Journal of Theoretical and Applied Information Technology
31st May 2019. Vol.97. No 10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

IRBYTSA: A NOVEL LINK-SCHEDULING ALGORITHM FOR IEEE 802.15.4E TSCH NETWORK

[1,3]IMAN HEDI SANTOSO, [2]KALAMULLAH RAMLI
[1,2]Department of Electrical Engineering, Faculty of Engineering, Universitas Indonesia, Indonesia
E-mail: [1]iman.hedi@ui.ac.id, [2]k.ramli@eng.ui.ac.id, [3]mnhedi@gmail.com

ABSTRACT

This paper proposes a novel algorithm named IRByTSA, as a scheduling algorithm that is both simpler and faster than TASA, in terms of its speed in generating link-schedule decision. The simplification makes IRByTSA a low-complexity algorithm. The IRByTSA algorithm owes its relatively low complexity to the following procedures: maximum matching to generate *link-schedule;* maximization of nodes performing simultaneous transmissions by prioritizing from higher-ranking nodes to leaf node; transmission of all queued data packets in bursts *(bursty);* and provides each node with a transmission opportunity based solely on turn. The research confirms that the resulting complexity is $O(\lfloor n^{0.65} \rfloor . n)$, which indicates that IRByTSA is a low-complexity algorithm. The advantage of using such a fast and low-complexity algorithm is increased network scalability, as the reduction in complexity and increase in speed enable the existing PCE to serve additional networks. .

Keywords: *IoT, IEEE802.15.4e, Scheduling, Algorithm, Matching.*

1. INTRODUCTION

The Internet of Things (IoT) has attracted significant research interest globally as a potential solution to pressing social issues related to health, demography, welfare, food security and agriculture, energy, transportation, and efficient use of resources. For that reason, IoT will be an important element in the future Internet, connecting billions of interconnected heterogeneous objects [1,2]. Projections indicate that, by 2020, up to 24 billion devices will be connected to the Internet [3], supported by the emergence of IoT. However, as the current Internet infrastructure cannot accommodate this rapid development, a new network architecture is needed to manage the dramatic increase of IoT flow, allowing multiple services with different QoS requirements to co-exist [4].

Current thinking advocates a centralized network control system for IoT network management, with the development of a complex routing topology and simplified operation for users in non-IT environments. The need to implement a centralized control system characterizes the intersection between the Internet of Things (IoT) and Software-Defined Network (SDN) [5]. In an attempt to realize a centralized control system, the IETF 6TiSCH workgroup was established in November 2013. The goal of 6TiSCH WG is to connect IEEE802.15.4e TSCH MAC layer to IPv6 on the top layer. The 6TiSCH WG protocol stack is based on existing standards for the Internet of Things, including RPL, 6LoWPAN, and CoAP. The workgroup has been developing an architecture that will allow low-power wireless devices (sometimes called "motes") to form a multi-hop low-power lossy network (LLN) [6]. Figure 1 below describes the 6TiSCH protocol stack for TSCH-based LLN [7].

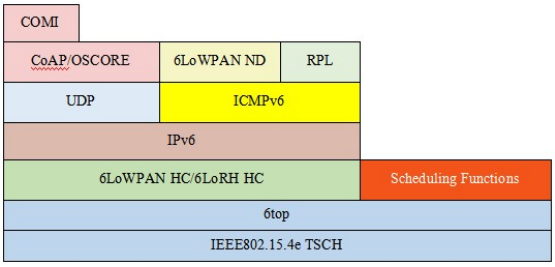


Figure 1: 6TiSCH Protocol Stack.

As shown in Figure 1, a sub-layer is responsible for handling the matters related to scheduling, namely Scheduling Functions. The TSCH technology adopted by LLN requires such a scheduling function because each of its nodes can only transmit or receive at particular time slots. This scheduling function generates a scheduling scheme that informs LLN nodes when to transmit, receive, or enter an idle state. How a scheduling

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

scheme is developed determines the volume of traffic generated by LLN, length of packet delay, and energy consumption at each node. In the present research, the communication scheduling scheme is controlled in centralized mode by a dedicated server running an algorithm to generate a communication schedule, which is in turn the schedule will be informed to all LLN nodes.

This research builds on a previous study [8] that produced a centralized scheduling algorithm for IEEE802.15.4e TSCH as an improved version of TASA [9, 10, 11]. The scheduling algorithm proposed in that research [8] was named Iman-Ramli TASA (IR-TASA). In research [8], it was proven that IRTASA's link scheduling algorithm was better than TASA in terms of the speed in generating a link scheduling decision. IRTASA obtained the number of active timeslots, which was the same or near the minimum, more rapidly than did TASA. However, IRTASA is too complicated and thus needs to be simplified.

Research paper [8] also referred to several other studies [12–14] describing modifications that enhanced TASA performance. Other papers on centralized scheduling for IEEE802.15.4e TSCH include Choi et al. [15], which proposed centralized link scheduling (CLS) utilizing Routing Protocol for LLN (RPL) to generate scheduling decisions. Choi et al. compared CLS to DETAS in terms of the number of control messages needed to create a schedule. Their goal [15] was to develop a scheduling algorithm that required fewer control messages. In another study, Livolant et al. [16] explored a signaling mechanism for a centralized scheduling algorithm called MODESA, with reference to the number of messages needed to install and upgrade a scheduling scheme on a network. For the purposes of performance comparison, they used three standards: CoAP, CoMI, and OCARI, finally recommending OCARI because it requires fewer messages than the other two.

The prior studies above, which are related to TASA [12–14] and scheduling algorithms for TSCH [15–16], did not discuss the speed of the scheduling algorithms in generating scheduling decisions or quantitatively assess the complexity of the algorithms they proposed. The absence of studies that investigate the speed and complexity of TSCH link scheduling algorithms is the motivation behind research [8] and the current study. These two factors must be examined with regard to how much influence they have on server performance. The hypothesis of this study is that the lower the complexity of the algorithm and the faster the generation of the scheduling decision, the lighter the burden of the algorithm on the server. Therefore, the server can be used to do other tasks.

Given the motivation to develop a low-complexity algorithm, in the present research, the IRTASA scheduling algorithm is enhanced to create a new algorithm called IRByTSA. IRByTSA is significantly simpler than IR-TASA or TASA in terms of how a set of matching links is established. This simplification makes IRByTSA a low-complexity algorithm. IRByTSA uses a graph maximal matching procedure and adopts the bursty principle during node transmission. For that reason, the algorithm is called the Iman-Ramli Bursty Transmission Scheduling Algorithm (IRByTSA) to emphasize this use of the bursty transmission principle, which distinguishes it from TASA. Another difference between IRByTSA and TASA is that, rather than using the principle of traffic-aware, IRByTSA provides each node with a transmission opportunity based solely on turn, which further contributes to the greater simplicity of IRByTSA. (The traffic awareness principle is explained in Section 2.2.) This paper details IRByTSA procedures, reviewing its lower complexity and the advantages of a simple scheduling algorithm for scalability, so bridging gaps in previous studies. And here is the assumptions and limitations used in this research:

a) Master node knows the network conditions, such as: network topology and its changes, number of packets queued in every node.
b) Each node can synchronize itself to the network and know the transmission or receive schedule based on the information provided by master node.
c) Each node will transmit 1 packet regularly within each slotframe.

The rest of the paper is structured as follows. Section II discusses underlying theories. Section III discuss the proposed algorithm, IRByTSA. Section IV describes the research findings in relation to complexity and the positive outcome of implementation in an IEEE802.15.4e network. And finally, in Section V the paper ends with conclusions.

2. UNDERLYING THEORIES

2.1 IEEE802.15.4e TSCH

IEEE802.15.4 is a standard for low-rate Wireless Personal Area Networks (LRWPAN); IEEE802.15.4e is a redesign of the IEEE802.15.4 MAC protocol that retains the physical layer of that

Journal of Theoretical and Applied Information Technology
31st May 2019. Vol.97. No 10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

standard. Using a Time Synchronized Channel Hopping (TSCH) strategy to improve transmission reliability and energy efficiency, IEE802.15.4e TSCH is suitable for use as part of the Internet of Things protocol stack [17]

To achieve this improved transmission reliability and energy efficiency, IEEE802.15.4e TSCH combines timeslot access method with multi-channel and channel-hopping capabilities. The timeslot access method eliminates collisions among competing nodes, providing deterministic latency to applications that use it. Because there are more active nodes in every timeslot, multi-channel or multi-frequency capability can improve network capacity. In each timeslot, active nodes can transmit data using different frequencies. The channel-hopping approach improves communication reliability by reducing the impact of interference and multipath fading. In short, TSCH can improve network capacity, transmission reliability, and latency while maintaining duty cycle at a low level. TSCH can also be used in different network topologies [18].

As explained above, channel hopping can increase communication reliability because active nodes use different frequencies when transmitting data. In TSCH, the different frequencies are related to the channel offset parameters (ChOf); this is translated into a frequency by using the following function:

$$f = F\{(ASN + chOf) \bmod n_{ch}\} \quad (1)$$

where ASN is Absolute Slot Number (i.e., total timeslots used since network operation is initiated or since a particular point in time set by PAN coordinator), which increases with the passing of timeslots in the network; F is a function based on a look-up table of ready-to-use frequencies; chOF is channel offset; and nch is total available frequencies, where $0 \leqq chOf \leqq (nch-1)$. In an IEEE802.15.4e network, the value of n_{ch} is 16 [17].

2.1.1 Slotframe structure

A slotframe is a group of timeslots that repeat over time; each timeslot allows sufficient time for a pair of devices to transmit data and ACK to each other. The example in Figure 2 shows a slotframe with 6 timeslots. In each timeslot, network nodes can transmit or receive the data or enter a sleep state. Figure 3 shows data and ACK transmission timing within a single timeslot. Based on the 802.15.4e standard, the default timeslot duration is 10 ms [17].

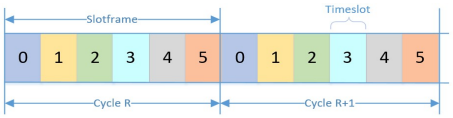


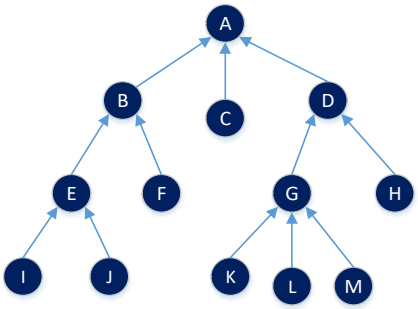
Figure 2: TSCH slotframe with 6 timeslots

Figure 3 shows a node sends a data packet after precisely TsTxOffset µs of the timeslot. To overcome the slight desynchronization, the receiving node must detect the channel Guardtime µs before data are transmitted. If a package is not received within TxRx Wait µs after TsRxOffset, the receiving node will shut down its radio component to save energy[18,19].

Slotframe size (i.e., total timeslots in a slotframe) will determine how frequently a timeslot repeats, which serves as a form of timing for nodes when communicating. The smaller the slotframe size, the more frequently nodes potentially send data; as a consequence, the duty cycle will increase. There is no standard slotframe size, as this depends on the application; a slotframe may range from 10 ts to 1000 ts [20]. In network activity, each node follows a preset schedule that specifies when it should transmit or receive data or sleep. To save energy, nodes shut down their radio (the most energy-consuming element) on entering sleep mode [17].

2.1.2 Link-scheduling

The main element of TSCH is link-scheduling—that is, allocation of certain links to each node for sending or receiving data. According to the IEEE802.15.4e standard, a link is "The pairwise assignment of a directed communication between devices in a given timeslot on a given channelOffset." In TSCH networks, a scheduling scheme is crucial for communication success and must be created carefully; for example, when node E has a transmit slot to node B, then B is ready to receive the data from node E in the same timeslot. [17].



a. Tree Topology TSCH Network

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5		I → E	L → G		G → D	
4		G → D		M → G		
3		F → B	J → E			D → A
2	K → G			E → B	B → A	
1	E → B	C → A	H → D	D → A		
0	D → A		B → A			
	0	1	2	3	4	5

b.Possible Link-Schedule for Data Aggregation

Figure 4: A Tree Topology Network with Its Possible Link-Schedule

Figure 4 illustrates a simple network in which a node activity schedule is already specified for each timeslot through the master node. The directed graph in the tree topology network point to a link connecting a node to its parent. The slotframe consist of 12 timeslots, and 6 offset channels are used. Each node in the network performs its transmit, receive, and sleep activity on the basis of the assigned schedule. The example assumes that each node will transmit 1 packet regularly within each slotframe; within this slotframe duration, any queued data in each node must be transmitted to the master node A. Based on the schedule in Figure 4(b), node D will be ON to send data to node A at timeslot 0, channel offset 0; timeslot 3, channel offset 1; and timeslot 5, channel offset 3. Next, D will enter ON mode to receive data from node G at: timeslot 1, channel offset 4; node H at timeslot 2, channel offset 1; node G at timeslot 4, channel offset 5. This means that, in other timeslots, node D will enter sleep mode to save energy. To reach master node A, the data in node K will have to queue until timeslot 0, timeslot 1, and timeslot 3 before being transmitted gradually through nodes G and D. At timeslot 0, channel offset 2, node K will send its data to node G, and node G will then send data received from node K to node D at timeslot 1, channel offset 4. Finally, node K data will arrive at master node A using timeslot 3, channel offset 1. This is how the MAC layer executes the link schedule provided by the master node.

While the IEEE802.15.4e standard already specifies how the MAC layer executes a schedule, it does not specify how a schedule is to be established. This can be done using either a centralized or a distributed approach. In a scheduling system that uses a centralized approach, there is a special node that are responsible for building and maintaining a "network schedule", this special node is called master node. Each network's node regularly reports the current conditions to that master node. The reported conditions may relate to node connectivity or the amount of data generated by each node. In a scheduling system that uses a distributed approach, each node can decide for itself which link is used to communicate with neighboring nodes. Because the present research aims to improve the link-scheduling mechanism in TASA, this paper will deepen the centralized approach in generating link-scheduling and how its implementation in the 6TiSCH network [17].

2.2 Traffic Aware Scheduling Algorithm (TASA)

The RFC 7554 document mentions TASA as an alternative scheduling algorithm applicable to IEEE802.15.4.e-based IoT networks. Palattella et al. [11] suggested adding TASA to the IEEE802.15.4e standard, which has not yet specified how scheduling is to be established. The following paragraphs summarize TASA as suggested in [9–11].

In TASA, tree topology networks are modelled by a directed graph G = (V, E). V is a group of N devices, where N = |V|, and V = {n_0, n_1, …, n_{N-1}}. E is a group of links connecting each node (n_i) with its parent (p_i). The role of network coordinator or master node is performed by n_0, and n_i ($1 \le i \le N - 1$) is the i-th generic node in the network. Figure 5 illustrates the graph G.

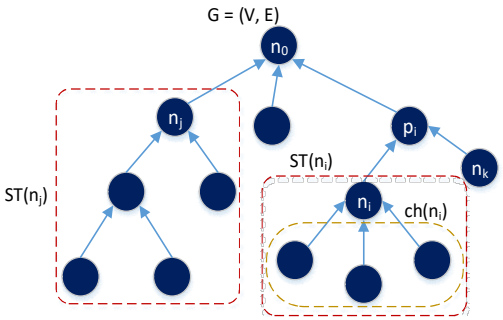


Figure 5: Graph G = (V, E): Modelling a Tree Topology Network

As shown, each n_i (where $\forall n_i \in V$ and $i \neq 0$) can have i) a parent node p_i; ii) a sub-tree ST(n_i), which is structured from n_i itself and all nodes connected to n_i, directly or by means of multihop; and iii) a group of child nodes ch(n_i). Each n_i node is connected to its parent p_i by a dedicated link (n_i, p_i) ∈ E.

In TASA, each n_i node in graph G other than the PAN coordinator (n_0) is assumed to constantly generate packets. Each node n_i will relay its data to parent node p_i until all data have reached the master node, within the duration of 1 slotframe. The total number of packets transmitted to the PAN coordinator in a slotframe is,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$\overline{Q} = \sum_{i=1}^{N-1} \tilde{q}_i \qquad (2)$$

In terms of sub-trees, global queue level $Q_i(k)$ is defined as the total packets queuing in nodes in $ST(n_i)$ at a certain slot k, where

$$Q_i(k) = \sum_{j|n_j \in ST(n_i)} q_j(k) \qquad (3)$$

In TASA, the links to be used for transmitting and receiving data must be free from two types of conflict: Duplex Conflict and Interference Conflict. Duplex Conflict (DC) occurs because each node in a network does not transmit and receive at the same time and cannot receive data simultaneously from its child nodes $ch(n_i)$. DC_i is a group of edges or links between node n_i and its child that cannot be used to transmit or receive during a certain timeslot because the nodes connected by these links are active. Only DC-free links (DCFL) can be scheduled in the same timeslot. A DCFL group in timeslot k is represented by DCFL(k). Interference conflict (IC) refers to links that interfere with one another when placed in the same channel offset. $ICFL_c(k)$ refers to a group of IC-free links (ICFL) that can use the same channel offset c at timeslot k. As this research focuses only on link scheduling, DCFL is the type that will be deepened.

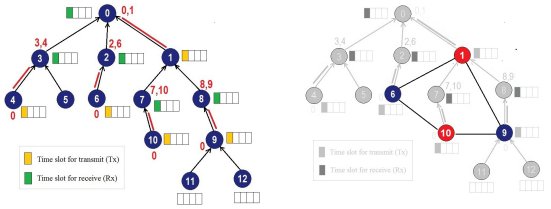
TASA assumes that the master node/PAN coordinator recognizes the G graph, physical connectivity graph P, and the traffic load generated by each node $\forall n_i \in V$, for $i \neq 0$. Based on this information, the master node n_0 will execute the TASA procedure to generate a schedule to be followed by all child nodes in the network.

TASA involves two main procedures: i) matching and ii) coloring. Both procedures are used iteratively on graph G in and for each timeslot (k). A matching procedure is used to obtain DCFL(k) that can be scheduled in timeslot k. In generating DCFL(k), the parent node p_i selects node n_i from among the child nodes, based on the following equation:

$$Q_i(k) = \max\{Q_j(k) \mid n_j \in ch(p_i) \wedge q_j(k) \neq 0\} \qquad (4)$$

In the equation 4, timeslot k is provided for link (n_i, p_i) only when node n_i has a packet to transmit to its parent p_i. For the coloring procedure, TASA applies this to the interference graph I(k). Nodes that are adjacent in I(k) are interfering and therefore require different channel offsets. The matching procedure will be further discussed in the next section because the aim of this research was to propose a new link-scheduling algorithm.

Figure 6 illustrate the application of matching and coloring procedures in a network.



(a) DCFL(k) *(b) ICFL(k)*

Figure 6: Matching and Coloring in TASA [8]

To save power and meet delay requirements, TASA works to give minimum active timeslot for each node. Where active timeslot is represented by the symbol λ, TASA will send a traffic load of $\bar{Q}$ to the PAN coordinator in a λ timeslot duration of a slotframe with S timeslots. Within the remaining duration for (S - λ), all nodes will enter idle/off mode. As Palattella explains, the number of active slots (λ) in a slotframe is minimally $\bar{Q}$. Figure 7 illustrates λ in a slotframe:

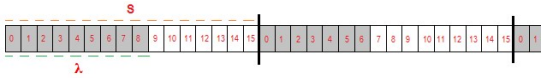


Figure 7: Active Slot (λ) in a Slotframe (S)

2.3 Maximal Matching Algorithm

As explained Section 2.2, in developing the Traffic Aware Scheduling Algorithm (TASA), Pattella et al. [11] has defined Duplex Conflict (DC). To solve the DC problem, Pattella et al. [11] propose to use the graph matching in its methodology, although no particular type of matching is mentioned. Of the existing types of graph matching (maximum, perfect, near perfect, and maximal [21,22]), maximal matching was used in the present research to build the scheduling algorithm. Despite its simplicity, maximal matching has proved effective in meeting the research objective of a low-complexity scheduling algorithm with active timeslot (λ) outputs that are similar to or better than TASA.

The maximal matching algorithm that is used in this research is as follows:

Maximal Matching (G; V; E)

1. M = Ø
2. While(no more edges can be added)
   - 2.1. Starting from the lowest rank of edge, select an edge, e, which has no vertex in common with edge in M
   - 2.2. $M = M \cup e$
3. Return M

G is a directed graph with tree topology as shown in Figure 4(a) above. V and E are the set of vertices and edges in graph G. M is a matching of graph G, which is a subset of the edges E, such that no vertex in V is incident on more than one edge in M.

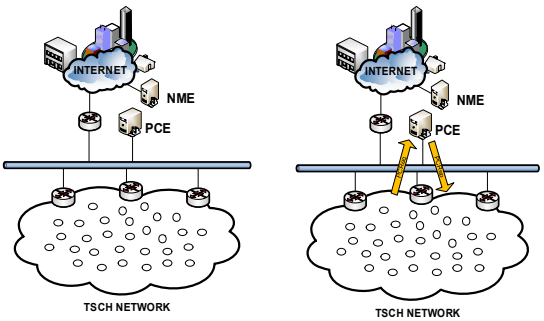
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Combined with procedures in Section 3, this maximal matching algorithm produces the IRByTSA scheduling algorithm.

2.4 Centralized Scheduling in 6TiSCH Networks

In a centralized scheduling system of 6TiSCH network, the Path Computation Element (PCE) takes charge of computing a schedule (see Figure 8(a)). As a management entity (ME), PCE is responsible for generating and maintaining a TSCH schedule based on information from the network nodes concerning network conditions and traffic load. By controlling and optimizing the network as a whole, PCE can develop a schedule and ensure accurate fulfilment of QoS requirements for all traffic flows in the network [4, 6]. To the best of our knowledge, standardization on signaling for IEEE802.15.4e TSCH network has not been established, therefore further research is warranted.

To determine the mechanism and format of control messages to be sent to the network nodes, 6TiSCH WG first defines the requirements for the protocol to be used by PCE to communicate with the 6TiSCH network, acknowledging existing protocols that may (partly) meet those requirements. Path Computation Element Communication Protocol (PCEP) defines the method of establishing communication between the Path Computation Client (PCC) and PCE. PCC can prompt PCE for path computation by means of a PCReq message specifying the relevant protocol requirements. In response to the PCReq message, PCE sends a PCRep that determines whether the path request and its requirements can be met. Hence, PCEP is capable of carrying signaling messages from node (PCC) to PCE, which contains network nodes' scheduling requirements [4]. To maintain scalability and throughput, a 6TiSCH network may have a number of cluster networks linked to a high-speed backbone [6]. Based on working draft [7] and earlier research [4], Figure 8 illustrates this as follows.



(a) A 6TiSCH network (b) Signaling in a 6TiSCH network

Figure 8: Centralized Scheduling in 6TiSCH Network

3. PROPOSED ALGORITHM

IRByTSA uses the following procedures:

1) Each node in the network (other than leaf nodes and master node) may act as parent and child. Each parent node supervises one or several child nodes, and each node always has one parent node. All nodes send their data to the master node (node 0 or n0) through their parents. Each node is connected to its parent by a link that marks the transmit and receive relationship between a child node and its parent. The link between node i (n_i) and its parent node j (n_j) is represented by l_{ij}.
2) As described in [9–11], a node in a network cannot transmit and receive simultaneously or receive data from multiple nodes at once, and a gradual transmitting and receiving process is therefore required in order to send all packets from network nodes to the master node. All nodes in the network transmit/receive data simultaneously based on the transmit/receive schedule specified by the master node. The transmit/receive process occurs in several rounds until all data from the nodes reach the master (n_0) and the condition of $q_0 = \bar{Q}$ is met, where q_0 is queue size of node 0 and $\bar{Q}$ is total packets queued to be sent to n_0 within a single slotframe; that is, $\bar{Q} = \Sigma q_i$. The symbol s represents the round or step, and SS denotes the total number of steps needed to reach the condition $q_0 = \bar{Q}$. The transmit/receive schedule for each node is determined on the graph theory principle of maximal matching.
3) The nodes having their turn in this scheduling algorithm are collected in a set which is called an **Active** set.
4) Child nodes send data to their parent in turn. If, in a certain round, there are no data to send when the node has its turn, the turn passes to a sibling that has data to send. Transmission begins with high-level nodes (those closer to the master node) and proceeds to lower-level nodes (farther from the master node) as far as the lowest-level node (leaf node).
5) During its turn, each node performs bursty transmission—that is, queued data are sent all at once. Because of this bursty characteristic, this type of algorithm is called bursty transmission scheduling algorithm in this research.
6) The number of timeslots needed by each node equals the number of packets to be sent because IEEE 802.15.4e TSCH requires 1 timeslot to send 1 packet. Where ts_i represents the total

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

timeslots needed by each ni to transmit the data, $ts_i(s)$ represents the total timeslots needed by n_i to transmit data in step/round s. In the present research, the value of $ts_i(s)$ equals to $q_i(s)$, where $q_i(s)$ is the number of packets queued in n_i in a given round s. This is expressed as:

$$ts_i(s) = q_i(s) \quad (5)$$

7) Total active timeslots (λ) needed for one frame is determined by the equation:

$$\lambda = \sum_{s=1}^{SS} TS(s) \quad (6)$$

where TS(s) is the number of timeslots needed in each round/step (s). TS(s) is obtained using the following equation:

$$TS(s) = \max\{ts_i(s) | i \epsilon Active\} \quad (7)$$

The above procedures are illustrated in Figure 9, showing initial condition, gradual transmission of packets to the master node, and the 8 rounds/steps required to transmit all queued data in each node to the master node, from first to eighth scheduling round. Packets are represented by the red numbers, each referring to the node number. Figure 9 also illustrates how the schedule is arranged by maximal matching. At each step, no single node gets both transmitting and receiving turns at the same time. At any given point in time, each node in the network can be in only one state: transmitting, receiving, or idle/sleep. The maximal matching procedure also ensures that the maximal number of nodes transmit or receive data in each step/round.

The initial condition is shown in more detail in Figure 10, where the master node or node 0 is the highest level. As shown in Figure 10 below, each node has one packet queuing for transmission to the master node. All the data queued in nodes 1 to 15 are transmitted to node 0 through their respective parents.

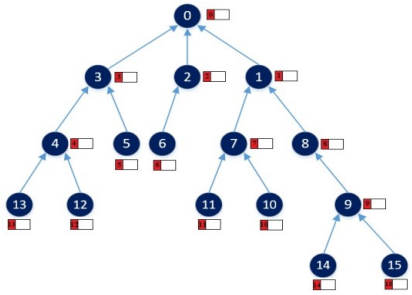


Figure 10: Network Topology with Packet Queue.

IRByTSA procedure number 3 is designed to minimize the number of active timeslots (λ) needed in one frame. The smaller the λ value needed in a frame, the more time nodes remain in an idle state, during which their total energy consumption is smaller than in an active state. Because of the limitations of battery power [19], energy consumption is an important consideration for zigbee-based technology.

As explained below, procedure 4 yields a λ value that approximates the minimum number of timeslots (Q) based on the following two factors:

a) The procedure ensures that there is no queue build-up in any node because the Qi value of each child of the master node is regularly reduced by providing a window for data transmission to its parent. Queue build-up at any node makes it impossible to minimize TS(s) because TS(s) always targets the possible maximum value of each data queue. Where TS(s) cannot be minimized, it is impossible to obtain a λ value that approximates $\bar{Q}$, which is the lower limit of λ [11].

b) As a result of this procedure, a packet is always sent to the master node, and the Qi value of each master node's children will be reduced in each cycle of the schedule. A decrease in Qi signifies the transmission of data stored in the network node to the master node through the master node's children. In this way, the Qi value of the entire network will become smaller as the round increases, in time resulting in the gradual decrease of TS(s).

Queue build-up at any node can result in an increase in TS(s) because (based on procedures 6 and 7) the IRByTSA algorithm's TS(s) value takes the highest tsi(s) value among the active nodes in a certain s round. TS(s) must equal the highest tsi(s) if data to be transmitted in a session/round is to be sent to the parent; if TS(s) is smaller than the data queued at even one node, this will prevent transmission of some data as a result of inadequate timeslot allocation. Based on Figure 9, the number of active timeslots needed in a slotframe is elaborated below, using three equations: eq. 5, eq. 6 and eq. 7. Based on eq. 6 and 7, a new equation can be derived:

$$TS(s) = \max\{q_i(s) \mid i \epsilon Active\} \quad (8)$$

The steps to the active timeslot (λ) are as follows:

$$\lambda = \sum_{s=1}^{8} TS(s) = TS(1) + TS(2) + TS(3) + TS(4) + TS(5) + TS(6) + TS(7) + TS(8)$$

$$TS(1) = \max\{q_i(1) \mid i \epsilon Active\}$$

$$= \max\{q_1(1), q_4(1), q_6(1), q_9(1), q_{10}(1)\}$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$= \max\{1, 1, 1, 1, 1\} = 1$ timeslot

$TS(2) = \max\{q_i(2) \mid i \in Active\}$

$= \max\{q_2(2), q_5(2), q_7(2), q_{12}(2), q_{14}(2)\}$

$= \max\{2, 1, 2, 1, 1\} = 2$ timeslot

$TS(3) = \max\{q_i(3) \mid i \in Active\}$

$= \max\{q_3(3), q_8(3), q_{11}(3), q_{13}(3)\}$

$= \max\{3, 2, 1, 1\} = 3$ timeslot

$TS(4) = \max\{q_i(4) \mid i \in Active\}$

$= \max\{q_1(4), q_4(4), q_9(4)\}$

$= \max\{4, 2, 2\} = 4$ timeslot

$TS(5) = \max\{q_i(5) \mid i \in Active\}$

$= \max\{q_3(5), q_7(5)\}$

$= \max\{2, 1\} = 2$ timeslot

$TS(6) = \max\{q_i(6) \mid i \in Active\} = \max\{q_1(6)\}$

$= \max\{1\} = 1$ timeslot

$TS(7) = \max\{q_i(7) \mid i \in Active\}$

$= \max\{q_8(7)\} = \max\{2\} = 2$ timeslot

$TS(8) = \max\{q_i(8) \mid i \in Active\}$

$= \max\{q_1(8)\} = \max\{2\} = 2$ timeslot

$\lambda = \Sigma TS(s) = 1 + 2 + 3 + 4 + 2 + 1 + 2 + 2$

$= 17$ timeslot

Procedure number 5 explain that each node performs bursty transmission, to better understand the busty principle, Figure 11 focuses in more detail on the initial condition, round 1, and round 2 as shown in Figure 9.

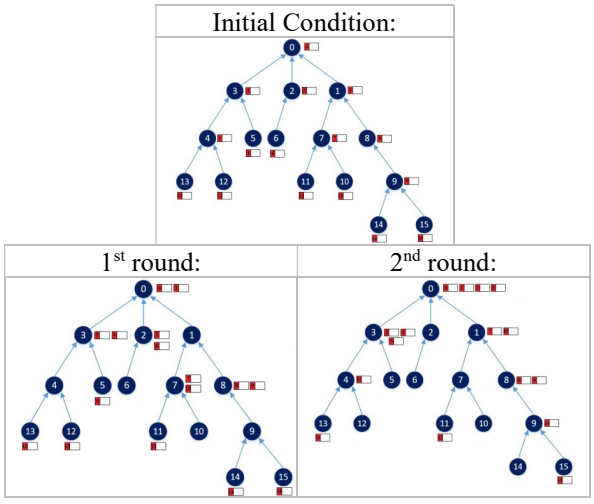


Figure 11: Bursty transmission in IRByTSA

At the initial condition, all nodes each have only 1 packet to transmit to the master node (in this case, node 0). As shown in Figure 11, in every scheduled round, several nodes are waiting their turn to transmit data. When transmitting, nodes send all queued data. When all data have been transmitted, there is nothing more in the queue. This procedure is referred to here as *bursty transmission.*

Table 1: Symbols used in IRByTSA

Variables	Description
ND	Total number of network nodes
NCD[i]	Total number of child nodes of node [i]
TS(s)	Number of timeslot needed in each round
$\bar{Q}$	Total number of packets queued to be sent to master node within a single slotframe
i	Number of Node
y	Turn (variable)
s	Round
q_z	Queue size of node z
turn_now[i]	Transmission turn for one child of node i in a specific round
child_of_node[i][j]	Specific child of node[i]
mod_node[i]	Transmission turn between child nodes of node [i]

All of the above procedures are implemented in IRByTSA algorithm, as shown in Figure 12. The algorithm uses several key variables, which are described in Table 1.

4. RESULT AND DISCUSSION

4.1 Complexity Analysis of IRByTSA

Looking at Figure 12, we can see that there are two main loops, which we will call L_1 and L_2. During each iteration, there is a chain of commands ready to execute; L_1 is an external loop while L_2 is an internal loop. To determine the complexity of asymptotic time in IRByTSA, consider the command lines for L_1 and L_2. Based on the references [24–27], the algorithm's complexity, T(n), can be derived using the following formulas.

If, $Tx(n) = O(f(n))$ and $Ty(n) = O(g(n))$, then:

a) $T(n) = Tx(n)+Ty(n) = O(\max(f(n),g(n)))$, or
$T(n) = Tx(n)+Ty(n) = O(f(n)+g(n))$

b) $T(n) = Tx(n).Ty(n) = O(f(n)).O(g(n))$
$= O(f(n)g(n))$

c) If $g(n) \leqq f(n)$ for all $n \geqq n0$ (for a specific n_0 value), therefore $O(f)+O(g) = O(f)$

d) $T(n) = O(cf(n)) = O(f(n))$ where c is a constant

e) $f(n) = O(f(n))$

O symbol is called Big-O notation, this notation is used extensively to express the upper limit for the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

operation of an algorithm as its input grows [20]. Based on the above, L_2's algorithmic complexity is:

$$\begin{aligned}T_2(n) &= n\,.\,\{O(1) + O(1) + \max[O(1)+O(1)+O(1) \\ &\quad + O(1),O(1)+O(1)] + O(1) + O(1)\} \\ &= n\,.\,\{O(\max(1,1))+\max[O(\max(1,1)) \\ &\quad + \\ &O(\max(1,1)),O(\max(1,1))]+O(\max(1,1))\} \\ &= n\,.\,\{O(1)+\max[O(1)+O(1),O(1)]+O(1)\} \\ &= n\,.\,\{O(1)+\max[O(\max(1,1)),O(1)]+O(1)\} \\ &= n\,.\,\{O(1)+\max[O(1),O(1)]+O(1)\} \\ &= n\,.\,\{O(1)+O(1)+O(1)\} \\ &= n\,.\{O(\max(1,1))+O(1)\} \\ &= n\,.\,\{O(1)+O(1)\} = n\,.\,\{O(\max(1,1))\} \\ &= n\,.\,\{O(1)\} = O(n\,.\,1) \\ &= O(n) \end{aligned} \quad (9)$$

For loop L_1, the number of iterations/rounds in the loop is determined by whether $q_0 < \bar{Q}$; iteration stops where $q_0 \geqq \bar{Q}$. In IRByTSA, the total iterations/rounds needed in L_1 to ensure that q_0 equals $\bar{Q}$ is smaller than in TASA because in IRByTSA, scheduling is not done in and for every time slot period but should be done during the transmission turn for each child node, beginning with children of the master node (n_0). It follows that the number of iterations needed to generate a schedule is not directly correlated to the number of nodes in each network. As for TASA, according to [11], the minimum number of active timeslots needed (λ) is equal to $\bar{Q}$. And according to the limitation in this research, the number of iterations in TASA will always be the same as the number of network nodes. As for IRByTSA, the number of iterations needed to generate a schedule is always smaller than the total number of nodes.

Table 2 compares TASA and IRByTSA in terms of the number of rounds/steps (SS) needed to send all of the network's data to the master node (n_0) and the size of active timeslot (λ). The scheduling algorithm shows good performance if a minimum λ can be achieved with a minimum number of rounds/steps (SS).

Table 2. Comparison of TASA and IRByTSA

Number of Nodes (n)	TASA		IRByTSA		$R = \frac{SS_{TASA}}{SS_{IRByTSA}}$
	SS_{TASA}	λ	$SS_{IRByTSA}$	λ	
10	10	10	4	9	2.50
20	20	20	7	19	2.86
30	30	30	9	29	3.33
40	40	40	11	40	3.64
50	50	50	11	51	4.55
60	60	60	13	60	4.62
70	70	70	12	70	5.83
80	80	80	13	80	6.15
90	90	90	14	90	6.43
100	100	100	14	100	7.14

In TASA, the number of iterations increases linearly with network nodes. In IRByTSA, with a λ value very close to TASA, the increase in number of iterations is relatively small for 10 to 100 nodes. The SS value also shows the speed of each scheduling algorithm in creating scheduling decisions. The smaller the SS value, the more rapidly the scheduling decision is generated, and vice versa. As seen in Table 2, $SS_{IRByTSA} < SS_{TASA}$ for all network sizes ranging from n = 10 to n = 100; this finding indicates that IRByTSA is always faster than TASA in generating scheduling decisions. The R value indicates the difference in speed of the two algorithms; that is, it compares SS_{TASA} and $SS_{IRByTSA}$. As shown in Table 2, for n = 10 to n = 100, the value of R is always greater than one; thus, IRByTSA is always faster than TASA.

As explained above, the SS column in Table 2 also indicates the number of iterations needed by IRByTSA and TASA to generate a transmit/receive schedule for each node in order to obtain a $q_0 = \bar{Q}$ condition. Table 3 shows the relation between $NI_{IRByTSA}$ and number of network nodes (n), where the relation is symbolized with X. With the X symbol, relationship between $NI_{IRByTSA}$ and n can be stated as follows: $NI_{IRByTSA} = n^X$.

Table 3: Number of iterations (NI) by number of network nodes (n)

Number of Nodes (n)	Number of Iterations for L_1 of IRByTSA ($NI_{IRByTSA}$)	Relation between $NI_{IRByTSA}$ and n $X = \left(\log NI_{IRByTSA}(n) / \log n\right)$
10	4	0.60206
20	7	0.64956
30	9	0.64602
40	11	0.65003
50	11	0.61296
60	13	0.62646
70	12	0.58489
80	13	0.58533
90	14	0.58648
100	14	0.57306

Table 4: Values for Several Relations

Number of Nodes (n)	$NI_{Linier}(n)$	$NI_{IRByTSA}(n)$	$NI_{RoundUp}(n)$
10	10	4	5
20	20	7	8
30	30	9	10
40	40	11	11
50	50	11	13
60	60	13	15
70	70	12	16
80	80	13	18
90	90	14	19
100	100	14	20

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Furthermore, Table 4 shows the values for several relations: $NI_{Linier}(n) = n$, $NI_{IRByTSA}(n)$, and $NI_{RoundUp}(n) = O(\lceil n^{0.65} \rceil . n)$. The $NI_{RoundUp}(n)$ function is chosen to present the upper limit for the L_1 iteration which will not be passed by the actual L_1 iteration ($NI_{IRByTSA}$). It is important to use $NI_{RoundUp}$ function in the present context because this research aims to determine the complexity of IRByTSA mathematically which is closest to the real conditions.

As shown in Table 4, the number of iterations/rounds in L_1 before achieving $q_0 = \bar{Q}$ for IRByTSA ($NI_{IRByTSA}$) is never greater than $\lceil n^{0.65} \rceil$. Figure 13 illustrates this more clearly.

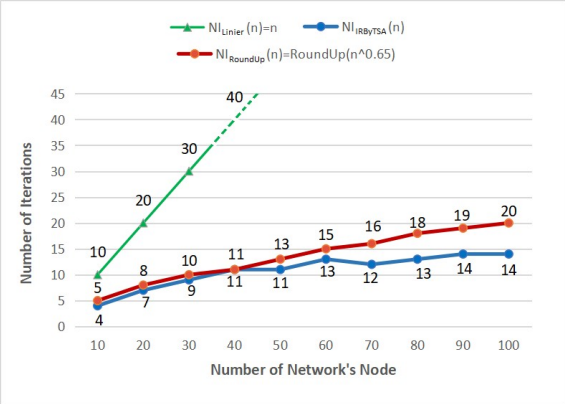


Figure 13: Values for Several Relations

Figure 13 shows a $NI_{Linier}(n)$ graph (colored green), representing cases where the number of iterations in L_1 is equal to the number of network nodes. The $NI_{IRByTSA}(n)$ graph (blue) shows the number of L_1 iterations on IRByTSA ($NI_{IRByTSA}$), which is clearly much smaller than $NI_{Linier}(n)$. The red graph represents the $NI_{RoundUp}$ (n) function where $NI_{RoundUp}(n) = \lceil n^{0.65} \rceil$ and shows that the value of $\lceil n^{0.65} \rceil$ is very close to $NI_{IRByTSA}(n)$.

Because iteration L_2 falls within L_1, where there are no more lines of commands to execute other than those in L_2, the combined complexity of L_1 and L_2 is:

$$T(n) = \lceil n^{0.65} \rceil . O(n) = O(\lceil n^{0.65} \rceil) . O(n) = O(\lceil n^{0.65} \rceil . n) \quad (10)$$

Equation (10) above shows that the complexity for IRByTSA is $O(\lceil n^{0.65} \rceil . n)$.

Referring to [24], the complexity of IRByTSA can also be expressed as O(n.n) or O (n^2), as the $NI_{IRByTSA}(n)$ graph never exceeds $NI_{Linier}(n)$. An algorithm in the range O(1) to O(n^3) can be categorized as a good polynomial algorithm because it is tractable (easy in computational terms) [24]. Because the complexity of IRByTSA is $O(\lceil n^{0.65} \rceil . n)$, therefore IRByTSA can be categorized as a good algorithm.

4.2 The Relationship Between Scheduling Algorithm Performance and Network Scalability

As stated in the Introduction, the two aspects of scheduling algorithms studied in this research are algorithm complexity and speed in generating scheduling decisions. A scheduling algorithm that rapidly generates scheduling decisions can be ascertained to impose a lighter burden on the server compared with slower algorithms (The server referred to in this 6TiSCH network is PCE.). Thus, IRByTSA, which is faster than TASA, will positively impact PCE performance.

Regarding the complexity of scheduling algorithms, research [28] showed experimentally that low-complexity scheduling algorithm will use less PCE resource. That research concludes that three PCE modules are used to process path computation: Network, Session Management, and Processing. These three modules involve five processes: network read, session processing, computer request queuing, computation processing, and response sending. Of the three modules, processing is most affected by algorithm complexity. As shown in [28], processing time in the computation module increases for the three algorithms according to increased network size. The computational complexity of the three algorithms is O(V. ln(E)), O(2V.ln(E)), and O(V^2), respectively. The increase in processing time is more significant for the algorithm with O(V^2) complexity as compared to the other two. The algorithm with O(V.ln(E)) complexity has the same complexity as IRByTSA—that is, O(K.n), where K is a constant.

A low-complexity computational algorithm such as IRByTSA algorithm allows the existing PCE (usually used for algorithms of high complexity) to serve more network clusters. The simpler the computational algorithm used in a PCE server, the more network clusters can be served by the existing PCE. The number of network clusters that can be served by a PCE server depends on the capacity of the PCE server. Based on Figure 8, Figure 14 illustrates the scalability of the 6TiSCH network when PCE is enabled to serve multiple network clusters.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

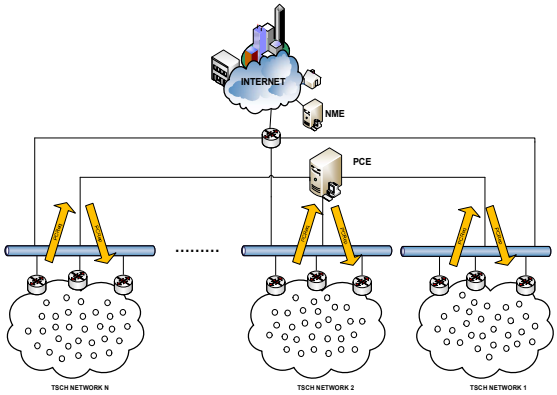


Figure 14: One PCE Server Serves Many Network Clusters

"According to the data presented in paper [28], finishing 1,500 path computation requests using an algorithm with O(K.n) complexity needs an average time of 0.4 s. In TSCH-based IoT, document RFC 7554 [20] mentions that the number of timeslots per slotframe is between 10 and 1,000, and the duration per timeslot is 10 ms. In a TSCH network, the number of timeslots correlates with the number of nodes. If the total timeslots per slotframe is between 10 and 1,000, then the maximum number of nodes in the network is between 10 and 1,000. Accordingly, the time needed to handle a path computation request in a network smaller than 1,500 will need less than 0.4 s. For example, assume that a network administrator has a network with 500 nodes with a slotframe of 1,000 timeslots (S = 1000) and a duty cycle of 50% (λ = 500 timeslots). Given these parameters, each slotframe will have a remaining duration time of 500 timeslots (S - λ = 500 timeslots) or 5 s for nodes to enter an idle state. This 5 s duration is considerably longer than the 0.4 s needed to complete a path computation request from 1,500 network nodes. Thus, for a network with a duty cycle ≤ 50% and low scheduling algorithm complexity, this research suggests the research prospect of examining the feasibility of completing the signaling process between nodes and PCE within the duration of (S - λ). If it is feasible, then the signaling process will not need to be allocated a special duration. Signaling must be simplified to reduce signaling cost, as discussed in research [16]."

5. CONCLUSION

This research succeeded in building a centralized scheduling algorithm called IRByTSA, which is much lower in complexity than IR-TASA and faster than TASA in terms of generating link-scheduling decisions. The algorithm's low computational complexity can be attributed to its use of procedures such as maximal matching process to generate link-schedule; maximizing the number of nodes that transmit simultaneously by proceeding from the highest-ranked node and continuing downward to the leaf node; and on a transmit turn, each node sends all of its queued data in bursty mode. Using these procedures, IRByTSA generates link scheduling decisions more quickly than TASA and has low complexity. The former algorithm creates such decisions 2.5 to 7.14 times more rapidly than the latter for network sizes ranging from n = 10 to n = 100. The calculations show that IRByTSA has a complexity level of $O(\lceil n^{0.65} \rceil . n)$, which indicates low complexity. The advantage of using such a fast and low-complexity algorithm is increased network scalability, as the reduction in complexity and increase in speed enable the existing PCE to serve additional networks. Meanwhile, as explained in Section 2.4, standardization on signaling for IEEE802.15.4e TSCH networks has not been established. Thus, this research will focus on the signaling aspects of IEEE802.15.4e TSCH networks in the future.

REFERENCES:

[1] Atzori, L., Iera, A., Morabito, G. "Understanding the Internet of Things: Definition, potentials, and societal role of a fast evolving paradigm", *Ad Hoc Networks*, 2017, 56: 122–140.

[2] Li, S., Da Xu, L., Zhao, S. "The internet of things: A survey", *Information Systems Frontiers*, 2015, 17.2: 243–259.

[3] Gubbi, J., et al. "Internet of Things (IoT): A vision, architectural elements, and future directions", *Future generation computer systems*, 2013, 29.7: 1645–1660.

[4] Palattella, M. R., et al. "6tisch wireless industrial networks: Determinism meets ipv6", *Internet of Things* (pp. 111–141). Springer, Cham, 2014.

[5] Thubert, P., Palattella, M. R., Engel, T. "6TiSCH centralized scheduling: When SDN meet IoT", *2015 IEEE Conference on Standards for Communications and Networking (CSCN),* IEEE, 2015, pp. 42–47.

[6] Dujovne, D., et al. "6TiSCH: Deterministic IP-enabled industrial internet (of things)", *IEEE Communications Magazine*, 2014, 52.12: 36–41.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[7] Thubert, P. "An architecture for IPv6 over the TSCH mode of IEEE 802.15. 4. Working Draft", *IETF Secretariat*, Internet-Draft draft-ietf-6tisch-architecture-14. 2018 (April).

[8] Santoso, I.H., Ramli, K. :Speed improvement of centralized scheduling for IEEE 802.15.4e TSCH network", *Journal of Communications*, 2017, 12.12: 661–667, DOI: 10.12720/jcm.12.12.661-667.

[9] Palattella, M. R., et al. "Traffic aware scheduling algorithm for reliable low-power multi-hop IEEE 802.15. 4e networks", *2012 IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC),* IEEE, 2012, pp. 327–332.

[10] Palattella, M. R., et al. "Traffic-aware time-critical scheduling in heavily duty-cycled ieee 802.15. 4e for an industrial iot", *Proceedings of IEEE Sensors 2012*, 2012, pp. 1–4.

[11] Palattella, M. R., et al. "On optimal scheduling in duty-cycled industrial IoT applications using IEEE802. 15.4 e TSCH", *IEEE Sensors Journal*, 2013, 13.10: 3655–3666.

[12] Farias, A. A., Dujovne, D. "A queue-based scheduling algorithm for PCE-enabled Industrial Internet of Things networks", In: *2015 Sixth Argentine Conference on Embedded Systems (CASE),* IEEE, 2015. pp. 31–36.

[13] Min, M., et al. "Traffic aware multiple slotframes scheduling algorithm in industrial IoT applications using IEEE802. 15.4 e TSCH", In: *2015 IEEE 16th International Conference on Communication Technology (ICCT),* IEEE, 2015, pp. 608–614.

[14] Gaillard, G., et al. "Enabling flow-level reliability on FTDMA schedules with efficient hop-by-hop over-provisioning*",* PhD Thesis, INRIA Grenoble-Rhône-Alpes, 2016.

[15] Choi, K. H.; Chung, S. H. "A new centralized link scheduling for 6TiSCH wireless industrial networks", *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*. Springer, Cham, 2016, pp. 360–371.

[16] Livolant, E., Minet, P., Watteyne, T. "The cost of installing a 6tisch schedule", *International Conference on Ad-Hoc Networks and Wireless*. Springer, Cham, 2016, pp. 17–31.

[17] Palattella, M. R., et al. "Standardized protocol stack for the internet of (important) things", *IEEE communications surveys & tutorials 15.3*, 2013, pp. 1389-1406.

[18] De Guglielmo, D., Simone Brienza, and Giuseppe Anastasi. "IEEE 802.15. 4e: A survey", *Computer Communications, 88*, 2016. pp. 1-24.

[19] 802.15.4e-2012: IEEE Standard for Local and Metropolitan Area Networks – Part 15.4: Low-Rate Wireless Personal Area Networks (LRWPANs) Amendment 1: MAC Sublayer, *Institute of Electrical and Electronics Engineers Std*., 16 April 2012.

[20] Watteyne, T., Palattella, M. R., Grieco, L. "Using IEEE 802.15. 4e time-slotted channel hopping (TSCH) in the internet of things (IoT): Problem statement". 2015.

[21] West, D. B. "Introduction to graph theory (Vol. 2)", *Prentice Hall*, Upper Saddle River, 2001.

[22] Winter, 2005, "Maximum matching", CS105. [Online] https://www.cs.dartmouth.edu/~ac/Teach/CS105-Winter05/Notes/kavathekar-scribe.pdf

[23] Sahoo, P., Pattanaik, S., Wu, S. L. "A reliable data transmission model for IEEE 802.15. 4e enabled wireless sensor network under wifi interference", *Sensors*, 2017, 17.6: 1320.

[24] Rosen, K. H. "Discrete mathematics and its applications", McGraw-Hill, New York, 2011.

[25] Azmoodeh, M. "*A*bstract Data Types and Algorithm", Macmillan, London, 1988.

[26] Brassard, G., Bratley, P. "Algorithmics: Theory & practice", Prentice-Hall, 1988.

[27] Zindros, D. "A gentle introduction to algorithm complexity analysis", 2012.

[28] Chamania, M., Drogon, M., Jukan, A. "An open-source path computation element (PCE) emulator: Design, implementation, and performance", *Journal of Lightwave Technology*, 2012, 30.4: 414–426.

Journal of Theoretical and Applied Information Technology
31st May 2019. Vol.97. No 10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

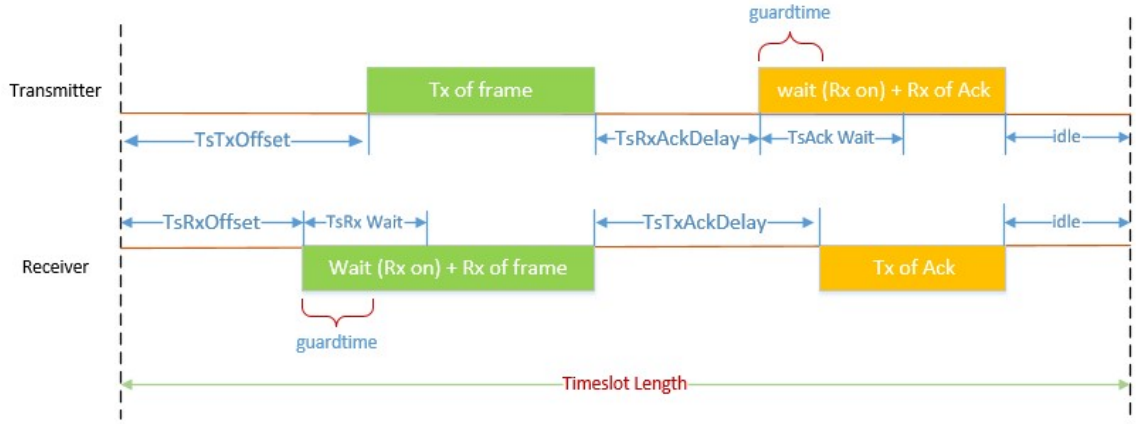


Figure 3: Data and Ack Transmission in a Timeslot

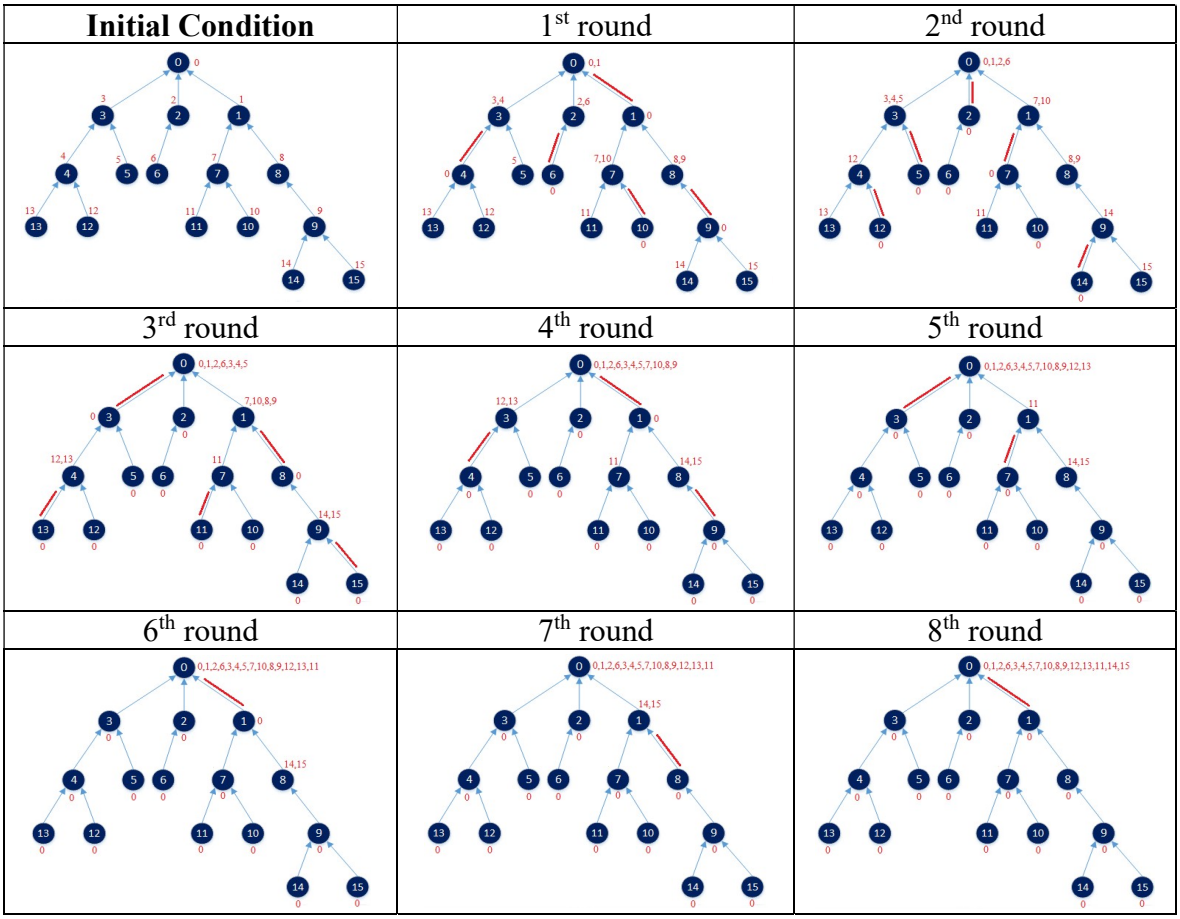


Figure 9: Transmission and Reception Scheduling using IRByTSA

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

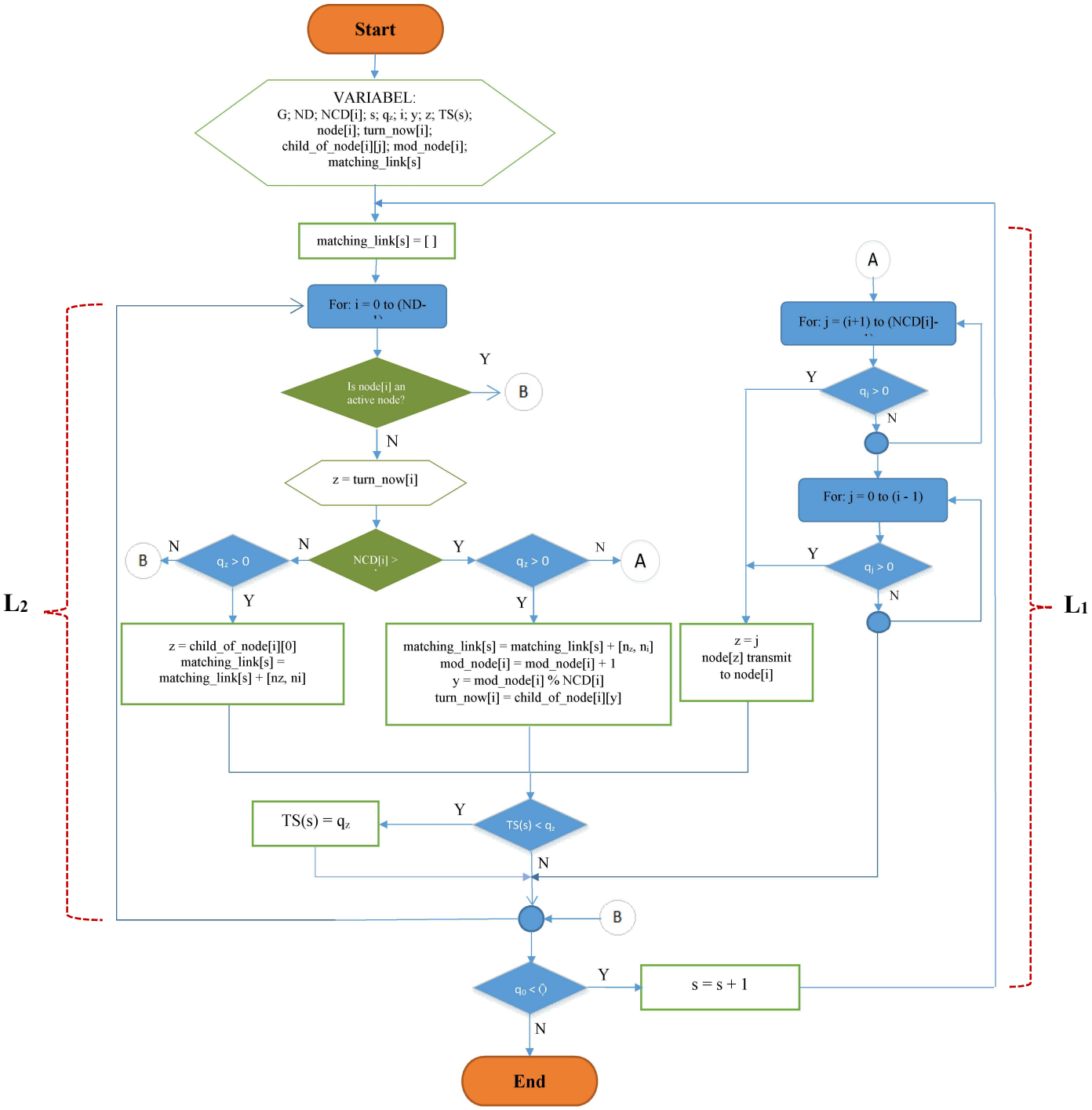


Figure 12: Flowchart of IRByTSA algorithm