one can just use the Equation (5) and (6) in Section IV.A. After that, we can calculate a time-collision and energy-collision free time slot and channel for each aggregated node as in the CTP Computation algorithm. Finally, we can schedule the CTPs as in the proposed algorithms for both the 1-coverage and q-coverage MLAS problem to obtain the aggregation schedule $\mathcal{S}$. The detail will be introduced in our future work.

VI. Performance Analysis

In this section, we analyze the correctness and latency bound of the proposed algorithms for the 1-coverage and q-coverage MLAS problem in BF-WSNs.

First, we analyze the correctness of the proposed BUAS and TDAS algorithm for the 1-coverage MLAS problem in BF-WSNs, which is shown in the following lemmas and theorems.

Lemma 2. *For node $u \in V^l_{dom}$ $(R \geq l \geq 2)$, TDAS can always find a Connector as its parent.*

Lemma 3. *For node $u \in V^{l-1}_{con}$ $(R \geq l \geq 2)$, TDAS can always find a Dominator as its parent.*

Theorem 3. *The aggregation tree of TDAS forms a CDS of the wireless network G.*

Lemma 2, Lemma 3, and Theorem 3 can be easily verified, and we omit the proof here.

Theorem 4. *BUAS and TDAS generate an aggregation tree satisfying the 1-coverage requirement.*

Proof: First, we prove that the aggregation tree of BUAS can satisfy the 1-coverage requirement in the network G, which means, for $\forall u \in V$, we have $T \cap \{NB(u) \cup \{u\}\} \neq \emptyset$. Let V_{dom} denote the set of dominators generated by BUAS. If u is a dominator, we can have $T \cap \{u\} \neq \emptyset$ since all the dominators have been added into the aggregation tree. If u is not a dominator, then it has been dominated at least one node in V_{dom}. This is because if there exists a node u which has not been dominated by a node in V_{dom}, it will be marked White, and according to BUAS, the algorithm won't stop if there exist White nodes. It means if u is not a dominator, then we can have $T \cap NB(u) \neq \emptyset$. Therefore, the aggregation schedule of BUAS can satisfy the 1-coverage requirement. Second, according to Theorem 3, the aggregation tree of TDAS forms a CDS of the network G. Similar as the above analysis, we can also have the aggregation tree of TDAS can satisfy the 1-coverage requirement, which means, for $\forall u \in V$, we have $T \cap \{NB(u) \cup \{u\}\} \neq \emptyset$. The theorem is proved. ■

Lemma 4. *For any node $u \in T$, the CTP Computation Algorithm can always return a time-collision and energy-collision free CTP.*

It can be proved by exploiting the proposed four bit vectors in Subsection IV.A. Now, according to the above analysis, the correctness of the proposed BUAS and TDAS algorithm can be guaranteed in Theorem 5.

Theorem 5. *BUAS and TDAS generate a data-freshness guaranteed, time-collision and energy-collision free aggregation schedule for the 1-coverage MLAS problem in BF-WSNs.*

Next, we analyze the latency bounds of TDAS for the 1-coverage MLAS problem.

Lemma 5. *[17] Let ξ be the upper bound of the number of dominators of a dominator within its 2-hop neighborhood. Then $\xi = \lfloor 2\pi/arccos(1 + 1/\epsilon) \rfloor$, where ϵ in $[0.05, 1]$ is a small number ensuring that two nodes are not neighbors.*

Theorem 6. *The latency bound of TDAS is at most $(\xi+3)*(\varpi+1)*R+\varpi+\xi-2(\xi+1)*(\varpi+1)$, where Δ denotes the maximum degree, R is the network radius, and $\varpi = max\{\lceil \frac{E_t}{\delta_u} \rceil, \forall u \in V\}$ is the maximum charging time to transmit a packet.*

Proof: As for the 1-coverage MLAS problem, TDAS schedule the dominator and connector nodes level by level, and two kinds of communications are performed at each level: (i) the Dominator nodes at level l to their Connector parents, the latency is denoted by $\mathcal{L}^l_{DC}$, and (ii) the Connector nodes at level l to their Dominator parents, the latency is denoted by $\mathcal{L}^l_{CD}$. Since these two types of communications are conducted intermittently from level R to the first level, therefore, the data aggregation latency of TDAS, *i.e.*, $\mathcal{L}_{TDAS}$, can be calculated as $\mathcal{L}_{TDAS} = \sum^1_{l=R}(\mathcal{L}^l_{DC} + \mathcal{L}^l_{CD})$. In the following, we will analyze the latency of $\mathcal{L}^l_{DC}$ and $\mathcal{L}^l_{CD}$ respectively.

First, we prove that $\mathcal{L}^l_{DC} \leq 4*\varpi+4$ for each level l $(2 \leq l \leq R)$. According to Huang *et. al.* [12], a connector node u has at most 5 dominator neighbors. Since one of these dominator neighbors need to be acted as u's parent, thus, there are at most 4 dominator neighbors competing to be u's children. In addition, since $E_r \leq E_t$, then the number of time slots for node u to receive a packet is at most ϖ. Therefore, all the dominator neighbors can be scheduled in at most $4 * \varpi + 4$ time slots, where $4 * \varpi$ denotes the charging time for the connector to receive all its children's packets, and 4 denotes the number of time slots to avoid time-collision from other communication tasks. Thus, we can have $\mathcal{L}_{DC}^l \leq 4 * \varpi + 4$.

Second, we prove $\mathcal{L}_{CD}^l \leq (\xi - 1) * (\varpi + 1)$ for each level l ($2 \leq l \leq R$). According to Lemma 4, each dominator u has at most ξ dominator neighbors within its 2-hop neighborhood. Therefore, for each dominator node u at level l ($2 \leq l \leq R$), there are at most $\xi - 1$ connector neighbors need to be scheduled (there is one connector node needs to be acted as its parent). Thus, the latency of the communication from the connector nodes to the dominator nodes at each level l ($2 \leq l \leq R$) is at most $(\xi - 1) * (\varpi + 1)$ time slots, *i.e.*, $\mathcal{L}_{CD}^l \leq (\xi - 1) * (\varpi + 1)$. Additionally, at level 1, since the sink always has enough energy, thus we can have $\mathcal{L}_{CD}^1 \leq \varpi + \xi$.

According to the above analysis, then we can estimate the aggregation latency of TDAS as:

$$\begin{aligned} \mathcal{L}_{TDAS} &= \textstyle\sum_{l=R}^{1} (\mathcal{L}_{DC}^l + \mathcal{L}_{CD}^l) \leq \\ &(4 * \varpi + 4 + (\xi - 1) * (\varpi + 1)) * (R - 2) + 4 * \varpi + 4 + \varpi + \xi \\ &\leq (\xi + 3) * (\varpi + 1) * R + \varpi + \xi - 2(\xi + 1) * (\varpi + 1). \end{aligned} \tag{7}$$

■

Finally, we analyze the proposed three algorithms, *i.e.*, BUAS-q, TDAS-q, and LCAS, for the q-coverage MLAS problem in BF-WSNs. The correctness of these three algorithms is shown in Theorem 7, and the proof is omitted here.

Theorem 7. *BUAS-q, TDAS-q, and LCAS generate a data-freshness guaranteed, time-collision and energy-collision free aggregation schedule for the q-coverage MLAS problem in BF-WSNs.*

Theorem 8. *The latency bound of TDAS-q and LCAS is at most* $(\xi + 3) * (\varpi + 1) * R + \varpi + \xi - (2\xi + 3 - \lceil \Delta * q \rceil) * (\varpi + 1)$, *where q denotes the coverage requirement.*

Proof: To prove the latency bound of TDAS-q and LCAS in BF-WSNs, we first prove the latency of leaf nodes, *i.e.*, $\mathcal{L}_{Leaf} \leq (\lceil \Delta * q \rceil - 1) * (\varpi + 1)$. According to Theorem 2, there are at most $N_u = \lceil |D(u) \cup \{u\}| * q \rceil - 1$ neighbors need to be scheduled. Since $D(u) \subseteq NB(u)$, then we can have $N_u \leq \lceil (|NB(u)| + 1) * q \rceil - 1 \leq \lceil \Delta * q \rceil$. In addition, one of these neighbors needs to act as the connector, then there are at most $\lceil \Delta * q \rceil - 1$ leaf nodes need to be scheduled. As a result, we can have $\mathcal{L}_{Leaf} \leq (\lceil \Delta * q \rceil - 1) * \varpi + \lceil \Delta * q \rceil - 1) = (\lceil \Delta * q \rceil - 1) * (\varpi + 1)$. According to Theorem 6, we can obtain the latency of TDAS-q and LCAS is at most $(\xi+3)*(\varpi+1)*R+\varpi+\xi-2(\xi+1)*(\varpi+1)+(\lceil \Delta * q \rceil-1)*(\varpi+1) = (\xi + 3) * (\varpi + 1) * R + \varpi + \xi - (2\xi + 3 - \lceil \Delta * q \rceil) * (\varpi + 1)$. The theorem is proved. ■

Lemma 6. *The time complexity of the Candidate Transmission Plan Computation Algorithm is at most* $O(\Delta^2 * \varpi)$.

Proof: First, since the minimum number of time slots for node u to be ready to transmit a packet is ϖ, it takes at most $O(\Delta + \omega)$ time to calculate node u's earliest transmitting time. Second, node u needs to calculate a CTP from neighbor $v \in NB_{up}(u)$, *i.e.*, $CTP_v(u)$, which needs at most $O(2\Delta + (\Delta - 1) * \varpi)$ time. This is because it requires at most 2Δ time slots for node u and node v to avoid time-collisions, and at most $(\Delta - 1) * \varpi$ time slots for node v to receive all its children's packets (node v has at most $\Delta - 1$ children). Therefore, the time complexity of the Candidate Transmission Plan Computation Algorithm is $O(\Delta + \varpi + \Delta * (2\Delta + (\Delta - 1) * \varpi)) = O(\Delta^2 * \varpi)$. ■

With Lemma 6, the time complexity of the proposed algorithms can be obtained as in Theorem 9, of which the proof is omitted here.

Theorem 9. *The time complexity of BUAS and BUAS-q is at most* $O(n^2 * \Delta^2 * \varpi)$, *the time complexity of TDAS, TDAS-q, LCAS is at most* $O(n^2 * \Delta^3 * \varpi)$, *where n denotes the number of nodes in the network.*

VII. Simulation Results

In this section, we evaluate the empirical performance of the proposed algorithms for both the 1-coverage and q-coverage MLAS problem through extensive simulations. Specially, the following algorithms are compared to verify the performance of the proposed methods.

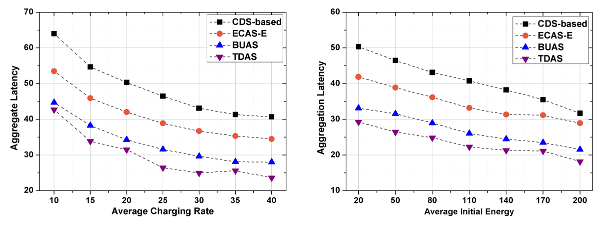
First, for the 1-coverage MLAS problem, we exploit the CDS-based algorithm as the baseline algorithm, in which a CDS is constructed as in [14], and then we exploit a scheduling method by time slot. For each time slot, we check if there exists a transmission link satisfying the battery level and time-collision constraint. The existing aggregation scheduling algorithm for BF-WSNs, which aims for the whole network, *i.e.*, ECAS-E [4], is also evaluated as a baseline algorithm.

Second, for the q-coverage MLAS problem, the recently proposed algorithm, *i.e.*, DEAS [1], is evaluated to demonstrate the efficiency of the proposed algorithms on the performance of both the coverage quality and aggregation latency. Furthermore, we also use the CDS-based and SPT-based methods as the baseline algorithms in this scenario. In these two methods, a CDS-based and a SPT-based aggregation tree is constructed respectively in the network G, and we remove some leaf nodes under the q-coverage requirement. Notice that, the existing ECAS-E [4] algorithm is also evaluated as a baseline algorithm in this scenario.

In the simulations, we mainly focus on the performance of aggregation latency and coverage quality of each method under various network topologies and coverage requirement q. Firstly, as in [1], we randomly deploy 100 BF-nodes in a 200m×200m field, where the parameters are set as $E_t = 100uJ$, $E_r = 80uJ$, $B_{min} = 0uJ$, and $B_{max} = 10mJ$ respectively. The average charging rate of each node is ranging from $10uJ/\tau$ to $100uJ/\tau$, and the initial energy at each node is ranging from $0uJ$ to $400uJ$. We set the coverage requirement q from 10% to 70% in the experiments. Secondly, as in [17], to test more network topologies, we use a network tool, *i.e.*, Networkx [24], to generate different network topologies while the number of nodes is ranging from 100 to 400. In all the simulations, each plotted point represents the average of 100 executions.

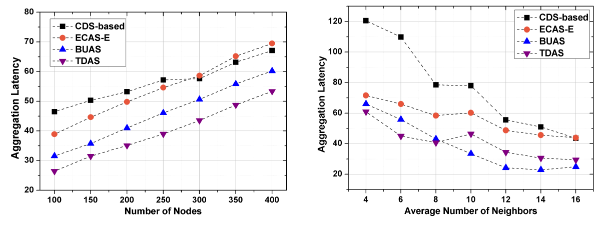
A. Aggregation Latency for 1-Coverage Problem

First, we evaluate the performance of the proposed BUAS and TDAS methods for the 1-coverage MLAS problem. Note



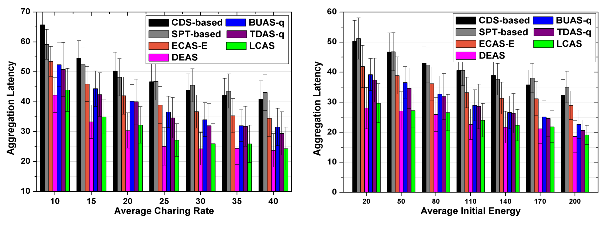
(a) Charging Rate (b) Initial Energy

Fig. 7. Aggregation latency.



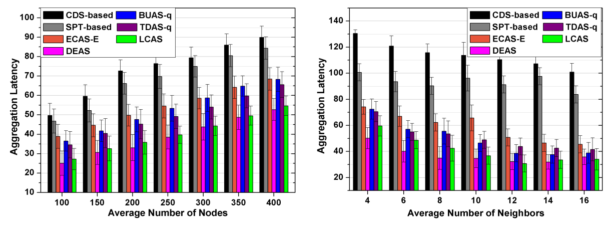
(a) Number of Nodes (b) Number of Neighbors

Fig. 8. Aggregation latency.



(a) Charging Rate (b) Initial Energy

Fig. 9. Aggregation latency.



(a) Number of Nodes (b) Number of Neighbors

Fig. 10. Aggregation latency.

that, the traditional CDS-based and the existing ECAS-E method are compared as the baseline algorithms in this group of experiments.

Fig.7 shows the aggregation latency when we randomly deploy the nodes in the target area and vary the charging rate and initial energy at each node. One can first observe that, the proposed BUAS and TDAS methods perform much better than the CDS-based and ECAS-E methods. Compared to the CDS-based method, the aggregation latency can be reduced by almost a half on average. One can also find that the latency of CDS-based method is even larger than the one of ECAS-E, which aims for the whole network. This is because many nodes may choose the same node as their parents in the CDS-based method, which may result in the parents need to charge for a long time and thus a larger latency. While the other methods try to choose the parent for each node adaptively to reduce the aggregation latency. Compared to ECAS-E, the latency of BUAS and TDAS is much less since they can choose the nodes participating in the aggregation process smartly.

Fig.8 presents the aggregation latency of each method under different network topologies, *i.e.*, network size n and average number of neighbors ψ. In both scenarios, the aggregation latency of the proposed BUAS and TDAS methods is still much less than the one of others, which demonstrates the efficiency of the proposed methods for the 1-coverage MLAS problem in BF-WSNs. An interesting finding is that the latency of CDS-based method is first larger than the one of ECAS-E when the number of nodes is less than 300. But when we continue to increase the number of nodes, the ECAS-E performs worse. This demonstrates the efficiency of utilizing the coverage aware aggregation scheduling algorithm for BF-WSNs especially when the number of nodes in the network is large. Another finding is that, the latency of BUAS decreases greatly when we increase the average number of neighbors. It even performs better than TDAS when the average number of neighbors is high. This is because BUAS can find more choices to choose the parent with low latency when the average number of neighbors is high, while TDAS can only choose a part of neighbors as the parent.

B. Aggregation Latency for the q-Coverage Problem

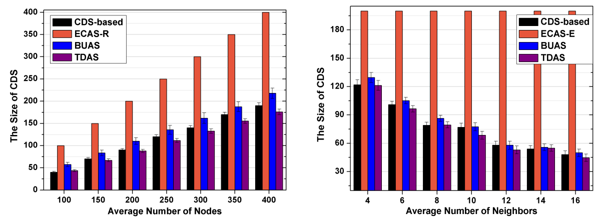
Second, we evaluate the performance of the proposed algorithms, *i.e.*, BUAS-q, TDAS-q, and LCAS, for the q-coverage MLAS problem, and q is set 35% in this group of experiments.

Fig.9 shows the aggregation latency of the proposed algorithms under different charging rate and initial energy. Firstly, we can see that LCAS and DEAS generate the least aggregation latency in both two scenarios. Although DEAS generates a little less latency than LCAS, it cannot guarantee all the nodes in the network can be covered (which will be shown later). Compared to DEAS, LCAS uses more nodes to cover the whole network under the same coverage quality q, which results in a little higher but comparable latency. Secondly, we can find that the latency of TDAS-q is larger than the one of LCAS although they share the same backbone of the aggregation tree. This is because TDAS-q schedule the leaf nodes and the backbone separately, which results in much extra waiting time of the backbone nodes.

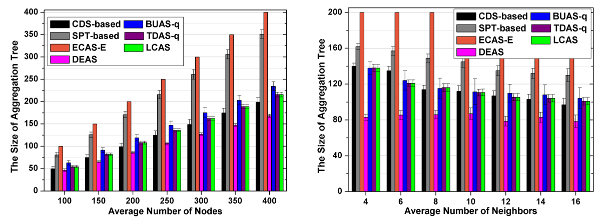
Fig.10 presents the aggregation latency of the proposed algorithms for different network size n and average number of neighbors ψ. The first observation from Fig.10 is that the CDS-based and SPT-based methods generate the largest latency in all scenarios, especially when the average number of neighbors is high, which demonstrates that the fixed-structure based aggregation scheduling methods is not efficient for BF-WSNs. One can also find that, although the CDS-based method performs better than ECAS-E for the 1-coverage MLAS problem when there are more nodes in the network, it performs worse for the q-coverage MLAS problem. This is because it just utilizes the dominator nodes to aggregate all their neighbors' message.

C. The Size of Aggregation Tree

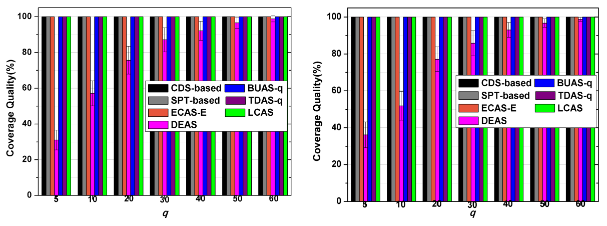
In this group of experiments, we evaluate the performance of the size of aggregation tree, *i.e.*, $|T|$, of each method. Note that, the larger size of $|T|$ means more nodes need to be aggregated and more transmissions are required in BF-WSNs.



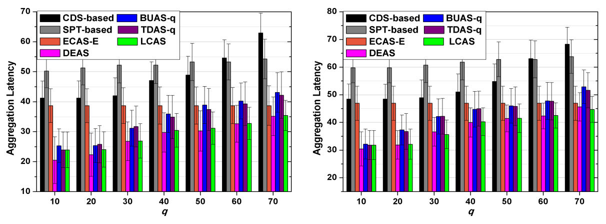
(a) Number of Nodes (b) Number of Neighbors
Fig. 11. The Size of CDS.



(a) Number of Nodes (b) Number of Neighbors
Fig. 12. The Size of Aggregation Tree.



(a) $|V|$ = 100 (b) $|V|$ = 200
Fig. 13. Coverage Quality Analysis.



(a) $|V|$ = 100 (b) $|V|$ = 200
Fig. 14. Aggregation Latency.

Fig.11 shows the size of aggregation tree of each method for the 1-coverage MLAS problem in BF-WSNs. One can find that the size of $|T|$ of ECAS-E is much higher than other methods. This is because ECAS-E needs to aggregate all the nodes in the network, which means the size of $|T|$ is equal to $|V|$. This results in it consumes at least 1 time more energy than the CDS-based method and the proposed BUAS and TDAS method. This demonstrates that the proposed coverage aware aggregation scheduling methods can not only reduce the aggregation latency but also the number of transmissions and the consumed energy. In addition, compared to BUAS, we can find the size of $|T|$ of TDAS is less. This is because BUAS cannot return a true CDS of the network G and may bring some redundant nodes in the aggregation tree.

Fig.12 shows the size of aggregation tree of each method for the q-coverage MLAS problem. The coverage requirement q is set 35% in this group of experiments. Similarly, we can find that the size of $|T|$ of ECAS-E is still the largest in this scenario. On the contrary, we can find that the size of $|T|$ of DEAS is the least compared with other methods. This is because DEAS only choose $|V| * q$ nodes to participate in the aggregation process and cannot guarantee all the nodes in the network can be covered. Specially, there are almost 70% of nodes are not covered in DEAS (which will be shown later). The SPT-based method results in a larger size of $|T|$ due to we only remove some leaf nodes in the SPT-based tree to satisfy the coverage requirement q.

D. Coverage Quality Analysis

To further understand the performance of the proposed methods, we investigate the coverage quality of each method. In this paper, we denote the coverage quality as the number of nodes which are not in the aggregation tree and have not been covered by a tree node, which is computed by $CQ(u) = \frac{|\{u \mid u \notin T \ \&\ u \notin NB(v), \forall v \in T\}|}{|V|}$.

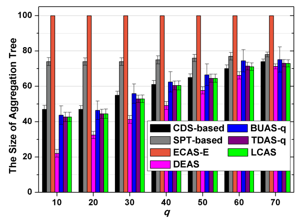
Fig.13 displays the coverage quality of each method as a function of q value, where the number of nodes is set 100 and 200 in Fig. 13(a) and Fig. 13(b) respectively. The first observation from Fig.13(a) and Fig.13(b) is that the coverage quality of each method except the DEAS method is 100%. Specially, when we set $q = 5\%$ in the experiments, the coverage quality of DEAS is only 30%, which means almost 70% of nodes are not covered. This will greatly reduce the accuracy of the aggregation result. On the contrary, by constructing the aggregation tree smartly, the proposed BUAS, TDAS, and LCAS method can not only guarantee all the nodes in the network being covered, but also obtain a small aggregation latency.

E. Performance under Different Coverage Requirement

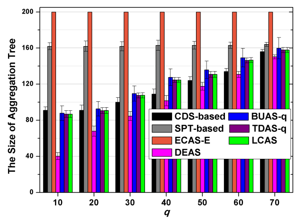
Last, we evaluate the performance of the proposed methods under coverage requirement q.

Fig.14 shows the aggregation latency of each method as a function of coverage requirement q. Compared to the CDS-based and SPT-based methods (which can also guarantee the coverage quality), the aggregation latency of the proposed methods is much less. Especially, the LCAS method can reduce the aggregation latency by almost 1 time than the CDS-based and SPT-based methods. One can also find that, the aggregation latency of each method grows with the coverage requirement q in both two scenarios. And the latency of CDS-based method grows much fast than other methods since the leaf nodes only choose the dominators as the parent.

Fig.15 shows the size of aggregation tree of each method for the q-coverage MLAS problem. One can find that the size of $|T|$ of DEAS is still the least compared to other methods, but the gap between it and the proposed methods (*i.e.*, BUAS-q, TDAS-q, LCAS) is narrow when q is large. This is because most nodes can be covered when a large number of nodes participate in the aggregation process. However, DEAS cannot guarantee all the nodes in the network can be covered even when q is set 70%. In addition, one can also find that the size of $|T|$ of each method except ECAS-E grows with the increase of coverage requirement q.



(a) $|V| = 100$



(b) $|V| = 200$

Fig. 15. The Size of Aggregation Tree.

VIII. Conclusions and Future Work

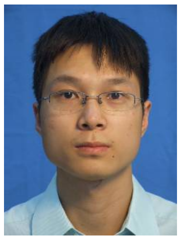
In this paper, the 1-coverage MLAS problem in BF-WSNs is firstly studied, in which each node can be covered by at least one node who participates in the aggregation process. To reduce the latency, we intertwine the selection of aggregated nodes and the computation of a collision-free communication schedule simultaneously, and two latency aware algorithms are proposed. Secondly, the q-coverage MLAS problem in BF-WSNs is also studied to satisfy the arbitrary coverage requirement q, and three latency and coverage aware algorithms are proposed.

As for future work, we will continue to study both the 1-coverage and q-coverage MLAS problem in BF-WSNs under the physical interference model, including the theoretical analysis and the empirical performance. Under such model, the transmission links are not only interfered by the neighboring nodes but also the nodes several hops away. In addition, the proposed algorithms for the 1-coverage and q-coverage MLAS problem when the charging rate of each node is varying with time will also be investigated.

References

[1] K. Chen, H. Gao, Z. Cai, et al. "Distributed Energy-Adaptive Aggregation Scheduling with Coverage Guarantee For Battery-Free Wireless Sensor Networks," In Proceedings of INFOCOM, Paris, France, 2019, pp. 1018-1026.
[2] T. Zhu, J. Li, H. Gao, Y. Li. "Broadcast Scheduling in Battery-Free Wireless Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*. vol. 15, no. 4, pp. 1–34, Oct 24, 2019.
[3] T. Shi, S. Cheng, J. Li and Z. Cai, "Constructing connected dominating sets in battery-free networks," IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, Atlanta, GA, 2017, pp. 1-9.
[4] Q. Chen, H. Gao, Z. Cai, et al. "Energy-collision aware data aggregation scheduling for energy harvesting sensor networks," In Proceedings of IEEE INFOCOM, 2018, pp. 117–125.
[5] D. Yu, L. Zhang, Q. Luo, et al. "Fast Skyline Community Search in Multi-Valued Networks," Big Data Mining and Analytics, vol. 3, no. 3, pp. 171–180, 2020.
[6] J. Li, M. Siddula, X. Cheng, et al. "Approximate Data Aggregation in Sensor Equipped IoT Networks," Tsinghua Science and Technology, vol. 25, no. 1, pp. 44–55, 2020.
[7] H. Yousefi, M. Malekimajd, M. Ashouri, and A. Movaghar, "Fast aggregation scheduling in wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 6, pp. 3402–3414, Jun. 2015.
[8] H. Lin and W. Chen, "An Approximation Algorithm for the Maximum-Lifetime Data Aggregation Tree Problem in Wireless Sensor Networks," in IEEE Trans. on Wire. Comm., vol. 16, no. 6, pp. 3787-3798, June 2017.
[9] X. Wang, Q. Li, N. Xiong, Y Pan, "Ant Colony Optimization-Based Location-Aware Routing for Wireless Sensor Networks," Wireless Algorithms, Systems, and Applications, 2008, pp. 109-120.
[10] X. Chen, X. Hu, and J. Zhu. "Minimum Data Aggregation Time Problem in Wireless Sensor Networks". In Proceedings of MSN, December 2005.
[11] B. Malhotra, I. Nikolaidis, M. Nascimento. "Aggregation convergecast scheduling in wireless sensor networks". Wire. Netw., 2010, 319-335.
[12] S. C. H. Huang, P.J. Wan, C. T. Vu, Y. S. Li, and F. Yao. "Nearly Constant Approximation for Data Aggregation Scheduling in Wireless Sensor Networks". In Proceedings of INFOCOM, pages 366-372, 2007.
[13] M. Ren, L. Guo, J. Li. "A New Scheduling Algorithm for Reducing Data Aggregation Latency in Wireless Sensor Networks." International Journal of Communication, Network and System Sciences, 3(8): 679-688, 2010.
[14] P. J. Wan, S. C. H. Huang, L. Wang, et al. "Minimum-latency Aggregation Scheduling in Multihop Wireless Networks". In Proceedings of MOBIHOC, pages 185-194, May 2009.
[15] B. Yu, J. -Z. Li, and Y. Li. "Distributed Data Aggregation Scheduling in Wireless Sensor Networks," INFOCOM, 2009.
[16] X. Xu, S. Wang, X. Mao, S. Tang, and X. Y. Li. "A Delay Efficient Algorithm for Data Aggregation in Multi-hop Wireless Sensor Networks". TPDS, 22(1): 163-175, January 2011.
[17] M. Bagaa, M. Younis, D. Djenouri, et al. "Distributed Low-Latency Data Aggregation Scheduling in Wireless Sensor Networks". ACM Trans. Sen. Netw. 11, 3, Article 49 (April 2015), 36 pages.
[18] B. Yu and J. -Z. Li. "Minimum-Time Aggregation Scheduling in Duty-cycled Wireless Sensor Networks". Journal of Computer Science and Technology, 26(6): 962-970, November 2011.
[19] N. P. K. Ha, V. Zalyubovskiy, H. Choo. "Delay-Efficient Data Aggregation Scheduling in Duty-cycled Wireless Sensor Networks". In Proceedings of RACS, October 23-26, 2012, San Antonio, TX, USA.
[20] X. Xu, J. Cao, P.-J. Wan, "Fast group communication scheduling in duty-cycled multihop wireless sensor networks" in Proceedings of Wireless Algorithms Systems and Applications, 2012, pp. 197–205.
[21] Q. Chen, H. Gao, Z. Cai, et al. "Distributed Low-Latency Data Aggregation for Duty-Cycle Wireless Sensor Networks." *IEEE/ACM Transactions on Networking (TON)*, vol. 26, no. 5, pp. 2347–2360, Oct. 2018.
[22] X. Jiao, W. Lou, S. Guo, et al. "Delay Efficient Scheduling Algorithms for Data Aggregation in Multi-Channel Asynchronous Duty-Cycled WSNs," in IEEE Transactions on Communications, vol. 67, no. 9, pp. 6179-6192, Sept. 2019.
[23] S. Guo, L. He, Y. Gu, et al. "Opportunistic flooding in low-duty-cycle wireless sensor networks with unreliable links", IEEE Trans. Comput., Vol. 63, no. 11, pp. 2787-2802, 2014.
[24] Networkx. http://networkx.lanl.gov.

Zhipeng Cai received his PhD and M.S. degrees in the Department of Computing Science at University of Alberta, and B.S. degree from Beijing Institute of Technology. Dr. Cai is currently an Associate Professor in the Department of Computer Science at Georgia State University. Dr. Cai's research areas focus on Networking, Privacy and Big data. Dr. Cai is the recipient of an NSF CAREER Award. Dr. Cai is now a Steering Committee Co-Chair for WASA. He is an editor/guest editor for Algorithmica, Theoretical Computer Science, Journal of Combinatorial Optimization, IEEE/ACM Transactions on Computational Biology and Bioinformatics. He is a senior member of the IEEE.



Quan Chen received his BS, Master and PhD degrees in the School of Computer Science and Technology at Harbin Institute of Technology, China. He is currently a lecturer in the School of Computers at Guangdong University of Technology. In the past, he worked as a postdoctoral research fellow in the Department of Computer Science at Georgia State University. His research interests include routing and scheduling algorithms in wireless networks and sensor networks.