

# Fast Data Collection in Linear Duty-Cycled Wireless Sensor Networks

Zhong Shen, *Member, IEEE*, Hai Jiang, *Member, IEEE*, and Zhongjiang Yan, *Member, IEEE*,

**Abstract**—We study fast data collection in linear duty-cycled wireless sensor networks. We first present a benchmark algorithm that can achieve optimality in a general case (i.e., in non-duty-cycled case). Then based on the insights obtained in the general case, we propose an optimal algorithm for the case when each sensor only works at one slot in a cycle in the duty-cycled mode, and a distributed duty-cycled algorithm. It is proved that the two latter algorithms with cycle length equal to three have bounded performance gap to the benchmark algorithm in the non-duty-cycled case. Simulation results are used to demonstrate the effectiveness of the proposed algorithms.

**Index Terms**—Sensor networks, data collection, duty cycle.

## I. INTRODUCTION

One essential mission of a wireless sensor network (WSN) is to efficiently deliver data collected by sensors to a base station (BS). And in many WSN applications (e.g., the WSNs deployed along roadside to detect car accidents, animals, and other dangerous road conditions), it is desired that the data can be delivered as soon as possible, to minimize the completion time of data collection. The completion time may be affected by the following factors. First, a sensor is usually equipped with a half-duplex transceiver, which means it cannot send and receive simultaneously. Second, collisions may happen due to interference among sensors' transmissions.

Generally, minimum-time data collection in WSNs is an NP-complete problem [1]. Nevertheless, some optimal algorithms (noting that an optimal algorithm in this paper means the algorithm minimizes the completion time of data collection) have been provided in the literature for some special cases. Reference [2] gives an optimal data collection algorithm for a linear WSN. The data collection problem is transformed into a data distribution problem, and then the schedules in the data distribution problem are mapped into an optimal data collection solution. For a linear WSN in which each sensor has one packet to send, a distributed data collection algorithm

is proposed in [3], in which sensors rotate among three states (“transmission”, “idle”, and “receive”) while their initial states depend on their distance to the BS. Optimal data collection algorithms are given in [4] for a WSN with a tree topology, by assuming that interference is eliminated by power control or multichannel scheduling.

All above works assume that sensors are ready to receive or transmit packets at all time (i.e., they are in non-duty-cycled mode). However, usually WSNs are expected to work for a long time (for example, several months or even several years). If a sensor keeps running continuously, it may run out of battery in a few days. Therefore, a viable solution is to apply a duty-cycled mode: during each cycle of a number of slots, each sensor works in one or more working slots and sleeps in other slots. In particular, if each sensor is allowed to work at only one slot in a cycle, the working mode of the WSN is called *low power listening (LPL) mode* [5], [6]. Recently, duty-cycled WSNs have attracted many research attentions. The major research efforts focus on routing and load balancing [5]–[7]. To the best of our knowledge, there is no work in the literature to address fast data collection in duty-cycled WSNs considering the aforementioned two limiting factors (i.e., half-duplex transceiver and interference).

To fill the research gap, in this paper, we investigate minimum-time data collection in duty-cycled WSNs with linear topologies, such as WSNs deployed along gas pipelines (to detect gas leakage), highways (to detect accidents), and underground tunnels (to detect gas and/or water), etc. To gain insights for designing data collection algorithms in duty-cycled WSNs, we first present a benchmark algorithm that is optimal in a more general case, i.e., in non-duty-cycled case. Then based on insights obtained in the benchmark algorithm, we design an optimal algorithm for duty-cycled case, and a distributed duty-cycled algorithm whose performance gap to the benchmark algorithm is bounded if the cycle length is equal to three. Although reference [2] also gives a centralized optimal solution for a non-duty-cycled linear WSN, the insights in [2] may not be able to be used in duty-cycled case. This is because the scheme in [2] is based on the symmetry property of the data collection schedule (from sensors to the BS) and data distribution schedule (from the BS to sensors). The symmetry property, which holds in non-duty-cycled case, does not hold for duty-cycled case anymore due to the fact that data transmission in duty-cycled case is receiver-based (i.e., a node can receive packets only at its working slots, as shown in the subsequent Section III). Noting that optimal algorithm for linear non-duty-cycled WSNs is generally not unique, we solve the problem from a new perspective in this

Manuscript received May 31, 2013; revised September 16, 2013; accepted October 23, 2013. The associate editor coordinating the review of this paper and approving it for publication was Dr. J. Misić. This work was supported by the National Natural Science Foundation of China under Grants 61172069 and 61201157, the 111 Project under Grant B08038 with Xidian University, China, and the Natural Science and Engineering Research Council (NSERC) of Canada. Z. Shen is with the State Key Laboratory of Integrated Service Network, Xidian University, Xi'an, Shaanxi 710071, China (e-mail: z.shen.xidian@gmail.com). H. Jiang is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada T6G 2V4 (e-mail: hai1@ualberta.ca). Z. Yan is with School of Electronics and Information, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, China (e-mail: zhjyan@nwpu.edu.cn). Copyright (c) 2013 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubpermissions@ieee.org.

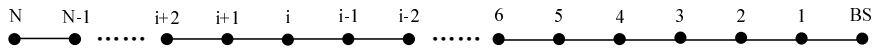


Fig. 1. The considered linear WSN.

paper, and provide insights that we may achieve optimality by giving higher priority to nodes closer to the BS and making simultaneous transmissions apart by at least three hops. As can be shown in Section III, these insights can be used in duty-cycled case. Moreover, by using our proposed benchmark algorithm as a reference point, we show in Section III-B that our optimal duty-cycled algorithm and distributed duty-cycled algorithm have bounded performance gap to the optimal performance in a general case (i.e., in non-duty-cycled case), while the performance bounds cannot be derived if we use the algorithm in [2] as a reference point for comparison.

The remainder of this paper is organized as follows. Section II proposes the benchmark algorithm in non-duty-cycled case and proves its optimality. Section III presents our optimal duty-cycled algorithm and distributed duty-cycled algorithm, and discusses their performance gap from the benchmark algorithm in non-duty-cycled case. Section IV is devoted to performance evaluation of the algorithms, followed by conclusions in Section V.

## II. BENCHMARK ALGORITHM IN NON-DUTY-CYCLED CASE

Consider a linear WSN shown in Fig. 1. The network is denoted  $G = (V, E)$ , with  $V = \{0, 1, 2, \dots, N\}$  being the set of nodes and  $E = \{(i, i-1) | 1 \leq i \leq N\}$  being the set of wireless links. Node 0 is the BS, and Node  $i$  is  $i$  hops away from the BS. Each node (say Node  $i$ ) is equipped with an omnidirectional antenna and a half-duplex transceiver, and is able to communicate directly only with its two closest neighbors (Node  $i+1$  on the left handside, and Node  $i-1$  on the right handside).<sup>1</sup> Assume Node  $i$  has  $w_i$  ( $\geq 0$ ) packets, to be sent to the BS. For presentation simplicity, the packets are assigned increasing indexes from Node 1 to Node  $N$ : the  $w_1$  packets of Node 1 are indexed as  $1, 2, \dots, w_1$ , the  $w_2$  packets of Node 2 are indexed as  $w_1+1, w_1+2, \dots, w_1+w_2$ , and so on. Each node, say Node  $i$ , only transmits to its next-hop node, Node  $i-1$ . Without loss of generality, when a node transmits to its next hop, it always sends the packet with the smallest index among all packets at its buffer (including its own packets and its received packets from other nodes). Therefore, during the data collection, it is impossible that a packet with a larger index is closer to the BS than a packet with a smaller index. So, at the BS, Packet 1 arrives first, while the packet with the largest index arrives the last. Our target is optimal data collection algorithm that has the minimum completion time for all the packets to arrive at the BS.

Similar to [1]–[4], time is divided into fixed-length slots, while in each time slot, multiple nodes may be scheduled

to transmit simultaneously (here when we say a node is scheduled, it means the node can transmit a packet).<sup>2</sup> At a time slot, when Node  $i$  is scheduled to transmit (to Node  $i-1$ ), Nodes  $i+1$  and  $i-1$  cannot be scheduled to transmit (to Nodes  $i$  and  $i-2$ , respectively) due to the half-duplex transceiver, and Nodes  $i+2$  and  $i-2$  cannot be scheduled to transmit (to Nodes  $i+1$  and  $i-3$ , respectively) due to the hidden terminal problem. In other words, among any three consecutive nodes, only one node can be scheduled to transmit at a time slot.

For a data collection algorithm, when the packet with the largest index arrives at Node 3 at a time slot (say Slot  $t$ ), all packets that have not arrived at the BS yet are with Nodes 3, 2, and 1. Since only one node can transmit among these three nodes, it does not change the completion time if we arbitrarily schedule the remaining packets in time slots starting from Slot  $t+1$ . Since our target is the minimal completion time of gathering all packets, without loss of generality, we consider only data collection algorithms satisfying the following *last-3-node assumption*: When the packet with the largest index arrives at Node 3 at a time slot, at any subsequent slot the data collection algorithms always schedule the remaining packet with the smallest index to be sent.

Note that all assumptions related to packet index are made to simplify presentation, and they are not mandatory for the proposed algorithms because our goal is to minimize the completion time of data collection. For example, if Node  $i$  is scheduled to transmit by an algorithm, it can transmit any arbitrary packet in its buffer without changing the completion time of the algorithm.

---

### Algorithm 1: Greedy data collection algorithm

---

```

1 begin
2    $t \leftarrow 1$ ;
3   while  $Buffer[0] < w_1 + w_2 + \dots + w_N$  do
4      $i \leftarrow 1$ ;
5     while  $i \leq N$  do
6       if  $Buffer[i] > 0$  then
7         Node  $i$  transmits a packet to Node  $i-1$ 
          at Slot  $t$ ;
           $Buffer[i] \leftarrow Buffer[i] - 1$ ;
           $Buffer[i-1] \leftarrow Buffer[i-1] + 1$ ;
           $i \leftarrow i + 3$ ;
8       else
9          $i \leftarrow i + 1$ ;
10     $t \leftarrow t + 1$ ;

```

---

<sup>1</sup>If Node  $i$  can communicate directly with two nodes on one side, for example, Nodes  $i-1$  and  $i-2$ , since Node  $i-2$  is closer to the destination (the BS), Node  $i-1$  will be removed from the considered node set, and can serve as a backup node (i.e., can be activated if Node  $i-2$  does not work well). Accordingly, the original Nodes  $i, i+1, \dots, N$  will be indexed as Nodes  $i-1, i, \dots, N-1$ , respectively, in the considered node set.

<sup>2</sup>Note that a time slot can accommodate the transmission of a packet and the corresponding acknowledgement (ACK). In specific, each time slot has two portions: transmission portion and ACK portion. If at a time slot, Node  $i$  would like to send a packet to Node  $i-1$ , then at the transmission portion of the time slot, Node  $i$  sends the packet to Node  $i-1$ , and at the ACK portion of the time slot, Node  $i-1$  sends ACK back to Node  $i$ .

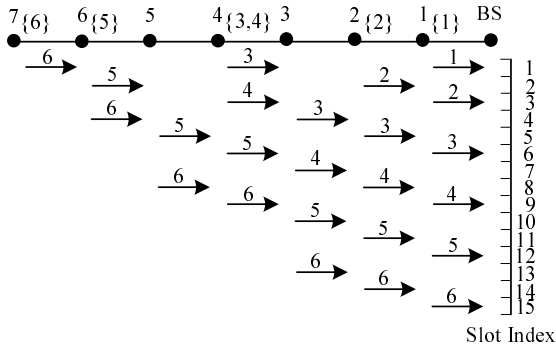


Fig. 2. The greedy data collection algorithm for a 7-node linear WSN.

Since nodes closer to the BS have more traffic load, we propose the above Algorithm 1, which is a greedy algorithm that gives priority to those nodes. In Algorithm 1,  $Buffer[i]$  ( $i = 0, 1, \dots, N$ ) denotes the number of packets at the buffer of Node  $i$ . So at the beginning,  $Buffer[0] = 0$ ,  $Buffer[i] = w_i$ ,  $i = 1, 2, \dots, N$ . In the algorithm,  $t$  denotes time slot index. At  $t = 1$ , we first schedule the node (say Node  $i$ ) that has packets and is closest to the BS. Then Node  $i + 3$  is checked (in Line 10 of Algorithm 1) (recalling that among any three consecutive nodes, only one node can be scheduled to transmit at a time slot): If the node has packets, then it is scheduled to transmit at the slot; otherwise, the node that is one-hop farther away from the BS is checked (in Line 12 of Algorithm 1). This process is repeated until no more node can be scheduled, which completes the scheduling for  $t = 1$ . For  $t = 2, 3, \dots$ , the same procedure is repeated until all the packets arrive at the BS. Fig. 2 shows an example in a 7-node linear WSN. At the nodes, the numbers inside the brackets  $\{\cdot\}$  are the indexes of packets generated at the corresponding nodes. Each arrow means a transmission, while the number above the arrow means the index of the packet that is transmitted. The scale on the right represents the time slot scale. It can be seen that the data collection takes 15 time slots.

In data collection, if Packet  $k$  is generated at Node  $i$ , its transmission from Node  $i$  to the BS can be described by a time schedule  $(t_i^k, t_{i-1}^k, \dots, t_3^k, t_2^k, t_1^k)$ , where  $t_j^k$  ( $j = i, i-1, \dots, 2, 1$ ) is the index of the slot when Node  $j$  transmits Packet  $k$ . For instance, for the scheduling shown in Fig. 2, the time schedules of Packets 1, 2 and 6 are (1), (2, 3), and (1, 4, 8, 9, 13, 14, 15), respectively.

**Lemma 1:** Consider a linear WSN with  $N$  nodes and  $K$  packets to be transmitted to the BS. Assume Packet  $K$  is generated at Node  $i$ . For an optimal data collection solution of the network, denote  $t_1^K$  as the transmission moment (slot index) of Packet  $K$  at Node 1, which is also the minimum completion time of data collection of the  $K$  packets. Suppose another packet, denoted  $K + 1$ , is added at Node  $j$  where  $j \geq i$ . For optimal data collection of the  $K + 1$  packets, denote  $t_n^l$  ( $n = 1, 2, \dots, N; l = 1, 2, \dots, K + 1$ ) as the index of the slot when Node  $n$  transmits Packet  $l$ . Then  $t_1^{K+1}$  is the minimum completion time of the data collection. We have 1)  $t_1^{K+1} = t_1^K + 1$  if  $i = j = 1$ ; 2)  $t_1^{K+1} = t_1^K + 2$  if  $i = j = 2$  or  $i = 1, j = 2$ ; 3)  $t_1^{K+1} \geq t_1^K + 3$  if  $j \geq 3$  and  $j \geq i$ .

*Proof:* Nodes 3, 2, 1 form a bottleneck of the data collection because at any slot, only one node among them is allowed to transmit. Therefore, Cases 1) and 2) are obviously true. For Case 3), at an optimal data collection solution for the  $K + 1$  packets, the time when Node 3 transmits Packet  $K + 1$ , denoted  $t_3^{K+1}$ , must be later than  $t_1^K$ , because otherwise, we can remove Packet  $K + 1$  from the transmission schedule and get a schedule for  $K$  packets that has a completion time less than  $t_1^K$  (recalling that based on the last-3-node assumption, when Packet  $K + 1$  is sent at Node 3, all other packets have already arrived at the BS). And it takes two more slots for Packet  $K + 1$  to be sent from Node 2 to the BS. So  $t_1^{K+1} = t_3^{K+1} + 2 \geq t_1^K + 3$  if  $j \geq 3$  and  $j \geq i$ . ■

**Theorem 1:** Algorithm 1 is optimal.

*Proof:* We use mathematical induction to prove.

If there is only one packet to transmit, it is apparent that Algorithm 1 uses minimum time slots in data collection.

Suppose Algorithm 1 uses minimum time slots in data collection of any  $k$  packets in an  $N$ -node network. Now we use Algorithm 1 to schedule  $k + 1$  packets in the  $N$ -node network. We denote  $t_n^l$  as the scheduling time of Packet  $l$  ( $l \in \{1, 2, \dots, k, k + 1\}$ ) at Node  $n$  ( $n \in \{1, 2, \dots, N\}$ ). We denote  $t_n^l$  as the scheduling time of Packet  $l$  ( $l \in \{1, 2, \dots, k\}$ ) at Node  $n$  when Algorithm 1 is applied to schedule only Packets 1, 2, ..., and  $k$ . Then  $t_1^k$  is the minimum number of slots for data collection of Packets 1, 2, ..., and  $k$ . Apparently  $t_n^l = t_n^l$ ,  $n \in \{1, \dots, N\}$ ,  $l \in \{1, \dots, k\}$ . There are two cases as follows.

Case 1. Suppose Packets  $k + 1$  and  $k$  are both generated at Node  $i$ ,  $i > 3$  (for  $i \leq 3$ , it is easy to prove).

Consider that Algorithm 1 is applied for the  $k + 1$  packets. It can be seen that at any moment Packet  $k + 1$  is within three hops from Packet  $k$ . So at the beginning of Slot  $t_3^k$  (i.e., the slot when Packet  $k$  is sent by Node 3 to Node 2), Packet  $k + 1$  is with Node 6, 5, 4, or 3, and all other packets (Packets 1, 2, ...,  $k - 1$ ) have already arrived at the BS. Then Algorithm 1 takes the subsequent two slots, which are Slots  $t_2^k (= t_3^k + 1)$  and  $t_1^k (= t_3^k + 2)$ , to send Packet  $k$  from Node 2 to the BS. At the end of Slot  $t_1^k$ , Packet  $k + 1$  should be with Node 3. Since all other packets have arrived at the BS at this moment, Algorithm 1 takes three more slots to deliver Packet  $k + 1$  from Node 3 to the BS. In other words,  $t_1^{k+1} = t_1^k + 3$ . Recall that  $t_1^k = t_1^k$  (the minimum time slots needed for data collection of Packets 1, 2, ...,  $k$ ). From Lemma 1, it can be concluded that  $t_1^{k+1}$  is the minimum number of slots for data collection of Packets 1, 2, ..., and  $k + 1$ .

Case 2. Suppose Packets  $k + 1$  and  $k$  are generated at Nodes  $j$  and  $i$ , respectively, with  $j > i$ .

When  $j \leq i + 3$ , it means that Packet  $k + 1$  is initially within three hops from Packet  $k$ . Then when Algorithm 1 is used for the  $k + 1$  packets, at any slot Packet  $k + 1$  is always within three hops from Packet  $k$ . So at the beginning of Slot  $t_3^k$ , Packet  $k + 1$  is with Node 6, 5, 4, or 3. Similar to the proof in Case 1, it can be proved that  $t_1^{k+1}$  is the minimum number of slots for data collection of Packets 1, 2, ..., and  $k + 1$ . So next we consider the scenario when  $j > i + 3$ . Since there are no packets between Node  $j$  and Node  $i$ , by Algorithm 1 Packet  $k + 1$  is transmitted in consecutive slots, starting from Slot 1 until Packet  $k + 1$  is three hops away from Packet  $k$ .

If  $t_3^k \leq j - 6$  (i.e., when Packet  $k$  is scheduled at Node 3, Packet  $k + 1$  cannot reach Node 6), Packet  $k + 1$  is always more than three hops away from Packet  $k$ , and thus, Packet  $k + 1$  is scheduled continuously from Slot 1 until it arrives at the BS at Slot  $t_1^{k+1} = j$ . Apparently this completion time of data collection of the  $k + 1$  packets is minimum. If  $t_3^k > j - 6$ , it means that Packet  $k + 1$  can be within three hops away from Packet  $k$  at a specific slot. Then similar to the proof in Case 1, Algorithm 1 uses the minimum number of slots for data collection of the  $k + 1$  packets. ■

The major drawback of Algorithm 1 is that it is a centralized algorithm and needs global information. So it is to be used to provide design insights and used as a comparison benchmark in Section III.

### III. OPTIMAL AND DISTRIBUTED ALGORITHMS IN DUTY-CYCLED CASE

#### A. Optimal Algorithm in Duty-Cycled Case

In a duty-cycled scheduling, consider that each node (except the BS) is assigned one working slot in a cycle of  $T$  slots. During each cycle, a node (say Node  $i$ ) wakes up at its working slot to receive a packet (if any) transmitted from Node  $i + 1$ . Node  $i$  can also wake up to transmit a packet to its next hop neighbor, Node  $i - 1$ , at the working slot of Node  $i - 1$ . In other slots, Node  $i$  is in sleeping state, to save power. The BS is assumed to be in receiving state all the time. Similar assumptions are also adopted in [5].

Note that the transmission is receiver-based: a feasible transmission from Node  $i$  to Node  $i - 1$  at Slot  $t$  of a cycle must satisfy two conditions: 1) Slot  $t$  is the working slot of Node  $i - 1$ ; 2) Node  $i$  has packets to transmit. Denote the working slot of Node  $i$  as  $\tau_i \in \{1, 2, \dots, T\}$ , and assume  $T \geq 3$ . The duty-cycled mode saves nodes' energy but may increase the completion time of data collection because a node only receives a packet at its working slot in each cycle. Recall that one insight in Algorithm 1 is that simultaneous transmissions should be apart by at least three hops. Therefore, the working slots of the nodes should be assigned such that the working slots of any three consecutive nodes are different from each other. This can be done by a working slot assignment scheme when the WSN is initialized. To achieve this, we propose the following working slot assignment scheme for the initialization stage of a linear WSN with  $T \geq 3$ . When the WSN is initialized, the BS sends a SLOT\_ASSIGN packet, in which there is a field called WORKING\_SLOT with initial value arbitrarily selected from  $\{1, 2, \dots, T\}$ . When a node receives the packet, it takes the value in the WORKING\_SLOT field as its working slot index. The node then forwards the packet to the other side of the network, but with the value in the WORKING\_SLOT field reduced by 1 (if the value equals 0 after the reduction, the value is set to  $T$  instead). By this working slot assignment scheme, the working slots of any  $T$  consecutive nodes are different from each other.

The following algorithm, denoted Algorithm 2, is proposed:

- 1) Node  $i$  ( $2 \leq i \leq N$ ) transmits at Slot  $\tau_{i-1}$  if it has packets;

- 2) at any slot other than Slot  $\tau_2$  and  $\tau_1$ , Node 1, if it has packets, transmits a packet to the BS;
- 3) at Slot  $\tau_2$ , Node 1, if it has packets, transmits when Node 3 has no packet;
- 4) at Slot  $\tau_1$ , Node 1, if it has packets, transmits when Node 2 has no packet.

In other words, in Algorithm 2, each node (except Node 1) can transmit to its next-hop node at the working slot of the next-hop node, while Node 1 can transmit to the BS at slots when Nodes 3 and 2 do not transmit.

In Algorithm 2, Node 1 is given more chances to transmit than other nodes. So Algorithm 2 is also kind of greedy. We have the following theorem.

*Theorem 2:* Algorithm 2 uses minimum number of slots in data collection in a linear WSN with a given working slot assignment in which each node except the BS is assigned one working slot in a cycle, and the working slots of any three consecutive nodes are different from each other.

*Proof:* Consider Node  $i$ ,  $i \in \{4, 5, \dots, N\}$ . Since Node  $i$ 's receiving node, Node  $i - 1$ , has a different working slot from working slots of those nodes that are within two hops of Node  $i - 1$ , it is optimal for Node  $i$ , if it has packets, to transmit a packet to Node  $i - 1$  at the working slot of Node  $i - 1$ . So it is optimal to apply Part 1) of Algorithm 2 to Nodes 4, 5, ..., and  $N$ . Next we consider Nodes 3, 2, and 1. The receiving node of Nodes 3 and 2 are Nodes 2 and 1, with working slot being  $\tau_2$  and  $\tau_1$ , respectively. The receiving node of Node 1 is the BS, which can receive at any slot. Then apparently it is optimal to apply Part 2) of Algorithm 2. Next we prove that, for Slot  $\tau_2$ , it is optimal to apply Part 1) of Algorithm 2 to Node 3, and apply Part 3) of Algorithm 2 to Node 1. In other words, we need to prove that, when both Nodes 3 and 1 have packets at Slot  $\tau_2$ , it is optimal to let Node 3 transmit. We use proof by contradiction. Suppose it is not optimal to always schedule Node 3 at Slot  $\tau_2$  when both Nodes 3 and 1 have packets. Then there exists an optimal schedule, denoted  $\mathcal{O}$ , such that in Slot  $\tau_2$  of some cycles (say  $L$  cycles denoted  $C_L, C_{L-1}, \dots, C_1$  where  $C_L < C_{L-1} < \dots < C_1$ ) both Nodes 1 and 3 have packets, but Node 1 is scheduled. We have the following modification to  $\mathcal{O}$ : Starting from Cycle  $C_1 + 1$ , find the first cycle, denoted Cycle  $c$ , in Slot  $\tau_2$  of which Node 3 transmits and does not have packet left in its buffer after the transmission; then let Node 3 transmit at Slot  $\tau_2$  of Cycle  $C_1$ , and let Node 1 transmit at Slot  $\tau_2$  of Cycle  $c$ . The resulted schedule, denoted  $\mathcal{O}^1$ , is a feasible schedule and has the same completion time as that of  $\mathcal{O}$ . Similarly we modify  $\mathcal{O}^1$  such that Node 3 transmits in Cycle  $C_2$  while Node 1 transmits in a later cycle. The resulted schedule, denoted  $\mathcal{O}^2$ , is a feasible schedule and has the same completion time as that of  $\mathcal{O}^1$ . This procedure is repeated for Cycles  $C_3, \dots, C_L$ , and eventually we get schedule  $\mathcal{O}^L$ , which is a feasible schedule and has the same completion time as that of  $\mathcal{O}$ , and thus, is also optimal. The optimality of schedule  $\mathcal{O}^L$  contradicts our assumption (i.e., it is not optimal to always schedule Node 3 at Slot  $\tau_2$  when both Nodes 3 and 1 have packets) since schedule  $\mathcal{O}^L$  always lets Node 3 transmit at Slot  $\tau_2$  when both Nodes 3 and 1 have packets.

Similarly we can prove that when both Nodes 2 and 1 have

packets at Slot  $\tau_1$ , it is optimal to let Node 2 transmit. ■

Theorem 2 indicates that Algorithm 2 is optimal for duty-cycled scheduling. Since duty-cycled scheduling can be viewed as a special case of non-duty-cycled scheduling, the completion time of Algorithm 2 should be not less than that of Algorithm 1. However, in some special cases (e.g., when nodes have the same number of packets to be sent, which may happen when nodes periodically report their states for network diagnosis or respond to the query of the BS), the two algorithms have the same completion time, as shown in the subsequent Theorems 3 and 4. In the sequel, for presentation simplicity, we assume all algorithms in duty-cycled mode (i.e., Algorithms 2 and 3) use the proposed working slot assignment scheme to assign working slots to nodes.

*Theorem 3:* When there are  $N$  nodes and a BS in a linear duty-cycled WSN, and each node has  $w$  packets to transmit to the BS, the completion time of Algorithm 2 is  $(N-1)w+1$  cycles if  $\tau_1 = T$ , or  $(N-1)w$  cycles if  $\tau_1 < T$ .

*Proof:* At each of the first  $w$  cycles, each node other than Nodes  $N$  and 1 sends a packet and receives a packet, while Node  $N$  sends a packet, and Node 1 receives a packet and it can send at most  $T-2$  packets as it has  $T-2$  “spare” slots (those other than Slots  $\tau_1$  and  $\tau_2$ ). So at the end of Cycle  $w$ , Node  $N$  has no packet. Similarly, at the end of Cycle  $2w$ , Node  $N-1$  has no packet; and so on; and at the end of Cycle  $(N-2)w$ , Node 3 has no packet, Node 2 has  $w$  packets, and Node 1 has at most  $w$  packets (Node 1 has  $w$  packets if  $T = 3$ , or 1 packet if  $T > 3$ ).

From Cycle  $(N-2)w+1$  to Cycle  $(N-1)w$ , Node 1 can transmit in at least two slots in each cycle. If  $\tau_1 = T$  (i.e., the working slot of Node 1 is at the end of a cycle), at the end of Cycle  $(N-1)w$ , Node 2 has no packet, and Node 1 has 1 packet. Then Algorithm 2 takes one more cycle to deliver the packet in Node 1 to the BS. So the completion time of data collection is  $(N-1)w+1$  cycles. If  $\tau_1 < T$ , at the end of Cycle  $(N-1)w$ , Node 2 and Node 1 both have no packet. So the completion time is  $(N-1)w$  cycles. ■

*Theorem 4:* For a linear WSN where each node (except the BS) has  $w$  packets, the completion time of Algorithm 2 with  $T = 3$  and  $\tau_1 < 3$  is the same as that of Algorithm 1.

*Proof:* Denote the number of nodes (except the BS) as  $N$ . For Algorithm 2 with  $T = 3$  and  $\tau_1 < 3$ , the completion time is  $3(N-1)w$  slots, according to Theorem 3.

Next we assume  $N$  is a multiple of 3. The scenarios when  $N$  is not a multiple of 3 can be treated similarly.

For Algorithm 1, we have:

- At the end of Slot  $w$ , the numbers of packets at Nodes 1, 2, ...,  $N$  are  $0, w, 2w, 0, w, 2w, \dots, 0, w, 2w, 0, w, w$ , respectively;
- At the end of Slot  $3w$ , the numbers of packets at Nodes 1, 2, ...,  $N$  are  $0, 0, 3w, 0, 0, 3w, \dots, 0, 0, 3w, 0, 0, w$ , respectively, and there are totally  $(N-2)w$  packets remaining at the nodes;
- Starting from Slot  $3w+1$ , Algorithm 1 delivers a packet to the BS after every three slots.

So the completion time of Algorithm 1 is  $3w+3 \times (N-2)w = 3(N-1)w$  slots, the same as that of Algorithm 2. ■

For a more general scenario (i.e., nodes have different number of packets), the difference of completion time of Algorithm 2 with  $T = 3$  and Algorithm 1 is bounded, to be shown in Section III-B.

### B. Distributed Duty-Cycled Algorithm

To implement Algorithm 2 in a distributed manner, the major challenge is that Node 1 needs to know whether Nodes 3 and 2 have packets to send so that it can “steal” unused slots by Nodes 3 and 2, which is hard to achieve in a distributed manner. Accordingly we let Node 1 only steal unused slots by Node 2 as follows. At the very beginning of Slot  $\tau_1$ , when Node 1 wakes up and senses the channel to be idle (which means Node 2 is not transmitting), then Node 1 transmits at Slot  $\tau_1$ . The resulted distributed duty-cycled algorithm is called Algorithm 3. Since the channel sensing time can be very short (for example, in IEEE 802.11 medium access control standard, 20 microseconds is more than sufficient to detect transmission of other nodes, while a typical time slot duration in WSNs is at least a few dozen milliseconds), this sensing overhead is negligible.

Define  $f_a(w_1, w_2, \dots, w_N)$  as the completion time of Algorithm  $a$  ( $a$  is algorithm index) for a linear WSN with Node  $i$  ( $i = 1, 2, \dots, N$ ) originally having  $w_i$  packets. The following theorem gives a performance bound for Algorithm 3.

*Theorem 5:* For Algorithm 3 with  $T = 3$  and with the proposed working slot assignment scheme, we have

$$f_3(w_1, w_2, \dots, w_N) \leq f_1(w_1, w_2, \dots, w_N) + 3(w_1 + w_2) + 2. \quad (1)$$

*Proof:* Assume by the working slot assignment scheme, the working slots of the nodes are  $(\tau_1, \tau_2, \dots, \tau_N)$ .

We have

$$\begin{aligned} & f_3(w_1, w_2, \dots, w_N) \\ & \leq f_3(0, 0, w_3, w_4, \dots, w_N) + f_3(w_1, w_2, 0, 0, \dots, 0) \quad (2) \\ & \leq f_3(0, 0, w_3, w_4, \dots, w_N) + 3(w_1 + w_2). \end{aligned}$$

Here the second inequality is because when only Nodes 2 and 1 have packets, Algorithm 3’s completion time (units: slots) is at most three times the number of packets in Nodes 2 and 1.

Then consider the following modification of Algorithm 1: change Step 6 from “**if**  $Buffer[i] > 0$  **then**” to “**if**  $Buffer[i] > 0$  and Node  $i$  does not transmit in the previous two slots (Slots  $t-1$  and  $t-2$ ) **then**”. The resulted algorithm is called Algorithm 4. It can be proved that for a linear WSN in which Nodes 2 and 1 do not have packets, Algorithm 4 is optimal. The proof is almost the same as that of Theorem 1, and thus, is omitted here. Since Algorithm 1 is also optimal, we have  $f_4(0, 0, w_3, w_4, \dots, w_N) = f_1(0, 0, w_3, w_4, \dots, w_N)$ .

Consider a linear WSN with  $(0, 0, w_3, w_4, \dots, w_N)$  as the vector of packets originally at the nodes. Consider that Algorithm 4 is used for the WSN. Then in the schedule, each node transmits at most once in any three consecutive slots. We group every three slots into a cycle. In the schedule of Algorithm 4, for each transmission of Node  $i$  ( $i \in \{1, 2, \dots, N\}$ ), if the transmission time is not Slot  $\tau_{i-1}$  of a cycle (here  $\tau_0 \in \{1, 2, 3\} \setminus \{\tau_2, \tau_1\}$ ), then we postpone the transmission time to the next Slot  $\tau_{i-1}$  (which should be either in the current

cycle or in the next cycle). So each transmission is postponed by at most two slots. The resulted transmission schedule is still a feasible schedule. In the postponed schedule, Node  $i$  only transmits at Slot  $\tau_{i-1}$ . Thus, the postponed schedule's completion time is not less than the completion time of Algorithm 3. In other words, we have  $f_3(0, 0, w_3, \dots, w_N) \leq f_4(0, 0, w_3, \dots, w_N) + 2$  (recalling that each transmission is postponed by at most two slots).

Based on the above results and  $f_1(0, 0, w_3, w_4, \dots, w_N) \leq f_1(w_1, w_2, \dots, w_N)$ , we have (1). ■

Theorem 5 indicates that the completion time difference between Algorithm 3 with  $T = 3$  and Algorithm 1 is bounded by  $3(w_1 + w_2) + 2$ . Since Algorithm 3 is implemented in a duty-cycled mode, its completion time is not less than that of Algorithm 2, which is optimal in duty-cycled mode. Therefore, the completion time difference between Algorithm 2 with  $T = 3$  and Algorithm 1 is also bounded by  $3(w_1 + w_2) + 2$ .

Generally, for Algorithm 3, a larger cycle length  $T$  can achieve more energy saving (since a node can be in sleeping state for longer time), but may increase the completion time. On the other hand, a smaller  $T$  can benefit the completion time but consumes more energy.

Next we extend Algorithm 3 to low duty-cycled case by using adaptive duty-cycle technique, to save energy and reduce completion time. In specific, when the WSN is set up, an *idle mode* is used with a large cycle length  $T_{\text{idle}}$ , which is a multiple of 3. The working slots are assigned according to the proposed working slot assignment scheme. In the idle mode, once a node, say Node  $i$  with working slot  $\tau_i$ , receives a packet (i.e., Node  $i+1$  detects an event or has received packets from Node  $i+2$ ), Node  $i$  will forward the packet to the next node (Node  $i-1$ ), and change to *working mode* with cycle length  $T_{\text{work}} = 3$  and use working slot  $((\tau_i - 1) \bmod 3) + 1$ . Node  $i$  also sets up a timer (e.g.,  $2T_{\text{idle}}$  slots) for the working mode. If no further packet is received within the timer duration, then Node  $i$  changes back to the idle mode with cycle length  $T_{\text{idle}}$ . When  $T_{\text{idle}}$  is a multiple of 3, it can be guaranteed that during the switching between the idle mode and the working mode, there is no conflict in the working slots of any three consecutive nodes.

The adaptive duty-cycle technique can strike a good balance between the completion time of data collection and network lifetime: when there is no traffic, nodes work in the idle mode with a large cycle length  $T_{\text{idle}}$  so as to save energy; when there is traffic, the network changes to working mode with  $T_{\text{work}} = 3$  to achieve fast data collection (recalling that completion time of Algorithm 3 with  $T = 3$  is close to the minimum completion time, with a bounded difference, as indicated in Theorem 5), and after all traffic has been delivered, the network returns to the idle mode to save energy.

### C. Discussion

There are still some issues when the distributed algorithm (Algorithm 3) is implemented in real applications. One issue is that time synchronization is needed among nodes so as to implement slotted transmissions. However, only a much loose time synchronization is needed in our distributed algorithm. A

node, say Node  $i$ , only needs to synchronize with its next hop, Node  $i-1$ , at the initialization stage. Such time synchronization can be achieved by a time synchronization protocol such as the Flooding Time Synchronization Protocol (FTSP) in [8] with only a few messages (because of the linear nature of the considered WSNs). After initial synchronization, by the FTSP, Nodes  $i$  and  $i-1$  can have clock synchronization as accurate as 2.24 microseconds through a few packet exchanges between them every 15 minutes. So, it can be seen that the cost of time synchronization is low in our distributed algorithm.

Another issue is unreliable wireless communication links, which may cause transmission failures, and retransmissions are needed. Our Algorithm 3 can still work with transmission failures, as follows. In a linear WSN in which Node  $i$  ( $i = 1, 2, \dots, N$ ) originally has  $w_i$  packets to be sent to the BS, the total number of different packets that should be transmitted by Node  $i$  is  $\sum_{l=i}^N w_l$ . Considering transmission failures, for  $\sum_{l=i}^N w_l$  different packets, denote  $q_i$  as the expected total number of Node  $i$ 's transmissions/retransmissions and denote  $v_i = q_i - \sum_{l=i}^N w_l \geq 0$ . Then the problem is equivalent to data collection in a linear WSN in which Node  $i$  ( $i = 1, 2, \dots, N$ ) originally has  $w_i + v_i$  packets to be sent to the BS and all transmissions are successful. Then the objective of our algorithm is to minimize the *expected* data collection time.

One shortcoming of linear topologies is that any node or link failure (which cannot be solved by retransmissions, such as hardware problem or battery depletion) may disconnect the network. However, as mentioned before, there are many applications in which the WSNs have linear topologies, and it is not cost-effective to deploy a dense mesh network. To solve the node/link failure problem in a linear WSN, we can deploy some nodes as backup nodes, which will become active if node/link failures happen. Even if there is no backup node, nodes around the failed nodes can detect the link failure and take some actions to solve the problem (such as increase the transmission power).

## IV. PERFORMANCE EVALUATION

We evaluate the proposed algorithms over a customized C/C++ simulator. The number of nodes,  $N$ , takes values from 50 to 100. Two typical traffic models are simulated: one is periodic reporting, the other is event detection. For periodic reporting, nodes report data to BS every 30 minutes. In each reporting, for each Node  $i$  ( $i \in \{1, 2, \dots, N\}$ ),  $w_i$  is randomly chosen from 1 to 9. For event detection, assume events occur according to a Poisson process with rate  $\lambda = 2$  per hour, which implies that event inter-arrival times have exponential distribution with mean value being 30 minutes. When an event happens,  $w_i$  randomly takes values from  $\{0, 1, 2, 3\}$  with probability  $\{50\%, 16.7\%, 16.7\%, 16.7\%\}$ . Unreliable wireless links are also simulated by setting the successful probability of a transmission/retransmission to be 0.8. The length of one slot is 20 milliseconds, and the statistics are collected for 120 hours network time.

Fig. 3 and Fig. 4 show the average completion time of Algorithms 1, Algorithm 2 with  $T = 3$ , Algorithm 3 with  $T = 3$  and with adaptive duty cycle (with  $T_{\text{idle}} = 21$  and

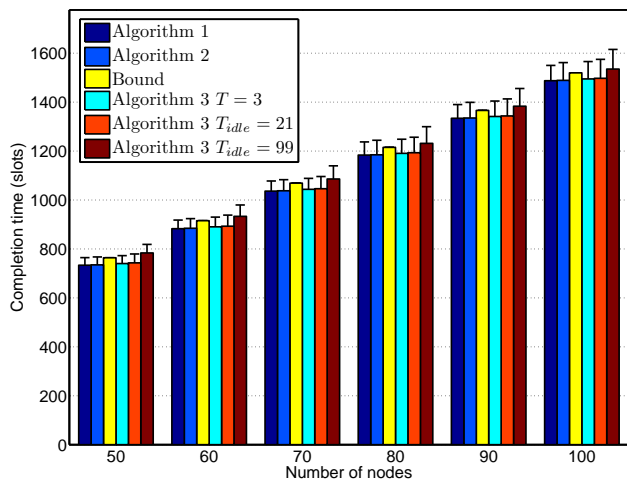


Fig. 3. Completion time of different algorithms for periodic reporting every 30 minutes.

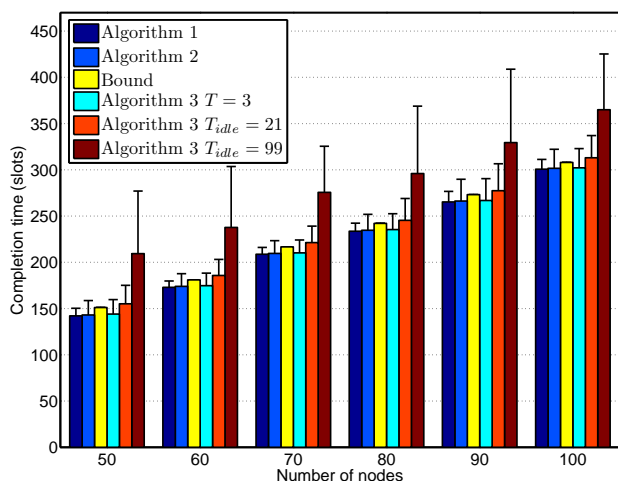


Fig. 4. Completion time of different algorithms for event detection with mean event inter-arrival time being 30 minutes.

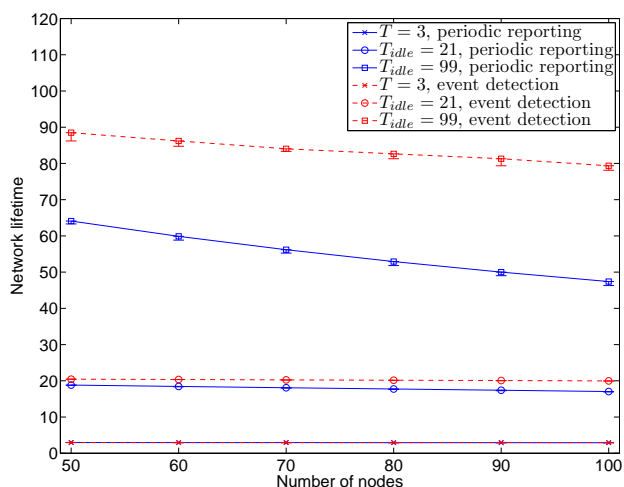


Fig. 5. Normalized network lifetime of Algorithm 3.

$T_{idle} = 99$ , i.e., initial duty cycle is approximately 5% and 1%, respectively), for periodic reporting and event detection, respectively. We also simulate the scheme in [2], and confirm that our Algorithm 1 and the scheme in [2] have the same completion time when the transmission links are reliable (although the schedules in the two algorithms are different). So we do not show the simulation results of the scheme in [2], and only Algorithm 1 is shown as the benchmark. The upper bound of completion time of Algorithm 3 with  $T = 3$  (as indicated in Theorem 5) is also shown in the two figures. In the two figures, the histogram means the average completion time with reliable transmission links, while the line segments above the histogram mean the completion time increase due to unreliable transmission links.

From the two figures, it can be seen that Algorithm 2 with  $T = 3$  has slightly larger completion time than Algorithm 1, while Algorithm 3's completion time with  $T = 3$  is very close to that of Algorithm 2. Compared with Algorithm 1, the completion time of Algorithm 3 with  $T_{idle} = 21$  increases by no more than 2% (3% for unreliable links) for periodic reporting, and it increases by no more than 10% (17% for unreliable links) for event detection. Algorithm 3 with  $T_{idle} = 99$  still performs well for periodic reporting, while its completion time increases apparently for event detection. The reason is as follows. For periodic reporting, as all nodes have packets to send, they can quickly switch to working mode. For event detection, the traffic is sparsely distributed along the linear network, and it takes some slots (e.g., half a cycle on average) for downstream nodes to change to working mode. And it takes much more slots when transmission links are unreliable, as indicated by Fig. 4.

We next evaluate the network lifetime of Algorithm 3 with  $T = 3$  and with adaptive duty cycle. In each cycle, a node is in receiving state at its working slot, in transmitting state if it transmits at the working slot of its next-hop neighbor, or in sleeping state at other slots. According to [9], the power consumption for receiving and transmitting in a WSN are similar, while the power consumption in sleeping state is very small and thus can be neglected. Therefore, in our simulation, we assume that the energy consumption of a node at a time slot is a constant positive value if the node is in transmitting or receiving state, and the energy consumption is zero when the node is in sleeping state. Since all packets have to go through Node 1, we take the lifetime of Node 1 as the network lifetime. We regard the lifetime of Node 1 when it is always in receiving or transmitting state as one, and we measure the normalized network lifetime of our Algorithm 3 with  $T = 3$  and with adaptive duty cycle (with  $T_{idle} = 21$  and  $T_{idle} = 99$ ), as shown in Fig. 5. In the figure, the line segment under each discrete mark means the decrease of network lifetime due to unreliable transmission links. It can be seen that for a given traffic model (periodic reporting or event detection), the network lifetime is approximately proportional to the cycle length. The network lifetime tends to decrease as the number of nodes in the network increases, as shown in Fig. 5. This is reasonable because, with more nodes, more traffic may be generated which eventually needs to be transmitted by Node 1.

## V. CONCLUSIONS

For data collection in a linear WSN, we first provide an optimal solution for non-duty-cycled case as a benchmark, and then we propose an optimal algorithm for duty-cycled case and a distributed duty-cycled algorithm. Our results will provide theoretical performance bounds for data collection time in linear WSNs. The insights gained in our proposed algorithms can be helpful for developing data collection algorithms for WSNs with other topologies.

## REFERENCES

- [1] S. C. Ergen and P. Varaiya, "TDMA scheduling algorithms for wireless sensor networks," *Wireless Networks*, vol. 16, no. 4, pp. 985-997, May 2010.
- [2] C. Florens and R. McEliece, "Packets distribution algorithms for sensor networks," in *Proc. IEEE INFOCOM'03*, pp. 1063-1072, 2003.
- [3] S. Gandham, Y. Zhang, and Q. Huang, "Distributed time-optimal scheduling for convergecast in wireless sensor networks," *Computer Networks*, vol. 52, no. 3, pp. 610-629, Feb. 2008.
- [4] O. D. Incel, A. Ghosh, B. Krishnamachari, and K. Chintalapudi, "Fast data collection in tree-based wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 11, no. 1, pp. 86-99, Jan. 2012.
- [5] Y. Gu and T. He, "Dynamic switching-based data forwarding for low-duty-cycle wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 12, pp. 1741-1754, Dec. 2011.
- [6] S. Lai and B. Ravindran, "On distributed time-dependent shortest paths over duty-cycled wireless sensor networks," in *Proc. IEEE INFOCOM'10*, 2010.
- [7] S. Xiong, J. Li, M. Li, J. Wang, and Y. Liu, "Multiple task scheduling for low-duty-cycled wireless sensor networks," in *Proc. IEEE INFOCOM'11*, pp. 1323-1331, 2011.
- [8] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *Proc. 2nd International Conference on Embedded Networked Sensor Systems (SenSys'04)*, pp. 39-49, 2004.
- [9] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. IEEE INFOCOM'02*, pp. 1567-1576, 2002.