SDORP: SDN based Opportunistic Routing for Asynchronous Wireless Sensor Networks

Muhammad Umar Farooq, Xingfu Wang*, *Member*, *IEEE*, Ammar Hawbani*, Liang Zhao, *Member*, *IEEE*
Ahmed Al-Dubai, *Senior Member*, *IEEE*, Omar Busaileh

Abstract—In wireless sensor networks (WSNs), it is inappropriate to use conventional unicast routing due to the broadcast storm problem and spatial diversity of communication links. Opportunistic Routing (OR) benefits the low duty-cycled WSNs by prioritizing the multiple candidates for each node instead of selecting one node as in conventional unicast routing. OR reduces the sender waiting time, but it also suffers from the duplicate packets problem due to multiple candidates waking up simultaneously. The number of candidates should be restricted to counterbalance between the sender waiting time and duplicate packets. In this paper, software-defined networking (SDN) is adapted for the flexible management of WSNs by allowing the decoupling of the control plane from the sensor nodes. This study presents an SDN based load balanced opportunistic routing for duty-cycled WSNs that addresses two parts. First, the candidates are computed and controlled in the control plane. Second, the metric used to prioritize the candidates considers the average of three probability distributions, namely transmission distance distribution, expected number of hops distribution and residual energy distribution so that more traffic is guided through the nodes with higher priority. Simulation results show that our proposed protocol can significantly improve the network lifetime, routing efficiency, energy consumption, sender waiting time and duplicate packets as compared with the benchmarks.

Index Terms—Asynchronous wireless sensor networks, opportunistic routing, load balancing, duty-cycle.

1 Introduction

Traditional wireless sensor networks (WSNs) consist of a large number of randomly deployed nodes with limited sensing, computing, and wireless communication capabilities [1], [2]. Since the network design is built by physical resources, thus, designing of efficient routing approach is a fundamental step towards achieving the least consumption of resources [3], [4]. Traditional WSNs are believed to be specified application, which makes it extremely hard to respond to events and reconfiguring high-level policies across the network. It is, therefore, necessary to specify these high-level policies in terms of distributed low-level configuration [5], [6]. Although many difficulties do exist such as sleep mechanisms or various types of data fusion which could be result of unstable links or large amounts of energy waste that cannot be solved in ossified network architecture. Various mechanisms need to be used as part of a large scale WSN management to facilitate maintainability and system self-healing. When the energy resource becomes insufficient, the system should be able to adjust the parameters on the basis of factors such as reduction in service quality. In the existing WSN configuration, this is a challenge where all the data and control packets are routed across the limited network band.

It is for this reason the design of a network management system is a challenging task, especially on a large-scale network which is typically seen as the second phase of project planning. In general, many problems inherent to traditional WSNs are deeply rooted in the network architecture. Each node is manufactured as an autonomous system to accommodate all the functionalities from the physical layer up to the application layer that executes both the network control and data forwarding. This architecture works well, particularly with small scale short-range WSNs due to well-designed algorithms. However, it lacks flexibility and simplicity which makes it difficult to manage when trying to implement a large scale with low power and long-range WSNs. Several recent techniques have been studied and implemented to solve this problem in the management of WSNs, in which some of the techniques revolve around the idea of *Software Defined Networking* (SDN) [7], [8].

The SDN architecture enhances the network intelligence by separating the control plane from the data plane. The separation of planes provides flexibility and efficiency in management and vendor independence (due to the standardized interface). It is because of this that nodes in the data plane only perform the data forwarding without doing the network control functionalities like routing strategies, topology management, and increasing their energy efficiency etc. On the other hand, the control plane performs the network control functionalities like scheduling, routing rules, comprehensive duty cycling, and topology management etc., armed with the network-wide view [9]. In addition, it prevents WSNs from being an application specific where the data plane virtually supports all the routing rules and the control plane distinguishes the application layer

- *Muhammad. U. F. (muhammad@mail.ustc.edu.cn), Xingfu W., Ammar H. {wangxfu, anmande} @ustc.edu.cn, and Omar B. (busaileh@mail.ustc.edu.cn) are with School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China; * Corresponding author*
- *Liang Z. (lzhao@sau.edu.cn); Shenyang aerospace University ; Shenyang, 110136, China;*
- *A.Al-Dubai is with School of Computing; Edinburgh Napier University, UK; (a.al-dubai@napier.ac.uk);*

from physical layer to support various applications with different hardware to run within the same architecture of the physical network. Moreover, it also prevents WSNs from being vendor specific by empowering multiple vendors to design non-isolated application-customizable devices by sharing similar features. Apart from accelerating the innovation in protocols, production, and prototyping, this evades the complex network management caused by multi-vendor node deployment which shows the network intelligently controlled in a vendor independent manner [10].

SDN managed the frequent change of functionalities and improved a centralized view within the network. Consequently, it offers flexibility, scalability and centralized control to attain the better network efficiency. Conversely, it requires provision of duty cycling in WSNs because the energy consumption of the sensor node is an essential problem to be solved in this technology where sensors do not take measurements.

Duty cycling in WSNs is one of the primary mechanisms used for minimizing energy consumption [11]. In this approach, the nodes switched between active/sleep states based on the predefined active and sleep intervals. There have been numerous sleep schedule mechanisms for the *Medium Access Control* (MAC) layer [1], [2], [11], [12], [13], that help to minimize energy consumption by putting nodes into a sleep state when there are no data packets to be sent or received. These mechanisms on sleep schedule coordinates by one of the two approaches, namely synchronous and asynchronous. In the synchronous MAC protocols, nodes are synchronized with the wakeup schedule of other nodes because at the same time all of them are active to exchange the data packets. However, maintaining time of synchronization can be expensive in terms of a high control packet overhead. It also consumes too much energy which is intolerable in the large-scale energy constrained network. On the contrary, asynchronous MAC protocols do not require the periodic time of synchronization between the nodes to wake up at the same time. In fact, the nodes randomly switched between active/sleep modes in the network which is more energy efficient as no additional packets circulate in the network. Asynchronous protocols can attain greater scalability and higher energy efficiency than synchronized protocols without periodic time of synchronization. For this reason, in this paper, we decided to work with asynchronous approach. The detailed explanation of sleep scheduled mechanisms can be found in Subsection 2.2.

Opportunistic Routing (OR) is a new design of routing protocols where the nodes take multiple candidates instead of a single candidate to forward the packet. This approach could reduce the sender waiting time and fit well with the asynchronous MAC protocols. Opportunistic Routing in Wireless Sensor Networks (ORW) [12] protocol is based on this approach to minimize energy consumption.

Considering the ORW protocol, the Expected Duty Cycled Wakeups (EDC) is used as a routing metric for the computation of forwarder sets of each node based on the neighbor information. In EDC approach, each node knows its distance from other nodes. Starting with an empty set, the neighbor nodes are inserted into the forwarder set in the ascending order of their EDC's. Every time when new neighbor is inserted the EDC of the node is recomputed. The EDC of the neighbor node must be smaller than the current EDC of the node. ORR [13] improved over ORW by considering the residual energy of each node to select their forwarder set. Their approach reduces the individual node scheduling computation which simultaneously minimizes the cost and energy consumption of the nodes.

We intend to extend the EDC and significantly improve ORW and ORR by adding three attributes namely, transmission distance, expected number of hops and residual energy. ORW selects the forwarders based on expected wait time, therefore, nodes with large number of neighbors are more likely to become forwarders candidature. ORR selects the forwarder based on the larger residual energy of nodes more often. We select the forwarders based on shorter transmission distance, minimum expected number of hops towards the sink and larger residual energy of the nodes. The reasons behind the selection of these distributions are explained in Subsection 4.3.

Therefore, in this paper, we apply the SDN approach to WSN that enables the users to centrally control the network over an opportunistic routing stack. The above-mentioned studies provided convincing motivations for expanding the SDN concept with OR to the traditional WSN and propose an SDN based opportunistic routing for asynchronous duty-cycled WSNs (*SDORP*). We use the same approach as in ORW and ORR but improving over them by deeply addressing the flow computation and Expected Duty-Cycle Wakeup (EDC) to achieve the following objectives. First, the SDN concept integrated with WSNs which gives the flexible management of the network and routing strategies from the control plane to the data plane to enhance the network and energy efficiency. Second, calculating the flows using EDC metric with the integration of probability distributions on the controller is defined for each node as well as controls the number of candidates for each node. Third, the candidates of each node are prioritized based on the following attributes, which are defined as the average of three probability distributions, namely transmission distance distribution, expected number of hops distribution and residual energy distribution. The goal of this paper is to counterbalance between the sender waiting time and the number of duplicate packets along with the improvement of network efficiency through flexible management of the network. In summary, the key contributions of this work are as follows:

1) A new load balancing enhancement method has been developed, namely extended Expected Duty Cycled Wakeups (EEDC) by including three attributes in the first term of EDC, transmission distance, expected number of hops and residual energy.
2) A new protocol, namely *SDORP* is proposed which is able to integrating EEDC with MINI-SDN [14] to enhance load balancing and facilitates efficient management to reduce energy consumption and enhance the network efficiency. This work presents a more comprehensive solution to SDN based duty cycled opportunistic routing in WSNs.
3) A detailed probabilistic analysis is carried out to analyze the performance of the *SDORP* network by calculating the expected energy cost, the redundant

packets and the sender waiting time.
4) We implemented the *SDORP* that is based on [14], [15]. The source code is available at https://github.com/howbani/sdorp.

The rest of the paper is organized in the next sections. The Section 2 explains the related work. Section 3 presents the preliminaries. Section 4 presents the system model and problem formulation. Section 5 presents the proposed protocol. Section 6 presents the analysis. Section 7 evaluates the performance of the proposed protocol using extensive simulation. Finally, Section 8 concludes the paper.

2 Related Work

Various mechanisms for wireless sensor networks have been proposed over a decade. Some of them are adopted from different kinds of wireless networks such as LANs, but SDN based opportunistic routing makes a unique protocol designed for wireless sensor networks. Following are the previous studies which are divided into two parts. SDN based WSNs and Opportunistic Routing in WSNs.

2.1 SDN based WSNs

The complexity of the WSN keeps growing in such a way that dense networks cannot be controlled and maintained manually in real-time. The *Software Defined Wireless Sensor Network* (SDWSN) complies with this requirement that overcomes the traditional WSN limitations and induces it with control and auto-monitoring. It is commonly believed that SDN is an inevitable trend for WSN development with the rapid growth of various information and communication technologies.

Software-Defined Networking concept is introduced in wireless sensor networks with organized and hierarchical management to solve some intrinsic challenges in WSN management. In [16], the authors expressed that SDN in traditional WSN should support energy awareness actions which is currently being investigated in WSNs such as in-network data aggregation, duty-cycling, and cross-layer optimization. SDWSN architecture [17] merges context-aware and policy-based routing modules according to the SDN standards. Another effort was made to merge WSN and SDN in order to explain the inherent challenges of WSN [18]. In [19], an architecture was developed to categorize control plane and data plane individually, where OpenFlow is described as a standard SDWSN communication protocol among both planes. Subsequently, to reduce the energy consumption and intricacy of sensor nodes structure, the smart grid WSNs utilized SDWSN. Moreover, SDN was utilized for managing WSNs under the smart software-defined control by keeping the controller at the base location. It was suggested and argued that SDN can solve most of the intrinsic challenges of WSNs through smart management [20]. In order to simplify the management of the network, a networking solution SDN-WISE was introduced [21]. In [22], the improved SDWSN framework has been introduced to enhance the network reliability which minimizes the network coupling by separating the control and data planes and used an external controller to assign flow rules. This framework has reduced the energy consumption of the network, but to a certain limit as expected because they did not consider any energy aware algorithm. In [23], the authors proposed an energy-aware cognitive-based SDWSN protocol by utilizing Reinforced Learning (RL) for monitoring applications. This protocol was introduced to improve the energy efficiency for WSN monitoring applications. In [24], a tiny OS-based SDN mechanism has been introduced which utilizes the multiple controllers within the SDWSN. This work focused solely on hardware architecture. In [25], the authors proposed a routing algorithm for data routing on software-defined WSNs in which the nodes are divided into clusters and each of them allocated to a control node (cluster head). The control node is selected by the controller based on the transmission distance and the remaining energy of the nodes. The selection of control nodes at the control plane is formulated as an NP-hard problem and then optimized by utilizing the Practical Swarm Optimization (PSO) algorithm.

The above-mentioned literature demonstrated the feasibility and acceptability of SDWSN. However, they lack of combined focus on both planes regarding exchange of information. This information exchange is an important factor between two planes. In order to overcome the message overhead in the network, the information exchange between two planes needs to be reduced. Besides, the combined problem of load balancing and data routing between two planes has not been properly addressed. Therefore, we go beyond the state of the art by utilizing the software-defined concept to WSNs where such kind of framework offers information exchange between two planes and overcomes the above-mentioned problems. For the detailed explanation refer to Section 3.

2.2 Opportunistic Routing in WSNs

A number of diverse mechanisms have been proposed for *Opportunistic Routing* (OR) with duty-cycling based on MAC to assist data routing in WSNs. The majority of OR protocols are devoted to routing metric, candidate selection, and candidate coordination.

Existing MAC protocols of WSNs, divided into synchronous and asynchronous MAC, are developed on top of duty-cycling. S-MAC [26] is a synchronized protocol designed for WSNs. The nodes are based on periodic sleep and wake-up method. The nodes exchange their schedule by periodically broadcasting SYNC packets and use the RTS/CTS mechanism to avoid collision. The AIMRP protocol proposed in [27] based on S-MAC utilizes a hop count coordinate system to the sink. A node's candidate set is formed by the neighbor nodes with smaller hop count rather than it's own. To minimize the waiting cost, the first node that wakeup will be selected as a forwarder. However, it is not a good way to build a candidate set only with the hop count.

On the contrary, asynchronous MAC protocols work with different wake-up times. In order to communicate between two nodes, the sender or the receiver must wait for wake-up of its counterpart. Some of the opportunistic routing protocols [12], [28], [29] operates with sender-based MAC protocols (B-MAC [30], X-MAC [31] or BoX-MAC [32]). A sender sends a preamble packet before transmitting

a data packet. The candidate node which wakes-up first that hear the packet and sends back an acknowledgment will become the forwarder.

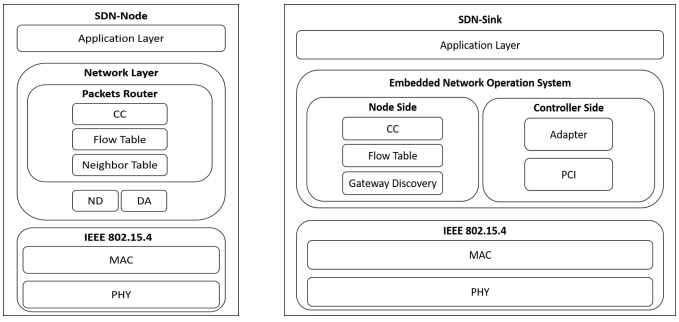
OR is a useful approach to further minimize the sender waiting time by allowing multiple candidates, hence any of the candidates can receive and forward the packet. ExOR [33] utilized a routing metric called an Expected Transmission Count (ETX). Unfortunately, there is no precise OR metric in ETX because the packet can be guided through any of the potential paths. Furthermore, ExOR believes that each node is in wake-up mode and can overhear packets, which does not apply to duty-cycled WSNs. Later, the Expected Any-Path Transmission (EAT) [34] improved upon the ETX. The energy-efficient opportunistic routing protocol named EEOR [35], where each node selects and prioritizes the candidate list which is formed by the neighbors closer to the destination. In terms of energy consumption, average delivery delay and packet loss ratio EEOR performs better than ExOR. However, EEOR is similar to ExOR where nodes are awake all the time and cannot be used in the asynchronous scenario. GeRaF [36] is similar to ExOR where Distance Progress (DP) has been introduced which is an estimate of the remaining routing distance to the destination. The priority is given to those candidates that are close to the destination. Subsequently, DP is extended to Expected Distance Progress (EDP) in DPOR [37]. EDP considers the link delivery probability and the node position together. Thus, EDP is more reliable than the DP. ORW utilized a new metric called an Expected number of Duty-Cycled wakeups (EDC) for the global metric and proposed a candidate set algorithm for each node that could minimize the EDC to the sink [12]. Duplicate-Detectable Opportunistic Forwarding (DOF) protocol is capable of solving duplicate transmissions in traditional OR protocols operating with sender-based MAC protocols [28]. ORR [13] and ORIA [38] are almost similar to ORW, both are based on EDC with slight differences. ORR eliminates the third term in EDC (i.e., eliminates the forwarding cost from EDC) and adds the remaining energy parameter to the first term of EDC. ORIA considers two metrics hop count and EDC.

Although all of these routing techniques can reduce energy consumption to enhance the energy efficiency of individual nodes or the entire network. However, they did not consider the effective load balancing approach among the nodes while designing their OR metric to achieve the longer network lifetime. Therefore, in our proposed protocol, software-defined networking is adapted to the opportunistic routing for the flexible management of WSN. In addition, we have considered the three attributes (transmission distance, expected number of hops, residual energy) in the EDC metric for effective load balancing between nodes. The EDC is computed and the number of candidates is controlled in the control plane which are elaborated in Section 5.

3 PRELIMINARIES

3.1 Integration of SDN

The above-mentioned literature contributed significantly and offered convincing motivations for considering SDN concepts into opportunistic routing in WSNs. Therefore, in this paper, the SDN architecture is adapted from our previous work [14]. The model contains the control plane, data plane and the sink. The sink acts as a gateway between the control plane and the data plane. The SDN controller could be located externally on a remote server or internally in the sink. In this work, the controller is considered to be an autonomous external device that interacts directly with the sink instead of the sensor nodes. The SDN architecture is composed of three components (Node, Sink, and Controller) that are built-in sub architectures.



(a) SDN-Node (b) SDN-Sink

Fig. 1: SDN protocol stack for node and sink

Fig. 1a presents the SDN-Node protocol stack that operates the basic functionalities of PHY and MAC layers. At the top of the protocol stack, there is network layer which operates the micro-control unit that consisting of three applications: Neighbor Discovery (ND), Data Aggregation (DA) and Packet Router. In ND, each node is allowed to discover and store information about the neighboring nodes. DA performs data aggregation or decision fusion in order to provide complete coverage for a certain area. The sensor transmits data to the neighbors and each neighbor transmits data to their neighbors and so on. One node can receive an enormous number of repetitive data from various neighbors that can be generated from the same origin node or even by redundant nodes. The packet router consists of three applications: Candidates Coordination (CC), Flow Table and Neighbor Table. The candidate coordination (CC) is used by the sender node to determine which of the candidate required to forward the packet. The candidate with highest priority must forward and receive the packet and the other candidates will drop it. In general, this mechanism requires the signaling between the nodes to avoid redundant packets. The flow table plays a critical role in SDN, which stores the matching rules populated by the control plane for controlling and directing the packet flows in SDN. The neighbor table stores information about the neighbor nodes of each node.

Fig. 1b presents the SDN-Sink protocol stack that operates the basic functionalities of PHY and MAC layers. At the top of the protocol stack, the Embedded Network Operation System (ENOS) is an intermediary interface that operates from sink to nodes and sink to controller communications. The ENOS consists of two principal sides, the controller and the node sides. The node and the controller sides are responsible for intercommunications of nodes to sink and controller to sink respectively. The node side consists of three applications: Candidates Coordination (CC), Flow

Table and Gateway Discovery. We have already explained above the candidates coordination and flow table because they implement the same functionality as in the packets router. The access nodes are discovered by the gateway discovery. For instance, the neighboring nodes of the sink that can communicate directly through one hop. The controller consists of two applications: PCI (Programmable Communication Interface) and Adapter. The PCI application is responsible for managing the communication between the sink and controller. The adapter application is responsible for interpreting the network policies and messages in a way that sensor nodes and controller can understand.

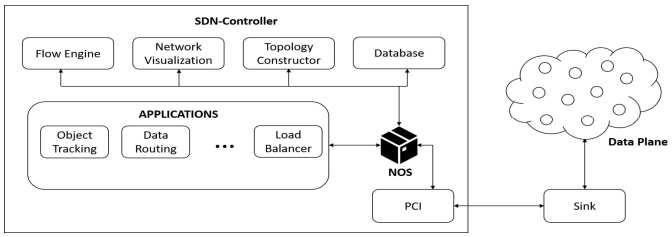


Fig. 2: SDN protocol stack for controller

Fig. 2 presents the SDN-Controller protocol stack where it performs the network service functionalities in Network Operating System (NOS). The NOS consists of Flow Engine (FE), Network Visualization (NV), Topology Constructor (TC), Database, PCI and Applications. The PCI application is responsible for managing the communication between sink and controller. The design of the database application is to store topological gathered data, particularly node's ID, node's battery level, node's neighbors, node's location, and sink, etc. The Topology Constructor application uses vertices and edges to build a graph-based pattern for the sensor network based on the data stored in the database. The network visualization builds a coherent and comprehensive representation of the network's statistics and current state by using the topology constructor and database for topological gathered data. The controller's flow engine is an important application that takes all the data forwarding decisions and implements all high/low levels network policies. Furthermore, it is responsible for translating the network event driven policies into forwarding rules implemented in the nodes and also for updating the nodes and sink flow tables respectively.

In the adapted SDN architecture [14], the MINI-FLOW is designed to compute and manage the routing flows and the paths between the controller and the end nodes. It consists of three mechanisms: the routing flow from nodes to sink, the routing flow from controller to nodes (Flow Instantiation) and the data routing at the network initialization level. Also, its implementation is based on the low power listening BoX-MAC protocol [32].

TABLE 1: Flow Table

Node ID	EDC Priority	ACK	Action	Statistics
1	2.26672	1	Forward	39
2	2.53900	1	Forward	25
3	3.46850	1	Drop	9

According to Table 1, the flow table saves the routing flows. The routing flows in a sender node have three neighboring nodes and each flow is assigned with EDC priority value. The flow table is designed to emphasize on two objectives. The first objective is to reduce the information exchange between the control and the data plane where the controller delivers the routing flows to each node, but for each packet, it does not need to specify the flows. This reduces the exchange of information between the two planes. The flows are updated by the controller on the basis of the statistics obtained from the end nodes. The second objective for duty cycled nodes is to shorten the waiting time and minimize the redundant packets, with multiple flows allocated to each node. The EDC priority value of each node is computed in the Flow Engine. Whenever a sender node needs to send a packet, it first sends the preamble packet to its neighboring nodes. The awakened neighboring nodes will hear and receive that packet, and then send the ACK back to the sender. The ACK value is set to 1.0 by the sender in the flow table. According to the EDC priority, if the flow is matched, then the corresponding action is performed and the information is updated in the statistics. The information of the node statistics is reported regularly to the controller. The computations of EDC priority and Actions are explained in Section 5.

TABLE 2: Neighbors Table

Node ID	Battery Level	Estimated Location	$E[\mathcal{H}(\mathbf{i})]$	$\tau(\mathbf{i})$
$n1$	Φ(i), 1	$x1, y1$	$E[\mathcal{H}(\text{i}), 1]$	τ(i), 1
$n2$	Φ(i), 2	$x2, y2$	$E[\mathcal{H}(\text{i}), 2]$	τ(i), 2
$n3$	Φ(i), 3	$x3, y3$	$E[\mathcal{H}(\text{i}), 3]$	τ(i), 3
$n4$	Φ(i), 4	$x4, y4$	$E[\mathcal{H}(\text{i}), 4]$	τ(i), 4

Based on the network wide-view the controller fills the flow table and each node fills its own neighbor table in a distributed manner as shown in Table 2 where the node n_i stored the required information. We assume that the node n_i has four neighboring nodes $\mathbb{N}_i = \{n_1, n_2, n_3, n_4\}$, the function $\Phi(n_j \in \mathbb{N}_i)$ returns the battery level of the neighboring nodes n_j, $E[\mathcal{H}(n_j \in \mathbb{N}_i)]$ returns the expected number of hops of the neighboring nodes from the node n_j to the sink n_b and $\tau(n_j \in \mathbb{N}_i)$ returns the transmission distance from the node n_i to node n_j. The estimated location returns the location of the neighboring nodes.

4 System Model and Problem Formulation

4.1 System Model

A standard software-defined wireless sensor network consists of three layers: application plane, control plane and the data plane as shown in Fig. 3. The application plane interacts with the different applications of the network. The control plane is responsible for topology management, data transmission in terms of flow control and load balancing across the network. This reduces the energy resources of the sensor nodes individually. The data plane consists of sensor nodes, $\mathbb{N} = \{n_1, n_2, ..., n_m\}$, with $m = |\mathbb{N}|$ where the sensor functions are performed for data generation and forwarding which is based on the routing rules assigned by the control plane.

Consequently, the operational network model of a SD-WSN architecture is assumed to be a directed graph $G = (V, E)$ where V is the vertex set including sensor nodes and E denotes the directed communication links connecting

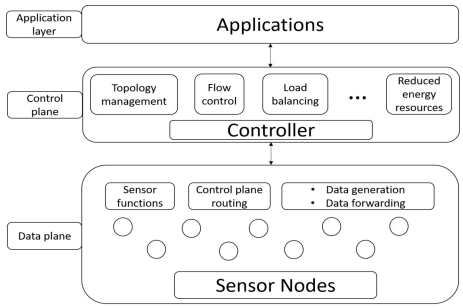


Fig. 3: Software defined wireless sensor network model

sensor nodes. The SDN controller manages all the sensor nodes in the network. Some of the notations used in this paper are summarized in Table 3.

TABLE 3: Notations

Notation	Definition
$\mathbb{N}$	$\mathbb{N} = \{n_0, n_1, n_2, ..., n_m\}; n_i \in \mathbb{N}$ is a sensor node; m is the size of $\mathbb{N}$.
$\mathbb{N}_i$	The neighboring nodes of n_i; m_i is the size of $\mathbb{N}_i$.
n_b	The sink.
n_s	The source node.
n_t	The target node.
$E[\mathcal{H}_{i,j}]$	The expected number of hops of n_i's neighbor n_j to sink n_b.
$\tau_{i,j}$	The transmission distance from n_i to n_j.
$\Phi_{i,j}$	The residual energy of the of n_i's neighbor n_j.
e_*	The initial energy.
e	The *Euler's Constant*. Its value is approximately equal to 2.71828.
R	The radius of communication.
Υ	The energy control variable for energy model.
π	The default value of Pi is approximately equal to 3.14159.

4.2 Energy Consumption Model

In this study, a highly simplified energy consumption model [39] is adapted for transmitting and receiving the packets. The model used both the free space (ε_{fs}) (d^2 energy loss) and the multi-path fading channels (ε_{mp}) (d^4 energy loss) which totally depends on the distance between the sender and the receiver calculated by *Euclidean distance* d using Eq. (6). The path loss can be compensated by using the energy control mechanism. The required transmission energy consumption can be computed in Eq.(1) to transmit the k bits packet for the distance d.

$$E_{Tx}(k,d) = E_{Tx-elec}(k) + E_{Tx-amp}(k,d) = \begin{cases} k.E_{elec} + k \cdot \varepsilon_{fs} d^2, & d < d_* \\ k.E_{elec} + k \cdot \varepsilon_{mp} d^4, & d \geq d_* \end{cases} \quad (1)$$

The required receiving energy consumption can be computed in Eq. (2) to receive k bit packet for the distance d.

$$E_{Rx}(k) = E_{Rx-elec}(k) = k.E_{elec} \quad (2)$$

E_{elec} is the energy consumed per unit to operate the transmitter or receiver circuit which depends on various factors such as filtering, signal propagation, modulation and digital coding. ε_{mp} and ε_{fs} are elected for the transmission amplifier model. The transmitted data packet length is denoted by k, the d_* is the transmission distance threshold value which is computed by Eq. (3).

$$d_* = \sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}} \quad (3)$$

4.3 Problem Statement and Formulation

It is possible that a candidate node with a greater number of hops to the sink can become a forwarder based on the structure of the EDC in ORW. In this case, if the node uses a greater number of hops to the sink and longer transmission distance towards the next hop node which can negatively affect the energy consumption in two ways. First, the greater number of hops from the sink consumes more energy as compared to smaller number of hops. Second, if the hop to hop transmission distance is equal or greater than the threshold d_*, it will also affect the energy consumption of the network.

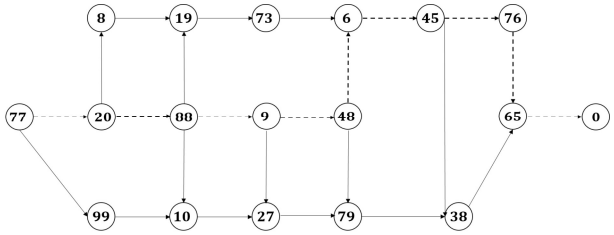


Fig. 4: Greater and smaller number of hops

Fig. 4 shows the difference between the greater and the smaller number of hops to the sink. It depicts that the node n_{77} selects the path towards the sink n_0 which is expressed as dotted line. The path contains greater number of hops to the sink consumes more energy as compared to smaller number of hops. The transmission distance of each hop is assumed to be less than the threshold d_* as expressed in Eq. (1).

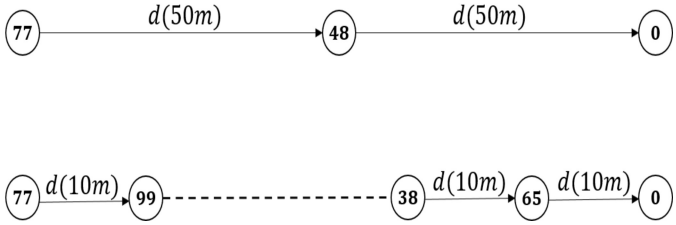


Fig. 5: Longer and shorter transmission distance

In Subsection 4.2, based on Eq. (1) and Eq. (2), we derived the Eq. (4). Here in Eq. (4), we consider the multi-hop scenario with m number of nodes in the network. The energy required to transmit the k bits packet from one node to another node, depending on the transmission distance d of each other, is denoted by $E_{Tx}(k,d)$. The energy required in receiving the k bits packet is denoted by $E_{Rx}(k)$. The required transmission power is based on the transmission amplifier $\varepsilon_{amp}.d^{\Upsilon}$, the free space ε_{fs} and the multi-path ε_{mp} models as expressed in Eq. (1). Here, Υ is the energy control variable for the energy model. Fig. 5 shows the difference between longer and shorter transmission distance towards the next hop node. If we consider the shorter transmission distance between node n_{77} and node n_{99} assuming less than the threshold d_* then it will apply free space ε_{fs} model

with d^2 to consume transmission energy. If we consider the longer transmission distance between node n_{77} and node n_{48} assuming equal or greater than the threshold d_* then it will apply multi-path ε_{mp} model with d^4 to consume transmission energy. Hence, for all multi-hop cases depicted in Fig. 4 and Fig. 5, we expressed the total energy consumption for transmitting and receiving the k bits packet by m number of nodes in the network in Eq. (4) and the generalized term is obtained by Eq. (5).

$$\begin{aligned} E_{MH}(m,k,d) &= m.E_{Tx}(k,d) + (m-1).E_{Rx}(k) \\ &= m.k.(E_{trans} + \varepsilon_{amp}.d^{\Upsilon}) + (m-1).k.E_{recv} \\ &= k.(m.(E_{trans} + E_{recv} + \varepsilon_{amp}.d^{\Upsilon}) - E_{recv}) \end{aligned} \tag{4}$$

$$\begin{aligned} E^{all}_{MH}(k,d) &= \sum_{l=1}^{m} E_{MH}(l,k,d) \\ &= \textstyle\sum_{l=1}^{m} l.E_{Tx}(k,d) + (l-1).E_{Rx}(k) \\ &= \tfrac{m(m+1)}{2}.k.(E_{trans} + \varepsilon_{amp}.d^{\Upsilon}) + \tfrac{m(m-1)}{2}.k.E_{recv} \end{aligned} \tag{5}$$

However, in order to achieve a greater network lifetime, it is very important to balance the load among the nodes. ORW did not consider the load balancing among the nodes and ORR only considers the residual energy of each node for selecting the forwarders. The nodes with the higher residual energy will become the forwarders more often. Both ORW and ORR did not consider the transmission distance between nodes while calculating the EDC. Thus, considering only the residual energy in ORR will not give much effective results because the transmission distance between nodes is strongly related to the energy consumption of sender and the receiver as described in *Energy Consumption Model* Subsection 4.2.

In order to balance the load among the nodes, we consider the three attributes: residual energy, the transmission distance and the expected number of hops. The nodes with higher residual energy, shorter transmission distance and minimum expected number of hops will be selected as forwarders more often. The reason behind considering the expected number of hops is that, in the design of the EDC, it is possible that a node can become a forwarder with a greater number of hops to the sink. This will reduce the above-mentioned problem which simultaneously affects to reduce the energy consumption of the network.

The forwarding structure is demonstrated to be loop free feature. Since the forwarder sets established the graph which forms a Destination Oriented Directed Acyclic Graph (*DODAG*). A *DODAG* structure upstream node always has a lower EDC than a downstream node. We propose a model for selecting forwarder which includes the average of three probability distributions in the first term of EDC. The Section 5 explains the proposed methodology.

5 The Proposed Protocol

The main purpose of this study is to counterbalance the two contradictory problems: sender waiting time and packets duplication. We propose the SDN based opportunistic routing which deeply addresses the computation of the routing flows with EDC. The rest of the section focuses on the data reporting at network initialization, computing the routing flows (i.e., the paths from the end nodes to the controller), and flow instantiation (i.e., controller assigns flows to the end nodes)..

5.1 Network Initialization

A distributed algorithm is used to initialize the network to report the required information to the controller because the controller does not have any information about the network nodes. In order to compile the topological data and report to the controller, each node transmits the beacon packets during the initialization stage. The data is stored in the Database and this can be used by the Network Visualization (NV), Topology Constructor (TC) and Flow Engine (FE). Algorithm 1 describes the initialization of the network in a distributed manner and how the topological data is gathered in the *Neighbors Table* (Table 2). During the initialization phase the only information regarding location of each node in the network should be collected. We apply the *Euclidean distance* Eq. (6) to calculate to distance between each node to sink.

$$d_{i,j} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{6}$$

Algorithm 1: Network Initialization
n_b is the sink node and the default distance value of $n_b = 0$.

```
1  Q = ∅; // define a queue
2  d_{i,b} = ∞;
3  NetworkInitialization(n_b); // Function Called in Line 4
4  NetworkInitialization(n_x)
5  The node n_x sends the beacon packet
6  n_x.true=true; // Beacon Sent
7  foreach node n_i that hears the packet sent by n_x do
8      if d_{x,b} < d_{i,b} ∧ d_{i,b} == ∞ then
9          Calculate the distance d_{i,b} from n_i to n_b // using Eq. (6)
10         n_i.Send Response to n_x;
11     end
12     if n_i.true ≠ true ∧ n_i ∉ Q then
13         Q.Enque(n_i);
14     end
15 end
16 while Q.count > 0 do
17     n_j = Q.dequeue();
18     NetworkInitialization(n_j);
19 end
```

By performing the following steps, the compiled data is reported to the sink. Firstly, the sender node sends a preamble packet and after receiving these packets by the neighboring nodes, they return ACK packets to the sender subject to their availability. Secondly, the packet router selects a forwarding candidate from the neighboring nodes, which expresses their availability such that the distance of the forwarding candidate node is smaller than the distance of the sender to the sink. Finally, the ID of the forwarding candidate is mentioned by the packet router in its IDs field of the control packet. This process is carried out till the packet approaches the sink.

5.2 The Routing Flows

The node-to-controller data routing is determined in the *Flow Table* (Table 1) based on the priority value of each flow. Since each node has numerous candidates at each transmission phase. Consequently, each node has numerous

paths to reach at the sink. To optimize the path selection, our proposed protocol includes three important attributes in the first term of EDC to obtain the higher link quality. A node that has links with higher quality will have smaller EDC, because the expected sender waiting time is shorter. The attributes in EDC are defined as the average of the three probability distributions expressed in Eq. (17). The distributions are transmission distance distribution Eq. (8), the expected number of hops distribution Eq. (14), and the residual energy distribution Eq. (16).

5.2.1 Transmission Distance Distribution

To compute the transmission distance between the sender node to each of its neighbor $\tau_{i,j}$, we consider the *Euclidean distance* Eq. (6) from n_i to n_j. After getting the distance from sender node to each neighbor, we normalized it between $[0-1]$ by Eq. (7), and then obtained the exponential distribution of transmission distance by using the probability mass function in Eq. (8), where e is the *Euler's Constant*. The aim of this distribution is to set a higher priority to those nodes which are closer to the sender. In Eq. (8), $\lambda_\tau \geq 0$ is the distribution control variable, the higher the distribution control variable expresses the greater probability for the nodes which have closer transmission distance to the sender to be selected as forwarders. The default value of the control variable is set to $\lambda_\tau = 0.5$.

$$\bar{\tau}_{i,j} = (\frac{\tau_{i,j}}{R}) \quad \forall\, n_j \in \mathbb{N}_i \tag{7}$$

$$\tilde{\tau}_{i,j} = \frac{1 - \mathrm{e}^{(1-\frac{1}{(\bar{\tau}_{i,j})^{\lambda_\tau}})}}{\sum_{v=1}^{m_i} 1 - \mathrm{e}^{(1-\frac{1}{(\bar{\tau}_{i,v})^{\lambda_\tau}})}} \quad \forall\, n_j \in \mathbb{N}_i, \lambda_\tau \geq 0 \tag{8}$$

5.2.2 Expected Number of Hops Distribution

To compute the expected number of hops, we first calculate the *Euclidean distance* Eq. (6) $d_{i,b}$ from the node n_i to the sink node n_b. If the distance $d_{i,b}$ is greater than the communication radius R, then the node n_i uses the intermediate nodes to forward the packet towards sink n_b using two or more hops.

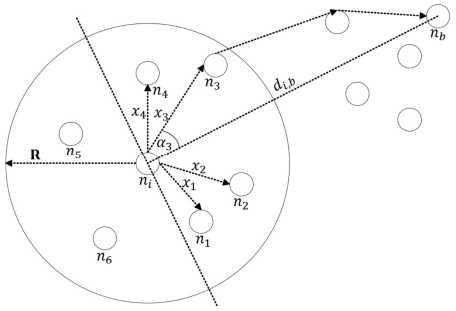


Fig. 6: The distance between node and neighboring nodes

Let us consider the Fig. 6 where the node n_i has independent random neighbor nodes having the random distribution with the communication radius $[0-R]$. The density function of distance x between the node n_i and neighbor nodes n_j is computed by Eq. (9).

$$f_{X\alpha}(x,\alpha) = \frac{2x}{\pi R^2} \tag{9}$$

Where $-\frac{\pi}{2} \leq \alpha \leq \frac{\pi}{2}$ and $0 \leq x \leq R$. Integrating the Eq. (9) over α gives the density function of $f_X(x)$ in Eq. (10).

$$f_X(x) = \frac{2x}{R^2} \tag{10}$$

According to the concept of maximum of n random variables in [40], the sender node n_i computes the neighbor node with the expected maximum distance $E[\mathcal{M}_{n_j}]$ within communication radius towards sink n_b. The expected maximum distance neighbor node $E[\mathcal{M}_{n_j}]$ is computed by Eq. (11). Based on Eq. (10) and Eq. (11), the expected number of hops from each node n_i to sink n_b is computed by Eq. (12). If the distance $d_{i,b}$ lies within the communication radius R, it will count expected one hop.

$$E[\mathcal{M}_{n_j}] = \left(\frac{2m_i}{2m_i+1}\right) * R \quad \forall\, n_j \in \mathbb{N}_i \tag{11}$$

$$E[\mathcal{H}_i] = \frac{2E[\mathcal{M}_{n_j}]}{R^2} * d_{i,b} \quad \forall\, n_j \in \mathbb{N}_i \tag{12}$$

The $E[\mathcal{H}_{i,j}]$ represents the expected number of hops of n_i's neighbor node n_j. The random variable $E[\mathcal{H}_i] = (E[\mathcal{H}_{i,1}], E[\mathcal{H}_{i,2}], E[\mathcal{H}_{i,3}], ..., E[\mathcal{H}_{i,m_i}])$ is defined by the expected number of hops of each neighbor node n_j to sink n_b. In an analogy with Eq. (7) and Eq. (8), the expected number of hops is normalized between $[0-1]$ by Eq. (13), and then obtained the exponential distribution by using the probability mass function in Eq. (14). The distribution set a higher priority to those forwarders which have the minimum expected number of hops towards sink n_b. In Eq. (14), $\lambda_\mathcal{H} \geq 0$ is the distribution control variable, the higher the distribution control variable expresses the greater probability for the nodes which have minimum expected number of hops. The default value of the control variable is set to $\lambda_\mathcal{H} = 0.5$.

$$E[\bar{\mathcal{H}}_{i,j}] = \left(\frac{E[\mathcal{H}_{i,j}]}{R}\right) \quad \forall\, n_j \in \mathbb{N}_i \tag{13}$$

$$E[\tilde{\mathcal{H}}_{i,j}] = \frac{1 - \mathrm{e}^{(1-\frac{1}{(E[\bar{\mathcal{H}}_{i,j}])^{\lambda_\mathcal{H}}})}}{\sum_{v=1}^{m_i} 1 - \mathrm{e}^{(1-\frac{1}{(E[\bar{\mathcal{H}}_{i,v}])^{\lambda_\mathcal{H}}})}} \quad \forall\, n_j \in \mathbb{N}_i, \lambda_\mathcal{H} \geq 0 \tag{14}$$

5.2.3 Residual Energy Distribution

The $\Phi_{i,j}$ represents the residual energy of n_i's neighbor node n_j. The random variable $\Phi_i = (\Phi_{i,1}, \Phi_{i,2}, \Phi_{i,3}, ..., \Phi_{i,m_i})$ is defined by the residual energy of each neighbor node n_j. The random variable Φ_i is normalized between $[0-1]$ by Eq. (15), where e_* is the initial energy of node n_j. The exponential distribution of residual energy is obtained by using the probability mass function in Eq. (16). The aim of this distribution to set a higher priority to those nodes which have the greater remaining energy. In Eq. (16), $\lambda_\Phi \geq 0$ is the distribution control variable, the higher the distribution control variable expresses the greater probability for the nodes which have greater residual energy to be selected as forwarders. The default value of the control variable is set to $\lambda_\Phi = 1$.

$$\bar{\Phi}_{i,j} = (\frac{\Phi_{i,j}}{e_*}) \forall\, n_j \in \mathbb{N}_i \tag{15}$$

$$\tilde{\Phi}_{i,j} = \frac{e^{(1-\frac{1}{(\bar{\Phi}_{i,j})^{\lambda_\Phi}})}}{\sum_{v=1}^{m_i} e^{(1-\frac{1}{(\bar{\Phi}_{i,v})^{\lambda_\Phi}})}} \forall n_j \in \mathbb{N}_i,\ \lambda_\Phi \geq 0 \quad (16)$$

5.2.4 Link Estimation

In the link estimation, the controller calculates the average of three attributes: the transmission distance distribution ($\tilde{\tau}_{i,j}$) Eq. (8), the expected number of hops distribution ($E[\tilde{\mathcal{H}}_{i,j}]$) Eq. (14) and the residual energy distribution ($\tilde{\Phi}_{i,j}$) Eq. (16). It also defines the distributions term for each node n_i by the vector $\bar{\mathcal{L}} = (\bar{\mathcal{L}}_{i,1}, \bar{\mathcal{L}}_{i,2}, ...\bar{\mathcal{L}}_{i,m_i})$ such that $\tilde{\mathcal{L}}_{i,j} = (\tilde{\Phi}_{i,j} + \tilde{\tau}_{i,j} + E[\tilde{\mathcal{H}}_{i,j}])/3$. The average term is computed by Eq. (17). To obtain the higher link quality by balancing the load among the nodes, we consider this average distribution term in the first term of the EDC (18). The higher link quality will have a smaller EDC. The EDC is computed by the controller for each node to assign the flow rules which reduces the information exchange between controller and the end nodes. The three distributions are controlled by the three exponential control variables (λ_τ, $\lambda_\mathcal{H}$, and λ_Φ). Increasing the value of any control variable will increase the impact of the corresponding distribution. To maximize the network lifetime the value of λ_Φ is set to be greater than the value of λ_τ and $\lambda_\mathcal{H}$. The λ_Φ is designed to avoid selecting the nodes with lower residual energy in each transmission phase.

$$\tilde{\mathcal{L}}_{i,j} = (\tilde{\Phi}_{i,j} + \tilde{\tau}_{i,j} + E[\tilde{\mathcal{H}}_{i,j}])/3$$

$$= \frac{1}{3}\left[\frac{e^{(1-\frac{1}{(\bar{\Phi}_{i,j})^{\lambda_\Phi}})}}{\sum_{v=1}^{m_i} e^{(1-\frac{1}{(\bar{\Phi}_{i,v})^{\lambda_\Phi}})}} + \frac{1-e^{(1-\frac{1}{(\bar{\tau}_{i,j})^{\lambda_\tau}})}}{\sum_{v=1}^{m_i} 1-e^{(1-\frac{1}{(\bar{\tau}_{i,v})^{\lambda_\tau}})}} + \frac{1-e^{(1-\frac{1}{(E[\bar{\mathcal{H}}_{i,j}])^{\lambda_\mathcal{H}}})}}{\sum_{v=1}^{m_i} 1-e^{(1-\frac{1}{(E[\bar{\mathcal{H}}_{i,v}])^{\lambda_\mathcal{H}}})}}\right] \forall n_j \in \mathbb{N}_i \quad (17)$$

$$EDC_i = \frac{1}{\tilde{\mathcal{L}}_{i,j}.(n_i)} + \frac{\sum_{n_j \in \mathbb{N}_i} EDC_j}{n_i} \quad (18)$$

5.3 Flow Instantiation

The role of the controller is very critical in terms of handling the intelligence of the network at a central place. A heuristic based distributed algorithm is designed to compute the path from the controller to the end nodes. The purpose is that the controller wants each node to know about how the data is treated. The controller computes and prioritizes the routing flow rules and store into the flow table of each node as shown in Table 1. Also, it makes decision to control the transmission flows in the data plane. The controller manages and controls the network and keeps up to date the topology and the state of the network. According to Fig. 7, there are multiple reverse paths from sink node n_b to end node n_{61}. In order to disseminate the user defined operations to each node, the controller builds a sub-graph $G_i = (V_i, E_i)$ for the target node n_t and applies the *Euclidean distance* Eq. (6) from node to neighbor node and the next hop node to the target node to compute the shortest path.

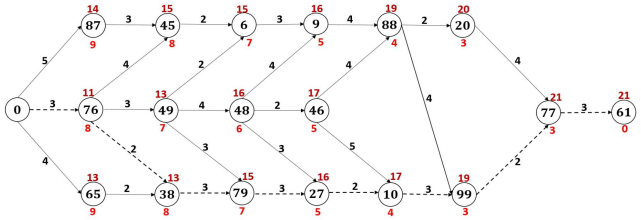


Fig. 7: The sub-graph of the end node n_{61}

The Fig. 7 shows the example of potential paths from n_0 to n_{61}, each path contains the same number of hops. We assume that the edges weight contains the distance from node to next hop node. The weight under the node contains the distance from the node to the target node n_{61}. The weight above the node contains the cumulative distance of the source node to the current node reaching towards the target node, while the vertex value inside the node shows the ID of the sensor node.

According to *Algorithm 2*, the controller sorts out and utilizes the possible shortest route expressed in dotted lines to assign the flow rules from n_0 to n_{61}.

Algorithm 2: Flow Instantiation
n_b is the sink and the source node, n_t is the target node. Compute the shortest distance path from sink to the target node.

```
1  Q = ∅; // define an open list queue
2  C = ∅; // define close list queue
3  S = ∅; // define shortest path queue
4  foreach node n_i in the network do
5      F[n_i] ← ∞
6      prev[n_i] ← undefined
7      Q.Enque(n_i);
8  end
9  D_value[n_b] = 0;
10 while Q ≠ empty do
11     u ← node in Q with minimum F[u] // [n_b] = 0
12     Q.Remove(u);
13     C.Enque(u);
14     x ← Target_node[n_t];
15     if x == u then
16         if prev[u] is defined ∨ u == n_b then
17             while u is defined do
18                 S.Enque(u);
19                 u ← prev[u];
20             end
21         end
22         return x;
23     end
24     foreach neighbor n_i of u do
25         D_value[n_i] ← D_value[u] + d_{u,n_i}; // d_{u,n_i} using Eq. (6)
26         Target_value[n_i] ← d_{n_i,n_t}; // d_{n_i,n_t} using Eq. (6)
27         Final_value[n_i] ← D_value[n_i] + Target_value[n_i];
28         if Final_value[n_i] < F[n_i] then
29             F[n_i] ← Final_value[n_i];
30             prev[n_i] ← u;
31         end
32     end
33     return F[n_i], prev[n_i];
34 end
```

5.4 Candidates Coordination

Coordination among candidates in OR is a challenging issue. It is the mechanism that candidates use to decide whether to forward or discard the received packet. A constructive mechanism for candidate coordination should prevent the transmission of more than one packet by choosing

the best candidate for forwarding the packet. The mechanism for candidate coordination generally involves the signaling between the nodes. If there is an imperfection of coordination among candidates, this can lead to duplicate packet transmissions from different candidates. The OR protocol must use reliable signals to have complete candidate coordination. Generally, the active/sleep states of the MAC layer are coordinated by one of the two approaches: synchronous and asynchronous [41]. In the synchronous approach of MAC protocol, several nodes wake-up at the same time and have the same active period. While in the asynchronous approach of the MAC protocol, the sender and receiver nodes should be engaged in the active state as the nodes randomly define their wake-up intervals and active periods. The candidate coordination of MAC layer executes in the following steps. 1) The next hop (forwarder) decided by the network layer using a predefined routing metric e.g., link quality. 2) The MAC layer waits for the forwarder node to wake-up and receive the packet. 3) The forwarder node sends the ACK back to the sender when the packet is received. In order to avoid the negative effect of multiple receiver, the coordination among the candidates is an essential mechanism. However, if multiple candidates are in active mode, the candidate coordination selects the forwarder node that has lower EDC in Table 1 to ensure that the data packet will be sent to one candidate.

The Routing Flow Action: The action of the routing flow to be performed on the packets (forward, drop) is described by Eq. (19), where $\overrightarrow{C}_i$ is the routing candidates threshold value that are selected by the node n_i.

$$\overrightarrow{Act}(n_i) = \begin{cases} \textit{Forward } \overrightarrow{C}_i \leq \left\lceil \sqrt{\left(1+\frac{m_i}{\pi}\right)} \right\rceil; \; EDC_j \leq \frac{\sum_{j=1}^{m_i} EDC_j}{m_i}; \; \forall\, n_j \in \mathbb{N}_i \\ \textit{Drop } \overrightarrow{C}_i > \left\lceil \sqrt{\left(1+\frac{m_i}{\pi}\right)} \right\rceil; \; EDC_j > \frac{\sum_{j=1}^{m_i} EDC_j}{m_i}; \; \forall\, n_j \in \mathbb{N}_i \end{cases} \tag{19}$$

In accordance with the remaining node energy, the priority value is updated by time. Otherwise, energy will be diminished sooner by the end nodes with greater priority values. The controller updates the flows according to end node statistics. In our work, the controller updates the routing flows for the end nodes every time it loses 5 percent of its energy. The controller recomputes the EDC and updates the end nodes flow table.

6 Analysis

This section provides performance analysis of our proposed protocol *SDORP*. We will express mathematically the energy cost, the number of redundant packets and the sender waiting time. In OR, the major influence on the performance of *SDORP* is the number of active candidates assigned to the sender node in each hop. We will, therefore, calculate the probability of active candidate nodes for sender node before we evaluate the expected energy cost, the redundant packets and the sender waiting time. The sensor nodes are assumed to be asynchronous having a uniform distribution with a continuous interval of the same active period of length, say t, then the distribution of being active for any node is $\frac{1}{t}$ in $[a, a+t]$. We assume that the starting point of each interval is randomly selected within $[0, T]$. For a given node n_i, let there exist m_i neighbors with the maximum number of candidate nodes are $c_i = \left\lceil \sqrt{\left(1+\frac{m_i}{\pi}\right)} \right\rceil$. The selection of candidate nodes for the sender node n_i and their activation time is distributed by Eq. (20).

$$\sum_{x=0}^{c_i} \binom{c_i}{x} \left(\frac{1}{t}\right)^x \left(1-\frac{1}{t}\right)^{c_i-x} \tag{20}$$

The probability $\mathbb{P}(\wp = 0)$, where all candidate nodes assigned to the sender node n_i are in sleep state is given by Eq. (21).

$$\mathbb{P}_{(\wp=0)} = \frac{\left(1-\frac{1}{t}\right)^{c_i}}{\sum_{x=0}^{c_i} \binom{c_i}{x} \left(\frac{1}{t}\right)^x \left(1-\frac{1}{t}\right)^{c_i-x}} \tag{21}$$

The probability $\mathbb{P}(\wp = 1)$, where there is only one active candidate node for a sender node n_i is given by Eq. (22).

$$\mathbb{P}_{(\wp=1)} = \frac{\binom{c_i}{1} \left(\frac{1}{t}\right)^1 \left(1-\frac{1}{t}\right)^{c_i-1}}{\sum_{x=0}^{c_i} \binom{c_i}{x} \left(\frac{1}{t}\right)^x \left(1-\frac{1}{t}\right)^{c_i-x}} \tag{22}$$

The probability $\mathbb{P}(\wp < 2)$, where all candidate nodes assigned to the sender node n_i are in sleep state or only one active candidate node among them, is given by Eq. (23).

$$\begin{aligned} \mathbb{P}(\wp < 2) &= \mathbb{P}(\wp = 0) + \mathbb{P}(\wp = 1) \\ &= \frac{\sum_{x=0}^{1} \binom{c_i}{x} \left(\frac{1}{t}\right)^x \left(1-\frac{1}{t}\right)^{c_i-x}}{\sum_{x=0}^{c_i} \binom{c_i}{x} \left(\frac{1}{t}\right)^x \left(1-\frac{1}{t}\right)^{c_i-x}} \end{aligned} \tag{23}$$

The probability $\mathbb{P}(\wp \geq 2)$, where there are two or more active candidate nodes for a sender node n_i is given by Eq. (24).

$$\mathbb{P}_{(\wp\geq 2)} = 1 - \mathbb{P}_{(\wp<2)} \tag{24}$$

To generalize the probability of active candidate nodes $\mathbb{P}(\wp)$ for a sender node, the probability where there are exactly $\mathbb{P}(1 < \wp \leq c_i)$ active candidate nodes for sender node n_i is given by Eq. (25).

$$\mathbb{P}_{(1<\wp\leq c_i)} = \mathbb{P}_{(\wp=2)} + \mathbb{P}_{(\wp=3)} + \cdots + \mathbb{P}_{(\wp=c_i)} \tag{25}$$

with,

$$\mathbb{P}_{(\wp=2)} = \frac{\binom{c_i}{2} \left(\frac{1}{t}\right)^2 \left(1-\frac{1}{t}\right)^{c_i-2} - \mathbb{P}(\wp_1 \cap \wp_2)}{\sum_{x=0}^{c_i} \binom{c_i}{x} \left(\frac{1}{t}\right)^x \left(1-\frac{1}{t}\right)^{c_i-x}} \tag{26}$$

Suppose that the first node is active between $[a, a+t]$ since t is fixed for all active nodes. The second node is activated randomly at $[a_1, a_1+t]$, where $a < a_1$, and their joint activation last in the interval $[a_1, a+t]$. Then the probability of intersection of two active nodes is expressed in Eq. (27).

$$\mathbb{P}_{(\wp_1 \cap \wp_2)} = \mathbb{P}(\wp_1) \cdot \mathbb{P}_{(\wp_2|\wp_1)} = \frac{1}{t} \cdot \frac{1}{a+t-a_1} \tag{27}$$

Hence,

$$\mathbb{P}_{(\wp=2)} = \frac{\binom{c_i}{2} \left(\frac{1}{t}\right)^2 \left(1-\frac{1}{t}\right)^{c_i-2} - \frac{1}{t} \cdot \frac{1}{a+t-a_1}}{\sum_{x=0}^{c_i} \binom{c_i}{x} \left(\frac{1}{t}\right)^x \left(1-\frac{1}{t}\right)^{c_i-x}} \tag{28}$$

Similarly, we further elaborate the following equations.

$$\mathbb{P}_{(\wp=3)} = \frac{\binom{c_i}{3} \left(\frac{1}{t}\right)^3 \left(1-\frac{1}{t}\right)^{c_i-3} - \frac{1}{t} \cdot \frac{1}{a+t-a_1} \cdot \frac{1}{a+t-a_2}}{\sum_{x=0}^{c_i} \binom{c_i}{x} \left(\frac{1}{t}\right)^x \left(1-\frac{1}{t}\right)^{c_i-x}} \tag{29}$$

$$\mathbb{P}(\wp=c_i)=\frac{\binom{c_i}{c_i}\left(\frac{1}{t}\right)^{c_i}\left(1-\frac{1}{t}\right)^{c_i-c_i}-\prod_{h=1}^{c_i}\frac{1}{t}\cdot\frac{1}{a+t-a_h}}{\sum_{x=0}^{c_i}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}} \tag{30}$$

Therefore,

$$\begin{aligned}\mathbb{P}(1<\wp\le c_i)&=\frac{\sum_{x=2}^{\wp}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}-\mathbb{P}(\wp_1\cap\wp_2\cap\cdots\cap\wp_{c_i})}{\sum_{x=0}^{c_i}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}}\\&=\frac{\binom{c_i}{2}\left(\frac{1}{t}\right)^{2}\left(1-\frac{1}{t}\right)^{c_i-2}-\frac{1}{t}\cdot\frac{1}{a+t-a_1}}{\sum_{x=0}^{c_i}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}}\\&+\frac{\binom{c_i}{3}\left(\frac{1}{t}\right)^{3}\left(1-\frac{1}{t}\right)^{c_i-3}-\frac{1}{t}\cdot\frac{1}{a+t-a_1}\cdot\frac{1}{a+t-a_2}}{\sum_{x=0}^{c_i}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}}\\&+\cdots+\frac{\binom{c_i}{c_i}\left(\frac{1}{t}\right)^{c_i}\left(1-\frac{1}{t}\right)^{c_i-c_i}-\prod_{h=1}^{c_i}\frac{1}{t}\cdot\frac{1}{a+t-a_h}}{\sum_{x=0}^{c_i}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}}\end{aligned} \tag{31}$$

The probability of exactly $\mathbb{P}(1<\wp\le c_i)$ active candidate nodes and $\mathbb{P}(c_i-\wp)$ sleep candidate nodes for the sender node n_i is obtained by Eq. (32).

$$\begin{aligned}\mathbb{P}(1<\wp\le c_i)&=\frac{\sum_{x=2}^{\wp}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}-\frac{1}{t}\sum_{y=1}^{\wp-1}\prod_{h=1}^{y}\frac{1}{a+t-a_h}}{\sum_{x=0}^{c_i}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}}\\\mathbb{P}(c_i-\wp)&=1-\mathbb{P}(\wp>0)\end{aligned} \tag{32}$$

6.1 Expected Energy Cost

According to the *Energy Consumption Model* described in Subsection 4.2, the energy cost $E_{Tx}(i,j,k)$ for transmitting a packet of size k from n_i to n_j is formulated in Eq. (33). The energy cost $E_{Rx}(j,k)$ for receiving a data packet of size k by n_j is formulated in Eq.(34).

$$E_{Tx}(i,j,k)=\begin{cases}k\cdot E_{elec}+k\cdot\varepsilon_{fs}\cdot d_{i,j}^2 & d_{i,j}<d_*\\ k\cdot E_{elec}+k\cdot\varepsilon_{mp}\cdot d_{i,j}^4 & d_{i,j}\ge d_*\end{cases} \tag{33}$$

$$E_{Rx}(j,k)=k\cdot E_{elec} \tag{34}$$

As a matter of fact, in our proposed protocol *SDORP*, it is necessary to compute and sums the energy costs of transmitting and receiving the beacon packets, the ACK packets, and the data packets in order to compute the total energy costs of each transmission phase. For this reason, whenever a sender node n_i is ready to forward a packet to one of its candidate nodes, it will first transmit a beacon packet to its candidates. The beacon packet will be received by candidates if they are active and return the ACK packet to the sender. The sender node will know about their availability after receiving the ACK packet. Based on *SDORP* routing flow strategy, the sender node decides the final candidate among them and sends the data packet to it. $T_{EEC}(i,j,k,\wp)$ denotes the expected total energy cost when there are $\wp$ active candidates that are assigned to sender node n_i.

The energy cost of transmitting and receiving beacon packet is expressed in equation Eq. (35).

$$\begin{aligned}T_{BP}(i,j,k_b,\wp)&=E_{Tx}(i,j,k_b)+\sum_{j=0}^{\wp}E_{Rx}(j,k_b)\\&=k_b\begin{cases}E_{elec}+\varepsilon_{fs}\cdot d_{i,j}^2+\wp\cdot E_{elec} & d_{i,j}<d_*\\ E_{elec}+\varepsilon_{mp}\cdot d_{i,j}^4+\wp\cdot E_{elec} & d_{i,j}\ge d_*\end{cases}\end{aligned} \tag{35}$$

The energy cost of transmitting and receiving the ACK packet is expressed in Eq. (36).

$$\begin{aligned}T_{AP}(i,j,k_a,\wp)&=\sum_{j=0}^{\wp}E_{Tx}(j,i,k_a)+\sum_{j=0}^{\wp}E_{Rx}(i,k_a)\\&=k_a\left(\wp\cdot E_{elec}+\sum_{j=0}^{\wp}\begin{cases}E_{elec}+\varepsilon_{fs}\cdot d_{i,j}^2 & d_{i,j}<d_*\\ E_{elec}+\varepsilon_{mp}\cdot d_{i,j}^4 & d_{i,j}\ge d_*\end{cases}\right)\end{aligned} \tag{36}$$

The energy cost of transmitting and receiving the data packet is expressed in Eq. (37).

$$\begin{aligned}T_{DP}(i,j,k)&=T_{DP}(i,j,k)+R_{DP}(j,k)\\&=k\begin{cases}E_{elec}+\varepsilon_{fs}\cdot d_{i,j}^2+E_{elec} & d_{i,j}<d_*\\ E_{elec}+\varepsilon_{mp}\cdot d_{i,j}^4+E_{elec} & d_{i,j}\ge d_*\end{cases}\end{aligned} \tag{37}$$

Hence, the expected energy cost for $\wp$ active candidates that are assigned to the sender node n_i is formulated in Eq. (38).

$$T_{EEC}(i,j,k,\wp)=T_{BP}(i,j,k_b,\wp)+T_{AP}(i,j,k_a,\wp)+T_{DP}(i,j,k) \tag{38}$$

Finally, based on Eq. (38) the expected energy cost $T_{hop}(i,j)$ for each transmission phase is obtained by Eq. (39) in accordance with $\mathbb{P}(\wp=\chi)$ which is the probability of exactly $\wp\ge 1$ candidates that are assigned to the sender node n_i.

$$T_{hop}(i,j)=\sum_{\chi=1}^{c_i}\mathbb{P}(\wp=\chi)\cdot T_{EEC}(i,j,k,\chi) \tag{39}$$

Based on Eq. (39), the expected energy cost for delivering the data packet along the routing path $\mathbb{P}=\{n_1,n_2,\cdots,n_\varrho\}$ is obtained by Eq. (40) where ϱ denotes the number of nodes in the path $\mathbb{P}$ and $\mathbb{P}(\wp=\chi)$ is given by Eq. (32).

$$\begin{aligned}T_{path}&=\sum_{y}^{\varrho}T_{hop}(y,y+1)\\&=\sum_{y}^{\varrho-1}\sum_{\chi=1}^{c_y}\mathbb{P}(\wp=\chi)\cdot T_{EEC}(y,y+1,k,\chi)\end{aligned} \tag{40}$$

6.2 Waiting Time

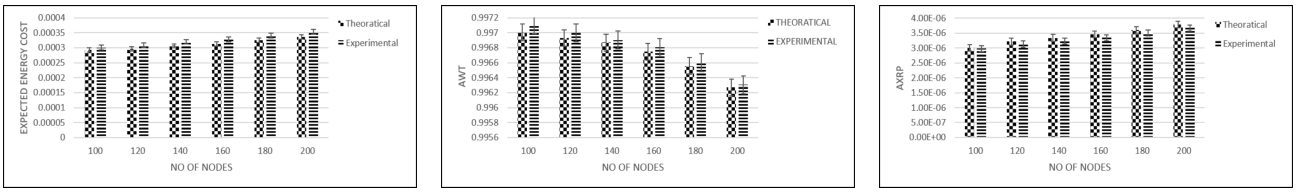
The average waiting time (AWT) represents the average number of times the sender has to wait until at least one of its candidates to wakes up and receives the data packet. Based on Eq. (21) we obtained the AWT for a packet in one hop in Eq. (41). Consequently, the AWT along the routing path $\mathbb{P}=\{n_1,n_2,\cdots,n_\varrho\}$ is obtained by Eq. (42).

$$AWT_i=\frac{\left(1-\frac{1}{t}\right)^{c_i}}{\sum_{x=0}^{c_i}\binom{c_i}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_i-x}} \tag{41}$$

$$AWT_{path}=\sum_{y=0}^{\varrho}AWT_y=\sum_{y=0}^{\varrho}\frac{\left(1-\frac{1}{t}\right)^{c_y}}{\sum_{x=0}^{c_y}\binom{c_y}{x}\left(\frac{1}{t}\right)^{x}\left(1-\frac{1}{t}\right)^{c_y-x}} \tag{42}$$

6.3 Redundant Packets

The sender node needs to identify which of its candidates are active to decide the final candidate. For that reason, a packet is broadcasted by the sender node and the packet is sent to its several candidates that are awoken. Based on *SDORP* routing flow strategy, one candidate is chosen among them to transmit the packet, and others will abort the received packet. The abortion of the packets is regarded as redundant. Based on the probability of redundant packets $\mathbb{P}(\wp\ge 2)$ Eq. (24), we obtained the average number of redundant packets in each hop by Eq. (43). Consequently,



(a) Expected energy cost (b) Waiting time (c) Redundant packets

Fig. 8: Validation of theoretical and experimental results

the average number of redundant packets along the path $\mathbb{P} = \{n_1, n_2, \cdots, n_\varrho\}$ is obtained by Eq. (44).

$$AxRP_i = \sum_{\chi=2}^{c_i} (\chi - 1) \cdot \mathbb{P}(\wp = \chi) \tag{43}$$

$$AxRP_{path} = \sum_{y=1}^{\varrho} \sum_{\chi=2}^{c_y} (\chi - 1) \cdot \mathbb{P}(\wp = \chi) \tag{44}$$

6.4 Validation of Analysis

We validate the expected energy cost, the average waiting time and the average redundant packets over the number of nodes by comparing the theoretical and experimental results. The accuracy of probability analysis is shown by comparing these results obtained from their average values.

Expected Energy Cost. Fig. 8a shows the feasibility of theoretical and experimental results of expected energy cost for each transmission phase by varying the number of nodes. It increases with an increase in the number of nodes. The reason behind is, the number of candidates for each node increases as the network size increases gradually.

Waiting Time. Fig. 8b shows the feasibility of theoretical and experimental results of average waiting time for each transmission phase by varying the number of nodes. The average waiting time gets higher with smaller number of candidates and gets lower with the greater number of candidates due to the variation in the network size.

Redundant Packets. Fig. 8c shows the feasibility of theoretical and experimental results of average number of redundant packets for each transmission phase by varying the number of nodes. The average number of redundant packets gets higher with the increase in the number of nodes. The reason behind is, large network size generates a higher number of candidates to each node.

7 Performance Evaluation

7.1 Simulation Scenario

The proposed protocol is tested and evaluated its performance via simulations with a simulator written in visual studio 2015 (C# WPF) [14], [15]. The source code and the documents are available online through the link: https://github.com/howbani/sdorp

To ensure the generality of the simulation results, we randomly deployed the nodes and the sink node which is located in the center of the square-shaped monitoring area. The controller is positioned in the sink node for our convenience in the simulation phase. We used one sink and one controller. Each node operates the BoX-MAC and nodes have the same active (1s)/sleep (2s) periods in the simulation phase. Each node is powered by a battery of 0.5J and consumes energy according to the *Energy Model* as explained in Subsection 4.2. We carry out the simulation 20 times in each parameter set to guarantee the accuracy of simulation results and all the results are obtained from their average values.

TABLE 4: Simulation Parameters

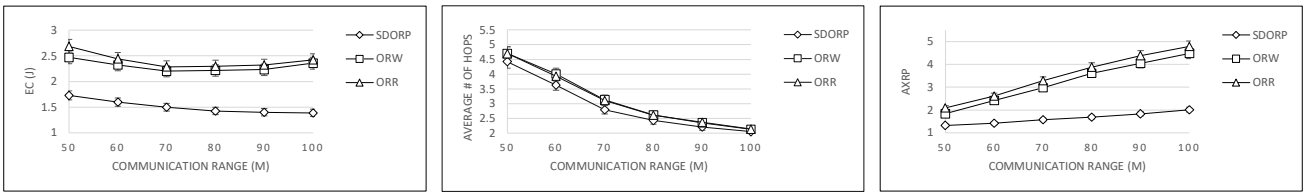
Parameter	Value
Number of nodes	Varies from 100 to 200
Communication range	Varies from $50m$ to $100m$
Active time	$1s$
Sleep time	$2s$
Packet rate	$1/0.1s$
Simulation time	$300s, 480s$
Packet size	$128\,bytes$

7.2 Evaluation Metrics

The following evaluation metrics are used and defined for the evaluation of routing protocol performance.

- **Energy Consumption**: The total energy consumption required to transmit and receive the packets from the source to destination in a given simulation time.
- **Average Number of Redundant Packets (AxRP)**: When a packet is broadcasted by a sender node and the packet is sent to its several awoken candidates. Among them, one candidate is chosen to transmit the packet, and the others, after coordination, will abort the received packet. The abortion of the packets is regarded as redundant. AxRP is the average number of redundant packets at a given simulation time.
- **Average Routing Distance Efficiency (RDE)**: The average routing distance efficiency is calculated by the ratio of the distance between each source to destination to the actual routing distance of the source to destination for a packet that proceeded along the path.
- **Average Waiting Time (AWT)**: The average number of times the sender has to wait until at least one of its candidates wakes up and receives the packet at a given simulation time.
- **Network Lifetime**: The time from the start of the simulation until the first node dies.

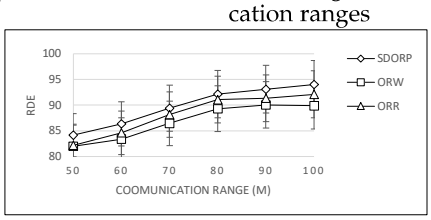
The simulation results presented below with the comparison of the two main protocols named *ORW* and *ORR* respectively.



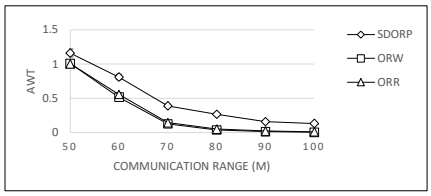
(a) Energy consumption over communication ranges

(b) Average # of hops over communication ranges

(c) AxRP over communication ranges



(d) RDE over communication ranges



(e) AWT over communication ranges

Fig. 9: Performance over communication ranges

ORW utilizes the novel opportunistic routing metric, *Expected Duty Cycled* Wakeups (EDC), where each node computes its EDC and forwarder set that is based on the number of neighbor nodes. ORW does not restrict the number of forwarders which causes the multiple receiver problems more often. It also did not consider the load balancing problem as mentioned in Subsection 4.3.

ORR an opportunistic routing protocol based on residual energy, that utilizes the *forwarder score* as a routing metric. The objective of ORR is to compute the finest forwarder that is based on *forwarder score* computations. ORR considered the load balancing problem by considering only residual energy while selecting forwarder set but they did not consider the transmission distance and the number of hops as mentioned in Subsection 4.3.

7.3 Results

We evaluate the performance of the network under various scenarios *Test 1* to *Test 5*. The network simulation parameters are mentioned in Table 4.

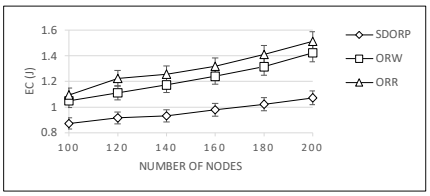
Test 1- Performance for Varying the Communication Ranges.

The communication range varies from $50m$ to $100m$. The simulation time to test this scenario is $480s$. The improved results of various performance metrics are shown in a graphical view Fig. 9a−9e.

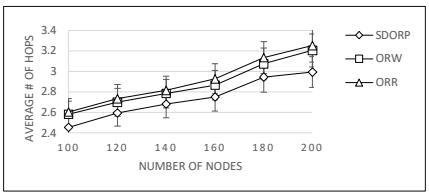
Fig. 9a presents the evaluation of energy consumption. Here we concluded that the energy consumption of the network decreases as the communication range increases and reaches $100m$. This shows much better results in the case of *SDORP* as compared to ORW and ORR. In the previous compared techniques the total energy consumption decreases until the communication range reaches $70m$, after that the energy consumption increases gradually. This happens because, the number of forwarders increases for each node as the communication range increases, which were not well controlled. Even though in ORR, the sink periodically calculates the maximum number of forwarders for each node and it also updates its EDC to achieve load balancing by considering residual energy among nodes. This implies that each node sends its current energy status to the sink in every update procedure. The sink recalculates the EDC and updates the node value. This consumes a lot of energy during calculation and cripples the network performance. ORW calculates its EDC in the initialization process. ORW did not consider to control the number of forwarders for each node. The adverse effect of duplicate packets affects both ORR and ORW. *SDORP* calculates its EDC in the controller to assign flows to each node. It achieved better energy consumption for the following reasons. 1) It calculates the expected number of hops so that packets are guided mostly through the paths with the minimum expected number of hops towards the sink. 2) *SDORP* forwarders are controlled using Eq. (19) such that few candidates are allowed to be forwarders for each node. This will help to reduce the negative effect of duplicate packets generated from multiple receivers.

Fig. 9b shows the comparison of average number of hops between ORW, ORR and *SDORP*. The results show that the average number of hops over communication ranges is better in the case of *SDORP* compared to the benchmarks. Fig. 9b also depicts that the average number of hops decreases as the communication range increases from $50m$ to $100m$. Since smaller number of hops implies shorter routing distance, which in turn consumes less energy. *SDORP* utilizes the EDC metric in the controller where it adds the *Expected Number of Hops Distribution* Eq. (14) parameter in the first term of EDC so that packets are regularly guided through the minimum expected number of hops towards the destination. ORW and ORR did not consider the number of hops.

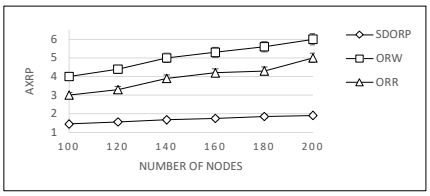
Fig. 9c illustrates the comparison of AxRP between ORW, ORR and *SDORP*. It shows that AxRP over the communication ranges achieved better results in the case of *SDORP* compared to the benchmarks because it generates a smaller number of redundant packets. Fig. 9c also depicts that the average number of redundant packets increases as the communication range increases from $50m$ to $100m$. The greater communication ranges lead the node to select more candidates as forwarders which in turn generates a higher number of redundant packets. *SDORP* utilizes Eq. (19) to control the number of forwarders for each node of the



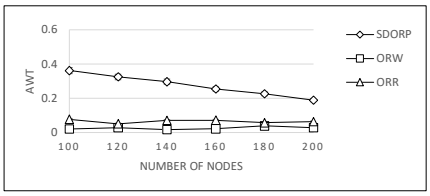
(a) Energy consumption over number of nodes



(b) Average # of hops over number of nodes



(c) AxRP over number of nodes



(d) AWT over number of nodes

Fig. 10: Performance over number of nodes

routing flows while in contrast, the ORR and ORW scenarios are different, as explained above.

Fig. 9d illustrates the performance of RDE over the communication ranges. *SDORP* achieves significant results and outperforms the benchmarks because packets are guided through the shortest distance nodes to the sender node. The reason behind that is, *SDORP* utilizes the *Transmission Distance Distribution* Eq. (8) while ORR and ORW did not consider the transmission distance.

Fig. 9e shows the comparison of AWT between the ORW, ORR and *SDORP*. The results show that the AWT over the communication ranges is acceptable for all protocols. *SDORP* has achieved satisfactory waiting time and less redundant packets. Fig. 9e also depicts that the AWT decreases as the communication range increases from $50m$ to $100m$ because the greater communication range allows each node to select more candidates as forwarders. The increment in the number of candidates for each node has both positive and negative effects. The negative effect is that the number of redundant packets increases which simultaneously affect the energy consumption of the network. In contrast, the positive effect is that the sender does not have to wait for a particular candidate node to wake up and receive the packet which in turn reduces the number of waiting times. The nature of the asynchronous duty-cycled WSN reflects these two contradictory effects. Therefore, *SDORP* is designed to counterbalance between the waiting time and the redundant packets.

Test 2- Performance for Varying the Number of Nodes.

The number of nodes varies from 100 to 200. The simulation time to test this scenario is $300s$ and the communication range is set to $80m$. The improved results of various performance metrics are shown in a graphical view Fig. 10a−10d.

Fig. 10a presents the evaluation of energy consumption for varying the number of nodes. Here we concluded that the energy consumption of the network increases as the number of nodes increases which leads to large network size. When the size of the network is large, the number of candidates assigned to each node increases because of this, it generates more redundant packets and consumes a lot of energy. However, the large network size increases the number of hops and enforces the packet to travel through longer paths that also consume a lot of energy. The Fig. 10a shows much better results in the case of *SDORP* due to the utilization of Eq. (19) to control the number of forwarders such that few candidates are allowed to be forwarders for each node and also minimize the number of hops towards the destination by using *Expected Number of Hops Distribution* Eq. (14) which outperforms the benchmarks.

Fig. 10b shows the comparison of average number of hops between ORW, ORR and *SDORP*. The number of hops increases with an increase in the number of nodes. The results show that the average number of hops over the number of nodes is better in the case of *SDORP* as compared to the benchmarks.

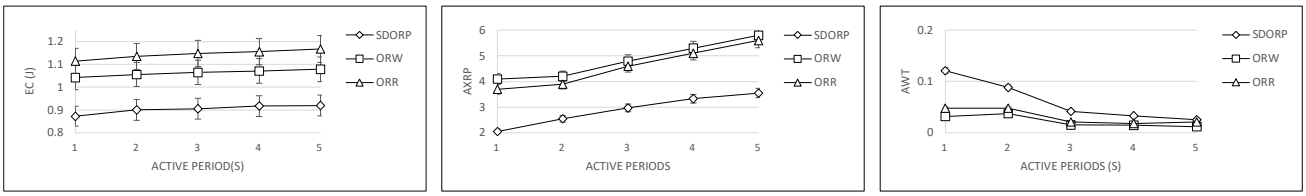
The AxRP is depicted in Fig. 10c. Here, we concluded that the AxRP increases with an increase in the number of nodes due to the higher network size that leads to generate more candidates to each node. The comparison illustrates the results of AxRP over the number of nodes are better in the case of *SDORP* as compared to the benchmarks for the same reason explained in Test 1.

Fig. 10d illustrates the comparison of AWT between the ORW, ORR and *SDORP* varying the number of nodes. We concluded that for each path AWT decreases with an increase in the number of nodes due to the larger network size that leads each node to select more candidates as forwarders. The results show that AWT over the number of nodes is acceptable for all protocols. In this scenario, *SDORP* has achieved a satisfactory waiting time and less redundant packets. The details are explained in Test 1.

Test 3- Performance for Varying the Wake-up Intervals.

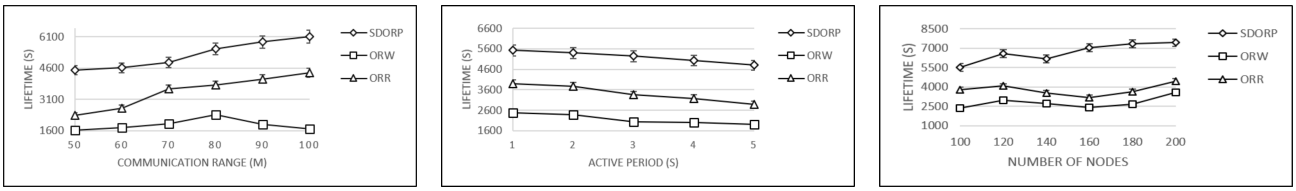
In this test, we studied the influence of the duty-cycle on the network performance with a variation in wake-up interval from $1s$ to $5s$ and each node sleeps for $2s$. The simulation time to test this scenario is $300s$, the communication range is set to $80m$ and the number of nodes are set to 100.

Fig. 11a shows the comparison of ORW, ORR and *SDORP* to evaluate energy consumption over wake-up intervals. The results show that energy consumption increases with an increase in the wake-up intervals. The energy consumption increases when the node stays active for a long



(a) Energy consumption over wake-up intervals (b) AxRP over wake-up intervals (c) AWT over wake-up intervals

Fig. 11: Performance over wake-up intervals



(a) Network lifetime over communication ranges (b) Network lifetime over wake-up intervals (c) Network lifetime over number of nodes

Fig. 12: Performance of network lifetime

time due to this reason it also increases redundant packets. *SDORP* shows much better performance compared to the benchmarks for the same reasons explained in Test 1.

The AxRP performance is evaluated in Fig. 11b. The results show that the AxRP increases with an increase in the wake-up intervals. The reason behind that is, nodes wake-up for a long time which simultaneously increases the probability of multiple receivers. *SDORP* achieves better results compared to the benchmarks for the same reasons explained in Test 1.

The results of the AWT evaluation over wake-up intervals are shown in Fig. 11c, where *SDORP* achieved an acceptable results along with the benchmarks. In this scenario, *SDORP* has achieved a satisfactory waiting time and less redundant packets. Here we have observed that the longer active time allows the nodes to have more candidates and for this reason, the AWT is reduced concurrently. For more details, refer to Test 1.

Test 4: Performance of Network Lifetime.

The test evaluated the performance of network lifetime over communication ranges and wake-up intervals respectively. The number of nodes are set to 100.

Fig. 12a shows the performance of network lifetime over communication range varying from $50m$ to $100m$. The benchmarks show poor performance as compared to *SDORP*. The reason behind that is, the ORW's impact on network lifetime is worst because it lacks the energy balancing approach as well as it generates too many redundant packets which consumed a lot of energy. ORR achieved better network lifetime than ORW because it controlled the maximum number of forwarders. ORR considers the energy balancing approach but it also needs to update its EDC metric periodically. In this scenario, it consumes much energy in every updating iteration. The details are explained in Test 1. *SDORP* utilizes the SDN concept to WSNs for flexible management as well as the three distributions in EDC metric to set the highest priority to those nodes which have greater remaining energy, less transmission distance to the sender in each transmission phase and minimum expected number of hops towards the destination.

Fig. 12b illustrates how wake-up intervals affect the performance of the network lifetime. The wake-up intervals vary from $1s$ to $5s$ and the node sleeps time is set to $2s$. The communication range is set to $80m$. The following observations can be concluded based on the results of Fig. 12b. First, the network lifetime decreases with an increase in the wake-up intervals. Second, when the node is active for a long time, it negatively affects the probability of multiple receivers which generates a large number of redundant packets as well. *SDORP* achieves much better results compared to the benchmarks.

The network lifetime is evaluated by varying the number of nodes and the results are depicted in Fig. 12c. *SDORP* achieved promising results compared to the benchmarks. This is due to utilization of efficient network management and effective load balancing approach among the nodes. ORR attained better results compare to ORW by considering the energy parameter. While ORW did not consider the load balancing at all and the nodes deplete their energy quickly.

Test 5: Impact of Control Variables.

The test evaluated the performance of average number of hops and network lifetime varying control variables. The number of nodes are set to 100, the communication range varies from $50m$ to $100m$ and the node active/sleep period is set to $1/2s$.

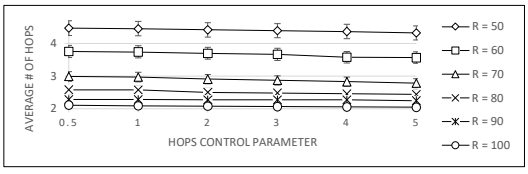


Fig. 13: Average # of hops over hops control variable

The attributes control variables (λ_τ, $\lambda_\mathcal{H}$, λ_Φ) can be modified to meet the application requirements. The link

quality estimation is based on the average result of the three attributes that is considered in the first term of the EDC. Any control variable will enhance the influence of the respective distribution by increasing the value. The performance impact of control variables is evaluated in two parts. First, we evaluated the average number of hops varying the control variable $\lambda_{\mathcal{H}}$, it varies from 0.5 to 5. While the residual energy and transmission distance control variables (λ_{Φ}, λ_{τ}) are set to 0.2. The packet rate is set to $1/1s$.

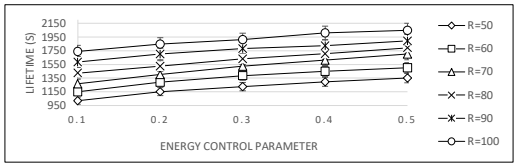


Fig. 14: Network lifetime over energy control variable

Fig. 13 shows that as the value of $\lambda_{\mathcal{H}}$ increases the average number of hops decreases because the $\lambda_{\mathcal{H}}$ control variable prioritize the node that has minimum number of hops to the sink. Second, we evaluated the network lifetime varying the control variable λ_{Φ} from 0.1 to 0.5. While the values of other control variables λ_{τ} and $\lambda_{\mathcal{H}}$ are set to 0.2 and initial energy is set to 0.1J. Each node sends 1 packet/0.1s. Fig. 14 shows that the impact of energy control variable λ_{Φ} on the network lifetime.

8 Conclusion

A deep and detailed study of opportunistic routing and software-defined wireless sensor networks gives a great deal of an improved solution to traditional opportunistic routing protocols. Opportunistic routing works well with asynchronous duty-cycled MAC protocols that address some issues: First, the sender waiting time where the sender must wait till its receiver wakes up. Second, the duplication of packets when multiple candidates assign to each node reduces the waiting time, but the multiple receiver wakes up at the same time that causes the packet duplication. These two problems are contradictory to each other. Third, the load balancing among the nodes is required to improve the overall network performance. The proposed *SDORP* protocol utilizes the SDN approach towards opportunistic routing which separates the control plane and data plane. The benefit of the SDN approach is to provide flexible management in WSNs which enables centralized control of the whole network to simplify the deployment of on-demand network-wide management protocols and applications. The protocol addresses the above-mentioned problems by including the heuristic function that combines three attributes, transmission distance, expected number of hops and residual energy for each node. The controller computes the EDC with the integration of the average attribute term for each node in the network. The evaluation of our proposed protocol with simulation results shows better performance as compared to benchmark solutions.

Acknowledgments

This paper is supported by the "Fundamental Research Funds for the Central Universities NO. WK2150110007" and by the National Natural Science Foundation of China (NO. 61772490, 61472382, 61472381 and 61572454).

References

[1] X. Zhang, L. Tao, F. Yan, and D. K. Sung, "Shortest-latency opportunistic routing in asynchronous wireless sensor networks with independent duty-cycling," *IEEE Transactions on Mobile Computing*, 2019.

[2] A. Pegatoquet, T. N. Le, and M. Magno, "A wake-up radio-based mac protocol for autonomous wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 27, no. 1, pp. 56–70, 2018.

[3] A. Karaagac, E. De Poorter, and J. Hoebeke, "In-band network telemetry in industrial wireless sensor networks," *IEEE Transactions on Network and Service Management*, 2019.

[4] J. Huang and B.-H. Soong, "Cost-aware stochastic compressive data gathering for wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1525–1533, 2018.

[5] I. Haque, M. Nurujjaman, J. Harms, and N. Abu-Ghazaleh, "Sd-sense: An agile and flexible sdn-based framework for wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1866–1876, 2018.

[6] H. Kim and N. Feamster, "Improving network management with software defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114–119, 2013.

[7] Y. Jararweh, M. Al-Ayyoub, E. Benkhelifa, M. Vouk, A. Rindos *et al.*, "Software defined cloud: Survey, system and evaluation," *Future Generation Computer Systems*, vol. 58, pp. 56–74, 2016.

[8] S. Gao, Y. Zeng, H. Luo, and H. Zhang, "Scalable control plane for intra-domain communication in software defined information centric networking," *Future Generation Computer Systems*, vol. 56, pp. 110–120, 2016.

[9] J. Wang, Y. Miao, P. Zhou, M. S. Hossain, and S. M. M. Rahman, "A software defined network routing in wireless multihop network," *Journal of Network and Computer Applications*, vol. 85, pp. 76–83, 2017.

[10] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "Fragmentation-based distributed control system for software-defined wireless sensor networks," *IEEE transactions on industrial informatics*, vol. 15, no. 2, pp. 901–910, 2018.

[11] T. Dinh, Y. Kim, T. Gu, and A. V. Vasilakos, "An adaptive low-power listening protocol for wireless sensor networks in noisy environments," *IEEE systems journal*, vol. 12, no. 3, pp. 2162–2173, 2017.

[12] E. Ghadimi, O. Landsiedel, P. Soldati, S. Duquennoy, and M. Johansson, "Opportunistic routing in low duty-cycle wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 4, p. 67, 2014.

[13] J. So and H. Byun, "Load-balanced opportunistic routing for duty-cycled wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 16, no. 7, pp. 1940–1955, 2016.

[14] A. Hawbani, X. Wang, L. Zhao, A. Al-Dubai, G. Min, and O. Busaileh, "Novel architecture and heuristic algorithms for software-defined wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 28, no. 6, pp. 2809–2822, 2020.

[15] A. Hawbani, X. Wang, A. Abudukelimu, H. Kuhlani, Y. Al-sharabi, A. Qarariyah, and A. Ghannami, "Zone probabilistic routing for wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 728–741, 2018.

[16] S. Costanzo, L. Galluccio, G. Morabito, and S. Palazzo, "Software defined wireless networks (sdwn): Unbridling sdns," in *European workshop on software defined networking*, 2012, pp. 1–6.

[17] S. Shanmugapriya and M. Shivakumar, "Context based route model for policy based routing in wsn using sdn approach," in *BGSIT National Conference on Emerging Trends in Electronics and Communication*, 2015.

[18] T. Luo, H.-P. Tan, and T. Q. Quek, "Sensor openflow: Enabling software-defined wireless sensor networks," *IEEE Communications letters*, vol. 16, no. 11, pp. 1896–1899, 2012.

[19] T. Miyazaki, S. Yamaguchi, K. Kobayashi, J. Kitamichi, S. Guo, T. Tsukahara, and T. Hayashi, "A software defined wireless sensor network," in *2014 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2014, pp. 847–852.

[20] A. De Gante, M. Aslan, and A. Matrawy, "Smart wireless sensor network management based on software-defined networking," in *2014 27th Biennial Symposium on Communications (QBSC)*. IEEE, 2014, pp. 71–75.
[21] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "Reprogramming wireless sensor networks by using sdn-wise: A hands-on demo," in *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2015, pp. 19–20.
[22] Y. Duan, W. Li, X. Fu, Y. Luo, and L. Yang, "A methodology for reliability of wsn based on software defined network in adaptive industrial environment," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 74–82, 2017.
[23] R. Huang, X. Chu, J. Zhang, and Y. H. Hu, "Energy-efficient monitoring in software defined wireless sensor networks using reinforcement learning: A prototype," *International Journal of Distributed Sensor Networks*, vol. 11, no. 10, p. 360428, 2015.
[24] B. T. De Oliveira, L. B. Gabriel, and C. B. Margi, "Tinysdn: Enabling multiple controllers for software-defined wireless sensor networks," *IEEE Latin America Transactions*, vol. 13, no. 11, pp. 3690–3696, 2015.
[25] W. Xiang, N. Wang, and Y. Zhou, "An energy-efficient routing algorithm for software-defined wireless sensor networks," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7393–7400, 2016.
[26] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3. IEEE, 2002, pp. 1567–1576.
[27] S. Kulkarni, A. Iyer, and C. Rosenberg, "An address-light, integrated mac and routing protocol for wireless sensor networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, no. 4, pp. 793–806, 2006.
[28] D. Liu, M. Hou, Z. Cao, J. Wang, Y. He, and Y. Liu, "Duplicate detectable opportunistic forwarding in duty-cycled wireless sensor networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 24, no. 2, pp. 662–673, 2016.
[29] A. Hawbani, X. Wang, Y. Sharabi, A. Ghannami, H. Kuhlani, and S. Karmoshi, "Lora: Load-balanced opportunistic routing for asynchronous duty-cycled wsn," *IEEE Transactions on Mobile Computing*, vol. 18, no. 7, pp. 1601–1615, 2018.
[30] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 95–107.
[31] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*. ACM, 2006, pp. 307–320.
[32] D. Moss and P. Levis, "Box-macs: Exploiting physical and link layer boundaries in low-power networking," *Computer Systems Laboratory Stanford University*, vol. 64, no. 66, p. 120, 2008.
[33] S. Biswas and R. Morris, "Exor: opportunistic multi-hop routing for wireless networks," in *ACM SIGCOMM computer communication review*, vol. 35, no. 4. ACM, 2005, pp. 133–144.
[34] Z. Zhong, J. Wang, S. Nelakuditi, and G.-H. Lu, "On selection of candidates for opportunistic anypath forwarding." *Mobile Computing and Communications Review*, vol. 10, no. 4, pp. 1–2, 2006.
[35] X. Mao, X.-Y. Li, W.-Z. Song, P. Xu, and K. Moaveni-Nejad, "Energy efficient opportunistic routing in wireless networks," in *Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*. ACM, 2009, pp. 253–260.
[36] M. Zorzi and R. R. Rao, "Geographic random forwarding (geraf) for ad hoc and sensor networks: energy and latency performance," *IEEE transactions on Mobile Computing*, vol. 2, no. 4, pp. 349–365, 2003.
[37] A. Darehshoorzadeh and L. Cerda-Alabern, "Distance progress based opportunistic routing for wireless mesh networks," in *2012 8th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2012, pp. 179–184.
[38] J. So and H. Byun, "Opportunistic routing with in-network aggregation for asynchronous duty-cycled wireless sensor networks," *Wireless networks*, vol. 20, no. 5, pp. 833–846, 2014.
[39] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd annual Hawaii international conference on system sciences*. IEEE, 2000, pp. 10–pp.
[40] W. J. Ewens and G. R. Grant, *Statistical methods in bioinformatics: an introduction*. Springer Science & Business Media, 2006.
[41] M. U. Farooq, X. Wang, A. Hawbani, A. Khan, A. Ahmed, and F. T. Wedaj, "Torp: Load balanced reliable opportunistic routing for asynchronous wireless sensor networks," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2020, pp. 1384–1389.

Muhammad Umar Farooq Qaisar received his B.S. degree from International Islamic University Islamabad, Pakistan in 2012 and received his M.S. degree in Computer Science and Technology from University of Science and Technology of China in July 2017. He is doing Ph.D. in Computer Science and Technology from University of Science and Technology of China. His main research interests include WSN, SDN and Security.

Xingfu Wang received the B.S. degree in electronic and information engineering from Beijing Normal University of China in 1988, and the M.S. degree in computer science from the University of Science and Technology of China in 1997. He is an associate professor in the School of Computer Science and Technology, University of Science and Technology of China. His current research interests include Information Security, Data Management and WSN.

Ammar Hawbani is an associate professor of networking and communication algorithms in the School of Computer Science and Technology at the University of Science and Technology of China, China. He received the B.S., M.S. and Ph.D. degrees in Computer Software and Theory from the University of Science and Technology of China (USTC), Hefei, China, in 2009, 2012 and 2016, respectively. From 2016 to 2019, he worked as Postdoctoral Researcher in the School of Computer Science and Technology at USTC. His research interests include IoT, WSNs, WBANs, WMNs, VANETs, and SDN

Liang Zhao [M] is an associate professor at Shenyang Aerospace University, China. He received his Ph.D. degree from the School of Computing at Edinburgh Napier University in 2011. Before joining Shenyang Aerospace University, he worked as an associate senior researcher in Hitachi (China) Research and Development Corporation from 2012 to 2014. His research interests include VANETs, SDVN, FANETs, and WMNs.

Ahmed ADubai [SM] is Professor of Networking and Communication Algorithms in the School of Computing at Edinburgh Napier University, UK. He received the Ph.D. degree in Computing from the University of Glasgow in 2004. His research interests include Communication Algorithms, Mobile Communication, Internet of Things, and Future Internet.

Omar Busaileh received his B.S. degree in Electronic and Information Engineering from Hefei University of Technology. Currently a Master student in the School of Computer Science and Technology at USTC. His research interests mainly include WSNs, IoTs and SDN.