IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 19, 2018, accepted January 1, 2019, date of publication January 10, 2019, date of current version February 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2891944

Energy Efficient Scheduling in Wireless Sensor Networks for Periodic Data Gathering

SAURABH KUMAR, (Member, IEEE), AND HYUNGWON KIM, (Member, IEEE)
Department of Electronics Engineering, Chungbuk National University, Cheongju 28644, South Korea

Corresponding author: HyungWon Kim (hwkim@cbnu.ac.kr)

This research was supported by KIAT (Korea Institute for Advancement of Technology) grant funded by the Korea Government (MSS: Ministry of SMEs and Startups) (No. N0001132). This work was also supported in part by the Center for Integrated Smart Sensors funded by the Ministry of Science, ICT & Future Planning as Global Frontier Project, South Korea, (CISS-2018).

ABSTRACT Nodes in the wireless sensor networks are battery operated, and thus, the efficient use of the node's energy during wireless communication is pivotal for the long battery life. This paper addresses the challenges in energy-efficient data aggregation and transmission scheduling for each node by using time division multiple access (TDMA). We introduce multi-channel TDMA scheduling algorithms with the objective of minimizing the total energy consumption in the network. The proposed algorithms utilize multiple radio channels to bestow efficient scheduling while eliminating collisions and overhearing. First, we formulate an integer linear programming (ILP) algorithm that finds the minimum bound for the network's energy consumption. Then, we propose a near-optimal heuristic algorithm based on backtracking, which uses memoization to spurn the suboptimal schedules. Subsequently, we propose a computationally efficient heuristic algorithm by using Langford subset generation. The algorithm reduces the energy consumption in each timeslot while avoiding revisiting the same timeslot. We conducted extensive simulations to evaluate three proposed algorithms. The simulation results demonstrate that the proposed heuristic algorithms provide performance comparable to the optimum results of the ILP algorithm and reduce the magnitude of computation time.

INDEX TERMS Wireless sensor network, Internet of Things, meter reading, scheduling algorithm.

I. INTRODUCTION

Wireless sensor networks are changing the traditional way of data gathering [1]. Traditional devices can be replaced by the battery powered wireless sensors [2]. They capture data from the environment and then transmit the captured data to the sink node through multi-hop communication. The sink node accumulates the data from all the nodes and forwards to the control server via a wire-line network for further processing and analysis. In periodic data gathering applications such as automatic metering infrastructure (AMI), the nodes wake up and capture the data in each period [3]. Additionally, the nodes are expected to function continuously for the long time (2-10 years). Hence, in the case of drainage, the battery is replaced. However, each of the WSN nodes should consume minimum energy so that the batteries last for a longer period of time, in order to minimize the number of replacements. Moreover, all the nodes in the network should also not be drained at the same time since the network administrator cannot replace them simultaneously.

Nodes consume energy during sensing, computation, and communication. In this paper, we focus only on energy consumed in the communication process. To minimize the energy consumption in the communication, WSN protocols should address the following problems [4]:

A. COLLISIONS

Collision raises extra reception cost in the destination node and unnecessary retransmission cost in the source node. Collisions can be classified into two types. The first type is the inter-network collisions, which occur when a node receives the packets concurrently from two or more transmitters of the same network. Collisions of this type can be avoided by deterministic scheduling such as TDMA timeslot assignment. In addition, such collisions can be also avoided when the concurrent communications use different radio frequency (RF) channels. The second type is the intra-network collisions, which occur due to interference from the other networks or devices using the same RF channel. Use of carrier sensing at

the start of the TDMA timeslot before transmitting can help eliminate collisions of this category.

B. OVERHEARING

Unicast packets are transmitted to a single destination. Due to the broadcast nature of the wireless radio, however, all one-hop neighbors (within wireless range) of the source node will hear the transmitted packet if their transceiver is in receiving mode. The nodes, with the exception of the target receivers, drop the overheard packets. The overhearing problem can be alleviated by switching off the non-targeted nodes' transceivers. This can reduce a significant portion of the energy consumption [5]. The TDMA based protocol can serve as an effective solution for eliminating overhearing by assigning timeslots for transmitting and receiving data packets. However, TDMA still cannot eliminate overhearing during waiting in the idle mode between receiving timeslots and the transmitting timeslot. Each node can switch off its transceiver once its transmission in the assigned timeslot is completed.

C. PROTOCOL OVERHEAD

Each of the physical, medium access control (MAC), and network layers append their header as well as trailer information [6] in every transmitted packet. In the case of secure communication, security protocol further augments the size of the packet substantially [7]. In the AMI applications, the size of the sensing data is often small. Thus, the protocol overhead is usually larger than the data generated by the sensor node. Data aggregation [8] can reduce the protocol overhead with respect to per packet data transmission and the number of transmitted packets. That results in reducing the total energy consumption in the network. Therefore, we propose a new aggregation protocol. In the proposed protocol, each node appends the data received from its subtrees with its own sensing data and construct a single packet, which increases the size.

D. IDLE LISTENING

A node in the idle mode is ready to receive but not currently receiving. Such a node consumes as much energy as a node that is receiving a packet [9]. Therefore, switching off the transceiver during the idle mode is quite essential to reduce unnecessary energy consumption. Since the transceiver's startup (sleep to active) energy is high [9], its frequency should be minimal. In the proposed convergence network for the AMI, each sensor node wakes up to receive the data packet from its child nodes. Since leaf nodes do not receive data from any nodes, they directly wake up at their transmission timeslot. Nodes other than leaf nodes, after receiving, subsequently aggregate the received data and also append their own sensing data. After finishing the transmission of the final aggregated packet, the nodes immediately switch off their transceivers. The above process is repeated in each data collection period.

As a result, the proposed algorithms reduce the energy consumption of the transceiver by addressing the above problems. In this paper, we propose scheduling algorithms that assign a timeslot and RF channel to each node. The algorithms take the aggregation spanning tree as the input and produce a contention-free/TDMA multi-channel schedule. In the proposed algorithms, a data packet is aggregated in each node and forwarded to the sink node in a multi-hop fashion. The algorithms are designed for energy efficient data gathering applications and thus enable the networks to prolong the battery replacement time. The main contributions of this paper are as follows:

- Conventional TDMA MAC protocols require frequent re-synchronization, which is a noteworthy battery draining operation. Nodes experience time drift due to frequency offset between clocks, if not re-synchronized. The proposed protocol defines a new TDMA slot that does not need frequent re-synchronization since it uses the synchronization margin, as explained in section III.A.
- In section III.B, this paper defines a new aggregation protocol, that reduces the protocol overhead, while ensuring the raw data generated by each sensor node is delivered to the sink node.
- We devise an integer linear programming (ILP) formulation that determines an optimum solution for the data gathering from all the wireless sensor nodes in a period. The objective of the ILP equations are to minimize energy consumption. This is explained in section III.D.
- Calculating an ILP solution requires high computational power for large networks. Therefore, we present heuristic algorithms: First a memoization based backtracking algorithm, which is nearly as optimal as the ILP solution. Later we present a Langford-subset based algorithm, which is best suited for the sensor network applications due to high computational power. Section IV describes the heuristic algorithms in detail.
- We analyze the bounds of computational complexity for the ILP solution (see section III.E) as well as for both the heuristic algorithms (see section IV).

Section V.A outlines setup and parameters used in the simulation. Section V.B shows extensive simulations to analyze computational complexity, energy consumption and latency of the network. Additionally, simulations also present the effectiveness of aggregation protocol. To end, Section VI discusses the conclusion and future work on TDMA scheduling for periodic data gathering applications.

II. RELATED WORK

TDMA scheduling with data aggregation has been studied by [8] and [10]–[14]. Upadhyayula *et al.* [8] proposed a tree-based scheduling protocol with the known compression (aggregation) factor γ or data growth factor $\alpha(= 1 - \gamma)$. Their protocol chooses the timeslot and code division multiple access (CDMA) code for transmission of each node's sensing data. The authors examined their protocol for three different aggregation ratios (0%, 50% and 100%) by using single channel. Our proposed protocol, on the other hand,

presents scheduling algorithms that use multiple RF channels. Incel *et al.* [10] classified the data gathering schemes as raw data forwarding and aggregated forwarding. The authors have proposed multi-channel scheduling algorithms based on two aggregation protocols, one-shot raw-data convergecast and periodic aggregated convergecast. In the raw data forwarding, each node's generated data is forwarded in an individual packet to the sink node, without any aggregation. On the other hand, in the periodic aggregated convergecast, each node generates the same size data after aggregating the data from its child nodes by using spatial correlation [15]. The protocol needs multiple data gathering periods to refresh the data from all the nodes. However, our target application requires raw sensing data, thus cannot utilize the spatial correlation based aggregated convergecasting.

Minimum latency scheduling with the latency bound of $O(24D + 6\Delta + 16)$ is proposed by Yu *et al.* [11], where D is the diameter of the network and Δ is the maximum degree of the nodes in the network. The authors have assumed in-network data aggregation, which is same as periodic aggregated forwarding [10]. They used only a single channel, unlike our proposed algorithms of multiple RF channels. Similar to [11], Erzin and Pyatkin [12] have also proposed in-network aggregation scheduling using a single channel. However, the authors have considered only the special cases of rectangular, triangular, and hexagonal grid networks. Annamalai *et al.* [13] have proposed an in-network aggregation heuristic algorithm, called the convergecasting tree construction and channel allocation algorithm (CTCCAA), which schedules collision-free data gathering. For the data collection from all the nodes within a period itself, the authors have allocated the parent node with a higher timeslot than the timeslots of the child nodes. The authors have used direct sequence spread spectrum (DSSS) codes instead of multiple RF channels. Therefore, the CTCCAA can be used in our target application with the framework changes. By using RF channels instead of DSSS code and data-appending forwarding instead of in-network aggregation, we have implemented the CTCCAA to compare the results with our proposed algorithms.

Akila *et al.* [14] presented an efficient TDMA scheduling algorithm by reducing the number of transmitted packets using merging of the same destination packets. Nodes continue to merge until the size of appended packet reaches the maximum size or maximum waiting is elapsed. Though, the merge operation is performed only for non-real time packets. Since, all the packets are destined to the same destination (sink node) and sensing data is not real time data. This algorithm is best fit for performance comparison. However, we have tailored the algorithm to use multiple channels and denoted by TDMA scheduling based on data merging (TSDM).

In our previous work, we have proposed TDMA scheduling algorithms with aggregation [16]–[18]. In [16], we introduced binary linear programming (BLP) formulation that schedules data forwarding with multiple RF channels, in which nodes wake up simultaneously, receive, then transmit their aggregated packet, and switch back to sleep mode. In [17], we formulated an ILP algorithm for the data aggregation scheduling in periodic data gathering to minimize energy consumption. The ILP algorithm uses multiple RF channels for concurrent transmissions in the wireless interference range. While the ILP formulation provides the optimum solution, the complexity grows rapidly with the size of the network. Hence, we proposed a heuristic algorithm, linear time scheduling algorithm (LTSA) in [18]. The LTSA uses the number of hops to the sink node as the cost metric. Nodes in the queue are stored based on the hop count, hence the computational complexity of the LTSA is linear. The proposed paper is an extension of the ILP algorithm introduced in [17]. It enhances the ILP algorithm with a realistic energy consumption model. In this paper, we provide proof of the ILP algorithm attaining the lower bound of energy consumption. In addition, we propose two heuristic algorithms: the first one is aimed for a suboptimal solution close to the ILP algorithm, while the second one is targeted for the low computational complexity. We evaluate the theoretical upper bounds of computational complexity for both the algorithms, as well as compare their efficiency using simulation results.

III. PRELIMINARY

A. TIMESLOT INFORMATION

In the proposed method, the time division multiple access (TDMA) protocol divides the time frame of a data gathering period into a set of short timeslots of fixed length. Each node of the network is assigned a timeslot and it periodically transmits an aggregated data packet to the parent node in the assigned timeslot.

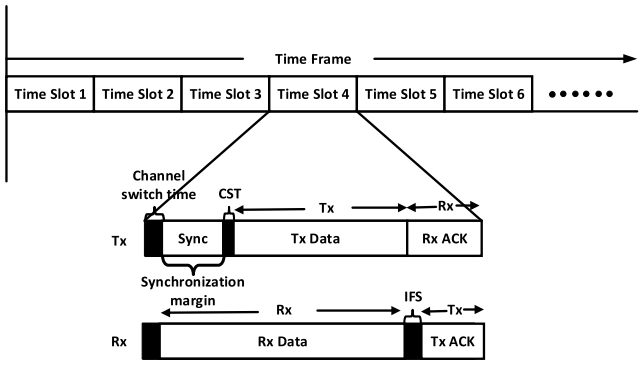

FIGURE 1. Division of time frame into timeslots, and the Tx and Rx's operation modes for a timeslot.

As shown in Fig. 1, the similar timeslot is used for the Tx operation in the transmitting node, while it is used for the Rx operation in the corresponding receiving (parent) node. The timeslot begins with a channel switch time, which is the short time reserved for switching the channel to a new channel allocated to the transmission. In the channel switch time, a device can neither transmit nor receive any packet. In addition to the channel switch time, Tx operation

is comprised of synchronization margin time (T_{sync}), carrier sensing time (CST), Tx data interval (time required to transmit the maximum size data packet), and Rx Ack interval (time required to receive an acknowledgment packet back from parent).

The synchronization margin time accounts for the fact that two communicating nodes may not be absolutely synchronized in time due to clock offset between them.

The receiver node can receive data; however, the transmitter does not transmit in T_{sync}. As a result, even if the TDMA protocol requires a tight synchronization, the proposed protocol can sustain up to $3 \times \frac{\text{synchronization-margin}}{2}$ time synchronization error [19], which decreases the need for frequent re-synchronization.

In the CST, the transmitting node senses the RF channel condition by detecting radio signals to avoid the packet collision from other networks in the vicinity. If the node detects an on-going communication (medium busy), it does not transmit the packet in the given period. Similar to the synchronization margin, the receiving node can receive the packet in the CST as well.

On the other hand, in the Rx operation, the channel switch time is followed by Rx data interval (for receiving maximum size data packet and also for compensating the synchronization margin time in the Tx node), interframe space (IFS), and Tx Ack interval (for transmitting an acknowledgment). The receiver node switches to receiving mode and ready to receive just after channel switch time. Thus, it can receive data at any point in the Rx data interval time. Based on the current time offset between the transmitting node and the receiving node, synchronization margin time is automatically adjusted in the Rx data interval (may split in two parts, one at the start and another at the end). The IFS is the time delay required between the end of packet transmission and the start of the acknowledgment transmission. This includes packet processing latency in the physical and MAC layers as well as channel sensing and signal propagation delay. Latency is measured from the time of the very first leaf node's transmission to the time of the sink node's last child's transmission.

B. DATA-APPENDING FORWARDING

Incel *et al.* [10] discussed aggregation protocols for two extreme cases of no data compression (raw-data forwarding) and full data compression (aggregated forwarding). The raw data forwarding relays data packets generated by each node individually. In contrast, in aggregated forwarding, the root node aggregates the data from each node in the subtree into the same size packet. Then it sends only one aggregated packet from the subtree's root node. The aggregated forwarding is applied when the data have a strong spatial correlation [15] or the goal is to gather summarized information, such as the maximum, minimum, or average. In this paper, we propose a new aggregation protocol. In the proposed protocol, the root node of each subtree appends the sensing data received from each node in the subtree with its own data.

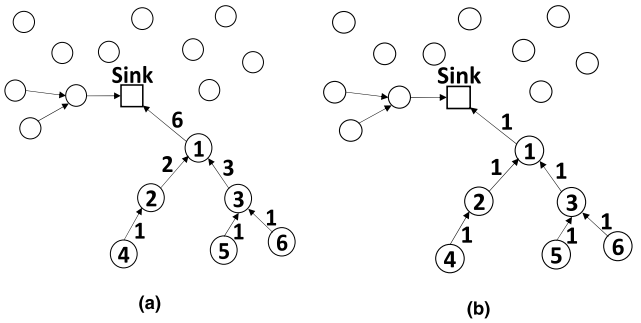

FIGURE 2. The number of packet transmissions for each node: (a) when raw-data forwarding is used, (b) when the proposed periodic aggregation is used.

In this paper, we consider AMI networks, where the individual sensing data is small in size. In general, it is well known that the transmission accounts for a substantially larger portion of the energy consumption than reception or data processing energy [9]. Fig. 2 compares the number of transmit packets for two aggregation algorithms. Fig. 2(a) illustrates a raw-data forwarding aggregation and Fig. 2(b) shows the proposed data-appending forwarding aggregation. In Fig. 2(a), each node's sensing data is transmitted in an individual packet. In the given example, node 1 transmits 6 packets. In contrast, in Fig. 2(b), node 1 aggregates the data from all the nodes in its subtree and transmits only one aggregated packet. While the size of aggregated packet increases with the number of nodes, it can still be transmitted in the Tx data interval defined in Fig. 1. The proposed aggregation, therefore, can eliminate the header overhead for 5 packets in Fig. 2(b) for node 1.

In the proposed protocol, a routing protocol [20] allocates the parent node, such that it satisfies the following constraints,

- Let each node generate a sensing data of size k, and the size of the header is h. Table 1. Specify the values used for all the parameters used in the paper. Then the total data size of an aggregated packet for any node v is computed by Eq. (1).

$$\text{data}_{\text{Tx}_v} = (\text{h} + \text{k}) + \sum\nolimits_{\forall \text{ node} \in \text{subtree(v)}} \text{k} \qquad (1)$$

- As expressed in Eq. (2), the size of the aggregated packet should not exceed the maximum packet size L, which can be transmitted in the Tx Data interval of one timeslot.

$$\text{data}_{\text{Tx}_v} \leq \text{L} \qquad (2)$$

For example, Node 5 and Node 6 in the Fig. 2(b) transmit data of size h + k, which are received by Node 3. After aggregation at Node 3, the size of data transmitted by it to the Node 1 becomes h + 3k. Similarly, Node 1 transmits a data of size h + 6k to the sink node, which should be less than or equal to L.

C. SENSOR NETWORK MODEL

This section describes the network topology considered in this paper. The network consists of N homogenous wireless sensor nodes distributed randomly over a target area.

TABLE 1. Simulation parameters.

Parameter	Value
Maximum packet size (**L**)	255 bytes
Data rate	76.8 kbps
TDMA timeslot length	33 ms
Sensing data size (**k**)	6 bytes
Header information size (**h**)	16 bytes
Ack packet size	21 bytes
Channel switch time	64 μs
Synchronization margin time	2 ms
Carrier sensing time (CST)	152 μs
Interframe space (IFS)	487 μs
$\mathbf{E_{Tx}}$	148.5 mJ/s
$\mathbf{E_{Rx}}$	56.1 mJ/s
$\mathbf{E_{Idle}}$	52.8 mJ/s
$\mathbf{E_{Sleep}}$	0.0033 mJ/s
Modulation	Frequency Shift Keying (FSK)
$\mathbf{P_{out}}$	+13 dBm
Transmission distance	250 m
$\mathbf{E_{Total}}$	3000 mA
$\mathbf{V_{out}}$	3.3 V

Each WSN node can communicate with the other nodes in the wireless range. Sensors are equipped with a battery of E_{Total} initial energy. The nodes attempt to extend its battery life by consuming minimum energy from E_{Total} in each data gathering period. Hence, the proposed algorithms determine the transmission schedule of timeslot and RF channel for the minimum energy consumption in the network.

NETWORK ARCHITECTURE

In this paper, WSN is represented by a directed graph G = (V, E). Here, V represents the set of N wireless sensor nodes including the sink node. Each sensor node v_i, except the sink node v_s, periodically generates data of size k. E is the set of directed edges (v_i, v_j), where v_i is the child node and v_j the parent node. Direction of the edge $v_i \rightarrow v_j$ denotes the direction of the data forwarding. The edges constitute a data forwarding spanning tree with the sink node as root. We assume that the industrial, scientific, and medical (ISM) band supports multiple radio frequencies for the concurrent communications without interference. Assigning RF channels to a set of concurrent communications is NP-Hard [21]. Hence, the proposed protocols assume that the network is available with the maximum of S RF channels.

In the proposed protocol each node v_i wakes up at the first of its children's timeslot to receive data and switch off the transceiver after its transmission timeslot. However, if any node does not have children then, it will wake up directly in its timeslot. The proposed scheduling algorithm allocates the timeslots such that the network consumes the minimum energy in one period of data collection.

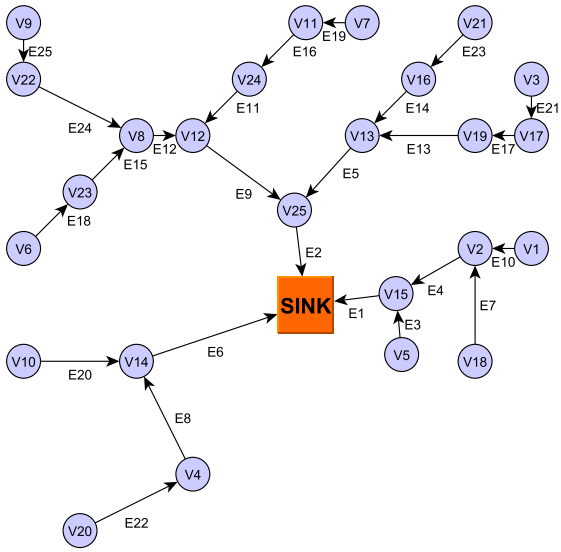

FIGURE 3. Routing graph of a network with 25 sensor nodes generated by Wiser simulator.

Additionally, for each node, it also determines the RF channel used for transmission. Whenever a parent node v_j wants to receive the data from the child node v_i, v_j switches its channel to v_i's allocated RF channel at v_i's timeslot. An example directed graph G = (V, E) of 25 nodes network is illustrated in Fig. 3. We assume that the sink node possesses the routing information and executes the scheduling algorithms. The algorithms determine the schedule of the timeslot and RF channel for each node. Afterwards, the sink node broadcasts the schedule and each node stores the schedules of itself and its child nodes.

D. INTEGER LINEAR PROGRAMMING (ILP) ALGORITHM

In the proposed protocol, each sensor node v_i is entailed to receive data from v_i's all child nodes before transmitting to v_i's parent node. Hence, node v_i wakes up early enough to receive the data from its earliest timeslot child node. v_i then receives from the other child nodes one at a time or wait in the idle mode till its' transmission timeslot. Once the data is received from all the child node, v_i aggregate the received data with its own data. It then transmits the aggregated data packet to its parent node in its allocated timeslot using the allocated RF channel. Multiple pairs of nodes can communicate concurrently in the same timeslot, if they use different RF channels. Concurrent communication minimizes the latency of data collection as well as the idle waiting time.

PROBLEM FORMULATION

The objective of the proposed integer linear programming (ILP) formulation is to minimize the sum of energy consumption of each node in the network. ILP algorithm deduces a set of tuples, $(T_{v_1}, C_{v_1}), (T_{v_2}, C_{v_2}), \ldots, (T_{v_N}, C_{v_N})$ for each of N nodes in the directed graph G = (V, E). Here, T_{v_i} is the timeslot and C_{v_i} is the RF channel allocated to node v_i for data transmission to its parent.

In the beginning of each transmission, the transmitter stays in the idle mode for synchronization margin time (T_{sync}) and CST. After that the transmitter sends the aggregated data packet to the receiver and receives an acknowledgment (Ack) packet back from the receiver. Thus, the total energy consumed by transmitting node v can be expressed by Eq. (3).

$$E_{Tx_v} = T_{sync} \times E_{Idle} + data_{Tx_v} \times E_{Tx} + data_{Ack} \times E_{Rx} \tag{3}$$

In the Eq. (3), E_{Idle} is the energy consumed by the transceiver in the idle mode (transceiver is in receive mode and waiting to receive data but not receiving) in unit time.

E_{Tx} and E_{Rx} are the energy consumed by the transceiver to transmit and receive one bit, respectively. Parameter $data_{Tx_v}$ is same as defined in Eq. (1) and $data_{Ack}$ is the data size of the acknowledgement packet.

After changing channel in CST, the receiver node is ready to receive the packet. Additionally, the receiver node v also wait in the idle mode for the T_{sync}, which can occur at the start, end or split in both the places of receiving packet. Once node receives the data packet, it sends an acknowledgement packet back after the IFS time. Node v receives data from all its child nodes before it transmits the aggregated data packet. Hence the total energy consumption for any node v while receiving data from all children can be represented by Eq. (4).

$$E_{Rx_v} = \sum_{\beta \in child(v)} (T_{sync} \times E_{Idle} + data_{Tx_\beta} \times E_{Rx} + data_{Ack} \times E_{Tx}) \tag{4}$$

Here $data_{Tx_\beta}$ represent the size of the data transmitted by the node v's child β. Eq. (4) calculates the energy consumed by the node v to receive data packet from all child nodes.

Each node v wakes up to receive the data from its earliest scheduled child. Afterwards, node v waits in the idle mode, while not receiving from other child nodes or until its transmitting timeslot. Node v, therefore, consumes a sizable portion of its energy during the ideal listening. The energy consumed by node v in the idle mode is expressed by Eq. (5).

$$E_{Idle_v} = T_{slot} \times E_{Idle} \times \left(T_v - \min\left(T_\beta | \forall_{\beta \in child(v)}\right) - \sum_{\beta \in child(v)} 1 \right) \tag{5}$$

where T_{slot} is the length of one timeslot and quantity inside the parenthesis of Eq. (5) represents the number of timeslots in the idle mode for node v in one period. In which, T_v is the node v's timeslot. Term $\min\left(T_\beta | \forall_{\beta \in child(v)}\right)$ finds the earliest timeslot among the timeslots T_β of each child node β of node v, which is same as the node v's wakeup timeslot. Since receiving energy is already calculated in Eq. (4), the last term, $\sum_{\beta \in child(v)} 1$ subtracts the number of timeslots used in receiving data from the child nodes.

After transmitting the aggregated data packet, the transceiver goes to sleep mode (consumes negligible energy) hence, the total energy consumed by node v in one data gathering period is calculated by Eq. (6).

$$E_v = E_{Tx_v} + E_{Rx_v} + E_{Idle_v} \tag{6}$$

The objective of the ILP can be transcribed to minimizing the sum of energy consumption for all the nodes, which is expressed by Eq. (7).

$$\text{Minimize: } \sum_{v \in V} E_v \tag{7}$$

The constraints of the ILP algorithm are expressed in the following inequalities Eq. (8) - (12).

a: DATA AGGREGATION AND ROUTING CONSTRAINTS

The aggregated packets are transmitted from the nodes to their parent node after receiving packets from their child nodes. Hence the timeslot allocated to a parent node must be after the timeslots of all its child nodes. For example, in Fig. 3 the value of the timeslot for node V12 must be larger than the values of timeslots for its child nodes V8 and V24. The constraint can be translated to "for all the directed edges in the graph, the parent node should be allocated at a higher timeslot than the child nodes". Eq. (8) ensures that for each directed edge $e_m : v_i \rightarrow v_j$ in the graph $G = (V, E)$, the parent node v_j can collect data from its child nodes v_i before transmitting.

$$\forall_{e_m:\, v_i \rightarrow v_j \in E} T_{v_j} - T_{v_i} > 0 \tag{8}$$

b: HALF DUPLEX SINGLE RADIO CONSTRAINT

Having multiple radios causes extra power consumption and escalates the cost. In the addressed application, each node is equipped with a single transceiver. A node cannot receive the data from its multiple children concurrently hence, child nodes must have different timeslots. In other words, any two children should not be allocated with the same timeslot. For example, in Fig. 3, node V12 has two child nodes V8 and V24. Nodes V8 and V24 should be allocated to different timeslots, so that their parent node V12 can receive data from both the nodes successfully without collision. The generalized formula for single radio constraint is expressed by Eq. (9). It describes that for any node v_i all the inbound edges (indicating the child nodes of v_i) should not share the same timeslots.

$$\forall_{v_i \in V} \left(\forall_{a,b} T_{v_a} \neq T_{v_b} | ((e_m : v_a \rightarrow v_i) \cap (e_n : v_b \rightarrow v_i) \cap e_m, e_n \in E) \right) \tag{9}$$

In addition, the wireless transceivers operate in the half-duplex communication mode. Thus, a node cannot transmit and receive simultaneously. The constraint for half-duplex is already addressed by the constraint Eq. (8), which ensures that the node's transmission timeslot is allocated after its child nodes' timeslots.

c: MAXIMUM RF CHANNELS CONSTRAINT

The ISM band supports multiple radio frequencies called channels. Among all the channels, we consider the set of channels that are exclusive to others and allow multiple transmitters to communicate in these channels without interference. Therefore, the network can allocate the same timeslot to the maximum of S nodes for concurrent communication. Where, S is the maximum number of available RF channels.

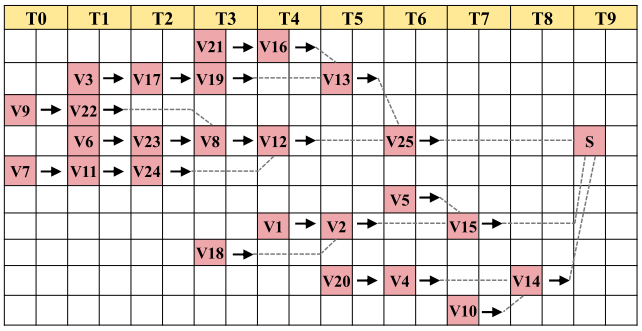

FIGURE 4. Gantt chart of schedule allocation by ILP algorithm for a 25 nodes network of Fig. 3, sink node is represented by S.

Fig. 4 illustrates an example scheduling result for the network of Fig. 3 in a form of Gantt chart using 4 RF channels. In Fig. 4, we can observe that the number of nodes allocated to each timeslot is less than $S = 4$. It requires 10 timeslots to satisfy $S = 4$.

Eq. (10) illustrates the constraint for the maximum RF channels.

$$\forall_{l\in 0\ldots N-1}\left(\sum_{v_i\in V}\left(T_{v_i} == l\right)\right) \leq S \tag{10}$$

In Eq. (10), timeslot value l varies between 0 to $N-1$, where N is the number of nodes. The value of the RF channel can vary between 1 to S, the maximum available channels.

Consequently, the basic constraints on the integer variable tuple (T_{v_i}, C_{v_i}) are expressed in Eq. (11) and Eq. (12).

$$0 \leq T_{v_i} \leq N-1 \tag{11}$$

$$1 \leq C_{v_i} \leq S \tag{12}$$

Here, T_{v_i} denote the timeslot allocated to node v_i, while C_{v_i} is the channel assigned to node v_i.

The energy consumption in the transmission, reception and idle mode is mentioned in Table 1, which also mention the length of one timeslot. The example allocation of Fig. 4 (ILP algorithm) consumes 81.03 mJ energy and takes 330 ms time in each data gathering period.

E. ASYMPTOTIC ANALYSIS OF ILP ALGORITHM

Lemma 1. For a directed graph $G = (V, E)$ representing a network of N nodes, the minimum latency of data aggregation and forwarding is $\left\lfloor \frac{N-1}{S} \right\rfloor + (S-1)$ timeslots, while the minimum energy consumption is expressed by Eq. (13) and the computational complexity is bounded by $O(N^{\frac{N^2}{S}})$.

$$O(E) = \Omega\left(\frac{N^2 \times k}{S} + N \times h + N \times data_{Ack} + S\right) \tag{13}$$

Proof: We assume that each scheduled node is represented by a 1×1 square domino as shown in the Fig. 5. The first 1 specifies that one timeslot is needed for transmitting an aggregated data packet from each node and the second 1 denotes that a RF channel is used in that transmission. Hence, we can illustrate the scheduling problem by placing the dominos of 1×1 for each node in a box of width S. The height of the box is the data aggregation latency for the network. To satisfies the data aggregation and routing constraints, Eq. (8) - (12), each parent domino is placed above all its children node's dominos.

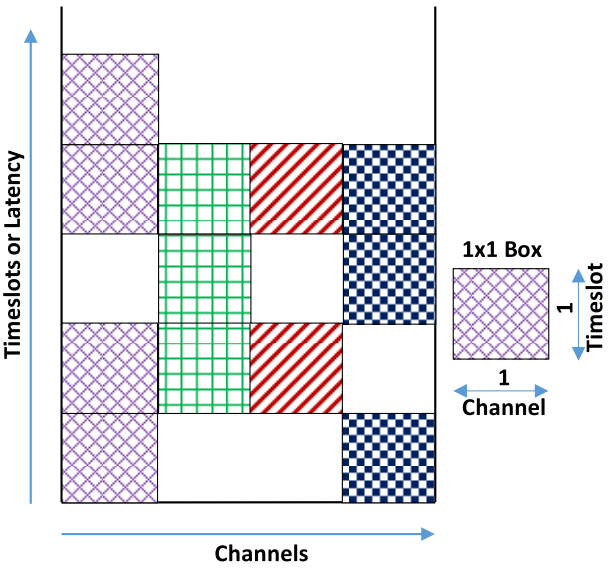

FIGURE 5. Scheduling of the nodes in the box of size S by 1 × 1 domino packing for the minimum latency.

The minimum height can be obtained, when the box is completely packed by the square dominos. Since, in each timeslot, a maximum of S nodes can be scheduled, the network requires S branches with $\left\lfloor \frac{N-1}{S} \right\rfloor$ nodes in each branch (Since sink does not transmit). Due to single radio constraint, the top-layer (one hop to the sink node) dominos should be placed in different heights (timeslots) to transmit the data to the sink node. Hence, the last $S-1$ nodes (without sink node) are placed in $S-1$ different timeslots. Hence, the total minimum latency is $\left\lfloor \frac{N-1}{S} \right\rfloor + (S-1)$ timeslots.

For simplicity without sacrificing the generality, we assume that all nodes generate sensing data of size k, which is aggregated at every hop in each branch S. Leaf nodes in the network has only its own data of size k. Since every node have one child, the aggregated data size for the subsequent node's in the branch increases linearly. Till the aggregated data of size $\left\lfloor \frac{N-1}{S} \right\rfloor$ for the last node. The size of the aggregated data

transmitted in each branch is given by Eq. (14).

$$\mathrm{k} + 2 \times \mathrm{k} + 3 \times \mathrm{k} + \ldots\ldots\ldots\ldots + \left\lfloor \frac{N-1}{S} \right\rfloor \times k \quad (14)$$

Eq. (14) can be summed to Eq. (15).

$$\frac{\left\lfloor \frac{N-1}{S} \right\rfloor \times \left(\left\lfloor \frac{N-1}{S} \right\rfloor + 1 \right)}{2} k \quad (15)$$

The above process is repeated for all S branches, so the total data transmitted in the network is given by Eq. (16).

$$\frac{\left\lfloor \frac{N-1}{S} \right\rfloor \times \left(\left\lfloor \frac{N-1}{S} \right\rfloor + 1 \right)}{2} k \times (S-1) \quad (16)$$

Asymptotic notation of Eq. (16) can be expressed as Eq. (17).

$$O(E) = \Omega\left(\frac{N^2 \times k}{S} \right) \quad (17)$$

Additionally, each packet adds a header of size h, and the data packets are acknowledged by an individual Ack packet of size $\mathrm{data_{Ack}}$. Therefore, the total power consumption is given by Eq. (18).

$$O(E) = \Omega\left(\frac{N^2 \times k}{S} + N \times h + N \times data_{Ack} \right) \quad (18)$$

The dominos in the top layer corresponding to the last hop are placed in S different timeslots. This makes the nodes in the top layer wait in the idle mode. Hence, all S nodes consume extra energy due to the idle timeslots before transmission. Hence the total minimum energy consumption for the network is given by Eq. (19).

$$O(E) = \Omega\left(\frac{N^2 \times k}{S} + N \times h + N \times data_{Ack} + S \right) \quad (19)$$

In each timeslot (indicated by layers in the box of Fig. 5), the ILP algorithm finds the subsets of all the combinations of S nodes or less from the set of N nodes, which satisfy the constraints. The algorithm, then calculates the solution for all possible allocations of timeslot and RF channel. In the end, the algorithm finds the solution for minimum energy consumption by comparing the energy consumption in each solution. The problem of finding minimum energy consumption schedule is similar to the classical N Queen Problem [22]. Hence, the computational complexity of the placement of dominos in a box with minimum energy consumption can be expressed as Eq. (20) and solved through Eq. (21) and (22).

$$\mathrm{T(N)} = \left(\frac{\mathrm{N}}{\mathrm{S}} \right) \times \mathrm{T(N-S)} \quad (20)$$

$$\mathrm{T(N)} = \left(\frac{\mathrm{N}}{\mathrm{S}} \right) \times \left(\frac{\mathrm{N-S}}{\mathrm{S}} \right) \times \ldots \times \left(\frac{1}{1} \right) \quad (21)$$

$$\mathrm{T(N)} = \mathrm{O}\left(\mathrm{N}^{\frac{\mathrm{N}^2}{\mathrm{S}}} \right) \quad (22)$$

The computational complexity of finding a minimum energy scheduling ILP solution is $\mathrm{O}(\mathrm{N}^{\frac{\mathrm{N}^2}{\mathrm{S}}})$.

IV. HEURISTIC ALGORITHMS

The ILP algorithm introduced in Section III.D presents the optimum solution for the directed graph G = (V, E). Due to its excessive computational complexity demonstrated in section III.E for the large networks, the ILP solution is impractical for general applications with large networks. Hence in this section, we propose two suboptimal heuristic algorithms, and analyze their performance.

Inputs to the algorithms are G = (V, E), directed routing graph of the network and S, the maximum number of available channels. The algorithms return a transmission schedule Schedule [T] [S] for each sensor node. Here, each value Schedule [i] [j] represents the node allocated with $\mathrm{i^{th}}$ timeslot and $\mathrm{j^{th}}$ channel.

To collect the sensing data at the sink (root) node from all the nodes in one period, the timeslots of the child nodes should be allocated before the timeslot of their parent node. Hence the algorithms commence by allocating the schedule from the leaf nodes towards the sink node in a bottom-up approach in the network.

A. BACKTRACKING ALGORITHM FOR CHANNEL AND TIMESLOT ALLOCATION (BACAT)

The scheduling algorithm introduced in this subsection explores all the feasible solutions by using backtracking. The algorithm is called Backtracking Algorithm for Channel and Timeslot Allocation (BACAT). It obtains an allocation solution with a minimal energy consumption, that is nearly equivalent to the ILP solution in terms of energy consumption. By employing memoization [23, p. 387], the algorithm prunes out the part of the backtracking tree that apparently does not lead to a least energy consumption solution. Algorithm 1 depicts a pseudo-code of BACAT.

BACAT maintains the least energy solution in Schedule [T] [S] found after recursive function calls till now. Whereas, the current instance of the backtracking tree schedule is stored in $\mathrm{Curr_{Solu[T][S]}}$. For memoization, the algorithm stores the cost metric of the current instance of schedule in $\mathrm{E_{Solu}}$. The least energy cost metric corresponding to Schedule [T] [S] is stored in $\mathrm{E_{min}}$. The cost metric is calculated based on the sum of the transmitting, receiving and idle energy consumption for each node in the network, which is described in the Eq. (23).

$$\mathrm{cost} = \sum_{\forall \mathrm{v} \in (\mathrm{V-S})} \mathrm{E_{Tx_v}} + \mathrm{E_{Rx_v}} + \mathrm{E_{Idle_v}} \quad (23)$$

The values $\mathrm{E_{Tx_v}}$, $\mathrm{E_{Rx_v}}$ and $\mathrm{E_{Idle_v}}$ in the Eq. (23) are the energy consumption of node v in transmission, reception, and idle mode respectively in a period, which are same as Eq. (6) in Section III.D.

In each solution, the algorithm obtains a schedule of timeslot and channel allocation to all the nodes until the algorithm reaches to the sink node. After reaching to the sink node, if the current solution has a smaller cost metric than the previous best solution ($\mathrm{E_{Solu}} < \mathrm{E_{min}}$), the algorithm stores both cost $\mathrm{E_{Solu}}$ and scheduling result $\mathrm{Curr_{Solu[T][S]}}$ into $\mathrm{E_{min}}$

Algorithm 1 Backtracking Algorithm for Channel and Timeslot Allocation (BACAT)

Input and notation

1. Directed Graph $G = (V, E)$
2. $S-$ Maximum number of Channels
3. $E_{min}-$ The minimum energy solution found till now
4. $Schedule[T][S]$ – 2D array to store result, $Schedule[i][j]$, denotes the node's allocation to transmit the aggregated packet at timeslot i by using wireless channel j
5. $E_{Solu}-$ Energy consumed in the current solution
6. $Curr_{Solu[T][S]}-$ Current scheduling solution by E_{Solu} energy consumption
7. $Q_{Ready}-$ Queue of the nodes ready to be scheduled in the next timeslot
8. $Q_{Idle}-$ Queue of the nodes waiting in Idle mode in the current timeslot
9. $timeslot-$ Current timeslot of execution

Output

1. The transmission schedule $Schedule\,[T]\,[S]$ for each node in V

#INITIALIZATION

1. $E_{min} \leftarrow \infty$
2. $E_{Solu} \leftarrow 0$
3. $Q_{Ready} \leftarrow Enqueue(v \in V | v.children == 0)$
4. $Q_{Idle} \leftarrow \emptyset$
5. $timeslot \leftarrow 0$

#RECURSIVE FUNCTION

1. **SCHEDULE$_S$LOT**
2. $(timeslot, S, E_{Solu}, E_{min}, Curr_{Solu}\,[]\,[S],\ Schedule\,[]\,[S], Q_{Ready}, Q_{Idle})$
3. ret-value = false
4. **if** $Q_{Ready} == \emptyset$
5. **if** $E_{Solu} < E_{min}$
6. $E_{min} = E_{Solu}$
7. $Schedule = Curr_{Solu}$
8. ret-value = true
9. **else** ret-value = false
10. **end if**
11. **else if** $E_{Solu} \geq E_{min}$
12. ret-value = false
13. **else**
14. **while** $Curr_{Solu}\,[timeslot] = Select\_Schedule\_node(Q_{Ready}) \neq SINK$
15. **for** $v \leftarrow \forall\ Curr_{Solu}\,[timeslot]$
16. $Q_{Ready} \rightarrow Dequeue(v)$
17. $Q_{Idle} \rightarrow Dequeue(v)$
18. $E_{Solu} += v.E_{Tx} + v.parent.E_{Rx}$
19. $Q_{Idle} \leftarrow Enqueue\,(v.parent)$
20. $Q_{Ready} \leftarrow Enqueue\,(v.parent)$
21. **end for**
22. **for** $u \leftarrow \forall Q_{Idle}$
23. **for** $v \leftarrow \forall\ Curr_{Solu}\,[timeslot]$
24. **if** $v.parent = u$
25. **break**
26. **end if**
27. **end for**
28. $E_{Solu} += E_{Idle}$
29. **end for**
30. ret-value = ret-value $|SCHEDULE\_SLOT\,(timeslot + 1, E_{Solu}, E_{min}, Curr_{Solu}[]\,[S]\,Schedule, []\,[S]\,, Q_{Ready}, Q_{Idle})$
31. **for** $u \leftarrow \forall Q_{Idle}$
32. **for** $v \leftarrow \forall\ Curr_{Solu}\,[timeslot]$
33. **if** $v.parent = u$
34. **break**
35. **end if**
36. **end for**
37. $E_{Solu} -= E_{Idle}$
38. **end for**
39. **for** $v \leftarrow \forall\ Curr_{Solu}\,[timeslot]$
40. $Q_{Ready} \rightarrow Dequeue(v.parent)$
41. $Q_{Idle} \rightarrow Dequeue(v.parent)$
42. $E_{Solu} -= v.E_{Tx} + v.parent.E_{Rx}$
43. $Q_{Idle} \leftarrow Enqueue(v)$
44. $Q_{Ready} \leftarrow Enqueue(v)$
45. **for end**
46. **end while**
47. **return** ret-value
48. **end if**

and Schedule [T] [S], respectively. This signifies that the current solution is guaranteed to provide a lesser cost solution than all the previous solutions explored. If, at the intermediate stage of the solution, the cost metric is higher than the current least energy solution ($E_{Solu} \geq E_{min}$). The algorithm does not proceed with that solution, instead it backtracks to the last timeslot.

BACAT first enqueues all the leaf nodes (v $\in$ V|v.children == 0) into queue Q_{Ready}, which stores the nodes ready to get scheduled with conditions: (1) the leaf nodes (do not have child nodes), (2) the nodes whose child nodes are already scheduled. BACAT algorithm starts by calling the recursive function from the timeslot 0. The input to the recursive function is two queues: Q_{Ready} and Q_{Idle}. Initially Q_{Idle}, starts as an empty queue, since the leaf nodes do not have children, so they wake up immediately in their transmitting timeslot's. In each timeslot, the algorithm considers all possible subsets of nodes satisfying the application constraints (Eq. (8) - (12) in ILP formulation in section III.D). The maximum size of the subset is given by S. For each subset scheduled in the current timeslot, the algorithm recursively executes the allocation of the rest of the unscheduled nodes for the next timeslots. In the end, the algorithm returns the least energy schedule Schedule [T] [S].

For example, in the network of Fig. 3, BACAT starts by enqueuing the leaf nodes V5, V18, V1, V6, V7, V10, V3, V20, V21 and V9 to Q_{Ready} queue for timeslot T0. Q_{Idle}, the queue for idle nodes is empty. In timeslot T0,

BACAT finds all possible subset of nodes that satisfy the maximum channel constraint S = 4. Such subsets are (V3, V9, V6, V7), (V6, V7, V5, V18) etc. For each subset, the algorithm calculates the cost and execute the recursive function SCHEDULE_SLOT for timeslot T1 with the updated parameters. If subset (V3, V9, V6, V7) is chosen for timeslot T0, the algorithm adds parents of scheduled node to the queue Q_{Ready}.

The updated queue Q_{Ready} contains nodes V5, V18, V1, V23, V11, V10, V17, V20, V21 and V22. The parents of the scheduled nodes, V17, V22, V23, V11 are also updated to queue Q_{Idle}. In the timeslot T1, if the new cost is higher than the previous least cost (initial least cost is ∞) value, BACAT backtracks to timeslot T0 and explores other subsets. If, the new cost is lower than the previous optimal cost, BACAT is executed in the timeslot T1, in the same way as in the timeslot T0.The same process is repeated for the rest of the timeslots. Fig. 6 shows the BACAT's scheduling result for the network of Fig. 3 in a form of Grantt chart by using 4 RF channels. Individual mode's energy consumption and length of one timeslot are specified in Table 1. The energy consumption for BACAT (Fig. 6) is 86.25 mJ, the energy consumption for ILP (Fig. 4) is 81.03 mJ. Both algorithms require 10 timeslots and correspondingly results to 330 ms latency.

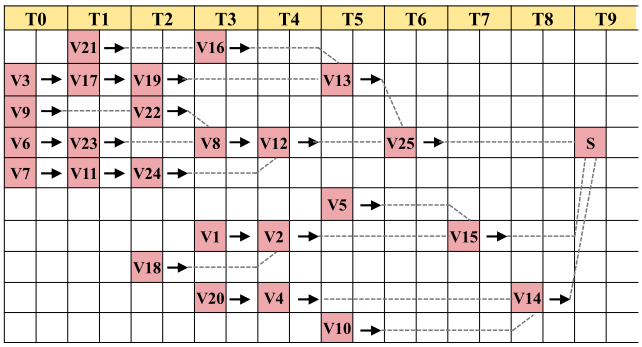

FIGURE 6. Gantt chart of BACAT schedule allocation for the network in Fig. 3.

The computational complexity of BACAT is expressed by the sum of two components: 1) The computation complexity for the current timeslot, and 2) The complexity for all recursive calls. Eq. (24) denotes this computational complexity for BACAT algorithm.

$$T(N) = N^2 + \frac{N}{S} \times T(N - S) \tag{24}$$

By using the recursion tree method [23, p. 88], the recursive expression of Eq. (24) can be expanded to Eq. (25).

$$T(N) = N^2 + \frac{N}{S}(N-S)^2 + \frac{N}{S} \times \frac{N-S}{S}(N-2S)^2 + \ldots + \frac{N}{S} \times \frac{N-S}{S} \times \ldots \times \frac{N-(N-S)}{S} \times S^2 \tag{25}$$

$$T(N) = O(N^{\frac{N}{S}}) \tag{26}$$

The upper bound of the complexity can be found by the substitution method [23, p. 83], which is expressed by Eq. (26).

The Eq. (26) provides the worst-case complexity, where none of the backtracking trees are pruned by BACAT. For medium or large size networks, BACAT prunes out a significant part of the backtracking tree. Even with the pruning, the computational complexity of BACAT still shows a tendency of exponential growth.

B. TIMESLOT-OPTIMAIZATION BASED SCHEDULING ALGORITHM (TOSA)

While the computational complexity of BACAT algorithm is substantially lower than ILP due to BACAT's efficient backtracking and pruning. Its complexity can be prohibitive for very large networks. To further reduce the complexity, this subsection introduces another heuristic algorithm called Timeslot-Optimization based Scheduling Algorithm (TOSA). The algorithm gives polynomial time computational complexity. TOSA generates a Langford subset [24] of maximum size S based on the lexicographical order [25] in each timeslot. The algorithm first computes all the subsets and then checks the application constraints (Eq. (8) - (12) in ILP formulation in section III.D) by using lexicographical order. Among all the lexicographical subsets, a Langford subset is defined as the subset with minimum cost. The cost metric is expressed by Eq. (27).

$$\text{cost} = \sum_{\forall v \in L\_Subset(V)} E_{Tx_v} + E_{Rx_v} + E_{Idle_v} \tag{27}$$

L_Subset in Eq. (27) indicates a lexicographical subset and the other variables are same as Eq. (23).

The pseudo-code of TOSA is described in Algorithm 2. TOSA starts from the leaf nodes ($v \in V | v.children == 0$) by enqueueing them to Q_{Ready} in the same way as BACAT.

TOSA calculates the cost in line 14 in Algorithm 2 for each of the subsets. It selects the Langford subset with minimum cost (E_{min}) schedule in each timeslot. It then continues to choose the schedule for rest of the nodes in the subsequent timeslots but does not revisit the same timeslot. Thus, it is computationally inexpensive, however, its solution is local optimal.

To describe the operation of TOSA, we use the same example network of Fig. 3. For timeslot T0, in the same way as BACAT, TOSA starts with adding leaf nodes V5, V18, V1, V6, V7, V10, V3, V20, V21, and V9 to Q_{Ready}, while setting Q_{Idle} as an empty queue. TOSA then finds the lexicographical subsets such as (6, 5, 7, 18), (5, 18, 20, 10) etc. The algorithm computes the cost for each subset based on Eq. (27), and the lowest cost subset (5, 18, 20, 10), the Langford subset, is allocated to T0. TOSA then adds the parents of allocated nodes to Q_{Idle} and Q_{Ready}. It then proceeds to next timeslots for rest of the nodes in Q_{Ready}. Fig. 7 shows the Grantt chart of TOSA's scheduling for the network in Fig. 3. The result shows 94.25 mJ of energy consumption, which is higher than the ILP's 81.03 mJ and BACAT's 86.25 mJ. While the latency

Algorithm 2 Timeslot-Optimization Scheduling Algorithm (TOSA)

Input and notation

1. Directed Graph $G = (V, E)$
2. S : Maximum number of RF Channels
3. E_{min} : The minimum energy consumed in the current timeslot
4. $Schedule[][S]$: 2D array to store result, $Schedule[i][j]$, denotes the node which transmits packet at timeslot i by using wireless channel j
5. Q_{Ready} : Queue of the nodes ready to be scheduled in the next timeslot
6. Q_{Idle} : Queue of the nodes waiting in Idle mode in the current timeslot

Output

1. The transmission schedule $Schedule[][S]$ for each node in V

Algorithm

7. $Q_{Ready} \leftarrow Enqueue(v \in V | v.children == 0)$
8. $Q_{Idle} \leftarrow \emptyset$
9. $timeslot \leftarrow 0$
10. **while** $Q_{Ready}.dequeue() \neq SINK$
11. $sch\_nodes[S] \leftarrow \emptyset$
12. $E_{min} \leftarrow \infty$
13. **while** $sch\_nodes = select\_schedule\_nodes(Q_{Ready}) \neq \emptyset$
14. $E \leftarrow 0$
15. **for** $v \leftarrow \forall sch\_nodes[S]$
16. $E+ = v.E_{Tx} + v.parent.E_{Rx}$
17. **end for**
18. **for** $u \leftarrow \forall Q_{Idle}$
19. **for** $v \leftarrow \forall sch\_nodes[S]$
20. **if** $u = v || u = v.parent$
21. **Break**
22. **end if**
23. **end for**
24. $E+ = E_{Idle}$
25. **end for**
26. **if** $E_{min} > E$
27. $E_{min} \leftarrow E$
28. $Schedule[timeslot] = sch\_nodes$
29. **end if**
30. **end while**
31. **for** $v \leftarrow \forall sch\_nodes[S]$
32. $Q_{Ready} \rightarrow Dequeue(v)$
33. $Q_{Idle} \rightarrow Dequeue(v)$
34. $Q_{Idle} \rightarrow Enqueue(v.parent)$
35. **end for**
36. $timeslot++$
37. **end while**

of TOSA 297 ms is lower than both the ILP and BACAT's 330 ms in one data gathering period.

The computational complexity T (N) of TOSA can be estimated by the sum of two components: 1) The complexity of Langford subset generation in the current timeslot from N nodes, and 2) The complexity of allocating the rest of N − S nodes in the next timeslots. Hence, T (N) is given by Eq. (28).

$$T(N) = N^2 + T(N - S) \tag{28}$$

$$T(N) = N^2 + (N - S)^2 + (N - 2S)^2 + \ldots + (2S)^2 + S^2 \tag{29}$$

$$T(N) = O\left(\frac{N^2}{S}\right) \tag{30}$$

The Eq. (28) can be expanded to Eq. (29) by using the recursion tree method of [23, p. 88]. The solution of Eq. (29) can be found by the substitution method [23, p. 83], which is expressed by Eq. (30). TOSA's complexity of Eq. (30) increases quadratically with the network size N, whereas BACAT's complexity of Eq. (26) grows exponentially with N, a substantial reduction in the computational complexity.

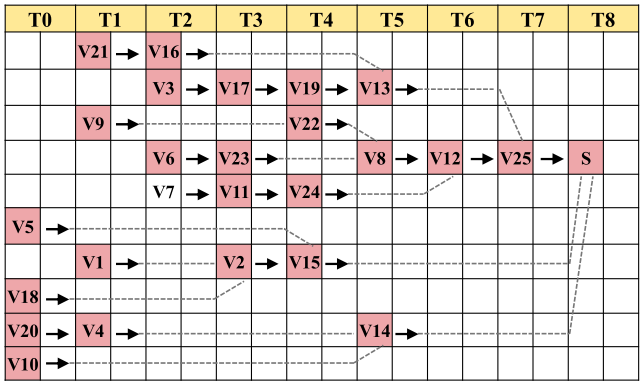

FIGURE 7. Gantt chart of TOSA schedule for the network in Fig. 3.

V. SIMULATION

A. SIMULATION SETUP

To quantify the performance of the proposed algorithms, we consider example networks of various sizes. The nodes in the networks are randomly placed in the sensing area of 1000m × 1000m, with the sink node at the center. Each network is converted to an aggregation tree network by the energy-balanced aggregation routing algorithm [20]. The tree for each network is then provided to the scheduling algorithms. We have implemented the routing algorithm in our C++ simulator called Wiser.

Considering a wireless metering application, in this work, we assume that each sensor node generates sensing data of 4 bytes, followed by a node identifier of 2 bytes (source node's address). Hence, every node generates payload data of $k = 6$ bytes in total to forward to the sink node. In addition to the payload (sensing data), each packet carries a header and a trailer data of 16 bytes.

1) SENSOR NETWORK TESTBED

In order to evaluate the proposed algorithms in the field experiment setup, we implemented a small testbed network of 20 wireless water meter sensors (including sink node), in a single wireless range. Each sensor node comprises state

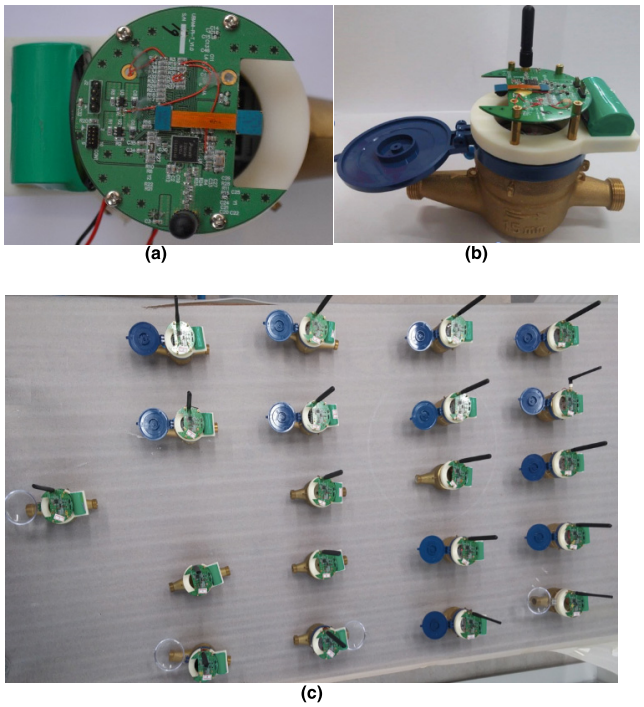

FIGURE 8. Sensor network testbed. (a) Top view of wireless water meter sensor. (b) Front view of the sensor. (c) Testbed network of 20 nodes (including sink).

of art IoT (Internet of Things) chip MKW01Z128 [26], which includes wireless transceiver and MCU. The sensor node and network are shown in Fig.8. For the robust performance, sensors are operated at system clock of 32 MHz, FSK modulation, and a data rate of 76.8 kbps. Table 1 summarizes the operating parameters of the sensor networks. The same parameters are used in our simulation experiments for realistic energy consumption estimation. The network in Fig.8(c) generates the aggregation routing tree using routing algorithm in [20] for scheduling result gathering.

We have implemented our proposed scheduling algorithms: (1) the ILP-based algorithm, (2) BACAT, and (3) TOSA.

2) IMPLEMENTATION OF ILP-BASED ALGORITHM

For each example network, Wiser automatically generates the ILP algorithm of Eq. (7) - (12) from the directed graph of tree network and the number of maximum available channels. The ILP formulas are then solved by CPLEX [27], an integer linear equation solving tool from IBM. We conducted experiments with the ILP-based algorithm for the networks up to 50 nodes for 4, 6 and 8 RF channels since, ILP algorithm demands excessive computation time for the networks of larger than 50 nodes.

3) IMPLEMENTATION OF BACAT AND TOSA ALGORITHMS

Like the ILP based algorithm, the two heuristic algorithms also take tree network and the number of available channels as input. In contrast to the ILP algorithm, the heuristic algorithms do not experience excessive computation time for the large networks. Thus, we simulated the heuristic algorithms with the networks up to 100 nodes. The algorithms are evaluated for three different number of RF channels, 4, 6 and 8 channels.

4) IMPLEMENTATION OF PREVIOUS ALGORITHMS

To compare the performance of the proposed algorithms, we also implemented two previous algorithms LTSA [18], modified CTCCAA [13] and TSDM [14] into the Wiser Simulator. LTSA is the linear-time scheduling algorithm that we had proposed in [18]. It employs a local search algorithm to allocate S nodes in each timeslot from T0 in a way that satisfies the constraints of Eq (8) - (12). Fig. 9 illustrates a scheduling result of LTSA for the example network of Fig. 3 (25 nodes) in a form of Gantt chart. While its linear search process provides high speed, LTSA does not satisfy our minimum energy consumption goal as the simulation results in section V.B.

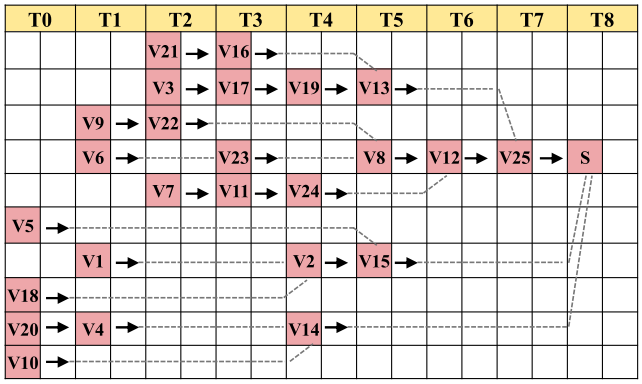

FIGURE 9. Gantt chart of LTSA TDMA schedule for the network in Fig.3.

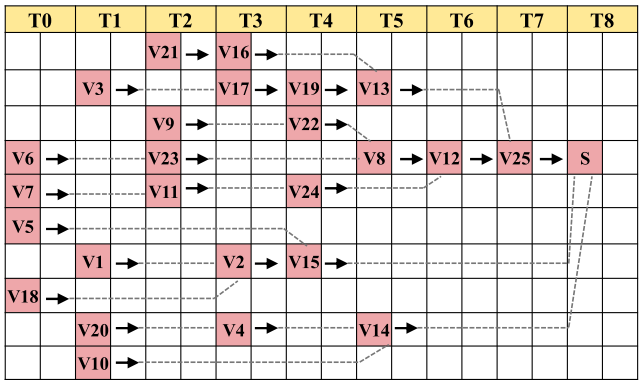

FIGURE 10. Gantt chart of CTCCAA TDMA schedule for the network in Fig. 3.

CTCCAA [13] is a previous algorithm called an in-network aggregation-based collision free data aggregation algorithm. While the original algorithm of CTCCAA used spread spectrum codes, we implemented it using multiple RF channels to conduct a fair comparison of its performance with the proposed algorithms. Hence, we call our implementation a modified CTCCAA. For details of CTCCAA, refer to Section II. Fig. 10 shows a scheduling result of the modified

CTCCAA for the network of Fig. 3. Since CTCCAA targets reducing the latency, it tends to result in higher energy consumption as demonstrated in section V.B. Another previous algorithm, TSDM is an aggregation TDMA algorithm based on data merging, the full details are exhibited in the related work (Section II).

B. SIMULATION RESULTS

This section investigates the performance of the three proposed algorithms namely, ILP, BACAT, and TOSA. These algorithms are compared with the previous algorithms LTSA, modified CTCCAA and TSDM, which are best suited for the comparison since they use TDMA timeslot with data aggregation. We conducted an extensive set of simulations for example networks with 10 to 100 nodes for 4, 6 and 8 channels. Each simulation result is obtained by taking the average of three measurements for each network size from different random networks. Random networks are generated by changing the seed of random number generator in Wiser. The algorithms are evaluated for computational complexity, energy consumption, latency of the network. In the end, we examined the different aggregation algorithms.

1) EVALUATION OF COMPUTATIONAL COMPLEXITY

The computational complexity of the proposed algorithms ILP, BACAT and TOSA are derived in section III.E, section IV.A and section IV.B respectively. In this subsection, the computational complexities of the previous algorithms LTSA, CTCCAA and TSDM are analyzed and compared with the proposed algorithms.

Based on the number of hops to the sink node as cost metric, LTSA assigns the maximum of S nodes in each timeslot. This step has a complexity of O(N). However, LTSA checks the constraints for each candidate node with the other nodes already allocated in the current timeslot, which requires a maximum of S comparisons. Therefore, the total computational complexity for LTSA is $O(N \times S)$.

Unlike the other algorithms, modified CTCCAA starts from allocating schedule to the sink node first. It sorts a queue CurrentList based on the number of child nodes, which leads to the complexity of $O(N \times \log N)$. Then, an RF channel is selected based on the aggregated list of the neighbors and child nodes. The algorithm selects the channel that is least used among all the RF channels. This operation gives complexity of $O(N^2 \times \log N)$. In the end, the algorithm selects the timeslot. Then data forwarding timeslot for each node is calculated by subtracting the allocated timeslot from the maximum allocated timeslot. This step adds additional complexity of $O(N^2)$. Hence, modified CTCCAA has a total complexity of $O\left(N \times \log N + N^2 \times \log N + N^2\right) = O(N^2 \times \log N)$.

The modified TSDM algorithm starts by checking for the leaf nodes, then for the leaf node it allocates two queues otherwise three. This step takes O(N) time. Later, for each N nodes, it allocates timeslot by comparing energy consumption by all possible combination of arranging subtrees (chooses minimum energy merging of data from subtrees). However, it does not revisit the subtree individually. Additionally, it checks the unused channel in the selected timeslot for transmission of non-real time packets, which takes $O\left(N^2 \times S\right)$ time. Hence, the total time complexity is $O\left(N + N^2 \times S\right) = O\left(N^2 \times S\right)$.

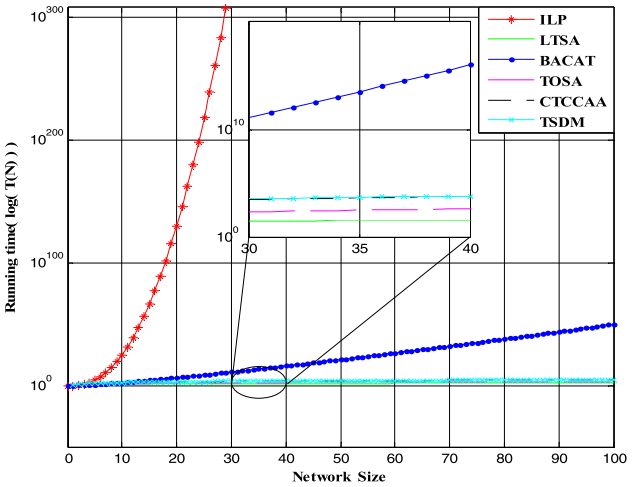

FIGURE 11. Result of the computation complexity of algorithms with respect to the number of nodes in the network with 4 RF channels.

Fig. 11 compares the computational complexity of all the algorithms with respect to the number of nodes, by using 4 RF channels. The computational complexity in Fig. 11 is represented in the logarithmic scale (log(T(N)) because the computational complexity of the ILP-based algorithm escalates rapidly with the network size. It exceeds 10^{300} computations even for a small network of only 28 nodes. In contrast, BACAT performs with substantially lower complexity. It also shows that LTSA has the minimum computation time due to its linear search process. The proposed algorithm TOSA performs faster than CTCCAA and TSDM, unlike ILP and BACAT.

2) EVALUATION OF ENERGY CONSUMPTION

With extensive simulations, we estimated energy consumption for the networks of various size from 10 to 100 nodes. Fig. 12(a), Fig. 12(b), and Fig. 12(c) compares the energy consumption under the constraint of 4, 6, and 8 RF channels, respectively. Node starts with initial energy of E_{Total}. The energy of each network is calculated as the sum of the energy drained by all the nodes (from initial energy) in the transmission, reception, and Idle modes in one data gathering period. Energy consumption metrics for algorithms ILP, BACAT and TOSA are given by Eq. (7), (23) and (27) respectively.

After completing the transmission of the aggregated data packet, each sensor node transitions to the sleep mode for the rest of the time until the next data gathering period. The energy consumed in the sleep mode is negligible and hence ignored.

The results evince that the ILP-based algorithm consumes the least energy. Its energy consumption increases linearly

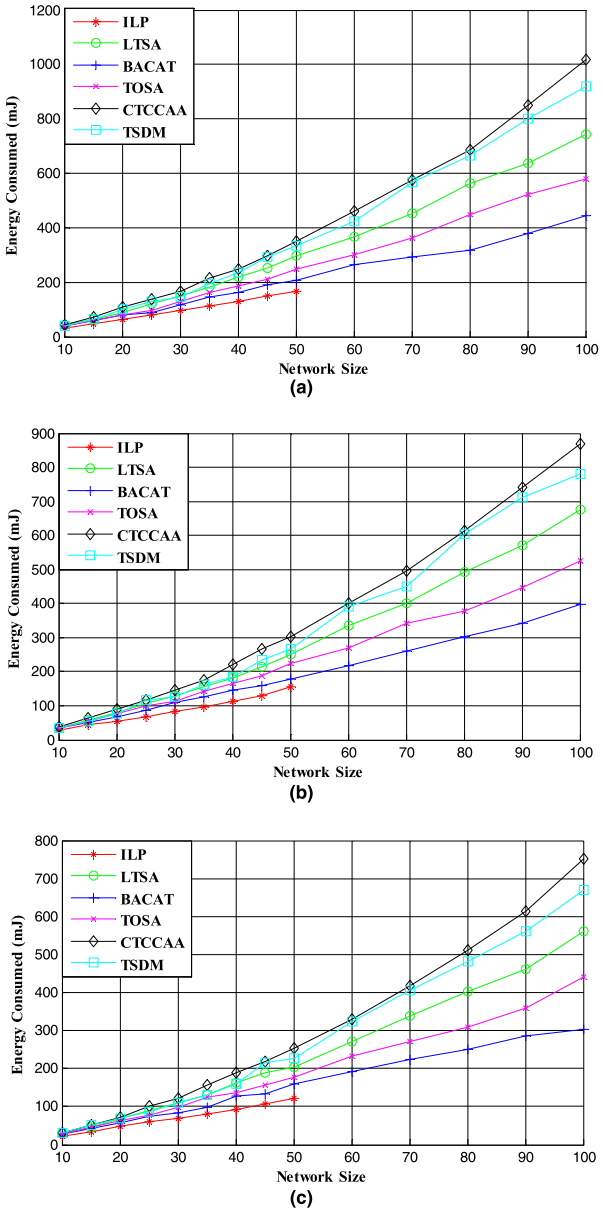

FIGURE 12. Energy consumption with respect to the network size in a data gathering period by using, (a) 4 channels (b) 6 channels (c) 8 channels.

with the network size. On the other hand, in the case of heuristic algorithms, the rate of increase of energy consumption is polynomial in nature. The results also exhibit that the energy consumption of BACAT is the lowest among all the heuristic algorithms and it is nearly commensurate with ILP. Since it determines the most efficient schedule among a large set of possible solutions using efficient backtracking. While the performance of TOSA is comparable with BACAT for the small size networks. BACAT outperforms for the large networks, which results in the energy saving of up to 20% in a data gathering period. The modified CTCCAA and TSDM algorithms consumes notably higher energy than all the other algorithms. The energy consumption of the CTCCAA grows rapidly with the size of the network, which is attributed from the inefficient operation of idle mode. In CTCCAA, even after a node finished its transmission, it waits for the other nodes to finish their transmission before switching to the sleep mode. As the network grows, the energy consumption of idle mode increases more rapidly is a critical drawback of CTCCAA. The TSDM performs better than CTCCAA since, unlike CTCCAA, it does not wait for other nodes to finish. However, it consumes more energy because, it optimizes the timeslot only while merging data from the subset trees. While TOSA and LTSA consume more energy than ILP and BACAT, their energy consumption is lower than TSDM and CTCCAA. Since TOSA optimizes the energy consumption of all the nodes in each timeslot, it consumes lesser energy than LTSA, which only optimizes each node's individual energy while minimizing the number of hops to the sink. Fig. 13 evinces the energy consumption of each algorithm for a network of 50 nodes with 4, 6 and 8 RF channels. The results exhibit that the energy consumption decreases with the increase of the number of RF channels. However, a decrease in the energy consumption is more significant in case of TOSA.

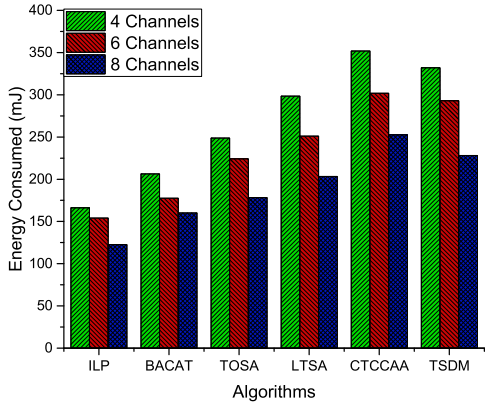

FIGURE 13. Comparison of consumed energy in a network of 50 nodes for each algorithm for a various number of channels.

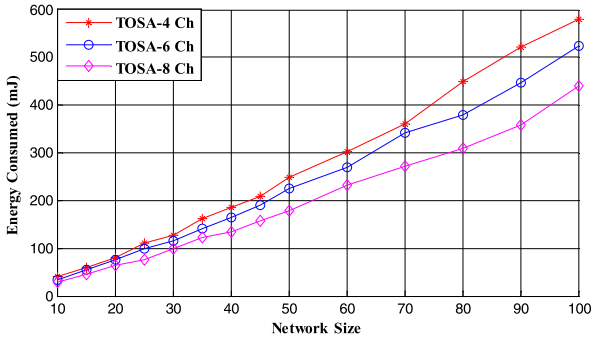

FIGURE 14. Energy consumption with respect to the number of nodes in the network for various bound on channels used for TOSA algorithm.

Hence, Fig. 14 compares the energy consumption of TOSA for 4, 6 and 8 RF channels with respect to the network size. The result of TOSA shows that the energy consumption decreases by minimum of 15% with an increase in the number of channels by 2.

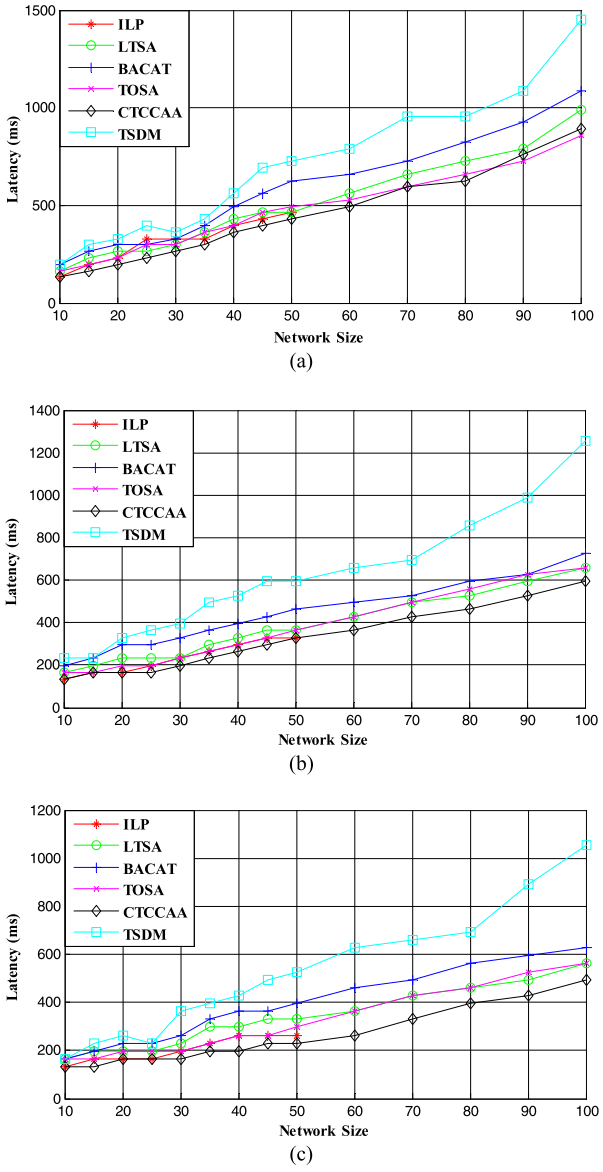

FIGURE 15. Latency with respect to the number of nodes in the network in one data gathering period with (a) 4 channels, (b) 6 channels, (c) 8 channels.

3) EVALUATION OF LATENCY

The latency is defined as the time taken by a network to forward sensing data from all the nodes to the sink node. In other words, it is measured from the time of very first leaf node's transmission to the time of the sink node's last child's transmission. Hence, the latency is the integer multiple of the length of one timeslot which is calculated as 33ms in our experiment. Fig. 15(a), Fig. 15(b) and Fig 15(c) show the latency measurement for the networks of various size with 4, 6 and 8 RF channels, respectively. It can be observed that the modified CTCCAA shows the short latency, since its objective is to optimize the latency. In CTCCAA, the nodes tend to wake up to receive or transmit data as early as possible. However, the nodes often wait longer in the idle mode to aggregate and transmit, which lead to high energy consumption. The latency of BACAT is longer than CTCCAA, since it optimizes the total energy consumption in the network. As a result, it minimizes the wait time in the idle timeslots, and hence gives significant energy saving compared with modified CTCCAA.

However, TSDM algorithm exhibits the maximum latency because if the level 2 packet is already transmitted, the data from the subtree waits for the next slot. Fig. 16 shows the comparison of latency for each algorithm for the network with 50 nodes over 4, 6 and 8 number of RF channels. The result signifies that by increasing the number of RF channels, the algorithms can reduce the total latency.

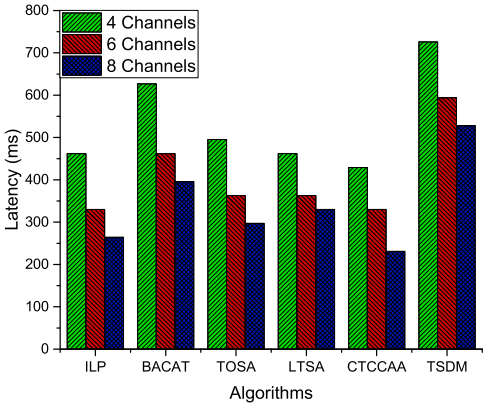

FIGURE 16. Latency for each algorithm for a 50 nodes network for various bound on RF channels used.

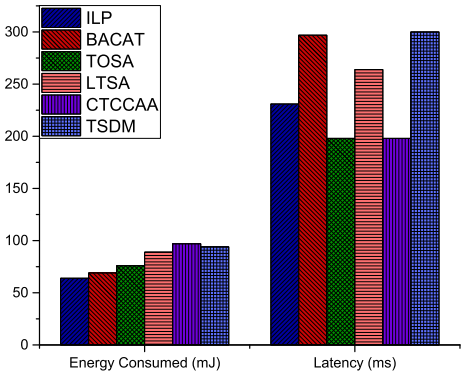

FIGURE 17. Results of 20 nodes testbed network by using 4 channels.

4) EVALUATION OF TESTBED NETWORK

Fig. 17 exhibits and compare the results of energy consumption and latency for testbed network of 20 nodes. The results of testbed network show the similar results as simulation network. ILP consumes the minimum energy, however its latency is higher than TOSA algorithm, which has lower time complexity.

5) EVALUATION OF AGGREGATION ALGORITHMS

Fig. 18 compares the energy consumed by the three data forwarding methods: raw-data forwarding, aggregated forwarding, and our proposed, data-appending forwarding in

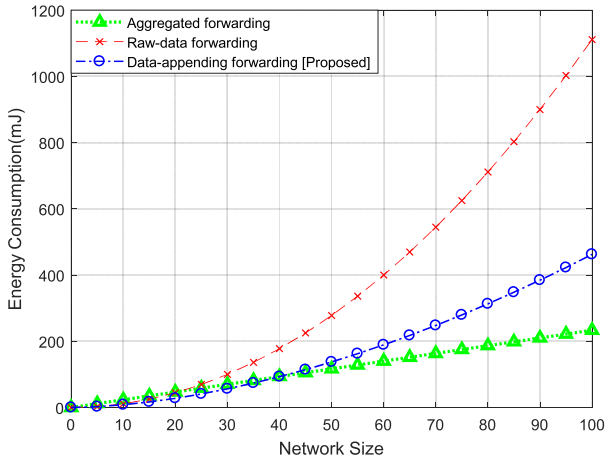

FIGURE 18. Energy consumption for aggregated forwarding, raw-data forwarding and proposed data-appending forwarding in one period of data collection.

one period. It uses the TOSA algorithm with the simulation setup defined in section V.A. AMI applications require each sensor's raw data reading for billing and analysis hence, the aggregated forwarding is not an appropriate solution, although it consumes the lowest energy. On the other hand, the raw-data forwarding consumes the highest energy since it forwards each node's sensing data in an individual packet without aggregation. The proposed, data-appending forwarding algorithm consumes the lesser energy than raw-data forwarding and still transmit the raw data to the sink node. It appends raw data from all the nodes in the subtree of a node in one packet.

VI. CONCLUSION AND FUTURE WORK

In this paper, contention-free TDMA scheduling algorithms for data gathering applications are proposed. The proposed algorithms minimize the energy consumption of the network in each period. To alleviate collisions and support concurrent communications, we utilize multiple RF channels. The paper proposed three algorithms: an optimum ILP algorithm, a near optimal heuristic algorithm BACAT, and computationally efficient heuristic algorithm TOSA, and compared their results. The ILP algorithm produces the most energy efficient schedule at the cost of exponentially increasing computational complexity. Whereas, the heuristic algorithms, BACAT and TOSA, offer computationally efficient scheduling operation, although they provide sub-optimum schedules for data gathering. We implemented the three scheduling algorithms and conducted an extensive set of simulations. Our experiments demonstrated that TOSA provide 21% shorter latency than BACAT, while its computation time is polynomial, which is extremely faster than the exponential computation time of ILP and BACAT algorithms. Therefore, TOSA algorithm results to be the best suited for WSN applications because of the lowest computational complexity at a small cost in energy consumption and latency.

REFERENCES

[1] M. Erol-Kantarci and H. T. Mouftah, "Wireless sensor networks for cost-efficient residential energy management in the smart grid," *IEEE Trans. Smart Grid*, vol. 2, no. 2, pp. 314–325, Jun. 2011.
[2] L. Li, H. Xiaoguang, C. Ke, and H. Ketai, "The applications of WiFi-based wireless sensor network in Internet of Things and smart grid," in *Proc. 6th IEEE Conf. Ind. Electron. Appl.*, Jun. 2011, pp. 789–793.
[3] A. W. DeWeerd, C. J. May, and S. C. Tilka, "System and method for reading and transmitting water meter data utilizing RF signals," U.S. Patent 6 351 223, Feb. 26, 2002.
[4] S. Kurt, H. U. Yildiz, M. Yigit, B. Tavli, and V. C. Gungor, "Packet size optimization in wireless sensor networks for smart grid applications," *IEEE Trans. Ind. Electron.*, vol. 64, no. 3, pp. 2392–2401, Mar. 2017.
[5] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: A top-down survey," *Comput. Netw.*, vol. 67, pp. 104–122, Jul. 2014.
[6] P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, and M. Sichitiu, "Analyzing and modeling encryption overhead for sensor network nodes," in *Proc. 2nd ACM Int. Conf. Wireless Sensor Netw. Appl.*, 2003, pp. 151–159.
[7] J. Zhu, Y. Zou, and B. Zheng, "Physical-layer security and reliability challenges for industrial wireless sensor networks," *IEEE Access*, vol. 5, pp. 5313–5320, 2017.
[8] S. Upadhyayula and S. K. S. Gupta, "Spanning tree based algorithms for low latency and energy efficient data aggregation enhanced convergecast (DAC) in wireless sensor networks," *Ad Hoc Netw.*, vol. 5, no. 5, pp. 626–648, 2007.
[9] Y. Jin, L. Wang, X.-Z. Yang, and D.-X. Wen, "Energy consumption in wireless sensor networks," *J.-Dong HUA Univ.-English Ed.*, vol. 24, no. 5, p. 646, 2007.
[10] 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.
[11] B. Yu, J. Li, and Y. Li, "Distributed data aggregation scheduling in wireless sensor networks," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2159–2167.
[12] A. Erzin and A. Pyatkin, "Convergecast scheduling problem in case of given aggregation tree: The complexity status and some special cases," in *Proc. 10th Int. Symp. IEEE Commun. Syst., Netw. Digit. Signal Process. (CSNDSP)*, Jul. 2016, pp. 1–6.
[13] V. Annamalai, S. K. S. Gupta, and L. Schwiebert, "On tree-based convergecasting in wireless sensor networks," in *Proc. IEEE Wireless Commun. Netw. (WCNC)*, vol. 3, Mar. 2003, pp. 1942–1947.
[14] V. Akila, T. Sheela, and G. AdilineMacriga, "Efficient packet scheduling technique for data merging in wireless sensor networks," *China Commun.*, vol. 14, no. 4, pp. 35–46, 2017.
[15] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-network aggregation techniques for wireless sensor networks: A survey," *IEEE Wireless Commun.*, vol. 14, no. 2, pp. 70–87, Apr. 2007.
[16] S. Kumar, H. Lim, and H. Kim, "Energy optimal scheduling of multi-channel wireless sensor networks for wireless metering," in *Proc. Int. Conf. Electron., Inf., Commun. (ICEIC)*, Jan. 2016, pp. 1–5.
[17] S. Kumar and H. Kim, "Low energy scheduling of minimal active time slots for multi-channel multi-hop convergence wireless sensor networks," in *Proc. Int. Conf. IEEE Comput., Netw. Commun. (ICNC)*, Jan. 2017, pp. 1051–1057.
[18] S. Kumar and H. Kim, "Low complexity energy efficient scheduling for intelligent data gathering wireless sensor networks," in *Proc. 3rd Asia Workshop IT Converg. KIICE (AIWTC)*, 2017.
[19] M. Elsharief, M. A. A. El-Gawad, and H. Kim, "Density table-based synchronization for multi-hop wireless sensor networks," *IEEE Access*, vol. 6, pp. 1940–1953, 2018.
[20] S. Kumar and H. Kim, "Energy efficient routing algorithm for convergence wireless sensor network with data aggregation," in *Proc. IEIE Summer Conf.*, 2016, pp. 1629–1632.
[21] A. Ghosh, O. D. Incel, V. S. A. Kumar, and B. Krishnamachari, "Multi-channel scheduling algorithms for fast aggregated convergecast in sensor networks," in *Proc. IEEE 6th Int. Conf. Mobile Adhoc Sensor Syst. (MASS)*, Oct. 2009, pp. 363–372.
[22] V. Thada and S. Dhaka, "Performance analysis of N-Queen problem using backtracking and genetic algorithm techniques," *Int. J. Comput. Appl.*, vol. 102, no. 7, pp. 26–29, 2014.
[23] T. H. Cormen, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2009.

[24] D. E. Knuth, *The Art of Computer Programming: Combinatorial Algorithms*, vol. 4A. London, U.K.: Pearson Education India, 2011.
[25] S. Zaks, "Lexicographic generation of ordered trees," *Theor. Comput. Sci.*, vol. 10, no. 1, pp. 63–82, 1980.
[26] *MKW01Z128 Data Sheet*. Accessed: Dec. 5, 2017. [Online]. Available: http://www.nxp.com/docs/en/data-sheet/MKW01Z128.pdf
[27] *IBM ILOG CPLEX V12.1: User's Manual for CPLEX*, Int. Bus. Mach. Corp., New York, NY, USA, 2009, p. 157, vol. 46, no. 53.

SAURABH KUMAR received the B.Tech. degree from the National Institute of Technology, Srinagar, India, in 2011. He is currently pursuing the dual M.S. and Ph.D. degrees with the Electronics Engineering Department, Chungbuk National University, South Korea. After receiving the B.Tech. degree, he was a Software Engineer with HCL Technologies. Later, he was with Samsung Research India-Bangalore as a Senior Software Engineer, until 2014, where he has mainly focused on wireless and camera software development. He is associated with the Mixed Signal Integrated Systems Lab as a Research Assistant. His research interests include MAC, network and application protocols for wireless sensor networks, the IoT, and vehicular ad-hoc networks.

HYUNGWON KIM (M'95) received the B.S. and M.S. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology, in 1991 and 1993, respectively, and the Ph.D. degree in electrical engineering and computer science from the University of Michigan, Ann Arbor, MI, USA, in 1999. In 1999, he joined Synopsys Inc., Mountain View, CA, USA, where he developed electronic design automation software. In 2001, he joined Broadcom Corporation, San Jose, CA, USA, where he developed various network chips, including a WiFi gateway router chip, a network processor for 3G, and 10-Gb Ethernet chips. In 2005, he founded Xronet Corporation, a Korea-based wireless chip maker, where, as a CTO and CEO, he managed the company to successfully develop and commercialize wireless baseband and RF chips and software, including WiMAX chips supporting IEEE802.16e and WiFi chips supporting IEEE802.11a/b/g/n. Since 2013, he has been with Chungbuk National University, Cheongju, South Korea, where he is currently an Associate Professor with the Department of Electronics Engineering. His current research interests include sensor read-out circuits, touch screen controller SoC, wireless sensor networks, wireless vehicular communications, mixed-signal SoC designs for low power sensors, and bio-medical sensors.