Yu B, Li JZ. Minimum-time aggregation scheduling in duty-cycled wireless sensor networks. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 26(6): 962–970 Nov. 2011. DOI 10.1007/s11390-011-1193-9

Minimum-Time Aggregation Scheduling in Duty-Cycled Wireless Sensor Networks

Bo Yu (于 博) and Jian-Zhong Li (李建中), *Member*, *CCF*

School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

E-mail: {bo_yu, lijzh}@hit.edu.cn

Received June 25, 2010; revised September 13, 2011.

Abstract Aggregation is an important and commonplace operation in wireless sensor networks. Due to wireless interferences, aggregation in wireless sensor networks often suffers from packet collisions. In order to solve the collision problem, aggregation scheduling is extensively researched in recent years. In many sensor network applications such as real-time monitoring, aggregation time is the most concerned performance. This paper considers the minimum-time aggregation scheduling problem in duty-cycled wireless sensor networks for the first time. We show that this problem is NP-hard and present an approximation algorithm based on connected dominating set. The theoretical analysis shows that the proposed algorithm is a nearly-constant approximation. Simulation shows that the scheduling algorithm has a good performance.

Keywords aggregation, duty-cycle, scheduling, sensor network

1 Introduction

Wireless sensor networks have been widely used in various applications such as habitat monitoring[1-3] and target tracking[4-5] in recent years. Data aggregation is an essential operation in wireless sensor networks and has been well studied recently[6-8]. As we know, a collision occurs when a sensor node hears two packets at the same time and neither of the packets can be received correctly. Collisions occur very often in wireless sensor networks and cause packet retransmissions, which waste a large amount of energy. In order to avoid wireless interference in sensor networks, aggregation scheduling mechanism has been proposed and studied recently[9-11]. These work schedules the aggregation by assigning a sending and receiving time slot (or multiple slots) to each node beforehand, according to the network topology, to make sure that no collisions occur, with the goal of minimizing the aggregation time.

Recently, duty-cycled wireless sensor networks have been considered[12], where each sensor node in the network works for a time duration and sleeps for the rest of the time in each working period. This helps to save large amounts of energy since the working time of the sensor nodes decreases significantly. Many sensor network applications need to maintain a long-lived network such as environmental monitoring, thus duty-cycled wireless sensor networks have been widely deployed[13-14] and much work has been done in the background of duty-cycled sensor networks[15-17]. This paper considers the minimum-time aggregation scheduling problem in duty-cycled wireless sensor networks for the first time. For the ease of description, we call this problem the dc-MTAS problem for short. In this paper, we first show that dc-MTAS problem is NP-hard. Then, a scheduling algorithm SA is presented to solve dc-MTAS, based on connected dominating set. We analyze the aggregation time of the algorithm, which is the optimization goal, and show that the aggregation time is not larger than $(15R(G,s)+\Delta-3)T$ time units, where $R(G,s)$ is the radius of the network graph G with respect to the sink s, and T is the number of time units in a working period, which can be seen as a constant. This means that SA is a nearly-constant approximation. Finally, we conduct extensive simulations to evaluate the performance of SA.

The rest of the paper is organized as follows. Section 2 introduces the related work. Section 3 presents the network model and formulates the dc-MTAS problem. We propose our scheduling algorithm SA in Section 4. Simulations are conducted to evaluate SA in Section 5. Section 6 concludes the paper.

Regular Paper

This work is supported by the Key Project of National Natural Science Foundation of China under Grant No. 61033015, the National Natural Science Foundation of China/Research Grants Council of Hong Kong Joint Research Scheme under Grant No. 60831160525, and the National Natural Science Foundation of China under Grant No. 60933001.

2 Related Work

In this section, we introduce the most relevant work to ours, which can be divided into two branches. The first branch of related work includes the work on aggregation and aggregation scheduling in wireless sensor networks. The second branch contains the work on duty-cycled wireless sensor networks. We give these studies a brief review in the following.

Aggregation in wireless sensor networks has been well studied recent years[6-7,18-21]. Aggregation operation includes simple aggregates such as MAX, MIN, COUNT, SUM, and also complex aggregates such as AVERAGE, MEDIANS and so on. Most of the work executes aggregation based on an aggregation tree[6-7,18,20]. The sink is selected as the root of the aggregation tree and each node sends its data up to its parent according to the tree. [19, 21] adopt routing mechanisms based on a connected graph where tree and multi-path coexist. [21-22] utilize synopsis to avoid double counting when exploiting multi-paths in wireless sensor networks. In contrast to accurate aggregation processing, approximate querying technique is proposed and studied in [8, 19]. These work on aggregation does not consider the collision problem in wireless sensor networks. Due to the broadcast nature of wireless transmissions, aggregation processing often suffers from collisions, i.e., a node hears two or more messages at the same time and can receive none of the messages correctly. To solve this problem, aggregation scheduling problem is proposed and studied. Minimum-time aggregation scheduling means to assign each node with the predetermined sending and receiving slots to assure that collisions do not occur, meanwhile minimizing the aggregation time. The related work is introduced as follows.

The minimum-time aggregation scheduling problem (MTAS) in wireless sensor networks is first introduced in [23]. The authors proved that the MTAS problem is NP-hard and proposed a scheduling algorithm with the time bound of $(\Delta - 1)R$, where Δ is the maximum degree of the network graph and R is the network radius. This is a $(\Delta - 1)$-approximation. Afterwards, some work proposes new algorithms to improve the time bound to nearly-constant approximations, including centralized[9-10] and distributed[24-25] algorithms. However, there has been no work on aggregation scheduling in duty-cycled wireless sensor networks.

There are also some work on duty-cycled wireless sensor networks, such as [12, 15-16]. A mechanism for providing periodic energy-efficient radio sleep cycles is proposed in [12] to minimize the communication latency. In [26], a short preamble MAC protocol is proposed for duty-cycled wireless sensor networks, which reduces the energy usage of sensor nodes and per-hop latency. A dynamic switch-based data forwarding scheme is designed in [15] and the combined effect of sleep latency and unreliable communication links is investigated. In [16], a delay-driven opportunistic flooding method is designed for low-duty-cycle wireless sensor networks with unreliable links. Multihop broadcast protocols for duty-cycled wireless sensor networks are proposed in [27-28]. In addition, several duty-cycle-aware broadcast algorithms are proposed in [17, 29]. The existing work on duty-cycled wireless sensor networks does not consider the aggregation scheduling problem.

To summarize, this paper is the first work on minimum-time aggregation scheduling in duty-cycled wireless sensor networks, to the best of our knowledge.

3 Preliminaries

3.1 Network Model

The network consists of n sensor nodes and one base station (also called a sink). Each sensor node can send or receive data to or from all directions. We model the sensor network as a unit disk graph (UDG[30]) $G = (V, E)$, where V is the set of sensor nodes and E is the set of edges. An edge $(u, v) \in E$ if u is in the transmission area of v.

Each node in a duty-cycled sensor network has two possible states: active state and sleeping state. A node in active state can sense the physical world, transmit a packet or receive a packet. A node in sleeping state turns off all of its function modules except a timer to wake itself up. We assume that the active/sleeping time, which is also called the working schedule, of each node is determined once the network is deployed. We divide the whole lifetime into multiple working periods with the same length and one working period W is further divided into T time units. Each time unit is long enough for a packet transmission. For simplicity, we assume that each node u is active in one of the T time units, say $A(u)$, in each working period. We call $A(u)$ the *active time unit* of node u, where $A(u) \in \{0, 1, \ldots, T-1\}$ for any node u. The duty-cycle is defined as the ratio between the active time and the whole working period. For example, in a network with a 12.5% duty-cycle, the working period is divided into 8 time units ($T = 8$) and each node is in active state in only one of the 8 time units. Note that our algorithm can be easily modified to adjust the case where nodes are active in multiple time units of a working period. In a duty-cycled sensor network, a node can transmit a packet at any time, but can only receive a packet when it is active.

3.2 Problem Definition

First, we introduce the definition of *collision*. A collision occurs at a node u if u hears two or more packets at the same time. This means that a node can receive a packet correctly if it hears only this packet at that moment. We assume that each node has one packet of sensing data. The aggregation completes when all the data packets in the network are sent to the sink. Then, the dc-MTAS problem can be formulated as follows. Given a sensor network graph $G(V, E)$ and the predetermined working schedules of all the nodes in V, assign a working period M and a receiving node v to each node $u \in V$, such that u sends data to v in v's active time unit $A(v)$ of working period M causing no collisions, minimizing the total aggregation time. Here M is a natural number, e.g., $M = 5, A(v) = 3$ means u sends data to v in the 3rd time unit of the 5th working period. Formally, we denote the schedule of a node u by $sch(u) = \langle (M, A(v)), v\rangle$. We presents the complexity of dc-MTAS in the following lemma.

Lemma 1. *dc-MTAS is NP-hard.*

Proof. Consider a special case of dc-MTAS where $T = 1$, which means that all the nodes are active in the only time unit of each working period. This special case of dc-MTAS is the MTAS problem, which is proved to be NP-hard in [23]. Thus, dc-MTAS (with $T \geqslant 1$) is NP-hard, according to [31]. □

4 Algorithm

In this section, we present our scheduling algorithm (SA) in detail. First, the algorithm description is given in Subsection 4.1. Then, analysis on the upper bound for the aggregation time of SA is shown in Subsection 4.2.

4.1 Algorithm Description

The algorithm SA consists of two phases. The first phase is the layered structure construction for aggregation (LSC). The second phase is the working period scheduling (WPS). We introduce them in detail as follows.

In the first phase of SA, we construct a layered structure for aggregation using the maximal independent set (MIS). We name this as layered structure construction (LSC). The pseudocode of LSC is shown in Fig.1.

First, we carry out a breadth-first search on the network graph G from sink s and compute the hop distance from each node to s. We divide all the sensor nodes into layers by the computed hop distance (note that this is not the eventual layered structure). For example, a node u which is 3 hops to the sink is in layer 3 currently. The sink itself forms layer 0. We also record the information of each node u's predecessor $\pi(u)$, which is defined as the node in u's upper layer from which u is visited during the breadth-first search. Here a node u's (in layer l) upper layer refers to layer $l - 1$.

```
1:  Compute $dist(s,u)$ by a breadth-first search and record
    $u$'s predecessor $\pi(u)$, for each node $u \in V$ //$s$ is the sink
2:  $layer(u) \leftarrow dist(s,u)$
3:  $nlayer(u) \leftarrow layer(u)$
4:  divide all the nodes into layers $L_1, \ldots, L_l$ according to
    the $layer(u)$
5:  $IND \leftarrow \varnothing, \quad IND_0 \leftarrow \{s\}$
6:  for $i \leftarrow 1$ to $l$ do
7:      find an MIS $IND_i \in L_i$ indep. of $IND$
8:      $IND \leftarrow IND \cup IND_i$
9:  $CNT \leftarrow \varnothing$
10: for $i \leftarrow 2$ to $l$ do
11:     for each node $u \in IND_i$ do
12:         if $\exists v \in CNT$ & $(u,v) \in E$ then
13:             $nlayer(u) \leftarrow nlayer(v) + 1$
14:         else
15:             $CNT \leftarrow CNT \cup \{\pi(u)\}$
16:             find $\pi(u)$'s dominator $d_{\pi(u)}$ whose layer $\leqslant i-1$
17:             $nlayer(\pi(u)) \leftarrow nlayer(d_{\pi(u)}) + 1$
18:             $nlayer(u) \leftarrow nlayer(\pi(u)) + 1$
```

Fig.1. Outline of layered structure construction.

Then, an MIS is constructed layer by layer (lines 6∼8). IND_i denotes the set of the independent nodes selected in the i-th layer, and the union of all IND_i forms an MIS of the network. Next, we select some nodes to interconnect the nodes in IND, which we call them the connector nodes, and these nodes form the set CNT. Thus, we obtain a layered structure with layers of independent nodes and connector nodes alternating. With the occurrence of the connector nodes, the layer information changes as shown in line 13 and lines 17∼18. The *nlayer* of each node reflects the eventual layered structure. The nodes in IND and CNT form a connected dominating set of the network. We call the rest nodes dominatee nodes.

Here we present an illustration to further explain the layered structure by Fig.2. The left side shows the layer of the nodes by breadth-first search, layer 0 to layer 5, where only the sink node is in layer 0. These layer information correspond to the *layer* in algorithm LSC. The black nodes in Fig.2 are independent nodes and they form the set *IND*. The gray nodes are connector nodes which interconnect the independent nodes, e.g., node y is a connector node which interconnects node x with the sink. The numbers in the brackets beside the black and gray nodes are the *nlayer* information for the nodes. The *nlayer* and *layer* information can be different, e.g., $layer(u) = 2$ while $nlayer(u) = 3$. This is because node u interconnects node z in IND_2 with node v in IND_3, thus $nlayer(u)$ is set to $nlayer(z)+1$ $(2+1=3)$, according to line 17 of LSC in Fig.1. Also, $nlayer(v)$ is set to $nlayer(u)+1$ $(3+1=4)$, according to line 18 in Fig.1. The *nlayer* information of the nodes represents the layered structure of the network.

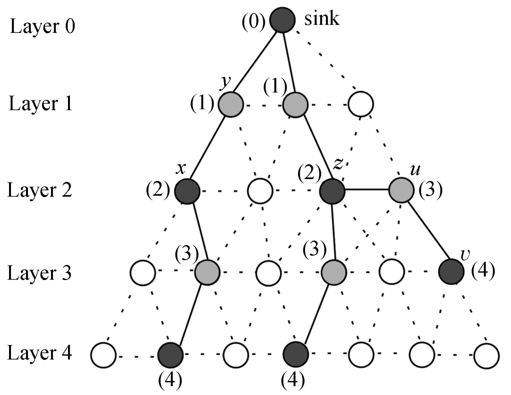


Fig.2. Illustration of the layered structure.

In the second phase of SA, we schedule the working period for each node in the network. The dominatee nodes are first scheduled and then the nodes in *IND* and *CNT* are scheduled layer by layer. We first present a subprocedure, minimal covering schedule (MC-schedule) with node sets X and Y as input. The function of MC-schedule(X,Y) is to complete scheduling of transmission from node set X to node set Y. We call X the sending set and Y the receiving set. The pseudocode of MC-schedule is shown in Fig.3.

```
$G' \leftarrow$ the induced bi-subgraph $G[X,Y]$
$X' \leftarrow X$, $Y' \leftarrow Y$, $M \leftarrow 0$
for $i \leftarrow 0$ to $T-1$ do
    $Y_i \leftarrow \{y \in Y' | A(y) = i\}$
for $X' \neq \varnothing$ do
    $M \leftarrow M+1$
    for $i \leftarrow 0$ to $T-1$ do
        $X_i \leftarrow \{x \in X' | \exists y \in Y_i, (x,y) \in E(G')\}$
        if $X_i = \varnothing$ then
            continue
        find a minimal cover $C_i$ of $X_i$, such that $C_i \subset Y_i$
        for each node $y \in C_i$ do
            $x \leftarrow$ a private neighbor of $y$ in $X_i$
            $sch(x) \leftarrow \langle (M,i), y \rangle$
            $X' \leftarrow X' \setminus \{x\}$
        $Y_i \leftarrow C_i$
```

Fig.3. Outline of minimal covering schedule.

MC-schedule divides the receiving set Y into T partitions: $Y_0, Y_1, \ldots, Y_{T-1}$ according to the active time units of nodes, i.e., $Y_i = \{u \in Y | A(u) = i\}$. As long as there are nodes unscheduled in X, MC-schedule iteratively performs the following for the set $Y_0, \ldots, Y_{T-1}$. In the M-th iteration, for each Y_i, MC-schedule first finds a minimal covering $C_i \subset Y_i$ of the set of nodes in X which are covered by Y_i. By the minimality of C_i, we can find a private neighbor x of each node y in Y_i, i.e., x is covered only by y rather than any other node in C_i. Then, the schedule of node x is set to $\langle (M,i), y \rangle$, which means x sends data to y in the i-th time unit of the M-th working period. C_i is assigned to Y_i after all the nodes in C_i finish scheduling in each subiteration. In this way, all the nodes in X find their sending working period since MC-schedule ends when there are no nodes unscheduled in X. The process of finding private neighbors ensures that no collision occurs in a time unit of any working period. This result naturally leads to the property that MC-schedule generates a collision-free schedule for the sending set X.

Fig.4 shows an example of the running process of MC-schedule. In this figure, $X = \{x_1, \ldots, x_8\}$ and $Y = \{y_1, \ldots, y_5\}$. Each working period is divided into 2 time units, i.e., $T = 2$. The active time unit of each node in Y is marked above the node label, e.g., $A(y_1) = 0$, $A(y_4) = 1$. By our notations, $Y_0 = \{y_1, y_2\}$, $Y_1 = \{y_3, y_4, y_5\}$.

The left-hand side subfigures show the subgraphs induced by the unscheduled nodes. The right-hand side subfigures show the scheduling process in each iteration. For example, in the 1st iteration, $\{y_1, y_2\}$ is a minimal cover of Y_0 and x_1 is a private neighbor of y_1. Thus, $sch(x_1) = \langle (1,0), y_1 \rangle$, that is, x_1 sends data to y_1 in time unit 0 of the 1st working period. For simplicity, this is shown by an arrow from x_1 to y_1 and working period 1 in a bracket beside node x_1, in Fig.4(b). Next, a minimal cover $\{y_3, y_5\}$ of Y_1 is selected, and x_2, x_8 are scheduled. The 1st iteration ends here. Once all the nodes in X are scheduled, the whole iteration ends. The final schedule of the network is shown in Fig.4(k). The aggregation from X to Y, by MC-schedule, needs 3 working periods, i.e., 6 time units.

In the following we propose the working period scheduling (WPS) algorithm, which is the second phase of SA. WPS schedules all the sensor nodes layer by layer, making use of subprocedure MC-schedule. The pseudocode of WPS is shown in Fig.5.

First, the dominatee nodes are scheduled using MC-schedule (lines 1~4). The sending set is assigned to the set of dominatee nodes and the receiving set is assigned to the set of the independent and connector nodes.

Next, the independent nodes and the connector nodes are scheduled from the deepest layer of the layered structure constructed in the first phase of SA, up to the nearest layer to the sink (lines 5~10). Note that we need to make sure that the nodes in layer i (here refers to *nlayer*) send data after the nodes in deeper layers than i finish sending data, in order to guarantee

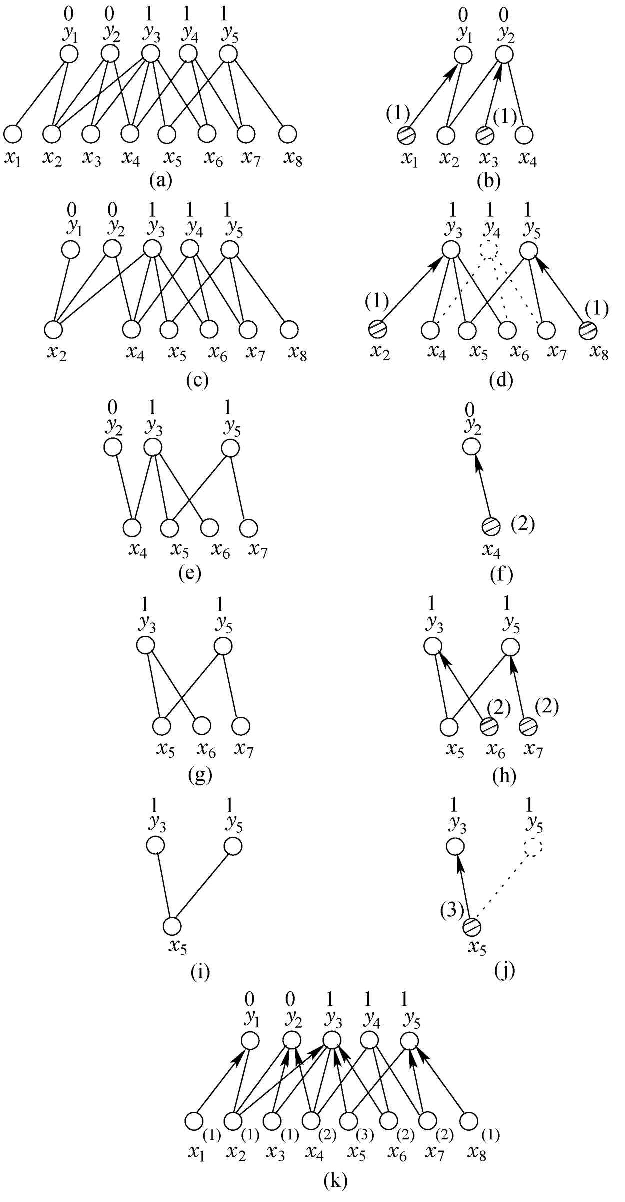


Fig.4. Illustration of MC-schedule.

```
$Y \leftarrow IND \cup CNT,\ X \leftarrow V \setminus Y$
minimal covering schedule $(X, Y)$
$l \leftarrow \max_{u \in V}\{nlayer(u)\}$
$w \leftarrow$ the maximum working period of $X$
for $i \leftarrow l$ down to 1 do
    $X \leftarrow \{u \in V \mid nlayer(u) = i\}$
    $Y \leftarrow \{u \in V \mid nlayer(u) = i - 1\}$
    minimal covering schedule $(X, Y)$
    delay the working period in the schedule of each node
    in $X$ by $w$
    $w \leftarrow$ the maximum working period of $X$
```

Fig.5. Outline of working period scheduling.

collisions do not occur across layers. Thus, we delay the working period of the scheduled nodes after calling the subprocedure MC-schedule by the largest working period of nodes in deeper layers. Recall that MC-schedule generates collision-free schedules, which means collisions do not occur in any layer. With the property of WPS that collisions do not occur across layers, we conclude that our scheduling algorithm SA generates a collision-free schedule of the whole network. This proves the correctness of SA. For the running time, it is easy to see that SA runs in polynomial time since LSC and WPS both run in polynomial time.

4.2 Analysis

In this subsection, we analyze the upper bound on the aggregation time of SA. We first present several lemmas as follows.

Lemma 2. *After layered structure construction, the following inequality holds.*

$$\max_{v \in V}\{nlayer(v)\} \leqslant 2(R(G, s) - 1),$$

where $R(G, s)$ denotes the radius of the network graph G with respect to the sink s.

Proof. According to LSC, an MIS of the network is first constructed and then connectors are selected to interconnect the independent nodes. Only when an independent node u finds a connector node v as its parent and v is in the same layer as v's parent w (another independent node), the *nlayer* information for u and v is modified, i,e., $nlayer(v)$ is set to $nlayer(w) + 1$ and $nlayer(u)$ is set to $nlayer(w) + 2$. The worst case is that each layer except layer 0 (the sink) is split into 2 layers. Since the original layer information satisfies $\max_{v \in V}\{layer(v)\} \leqslant R(G, s)$, the equality in the lemma follows. □

The following lemma gives an upper bound on the aggregation time of MC-schedule.

Lemma 3. *Let $sch(x) = \langle (M, i), y \rangle$ after MC-schedule(X, Y), where $x \in X$. Then, $M \leqslant |\{x' \in X | (x', y) \in E\}|$ for any node $x \in X$, where E is the edge set of the network graph G.*

Proof. Consider any node $x \in X$. If $sch(x) = \langle (M, i), y \rangle$ after MC-schedule(X, Y), then x is a private neighbor of y in the M-th iteration of MC-schedule. Thus, y must be in the minimal cover C_i for at least M iterations. This leads to the fact that y has at least M neighbors in X, which is the same meaning as the inequality in the lemma. □

The following lemma holds on the condition that the connector node set is minimal, which is proved in [10].

Lemma 4. *Let $IND_i = \{u \in IND | nlayer(u) = i\}$ and $CNT_i = \{u \in CNT | nlayer(u) = i\}$, then*

1) *Each connector node in CNT_i is adjacent to at most 4 independent nodes in IND_{i+1}.*

2) Each independent node in IND_i is adjacent to at most 11 connector nodes in CNT_i.

3) $CNT_0 \leqslant 12$.

Note that the condition in line 12 of the layer structure construction in Fig.1 guarantees the minimality of the connector node set. Thus, Lemma 4 holds for SA.

We analyze the aggregation time of SA in the following. The aggregation time consists of two parts: the aggregation time t_1 for dominatee nodes and the aggregation time t_2 for dominator nodes. Here the dominator nodes refer to the nodes in IND and CNT. The aggregation time of SA is the sum of t_1 and t_2 because SA makes the dominator nodes send data after all the dominatee nodes finish sending data.

We claim that t_1 does not exceed the largest working period wp of the dominatee nodes. According to Lemma 3, wp is not larger than the largest degree d in $G[X, Y]$ of the nodes in Y, where X is the set of the dominatee nodes and Y is the set of dominator nodes. It is obvious that $d \leqslant \Delta$, where Δ is the maximum vertex degree of the network graph G. Thus, $t_1 \leqslant \Delta$.

The dominator nodes are scheduled layer by layer, sequentially, according to WPS. The nodes in the j-th layer send data after all the nodes in the $(j+1)$-th layer finish sending data, for $j = 1, \ldots, R_m$, where $R_m = \max_{v \in V}\{nlayer(v)\}$. The aggregation time t_2 for the dominator nodes is the sum of the aggregation time of each layer, for the same reason as computing t_1. Referring to the LSC algorithm, the constructed layered structure guarantees that the maximum layer R_m of nodes is an even number, and the independent nodes are in the even-numbered layers and the connector nodes are in the odd-numbered layers. Consider the aggregation time t_{ind} from the set $\widetilde{X}$ of independent nodes in layer $2i$ to the set $\widetilde{Y}$ of connector nodes in layer $2i-1$. According to Lemma 3, t_{ind} must not be larger than $\max_{y \in \widetilde{Y}} |\{x' \in X | (x', y) \in E\}|$, which does not exceed 4 due to the first statement of Lemma 4. Thus, $t_{ind} \leqslant 4$, for $i = 1, \ldots, R_m/2$. Similarly, the aggregation time t_{cnt} from the set of connector nodes in layer $2i+1$ to the set of independent nodes in layer $2i$ satisfies $t_{cnt} \leqslant 11$ for $i = 1, \ldots, R_m/2-1$ and $t_{cnt} \leqslant 12$ for $i = 0$. This can be concluded from the 2nd and 3rd statements of Lemma 4. Then, the aggregation time for the dominator nodes is

$$\begin{aligned} t_2 &\leqslant (11+4) \times R_m/2 + 12 \\ &\leqslant 15(R(G,s)-1) + 12 = 15R(G,s) - 3. \end{aligned}$$

The second inequality sign follows from Lemma 2. Summing t_1 and t_2, we obtain that the aggregation time t of SA satisfies $t \leqslant 15R(G,s) + \Delta - 3$. This result means that the aggregation time of SA does not exceed $15R(G,s) + \Delta - 3$ working periods, which are $(15R(G,s) + \Delta - 3)T$ time units. This leads to the following theorem.

Theorem 1. *The aggregation time of SA does not exceed* $(15R(G,s) + \Delta - 3)T$ *time units.*

We claim that SA is a nearly-constant approximation algorithm. First, $R(G,s)$ is a lower bound on the aggregation time. The reason is that the farthest node to the sink s takes at least $R(G,s)$ time to send its sensing data to s. Secondly, T is the number of time units in one working period and it can be seen as a constant since T is small in practice. Typically, the duty-cycle in any wireless sensor network is always not lower than 1%, thus T can be bounded by 100, a constant number. For the above reasons, we conclude that SA is a nearly-constant approximation.

5 Performance Evaluation

We evaluate our scheduling algorithm SA by simulation in this section. Since there is no existing work on aggregation scheduling in duty-cycled sensor networks, we conduct a little modification on two existing scheduling algorithms for non-duty-cycled sensor networks to solve the dc-MTAS problem. We also propose a third algorithm BS, which is a naive baseline algorithm to solve dc-MTAS. We take the three algorithms as comparisons for SA and briefly introduce them in the following.

The first algorithm is the current best centralized algorithm E-PAS, proposed in [10]. The second algorithm is the current best distributed algorithm Clu-DDAS, proposed in [25]. Both algorithms generate a schedule of the node transmissions for aggregation in a non-duty-cycled wireless sensor network. We can formulate the output schedule of a node u by E-PAS or Clu-DDAS as (v, T), which means node u send its data to v in time slot T. To make the output schedule usable in a duty-cycled sensor network, given the output schedule (v, T) of a node u, we make u send its data to v in time slot $A(v)$ of the T-th working period (remind that $A(v)$ is the active time unit of node v). This simple modification enables E-PAS and Clu-DDAS to solve the dc-MTAS problem. We keep the original names of both algorithms after modification. The third algorithm BS constructs an aggregation tree and makes the nodes send data to the sink according to the tree structure and the node working schedule in a bottom-up way. BS does not make schedules beforehand and it makes nodes retransmit data when encountering collisions.

The network setting is as follows. The sensor nodes are randomly deployed in a region of size 200 m × 200 m and the sink is located in a corner. The transmission radius is 30 m. Each node randomly selects one active time unit as its working schedule beforehand. The aggregation time is measured in the number of working periods. The simulation for each setting is conducted for 20 times and the average value is computed as the result. The duty-cycles used in the simulation are 50%, 33.33%, 25%, 20%, 12.5%, 10%, 6.67%, 5%, corresponding to $T = 2, 3, 4, 5, 8, 10, 15, 20$, respectively.

Fig.6 shows the aggregation time of the four algorithms, in terms of working periods, with the number of nodes varying from 500 to 1 400. In Fig.6, the duty-cycles is set to 12.5%. The aggregation time increases with the number of nodes increasing for all of the algorithms, while the aggregation time for SA is 1 to 4 times shorter than that for the baseline BS and 1 to 2 times shorter than that for E-PAS and Clu-DDAS. For example, in the case of 1 200 nodes, the aggregation time for BS, E-PAS,Clu-DDAS and SA is 271, 93, 96, 43 working periods, respectively. The aggregation time of E-PAS and Clu-DDAS are more than 2 times longer than our algorithm SA, and the aggregation time for BS is almost 7 times of that of SA. The reason is that E-PAS and Clu-DDAS do not consider the cases where conflicting nodes in non-duty-cycled networks can transmit in the same working period in duty-cycled networks, as long as they transmit in different time units of one working period. As to BS, there can be many collisions and retransmissions making BS cost much more time.

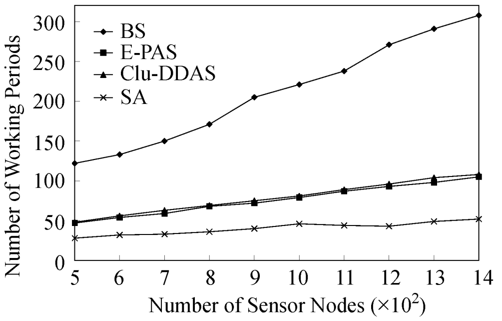


Fig.6. The number of working periods with different number of nodes for the four algorithms. The duty-cycle is set to 12.5%.

Fig.7 shows the aggregation time with the duty-cycle varying from 50% to 5%, where the number of nodes is set to 1 000. The number of working periods for BS and SA decreases when the duty-cycle gets smaller since there are more time units in each working period, enabling more nodes to send data in a working period. E-PAS and Clu-DDAS do not consider the duty-cycle issue thus the aggregation time for these two algorithms does not change with the duty-cycle decreasing. Note that the actual aggregation time increases when the duty-cycle gets smaller, though the number of working periods needed decreases. For example, in the case of 25% duty-cycle, a working period is divided into $T = 4$ time units and the number of working periods needed for SA is 59, resulting in 236 time units, while in the case of 12.5% duty-cycle, $T = 8$ and the number of working periods for SA is 46, resulting in 368 time units. This shows that low duty-cycle trades off aggregation time for lower energy cost.

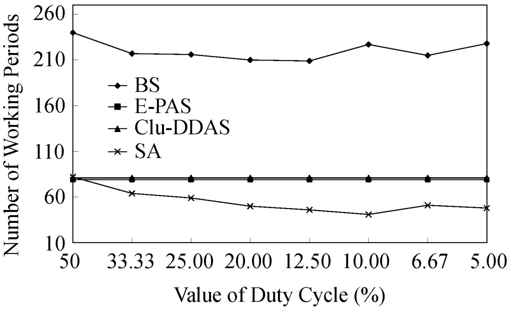


Fig.7. The number of working periods with different duty-cycles for the four algorithms. The number of nodes is set to 1 000.

In the following, we further study the factors that affect the aggregation time of SA. Fig.8 shows the working periods needed for SA with different number of sensor nodes in the cases of three duty-cycles, 25%, 12.5% and 6.67%. From Fig.8, we can conclude that the number of working periods increases when the number of sensor nodes increases, while a fixed-size network of lower duty-cycle needs less number of working periods for aggregation. However, the actual aggregation time slots needed may increase with duty-cycle decreasing, as is shown in Fig.9.

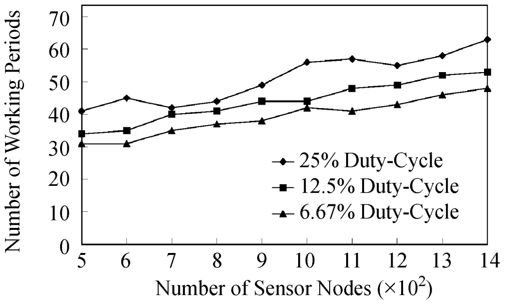


Fig.8. The number of working periods with different number of nodes for SA.

Fig.10 shows the number of working periods for SA with different number of sensor nodes in the cases of three transmission radii, 30 m, 40 m and 50 m, where the duty-cycle is set to 10%. When the transmission radius of sensor nodes increases, there are more nodes in u's transmission area, for each node u, which may lead to two trends. The first trend is that there are more interferences in packet transmissions, which means a

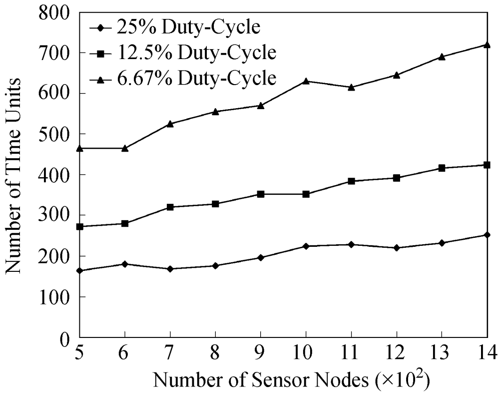


Fig.9. The number of total time slots with different number of nodes for SA.

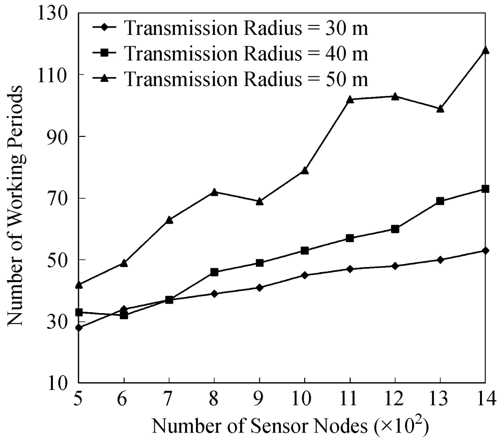


Fig.10. The number of working periods with different number of nodes for SA. The duty-cycle is set to 10%.

receiving node u has a higher probability of encountering a collision. In this case, with more potential colliding nodes, our scheduling algorithm SA may return a schedule with longer aggregation time to avoid collision. On the other hand, the second trend is that there are more options to choose in selecting a receiving node, for each sending node u, since there are more nodes in u's transmission area. This helps reduce the aggregation time in duty-cycled sensor networks because some nodes may have chances to advance their transmissions with more potential receiving nodes to be chosen. Fig.10 shows that the former trend dominates the latter for most of the cases, i.e., a larger transmission radius results in a longer aggregation time.

6 Conclusions

Aggregation scheduling has been studied recently to provide collision-free schedules for wireless sensor networks, in order to minimize the aggregation time. This paper considers minimum-time aggregation scheduling in duty-cycled wireless sensor networks for the first time. We show that this problem is NP-hard and present an approximation algorithm SA to solve this problem. Theoretical analysis shows that SA is a nearly-constant approximation. We carry out extensive simulations to study the performances of SA and the existing methods, including the state-of-the-art centralized and distributed scheduling algorithms, E-PAS and Clu-DDAS, and a naive baseline algorithm BS. The simulation result verifies our analysis and shows that SA has a better performance than existing methods. This paper considers aggregation scheduling under a simplified interference model where the interference radius equals the transmission radius. The aggregation scheduling problem in duty-cycled wireless sensor networks under other interference models, such as generalized protocol model, physical model, remains open and needs to be reinvestigated.

References

[1] Mainwaring A, Culler D, Polastre J, Szewczyk R, Anderson J. Wireless sensor networks for habitat monitoring. In *Proc. the 1st ACM International Workshop on Wireless Sensor Networks and Applications* (*WSNA 2002*), Atlanta, USA, Sept. 28, 2002, pp.88-97.

[2] Szewczyk R, Osterweil E, Polastre J, Hamilton M, Mainwaring A, Estrin D. Habitat monitoring with sensor networks. *Communications of the ACM – Wireless Sensor Networks*, 2004, 47(6): 34-40.

[3] Szewczyk R, Mainwaring A, Polastre J, Anderson J, Culler D. An analysis of a large scale habitat monitoring application. In *Proc. the 2nd ACM Conference on Embedded Networked Sensor Systems* (*Sensys 2004*), Baltimore, USA, Nov. 3-5, 2004, pp.214-226.

[4] Shrivastava N, Mudumbai R, Madhow U, Suri S. Target tracking with binary proximity sensors. *ACM Transactions on Sensor Networks*, 2009, 5(4): 251-264.

[5] Jeong J, Hwang T, He T, Du D. MCTA: Target tracking algorithm based on minimal contour in wireless sensor networks. In *Proc. the 26th Annual IEEE Conference on Computer Communications* (*INFOCOM 2007*), Anchorage, USA, May 6-12, 2007, pp.2371-2375.

[6] Madden S, Franklin M J, Hellerstein J M, Hong W. TAG: A tiny aggregation service for ad-hoc sensor networks. In *Proc. the 5th Symp. Operating Systems Design and Implementation* (*OSDI 2002*), Boston, USA, Dec. 9-11, 2002, pp.131-146.

[7] Shrivastava N, Buragohain C, Agrawal D, Suri S. Medians and beyond: New aggregation techniques for sensor networks. In *Proc. the 2nd ACM Conference on Embedded Networked Sensor Systems* (*Sensys 2004*), Baltimore, USA, Nov. 3-5, 2004, pp.239-249.

[8] Chu D, Deshpande A, Hellerstein J, Hong W. Approximate data collection in sensor networks using probabilistic models. In *Proc. the 22nd International Conference on Data Engineering* (*ICDE 2006*), Atlanta, USA, Apr. 3-8, 2006, p.48.

[9] Huang S C H, Wan P J, Vu C T, Li Y, Yao F. Nearly constant approximation for data aggregation scheduling in wireless sensor networks. In *Proc. the 26th Annual IEEE Conference on Computer Communications* (*INFOCOM 2007*), Anchorage, USA, May 6-12, 2007, pp.366-372.

[10] Wan P J, Huang S C H, Wang L, Wan Z, Jia X. Minimum-latency aggregation scheduling in multihop wireless networks. In *Proc. the 10th ACM International Symposium on Mobile*

Ad Hoc Networking and Computing (*MobiHoc 2009*), New Orleans, USA, May 18-21, 2009, pp.185-193.

[11] Gandham S, Zhang Y, Huang Q. Distributed minimal time convergecast scheduling in wireless sensor networks. In *Proc. the 26th International Conference on Distributed Computing Systems* (*ICDCS 2006*), Lisboa, Portugal, Jul. 4-7, 2006, p.50.

[12] Lu G, Sadagopan N, Krishnamachari B, Goel A. Delay efficient sleep scheduling in wireless sensor networks. In *Proc. the 24th Annual Joint Conference of the IEEE Computer and Communications Societies* (*INFOCOM 2005*), Miami, USA, Mar. 13-17, 2005, pp.2470-2481.

[13] Beutel J, Gruber S, Hasler A *et al.* PermaDAQ: A scientific instrument for precision sensing and data recovery in environmental extremes. In *Proc. the 8th ACM/IEEE Int. Conf. Information Processing in Sensor Networks* (*IPSN 2009*), San Francisco, USA, Apr. 13-16, 2009, pp.265-276.

[14] Burri N, von Rickenbach P, Wattenhofer R. Dozer: Ultra-low power data gathering in sensor networks. In *Proc. the 6th ACM/IEEE International Conference on Information Processing in Sensor Networks* (*IPSN 2007*), Cambridge, USA, Apr. 25-27, 2007, pp.450-459.

[15] Gu Y, He T. Data forwarding in extremely low duty-cycle sensor networks with unreliable communication links. In *Proc. the 5th ACM Conference on Embedded Networked Sensor Systems* (*Sensys 2007*), Sydney, Australia, Nov. 6-9, 2007, pp.321-334.

[16] Guo S, Gu Y, Jiang B, He T. Opportunistic flooding in low-duty-cycle wireless sensor networks with unreliable links. In *Proc. the 15th Annual International Conference on Mobile Computing and Networking* (*MOBICOM 2009*), Beijing, China, Sept. 20-25, 2009, pp.133-144.

[17] Wang F, Liu J. Duty-cycle-aware broadcast in wireless sensor networks. In *Proc. the 28th Annual IEEE Conference on Computer Communications* (*INFOCOM 2009*), Rio de Janeiro, Brazil, Apr. 19-25, 2009, pp.468-476.

[18] Deshpande A, Guestrin C, Madden S, Hellerstein J, Hong W. Model-driven data acquisition in sensor networks. In *Proc. the 30th International Conference on Very Large Data Bases* (*VLDB 2004*), Toronto, Canada, Aug. 31-Sept. 3, 2004, pp.588-599.

[19] Considine J, Li F, Kollios G, Byers J. Approximate aggregation techniques for sensor databases. In *Proc. the 20th International Conference on Data Engineering* (*ICDE 2004*), Boston, USA, Mar. 30-Apr. 2, 2004, pp.449-460.

[20] Sharaf M A, Beaver J, Labrinidis A, Chrysanthis P K. TiNA: A scheme for temporal coherency-aware in-network aggregation. In *Proc. the 3rd ACM International Workshop on Data Engineering for Wireless and Mobile Access* (*MobiDE 2003*), San Diego, USA, Sept. 19, 2003, pp.69-76.

[21] Manjhi A, Nath S, Gibbons P B. Tributaries and deltas: Efficient and robust aggregation in sensor network streams. In *Proc. the 24th ACM SIGMOD International Conference on Management of Data* (*SIGMOD 2005*), Baltimore, USA, Jun. 14-16, 2005, pp.287-298.

[22] Nath S, Gibbons P B, Seshan S, Anderson Z R. Synopsis diffusion for robust aggregation in sensor networks. In *Proc. the 2nd ACM Conference on Embedded Networked Sensor Systems* (*Sensys 2004*), Baltimore, USA, Nov. 3-5, 2004, pp.250-262.

[23] Chen X, Hu X, Zhu J. Minimum data aggregation time problem in wireless sensor networks. In *Proc. the 1st International Conference on Mobile Ad hoc and Sensor Networks* (*MSN 2005*), Wuhan, China, Dec. 13-15, 2005, pp.133-142.

[24] Yu B, Li J, Li Y. Distributed data aggregation scheduling in wireless sensor networks. In *Proc. the 28th Annual IEEE Conference on Computer Communications* (*INFOCOM 2009*), Rio de Janeiro, Brazil, Apr. 19-25, 2009, pp.2159-2167.

[25] Li Y, Guo L, Prasad S K. An energy-efficient distributed algorithm for minimum-latency aggregation scheduling in wireless sensor networks. In *Proc. the 30th International Conference on Distributed Computing Systems* (*ICDCS 2010*), Genoa, Italy, Jun. 21-25, 2010, pp.827-836.

[26] Buettner M, Yee G V, Anderson E, Han R. X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proc. the 4th ACM Conference on Embedded Networked Sensor Systems* (*Sensys 2006*), Boulder, USA, Oct. 31-Nov. 3, 2006, pp.307-320.

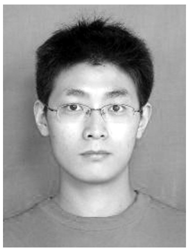
[27] Sun Y, Gurewitz O, Du S, Tang L, Johnson D B. ADB: An efficient multihop broadcast protocol based on asynchronous duty-cycling in wireless sensor networks. In *Proc. the 7th ACM Conference on Embedded Networked Sensor Systems* (*Sensys 2009*), Berkeley, USA, Nov. 4-6, 2009, pp.43-56.

[28] Lai S, Ravindran B. On multihop broadcast over adaptively duty-cycled wireless sensor networks. In *Proc. the 6th IEEE International Conference on Distributed Computing in Sensor Systems* (*DCOSS 2010*), Santa Barbara, USA, Jun. 21-23, 2010, pp.158-171.

[29] Jiao X, Lou W, Ma J, Cao J, Wang X, Zhou X. Duty-cycle-aware minimum latency broadcast scheduling in multi-hop wireless networks. In *Proc. the 30th International Conference on Distributed Computing Systems* (*ICDCS 2010*), Genoa, Italy, Jun. 21-25, 2010, pp.754-763.

[30] Clark B N, Colbourn C J, Johnson D S. Unit disk graphs. *Discrete Mathematics*, 1990, 86(1-3): 165-177.

[31] Garey M R, Johnson D S. Computers and Intractability: A Guide to the Theory of NP-Completeness. New York: W H Freeman & Co., 1979.



Bo Yu received his B.S. and M.S. degrees in computer science and technology from Harbin Institute of Technology, China, in 2006 and 2008, respectively. He is currently a Ph.D. candidate in the School of Computer Science and Technology, Harbin Institute of Technology. His research interests include wireless sensor networks, distributed algorithms.

Jian-Zhong Li is a professor in the School of Computer Science and Technology at the Harbin Institute of Technology, China. In the past, he worked as a visiting scholar at the University of California at Berkeley, USA, as a staff scientist in the Information Research Group at the Lawrence Berkeley National Laboratory, USA, and as a visiting professor at the University of Minnesota, USA. His research interests include data management systems, sensor networks and data intensive computing. He has published more than 150 papers in refereed journals and conferences, and has been involved in the program committees of major computer science and technology conferences, including SIGMOD, VLDB, ICDE, INFOCOM, ICDCS, and WWW. He is a member of CCF.