Peer-to-Peer Networking and Applications (2020) 13:796–815
https://doi.org/10.1007/s12083-019-00818-z

Effective TDMA scheduling for tree-based data collection using genetic algorithm in wireless sensor networks

Walid Osamy[1,4] · Ahmed A. El-Sawy[1] · Ahmed M. Khedr[2,3]

Received: 14 February 2019 / Accepted: 23 September 2019 / Published online: 4 November 2019

Abstract

Data collection is a major operation in Wireless Sensor Networks (WSNs) and minimizing the delay in transmitting the collected data is critical for a lot of applications where specific actions depend on the required deadline, such as event-based mission-critical applications. Scheduling algorithms such as Time Division Multiple Access (TDMA) are extensively used for data delivery with the aim of minimizing the time duration for transporting data to the sink. To minimize the average latency and the average normalized latency in TDMA, we propose a new efficient scheduling algorithm (ETDMA-GA) based on Genetic Algorithm(GA). ETDMA-GA minimizes the latency of communication where two dimensional encoding representations are designed to allocate slots and minimizes the total network latency using a proposed fitness function. The simulation results show that the performance of the proposed algorithm outperforms the existing state-of-the-art approaches such as Rand-LO, Depth-LO, DepthRe-LO, IDegRe-LO, and IDeg-LO in terms of average latency, average normalized latency, and average schedule length.

Keywords Genetic algorithm · Routing tree · Time division multiple access · Two dimensional encoding representation · Tree traversal · Wireless sensor networks

1 Introduction

WSN consists of a huge number of deployed devices in an area of interest to sense various physical information from the surrounding environment. The information collected by these devices are periodically transmitted to the sink and will be processed depending on the application requirements [1–4]. The limitations of communication makes the far away nodes from the sink to send their data to the sink through multi-hop using intermediary nodes over a tree-based routing topology.

To access communication media, two main protocols are commonly used: contention-based and schedule-based. Contention-based protocols such as Carrier-Sense Multiple Access (CSMA) are inefficient and may fail if the communication media are highly utilized or when the number of sensors is large due to high probability of collisions causing re-transmissions. While, in schedule-based or contention-free medium access such as Time Division Multiple Access (TDMA), sensors have to communicate using distinct time slots to avoid collisions. TDMA offers considerable improvements for data collection as it allows sensor nodes to enter into sleep mode during inactive time, which implies: attaining low duty cycles, reducing energy consumption, eliminating conflicts, overhearing, and idle listening which are the major sources of consumption of energy in WSNs [5–7]. Furthermore, TDMA-based communications are robust during peak loads and can supply a demonstrable accomplishment time of collecting data, for example, in timely detection of events [8]. Also, in TDMA when nodes are not in the transmission or reception mode, they fall into sleep mode to reduce energy consumption. In TDMA, when

✉ Walid Osamy
walid.osamy@fci.bu.edu.eg; w.elsherif@qu.edu.sa

Ahmed A. El-Sawy
ahmed.el_sawy@fci.bu.edu.eg

Ahmed M. Khedr
akhedr@sharjah.ac.ae

[1] Computer Science Department, Faculty of Computers and Artificial Intelligence, Benha University, Benha, Egypt

[2] Computer Science Department, University of Sharjah, Sharjah, 27272, UAE

[3] Mathematics Department, Zagazig University, Zagazig, Egypt

[4] Qassim University, Buridah, Saudi Arabia

a node is transmitting, its conflict nodes cannot transmit. The main types of conflicts are: primary conflict and secondary conflict. The principal strife happens when a sensor receives more than one destined transmissions or transmits and receives at the same time. The secondary strife occurs when the intended receiver of a specific transmission is within the range of another transmission proposed for other nodes.

TDMA scheduling protocols are classified as follows: pure MAC layer, hybrid and Cross-layer. In TDMA based MAC, time is split into a set of frames and every frame is split into fixed time slots and then every sensor node is designated to particular time slot in which transmission occurs. Hybrid algorithms integrate TDMA with other protocols, e.g., CSMA, etc. However, Cross-Layer algorithms utilize other layers in order to gain preferable schedules [9].

Minimizing latency in collecting data is essential for critical applications that take definite actions based on deadlines. In spite of the fact that minimizing the schedule length may minimize the latency under certain conditions, most of the existing protocols do not theorize the average latency experienced at each hop by individual packets [8]. Furthermore, some algorithms minimize the length of the schedule by maximizing the re-utilization of time slots, which does not necessarily result in schedules with minimal delay in some topologies. For example, a linear topology may permit higher spatial reuse, but due to larger number of hops from the sources to the sink, it may incur high latency. Therefore, minimizing the data collection latency may enforce more constraints along with minimizing the schedule length.

In the proposed work, WSN is composed of a finite set of sensor nodes and a sink. The sensors send their sensed data to the sink via a multi-hop path over the routing tree. Our contribution can be summarized as follows: Using the cross-layer aspect, we have used routing information tree based WSN. A new GA based approach is proposed that utilizes information from routing tree protocol to define communication time slots and reduces communication latency. Latency is expressed as the fitness function of GA and a two-dimensional chromosome matrix is employed to define TDMA schedule. Evaluation results demonstrate that the performance of the proposed approach exceeds several existing baseline tree based cross-layer approaches. Our main contribution can be summarized in the following points:

1. A new Genetic algorithm (GA) based ordering method is proposed where two-dimensional encoding representations are designed to allocate slots and minimizes the total network latency by using average time latency as the fitness function.
2. Extensive simulation results to prove the performance of the proposed technique are given. The comparison results with substantial approaches such as Rand-LO, Depth-LO, DepthRe-LO, IDegRe-LO and IDeg-LO using different routing trees and different locations of the sink show the following differences between our proposed ETDMA-GA and the existing algorithms.
   - ETDMA-GA has minimum average latency, minimum average schedule length and minimum normalized latency compared with the existing five substantial approaches in cases of DFSA tree, Min hop tree, even for three different sink locations
   - The duty cycle of ETDMA-GA has diminutive increase compared with the existing algorithms, while ETDMA-GA has remarkable gain in the average latency reduction.

The rest of this paper is tabulated as follows: the related research of the proposed approach is described in Section 2. In Section 3, the description of the proposed problem is given. In Section 4, we present the details of our proposed approach. The simulation results including the comparison with the existing baseline approaches are presented in Section 5. The work is concluded in Section 6,

2 Related research

WSN protocol stack typically contains the following layers: application, transport, network, data link, and physical, with each layer having its own responsibility. For example, the network layer has the responsibility to route the data supplied by the transport layer, and the transport layer assists in preserving the data flow. Data link layer responsibilities are multiplexing of data streams, data frame detection, medium access control (MAC) and error control [10]. The main function of the data link layer is MAC, where the main objective is to equitably and efficiently share the shared communication resources among all sensor nodes in order to accomplish enhanced performance of the network in terms of energy consumption, network throughput, and delivery latency.

The layered model works on the principle of separation which is neither effective nor guarantees optimal performance for WSNs as other features must also be taken into account, such as limited resources, e.g., energy, memory, radio interference or mobility and topology changes. These characteristics generate additional constraints while designing protocols in the layered model for an adaptive WSN. Therefore, cross-layer protocols are preferable as the dependencies between protocols from different layers can be exploited to overcome the limitations of each protocol and achieve an overall gain in performance [11].

Cross-layer protocols that allow flexibility between MAC sublayer and Network layer can be divided into three classes [12]: (a) Joining the functionality of MAC protocol and routing protocol (Joint MAC & Routing) (b) Routing protocols that use MAC protocol information (MAC-AWARE routing), and (c) MAC protocols that use routing protocol information (Routing-AWARE MAC). The block diagram of cross-layer approaches, along with the location of our proposed approach among these categories is given in Fig. 1.

In Joint MAC and Routing, the information and functionality of the Network and Data Link layers are grouped together and combined into a single layer. Address-Light Integrated MAC and Routing Protocol (AIMRP) [13], RSSI-Based Forwarding (RBF) [14], State-free Implicit Forwarding (SIf) [15] and IDeg-Routing & MAC [12] are examples of JOINT MAC & ROUTING. AIMRP, RBF and SIf are very similar and use the same principle by arranging the sensor nodes around the sink. The routing of data to the sink is also done by the same way, even if the names given to packages and messages are different. On the other hand, IDeg-Routing & MAC [12] is an approach that combines a contention-free MAC protocol with routing. IDeg-Routing & MAC is based on BFS traversal of the graph network. The contention-free schedule for the communications between sensor nodes and the geographic routing tree is obtained from IDeg-Routing & MAC.

In the MAC-AWARE ROUTING, the MAC protocol sends information time to the routing protocol and then the routing protocol finds the next hop for each sensor node such as ORW [16] (Opportunistic Routing in WSNs) and MAR-WSN (MAC-aware routing in WSNs) [17]. In the ROUTING-AWARE MAC category, the data link layer and more particularly the protocol MAC waits for routing protocol information to establish an order of communications for the sensor nodes. In this paper, we are concerned with the ROUTING-AWARE MAC protocol in a contention-free environment.

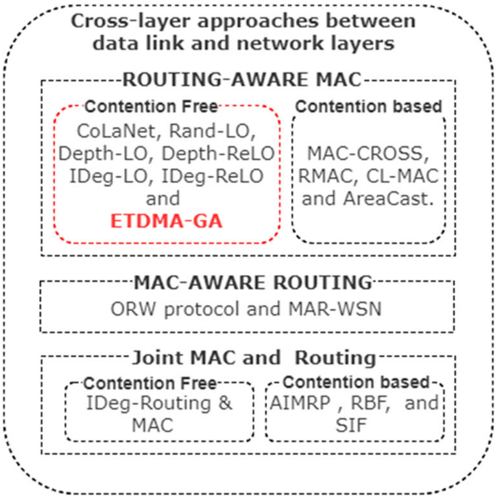


Fig. 1 Cross-Layer approaches between data link and network layers

As with conventional MAC protocols, MAC cross-layer approaches can have two forms: contention-based approaches and contention-free approaches. MAC-CROSS [18], RMAC [19], CL-MAC [20] and AreaCast [21] are examples of contention–based approaches. While CoLaNet [22], Rand-LO, Depth-LO, DepthRe-LO [23], IDeg-LO, and IDegRe-LO [24] are examples of contention-free approaches.

The main objective of MAC-CROSS [18] is to minimize the consumption of energy by decreasing the number of sensor nodes that need to remain awake to participate in the data transmission, where control packets: Request To Send (RTS) and Clear To Send (CTS) are slightly modified by adding a new field to each packet. RMAC (Routing enhanced MAC protocol) [19] is also a cross-layer MAC protocol with contention. It reduces end-to-end communication latency using a single control packet called PION that plays the role of RTS and CTS at the same time. Cross-Layer MAC protocol (CL-MAC) proposed in [20] efficiently manages multi-stream traffic and reduces communication latency. The communication in CL-MAC is performed using Flow Setup Packet (FSP) package which contains the transmitter address, the final destination address and the addresses of next hops. FSP serves as RTS message for the receiver node and at the same time as CTS message for the transmitter node. It also has the particularity to address several recipients at the same time. In [21], AreaCast is presented to improve the robustness of routing protocols and the rate of delivery of packages, while ensuring less energy consumption.

Unlike MAC-CROSS [18], RMAC [19], CL-MAC [20] and AreaCast [21] almost operate by the same way (based on RTS/CTS message exchange). CoLaNet [22], is a centralized contention-free approach that aims to save energy from sensors by building a routing tree called MinDegree; then, it uses a coloring algorithm on this tree to establish a TDMA for each sensor node. CoLaNet is a TDMA without contention and has two limitations: the sensor node with the largest number of neighbors in MinDegree routing tree can have any position in the tree and all neighbors of an already colored knot do not reflect the direction of communications. Moreover, CoLaNet is not oriented to optimize the communication latency.

Based on cross-layer decisions, communication latency is reduced [23] by the three proposed different slot allocation methods: Rand-LO, Depth-LO and DepthRe-LO. Rand-LO randomly schedules the leaves of the routing tree, then it schedules the others while ascending the tree. Depth-LO improves the Rand-LO by giving privilege to the faraway leaves from the sink to be scheduled earlier. In DepthRe-LO, scheduling priority is given to the sensors on the long paths in the routing tree.

In [24], the authors proposed two TDMA scheduling techniques called IDeg-LO and IDegRe-LO which use routing tree information and interference degree of each sensor. In IDeg-LO, according to interference degree, the leaf nodes and then internal nodes are scheduled in descending order. Each sensor is scheduled as close as to its children slots. In IDegRe-LO, based on the interference degree, the scheduled leaf is removed from the routing tree and the process is continued with other sensor nodes, which gives a chance to internal sensor nodes with greater interference degree to be scheduled earlier.

As in [24] and [23], our target is to determine the efficient and best design for TDMA schedule, correlated with information from routing protocol, to allocate a slot for every sensor node. However, the proposed ETDMA-GA is based on GA to efficiently schedule all sensor nodes and reduce communication latency for tree-based WSNs. ETDMA-GA uses a heuristic search, that randomly enumerates some possible solutions and generates other solutions which increases the search space for finding the best schedule in terms of reduced latency better than Rand-LO, Depth-LO, DepthRe-LO, IDeg-LO, and IDegRe-LO algorithms which use the traditional blind search methods to generate only one solution and improve it to get the final solution, i.e., the generated solution is always based on the starting point whether the start point is good or bad. However, ETDMA-GA can start from many points and evaluate every solution to select the best candidates. ETDMA-GA repeats these steps until it finds the best solution.

Other approaches are based on metaheuristics techniques such as [25–31]. In [26], a hybrid algorithm called HPSO is proposed that combines Particle Swarm Optimization (PSO) with simulated annealing (SA). HPSO finds the global optimal slot allocation by using the benefit of SA and PSO. A multi-objective optimization framework for TDMA scheduling is proposed in [27], where PSO is embedded with GA and it is used to enhance the searching ability. The PSO part avoids empty slots during scheduling, while the GA part ensures a powerful search ability to determine the optimal slot allocation. PSO-based scheduling algorithm called PSOSA which benefits from the search capability of PSO is proposed in [28]. In [25] a multi-objective immune algorithm is introduced for finding an optimal solution to TDMA scheduling problems. A hybrid algorithm is utilized in [29] by comparing both the evolutionary algorithm (EA) and PSO. In [30], the authors proposed two scheduling algorithms: level based and node-based. In node-based, the sensor nodes of the color corresponding to each slot with at least one packet are selected first and more sensor nodes are included afterwards. However, in level-based scheduling, the original network is first transformed to a linear network and the schedule of the original network is then obtained based on the coloring of the linear network.

GAs have been utilized to solve many problems in WSN. The most common utilization of GAs in WSN is to attain an energy-aware network under specific routing conditions as in [32, 33], improve localization accuracy as in [34] and deploy nodes to provide k-coverage and m-connectivity as in [35]. All previous works [25–31] are based on metaheuristics techniques and the routing information is not considered during the generation of TDMA scheduling, i.e., TDMA scheduling is generated without using any information from the routing protocol. Therefore, their schedules could be deficient in terms of latency reduction because the path is not selected by the routing protocol [23]. Moreover, algorithms such as CoLaNet do not optimize communication latency. In contention-free MAC approaches, TDMA communication latency is affected by non-correlation spatial decisions made in the routing protocols. In addition, TDMA is important because it allows for an explicit estimate of the end-to-end routing time by accurately calculating the delay of a packet sent by a node and transferred by its parent into the routing tree.

3 TDMA scheduling problem description

In our proposed approach, WSN is represented as a graph $G = (V, E)$, where V is the set of deployed sensor nodes and E is the set of communication links between nodes. There exist an edge between two nodes i and j, if i is within the communication range of j. Let sensor node $S_{root} \in V$ is the root such that all data traffic from every sensor node $i \in V$ is routed through S_{root} using routing path. We assume that all sensor nodes have the same communication range, i.e., the links between nodes are symmetric. Moreover, each node cannot transmit and receive simultaneously and cannot hear two messages at the same time (half-duplex radio transceiver). Figure 2 shows an example of a WSN graph and its routing tree rooted at node 1.

In TDMA scheduling, the time frame is divided into equal intervals called time slots. Each time slot is intended to receive or transmit a single packet from/to a neighboring sensor node. The main objective of TDMA scheduling is to assign different transmission and reception time slots to sensor nodes without collisions while using minimum energy consumption and minimum total number of slots

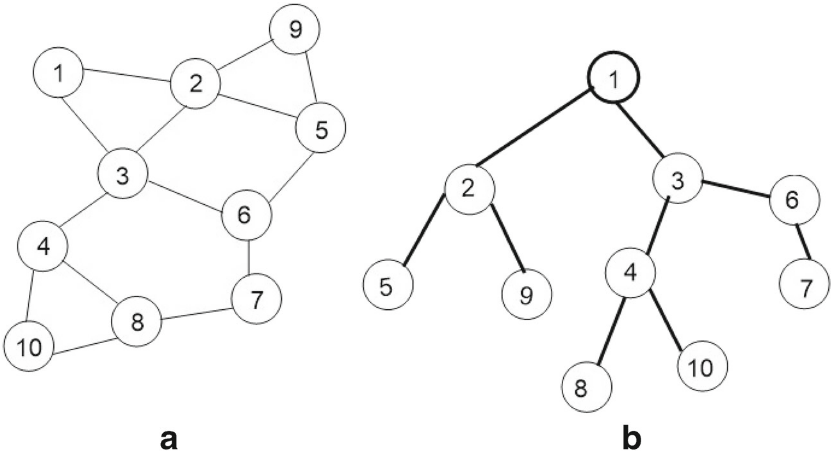


Fig. 2 Example: A WSN graph with its minimum hop routing tree

for transmission and reception. According to the work in [36], TDMA scheduling constraints can be summarized as follows:

1. Transmission and reception of a sensor node at the same time slot are not allowed.
2. Simultaneous reception between several 1-hop neighbors are prohibited.

Moreover, the immediate neighborhood of a sensor node s and its 2-hop neighborhood sensor nodes should guarantee that no interference occurs during the transmissions by a sensor node.

A WSN of N sensor nodes can be represented by a symmetric connectivity matrix $C_{N\times N}$, which is defined as $C[i][j] = 1$, if $e_{ij} = 1$; else $C[i][j] = 0$, where e_{ij} is the minimum number of edges between sensor node i and sensor node j and $i \neq j$. In addition, the conflict relationship in the network can be represented by an interference matrix $I_{N\times N}$, where if $e_{ij} \leq 2$, $I[i][j] = 1$; else $I[i][j] = 0$, where $i \neq j$. As a result, sensor nodes i and j can transmit data at the same time if e_{ij} is larger than 2. For example the corresponding connection matrix and interference matrix for the network in Fig. 2 is given below in Eqs. 1 and 2:

$$C[10][10] = \begin{matrix} 0&1&1&0&0&0&0&0&0&0 \\ 1&0&1&0&1&0&0&0&1&0 \\ 1&1&0&1&0&1&0&0&0&0 \\ 0&0&1&0&0&0&0&1&0&1 \\ 0&1&0&0&0&1&0&0&1&0 \\ 0&0&1&0&1&0&1&0&0&0 \\ 0&0&0&0&0&1&0&1&0&0 \\ 0&0&0&1&0&0&1&0&0&1 \\ 0&1&0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&1&0&0 \end{matrix} \tag{1}$$

$$I[10][10] = \begin{matrix} 0&1&1&1&1&1&0&0&1&0 \\ 1&0&1&1&1&1&0&0&1&0 \\ 1&1&0&1&1&1&1&1&1&1 \\ 1&1&1&0&0&1&1&1&0&1 \\ 1&1&1&0&0&1&1&0&1&0 \\ 1&1&1&1&1&0&1&1&1&0 \\ 0&0&1&1&1&1&0&1&0&1 \\ 0&0&1&1&0&1&1&0&0&1 \\ 1&1&1&0&1&1&0&0&0&0 \\ 0&0&1&1&0&0&1&1&0&0 \end{matrix} \tag{2}$$

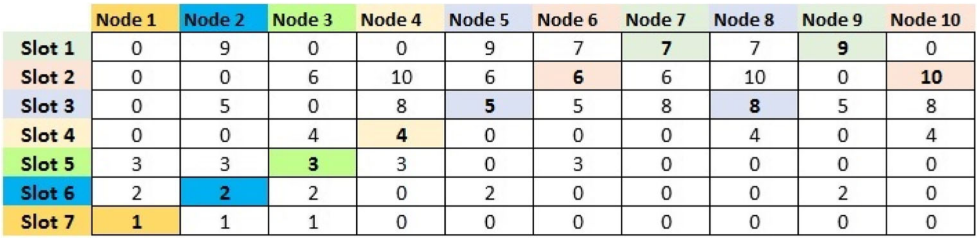
The TDMA scheduling problem in tree-based WSN can be formulated as follows: Sensor nodes generate data packets and they need to transfer these packets to the root node along the routing tree. The main objective is to schedule a time for each node, i.e., to allocate a slot for every sensor node without collision during transmission, minimize data delivery time and the total number of slots and maximize the network lifetime.

Optimal scheduling can be obtained by enumerating all valid schedules. But the complexity of such a solution will be exponential and hence heuristic solution within a reasonable time is desirable [24]. In this paper, GA is used as a heuristic approach to get the best schedule for sensor nodes in a WSN.

4 TDMA scheduling based on genetic algorithm

This section describes our proposed technique for scheduling the sensor nodes in a WSN. The algorithm is divided into two phases: initialization and TDMA schedule construction. In the initialization phase, the sink generates

Fig. 3 TDMA scheduling example for the WSN in Fig. 2



	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8	Node 9	Node 10
Slot 1	0	9	0	0	9	7	**7**	7	**9**	0
Slot 2	0	0	6	10	6	**6**	6	10	0	**10**
Slot 3	0	5	0	8	**5**	5	8	**8**	5	8
Slot 4	0	0	4	**4**	0	0	0	4	0	4
Slot 5	3	3	**3**	3	0	3	0	0	0	0
Slot 6	2	**2**	2	0	2	0	0	0	2	0
Slot 7	**1**	1	1	0	0	0	0	0	0	0

the routing tree using sensor nodes information such as Min Hop tree [37] or Depth First tree [38]. In TDMA schedule construction phase, the sink constructs the routing tree traversal, calculates the TDMA schedule and broadcasts the routes and the TDMA to all sensor nodes in the WSN. The used definitions and notations in the algorithm are provided first, and then we provide full description of each phase.

Definition 1 (Collision between nodes j and **i**) The message of sensor node j collides with the message of sensor node i if the value of $I[i][j]$ is 1, which means that sensor nodes i and j cannot transmit data at the same time. For example, in Eq. 2, I[1][2]=1, i.e., sensor nodes 1 and 2 cannot transmit at the same time.

Definition 2 (Free slot for node i) A slot x is said to be free for node i, if its allocation won't create a collision with node j, i.e., $I[i][j] \neq 1$. For example, in Fig. 3, slot 2 is assigned to node 6 and it is free for node 10, because $I[6][10] \neq 1$ (see Eq. 2).

Definition 3 (Traversal List (TL)) TL list contains the traversal order of the sensor nodes in the routing tree. For example, TL_1=7, 6, 10, 8, 4, 3, 9, 5, 2, 1 is the traversal for the tree in Fig. 2.

Definition 4 (Transmit/Receive slot) Schedule $Slot[i][j] = k$ means that slot i is allocated for node j and it can be used for transmitting if $j = k$, for receiving if $j \neq k$, and otherwise if $Slot[i][j] = 0$, i.e., $k = 0$, node j cannot communicate in this slot. For example, in Fig. 3, node 5 can transmit using slot 3 and nodes 2 and 6 can receive using slot 3.

In the next section, the description of each phase of the proposed algorithm is given.

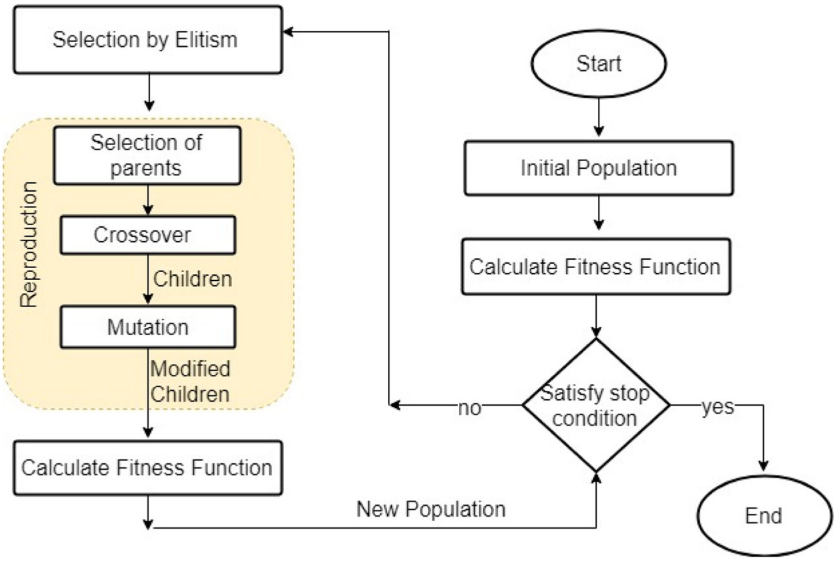
4.1 Initialization phase

In the initialization phase, using the information of sensor nodes, the sink generates the routing tree such as Min Hop tree [37] or Depth First tree [38]. Then, the sink constructs the traversal of the routing tree and calculates the TDMA schedule and broadcasts the routes and the TDMA in the WSN.

4.2 TDMA schedule construction phase

Using GA, a new scheduling algorithm called ETDMA-GA is proposed. ETDMA-GA finds the best routing tree traversal ordering and the corresponding TDMA schedule. The flow chart of GA-Schedule process is shown in Fig. 4, where the population of candidate solutions to an

Fig. 4 Flowchart of GA



optimization problem is evolved towards better solution. Each candidate solution is based on a set of properties that can be mutated and altered; traditionally, solutions are represented in binary as strings of *0*s and *1*s. But other encodings are also possible.

GA is an iterative process and the population in each iteration is called a generation. It starts from randomly generated individuals as *initial population* and in every generation, the fitness will be estimated for each individual. Fitness is usually the value of the objective function when the problem is being optimized. Fit individuals will be selected from the ongoing population, and each individual's genome is modified to create a new generation. The new generation of candidate solutions is used in the next iteration. The algorithm terminates when either a maximum number of generations has been generated or a satisfactory fitness level has been reached. Modeling GA needs Genetic representation of the solution and Fitness function to evaluate the generated solution. In the next subsections, the adaptation of GA stages to schedule slots for sensor nodes are presented.

4.2.1 Representation of TDMA slots using GA

In a given GA, integer representation is used to encode the chromosome. Each gene in the chromosome is mapped into a time slot vector and the gene's value indicates a single slot number of sensor nodes that are included to form a TDMA schedule. Each chromosome can be represented by a two-dimensional matrix with R rows where R is the number of slots. The number of slots is computed as the maximum interference degree ($IDegree$) in the network, where $IDegree = \arg\ \max_{1\leq i\leq n}(\sum_{j=1}^{n} I[i][j])$ and W columns (the number of sensor nodes).

4.2.2 Initial population

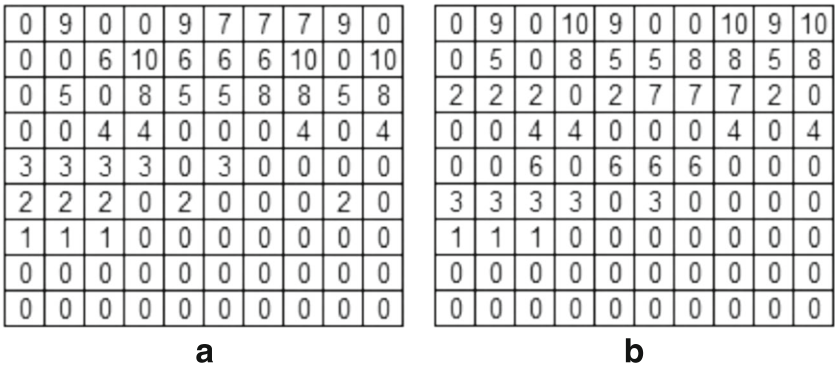
The first step in the GA process is the generation of an initial population. The initial population will be randomly generated as a predefined number of chromosomes in the population. Each chromosome is generated by executing the following slot allocation algorithm (see Algorithm 1):

1. Generate random routing tree traversal list (TL) ordering.
2. For the first node j in TL, allocate the first slot in the scheduling to j ($Slots[i][j] = j, i = 1$).
3. For each node k that has connection with j, i.e., $C[j][k] = 1$, allocate the current slot to k, i.e., $Slots[1][k] = j$ as a receiving slot.
4. For each next node l in TL do the following:
    - Starting from the first slot of the scheduling, search for a free slot (find the slot that does not have a collision node with l).
    - If no free slot is found, then allocate a new slot for l at the end of the scheduling ($Slots[i + 1][l] = l$).
    - For each node k that has connection with l, i.e., $C[l][k] = 1$, allocate the current slot for k, i.e., $Slots[i + 1][k] = l$ as a receiving slot for k.
5. Repeat Step 4 till all nodes in TL are scheduled.

Note that the generated scheduling matrix $Slots[m][n]$ with m slots and n sensor nodes are considered as the chromosome and if m is less than the chromosome length, zeros are added to the end of the chromosome matrix (padding slot).

For clarification, consider a randomly generated TL lists TL_1={7, 6, 10, 8, 4, 3, 9, 5, 2, 1} and TL_2={10, 9, 8, 7, 5, 2, 4, 6, 3, 1}, respectively for the networks in Fig. 2.

Considering TL_1, the algorithm starts by generating a new transmitting slot for node 7, i.e., $Slots[1][7] = 7$. Then, we find the list of nodes that are in connection with node 7 from matrix C in Eq. 1 which is $Clist = 8, 6$ (row 7, the column that has 1 value). Assign the current slot as a receiving slot for nodes 8 and 6 for node 7, i.e., $Slots[1][6] = 7$ and $Slots[1][8] = 7$. After the first sensor node in TL_1 is allocated in the schedule, the next node (node 6) in TL_1 is allocated in the schedule as follows: search Slots matrix for free slot for node 6. Slot 1 in the



0	9	0	0	9	7	7	7	9	0
0	0	6	10	6	6	6	10	0	10
0	5	0	8	5	5	8	8	5	8
0	0	4	4	0	0	0	4	0	4
3	3	3	3	0	3	0	0	0	0
2	2	2	0	2	0	0	0	2	0
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

a

0	9	0	10	9	0	0	10	9	10
0	5	0	8	5	5	8	8	5	8
2	2	2	0	2	7	7	7	2	0
0	0	4	4	0	0	0	4	0	4
0	0	6	0	6	6	6	0	0	0
3	3	3	3	0	3	0	0	0	0
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

b

Fig. 5 Generated chromosomes: Two different TDMA schedules for the WSN example in Fig. 2 **a** for TL_1 **b** for TL_2

Slots matrix is transmitting slot for sensor node 7 and since sensor node 7 is a collision node with sensor node 6; we cannot allocate slot 1 for node 6 (since from I matrix in Eq. 2, $I[6][7] = 1$). Therefore, the new slot will be allocated for node 6 (Slot 2), i.e., $Slots[2][6] = 6$. As before, the list of nodes that have connection with node 6 will be $Clist = 3, 5, 7$ and so Slot 2 will be allocated as receiving slot for nodes 3, 5, and 7. The same processes are repeated for nodes 10, 8, 4, 3, 9, 5, 2 and 1. The generated slots matrix is assigned to chromosome and padding slots are added to the chromosome matrix if the generated matrix length is less than the chromosome length. Figure 5a shows the chromosome obtained for TL_1 and Fig. 5b shows the chromosome obtained for TL_2. Note that slots 8 and 9 are padding slots (padding slots are removed before latency calculations).

4.2.3 Selection

The chromosomes with good fitness rates have higher opportunity to be chosen in the next population. The selection process decides which chromosomes of the current population will be transferred to the next population without change. To differentiate proper individuals, the selection method is adapted based on the principle of elitism where the best chromosomes (a few) in the current population are copied as they will be included in the next population, i.e., the individual with the highest fitness succeeds and will be added in the next population. Such individuals can otherwise be lost if they are not chosen to reproduce; or the crossover or mutation may destroy them. This elitist strategy of selection remarkably enhances the performance of the GA [39].

4.2.4 Crossover

Crossover is a process to generate the next population from the previous one. This process takes more than one parent from the current population to create a new child for the next population. For the two-dimensional representation, the crossover operation must be modified [40]. The crossover operation could be horizontal or vertical. In this work, the selection method is adapted based on the principle of Tournament [41], where k is the number of individuals randomly chosen from the population to form a group, then the best individual from this group is selected for further processing. The selection process from the population will be repeated until k groups are obtained and the best individual will be selected from each group. Finally, the individual with the highest fitness succeeds and will be added in the next generation.

Algorithm 1 SlotAllocation(TL).

```
Input: Traversal List (TL).
Ouput: $Slots$ allocation matrix (TDMA schedule).
TLindex = 1.
slotIndex=1.
nodeId = TL[TLindex];      ▷ assign the first slot for the first item in TL.
Slots[slotIndex][nodeId]= nodeId.
Clist = {$k$ : $k$ is a neighbor node for nodeId.$\forall 1 \le k \le n, k \neq nodeId$}
for each item in Clist do
    Slots[slotIndex][item] = nodeId.
end for
slotIndex=slotIndex+1, TLindex=TLindex+1.
while TLindex≤TL.Count do
    nodeId = TL[TLindex].
    Clist = {$k$ : $k$ is a neighbor node for nodeId, $\forall 1 \le k \le n, k \neq nodeId$}.
    freeSlotIndex=-1.
                                               ▷ Search For Free Slot
    for index = 1 and index < slots.Count do
        if $\forall$ $node$ $j$ and $Slots[index][j]$ is a transmission slot for $j$ and $j$ is not a collision node for node $i$ then
            freeSlotIndex= index               ▷ found Slot.
        else
            index++.
        end if
    end for
                                               ▷ Free Slot is not found
    if freeSlotIndex=-1 then
        Slots[slotIndex][nodeId] = nodeId.
        for each item in Clist do
            slot[slotIndex][item] = nodeId.
        end for
        slotIndex=slotIndex+1.
    else
                                               ▷ Free Slot is found
        Slots[freeSlotIndex][nodeId] = nodeId.
        for each item in Clist do
            Slots[freeSlotIndex][item] = nodeId.
        end for
    end if
    TLindex=TLindex+1.
end while
```

After parents selection, to generate the two children chromosomes g_{child1} and g_{child2} from the selected two parents $g_{parent1}$ and $g_{parent2}$; the crossover is performed horizontally over the rows of the parent chromosomes by generating a random integer r in range $[1, R]$ where, R is the chromosome length. Then the following steps are executed:

- If row $i \le r$ $(1 \le i \le R)$ then copy from gene 1 to gene i, i.e., from $g_{parent1}$ to g_{child1} and from $g_{parent2}$ to g_{child2}
- Otherwise, if row $i > r$ $(1 \le i \le R)$, then copy from gene i to gene R, i.e., from $g_{parent2}$ to g_{child1} and from $g_{parent1}$ to g_{child2}

Figure 6 shows the generated children $Child1$, $Child2$ after crossover operation using the two parent chromosomes Parent1 and Parent2.

4.2.5 Mutation

Mutation is used to preserve the genetic diversity of the population from inter-generational chromosomes [40]. A common mutation operator for permutation represents interchanges in the contents of two randomly selected genes, i.e., one or more parts of a chromosome are changed [39]. In mutation, the important parameter is the mutation probability which determines the frequency of mutation of the parts of the chromosome. If there is no mutation, the offsprings are generated immediately after crossover or directly copied without any changes. The proposed mutation operation is performed horizontally over the rows of the chromosome by executing the following steps:

- Generate a random number $r \in [0, 1]$.
- If $r >$ mutation rate, then stop; otherwise,
  - Generate two random integers r_1, r_2 $(1 \le r_1, r_2 \le R)$
  - If $r_1 = r_2$, go to the previous step,
  - Otherwise, swap the two rows r_1 and r_2.

Figure 7 shows the generated modified children Child1, Child2 after executing mutation operation over the two children chromosomes $Child1$, $Child2$.

4.2.6 Fitness function

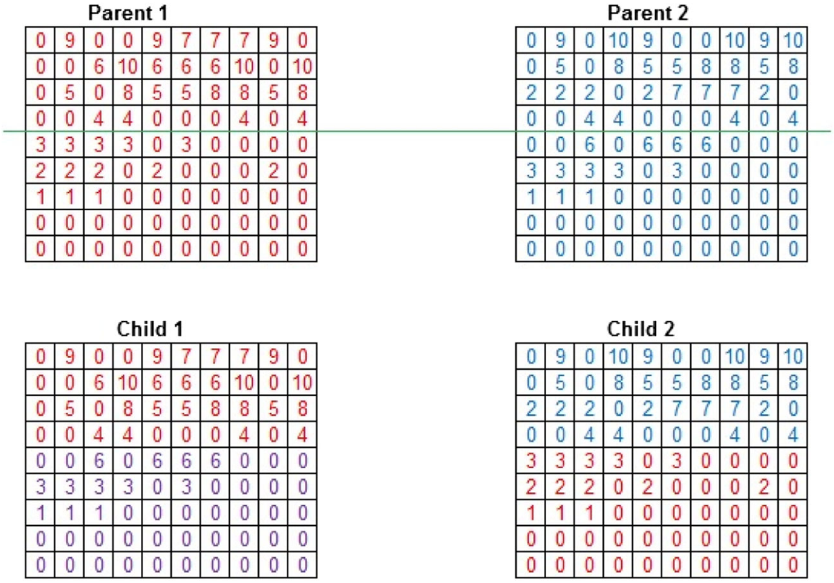
The probability of an individual to reproduce is computed using fitness function. The higher the fitness of an individual, the higher the opportunities to reproduce and mutate. In the proposed technique, the fitness function is used to represent the performance. The aim is to discover an arrangement for sensor nodes to be scheduled and to assign slot for each sensor node without collisions during data transmission while minimizing the data delivery time. The average latency for WSN with n sensor nodes is given by:

$$L_{average} = \frac{\sum_i^n L_i}{n-1}, \tag{3}$$

Here, L_i is the latency of sensor node i which is computed as the total number of waited slots of i till its data reaches the root node.

For clarification, consider the schedule in Fig. 3, the latency of sensor node 9 is 6 slots since node 9 has no waiting slots before starting transmission, plus 4 waiting slots for sensor node 2 (the parent of node 9) before transmission plus 1 slot for sensor node 2 to send. Similarly, L_2 = 6, L_3 =5, L_4 =5, L_5 =6, L_6 =5, L_7 =5, L_8 =5, L_9 =6, and L_{10} =5. According to Eq. 4, this schedule has average latency $L_{average}$=(6+5+5+6+5+5+5+6+5)/9= 5.33

The objective is to minimize average latency for WSN (minimizing Eq. 4). The fitness of each chromosome is evaluated by setting $\frac{1}{1+L_{average}}$ for valid chromosome and set to a very small number for invalid chromosomes.



Parent 1

0	9	0	0	9	7	7	7	9	0
0	0	6	10	6	6	6	10	0	10
0	5	0	8	5	5	8	8	5	8
0	0	4	4	0	0	0	4	0	4
3	3	3	3	0	3	0	0	0	0
2	2	2	0	2	0	0	0	2	0
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Parent 2

0	9	0	10	9	0	0	10	9	10
0	5	0	8	5	5	8	8	5	8
2	2	2	0	2	7	7	7	2	0
0	0	4	4	0	0	0	4	0	4
0	0	6	0	6	6	6	0	0	0
3	3	3	3	0	3	0	0	0	0
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Child 1

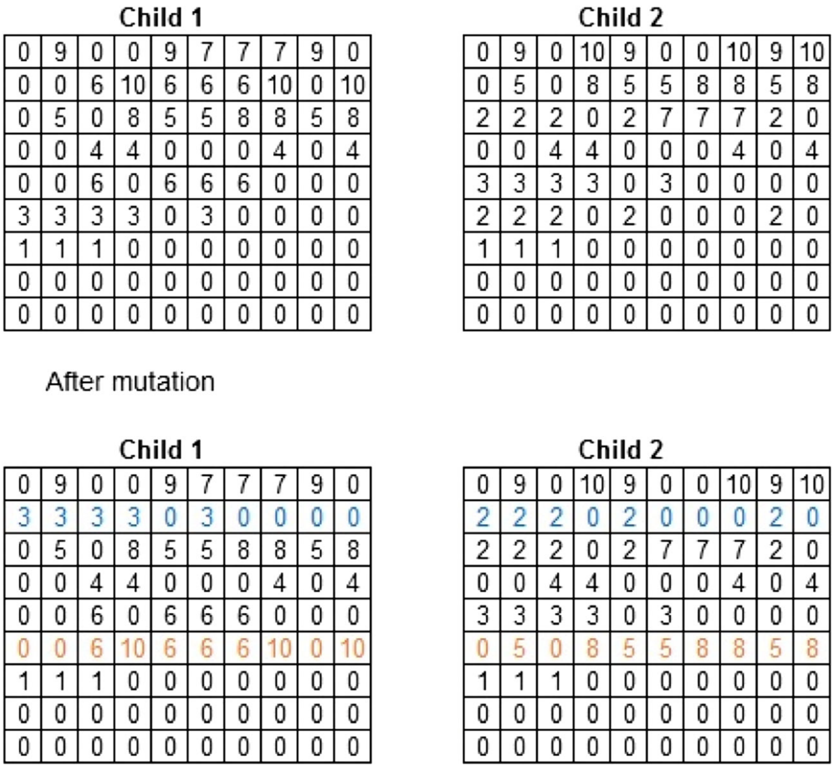
0	9	0	0	9	7	7	7	9	0
0	0	6	10	6	6	6	10	0	10
0	5	0	8	5	5	8	8	5	8
0	0	4	4	0	0	0	4	0	4
0	0	6	0	6	6	6	0	0	0
3	3	3	3	0	3	0	0	0	0
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Child 2

0	9	0	10	9	0	0	10	9	10
0	5	0	8	5	5	8	8	5	8
2	2	2	0	2	7	7	7	2	0
0	0	4	4	0	0	0	4	0	4
3	3	3	3	0	3	0	0	0	0
2	2	2	0	2	0	0	0	2	0
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Fig. 6 Crossover operations Example

Fig. 7 Mutation operations



Child 1

0	9	0	0	9	7	7	7	9	0
0	0	6	10	6	6	6	10	0	10
0	5	0	8	5	5	8	8	5	8
0	0	4	4	0	0	0	4	0	4
0	0	6	0	6	6	6	0	0	0
3	3	3	3	0	3	0	0	0	0
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Child 2

0	9	0	10	9	0	0	10	9	10
0	5	0	8	5	5	8	8	5	8
2	2	2	0	2	7	7	7	2	0
0	0	4	4	0	0	0	4	0	4
3	3	3	3	0	3	0	0	0	0
2	2	2	0	2	0	0	0	2	0
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

After mutation

Child 1

0	9	0	0	9	7	7	7	9	0
3	3	3	3	0	3	0	0	0	0
0	5	0	8	5	5	8	8	5	8
0	0	4	4	0	0	0	4	0	4
0	0	6	0	6	6	6	0	0	0
0	0	6	10	6	6	6	10	0	10
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Child 2

0	9	0	10	9	0	0	10	9	10
2	2	2	0	2	0	0	0	2	0
2	2	2	0	2	7	7	7	2	0
0	0	4	4	0	0	0	4	0	4
3	3	3	3	0	3	0	0	0	0
0	5	0	8	5	5	8	8	5	8
1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

4.3 Example scenario

Figure 8 shows a network example with 8 sensor nodes and its routing tree with node 1 as a sink [24].

In the case of Rand-LO allocation, first leaves in a random order of the routing tree will be scheduled. Then, it schedules sensor nodes in the ascending order of the routing tree. As a result, the traversal order of the routing tree in Fig. 8 will be 6-8-3-7-2-4-1-5. The resulting schedule is presented in Fig. 9a, with an average latency of 5.71. Depth-LO algorithm improves Rand-LO by assigning scheduling priority to the leaves that are far from the sink than the other leaves. Using Depth-LO, the traversal order of the routing tree in Fig. 8 will be 8-6-7-3-4-2-5-1. The Depth-LO's schedule is presented in Fig. 9b, with average latency 5.57.

In DepthRe-LO, sensor nodes on long paths in the routing tree are privileged and scheduled earlier than other sensor nodes. Using DepthRe-LO, the traversal order of the routing tree in Fig. 8 will be 8-6-7-4-3-5-2-1. The DepthRe-LO' schedule is presented in Fig. 10a with average latency 5.43. In IDeg-LO the order of the allocation is based on the interference degree of the sensor nodes, by considering leaves to be scheduled in the descending order of the interference degree and a random node is chosen if interference degrees are equal. Using IDeg-LO, the traversal order of the routing tree in Fig. 8 will be 8-6-7-4-3-2-5-1. The IDeg-LO's schedule is presented in Fig. 10b, with an average latency 5.57.

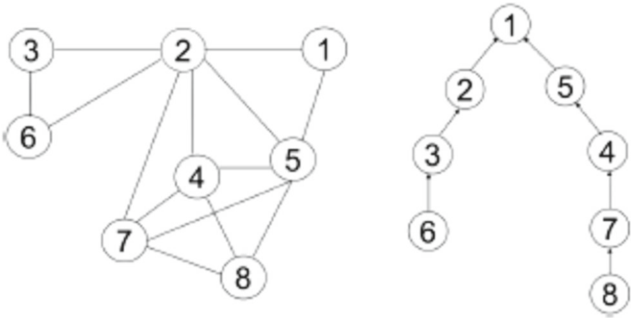
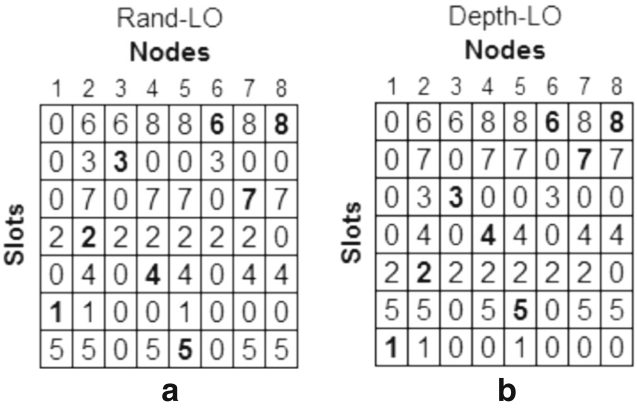


Fig. 8 A WSN graph with its routing tree [24]



Rand-LO

Nodes

Slots	1	2	3	4	5	6	7	8
	0	6	6	8	8	**6**	8	**8**
	0	3	**3**	0	0	3	0	0
	0	7	0	7	7	0	**7**	7
	2	**2**	2	2	2	2	2	0
	0	4	0	**4**	4	0	4	4
	1	1	0	0	1	0	0	0
	5	5	0	5	**5**	0	5	5

a

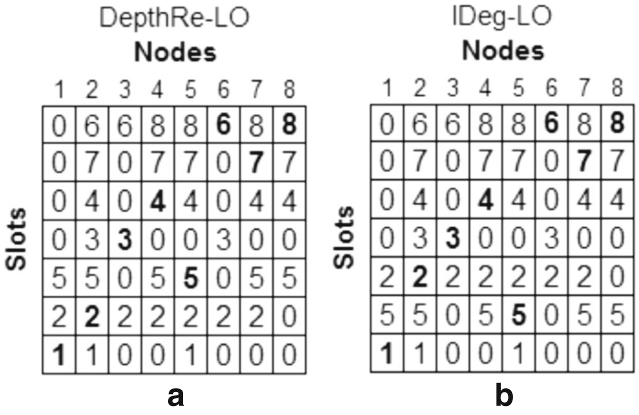
Depth-LO

Nodes

Slots	1	2	3	4	5	6	7	8
	0	6	6	8	8	**6**	8	**8**
	0	7	0	7	7	0	**7**	7
	0	3	**3**	0	0	3	0	0
	0	4	0	**4**	4	0	4	4
	2	**2**	2	2	2	2	2	0
	5	5	0	5	**5**	0	5	5
	1	1	0	0	1	0	0	0

b

Fig. 9 TDMA schedule for WSN in Fig. 8 using **a** Rand-LO **b** Depth-LO



DepthRe-LO

Slots \ Nodes	1	2	3	4	5	6	7	8
	0	6	6	8	8	**6**	8	**8**
	0	7	0	7	7	0	**7**	7
	0	4	0	**4**	4	0	4	4
	0	3	**3**	0	0	3	0	0
	5	5	0	5	**5**	0	5	5
	2	**2**	2	2	2	2	2	0
	1	1	0	0	1	0	0	0

a

IDeg-LO

Slots \ Nodes	1	2	3	4	5	6	7	8
	0	6	6	8	8	**6**	8	**8**
	0	7	0	7	7	0	**7**	7
	0	4	0	**4**	4	0	4	4
	0	3	**3**	0	0	3	0	0
	2	**2**	2	2	2	2	2	0
	5	5	0	5	**5**	0	5	5
	1	1	0	0	1	0	0	0

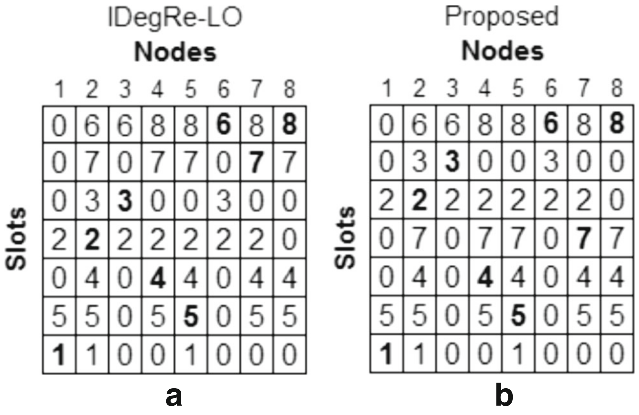
b

Fig. 10 TDMA schedule for WSN in Fig. 8 using **a** DepthRe-LO **b** IDeg-LO

In IDegRe-LO, the order of the allocation is based on the interference degree of leaf sensor nodes, then the leaves removed from the routing tree and the allocation process is repeated over new leaves. Using IDegRe-LO, the traversal order of the routing tree in Fig. 8 will be 8-6-7-3-2-4-5-1. The resulting schedule is presented in Fig. 11a, with average latency 5.14. Using the proposed ETDMA-GA, the traversal order of the routing tree in Fig. 8 will be 8-6-3-2-7-4-5-1. The ETDMA-GA' schedule is presented in Fig. 11b with an average latency 4.71.

5 Simulation results and performance analysis

Here, the simulation setup, the performance metrics and the simulation results along with discussions of the proposed ETDMA-GA and comparison of its results with the baseline algorithms such as Rand-LO, Depth-LO, DepthRe-LO, IDegRe-LO, and IDeg-LO are presented.



IDegRe-LO

Slots \ Nodes	1	2	3	4	5	6	7	8
	0	6	6	8	8	**6**	8	**8**
	0	7	0	7	7	0	**7**	7
	0	3	**3**	0	0	3	0	0
	2	**2**	2	2	2	2	2	0
	0	4	0	**4**	4	0	4	4
	5	5	0	5	**5**	0	5	5
	1	1	0	0	1	0	0	0

a

Proposed

Slots \ Nodes	1	2	3	4	5	6	7	8
	0	6	6	8	8	**6**	8	**8**
	0	3	**3**	0	0	3	0	0
	2	**2**	2	2	2	2	2	0
	0	7	0	7	7	0	**7**	7
	0	4	0	**4**	4	0	4	4
	5	5	0	5	**5**	0	5	5
	1	1	0	0	1	0	0	0

b

Fig. 11 TDMA schedule for WSN in Fig. 8 using **a** IDegRe-LO **b** proposed algorithm

5.1 Simulation setup

The used WSN simulator is designed by David J. Stein and Esq [42]. The simulation consists of two stages: deploying the network and running simulations. The simulator is written in C# using Microsoft Visual Studio.NET. The simulator is modified to cope with our desired mission such as adding the identification number for each sensor node to easily control and identify the position of each sensor and its status. For deployment method, we adopt $MAX - DPA$ algorithm [43] to generate random networks. In our simulation we have the following:

- 100 sensor nodes are deployed randomly on a 2D plane with sink placed at the corner of the deployment area.
- The communication range r_c is fixed as 30 m.
- Different network densities are simulated from 5 to 20 with increment of 5 by changing the network area size. The network density can be roughly calculated as $\rho = (\pi \times r_c^2 \times n)/A$, where r_c is the communication range, n is the number of nodes, and A is the deployed area [23, 24].
- The parameters of GA are as follows: population size=25, maximum generation=4000, crossover probability = 0.8, and mutation probability = 0.05.

The performance of the proposed ETDMA-GA is evaluated using minimum hop routing tree [37] and Depth-First Search Aggregation tree (DFSA) [38]. Then the performances results are compared with DepthRe-LO, Depth-LO, Rand-LO [24], IDeg-LO, and IDegRe-LO [23]. For confidence, the results are averaged using 100 different randomly generated networks using $MAX - DPA$ algorithm [43]. Figure 12 shows examples of used generated graphs with different densities 5, 10, 15, and 20.

5.2 Performance metrics

The used performance metrics as in [23, 24] to compare the performances of different approaches are as follows:

- *Average Latency* is computed as the sum of the latency of each sensor averaged by the total number of sensors and it is calculated as follows:

$$L_{average} = \frac{\sum_i^n L_i}{n-1}, \tag{4}$$

where, L_i is the latency for sensor i.
- *Schedule Length* is the number of slots used in the schedule.
- *Average Normalized Latency* is the average per link latency for all sensors in the network and it is computed as follows:

$$L_{normalized} = \frac{\sum_i^n L_i/h_i}{n-1} \tag{5}$$

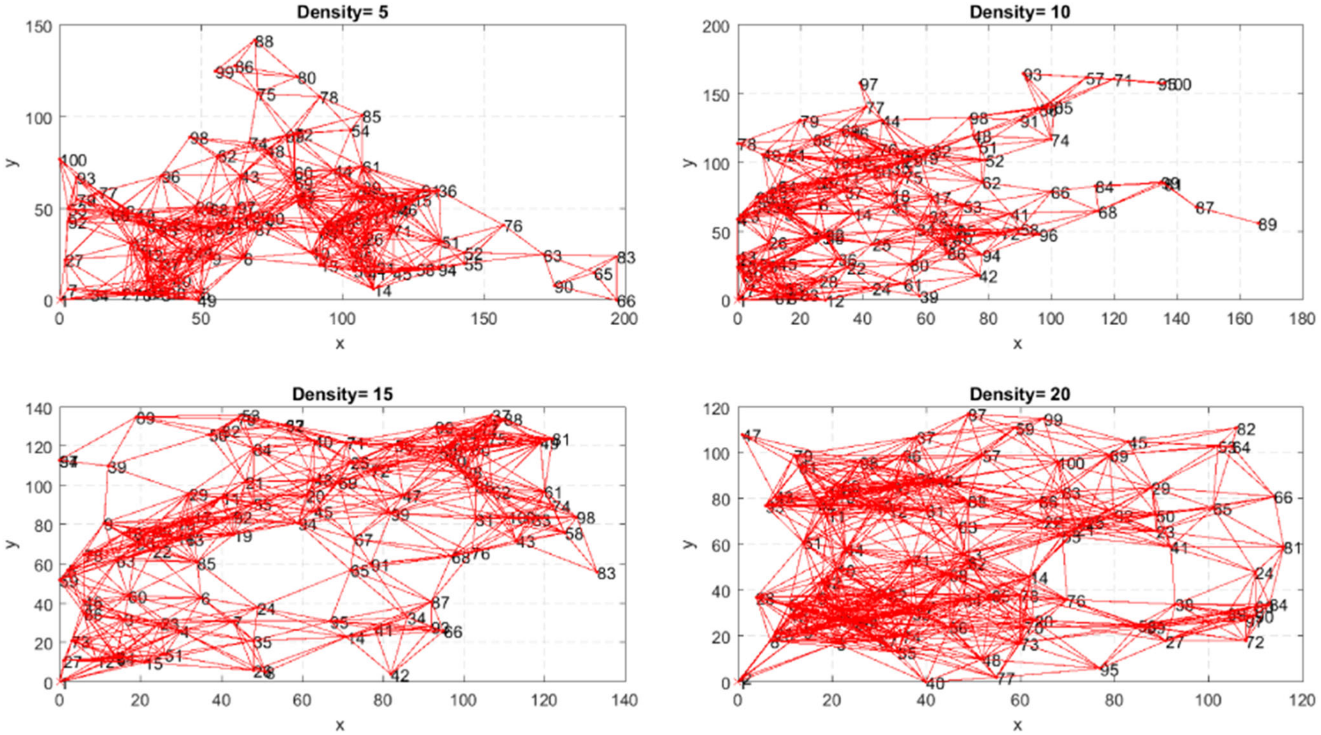


Fig. 12 Examples: Randomly generated graphs with densities 5, 10, 15 and 20

where L_i is the latency, h_i is the number of hops to sink for sensor node i.

- *Duty Cycle* is calculated as the ratio of the number of used slots in transmission or reception to the schedule length.

5.3 Simulation results and discussion

5.3.1 The overhead and convergence

Overhead of ETDMA-GA: As in CoLaNet [22], Rand-LO, Depth-LO, DepthRe-LO [23], IDeg-LO, and IDegRe-LO [24] approaches, ETDMA-GA is a centralized contention-free approach which will be executed by the sink. As we discussed above, the network is brought to a stable state without collisions by finding the best routing tree traversal order and the best TDMA schedule using suitable mapping to GA. The chromosomes are generated by executing the proposed slot allocation algorithm, which successfully finds the best scheduling matrix using the GA operations. The routing tree generated using sensor nodes information such as Min Hop tree [37] or Depth First tree [38] in the initialization phase will be used in the TDMA schedule construction phase to determine the best traversal order by the sink, which will then be used to create the best TDMA scheduling matrix. Finally, the sink will broadcast the best TDMA schedule to all sensor nodes in the network. At this stage, the nodes will be able to transmit messages without collisions making use of the best scheduling found by the ETDMA-GA. We measure the quality of our TDMA scheduling in terms of the number of control messages exchanged. The overall message complexity of ETDMA-GA will be $O(n)$ since it requires n messages to inform the generated scheduling to n nodes. This is the same as that of the overhead complexity of existing algorithms.

Convergence Time: It is crucial to identify whether the algorithm can reach the solution or not. The convergence time of GA based procedures is very important to be estimated to know for how long the procedure will run to get results. This time is surely dependent on the complexity of the algorithm, the defined fitness function and the GA parameters. When convergence occurs the difference between fitness functions is very small, meaning that there is no further progress to be gained, which in turn denotes the convergence of the algorithm.

However, the prediction is complicated to be computed because of its dependence on the application domain. Uniting fitness prediction with hamming distance prediction [44] provides an idea on this, where the predicted average fitness denotes the possible progress using GA and the predicted hamming average that denotes the remaining work. The average hamming distance (hamming average) is used as a measure of population diversity in GA to express convergence. The convergence rate has been used to determine the population size in micro GA [45]. The authors used an experimental approach primarily because of the complexity of the problem under consideration. It is mathematically difficult to identify the characteristics of the best solution. They reached a conclusion that micro GA could be converged in reasonable time. The convergence rate of GA has been discussed using the minorization condition in the Markov chain theory in [46]. They classified genetic algorithms into two cases: time-invariant genetic operators and time-variant genetic operators. For the former case, they have obtained the bound on its convergence rate on the general state space; for the latter case, they have bounded its convergence rate on the finite state space, i.e., the convergence in GA could be occurred and predicted.

In this paper, we use an experimental approach to determine the convergence time because of the complexity of the problem under consideration (NP-hard [47]) and the difficulty to identify the characteristics of its best solution [47]. In the first test, we run the programs with different network density, observing the best fitness reached with each density, see Fig. 13, where sufficient number of generations have been run such that convergence to near-optimal values was apparent. Figure 13 shows that the average latency (objective function) becomes converged in different densities 5, 10, 15, 20 at generation 1500, 3000, 2000, and 2300 respectively. It shows the values to which the proposed algorithm converged, as well as the time needed.

The algorithm stops when the number of generations reaches a specific value which satisfies a specific condition such as stability of average latency etc. Determining stopping criteria is an important factor in the design of GA. The experiment is performed to determine stopping criteria for the proposed algorithm. Different network densities are simulated from 5 to 20 in increment of 5 by changing the network area size with fixed deployment of 100 nodes. The sink is placed at the corner of the network field (0, 0) and the average latency is taken as a decision criterion. To determine the criteria for stopping the GA, the experiment is performed by varying the number of generations for a given population (25). Figure 13 shows the relationship between the objective value (i.e, average latency) and the number of generations for different densities. The graph is obtained by plotting the average latency of the best chromosome of each generation for the same experiment. As the number of generations increase, the value of the average latency decreases. In fact, the weak chromosomes of each generation will be filtered by the fitness function. Thus, the average latency will be improved over successive generations. We can observe that the average latency becomes stable towards the generation 4000 and increases slightly with the increase in the number of generations, i.e., the chances of making significant changes in the future generations are considerably low.

5.3.2 The average latency and the average normalized latency computing

In the second test, we compute the average latency and the Average normalized latency for the proposed ETDMA-GA and compare the results with the algorithms IDegRe-LO, DepthRe-LO, Depth-LO, Rand-LO, and IDeg-LO.

Figures 14 and 15 show that the proposed ETDMA-GA has the best reduction in average latency and average normalized latency among the others regardless of the routing tree type. Where, in case of DFSA tree, the best reduction in average latency is of the proposed ETDMA-GA followed by IDegRe-LO, DepthRe-LO, Depth-LO, Rand-LO, and then IDeg-LO. While in case of Min-Hop tree, also the proposed algorithm has the best latency reduction in average, followed by IDegRe-LO, DepthRe-LO, Depth-LO, IDeg-LO, and then Rand-LO. Because IDeg-LO prioritizes

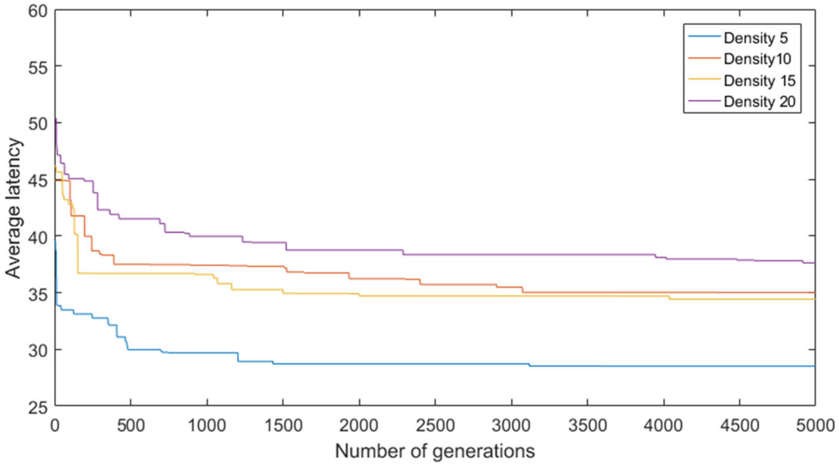


Fig. 13 Effect of number of generations on objective value

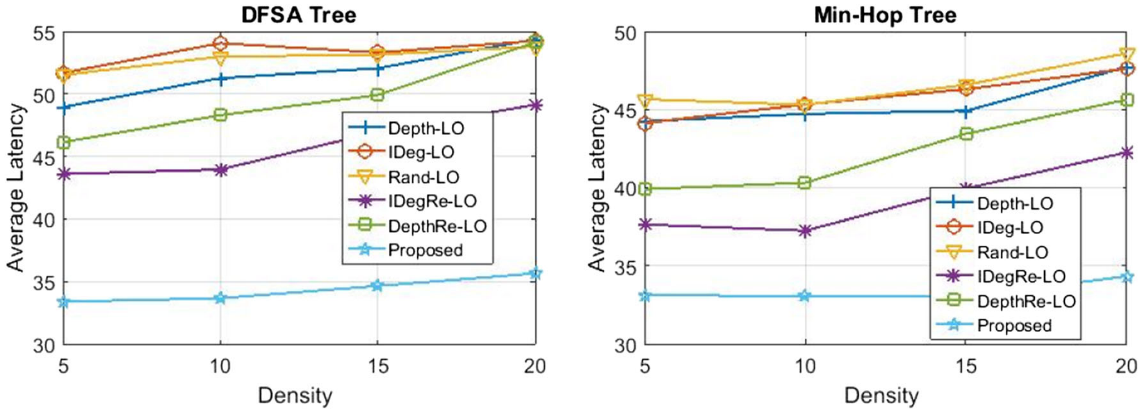


Fig. 14 Average latency of Rand-LO, Depth-LO, DepthRe-LO, IDegRe-LO, IDeg-LO, and proposed ETDMA-GA algorithms based on the information of DFSA and Min-Hop routing trees

the leaves of the routing tree with much interference while IDegRe-LO privileges internal sensor nodes with a lot of interference which shows that IDegRe-LO has better latency reduction than IDeg-LO. Moreover, with DFSA tree and Min-Hop tree, IDegRe-LO is better than DepthRe-LO, Rand-LO, IDeg-LO and Depth-LO because IDegRe-LO prefers to schedule first the critical sensors and sensors that are far from the sink which creates a lot of interference in the network. DepthRe-LO has better performance than Depth-LO as it allocates the first slots of TDMA to nodes with long paths.

In the third test, the average schedule length for the proposed ETDMA-GA is computed and the result is compared with the algorithms IDegRe-LO, DepthRe-LO, Depth-LO, Rand-LO and IDeg-LO. Figure 16 shows that the proposed ETDMA-GA has longer TDMA length than IDeg-LO by 0.2% in average for Min-Hop tree, and than IDegRe-LO by 3% in average for DFSA tree. The proposed ETDMA-GA has these additional slots due to the latency minimization.

In the fourth test, the Average latency for ETDMA-GA for different densities is computed and the comparison with the baselines algorithms is performed in the case of Min-Hop and DFSA trees. Figure 17 shows the Average gain in latency reduction by the proposed ETDMA-GA with different densities compared with Depth-LO, IDeg-LO, Rand-LO, IDegRe-LO and DepthRe-LO. It shows for Min-Hop tree that, for all densities, the latency average of the proposed algorithm is improved by 26.47%, 27.19%, 28.27%, 15.01%, 21.14% as compared to the latency of Depth-LO, IDeg-LO, Rand-LO, IDegRe-LO, DepthRe-LO, while for DFSA tree, the latency average of the proposed ETDMA-GA is improved by 33.55%, 35.61% , 35.04%, 25.22%, 30.81% as compared with the latency of Depth-LO, IDeg-LO, Rand-LO, IDegRe-LO and DepthRe-LO.

Figure 18 shows the standard deviation for latency with different densities for proposed ETDMA-GA, IDegRe-LO, DepthRe-LO, Depth-LO, Rand-LO and IDeg-LO algorithms. It is clear from Fig. 18 that the proposed

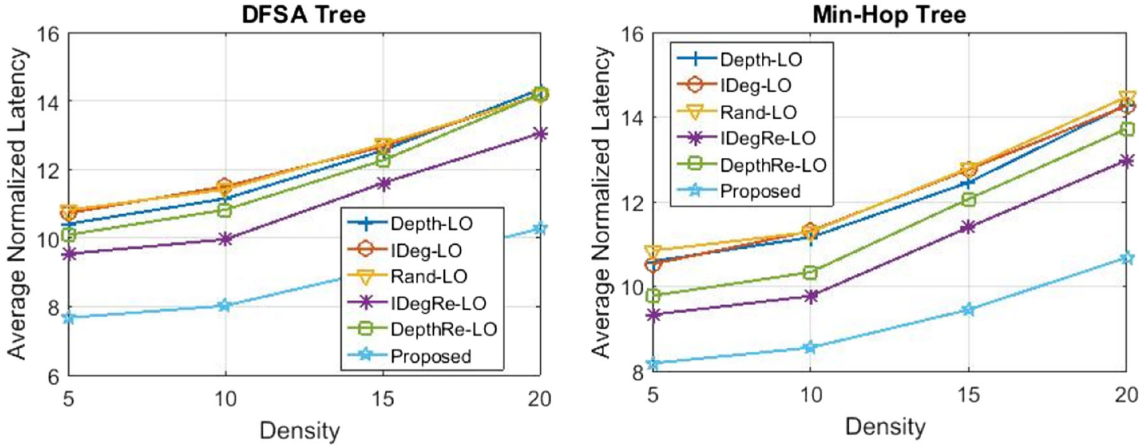


Fig. 15 Average normalized latency of DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO, IDegRe-LO, and proposed ETDMA-GA algorithms based on the information of DFSA and Min-Hop routing trees

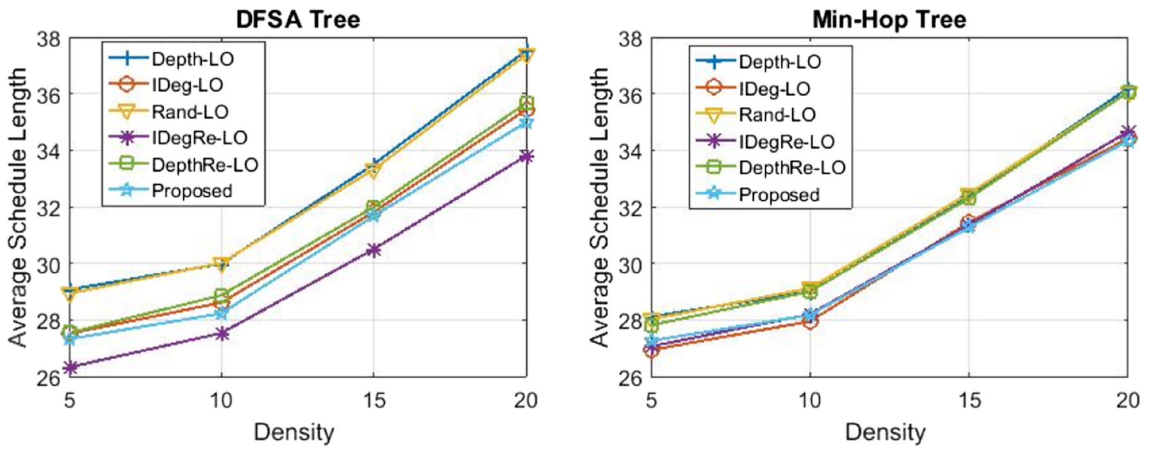


Fig. 16 Average schedule length of DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO, IDegRe-LO, and proposed algorithms based on the information of DFSA and Min-Hop routing trees

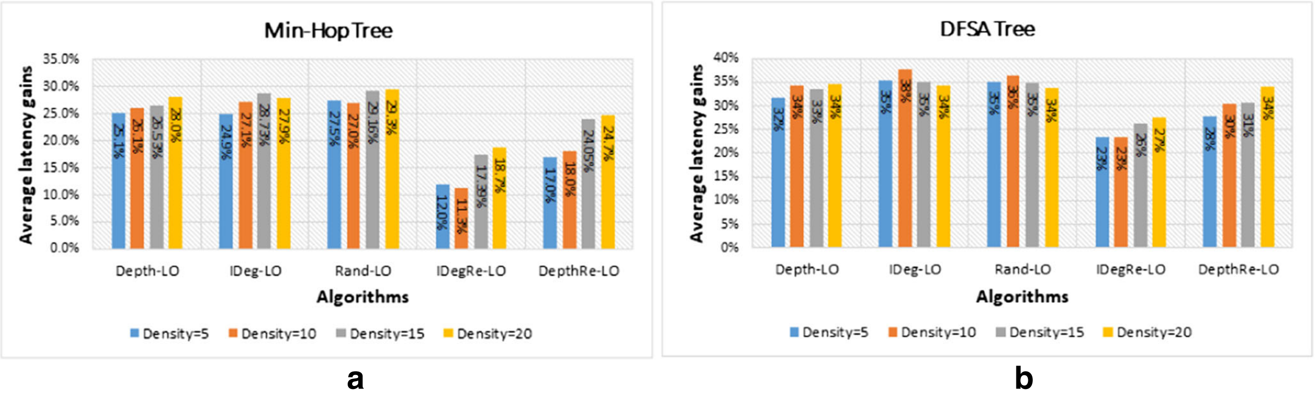


Fig. 17 Average gain in latency reduction with different densities by proposed ETDMA-GA, IDegRe-LO, DepthRe-LO, Depth-LO, Rand-LO, and IDeg-LO algorithms

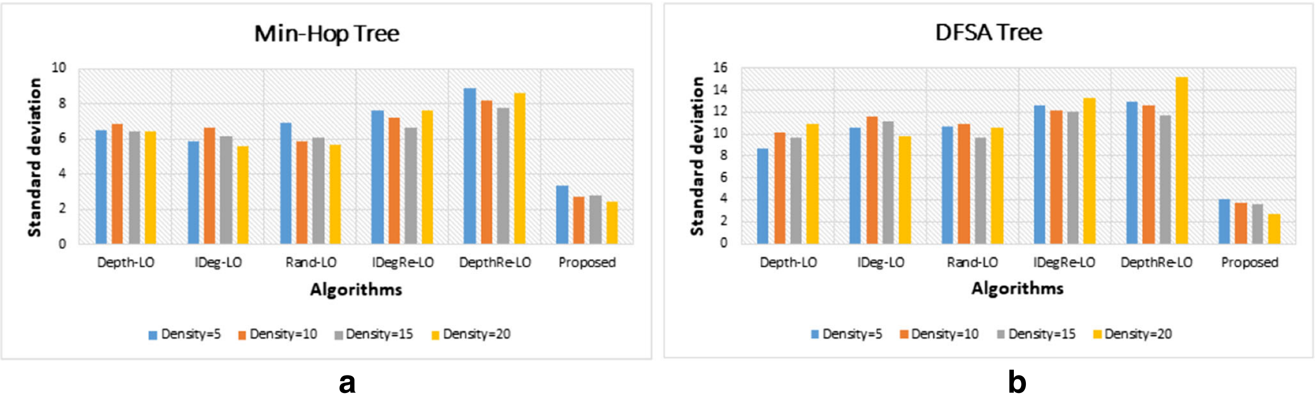


Fig. 18 Standard deviation for latency with different densities of proposed ETDMA-GA, IDegRe-LO, DepthRe-LO, Depth-LO, Rand-LO, and IDeg-LO algorithms

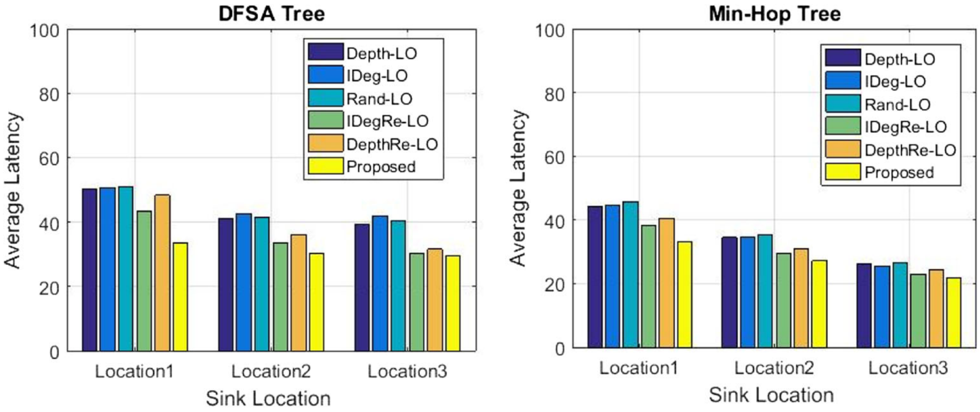


Fig. 19 Average latency of proposed ETDMA-GA, DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO and IDegRe-LO algorithms in case of DFSA and Min-Hop routing trees with three different locations of sink

approach has less standard deviation than the other approaches.

5.3.3 Effect of sink placement

In the fifth test, our target is to show the effect of moving the sink to different placements on the performance of proposed ETDMA-GA, DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO, and IDegRe-LO algorithms. In this test, 100 sensor nodes with communication range fixed at 30 m are deployed randomly in 150 $m\times$ 150 m area. The sink is located at the following different places: the corner of the network ($Location_1$), the middle of one edge of the network ($Location_2$) and the center of the network ($Location_3$). The performance is evaluated using Min-hop and DFSA routing trees. Figures 19, 20, and 21 show the average latency, average normalized latency and average schedule length of proposed ETDMA-GA, DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO and IDegRe-LO algorithms in case of DFSA and Min-Hop routing trees with three different locations of the sink.

In Fig. 19, the latency is improved at $Location_2$ by 21.80%, 22.59%, 22.74%, 22.89%, 23.46%, 17.74% for Depth-LO, IDeg-LO, Rand-LO, IDegRe-LO, DepthRe-LO, and the proposed ETDMA-GA, respectively as compared to $Location_1$ in case of using Min-Hop tree. However, in case of DFSA tree, the latency is improved at $Location_2$ by 18.09%, 15.54%, 18.51%, 22.38%, 24.93%, 9.33% for Depth-LO, IDeg-LO, Rand-LO, IDegRe-LO, DepthRe-LO, and the proposed ETDMA-GA, respectively compared to $Location_1$.

Moreover, the latency is improved at $Location_3$ by 41.04%, 42.77%, 42.08%, 39.96%, 39.61%, 33.92% for Depth-LO, IDeg-LO, Rand-LO, IDegRe-LO, DepthRe-LO, and proposed ETDMA-GA, respectively as compared to $Location_1$ in case of Min-Hop tree. However, in case of DFSA tree, the latency is improved at $Location_3$ by 21.82%, 17.28%, 20.80%, 30.05%, 34.55%, 11.48% for Depth-LO, IDeg-LO, Rand-LO, IDegRe-LO, DepthRe-LO, and the proposed ETDMA-GA, respectively compared to $Location_1$.

Figures 19, 20, and 21 show that when the sink location is moved to the network center ($Location_3$); the latency is minimized for all algorithms for both cases of routing trees because the paths from nodes to the sink is decreased.

5.3.4 Duty cycle

In the sixth test, the duty cycle of the proposed ETDMA-GA as the ratio of the active period (estimated by the number

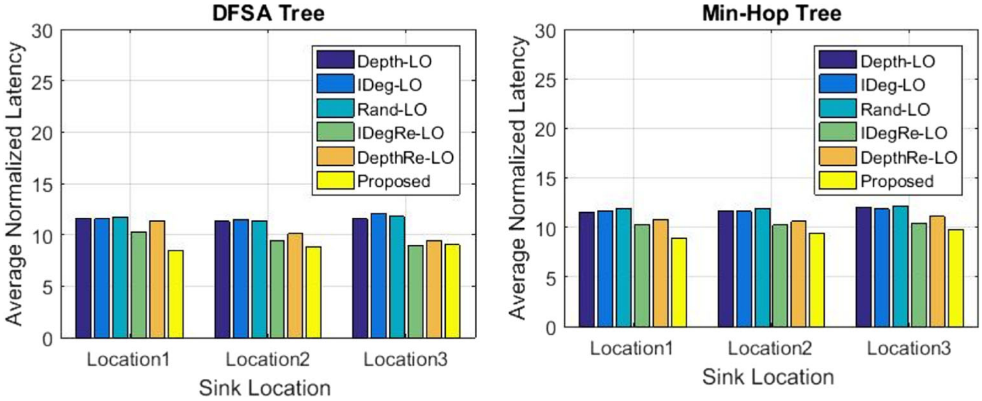


Fig. 20 Average normalized latency of proposed ETDMA-GA, DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO and IDegRe-LO algorithms in case of DFSA and Min-Hop routing trees with three different locations of sink

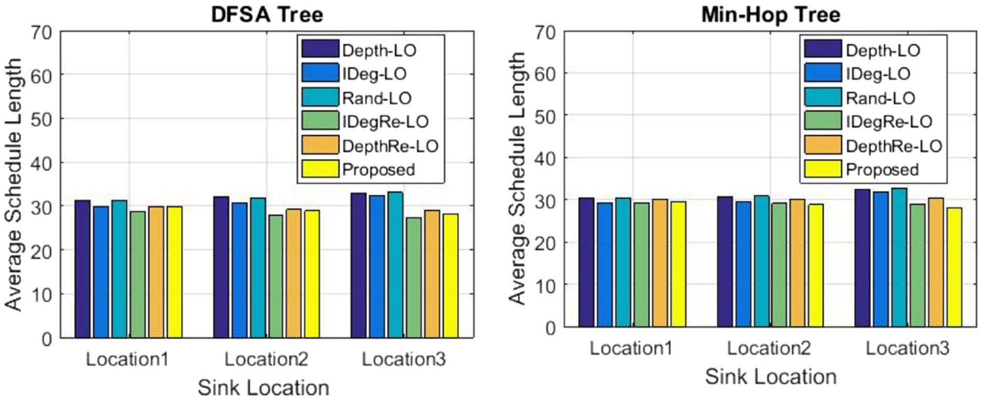


Fig. 21 Average schedule length of proposed ETDMA-GA, DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO and IDegRe-LO algorithms in case of DFSA and Min-Hop routing trees with three different locations of sink

of slots used either in transmission or in reception) to the total period (the schedule length) is computed. The results are compared with the existing algorithms such as DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO, IDegRe-LO. A lower duty cycle results in a lower energy consumption [48]. Figure 22 shows that for Min-Hop and DFSA trees, Rand-LO and Depth-LO have the best duty cycle due to their longer schedule length and then followed by DepthRe-LO, proposed ETDMA-GA, IDegRe-LO and IDeg-LO because their schedule lengths are smaller than those of Rand-LO and Depth-LO.

Moreover, Fig. 22 shows that in case of Min-Hop tree, the average percentages overall densities of duty gain of Depth-LO, Rand-LO, DepthRe-LO compared to the proposed ETDMA-GA are 3.3%, 3.4%,3.2%, respectively. While the average percentages overall densities of duty gain of proposed ETDMA-GA compared to IDeg-LO and IDegRe-LO are 0.7 % and 0.2%, respectively. In case of DFSA tree, the average percentages of duty gain of Depth-LO, IDeg-LO, Rand-LO, DepthRe-LO compared to proposed ETDMA-GA are 6.2%, 0.8%,6%, 1.5%, respectively. While the duty gain overall average percentages of proposed ETDMA-GA compared to IDegRe-LO is 3.3%. The main conclusion of this test is that the duty cycle of the proposed ETDMA-GA shows diminutive increase compared to other existing algorithms while the proposed ETDMA-GA achieves remarkable gain in average latency.

In summary, the proposed heuristic TDMA based genetic technique (ETDMA-GA) enumerates some of the possible solutions randomly and then generates other possible solutions by crossover and mutation operations. This makes the proposed algorithm to search better than existing blind search algorithms which start from initial point (may be bad point) and generate a solution. Such solution is based on this bad/good starting point. However, the proposed ETDMA-GA can start from many points, then evaluate each point and select the best and improve the worst. ETDMA-GA repeats these steps until it finds the final best solution.

6 Conclusion

In this work, a new TDMA scheduling based on genetic algorithm using latency as fitness function in order to reduce the latency of communications in tree based WSN have been proposed. Two dimensional chromosome matrix is designed

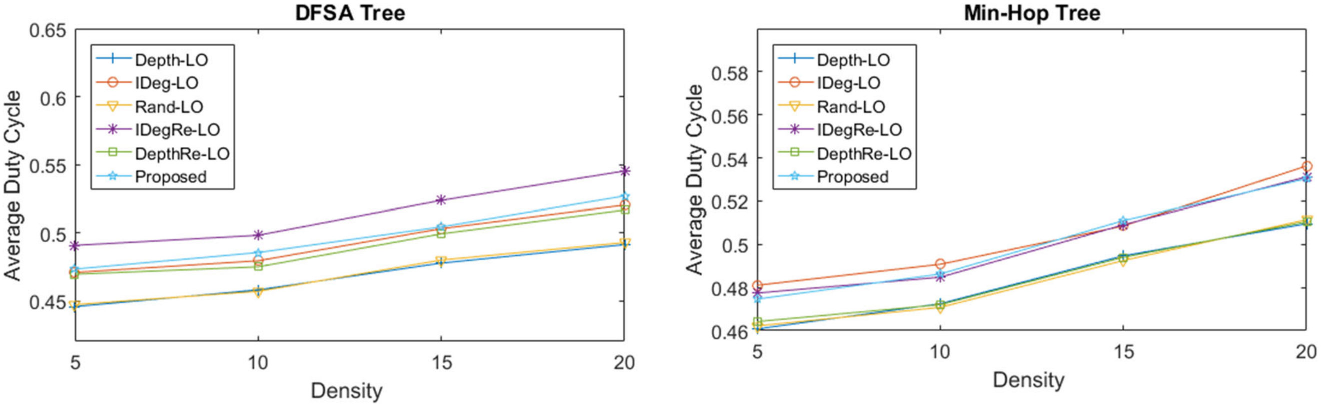


Fig. 22 Duty Cycle of the proposed ETDMA-GA, DepthRe-LO, Depth-LO, Rand-LO, IDeg-LO, and IDegRe-LO algorithms in case of DFSA and Min-Hop routing trees

to represent TDMA schedule and the required genetic operations are provided. The proposed algorithm utilizes the information that comes from routing protocol in order to find the best design for TDMA schedule. Simulation results show that the proposed algorithm has better performance than existing state-of-the-art approaches such as Rand-LO, Depth-LO, DepthRe-LO, IDegRe-LO and IDeg-LO in terms of average latency, average normalized latency and average schedule length.

References

1. Osamy W, Khedr AM, Aziza A, El-Sawya A (2018) Cluster-Tree Routing scheme for data gathering in periodic monitoring applications. IEEE Access 6:77372–77387
2. Osamy W, Salim A, Khedr AM (2018) An Information Entropy Based-Clustering Algo- rithm in Heterogeneous Wireless Sensor Networks. Wireless networks, Springier, https://doi.org/10.1007/s11276-018-1877-y(0123456789
3. Aziz A, Singh K, Osamy W, Khedr AM (2019) Effective algorithm for optimizing compressive sensing in IoT and periodic monitoring applications. J Netw Comput Appl 126(15):12–28
4. Osamy W, Khedr AM, Salim A, Agrawal DP Sensor network node scheduling for preserving coverage of wireless multimedia networks. IET Wireless Sensor Systems, 2019, https://doi.org/10.1049/iet-wss.2018.5119, IET Digital Library, https://digital-library.theiet.org/content/journals/10.1049/iet-wss.2018.5119
5. Demirkol I, Ersoy C, Alagoz F (2006) Mac protocols for wireless sensor networks: a survey. IEEE Commun Mag 44(4):115–121
6. Gandham S, Zhang Y, Huang Q (2008) Distributed time-optimal scheduling for convergecast, in wireless sensor networks. Computer Networks 52(3):610–629
7. Osamy W, El-sawy A, Khedr AM (2019) SATC: A simulated annealing based tree construction and scheduling algorithm for minimizing aggregation time in wireless sensor networks. Wireless Personal Communications, pp 1–18
8. Incel OD, Ghosh A, Krishnamachari B (2011) Scheduling algorithms for tree-based data collection in wireless sensor networks. In: Nikoletseas S, Rolim J (eds) Theoretical Aspects of Distributed Computing in Sensor Networks. Monographs in Theoretical Computer Science. An EATCS Series. Springer, Berlin
9. Sgora A, Vergados DJ (2015) D. d vergados, "A survey of TDMA scheduling schemes in wireless multihop networks. ACM Comput Surv 3:47. https://doi.org/10.1145/2677955
10. Fahmy HMA (2016) Protocol stack of WSNs. In: Wireless Sensor Networks Signals and Communication Technology. Springer, Singapore
11. Melodia T, Vuran MC, Pompili D (2005) The state of the art in cross-layer design for wireless sensor networks. In: Cesana M, Fratta L (eds) Wireless Systems and Network Architectures in Next Generation Internet. EuroNGI 2005. Lecture Notes in Computer Science, vol 3883. Springer, Berlin
12. Louail L, Felea V (2016) Routing and TDMA joint Cross-Layer design for wireless sensor networks. In: Mitton N, Loscri V, Mouradian A (eds) Ad-hoc, Mobile, and Wireless Networks. ADHOC-NOW
13. Kulkarni SS, Iyer A, Rosenberg C (2006) An address-light, integrated MAC and routing protocol for wireless sensor networks. IEEE/ACM Trans Netw 14(4):793–806
14. Awang A, Lagrange X, Ros Sanchez D (2009) A cross-layer medium access control and routing protocol for wireless sensor networks. In: 10èmes Journées Doctorales en Informatique et Réseaux, 2-4 février, Belfort, France
15. Chen D, Deng J, Varshney PK (2005) A state-free data delivery protocol for multihop wireless sensor networks, In: WCNC, pp 1818–1823
16. Landsiedel O, Ghadimi E, Duquennoy S, Johansson M (2012) Low power, low delay : opportunistic routing meets duty cycling. In: IPSN, pp 185–196
17. Louail L, Felea V, Bernard J, Guyennet H (2015) MAC-aware routing in wireless sensor networks. In: International Black Sea Conference on Communications and Networking (BlackSeaCom), IEEE, pp 225-229
18. Suh C, Ko Y, Son D (2006) An energy efficient Cross-Layer MAC protocol for wireless sensor networks. In: APWeb Workshops, pp 410–419
19. Du S, Kumar Saha A, Johnson DB (2007) RMAC : A Routing-Enhanced duty- cycle MAC protocol for wireless sensor networks. In: INFOCOM, pp 1478–1486
20. Hefeida M, Canli T, Khokhar AA (2013) CL-MAC : A Cross-Layer MAC Protocol for heterogeneous Wireless Sensor Networks. Ad Hoc Networks 11(1):213–225
21. Heurtefeux K, Maraninchi F, Valois F (2011) Areacast: A cross-layer approach for a communication by area in Wireless Sensor Networks. In: ICON, pp 112–117
22. Chou C, Chuang K (2005) Colanet : A Cross-Layer Design of Energy-Efficient Wireless Sensor Networks. In: ICW/ICHSN/ICMCS/SENET, pp 364–369
23. Louail L, Felea V (2016) Routing-Aware Time slot allocation heuristics in Contention-Free sensor networks. In: Mamatas L, Matta I, Papadimitriou P, Koucheryavy Y (eds) wired/wireless internet communications. WWIC
24. Louail L, Felea V (2016) Routing-aware TDMA scheduling for wireless sensor networks. In: 12^{th} Annual Conference on Wireless On-demand Network Systems and Services (WONS), Cortina d'Ampezzo, pp 1–8
25. Davarzani Z, Yaghmaee MH, Akbarzadeh-T MR (2011) TDMA scheduling in wireless sensor network using artificial immune system. In: Gaspar-Cunha A, Takahashi R, Schaefer G, Costa L (eds) Soft Computing in Industrial Applications. Advances in Intelligent and Soft Computing, vol 96. Springer, Berlin
26. Wang T, Wu Z, Mao J (2007) PSO-based hybrid algorithm for multi-objective TDMA scheduling in wireless sensor networks. In: 2007 Second International Conference on Communications and Networking in China, Shanghai , pp 850–854
27. Mao J, Wu Z, Wu X (2007) A TDMA scheduling scheme for many-to-one communications in wireless sensor networks. Comput Commun 30(4):863–872
28. Mao J, Wu X, Wu Z, Wang S (2006) A Novel Energy-Aware TDMA scheduling algorithm for wireless sensor networks. In: Cheng X, Li W, Znati T (eds) Wireless Algorithms, Systems, and Applications. WASA 2006. Lecture Notes in Computer Science, vol 4138. Springer, Berlin
29. Khamayseh Y, Mardini W, Ben Halima N (2018) Evolutionary algorithm for scheduling in wireless sensor networks. J Comput 13(3):262–270
30. Coleri Ergen S, Varaiya P (2010) TDMA Scheduling algorithms for wireless sensor networks. Wirel Netw 16(4):985–997. https://doi.org/10.1007/s11276-009-0183-0
31. Zibakalam V (2012) A New TDMA Scheduling Algorithm for Data Collection over Tree-Based Routing in Wireless Sensor Networks. ISRN Sensor Networks, vol 2012 Article ID 864694

32. Liu J, Ravishankar CV (2011) LEACH-GA: Genetic Algorithm-based energy-efficient adaptive clustering protocol for wireless sensor networks. Int J Mach Learn Comput 1:79–85
33. Bhatia T, Kansal S, Goel S, Verma AK (2016) A genetic algorithm based distance-aware routing protocol for wireless sensor networks. Comput Electr Eng 56:441–455
34. Sharma G, Kumar A (2017) Improved range-free localization for three-dimensional wireless sensor networks using genetic algorithm. Computers & Electrical Engineering
35. Kumar Gupta S, Kuila P, Jana PK (2016) Genetic algorithm approach for k-coverage and m-connected node placement in target based wireless sensor networks. Comput Electr Eng 56:544–556
36. Shen Y-J, Wang M-S (2008) Broadcast scheduling in wireless sensor networks using fuzzy Hopfield neural network. Expert Syst Appl. 34(2):900–907
37. Shao-Shan J, Huang C-H, Guangqiong Z (2007) A minimum hop routing protocol for wireless sensor networks. Hua University, 24 2007.12 [China 96.12], pp 45–57
38. Bertrand Fotue Fotso D (2013) Efficient data aggregation and routing in wireless sensor networks. Networking and Internet Architecture [cs.NI], Té,lécom ParisTech, chapter 3
39. Sivanandam SN, Deepa SN (2007) Introduction to Genetic Algorithms (1st ed.)," Springer Publishing Company Incorporated
40. Ming-wen T, Tzung-pei H, Woo-tsong L (2015) A Two-Dimensional Genetic Algorithm and Its Application to Aircraft Scheduling Problem," Mathematical Problems in Engineering, vol. 2015 Article ID 906305
41. Goldberg DE, Deb K (1991) A comparative analysis of selection schemes used in genetic algorithms. In: Rawlins GJE (ed) Foundations of genetic algorithms, morgan kaufmann, pp 69–93
42. Alfaqih TM, Al-Dhelaan AM, Mehedi Hassan M (2015) Wireless sensor network simulation environment. International Journal of Computer Applications, vol. 118, no. 17
43. Onat FA, Stojmenovic I (2007) Generating random graphs for wireless actuator networks. In: 2007 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, Espoo, pp 1–12
44. Louis SJ, Rawlins GJE (1993) Predicting convergence time for genetic algorithms. Foundations of Genetic Algorithms 2, Whitley L D., editor, Morgan Kaufmann, pp 141–161
45. Abu-Lebdeh G, Benekohal RF (1999) Convergence variability and population sizing in Micro-Genetic algorithms. Comput-Aided Civ Inf Eng 14:321–334
46. He J, Kang L (1999) On the convergence rates of genetic algorithms. Theor Comput Sci 229:23–39
47. Coleri Ergen S, Varaiya P (2010) TDMA Scheduling algorithms for wireless sensor networks 16(4):985–997
48. Shuguo Z, Ye-Qiong S, Zhi W, Zhibo W (2012) Queue-mac: a queue-length aware hybrid csma/tdma mac protocol for providing dynamic adaptation to traffic and duty-cycle variation in wireless sensor networks. In: 9^{th}, IEEE WFCS, pp 105–114

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Walid Osamy received his B.Sc. degree with honor in computer science in June 2000, the M.S. degree in June 2006, and the Ph.D. degree in September 2010 all in area of computer science and from the Faculty of science, Zagazig University, Egypt. He has been involved in the projects of Network Infrastructure, and Management information systems at Communication Information Technology Center (CITC), Zagazig University, Egypt. From December 2010 to January 2019 he has been working as assistant professor at the Department of Computer Science, College of Computers and Informatics, Benha University, Egypt. From February 2019 till now he is working as associate professor at the Department of Computer Science, College of Computers and Artificial Intelligence , Benha University, Egypt. From March 2015 until now, he has been working as assistant professor at Qassim University, Buridah, KSA. His research interests include Computational Intelligence and in the field of IoT (Mobile Computing, Wireless Sensor Networks (WSNs))

Ahmed A. El-Sawy Associate professor in computer science at Benha University since 2017. From 2017 I served as head of computer science in faculty of computers and artificial intelligence, Benha University, Egypt. I received B.S. and M.S. from Cairo University in 2000 and 2003 respectively. I received Ph.D. in computer science from Suez Canal University in 2011. My researches interests with using artificial intelligence in computer network, bioinformatics, chimoinformatics, and optimization. I interests with deep learning also.

Ahmed M. Khedr received his B.Sc. degree in Mathematics in June 1989 and the M.Sc. degree in the area of optimal control in July 1995 both from Zagazig University, Egypt. In July 1999 he received his M.Sc. and in March 2003 he received his Ph.D. degree both in area of Computer Science and Engineering from University of Cincinnati, Ohio, USA. From March 2003 to January 2004, he was a research assistant professor at ECECS Department University of Cincinnati, USA. From January 2004 to May 2009, he worked as assistant professor at Zagazig University, Egypt, from September 2009 to September 2010 he worked as associate professor at the Department of Computer Science, college of computers and Information Systems, Taif University, KSA, from September 2010 till now he is working as associate professor at the University of Sharjah, UAE, and from September 2014 till now he is working as a professor at the Department of Mathematics, Faculty of Science, Zagazig University, Egypt. In June 2009, he awarded the State Prize of distinction in advanced technology. In May 2013, he awarded the Sharjah Islamic Bank prize of distinction in research. In April 2015, he awarded the university of Sharjah prize of distinction in research. He has coauthored 65 works in journals and conferences relating with optimal control, Wireless Sensor Networks, Internet of Things, Distributing Computing, and Bioinformatics.