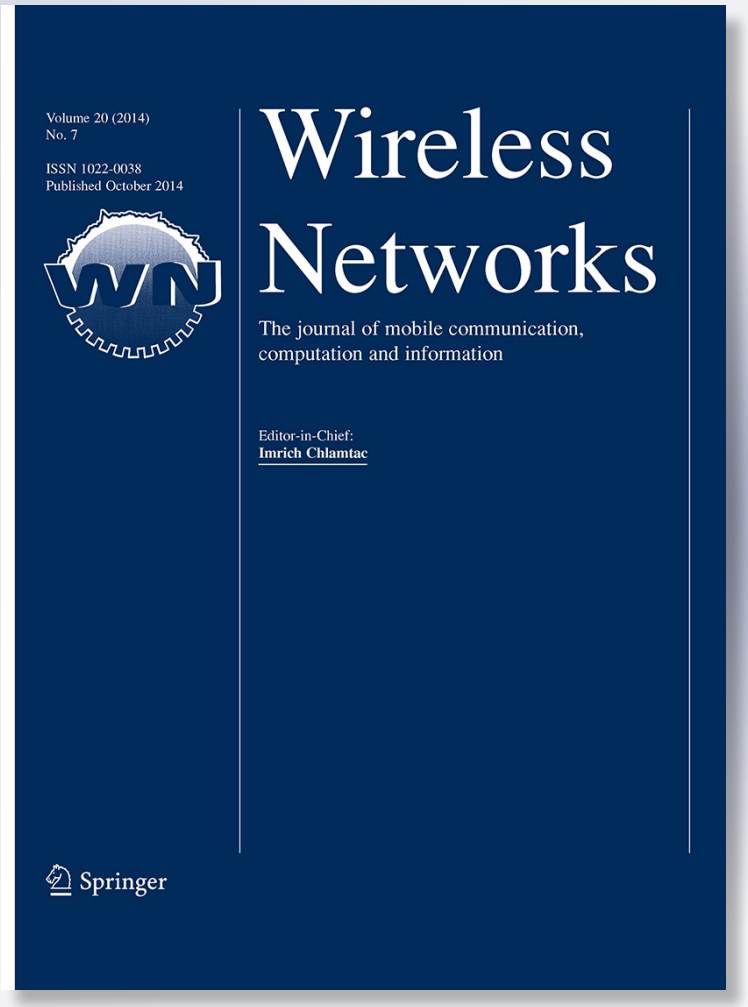
On centralized and distributed algorithms for minimizing data aggregation time in duty-cycled wireless sensor networks

Shiliang Xiao, Jingchang Huang, Lebing Pan, Yongbo Cheng & Jianpo Liu

Wireless Networks
The Journal of Mobile Communication, Computation and Information

ISSN 1022-0038
Volume 20
Number 7

Wireless Netw (2014) 20:1729-1741
DOI 10.1007/s11276-014-0706-1



Springer

Wireless Netw (2014) 20:1729–1741
DOI 10.1007/s11276-014-0706-1

On centralized and distributed algorithms for minimizing data aggregation time in duty-cycled wireless sensor networks

Shiliang Xiao · Jingchang Huang · Lebing Pan · Yongbo Cheng · Jianpo Liu

Published online: 27 February 2014

Abstract We study the problem of minimizing data aggregation time in wireless sensor networks (WSNs) under the practical duty-cycle scenario where nodes switch between active states and dormant states periodically for energy efficiency. Under the protocol interference model, we show that the problem is NP-hard and present a lower bound of delay for any data aggregation scheme. To solve the problem efficiently, we then construct a routing tree based on connected dominator set and propose two aggregation scheduling algorithms, which are the centralized Greedy Aggregation Scheduling (GAS) and the distributed Partitioned Aggregation Scheduling (PAS), so as to generate collision-free transmission schedules for data aggregation in duty-cycled WSNs. To minimize the total delay, GAS tries to achieve maximal concurrent transmissions in each time-slot during each frame by using global information, while PAS leverages a network partition based strategy and local information to ensure the largest degree of channel reuse across space and time domains. Theoretical analysis indicates that each algorithm consumes at most $O(R + \Delta)$ frames and achieves nearly constant factor approximation on the optimal delay. Here R and Δ are the network radius and the maximum node degree, respectively. We also evaluate the practicability of our algorithms by extensive simulations under various network conditions and the results corroborate our theoretical analysis.

S. Xiao (✉) · J. Huang · L. Pan · Y. Cheng · J. Liu
Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, No. 365, Changning Street, Changning District 200050, Shanghai, China
e-mail: shliangxiao@gmail.com

Keywords Data aggregation · Scheduling · Protocol interference model · Connected dominator set · Duty cycle · Wireless sensor networks

1 Introduction

Wireless sensor networks (WSNs) have long been used for gathering data from the physical world [14]. In many applications of WSNs such as environment monitoring, military surveillance and industrial control, a set of battery-powered sensor nodes situated in the targeted area are expected to sense the surroundings and report the sensory data to the sink for further analysis and utilization. The data reporting process can be viewed as an information flow from a number of nodes toward a single sink, or the so called *many-to-one communication* [16]. During the process of many-to-one communication, data may be *fused* at intermediate nodes when the sensory data are correlated or the summarized information is required, resulting in *data aggregation* [7]. Formally, data aggregation is defined as the global process of gathering data from multiple nodes, routing data through a multi-hop network and processing data along the path to the sink with the object of reducing data redundancy and power consumption [7]. The way of processing data is application-specific, which usually implies combining several data packets into a single one according to some aggregation functions such as MAX, MIN, SUM and etc.

To further reduce the energy consumed by sensor nodes, duty-cycle is employed as a common MAC technique for WSNs where nodes switch between active states and dormant states periodically [11]. Specifically, in duty-cycled WSNs (DC-WSNs), the lifespan of nodes are divided into equivalent frames, and in each frame, nodes turn off their

radios and go into sleep mode entirely for most of the time and only wake up within a short time for possible data communication through either transmitting or receiving a data packet [11, 24]. Duty-cycle helps to save large amount of energy since the working time of sensor nodes decreases significantly. Therefore, it is widely adopted as an essential component within the protocol design of many practical WSNs systems, e.g. [3, 4, 9].

For delay-sensitive applications, the time of utilizing the data is critical and out-of-date data is not only useless but sometimes even harmful [26]. At such circumstances, fast data aggregation operation is preferred. Throughout the literature, the *Minimizing Data Aggregation Time* (MDAT) problem has been richly explored [2, 6, 12, 23, 27], aiming at finding an efficient data aggregation scheme whose total time or delay, defined as the duration from the instant when the first node transmits to the instant when the sink receives the aggregated data, is minimized. Generally, a data aggregation scheme involves construction of a tree-based routing and scheduling of the link activities on the tree [6]. Although the MDAT problem has received much concentration in general WSNs, its counterpart in the more practical DC-WSNs, termed *Minimizing Data Aggregation Time in Duty-Cycled WSNs* (MDAT-DC) is largely unexplored. When duty-cycle is employed, the MDAT-DC problem becomes more intractable since not only routing and link scheduling but also the node activity scheduling must be considered. Here *link scheduling* determines which links to be active at what time under interference constraints, while n*ode activity scheduling* determines the active slot for each node during each frame. In order to achieve a minimum-delay data aggregation scheme for DC-WSNs, both link scheduling and node activity scheduling, plus routing, need to be optimized jointly. In this paper, we first show that the MDAT-DC problem is NP-hard in general, and then design provably efficient algorithms to solve this problem. Specifically, we make the contributions listed as below.

1. We construct a routing tree based on connected dominator set (CDS) [13]. Compared with previous works, our construction of routing tree differs in that it removes redundant connectors in the CDS and results in a tree with smaller number of children for each inner node. This speeds up the data aggregation process since large node degree serves as one of the bottlenecks in improving data aggregation delay, as will be verified in Sect. 5.4 detailedly.
2. We propose a centralized *Greedy Aggregation Scheduling* (GAS) algorithm and a distributed *Partitioned Aggregation Scheduling* (PAS) algorithm to coordinate the link and node activities on the constructed routing tree. To minimize the aggregation time, GAS tries to achieve maximal concurrent transmissions in each time-slot by utilizing global information while PAS exploits local information and a network partition based strategy to ensure the largest extent of channel reuse across space and time domains.
3. We conduct theoretical analysis to show that both GAS and PAS can generate collision-free transmission schedules at each time-slot during each frame for DC-WSNs, and moreover, consume aggregation time that are both upper bounded by $O(R+\Delta)$ frames, where R and Δ are the network radius and the maximum node degree, respectively. As R is a trivial lower bound, our algorithms achieve nearly constant factor approximation on the optimal delay.
4. We also evaluate the practicability of the proposed algorithms by extensive simulations under various network settings and the results further demonstrate the efficiency of our algorithms.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the network model and problem definition. The proposed routing tree construction and aggregation scheduling algorithms are provided in Sect. 4. The theoretical analysis and simulation results of the proposed algorithms are presented in Sects. 5 and 6, respectively. Finally, we conclude the paper and give some directions for future research in Sect. 7.

2 Related work

The MDAT problem is also termed Minimum Latency Aggregation Scheduling (MLAS) in the literature. Under the protocol interference model with normalized transmission and interference ranges, Chen et al. [6] prove that the MLAS problem is NP-hard and propose a data aggregation algorithm whose delay is shown to be bounded by $(\Delta-1)\cdot R$ time-slots, where R and Δ are the network radius and the maximum node degree, respectively. Since then, a number of excellent algorithms have been raised to improve the theoretical bounds of data aggregation delay. Based on maximal independent set (MIS), Huang et al. [13] propose an algorithm with a delay bound of $23R+\Delta-18$ time-slots. Since Δ contributes to an additive factor instead of a multiplicative one, their algorithm has a significant improvement compared with Chen et al.'s, especially when Δ is large. Later on, Wan et al. [23] develop three approximation algorithms that produce aggregation schedules of delay at most $15R+\Delta-4, 2R+\Delta+O(\log R)$, and $(1+O(\frac{\log R}{\sqrt{[3]R}}))R+\Delta$ respectively, based on three carefully selected connected dominator sets (CDS).

Recently, Nguyen et al. [21] further improve the delay bound to $12R + \Delta - 11$ using a new scheduling strategy based on neighboring dominators. By simultaneously executing of routing tree construction and aggregation scheduling, Bagga et al. [2] develop another improved algorithm with delay upper bounded by $\left(\lfloor\frac{2\pi}{\arccos\frac{1}{1+\varepsilon}}\rfloor + 4\right)R + \Delta - 4$, where ε is a constant between 0.05 and 1. For the case of multi-channel WSNs, Li et al. [17] also deign efficient approximation algorithms with provable delay bounds. It is necessary to clarify that all the above algorithms are centralized and hence, difficult to implement in practical WSNs. Aware of this inefficiency, Yu et al. [28] propose the first distributed data aggregation algorithm whose delay is bounded by $48R + 6\Delta + 16$, which is further improved by Xu et al. [26], with a much lower bound of $16R + \Delta - 14$.

Lately, MLAS under the physical interference model [8] also attracts much concentration from the literature and delay-efficient data aggregation algorithms are designed and analyzed as well. Xu et al. [27] develop an efficient scheduling algorithm with a latency bound of $O(\Delta + R)$ time-slots for random networks. By assuming that the maximum power levels of nodes are infinite, Li et al. [18] propose a scheduling algorithm for arbitrarily deployed networks with a latency bound of $O(\rho)$, where ρ is the *link length diversity* defined as the logarithm of the ratio between the lengths of the longest and shortest links. For the case where multiple power levels are available and the maximum power level is bounded, An et al. [1] propose a scheduling algorithm using a similar method with Xu et al.'s [27]. Another contribution of [1] is the derivation of the NP-hardness of MLAS under the physical interference model, which has not been obtained before.

Besides improving the delay performance, several works in the literature also consider the energy consumption and try to optimize both. Li et al. [19] develop an energy-efficient distributed scheduling for data aggregation based on a novel cluster-based aggregation tree, and an adaptive strategy for updating the schedule to accommodate dynamic network topology. Malhotra et al. [20] present a new tree construction method based on bipartite graph semi-matching and a new scheduling method based on a weight-priority scheme, and show that their algorithm can both improve the aggregation delay and extend the network lifetime by experiments. In addition, there exist several closely related problems with MDAT that are also widely investigated in recent years, such as the Minimum Data Collection Time (MDCT) problem [5, 14], Minimum Data Broadcast Time (MDBT) problem [25, 29], Minimum Data Gossiping Time (MDGT) problem [15], and their numerous variants.

3 Network model and problem definition

3.1 Network model

We consider a set of nodes, denoted by $\mathcal{V}$, deployed arbitrarily in a 2D plane where $v_s \in \mathcal{V}$ is the sink node. Each node is equipped with a single half-duplex transceiver and can transmit (receive) data to (from) all directions. All the nodes work in a shared wireless channel. For the network formed by the nodes in $\mathcal{V}$, we model it as a connected graph $G(\mathcal{V}, \mathcal{E})$, where $\mathcal{E}$ is the set of edges representing the communication links among nodes in $\mathcal{V}$. Given nodes $u, v \in \mathcal{V}$, edge (u, v) belongs to $\mathcal{E}$ if and only if u and v can communicate with each other directly, i.e. both u and v are within the transmission range of each other. Assume that all nodes are homogenous with the same transmission radius of r. Under the protocol interference model [10], the interference range of each node u is a circle with radius r_I centred at u and data transmission between sender v and receiver w is *collision-free* if and only if w is within the transmission range of v and meanwhile, outside the interference range of any other ongoing transmitter. r_I is also referred to as the interference radius and typically, $r_I \geq r$. For simplicity and similar to [23, 26], we assume that both r_I and r are normalized to one, i.e. the interference range of each node is the same as its transmission range.

In DC-WSNs, each node switches periodically between active state and dormant state. To be more specific, the lifespan of each node is divided into equivalent frames and each frame is further divided into equivalent time-slots. During each frame, the node only wakes up in the *active slot* for possible data communication and sleeps completely within the rest time-slots. We assume strict time synchronization and each time-slot is chosen to be long enough for transmitting or receiving a data packet. Let the *frame length*, i.e. the number of time-slots in each frame be fixed as F_0, then the *duty-cycle ratio*, defined as ratio of the active time to the total time in each frame, will be $\frac{1}{F_0}$.

3.2 Problem definition

Let $\mathcal{A}$ and $\mathcal{B}$ be two disjoint subsets of $\mathcal{V}$. We say that data is *aggregated* from $\mathcal{A}$ to $\mathcal{B}$ at time-slot j during frame i (for simplicity, we denote the time by $\{i, j\}$, where $i = 0, 1, 2, \ldots$ and $j = 0, 1, \ldots, F_0 - 1$) if all nodes in $\mathcal{A}$ transmit data to some nodes in $\mathcal{B}$ successfully, without suffering from any collision or interference. As such circumstances, the transmissions between the senders in $\mathcal{A}$ and the corresponding receivers in $\mathcal{B}$ constitute a valid schedule at time $\{i, j\}$, which is denoted by $Sche\{i, j\}$. In data aggregation, let $\{f, t\}$ denotes the time at which the sink finally obtains the aggregated packet. Then, the

MDAT-DC problem will aim to find a sequence of schedules $Sche\{0,0\}, Sche\{0,1\}, \ldots, Sche\{i,j\}, \ldots, Sche\{f,t\}$ $(0 \le i \le f;\ j = 0,1,\ldots,F_0-1$ when $i < f$ and $j = 0,1,\ldots,t$ when $i = f)$ subject to the constraints as below, such that the data aggregation time t_{DA}, calculated as $f+1$ frames (approximate value) or $f \cdot F_0 + t + 1$ time-slots (accurate value), is minimized.

1. At time $\{i, j\}$ for all i and j, data is aggregated from the senders in $Sche\{i, j\}$ to the nodes in $V\backslash$ {the union of all the sender sets in $Sche\{i', j'\}$, where $\{i', j'\}$ ranges from (0, 0) to $\{i, j\}$}. This is also known as the *causality* constraint in [13].
2. The union of the sender sets in all the schedules $Sche\{i, j\}$ where $\{i, j\}$ ranges from $\{0, 0\}$ to $\{f, t\}$ equals to $\mathcal{V}\backslash\{v_s\}$, ensuring that all data be gathered to the sink v_s at time $\{f, t\}$.
3. At any time $\{i, j\}$, the schedule $Sche\{i, j\}$ is valid and all the concurrent transmissions are ensured to be collision-free under the protocol interference model.
4. For any transmission between the sender u and the receiver v in each schedule $Sche\{i, j\}$, the active slots of u and v in frame i (denoted by $A_i(u)$ and $A_i(v)$, respectively) both equal to j and the rest slots of u and v in frame i are all dormant, meeting the duty-cycle requirement.

Note that if $F_0 = 1$, the MDAT-DC problem turns out to be the original MDAT problem, which has been proven to be NP-hard by Chen et al. [6] under the protocol interference model with $r_I = r$. Therefore, our problem, as a more general case when $F_0 \ge 1$, is also NP-hard, which leads to the following theorem showing the complexity of the MDAT-DC problem directly.

Theorem 1 *The problem of MDAT-DC is NP-hard.*

Let R be the radius of the graph G. For node u that are R-hops away from v_s, it needs at least R times of data transmission for the data at node u to reach v_s. Note that in DC-WSNs, each node can only transmit or receive at most one data packet in each frame. Therefore, each of the R transmissions will consume at least one frame and the total time to gather the data from u to v_s is lower bounded by R frames. That is, R frames is a lower bound (may not tight) to the MDAT-DC problem.

4 Data aggregation algorithms

In our solution for MDAT-DC, we first construct a routing tree, and then design two aggregation scheduling algorithms based on the tree which take into consideration the link activities on the constructed tree as well as the node activities in each frame under the duty-cycle scenario.

4.1 Routing tree construction

We adopt an existing approach proposed by Wan et al. [22] to construct a data aggregation tree $T(\mathcal{V}_T, \mathcal{E}_T)$ from the graph $G(\mathcal{V}, \mathcal{E})$ with some modifications. In [22], the authors select a connected dominator set (CDS) from $\mathcal{V}$ as the virtual backbone of an ad hoc network by the way of picking a maximum independent set (MIS) first and then a few connectors to interconnect the nodes in the MIS. We select the MIS by the same way but remove redundant connectors while interconnecting the nodes in the MIS, making the constructed CDS more reduced. The reduced CDS has its unique merit in that the node degree of the virtual backbone will be smaller and therefore, the delay of our scheduling algorithms will be more efficient, as will be analyzed in Sect. 5.4. Algorithm 1 shows the pseudo-code of the improved algorithm, the main steps of which are summarized as follows.

1. The nodes in G are initially marked WHITE and a *breadth-first-search* (BFS) rooted at the sink v_s is performed, resulting in a BFS tree named T_{bfs}. Then a MIS $\mathcal{U}$ is selected layer by layer beginning at v_s. The nodes in $\mathcal{U}$ are called *dominators* and marked BLACK. The set of the dominators in layer i of T_{bfs} is denoted by $\mathcal{U}_i$, where $0 \le i \le R$ and R is the graph radius of G.
2. We then pick some nodes in $\mathcal{V}\backslash\mathcal{U}$ to interconnect the dominators in $\mathcal{U}$. The selected nodes form a *connector* set $\mathcal{W}$. In specific, from layer 1 to layer R, for each node u in $\mathcal{U}$, its parent in T_{bfs}, denoted by $p(u)$, is selected as a connector, added to $\mathcal{W}$ and marked GREY. We then pick a dominator $d(p(u))$ for node $p(u)$ in the same or upper layer. Apparently, $p(u)$ is responsible for interconnecting the disjointed dominators u and $d(p(u))$. The corresponding edges are added into $\mathcal{E}_T$.
3. Afterwards, we remove the redundant connectors from $\mathcal{W}$. A connector w is *redundant* if and only if removing it does not make any of its dominator children in T_{bfs} be disconnected from its dominator $d(w)$. Algorithm 2 presents the pseudo-code of checking whether a connector is redundant or not. In the case that w is redundant and removed, each dominator child $c(w)$ of w is reconnected to $d(w)$ by another connector w^* which is adjacent to both $c(w)$ and $d(w)$. The removed connectors are marked WHITE again and the remaining connectors in $\mathcal{W}$ make up a reduced CDS together with the dominators in $\mathcal{U}$.

4. Finally, we connect each WHITE node (called *dominatee*) to one of its adjacent dominators randomly, resulting in a tree T, eventually. We can easily identify some properties of the tree T which are (i) each WHITE node has a parent marked BLACK, (ii) each BLACK node except the sink has a parent marked GREY and (iii) each GREY node has a parent marked BLACK.

Figure 1 shows an example of constructing a data aggregation tree by Algorithm 1. Specifically, Fig. 1(a) shows the input graph G and Fig. 1(b) shows the BFS tree T_{bfs} of G. Figure 1(c) shows the choice of a MIS, where nodes 0, 3, 4 and 8 are picked as a MIS and marked BLACK. Figure 1(d) shows the choice of connectors, where nodes 1, 2, and 5 are chosen as they are the parents of nodes 4, 3, and 8 in T_{bfs}, respectively. Then nodes 1, 2 and 5 pick nodes 0, 0 and 4 as their dominators, respectively. Note that node 1 is redundant because node 4 can use node 2 to interconnect with node 0. Hence node 1 can be removed, as showed in Fig. 1(e). Then the WHITE nodes 1, 6, and 7 are connected to their adjacent BLACK ones randomly. The produced data aggregation tree is showed in Fig. 1(f).

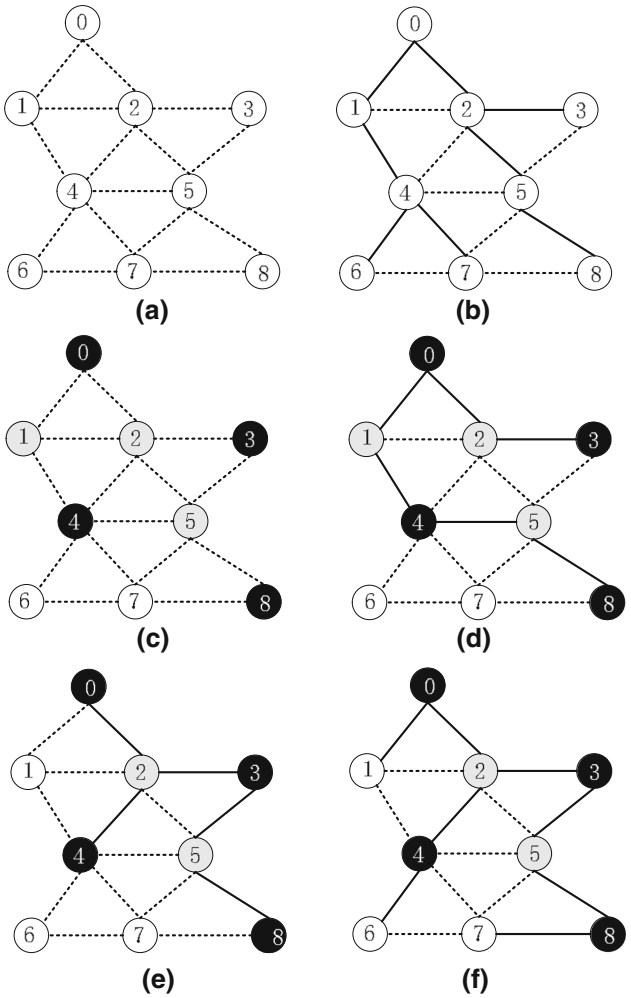


Fig. 1 An example of constructing a data aggregation tree.

Algorithm 1: Aggregation Tree Construction Algorithm

Input: $G(\mathcal{V}, \mathcal{E})$.

Output: $T(\mathcal{V}_T, \mathcal{E}_T)$.

Construct a BFS tree T_{bfs} rooted at v_s from G ;
Select a MIS $\mathcal{U}$ from $\mathcal{V}$ in G layer by layer;
$\mathcal{V}_T \leftarrow \mathcal{V}$, $\mathcal{E}_T \leftarrow \emptyset$;
for $i \leftarrow 1$ *to* R **do**
  for *each dominator* u *in* $\mathcal{U}_i$ **do**
    Add node $p(u)$ into $\mathcal{W}$ and mark it GREY;
    Add edges $(u, p(u))$ and $(p(u), d(p(u)))$ into $\mathcal{E}_T$;
  end
end
for *each connector* w *in* $\mathcal{W}$ **do**
  if RedundantConnector(w) **then**
    Mark w WHITE and remove edge $(w, d(w))$ from $\mathcal{E}_T$;
    for *each dominator child* $c(w)$ *of* w *in* T_{bfs} **do**
      Remove edge $(c(w), w)$ from $\mathcal{E}_T$, and add edges $(c(w), w^*)$, $(w^*, d(w))$ into $\mathcal{E}_T$;
    end
  end
end
for *each dominatee* v *in* $\mathcal{V}$ **do**
  Add edge $(v, d(v))$ into $\mathcal{E}_T$;
end
return $T(\mathcal{V}_T, \mathcal{E}_T)$;

Algorithm 2: RedundantConnector

Input: A connector w.

Output: TRUE if w is redundant, FALSE otherwise.

```
1 for each dominator child c(w) of w in T_bfs do
2     if (W_c(w)\{w}) ⋂ (W_d(w)\{w}) = ∅ then
3         return FALSE        /* W_i denotes node i's
          adjacent connectors */;
4     end
5 end
6 return TRUE;
```

4.2 Greedy Aggregation Scheduling

Greedy Aggregation Scheduling (GAS) is a centralized algorithm in which the sink is responsible for computing the schedule and disseminating the shedule results to all the nodes in the network by the way of broadcasting or flooding. Given that the global information of all nodes (e.g. interference relationship, node positions and schedule status) are known in prior, GAS greedily schedules as many nodes in each time-slot during each frame as possible, without violating the essential constraints stated in Sect. 3.2 Specifically, in GAS, we first schedule all the dominatees and then schedule the dominators and connectors from the bottom layer till the top layer in T. In each step of scheduling, we apply a common algorithm named *Greedy Scheduling Sub-Procedure* (GSSP) to generate a collision-free schedule sequence for an input sender set. We first introduce GSSP, the pseudo-code of which is shown in Algorithm 3.

Initially, we construct two temporary sets $\mathcal{M}_1$ and $\mathcal{M}_2$, which, respectively, contain the senders scheduled in the current time-slot of the current frame and the senders scheduled in any time-slot of the current frame. We then pick a node from the sender set $\mathcal{A}$ randomly, add it to $\mathcal{M}_1$ and remove it from $\mathcal{A}$. For each remaining node in $\mathcal{A}$, we will schedule it in the current time-slot provided that it satisfies the two conditions, as stated in Line 5 and Line 6 of Algorithm 3, respectively. We will prove by Theorem 2 in Sect. 5 that this kind of scheduling is ensured to be collision-free under the protocol interference model. By checking each node in $\mathcal{A}$ at least once, we make sure that as many as nodes be scheduled concurrently as possible. Subsequently, we accumulate the current time by one slot and repeat the above process

Algorithm 3: Greedy Scheduling Sub-Procedure (GSSP)

Input: A sender set $\mathcal{A}$, the graph $G(\mathcal{V}, \mathcal{E})$ and the data aggregation tree T.

Output: A schedule sequence $Sche\{f, t\}$.

```
1  M_1 ← ∅, M_2 ← ∅, f ← 0, t ← 0;
2  Randomly pick a node from A, add it to M_1 and
   remove it from A;
3  while A ≠ ∅ do
4      for each node v in A do
5          if ∀u ∈ M_1, p(u) ≠ p(v) && (p(u), v) ∉ E &&
           (p(v), u) ∉ E then
6              if ∀u ∈ M_2, p(v) ≠ p(u) then
7                  Add v to M_1 and remove it from A
8              end
9          end
10     end
11     Sche{f, t} ← M_1;
12     t ← t + 1;
13     if t ≤ F_0-1 then
14         M_2 ← M_2 ⋃ M_1, M_1 ← ∅;
15     else
16         M_1 ← ∅, M_2 ← ∅, t ← 0, f ← f + 1;
17     end
18 end
```

until $\mathcal{A}$ becomes NULL, i.e. all senders in $\mathcal{A}$ are scheduled to transmit their data to their parents in the aggregation tree T successfully. For each node u that is scheduled at time $\{f, t\}$, the active slot of node u in frame f will be set as t, i.e. $A_f(u) = t$.

Taking GSSP as a key ingredient, GAS schedules all nodes in the aggregation tree T sequentially. As first, the dominatees are scheduled and removed from T. The remaining tree, containing dominators and connectors only, is termed *backbone tree* and denoted by T_B. Afterwards, we schedule the nodes in T_B layer by layer. Specifically, let d_B be the depth of T_B and B_i $(0 \leq i \leq d_B)$ be the set of nodes in the i-th layer of T_B. In GAS, we schedule the senders from the bottom layer

($i = d_B$) till the top layer ($i = 1$) in T_B by always calling the sub-procedure GSSP, as shown in Algorithm 4.

Algorithm 4: Greedy Aggregation Scheduling

Input: The data aggregation tree $T(\mathcal{V}_T, \mathcal{E}_T)$, the graph $G(\mathcal{V}, \mathcal{E})$.

1 **GSSP**($\mathcal{V}\backslash(\mathcal{U}\bigcup\mathcal{W})$, G, T);

2 Remove all nodes in $\mathcal{V}\backslash(\mathcal{U}\bigcup\mathcal{W})$ from T, resulting in a backbone tree T_B;

3 **for** $i \leftarrow d_B$ *to* 1 **do**

4 | **GSSP**(B_i, G, T_B);

5 **end**

4.3 Partitioned Aggregation Scheduling

Though GAS can make efficient schedules based on global information, it is difficult to be implemented in practical distributed sensor networks due to its high computational complexity. Consequently, we propose a distributed alternative named Partitioned Aggregation Scheduling (PAS) which only needs local information and is amenable to distributed realization.

In PAS, we first use vertical lines $x = i \cdot \frac{1}{\sqrt{2}}$ for $i \in \mathbb{Z}$ and horizontal lines $y = j \cdot \frac{1}{\sqrt{2}}$ for $j \in \mathbb{Z}$ to partition the 2D plane into half-open and half-closed cells of side-length $\frac{1}{\sqrt{2}}$. Next, we color the grids such that one among every 4×4 grids is assigned with the same color. The used colors are indexed as integers between 0 and 15 and the color of node u is the same as that of the cell where u locates. That is, each node can determine its color using its relative position information in the 2D plane. This kind of *cell partition and coloring* ensures that when at most one node from every cell with a monotone color transmits (or receives) simultaneously, all the transmissions are collision-free, as will be verified by Theorem 4 in Sect. 5 We assume that each node u knows in prior some location information such as Color[u], the layer number $l(u)$ in the aggregation tree T and the number of its dominatee (respectively, dominator, connector) children, denoted by $N_{DeC}(u)$ (respectively $N_{DoC}(u)$, $N_{CoC}(u)$). Additionally, the depth of the backbone tree T_B, i.e. d_B is also available at each node. Similar to GAS, PAS schedules the dominatees first and then deals with the dominators and connectors sequentially. During the process of scheduling, PAS only allows nodes with the same color to transmit or receive concurrently. The detail of PAS is presented in Algorithm 5. Note that the aforementioned information about the tree can be acquired by each node during the tree construction process, and the relative position can be acquired by each node via some kind of localization technology. Therefore, PAS can be implemented in a distributed manner.

Algorithm 5: Partitioned Aggregation Scheduling

Input: The graph G, the data aggregation tree T.

Output: A schedule sequence $Sche\{f, t\}$.

1 Partition the 2D plane into cells with side length $\frac{1}{\sqrt{2}}$ and color the cells using 4×4 colors;

2 Each dominator u sends Color[u] and a different number from $\{1, 2, \cdots, N_{DeC}(u)\}$ to each dominatee child in T;

3 **if** *A dominatee v receives Color[u] and a number N* **then**

4 | v sets its schedule as $\{\lceil\frac{Color[u]}{F_0}\rceil \cdot (\Delta - 1) + N - 1, Color[u]\%F_0\}$;

5 **end**

6 Each connector u sends a different number from $\{1, 2, \cdots, N_{DoC}(u)\}$ to each dominator child in T;

7 Each dominator u sends Color[u] and a different number from $\{1, 2, \cdots, N_{CoC}(u)\}$ to each connector child in T;

8 **if** *A dominator v receives a number N* **then**

9 | v sets its schedule as $\{\lceil\frac{16}{F_0}\rceil \cdot (\Delta + 15 \cdot \frac{d_B - l[v]}{2}) + \lceil\frac{Color[u]}{F_0}\rceil \cdot 4 + N - 1, Color[u]\%F_0\}$;

10 **end**

11 **if** *A connector v receives Color[u] and a number N* **then**

12 | v sets its schedule as $\{\lceil\frac{16}{F_0}\rceil \cdot (\Delta + 4 + 15 \cdot \frac{d_B - l[v] - 1}{2}) + \lceil\frac{Color[u]}{F_0}\rceil \cdot 11 + N - 1, Color[u]\%F_0\}$;

13 **end**

14 Each node u transmits in time-slot t during frame f based on its schedule $\{f, t\}$;

5 Algorithm analysis

In this section, we first derive some useful properties of the data aggregation tree constructed in Sect. 4.1 Then we present theoretical analysis about the validity and time

efficiency of the aggregation scheduling algorithms GAS and PAS proposed in Sects. 4.2 and 4.3, respectively.

5.1 Analysis of the data aggregation tree

For the T produced by Algorithm 1, we denote the subtree of T which only contains dominators and connectors by T_B (also known as backbone tree that appears in Sect. 4.2). With regard to T_B, we have the following lemma.

Lemma 1 *Let d_B and R be the depth of T_B and the radius of the graph G, respectively, then we have $d_B \leq 2(R-1)$.*

Proof Since the sink v_s is a dominator, its neighbours in T_B are all connectors. Hence, the number of layers in T_B where at least one dominator exists does not exceed R (including layer 0). For every two adjacent layers of dominators, there exists one layer of connectors interconnecting the disjointed dominators. Thus the number of layers where at least one connector exists does not exceed $R-1$. Therefore, the total number of layers in T_B does not exceed $2R-1$, implying that the depth d_B does not exceed $2R-2$, i.e. $d_B \leq 2(R-1)$. This finishes the proof.

It is easy to see that in T_B, the nodes in the even layer are all dominators, the nodes in the odd layer are all connectors, and the depth d_B is an odd number. Next, we have the following lemmas which have been verified in the literature.

Lemma 2 [22] *Within a CDS of graph G, a connector is adjacent to at most 5 dominators.*

Lemma 3 [26] *Suppose that nodes u, v and w are three dominators in a CDS of graph G, and both v and w are within two hops from v. Let nodes v^* and w^* be the corresponding connectors interconnecting v and w with u, respectively. Then either v^* is adjacent to w or w^* is adjacent to v, provided that $\widehat{vuw} \leq 2\arcsin\frac{1}{4}$.*

Based on Lemma 2 and Lemma 3, we can achieve the following lemma which shows some useful property of the backbone tree T_B.

Lemma 4 *Let d_B be the depth of T_B. For each node u in layer $i-1$ $(1 \leq i \leq d_B)$ of T_B, the number of nodes in layer i that are adjacent to node u, denoted by $\Delta_i(u)$, can be bounded as below.*

$$\Delta_i(u) \leq \begin{cases} 4 & i \text{ is even} \quad (1a) \\ 11 & i \text{ is odd and } i > 1 \quad (1b) \\ 12 & i = 1 \quad (1c) \end{cases}$$

Proof According to Lemma 2, a connector is adjacent to at most 5 dominators. Note that each connector must has a dominator acting as its parent in the upper layer, so it is adjacent to at most 4 dominators in the lower layer. When i is even, the nodes in layer $i-1$ are connectors and the nodes in layer i are dominators. Therefore, for any connector u in layer $i-1$, it is adjacent to at most 4 nodes in the lower layer (i.e. layer i). That means $\Delta_i(u) \leq 4$.

According to Lemma 3, we can figure out that a dominator is adjacent to at most 12 connectors in T_B. We prove this by contradiction. Suppose that a dominator u is adjacent to n $(n \geq 13)$ connectors. For each connector, it is not removed by Algorithm 1 in the case that there exists at least one dominator which can only use it to interconnect with the corresponding dominator. So there are at least n dominators for which only one of the n connectors is available. With *drawer principle*, there exist two dominators v and w within two-hops of u satisfying $\widehat{vuw} \leq \frac{2\pi}{n} < 2\arcsin\frac{1}{4}$. Based on Lemma 3, either v^* is adjacent to w or w^* is adjacent v, where v^* and w^* are the corresponding connectors which v and w can only treat as connectors, respectively. This leads to contradiction and hence $n < 13$, indicating that at most 12 connectors is adjacent to a common dominator. Note that the nodes in the odd layer are connectors and the nodes in the even layer are dominators. Thus for the odd number i, each dominator u in layer $i-1$ is adjacent to at most 12 connectors in layer i, i.e. $\Delta_i(u) \leq 12$. Moreover, for the odd layer i when $i > 1$, node u in layer $i-1$ must have a connector parent in layer $i-2$. That means u is adjacent to at most 11 connectors in layer i when i is odd and meanwhile $i > 1$. This finishes the proof.

5.2 Analysis of GAS

Theorem 2 *Algorithm GAS produces a valid schedule.*

Proof In order to prove that the schedule produced by GAS is valid, we need to show that all the senders scheduled in the current time-slot during the current frame do not conflict mutually. In GAS, there are two cases where a sender is chosen to be scheduled. The first is shown in Line 2 of Algorithm 3, where we randomly pick a sender and schedule it. Since the picked sender is the only one in $\mathcal{M}_1$, it is surely not conflicting with others. The second is shown in Line 7 of Algorithm 3, where we schedule a sender v which satisfies both the two conditions as shown in Line 5 and Line 6, respectively. The first condition ensures that v is not conflicting with any scheduled sender in $\mathcal{M}_1$ based on the description of the protocol interference model. The second condition ensures that any receiver of the senders in $\mathcal{M}_1$ can only be activated once during a frame, meeting the duty-cycle requirement. In all, we can conclude that the schedule of GAS is valid, which finishes the proof.

Lemma 5 *In any given input set $\mathcal{A}$ in GSSP, there are at most $2\Delta - 1$ senders which are conflicting with each other and hence, any two nodes among them can not be scheduled simultaneously.*

Proof Let $\mathcal{X} \subseteq \mathcal{A}$ denote the set of senders in $\mathcal{A}$ which are mutually conflicting and $\mathcal{P}$ denote the set of the corresponding receivers of the senders in $\mathcal{X}$. We construct a bipartite graph $G_b(\mathcal{X}, \mathcal{P}, \mathcal{E}_x)$, where $\mathcal{E}_x$ is the set of edges initialized to NULL. For each node u in $\mathcal{X}$ and each node v in $\mathcal{P}$, we will add edge (u, v) into $\mathcal{E}_x$ as long as u and v are in the transmission range (or the interference range) of each other. This implies that if two nodes in $\mathcal{X}$ conflict with each other under the protocol interference model, at least one edge can be added into $\mathcal{E}_x$. Besides, the edge between each sender in $\mathcal{X}$ and the corresponding receiver in $\mathcal{P}$ can also be added into $\mathcal{E}_x$. Therefore, the number of edges in $\mathcal{E}_x$ satisfies

$$| \mathcal{E}_x | \geq C^2_{|\mathcal{X}|} + | \mathcal{X} | \tag{2}$$

On the other hand, for each node in $\mathcal{P}$, it is adjacent to at most Δ nodes in $\mathcal{X}$. That means

$$| \mathcal{E}_x | \leq | \mathcal{X} | \cdot \Delta \tag{3}$$

Put (2) and (3) together, we have

$$C^2_{|\mathcal{X}|} + | \mathcal{X} | \leq | \mathcal{X} | \cdot \Delta \tag{4}$$

Thus $| \mathcal{X} | \leq 2\Delta - 1$, which finishes the proof.

Lemma 6 *The number of frames needed by GSSP to schedule all the senders in the input set $\mathcal{A}$ is upper bounded by $\Delta + \left\lceil \frac{\Delta - 1}{F_0} \right\rceil$.*

Proof Similarly, let $\mathcal{X}$ denote the set of mutually conflicting senders in $\mathcal{A}$. Suppose that there are at most k nodes in $\mathcal{X}$ sharing the same receiver and they constitute a subset $\mathcal{X}_1 \subseteq \mathcal{X}$). Apparently, the k ($k \leq \Delta$) nodes in $\mathcal{X}_1$ must be scheduled using k frames. Since GSSP tries to schedule as many nodes in each time-slot as possible, part of the nodes in $\mathcal{X} \backslash \mathcal{X}_1$ may be scheduled during the k frames together with the nodes in $\mathcal{X}_1$. However, GSSP randomly picks nodes for scheduling. If all the nodes in $\mathcal{X} \backslash \mathcal{X}_1$ are checked before the nodes in $\mathcal{X}_1$, then it will take at most another $\left\lceil \frac{2\Delta - k - 1}{F_0} \right\rceil$ frames to schedule them. It should be clarified that some nodes in $\mathcal{X} \backslash \mathcal{X}_1$ with a common receiver may not be scheduled in these $\left\lceil \frac{2\Delta - k - 1}{F_0} \right\rceil$ frames since their shared receiver can only be activated once in a frame under the duty-cycle scenario. At such circumstances, these nodes can all be scheduled together with the nodes in $\mathcal{X}_1$, and k frames are enough for them to be scheduled. Therefore, we can conclude that the number of frames needed by GSSP to finish the schedule is bounded by $k + \left\lceil \frac{2\Delta - k - 1}{F_0} \right\rceil \leq \Delta + \left\lceil \frac{\Delta - 1}{F_0} \right\rceil$. The inequality holds since $k \leq \Delta$ and $F_0 \geq 1$. This finishes the proof.

Theorem 3 *The delay of the schedule produced by GAS is upper bounded by $(16 + \left\lceil \frac{13}{F_0} \right\rceil)R + \Delta + \left\lceil \frac{\Delta - 1}{F_0} \right\rceil - 13 - \left\lceil \frac{12}{F_0} \right\rceil$ frames.*

Proof In GAS, it takes at most $\Delta + \left\lceil \frac{\Delta - 1}{F_0} \right\rceil$ frames to schedule all the dominatees in T, according to Lemma 6. Next, it takes d_B rounds to schedule all the dominators and connectors in T_B by GSSP. During the i-th (i varies from d_B to 1) round, the nodes in layer i of T_B are scheduled. Based on Lemma 4 and Lemma 3, it takes at most $4 + \left\lceil \frac{3}{F_0} \right\rceil$ (respectively, $11 + \left\lceil \frac{10}{F_0} \right\rceil$) frames to schedule all the nodes in layer i of T_B, when number i is even(respectively, odd and $i > 1$). For the special case where $i = 1$, it needs at most $12 + \left\lceil \frac{11}{F_0} \right\rceil$ frames, instead. Thus the total number of frames consumed by GAS, denoted by t_{GAS}, can be bounded as below.

$$\begin{aligned} t_{GAS} &\leq \Delta + \left\lceil \frac{\Delta - 1}{F_0} \right\rceil + (15 + \left\lceil \frac{3}{F_0} \right\rceil \\ &\quad + \left\lceil \frac{10}{F_0} \right\rceil) \cdot \frac{d_B - 2}{2} + 16 + \left\lceil \frac{3}{F_0} \right\rceil + \left\lceil \frac{11}{F_0} \right\rceil \\ &\leq (16 + \left\lceil \frac{13}{F_0} \right\rceil)R + \Delta + \left\lceil \frac{\Delta - 1}{F_0} \right\rceil - 13 - \left\lceil \frac{12}{F_0} \right\rceil \end{aligned} \tag{5}$$

This finishes the proof.

5.3 Analysis of PAS

Theorem 4 *Algorithm PAS produces a valid schedule.*

Proof In PAS, we first schedule the dominatees. Specifically, if two dominatees are scheduled simultaneously, their dominator parents must have the same color. Based on the method of cell partition and coloring, each cell contains at most one dominator and every two dominators with the same color are separated by at least three cells. Note that the side length of each cell is $\frac{1}{\sqrt{2}}$, and the transmission range (interference range) of each node is 1. We can observe that when the dominators with a monotone color receive (or transmit) simultaneously, the transmissions are certainly collision-free under the protocol interference model. Therefore, the transmissions of the dominatees are successful.

We then schedule the dominators and connectors. Similarly, only dominators with a monotone color or

connectors whose dominator parents have a monotone colore are allowed to be scheduled simultaneously. Besides, the schedules of different layers in the aggregation tree are separated well apart to avoid overlapping. Therefore, the transmissions of the dominators and connectors are also successful. This finishes the proof.

Theorem 5 *The delay of the schedule produced by PAS is upper bounded by* $\left\lceil \frac{16}{F_0} \right\rceil (15R + \Delta - 14)$ *frames.*

Proof The proof is similar to that of Theorem 3. In PAS, it takes at most $\left\lceil \frac{16}{F_0} \right\rceil \cdot \Delta$ frames to schedule all the dominatees in T. Additionally, it takes at most $\left\lceil \frac{16}{F_0} \right\rceil \cdot 4$ (respectively, $\left\lceil \frac{16}{F_0} \right\rceil \cdot 11$) frames to schedule all nodes in the even (respectively, odd, but not 1) layer of the backbone tree T_B. For the special case of the first layer, it needs at most $\left\lceil \frac{16}{F_0} \right\rceil \cdot 12$ frames instead. Thus the total number of frames consumed by PAS, denoted by t_{PAS}, can be bounded as below.

$$\begin{aligned} t_{PAS} &\leq \left\lceil \frac{16}{F_0} \right\rceil \cdot \Delta + \left\lceil \frac{16}{F_0} \right\rceil \cdot 15 \cdot \frac{d_B - 2}{2} + \left\lceil \frac{16}{F_0} \right\rceil \cdot 16 \\ &\leq \left\lceil \frac{16}{F_0} \right\rceil \cdot (15R + \Delta - 14) \end{aligned} \tag{6}$$

This finishes the proof.

5.4 Remarks

From Theorem 3 and Theorem 5, we can observe that the theoretical upper bounds of GAS and PAS both largely depend on the network radius R. Specifically, the coefficients of R in the delay bounds of GAS and PAS are $16 + \left\lceil \frac{13}{F_0} \right\rceil$ and $15 \cdot \left\lceil \frac{16}{F_0} \right\rceil$, respectively. It is straightforward to see that the two coefficients are mainly determined by the node degree (see Lemma 4) of the data aggregation tree produced by Algorithm 1. To be more specific, the smaller the node degree is, the lower the delay bounds of GAS and PAS become. Consequently, the reduced CDS of Algorithm 1, resulting in a smaller node degree, indeed makes the delay of our scheduling algorithms more efficient.

We can also see that the delay of GAS and PAS are both bounded by $O(R + \Delta)$ frames. In Section 3.2, we have shown that R frames is a trivial lower bound to the MDAT-DC problem. Therefore, the approximation factors of GAS and PAS are both nearly constant. In other words, our algorithms have a provably good performance compared with the optimal solution.

6 Simulation

In this section, we evaluate the practical performance of GAS and PAS using MATLAB. In our simulation, the size of the network region is set as 200 m $\times$ 200 m, and a sink node is situated at the centre of the region. The transmission range and interference range of each node are both set as r. In the duty-cycle model, a frame contains F_0 time-slots and the length of each time-slot is normalized to 1 time unit. Since the network topology may largely impacts the performance of our algorithms involving geometric property, we generate 50 different topologies for each simulation. Specifically, for each topology, we generate uniformly distributed pseudo-random numbers as geometrical coordinates of the nodes. The simulation results presented below are average ones.

Firstly, we compare the aggregation time of GAS and PAS, as well as their theoretical upper bounds. We set F_0 as 5, which means that the duty-cycle ratio is 20 %. Later we will consider different values of F_0. By fixing r as $30m$, we plot the curves of aggregation time (measured by the number of slots used) with varied n in Fig. 2(a). We can observe that as n increases, the aggregation time of GAS and PAS both increase as well. This is not surprising since denser distribution of nodes leads to more collisions and interference, and therefore more slots are needed to schedule the conflicting nodes. In addition, we can see that GAS always consumes fewer time than PAS. This is due to the fact that GAS is a centralized algorithm which can make greedy schedules based on global collision information, while the distributed algorithm PAS cannot. Figure 2(a) also shows that both GAS and PAS work much better in practice, with much shorter aggregation time than the theoretical upper bounds. By fixing n as 400, we plot the curves of aggregation tine with varied r in Fig. 2(b). We can observe that the aggregation time of both GAS and PAS also increase, though much slower, with increasing r. As r increases, the network radius R is likely to be smaller while the maximum node degree Δ will be much larger since more nodes will crowd around each node. In this case, simultaneous transmissions are restrained and more slots are needed to finish the schedule. The upper bounds of GAS and PAS both decrease with increasing r as shown in Fig. 2(b), which are expected since their values mainly depend on R instead of Δ. We can also observe that when r becomes extremely large, i.e. bigger than 65m, the four curves plotted in Fig. 2(b) tend to be much closer. The reason behind is that when r increases, the interference range also increases monotonically. As r exceeds 65m, the interference range becomes so large that when a sender transmits, it will prevent other simultaneous transmissions from activating. In this case, GAS and PAS will perform

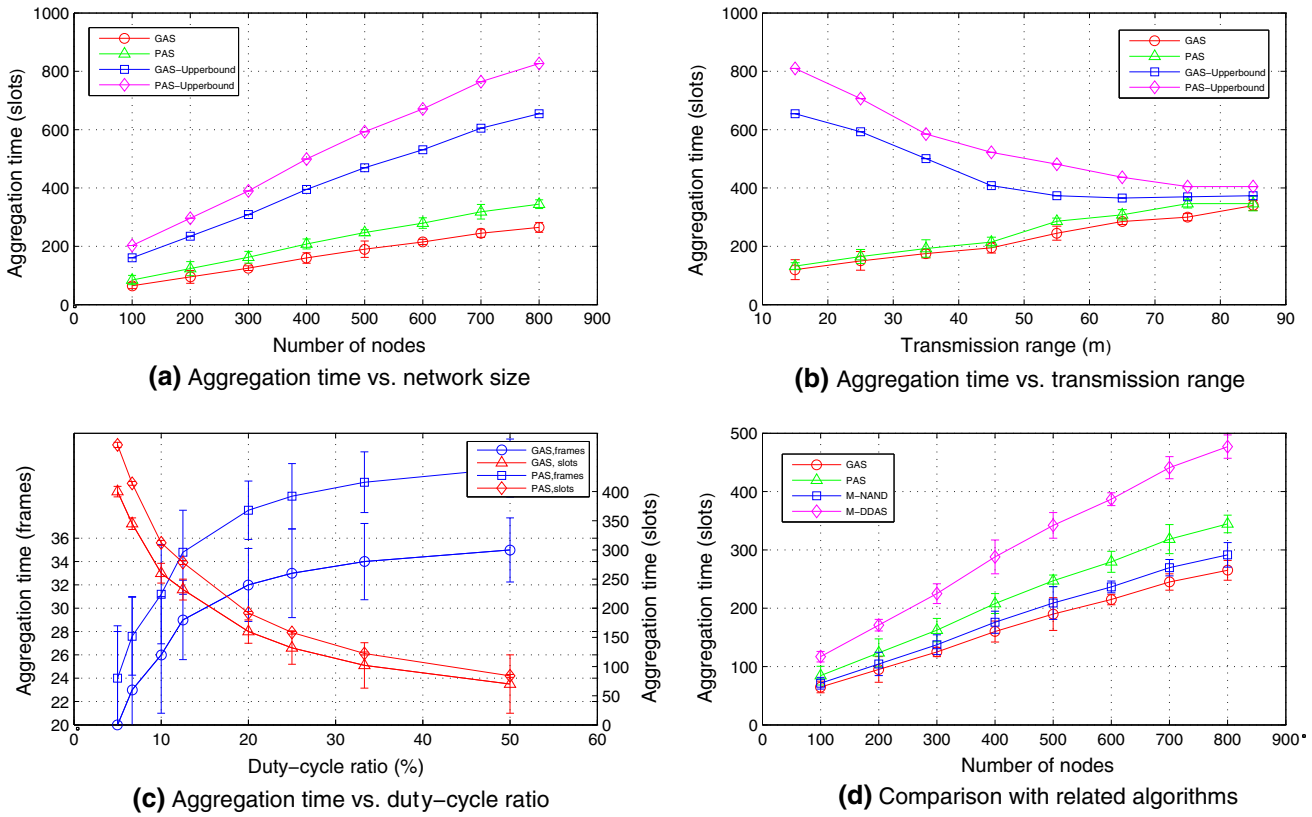


Fig. 2 Simulation results

rather poorly and the corresponding delay will approach to the theoretical upper bounds. That is, GAS and PAS exhibit their worst performance when the interference range becomes extremely large in a finite deployment area (200 m × 200 m in this case). In addition, we can also observe a reasonable phenomenon from Fig. 2(b) that across all values of r, the delay curves of GAS and PAS always remain below their respective upper bounds.

We then evaluate the performance of our algorithms with different duty-cycle ratios. Specifically, we fix n as 400, fix r as $30m$, and choose F_0 from the set of integers $\{2, 3, 4, 5, 8, 10, 15, 20\}$, which corresponds to duty-cycle ratios $\{50, 33.3, 25\,\%, 20, 12.5, 10, 6.67, 5\,\%\}$, respectively. We plot the curves of the aggregation time (measured by frames and time-slots) of GAS and PAS with varied duty-cycle ratios in Fig. 2(c). We can see that as the duty-cycle ratio increases (F_0 decreases), the number of frames used by GAS and PAS both increase as well. This can be easily explained by the fact that more nodes can be scheduled during a frame when the frame length is long while fewer nodes can be scheduled during a frame when the frame length is short. Nevertheless, the aggregation time, measured by the total time-slots used, decrease with increasing duty-cycle ratio as shown in Fig. 2(c). The implies that the length of a frame does have a remarkable impact on the practical performance of the proposed algorithms.

For completeness, we also compare GAS and PAS with two state-of-the-art algorithms designed for MDAT in WSNs where duty-cycle is not considered. The compared algorithms are the best centralized DAND proposed by Nguyen et al. [21] and the best distributed DDAS proposed by Xu et al. [26]. We slightly modify both algorithms and make them adapt to DC-WSNs as follows. If sender u transmits data to receiver v in time-slot t based on the schedule result of DAND (respectively, DDAS), then u will transmit in the active slot of v during frame t in the modified algorithm M-DAND (respectively, M-DDAS). We fix F_0 as 5 and r as 30 m. Figure 2(d) shows the aggregation time of different algorithms with varied n. We can observe that as n increases, all algorithms need more frames to finish the schedule. In addition, GAS always have shorter schedules than M-DADN and M-DDAS. This may due to the fact that conflicting nodes in non-DC-WSNs are able to transmit in the same frame in DC-WSNs provided that these nodes can be scheduled in different time-slots of a frame. Since M-DADN is also a centralized algorithm which makes efficient schedules based on global information, it achieves better performance compared with our distributed algorithm PAS. Nevertheless, the gap between

both algorithms is quite small and specifically, does not exceed 15% as shown in Fig. 2(d).

7 Conclusion and future work

In this paper, we study the minimum data aggregation time problem in practical duty-cycled wireless sensor networks (DC-WSNs) under the protocol interference model. The problem is shown to be NP-hard. We construct a data aggregation tree based on a reduced connected dominator set (CDS) and propose two aggregation scheduling algorithms named GAS and PAS to provide valid schedules with provably efficient delay. Theoretical analysis indicates that both algorithms can generate collision-free schedules subject to collision and duty-cycle constraints, and achieve nearly constant factors approximation on the optimal delay. Simulations further demonstrate the efficiency of the proposed algorithms and show that they work well under various network conditions.

Several interesting questions are left for future research. The first one is to derive efficient algorithms for MDAT under asynchronous DC-WSNs, which is a more practical case since WSNs are likely to be asynchronous as time synchronization is hard to implement. The second one is to consider the *signal-to-interference-plus-noise-ratio* (SINR) based interference model which has been shown to model wireless interference more accurately than the widely used protocol interference model [8]. The third one is to apply the design idea raised in this paper to other important operations in wireless networks such as broadcast, gossiping, and data collection.

References

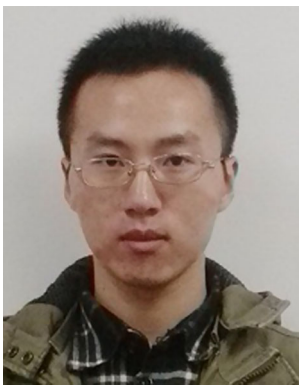
1. An, M. K., Lam, N. X., Huynh, D. T., & Nguyen, T. N. (2012). Minimum latency data aggregation in the physical interference model. *Computer Communications, 35*(18), 2175–2186.
2. Bagaa, M., Derhab, A., Lasla, N., Ouadjaout, A., & Badache, N. (2012). Semi-structured and unstructured data aggregation scheduling in wireless sensor networks (pp. 2671–2675). In *INFOCOM, 2012 Proceedings IEEE*. IEEE.
3. Cao, Y., Guo, S., & He, T. (2012). Robust multi-pipeline scheduling in low-duty-cycle wireless sensor networks (pp. 361–369). In *INFOCOM, 2012 Proceedings IEEE*. IEEE.
4. Cao, Z., He, Y., & Liu, Y. (2012). L2: Lazy forwarding in low duty cycle wireless sensor networks (pp. 1323–1331). In *INFOCOM, 2012 Proceedings IEEE*. IEEE.
5. Chen, S., Wang, Y., Li, X. Y., & Shi, X. (2009). Order-optimal data collection in wireless sensor networks: Delay and capacity. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2009 (SECON'09). 6th Annual IEEE Communications Society Conference on*, pp. 1–9. IEEE.
6. Chen, X., Hu, X., & Zhu, J. (2005). Minimum data aggregation time problem in wireless sensor networks. In: *Mobile Ad-hoc and Sensor Networks* (pp. 133–142). Berlin: Springer.
7. Fasolo, E., Rossi, M., Widmer, J., & Zorzi, M. (2007). In-network aggregation techniques for wireless sensor networks: a survey. *Wireless Communications IEEE, 14*(2), 70–87.
8. Goussevskaia, O., Wattenhofer, R., Halldórsson, M. M., & Welzl, E. (2009). Capacity of arbitrary wireless networks (pp. 1872–1880). In *INFOCOM 2009, IEEE*. IEEE.
9. Guo, S., Gu, Y., Jiang, B., & He, T. (2009). Opportunistic flooding in low-duty-cycle wireless sensor networks with unreliable links (pp. 133–144). In *Proceedings of the 15th annual international conference on Mobile computing and networking*. New York: ACM.
10. Gupta, P., & Kumar, P. R. (2000). The capacity of wireless networks. *Information Theory on IEEE Transactions, 46*(2), 388–404.
11. Han, K., Luo, J., Liu, Y., & Vasilakos, A. (2013). Algorithm design for data communications in duty-cycled wireless sensor networks: A survey. *Communications Magazine IEEE, 51*(7), 103–116.
12. Huang, S. H., Wan, P. J., Jia, X., Du, H., & Shang, W. (2007). Minimum-latency broadcast scheduling in wireless ad hoc networks (pp. 733–739). In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*. IEEE.
13. Huang, S. H., Wan, P. J., Vu, C. T., Li, Y., & Yao, F. (2007). Nearly constant approximation for data aggregation scheduling in wireless sensor networks (pp. 366–372). In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*. IEEE.
14. Ji, S., & Cai, Z. (2013). Distributed data collection in large-scale asynchronous wireless sensor networks under the generalized physical interference model. *ACM Transactions on Networking IEEE, 21*(4), 1270–1283.
15. Jiao, X., Lou, W., Wang, X., Ma, J., Cao, J., & Zhou, X. (2013). On interference-aware gossiping in uncoordinated duty-cycled multi-hop wireless networks. *Ad Hoc Networks, 11*(4), 1319–1330
16. Krishnamachari, B., Estrin, D., & Wicker, S. (2002). Modelling data-centric routing in wireless sensor networks. *IEEE Infocom, 2*, 39–44.
17. Li, D., Zhu, Q., Du, H., Wu, W., Chen, H., & Chen, W. (2011). Conflict-free many-to-one data aggregation scheduling in multi-channel multi-hop wireless sensor networks (pp. 1–5). In *2011 IEEE international conference on communications (ICC)*. IEEE.
18. Li, H., Wu, C., Hua, Q. S., & Lau, F. (2011). Latency-minimizing data aggregation in wireless sensor networks under physical interference model. *Ad Hoc Networks, 23*(2), 123–135.
19. Li, Y., Guo, L., & Prasad, S. K. (2010). An energy-efficient distributed algorithm for minimum-latency aggregation scheduling in wireless sensor networks. In 2010 *IEEE 30th international conference on distributed computing systems (ICDCS)*, pp 827–836. IEEE.
20. Malhotra, B., Nikolaidis, I., & Nascimento, M. A. (2011). Aggregation convergecast scheduling in wireless sensor networks. *Wireless Networks, 17*(2), 319–335.
21. Nguyen, T. D., Zalyubovskiy, V., Choo, H. (2011). Efficient time latency of data aggregation based on neighboring dominators in wsns (pp. 1–6). In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*. IEEE.
22. Wan, P. J., Alzoubi, K. M., Frieder, O. (2002). Distributed construction of connected dominating set in wireless ad hoc networks (pp. 1597–1604). In *INFOCOM 2002. Twenty-First annual joint conference of the IEEE computer and communications societies. Proceedings. IEEE, 3*. IEEE.

23. Wan, P. J., Huang, S. C. H., Wang, L., Wan, Z., Jia, X. (2009). Minimum-latency aggregation scheduling in multihop wireless networks (pp. 185–194). In *Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing*. New York: ACM.
24. Wang, F., Liu, J. (2009). Duty-cycle-aware broadcast in wireless sensor networks (pp. 468–476). In: *INFOCOM 2009, IEEE*. IEEE.
25. Xiao, S., Pan, L., Liu, J., Li, B., Yuan, X. (2013). Distributed broadcast with minimum latency in asynchronous wireless sensor networks under sinr-based interference. *International Journal of Distributed Sensor Networks*.
26. Xu, X., Li, X. Y., Mao, X., Tang, S., Wang, S. (2011). A delay-efficient algorithm for data aggregation in multihop wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems, 22*(1), 163–175.
27. Xu, X., Li, X. Y., Song, M. (2013). Efficient aggregation scheduling in multihop wireless sensor networks with sinr constraints. *IEEE Transactions on Mobile Computing, 12*(12), 2518–2528.
28. Yu, B., Li, J., Li, Y. (2009). Distributed data aggregation scheduling in wireless sensor networks (pp. 2159–2167). In: *INFOCOM 2009, IEEE*. IEEE.
29. Zhao, D., Chin, K. W. (2013). Approximation algorithm for data broadcasting in duty cycled multi-hop wireless networks. *EURASIP Journal on Wireless Communications and Networking, 2013*(1), 1–14.

Shiliang Xiao received the B.S. degree in Electronic and Communication Engineering from Zhejiang University, Hangzhou, China, in 2010 and is currently pursuing the Ph.D. degree in Communication and Information System at Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences (CAS), Shanghai, China. He is also with the University of Chinese Academy of Sciences, Beijing, China. His research interests include wireless ad hoc and sensor networks, algorithm design and analysis, and cloud computing.

Jingchang Huang received his B.S. degree in communication engineering from Yunnan University, Kunming, China, in 2010. During 2010 to 2011, he studied signal processing in University of Science and Technology of China, Hefei, China. He is currently working toward the Ph.D. degree in Science and Technology on Microsystem Laboratory, Shanghai Institute of Microsystem and Information Technology (SIMIT), Chinese Academy of Sciences (CAS), Shanghai, China. His research interests include array signal processing, patterns recognition and wireless sensor networks.

Lebing Pan received the B.S. degree in communication engineering from Tongji University, Shanghai, China, in 2010 and is currently pursuing the Ph.D. degree in communication and information system at Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences (CAS), Shanghai, China. He is also with the Key Laboratory of Wireless Sensor Networks and Communications, Chinese Academy of Sciences. His research interests include compressive sensing, baseband signal processing, and cognitive radio.



Yongbo Cheng received the B.S. degree in Computer Science and Technology from Wuhan University of Technology, in 2012. He is currently a master student at the Shanghai Institute of Microsystem and Information of Technology, CAS. His current research interest lies in wireless sensor networks.

Jianpo Liu received his B.S. degree in Computer Science and Technology from Guilin University of Electronic Science and Technology, Guilin, China, in 2002, and his M.S. degree from Xidian University, Xi'an, China, in 2010. He is now a research associate at Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences (CAS). His research interests include data fusion, target recognition and software systems.