IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 30, 2017, accepted March 6, 2017, date of publication March 20, 2017, date of current version April 24, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2684539

A Performance Simulation Tool for the Analysis of Data Gathering in Both Terrestrial and Underwater Sensor Networks

MUKHTAR GHALEB[1,2], EMAD FELEMBAN[3], SHAMALA SUBRAMANIAM[4], ADIL A. SHEIKH[1,6], AND SAAD BIN QAISAR[5], (Senior Member, IEEE)

[1]College of Sciences and Arts, Bisha University, Al Namas 21413, Saudi Arabia
[2]Faculty of Computer and Information Technology, Sana'a University, Sana'a, Yemen
[3]Computer Engineering Department, Umm Al-Qura University, Makkah 21955, Saudi Arabia
[4]Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 Serdang, Malaysia
[5]School of Electrical Engineering and Computer Science, National University of Sciences and Technology, Islamabad 44000, Pakistan
[6]Science and Technology Unit, Umm Al-Qura University, Makkah, Saudi Arabia

Corresponding author: M. Ghaleb (mukhtar_97@hotmail.com)

This work was supported in part by the NSTIP Strategic Technologies Program under Grant 11-INF1688-10 and in part by the King Abdul Aziz City of Science and Technology of the Kingdom of Saudi Arabia.

ABSTRACT Wireless sensor networks (WSNs) have greatly contributed to human-associated technologies. The deployment of WSNs has transcended several paradigms. Two of the most significant features of WSNs are the intensity of deployment and the criticalness of the applications that they govern. The tradeoff between volume and cost requires justified investments for evaluating the multitudes of hardware and complementary software options. In underwater sensor networks (USNs), testing any technique is not only costly but also difficult in terms of full deployment. Therefore, evaluation prior to the actual procurement and setup of a WSN and USN is an extremely important step. The spectrum of performance analysis tools encompassing the test-bed, analysis, and simulation has been able to provide the prerequisites that these evaluations require. Simulations have proven to be an extensively used tool for analysis in the computer network field. A number of simulation tools have been developed for wired/wireless radio networks. However, each simulation tool has several restrictions when extended to the analysis of WSNs. These restrictions are largely attributed to the unique nature of each WSN within a designated area of research. In addition, these tools cannot be used for underwater environments with an acoustic communication medium, because there is a wide range of differences between radio and acoustic communications. The primary purpose of this paper is to present, propose, and develop a discrete event simulation designed specifically for mobile data gathering in WSNs. In addition, this simulator has the ability to simulate 2-D USNs. This simulator has been tailored to cater to both mobile and static data gathering techniques for both topologies, which are either dense or light. The results obtained using this simulator have shown an evolving efficient simulator for both WSNs and USNs. The developed simulator has been extensively tested in terms of its validity and scope of governance.

INDEX TERMS Wireless sensor networks, underwater sensor networks, acoustic channel, discrete-event simulation, multi-hop data gathering and mobile data gathering.

I. INTRODUCTION

Sensor networks have positioned themselves as a credible member within the revolutionary network technologies. This is primarily due to the enormous benefits of sensor networks in various fields. It is becoming increasingly difficult to ignore the successful use of sensor networks in different areas that affect human civilization, such as health care, infrastructure, earthquakes, coral reefs, volcanoes and monitoring systems. Sensors are becoming more intelligent, smaller, lighter and cheaper. The use of sensor networks is increasing by leaps and bounds. Temperature, salinity, pressure level and flow are among the most measured parameters. These sensors, due to their design and deployment objectives, suffer from many limitations, such as limited energy power and storage capacity. It is generally not feasible to replace the batteries or even recharge them due to hazardous environments [1], [2].

Considering the various research areas for sensing the environment and temporarily storing the recorded data, the energy consumption for passing the data from each sensor to the base station *BS* is a pertinent issue in WSNs [3]. This is because the data must be transmitted via multiple nodes prior to reaching the *BS* for subsequent processing. This process is called the data gathering process. In this process, each sensor node is responsible for gathering the raw data from its surrounding environment. Subsequently, the data are temporarily stored before being forwarded to the *BS*.

An underwater sensor network [4] consists of a number of sensors that can perform various functions. In recent years, USNs have been widely used in underwater environments for a wide range of applications, such as pollution monitoring, health monitoring of marine organisms, monitoring coral reefs, and so forth.

Simulation tools are an effective method for evaluating algorithms and protocols at different stages of a project, such as design, development and implementation [5]. The available simulators have different features and distinct characteristics. An example is the NS2 simulator, which is a very popular network simulator; however, it is complicated by a long learning curve and requires advanced skills to perform meaningful and repeatable simulations [6]. In addition, NS2 has a number of restrictions on the energy model, packet formats and MAC protocols; lacks an application model; and does not scale well for WSNs [7].

The issue of cost among simulators has a great impact on determining the performance analysis tool. In general, some of the simulators are free, whereas others are commercial. Due to the unique nature of sensor networks, such as the vast number of sensors, and the energy model used, the eligibility for performance testing of the sensor network is different. Depending on the design purpose, simulators can be categorized into general purpose and specific purpose simulators [5]. Some of the most prominent general purpose simulators used are Network Simulator 2 (NS2), Objective Modular Network Test-bed (OMNET++), JavaSim (J-SIM), Optimized Network Engineering Tool (OPNET), Global Mobile Information Systems Simulation Library (GloMoSim), QualNet and Matrix Laboratory (MATLAB). They offer a specialized extension for WSNs. In contrast, some efforts have been devoted to developing simulators specialized for WSNs, such as Sensor Simulator (SensorSim), Castalia, VisualSense, Prowler, and Sensor Environment and Network Simulator (SENS). In this study, we propose a discrete event simulator that is capable of simulating the data gathering process in both underwater and terrestrial sensor networks.

The remainder of this paper is organized as follows. Section 2 reviews the general data gathering schemes. Section 3 presents the related works, followed by the proposed performance analysis tool in Section 4. Section 5 presents the validation and verification. The results and discussion are presented in Section 6. Finally, Section 7 concludes the article and states directions for future work.

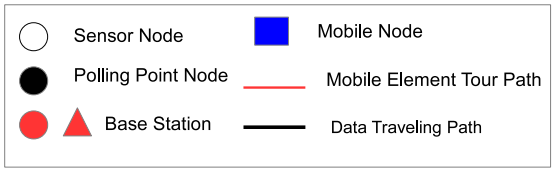


FIGURE 1. Figure legends.

II. DATA GATHERING REVIEW

Analysis of the data gathering tools is fundamental to be complemented with the details of the data gathering process. In this section, we discuss the nature of data gathering, which is the basis for developing the proposed simulator. We categorize this discussion into three main parts: multi-hop data gathering, mobile data gathering and hybrid data gathering. Figure 1 presents the legends for the figures presented in this article.

A. MULTI-HOP DATA GATHERING

Researchers have focused on studying the issue of energy savings using the multi-hop data gathering scheme. In this scheme, data packets do not traverse directly between nodes and the *BS* as it results in faster energy depletion of the nodes. In the multi-hop data gathering scheme, sensors are able to send their data to the *BS* through other sensors within their range and on the way to the *BS*. There are several data gathering approaches that address maximizing the lifetime in WSNs [8]–[18].

One research direction focuses on data aggregation techniques that merge the data while they are transported among the sensor nodes. This trend reduces the amount of data transport by avoiding double counting sensor readings and hence maximizes the network lifetime [8], [9]. Another approach focuses on routing-based clustering techniques in which each sensor is grouped into a cluster and sends its data to the *BS* through the cluster head only [10], [11]. To improve clustering techniques, more constraints are considered to select the cluster head, such as residual energy, distance from the *BS* and number of consecutive rounds in which a node is not selected as the cluster head [12]–[14]. In addition, more issues such as load balancing [15], [16], heterogeneous energy [17] and power management (i.e., sleep and wake up protocol) [18] were also considered with routing to enhance the network lifetime.

However, the data packets must visit multiple nodes before reaching the *BS*, which leads to increased communication among nodes and hence increased energy consumption. Thus, some nodes will die earlier and cause the *BS* to become unreachable, particularly those nodes located near the *BS* due to serving other nodes. In addition, the multi-hop scheme does not support sparse networks that are out of the range of the main network. Figure 2(a) illustrates the multi-hop data gathering approach, in which each child node sends its data to the parent node based on the shortest path tree and subsequently until reaching the tree root (i.e., *BS*).

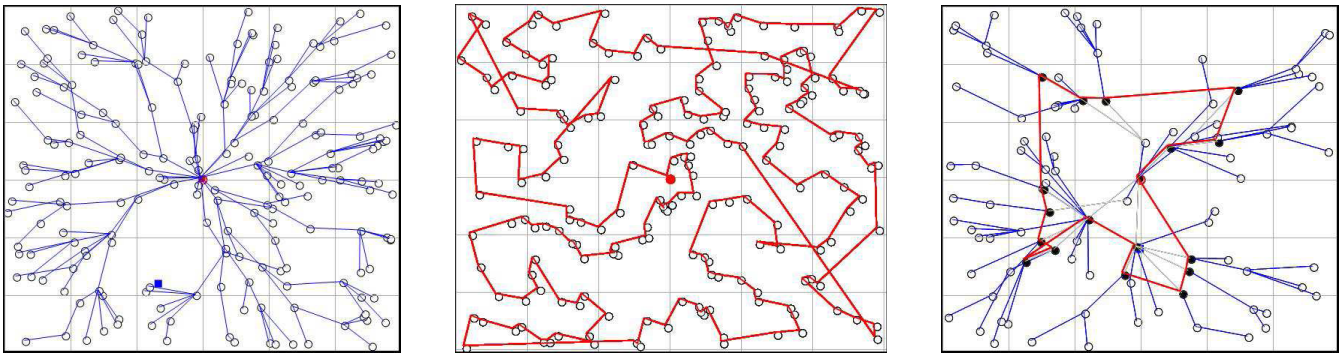


FIGURE 2. Illustrates the data gathering approaches. (a) Multi-hop data gathering. (b) Mobile data gathering. (c) Hybrid data gathering.

B. MOBILE DATA GATHERING

Data gathering saves a remarkable amount of energy when an appropriate technique is deployed [3]. Over the past few years, many researchers have employed mobile elements (*MEs*) to gather data via short-range communications [19]–[22]. These *MEs* are responsible for gathering data directly from sensors, which minimizes/eliminates the traversed packets between nodes, thereby resulting in remarkable energy savings. Figure 2(b) illustrates the mobile data collector tour path, which roams to reach every sensor node in the deployment area starting from the *BS* located at the center of the deployment area. Then, it consequently visits each sensor one by one before returning to the *BS*, which is exactly the legacy travel salesman problem (*TSP*). In addition, sparse networks in the mobile-based approach are no longer considered to be a problem due to covering the entire deployment field by the *ME*.

However, this approach is limited to the buffering capacity and energy level of each sensor node, which should wait a long time until the *ME* reaches them to upload their data. In addition, this leads to higher latency on data gathering due to the limited velocity of the mobile collector, which is 0.1-2 m/s [23] compared to several hundred meters using a multi-hop approach [24], [25].

C. HYBRID APPROACH

Over the past few years, researchers have focused on combining the multi-hop forwarding scheme with the use of the mobile data gathering scheme. This combination leads to ensuring that a balance between energy consumption and latency of the data gathering tour path [3], [26]–[30] is achieved. In this scheme, a trade-off between the latency of data gathering and energy consumption is achieved by selecting sensor nodes as caching points (polling points *PPs*), which shorten the tour length of the mobile collector. These nodes are responsible for local data aggregation from other nodes via a multi-hop manner and communicate directly with the mobile data collector when reaching their vicinity. The correlation between energy consumption and a node's hop count to the nearest caching point affects both peers and ultimately affects the network lifetime [31]. Figure 2(c) illustrates the tour path of a mobile data collector visiting all *PPs*, including the *BS*. Each *PP* gathers the data from affiliated sensors via multi-hop. In this scheme, the balance between the latency of data gathering and energy consumption maximizes the lifetime of the entire network.

TABLE 1. Comparison between three data gathering schemes.

Scheme	Data Carrier	Power Consumption	Latency	Support Sparse Network
Multi-hop data gathering	Relay nodes/ communication	Remarkably high due to subsequently forwarding data packets to the base station	Very low due to higher speed of forwarding data packets	Not supported due to coverage problems
Mobile Data Gathering	Mobile elements	Remarkably low due to short-range communication with mobile element	Remarkably high due to limited velocity of mobile element	Yes supported using the mobile element to reach sparse networks
Hybrid Data Gathering	Mobile element + relay nodes	Trade-off that is restricted to hop count for data to be forwarded to reach nearest polling point	Trade-off that is restricted to the tour path of mobile element	Yes supported using the mobile element to reach sparse networks

Table 1 presents comparisons among three data gathering schemes, namely, multi-hop, mobile and hybrid. From the above comparison, applying ME for data gathering could impact the lifetime of the entire sensor network, which remarkably reduces the energy consumption. In addition, combining ME with the multi-hop scheme minimizes the tour length of ME, which is restricted to the hop count.

III. RELATED WORK

Network simulation is one of the available evaluation methodologies that has the ability to save time and money in the field of computer networks [32]. The deployment of

sensor networks in a geographical area is costly, particularly in underwater environments. In this sense, it is important to ensure the performance of the network to avoid excess costs and time [33]. Thus, simulators are widely used for developing and testing network protocols, topologies, environmental constraints and scenarios. Moreover, it avoids investigating/evaluating the routing protocol in the real-world environment/on-site, which consumes time and money and cannot cover all the expected conditions. Thus, simulators are preferred to test and validate it first through extensive simulations with different topologies and different network limitations. After many years of developing network simulators by different researchers, there are many simulators available to end users, either free or as a commercial product. NS2, for example, is a famous network simulator that was originally constructed for wired networks [34] and then extended to address wireless networks. Thus, simulating sensor networks using the NS2 simulator is not preferred, which takes increasingly longer running times, particularly when the number of nodes is greater than 80 [35]–[37]. Castalia is another simulator tool constructed for WSNs based on the OMNET++ network simulator, which suffers from increases in execution time when the number of nodes exceeds 501 sensor nodes [33]. However, some of the simulators are general and have limitations for wireless networking for many participants of sensor nodes [38]. In addition, not all of the simulators are constructed specifically to address WSNs. Most of the simulators are constructed for fixed networks and then extended for WSNs, such as NS2, OMNET++ (Ext. SensorSim and Castalia) and OPNET [39]. Furthermore, the acoustic model is not supported in most WSN simulators. The next subsections present more details about network simulators, which are compared using many factors, such as their availability as free products to researchers, their ability to handle data gathering in dense WSNs, the complexity of working with the tools (i.e., easy to use), documentation availability, and good module support for WSNs.

- **Network Simulator-2 (NS2):** One of the most known general purpose network simulators is NS2. This simulator is an object-oriented discrete-event simulator that is implemented using a combination between C++ and OTcl. NS2 is primarily used in studies of routing, TCP and multicast protocol, lacking support on a framework for WSNs. It was originally constructed for fixed networks and then extended to address WSNs [36]. Due to the limitations for supporting WSN protocols in NS2, it is not a preferred network simulator for many researchers [38]. Moreover, it is not designed to cover all aspects of a WSN because it does not scale well when the simulation exceeds 100 sensor nodes [36]. In addition, NS2 is text based and lacks GUI capabilities for creating topologies and scenarios with limited visualization for node activities, such as transmission and reception of packets. However, NS2 suffers from being large and complicated for users [40]. In addition, there is a lack of customization in available applications in NS2. Packet formats, energy models, MAC protocols, and the sensing hardware models all differ from those found in most sensors [41]. Moreover, the primary disadvantage of NS2 is its limited scalability in terms of memory usage and simulation run time [35], [42]. In addition, WSN simulation is not easily supported by NS2, although many researchers are currently attempting to modify NS2 toward better WSN simulations [39]. Overall, based on the above, NS2 is not a preferred simulation platform for data gathering in WSNs, particularly with an increase in the deployed nodes (dense network) as it is not easily supported by NS2. Thus, SensorSim and Mannasim are extension simulators based on NS2 to address some of the aforementioned problems by adding several application models and several routing protocols such as LEACH. Furthermore, AquaSim [43] is built on top of NS2 to provide the basics for acoustic communications. The AquaSim simulation tool follows the object-oriented design style of NS2, supports three-dimensional modeling and can simulate acoustic signal attenuation, propagation delays and packet collisions. Because NS2 NAM lacks tools to visualize 3D underwater networks, AquaSim uses an additional tool, the Aqua-3D animator.
- **Network Simulator-3 (NS3):** Another known simulator is Network Simulator 3 (NS3), which is built on pure C++. This simulator was developed in 2007 to cover the limitations in NS2. The development goal of NS3, which is in the early stage, is to improve the simulation performance of NS2 [32]. A few simulation models exist, such as protocols (AODV, DSDV, OLSR and CLICK), utilities (config-store, flow-monitor, etc.), devices (LTE, WIMAX, MESH, WIFI, CSMA, etc.), energy, mobility, propagation and so forth. [44]. NS3 does not include the ZigBee (IEEE 802.15.4) protocol working with sensor networks [44]. Thus, NS3 is not an appropriate network simulator for wireless sensor networks.
- **Global Mobile System Simulator (GloMoSim):** GloMoSim [45] is a discrete-event simulator that was developed in 1998, and it was designed specifically for mobile wireless networks. This simulator is designed using the parallel discrete-event simulation capability provided by the Parallel Simulation Environment for Complex Systems - Parsc, which is a C-based simulation language. It has the ability to work with parallel environments. However, because WSNs are data-centric networks, many sensor network applications cannot be accurately simulated in GloMoSim [39]. GloMoSim is effective for simulating IP networks, but it is not capable of simulating any other types of networks and is limited to the network protocol [46]. However, there are many problems associated with GloMoSim, such as lack of support for any events occurring in the outside environment, all events must be generated from another node in the network, and it stopped releasing updates in 2000 [38]. In addition, there is no specific routing protocol

provided for sensor networks, and it is difficult for users to simulate large sensor networks [37]. The Aqua-Glomo simulator [47] is built on top of GloMoSim. In addition, GloMoSim is now updated as a commercial product called QualNet.

- **Optimized Network Engineering Tool (OPNET):** OPNET is an object-oriented, discrete-event, general purpose simulator that was launched in 1987 as the first commercial simulator originally designed for simulating fixed networks. OPNET does support a few available protocols that are implemented in NS2 or GloMoSim [38]. The OPNET modeler and its wireless suite are available free of charge to researchers who apply to their university program [48]. Similar to NS2, an increase in the number of nodes leads to an exponential increase in running time, and the throughput rapidly decreases when the number of nodes exceeds 100 [49]. Thus, OPNET is not an appropriate simulator tool for dense wireless sensor networks. However, it has a complex architecture and takes time to learn, and it is only commercially available and acquiring a license is expensive [39].
- **QualNet:** QualNet is the commercial version of GloMoSim with upgraded features, such as providing a comprehensive environment for designing protocols, creating and animating experiments and analyzing the results of those experiments [50]. It supports the simulation of wireless sensor network using the ZigBee library. However, the problem of QualNet for researchers is the license cost, which is expensive and only commercially available. The cost of a floating license for this simulation platform is 24000 USD, and the cost of a runtime license is 12000 USD based on comparison between OPNET and QualNet provided by a third party on June 2003.
- **J-Sim:** J-Sim is a discrete-event simulator written using the Java programming language. This simulator supports ad hoc routing protocols, such as DSR and AODV. However, J-Sim is relatively complicated to use [38]. Although J-Sim is not more complex than NS2, due to the popularity of NS2, people may spend more time learning NS2 rather than learning J-Sim. J-Sim has no visualization module, and users must use the NS2 Network Animator (NAM) to visualize the simulation results [48].
- **UWSIM** The UnderWater Simulator (UWSim) is also a simulator that has been used for modeling underwater sensor networks. UWSim is designed and implemented for testing scenarios specific to underwater sensor network environments, such as low frequency, high transmission, low bandwidth, limited memory, and power. The UWSim development follows C# object-oriented programming. Currently, UWSim has support for a limited number of functionalities, it is custom designed for a specific algorithm, and it calls for further extensions to support a wide range of UWSN simulation scenarios [51].
- **USNeT** The Underwater Sensor Network Simulation Tool (USNeT) [52] was designed and implemented in 2013, assuming the conditions that affect underwater communications. USNeT follows the object-oriented design style, and all network entities are implemented as classes in C++ encapsulating thread mechanisms.
- **WOSS** The World Ocean Simulation System (WOSS) [53] was constructed based on NS2 and the NS2-MIRACLE extension.

Table 3 presents a comparative analysis among the simulation tools. This table shows that these tools are either complicated, too general, possess limited visual interface or available as commercial version only. NS2, for instance, is too general and very complicated to learn with the lack of provided documentation. The identified limitations discussed above have motivated the proposed and developed performance analysis tool for data gathering in terrestrial and underwater sensor networks.

IV. PROPOSED PERFORMANCE ANALYSIS TOOL

There are a variety of network simulators found in the area of computer networks, as discussed in the previous section. These simulators are able to mimic the data gathering process in either static or mobile mode in WSNs. In addition, some simulators exist that were previously designed for underwater environments. There are trade-offs, such as complexity, limited visual support, energy model, license cost of the commercial simulators and the ability to simulate highly dense networks. This work proposes a discrete-event simulator model designed specifically for mobile data gathering in WSNs. In addition, the simulator has the ability to gather data using a multi-hop relay based on a tree architecture. Furthermore, this simulator is able to simulate underwater sensor networks. This section will describe the developed model of the simulator. It presents the simulation terminology and assumptions, simulation algorithms, simulation events, simulation scheduler, energy model, acoustic model, and simulation architecture.

A. SIMULATION TERMINOLOGY AND ASSUMPTIONS

The developed simulator has derived multiple parameters. These parameters are categorized into four different groups: the environment, sensor node, data gathering scheme and underwater. The environment parameters include the field size, which determines the deployment area size, and the number of sensor nodes deployed in the field.

In addition, the sensor node parameters include the transmission range, initial energy and the generated packet size. The data gathering parameters include the hop relay bound, which determines the nodes for local data aggregation as the pause location for the mobile element, and hybrid level, which determines the boundary of each data gathering scheme applied (i.e., dividing the deployment area if two data gathering schemes are applied). Furthermore, the underwater environment parameters, such as temperature, data rate,

TABLE 2. Simulation terminology.

Terminology	Description
Number of sensors	Number of deployed sensor nodes in the deployment area
Field Size	height and width of deployment area (meters)
Initial Energy	initial energy for each sensor node (joules)
Transmission Range	The transmission range of each sensor node (meters)
Packet Length	size of each packet transmitted/received from a sensor node (bits)
Hybrid Level	boundary of each data gathering scheme
Relay Hop	maximum hops a data packet traverses to reach the polling points (PPs)
Duty Cycle	number of packets transmitted to the BS
Salinity	salinity of sea water
Acidity	acidity of sea water
Sound speed	speed of sound in sea water (m/s)
Frequency	number of waves that pass by per second in underwater environment (KHz)
Data rate	speed at which data can be transmitted from one device to another (bps)
Depth	depth location of the sensor nodes at sea ground (m)

frequency, acidity, salinity, sound speed, and depth, are also included.

Table 2 presents the simulation terminology used in our proposed simulator tool. This terminology helps the user to easily prepare for creating scenarios. In addition, the user has the ability to change the terminology values based on the needed requirements. The following assumptions have been adopted in the developed simulator:

- A sensor node has the capability to detect its position using built-in GPS.
- All nodes are homogeneous with the same features, such as energy power and memory size.
- The static *BS* is located at the center or at the corner of the deployment area.
- The simulator assumed packet error free.

B. SIMULATION ALGORITHMS

In this section, we attempt to show the simulation process flow of the developed performance analysis tool. These processes are divided into different stages that follow each other chronologically. These stages include initializing variables, selecting the simulation environment, generating and deploying sensor nodes and constructing paths (i.e., routing). These processes start from the initialization of variables to the stage of obtaining the final results. Figure 3 illustrates the simulation process flow.

- **Initialize variables:** The developed simulator provides a user-friendly interface for entering the initial variables. The initialization module assigns the values of the variables based on the required scenario, such as the number of sensor nodes, deployment field size, transmission range and relay hop bound. The initialization step is followed by selecting the simulation environment (i.e., terrestrial or underwater). Figure 4 illustrates the user-

TABLE 3. Comparison between simulation tools.

	NS2	NS3	GloMoSim	OPNET	QualNet	J-Sim
Free / Commercial	Open Source	Open Source	Commercial	Commercial	Commercial	Open Source
complexity	High	Moderate	Moderate	Moderate	Moderate	High
Documentation	Poorly documented source code	Poor	Stopped releasing updates in 2000	a good manual and an introductory tutorial	Good documentation available	poor documentation available
Generic	Yes	Yes	No	Yes	Yes	No
WSN Modules	Energy Model, battery model, Mobility	Not yet	Sensor-network-specific MAC and network protocols, mobility model	mobility of nodes, ad hoc connectivity, node failure models, modeling of power consumption	Limited	Support ad hoc, Energy model
Interface	Limited visual support	Under development	Limited visual support	advanced graphical interface	Good visual support	Good GUI library no tools provided for network visualization
PL	C++ / OTcl	C++ / optional Python scripting	Parsec (C-Based)	C or C++ / Java	C++	Java
Limitations for WSN users	Questionable results when sensor nodes increased to certain level	Still at an early stage	Stopped releasing updates in 2000	It caters to industrial researchers rather than academic researchers	It is a very expensive simulator and acquiring license	Relatively complicated to use and no tools provided for network visualization

friendly interface of the developed simulator. The user interface consists of seven different areas: parameters panel for the simulation parameters, scenario panel to

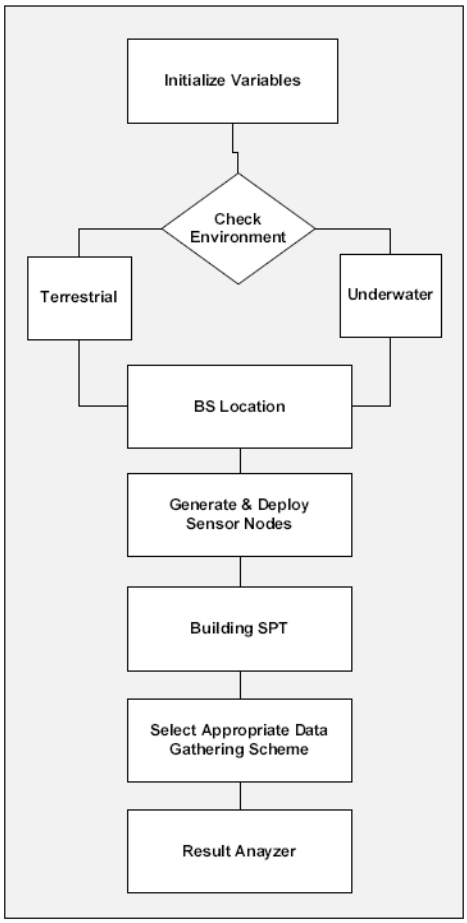


FIGURE 3. Simulation process flow.

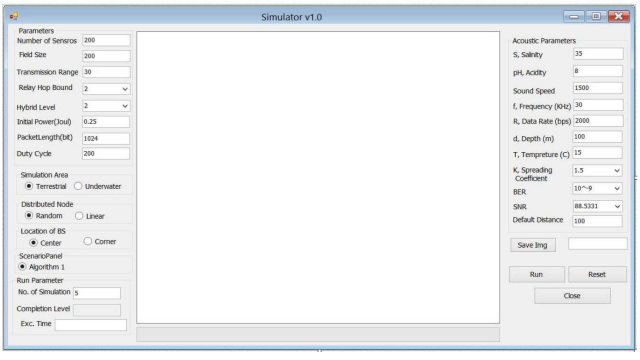


FIGURE 4. Simulation interface.

determine the data gathering scheme applied, run parameter panel that determines the number of runs for the current simulation experiment, and simulation area that mimics and presents the simulation animation. In addition, the location of the BS and node distribution are added to the interface. Furthermore, the acoustic model for the underwater environment is included in the simulation interface.

- **Generate and Deploy Sensor Nodes:** In this section, a number of sensor nodes must deploy over the monitored area. These nodes are deployed depending on the height and width of the deployment area based on a uniform random distribution between 0.0 and 1.0. The generated sensor nodes are represented by X, Y position values, the transmission range value and the initial energy value. Figure 5(a) shows that the sensors deployed randomly over the monitored area along with the BS placed at the center. Algorithm 1 presents the pseudocode for generating and deploying sensors for the required scenario that brings an array of sensors as the output.

Algorithm 1 : Deploy the sensors over the monitored area

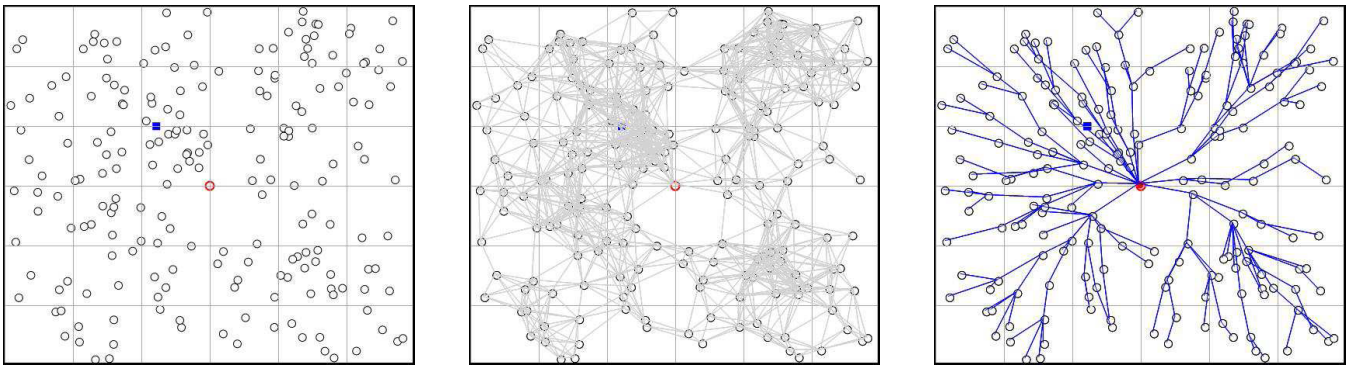
Input: A number of sensors n, square area size s
Output: Array of Sensors *Sensors[n]*
  $n \leftarrow$ number of sensors
  $s \leftarrow$ Topology size
  for $i = 0 \rightarrow n - 1$ **do**
    Set Transmission Range
    Set Initial Energy
    Set Random(X,Y) with respect to s
    sensor[i]= new Sensor(i, Xcor, Ycor, initial Energy, Transmission Range)
  end for
  return Sensors[n]

- **Constructing The Routing Path:** The developed simulator enables two types of paths to be constructed. One of the paths is the shortest path tree, and the other path is a full path tree. Figure 5(b) and Algorithm 2 illustrate the full path scenario, which connects each node with all its neighboring nodes that are within the respective transmission range. The process of a node starts from the *BS*, subsequently moving to the nearest neighboring node, and the process is repeated until all neighbors are reached.

Algorithm 2 Constructing Full-Path Algorithm

Input: A Sensor Network *G(V)*.
Output: A Sensor Network *G(V,E)*.
  $n \leftarrow$ number of sensors
  $s \leftarrow$ Topology Size
  for each $v \in V$ **do**
    Find the adjacency matrix
  end for
  for each $v \in V$ **do**
    for each $u \in$ adjacent[v] **do**
      Draw Edge (u, v)
    end for
  end for

The second path that is constructed provided by the developed simulator is the shortest path tree. This path



(a) (b) (c)

FIGURE 5. Illustrates the deployment and path construction process. (a) Deploy sensor nodes in specified area. (b) Construct the full path. (c) Construct the shortest path.

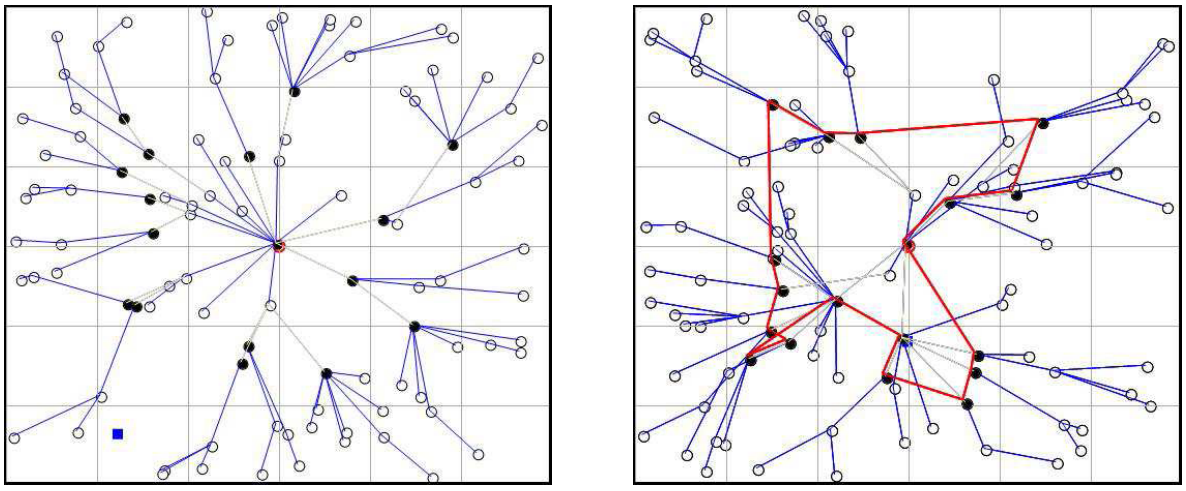
connects each node in the deployed area to the BS via minimum hop. First, the algorithm seeks to find the closest sensor to the root (i.e., the *BS*) within a defined transmission range. Assume that the closest sensor to the root is u. The next step is to seek the neighbors of the sensor u and calculate the distance (i.e., hop count) of each neighbor to reach the *BS* based on the distance of sensor u. Let us assume, for example, that the hop count of sensor u to reach the *BS* is 2; then, we add 1 to the neighbor in which their hop counts are greater than 3. Thus, the sensor u becomes the parent of sensor v (hence, the function named AdjustPath() is used to ensure the minimum hop count for each sensor node to reach the BS). Subsequently, this process is repeated for all the sensors. Finally, the shortest path tree is constructed based on the distance of each node. Note that there is only one path to the neighbors, which would naturally be the shortest path. In addition, the minimum hop relay count will be considered when selecting the position of the next hop sensor node, which leads to the shortest path to the *BS*. We represent the sensor here using an array of sensor nodes, which are used to create objects of sensors with the X, Y Cartesian position values, transmission range, parent Id and energy as the attributes of each sensor. Figure 5(c) illustrates that each node connects to only one node until the root of the tree (i.e., *BS*) is reached. Algorithm 3 illustrates this mechanism of constructing the shortest path tree. The data gathering occurs via a multi-hop manner to reach the *BS* for subsequent processing.

- **Selecting Data Gathering Scheme:** The completion of constructing the shortest path tree in the previous section is followed by selecting the data gathering scheme. In this section, the polling points mechanism is adopted for local data aggregation purposes. In addition, the developed simulator supports the mobile data gathering scheme. In this scheme, certain nodes called polling points will be selected among all sensor nodes, which are responsible for the local data aggregation of other nodes. These nodes, which are selected based on a centralized algorithm adopted from the articles [3], [30], aggregate the data from affiliated sensors while waiting for the *ME* to upload the data. The number of sensor nodes that are affiliated with each polling point varies, and it is determined by the hop bound and the node distribution. An example to elaborate this point is as follows. The number of bound hops is equal to 2. Therefore, each sensor will send its data to the nearest *PP* through a maximum of 2 hops or maybe less based on the location of the sensor node and *PP*. Figure 6(a) shows the number of *PPs* and sensor nodes that belong to each point. These *PPs* are sensor nodes that temporarily act as the BS and are willing to handle the *ME*.

Algorithm 3 Constructing Shortest-Path Algorithm

Input: :A sensor network *G(V,E), and the static data sink* π *as root*.
Output: :A Sensor Network *G(V,E)*.

```
for each u ∈ V do
    Dist[u] ← ∞
    Parent[u] ← NIL
end for
Dist[root] ← 0
while nTree ≤ V do
    u ← Min To Root
    for each v ∈ adjacent[u] do
        if weight(u,v) ≤ Dist[u]
        then
            Parent[v] ← u
            Dist[v] ← weight(u, v)
        end if
    end for
    nTree + + (Number of nodes in Tree)
    AdjustPath()
end while
```



(a) (b)

FIGURE 6. Illustrates (a) finding the polling points. (b) The tour path of mobile data collector.

The centralized algorithm used is adopted from the articles [3], [30], which illustrates the process used for selecting *PPs*. This process depends on the maximum number of hops d for data packets to traverse to reach the nearest *PP*. The user of the developed simulator is provided the provision for tuning this setting. In the developed simulator, this is performed by constructing the shortest path tree T, and then the farthest leaf (node) is selected based on the maximum hop distance from the root on T. Assume that d is equal to 2 and then moves up the tree, T, by two hops and mark this node as a *PP*. A new geometric tree rooted to *PP* is created by the child and sub-child of the current *PP* after removing them from tree T. All of the processes are repeated until all sensor nodes are removed from the tree, **T**, and each sensor reports to one and only one *PP*.

Figure 7 illustrates the process of selecting the appropriate *PPs*, which presents 25 nodes distributed and rooted to the BS. Thus, constructing the tree is as shown in Figure 7(a). The next step is to find the farthest leaf (node), which is sensor node number (24) with 5 hops away from the *BS*. Then, based on the algorithm, it moves up the tree T two hops, which is exactly to sensor node number 13. Now, mark this sensor node as the *PP*, and all sensors affiliated with this node (i.e., 24 and 9) will be removed from the tree T and will report to this *PP*. The same process is extended to other sensors until all of them are removed from the tree T and report to one of the five *PPs* (i.e., 13, 16, 4, 7, and the *BS*), as shown in Figure 7(b).

- **Finding The Shortest Tour Path:** The completion of selecting the appropriate *PPs* is followed by the process of gathering data that are aggregated at each *PP*. The *ME* that is responsible for collecting data from each *PP* should start the tour path from the *BS*.

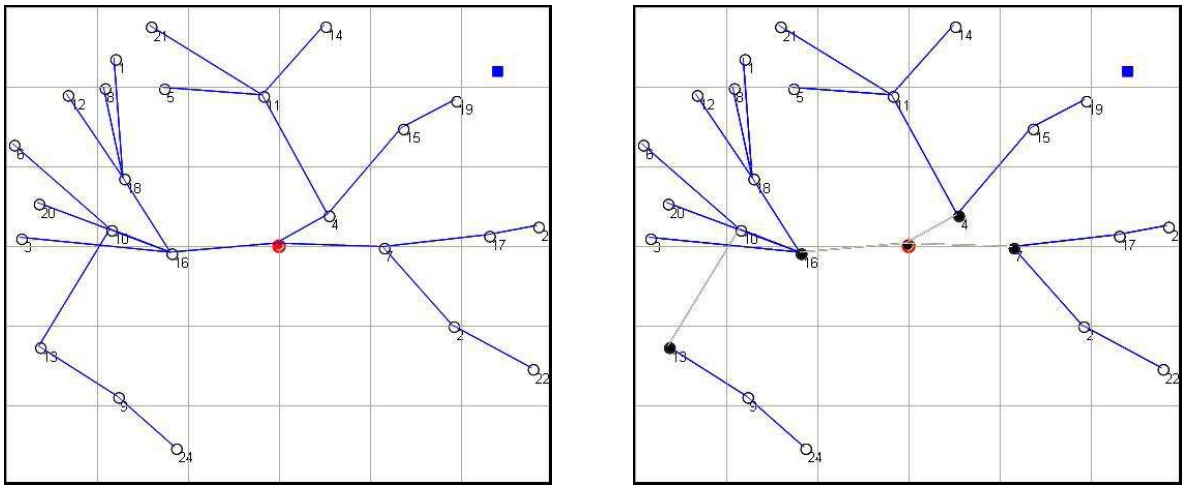
Algorithm 4 : Constructing The Tour Path Algorithm

Input: set of polling points PPs.
Output: NN and Tour Length visiting all PPs.

```
for each i, j ∈ V do
    Compute Dist[i,j] = Euclidean distance between i,j
end for
for i = 1 → V do
d[i] ← ∞
for j = 1 → V do
  if i ≠ j and Dist[i, j] ≤ d[i] then
    d[i] ← Dist[i, j]
    NN[i] ← j
  end if
end for
end for
Return NN, Dist
```

Subsequently, the ME traverses to each *PP* and sequentially returns to the *BS*. Figure 6(b) illustrates the tour path of the *ME* through the appropriate *PP*, which includes the *BS*. The movement of the *ME* is identical to the legacy *TSP*, which is an NP-hard problem. The nearest neighbor (*NN*) [54] algorithm is adopted to solve the *TSP*.

The *NN* algorithm is one of the methods used to solve the *TSP*. The *NN* algorithm works based on the concept of traversing from a city to the nearest city that is located closest to the previous city. Subsequently, the next city is selected until it has finished all the cities. The *NN* visits each city only once before returning to the *BS*. Algorithm 4 and Figure 6(b) illustrate the tour path of the *ME*, including all selected *PPs* and the *BS* based on the *NN* algorithm.



(a) (b)

FIGURE 7. An example to illustrate the find polling point algorithm (N = 25, d=2).

TABLE 4. Derived events.

Event	Function
Detect event	Each node is monitoring and detecting any events occurring in the surrounding environment.
Packet transmits	The normal node generates a packet with the specified length based on the detected event and sends it to its parent; if the node is a polling node, then the packets are aggregated and sent to the ME.
Packet receives	The nodes in the tree received multiple packets from a sub-child (i.e., affiliated sensors) if exists.
Transmit a beacon	The ME sends a continuous beacon during traversing the deployment field looking for the aggregation nodes to upload their data.
Received a beacon	The aggregation nodes that received a beacon from the ME should start uploading the aggregated data based on the time slots given by the ME.

C. EVENTS

There are two methods to derive an event. The first is an occurrence in the monitored environment that changes the statistical composition of the sensor node. The second occurs during the data gathering period when the ME reaches the polling nodes. The derivation of events is based on the change observed at each node. The mapping of the observations to the required simulator components has led to five main events in this work. These events are detect event, packet transmit, packet receive, transmit a beacon, and receive a beacon. The functions of the aforementioned events are summarized in Table 4.

D. SCHEDULER

The scheduler is the controller for the queue of execution events with respect to their timestamps. In the developed simulator, the time division multiple access (TDMA) is adopted for scheduling purposes. In TDMA, the bandwidth is allowed for the user for a short period of time only. It is divided into multiple channels based on users, and it dedicates a time slot for each one. In other words, it is a technology that allows users to use the same radio frequency (RF) without interference by allocating a unique time slot for each one. In addition, the nodes need to wait for the data from their children (if they exist) before sending their packets up the tree. The proposed scheduling segments the nodes into levels according to their distance in number of hops from the *BS*, and the schedule is constructed in reverse order of hop distance, as illustrated in Figure 8. The nodes within level three should send their data packets first, and then the nodes within level two aggregate the received packets with their own packets. To understand TDMA scheduling, we assumed the following:

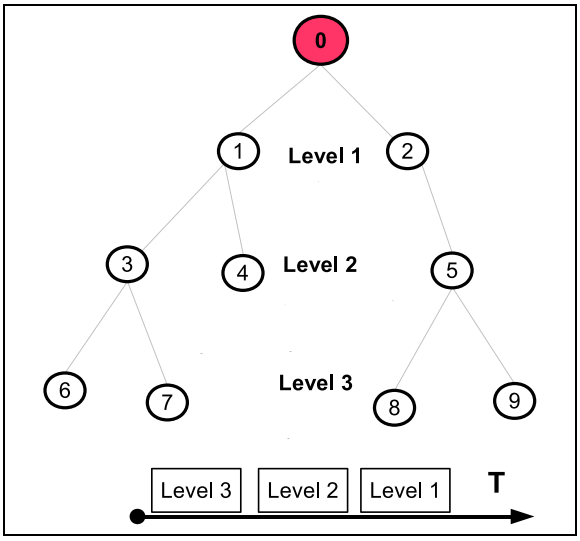


FIGURE 8. TDMA scheduling.

- Each node generates a single packet at the beginning of each duty cycle.
- Each node has the ability to aggregate all packets received from its children with its packet and produce one packet only for the upper level. The aggregation

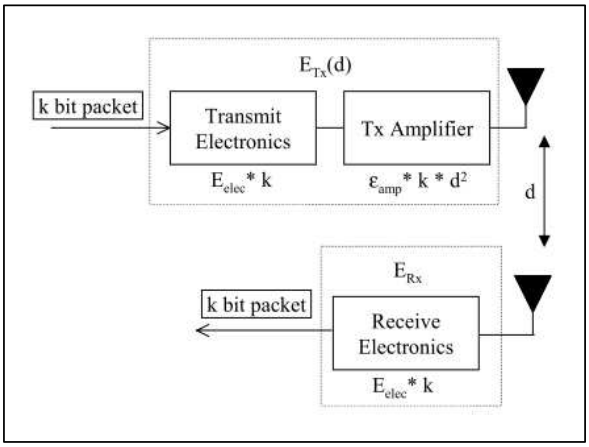


FIGURE 9. First-order radio model [55].

functions could be MIN, MAX, MEDIAN, COUNT and AVERAGE.

- Each node ideally should wait to receive all packets from its children prior to transmitting its own data.

E. ENERGY MODEL

To simulate and evaluate the dissipated energy, we have adopted a first-order radio model [55]. In this model, a simple model is assumed in which the radio dissipates $E_{elec} = 50nJ/bit$ to run the transmitter or receiver circuitry and $\varepsilon_{amp} = 100pJ/bit/m^2$ for the transmit amplifier. Thus, transmitting or receiving a K-bit message a distance (d) expends energy as follows:

$$E_{Tx}(K,d) = E_{elec} * K + \varepsilon_{amp} * K * d^2 \quad (1)$$

$$E_{Rx}(K) = E_{elec} * K \quad (2)$$

The energy required to transmit a K-bit message is greater than the energy used to receive a K-bit message. In addition, transmitting a K-bit message from node A to node B is equal to transmitting a K-bit message from node B to node A. Figure 9 illustrates the radio model adopted and its parameters.

F. ACOUSTIC MODEL

The communication via sound waves is called the acoustic model. In the underwater environment, the communication medium can be either radio, optical or sound (acoustic) waves. However, electromagnetic waves, such as optical and radio waves, suffer from high propagation losses and scattering problems. In addition, electromagnetic waves do not travel long distances in the underwater environment. Furthermore, radio waves require high transmission power and long antennas to communicate, while optical waves suffer from high signal attenuation and can only travel a short distance. Thus, sound waves are the best communication medium for the underwater environment. However, acoustic waves also have some limitations, such as long propagation delay and low available bandwidth. The acoustic model adopted in this simulator is based on the following equations:

- One form of the equation representing passive sonar is obtained from [56], as presented below:

$$SL(d,f) = A(d,f) + N(f) + SNR - DI \quad (3)$$

Where, DI is the directivity index which is equal to zero in this model (Omni-directional directivity). Signal-to-Noise Ratio (SNR) is measured at the receiver side while the $N(f)$ is the power spectral density of the ambient noise at frequency f. $A(d,f)$ is the path loss in acoustic channels.

- The ambient noise in the ocean can be modeled using four components [57], namely, water turbulence, surface ship, thermal noise and breaking waves.
  - Water turbulance (Nt), 10 log(Nt(f)) = 17 - 30 log(f).
  - Surface-ship (Ns), 10 log(Ns(f)) = 40 + 20(s - 0.5) + 26 log(f) - 60 log(f + 0.03).
  - Thermal noise (Nth), 10 log(Nth(f)) = −15 + 20 log(f).
  - Breaking waves (Nw), 10 log(Nw(f)) = 50 + $7.5w^{\frac{1}{2}}$ + 20 log(f) - 40 log(f + 0.4)

  where s is the shipping activity factor, 0 ≤ s ≤ 1, and w is the wind speed in m/s. The overall power spectral density in dB re μ per Hz of the ambient noise is given as the following :

$$N(f) = Nt(f) + Ns(f) + Nth(f) + Nw(f) \quad (4)$$

- The attenuation or path loss, A(d, f), of an acoustic signal with transmission range d in meters and frequency f in KHz is given in dB by [58]:

$$A(d,f) = k \times log(d) + \alpha(f)d \times 10^{-3} \quad (5)$$

- The absorption coefficient of sea water is expressed by Ainslie and McColm [59], [60] as follows:

$$\alpha(f) = v_1 \frac{f_1 f^2}{f_1 + f^2} + v_2 \frac{f_2 f^2}{f_2 + f^2} + v_3 f^2 \quad (6)$$

where,

  - $f_1 = 0.78(S/35)^{\frac{1}{2}} e^{\frac{T}{26}}$
  - $f_2 = 42e^{\frac{T}{17}}$
  - $v_1 = 0.106e^{\frac{PH-8}{0.56}}$
  - $v_2 = 0.52(1 + \frac{T}{43})(\frac{S}{35})e^{\frac{-h}{6}}$
  - $v_3 = 0.00049e^{-(\frac{T}{27} + \frac{h}{17})}$

T is the temperature in °C, h is the depth in meters, pH is water acidity, and S is the water salinity. The default values of pH and S are 8 and 35, respectively.

- The velocity of acoustic waves in water is considerably slower than in electromagnetic propagation (i.e., five time slower). The propagation velocity in meter/seconds is given as follows [61]:

$$c = 1.402385 \times 10^3 + 5.038813T - 5.799136 \times 10^{-2}T^2 + 3.287156 \times 10^{-4}T^3 - 1.398845 \times 10^{-6}T^4 + 2.787860 \times 10^{-9}T^5.$$

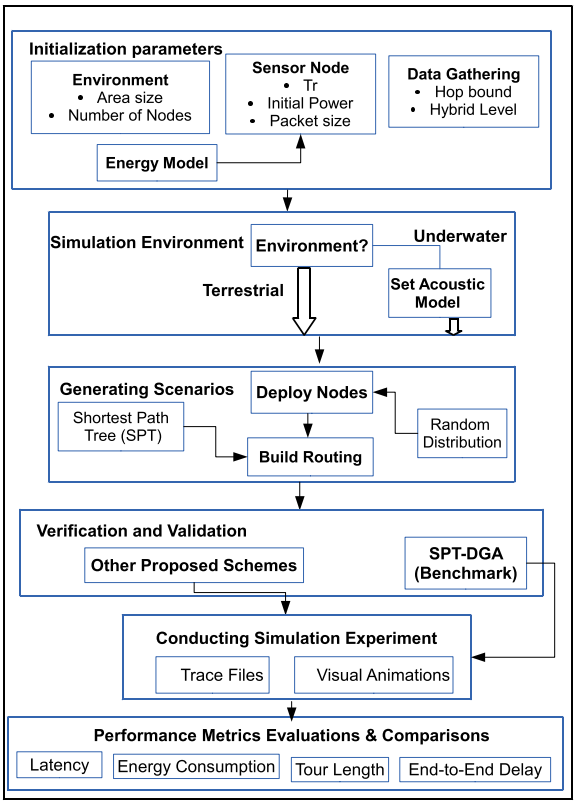


FIGURE 10. Simulation architecture.

where T is the water temperature in °C. This equation is valid in the range $0 < T < 95$°C.

G. PROPOSED SIMULATION ARCHITECTURE

The previous section has extensively explained the developed simulation based on the respective individual components. In this section, the consolidated and unified architecture that is developed in this work is elaborated. Figure 10 illustrates the components and the related correlations. The details are elaborated as follows:

- **Sensor Node**: To realize the sensor node architecture, we have constructed the sensor node class that provides the Id, position, initial energy, parent Id and the nearest neighbor Id. These nodes are configured to generate periodical stimuli signals and passes them to the nearest node, which are directed to the *BS* based on the shortest path tree or passes them to the targeted *PP* with a bounded hop. In addition, the energy model is constructed based on the first-order radio model [55].
- **Initializing Variables and Generating Scenarios**: Initializing variables and generating scenarios involves setting of the parameters, which is controlled by the users, that are determined by the present scenario for testing purposes. This includes the topology size, number of nodes, transmission range, initial energy, packet size and hop bound. In addition, it involves the deployment stage and the type of distribution nodes selected. Furthermore, the simulation environment should be selected prior to deploying the sensor nodes (i.e., terrestrial or underwater).
- **Routing**: The realization of the routing architecture has been achieved by constructing two types of routing algorithms. The first is responsible for constructing a shortest path tree to the *BS* based on graph theory. This stage involves the data traversing from all sensors to the *BS* via a multi-hop manner without any *ME*. The second algorithm is responsible for determining the shortest tour path that visits all *PPs* and the *BS* based on the *NN* algorithm. The data are gathered to a specific *PP* through a multi-hop manner and then wait for a *ME* for the uploading task. The ME visits each *PP* and the *BS*.
- **Data Gathering**: The realization of the data gathering architecture is achieved by constructing two components for collecting gathered data from the sensing field. The first component is the *BS*, which remains stationary, and the second component is the *ME*, which is responsible for collecting the data from all *PPs* across the deployed area.
- **Output**: The developed simulator has produced two types of output. The first is a visualization of various results in text boxes, such as the total tour length of the *ME*. The second is the trace file generated for further processing, which includes the specified performance metrics.

This proposed simulator has many features, such as ease of use, attractive graphical user interface, simplicity, low memory usage and ability to simulate a highly dense network. This simulator also has the following features:

1) **Extensibility**: User can define the sensor nodes with different attributes based on a required scenario. In addition, the user is capable of defining a distinct topology. Moreover, any algorithm in data gathering can be added using very direct and low complexity efforts.
2) **Flexibility**: The user can determine whether to deploy a single or many nodes based on the uniform random distribution.
3) **Practicability**: A user is given the ability to record the results of the simulation in external files, which can be opened by any generic text editor. In addition, the user is able to obtain the result directly through the provided graphical user interface when the simulation terminates in completion.

V. VERIFICATION AND VALIDATION

The essence of any performance analysis tool developed is the validity or correctness in depicting the actual system. There are many ways to verify and validate models [62]. Animation is one type of verification process, which monitors the dynamic display (i.e., moving pictures) of the simulated system by the user and the user detects the errors in a visual manner. In addition, comparing the simulated data and the real data through a simple test such as the graphical interface

is another type of validation model. The effectiveness of the simulator in this work was validated by comparing the results with those of [3], which serve as the benchmark.

To validate the proposed simulation tool, the results from this tool should ideally be compared with real data from the cited paper as the original creator of the algorithm. This research has adopted this practice; thus, [3] has been utilized for this purpose. The results are acquired from [3] via the extraction using DigitizeIt [63]. It is a tool used to extract numerical data from scanned graphs or charts. We imported the results graph from the above benchmark into the DigitizeIt tool.

To further validate the developed simulator, a comparison between our results and the benchmark was performed. The concept of error derivations was applied to validate the developed simulator. The error percentage is computed using the following equation, where x represents the results produced using the simulation tool and x' represents the results produced using the benchmark. However, by analyzing and comparing the acquired results of the developed simulator with the benchmark results, the validation is indeed realized. The evaluation procedure is designed to prove that the developed simulator indeed serves as a worthy performance analysis tool. Furthermore, this simulation tool has successfully contributed to the repository of tools for WSN analysis.

$$ErrorPercentage = \frac{\sum(x - x')}{\sum(x')} * 100 \quad (7)$$

In addition to the previous techniques to validate and verify the proposed simulation tool, many other techniques are presented to ensure constructing the right model, constructing the model correctly and revealing the existence of errors in the model (i.e., validation, verification and testing (VV&T)) [64]–[66]. These techniques are adopted and considered during the construction of our simulation model to ensure a standard verification, validation and testing.

1) Prior to VV&T, an investigation with the existing solution is considered by studying the alternative techniques (i.e., other simulation tools) that can be used to solve the existing problem. Thus, these tools are either costly (i.e., require a license) or too general and not specific to our problem.
2) In the developed simulator, each module has been extensively audited and checked for correctness, completeness and consistency to ensure that the simulation is conducted with respect to standards and guidelines.
3) To increase our confidence, a block-box testing (i.e., functional testing) is conducted using input data and generated output data. In addition, the developed simulation tool is designed and evaluated based on the objectives of this research. Thus, the credibility of the developed tool has been demonstrated by performing tests with the benchmark using the same conditions and environment as stated in the next section.
4) Debugging and tracing techniques are adopted in our simulation tool, displaying an instant value using text boxes or labels during run time. In addition, text files that trace the entire simulation environment (i.e., energy level, tour path, latency, and node distribution) are considered. Furthermore, the dot net framework environment (i.e., C#.net) helps to permit extending the process of validity.
5) Visualization (i.e., animation) is a key feature that assists in understanding the developed tool. Thus, a friendly graphical user interface (GUI) is developed with the simulation tool that displays images of the deployed area, node's position, *BS* position, the tree structure constructed based on multi-hop routing, PPs positions, and mimic the movement of the mobile element through a predetermined path during the execution, which enables us to visually discover errors.
6) The developed simulation is documented and planned as described in detail in the previous section in this paper.

TABLE 5. Simulators' main characteristics comparison.

Simulator Tools	Number of nodes	Execution Time
The Developed Simulator	Up to 5000	Moderate which based on number of nodes
NS2	Up to 100	Increasing dramatically when simulation exceeds 100 nodes
OMNET++	Up to 501	Increasing dramatically when simulation exceeds 501 nodes
GLOMOSIM	Up to 10,000	Depends on number of parallel processors
OPNET	Up to 100	Not appropriate for dense networks
J-SIM	Up to 500	Execution time is much larger than NS2

VI. RESULTS AND DISCUSSION

The developed simulation is run to ensure a similar environment with the respective benchmark. The main characteristics of the proposed simulator are compared with those of the existing simulators for WSNs, for instance, the execution time with the increasing number of nodes in comparison with other simulators. The main characteristics are summarized in Table 5.

The following subsections describe the results obtained via both terrestrial and underwater sensor networks.

A. TERRESTRIAL SENSOR NETWORKS

In this section, the simulator is tested and evaluated in terms of the tour length of the mobile data collector and the average relay hop count. Due to the randomness of deployment, the simulation for each performance point is the average of 500 times. The author of the benchmark neither mentioned the type of random distribution nor the multi-hop routing used to construct the shortest path in the article. In addition, the programming language/simulation tool was not mentioned. Thus, little variance could possibly appear in the below results, which were plotted using the gnuplot graphics utility version 4.4.

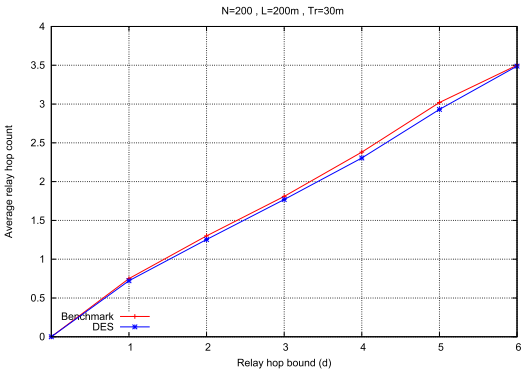


FIGURE 11. Average relay hop count as a function of hop bound (d).

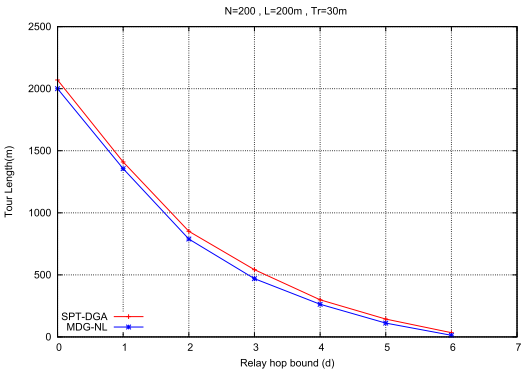


FIGURE 12. Tour length as a function of hop bound (d).

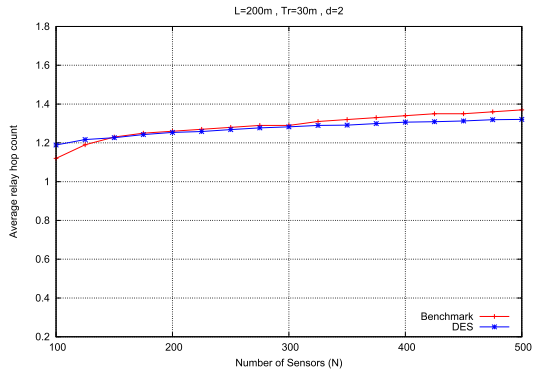


FIGURE 13. Average relay hop count as a function of the number of sensors (N).

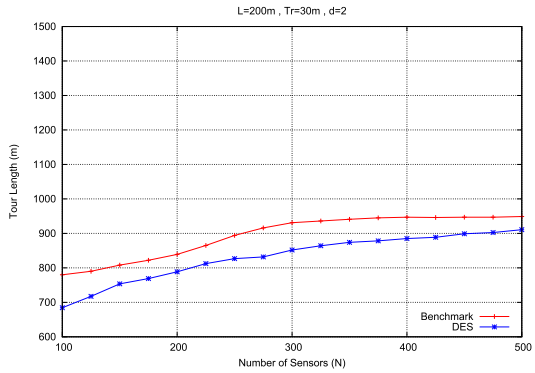


FIGURE 14. Tour length as a function of the number of sensors (N).

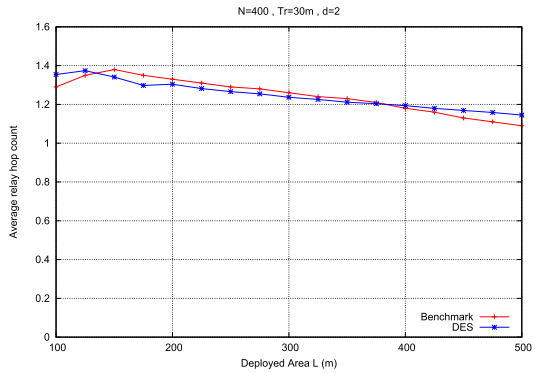


FIGURE 15. Average relay hop count as a function of deployment area (L).

Figure 11 demonstrates that the developed simulator has significantly represented the metrics of the benchmark correctly. It shows that the average relay hop count increases when the hop bound increases. This result is due to increasing the hop limit for each node to reach the nearest PP. It shows the developed DES acquired an average relay hop count compared to the benchmark. Figure 12 shows the effect of increasing hop relay bound on the tour length of the mobile data collector. It is clear that increasing the hop bound leads to shortening of the tour path due to minimizing the nodes selected as PPs that are visited by the mobile collector. The figure shows that the developed DES acquired a tour length comparable to the benchmark with an average deviation ratio equal to 6.51%.

Figure 13 presents the effects of increasing the relay hop bound on the average relay hop count as a function of the number of sensor nodes N. The average relay hop count slightly increases as the hop bound increases due to the deployment area size remaining unchanged. This figure shows that the developed DES acquired an average relay hop count comparable to the benchmark with an average deviation ratio equal to 1.12%. Figure 14 shows the variance that occurred on the tour length of the mobile collector between the DES and the benchmark in terms of the number of nodes N. It is clear that increasing the number of nodes leads to an increase in the tour length to a certain point. The developed DES has achieved a pattern resembling that of benchmark with an average deviation ratio equal to 7%.

Figure 15 presents the results pattern of the average relay hop count for the developed DES and the benchmark in terms of the deployment area size L. It shows that similarity is achieved with an average deviation ratio equal to 0.02%. Figure 16 presents the tour length variance of the developed DES and the benchmark in terms of the deployed area size L. A similar pattern is achieved with an average deviation ratio equal to 18%.

Figure 17 presents the pattern results of the average relay hop count of both DES and the benchmark in terms of the transmission range Tr. It shows an almost identical pattern

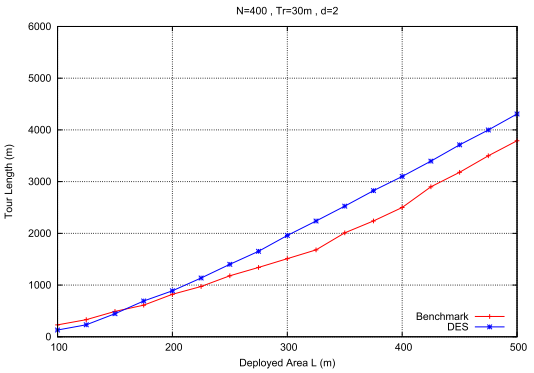


FIGURE 16. Tour length as a function of deployment area (L).

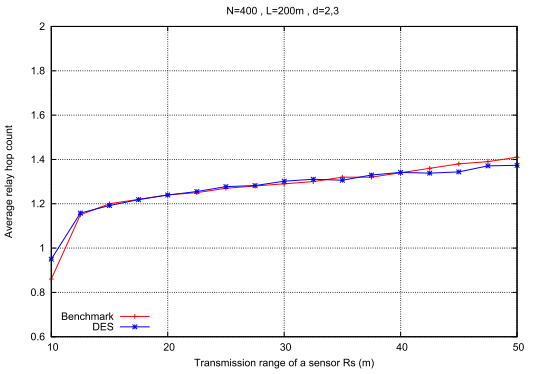


FIGURE 17. Average relay hop count as a function of the transmission range (Tr).

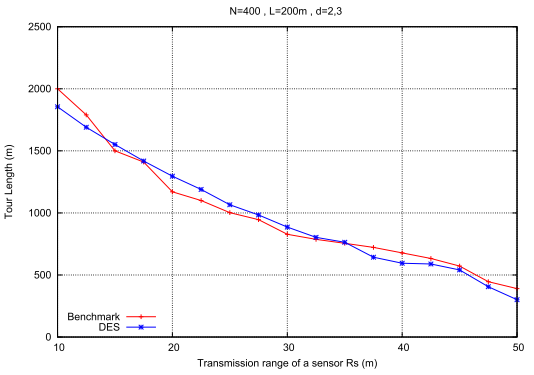


FIGURE 18. Tour length as a function of the transmission range (Tr).

with an average deviation ratio equal to 0.04%. Figure 18 presents the pattern results of the tour length in terms of the transmission range in both DES and the benchmark. It is clear that the pattern of DES is almost identical to the patterns of the benchmark with an average deviation ratio equal to 0.9%.

B. UNDERWATER SENSOR NETWORKS

In this section, the simulator is tested and evaluated in terms of the end-to-end delay and transmission power tour in underwater acoustic sensor networks.

Figure 19 illustrates the end-to-end delay in terms of the sensor nodes of a two-dimensional network. It is clear that

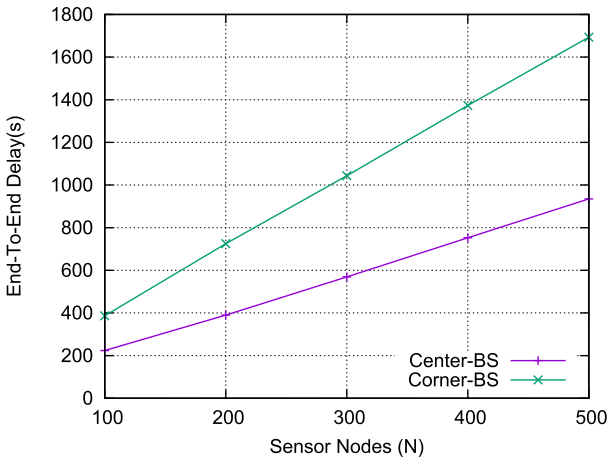


FIGURE 19. End delay vs. sensor nodes.

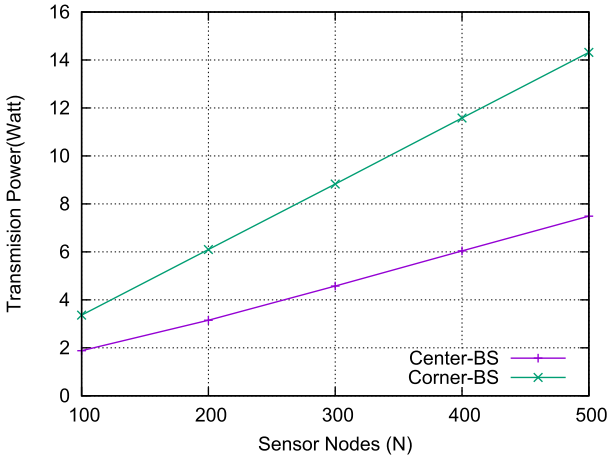


FIGURE 20. TxPower vs. sensor nodes.

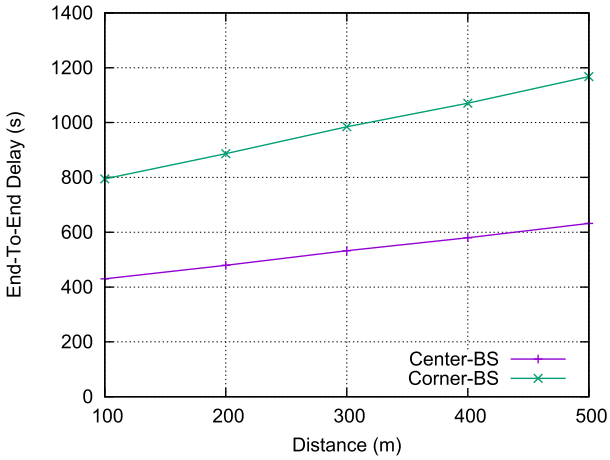


FIGURE 21. End delay vs. distance.

when the number of nodes (N) is smaller, the end-to-end delay in both networks is at the minimum level. The impact of increasing (N) on the delay is obvious, particularly when the number of nodes reaches 500. The location of the base station is affected by the total delay as shown in Figure 19 due to the

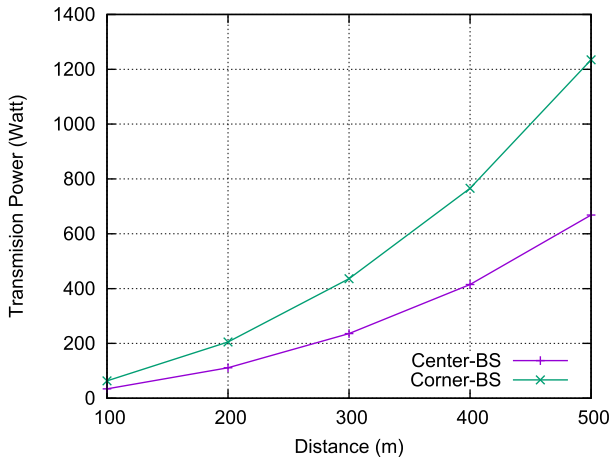


FIGURE 22. TxPower vs. distance.

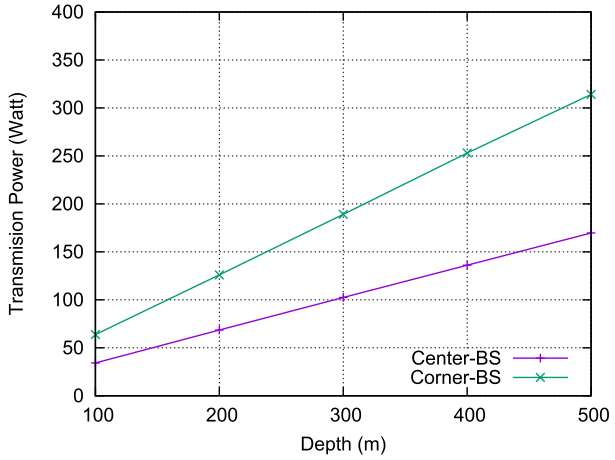


FIGURE 23. TxPower vs. depth.

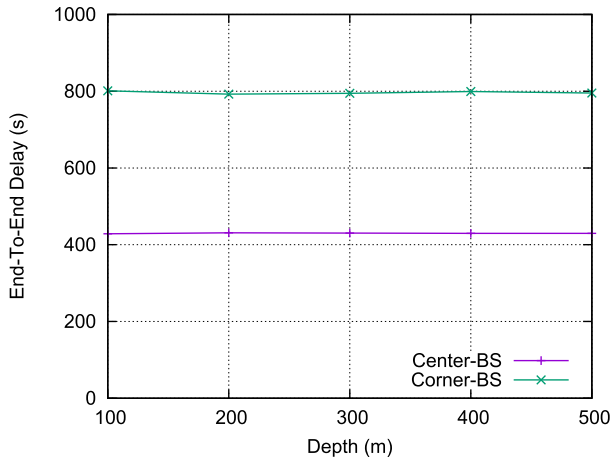


FIGURE 24. End delay vs. depth.

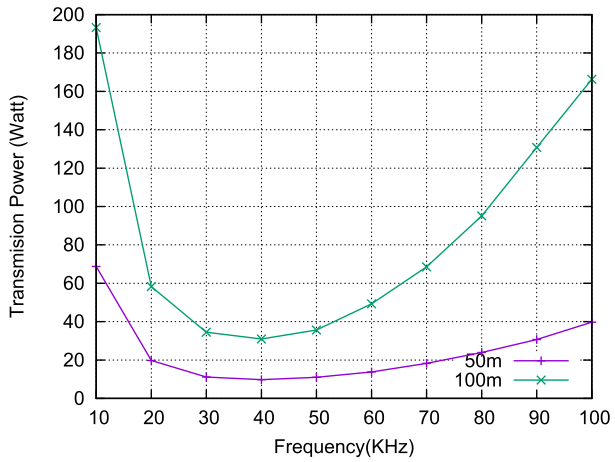


FIGURE 25. TxPower vs. frequency.

increase in the multi-hop level to reach the base station when it is located at the corner of the field.

Figure 20 illustrates the transmission power in terms of the sensor nodes of a two-dimensional network. As shown in this figure, when the sensor node (N) increases, the transmission power in both networks also increases. Each leaf node first sends a packet to its parent node. Then, the parent node generates its packet and sends both packets to a higher level of the tree until the base station is reached. Thus, increasing the number of nodes will increase the transmission power, as depicted in the figure.

Figure 21 shows the end-to-end delay in terms of distance among the sensor nodes of a two-dimensional network. As shown in this figure, when the distance among nodes increases, the end-to-end delay also increases in all scenarios. This result is due to increasing the propagation delay during the forwarding of packets. However, when the BS is located at the center, the propagation delay is at the lowest stage.

Figure 22 shows the transmission power in terms of distance among the sensor nodes of a two-dimensional network. Clearly, the power consumption is proportional to the distance, which means that a slight increase in distance leads to an increase in the transmission power in all cases. However, the lowest is when the BS is located at the middle of the field.

Figure 23 presents the transmission power in terms of the depth of a two-dimensional network. Clearly, increasing the depth requires an increase in the transmission power required to transfer one packet from one sensor to another. In contrast, the end-to-end delay is not affected at all by increasing the depth, as illustrated in Figure 24.

Figure 25 shows the transmission power in terms of frequency. As shown in this figure, the lowest transmission power falls in range from 20 to 50 KHz. Increasing the frequency beyond this range leads to a dramatic increase in the transmission power.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a unique and customized performance analysis tool for analyzing mobile data gathering strategies in WSNs. The focus has been on achieving the design and usability of a tool that is versatile and comprehensive. Unlike many existing simulators, this performance

analysis tool has been designed specifically for data gathering purposes and for cases with and without *MEs*. The routing protocol used in this analysis tool has been constructed based on graph theory. In addition, this simulator provides for both static data gathering and mobile data gathering. The use of the C# programming language enables others to adopt the principle of development and extend it to other domains. The OOP paradigms are used to ensure easy extension and modification with high scalability. This work has successfully contributed to the repository of tools for WSN analysis. The tool is to be further extended by adopting other routing protocols and by implement the MAC layer.

ACKNOWLEDGMENT

The authors thank the Science and Technology Unit for their logistic support.

REFERENCES

[1] Y. Singh, D. D. Barak, V. Siwach, and P. Rani, "Attacks on wireless sensor network: A survey," *Int. J. Comput. Sci. Manage. Stud.*, vol. 12, no. 3, pp. 2231–5268, 2012.

[2] Z. Rezaei and S. Mobininejad, "Energy saving in wireless sensor networks," *Int. J. Comput. Sci. Eng.*, vol. 3, no. 1, p. 23, 2012.

[3] M. Zhao and Y. Yang, "Bounded relay hop mobile data gathering in wireless sensor networks," *IEEE Trans. Comput.*, vol. 61, no. 2, pp. 265–277, Feb. 2012.

[4] S. K. Dhurandher, M. S. Obaidat, and M. Gupta, "Providing reliable and link stability-based geocasting model in underwater environment," *Int. J. Commun. Syst.*, vol. 25, no. 3, 356–375, 2012, doi: 10.1002/dac.1245. [Online]. Available: http://dx.doi.org/10.1002/dac.1245

[5] M. Imran, A. M. Said, and H. Hasbullah, "A survey of simulators, emulators and testbeds for wireless sensor networks," in *Proc. Int. Symp. Inf. Technol. (ITSim)*, vol. 2. Jun. 2010, pp. 897–902, doi: 10.1109/ITSIM.2010.5561571.

[6] H. Sundani, H. Li, V. Devabhaktuni, M. Alam, and P. Bhattacharya, "Wireless sensor network simulators a survey and comparisons," *Int. J. Comput. Netw.*, vol. 2, pp. 249–265, Feb. 2010.

[7] A. Abuarqoub, F. Alfayez, M. Hammoudeh, T. Alsboui, and A. Nisbet, "Simulation issues in wireless sensor networks: A survey," in *Proc. 6th Int. Conf. Sensor Technol. Appl. (SENSORCOMM)*, 2012, pp. 222–228.

[8] J.-H. Zhang, H. Peng, and T.-T. Yin, "Tree-adapting: An adaptive data aggregation method for wireless sensor networks," in *Proc. 6th Int. Conf. Wireless Commun. Netw. Mobile Comput. (WiCOM)*, Sep. 2010, pp. 1–5, doi: 10.1109/WICOM.2010.5601092.

[9] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," in *Proc. 2nd Int. Conf. Embedded Netw. Sensor Syst.*, 2004, pp. 250–262.

[10] R. Zhang, L. Wang, S. Geng, and Z. Jia, "A balanced cluster routing protocol of wireless sensor network," in *Proc. Int. Conf. Embedded Softw. Syst. Symp. (ICESS Symposia)*, 2008, pp. 221–225, doi: 10.1109/ICESS.Symposia.2008.40.

[11] R. Chang and C.-J. Kuo, "An energy efficient routing mechanism for wireless sensor networks," in *Proc. 20th Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, vol. 2. Apr. 2006, p. 5, doi: 10.1109/AINA.2006.86.

[12] A. Khan, A. B. Abdullah, and A. U. Rahman, "Data delivery optimization by efficient cluster head selection of wireless sensor network," in *Proc. 5th Int. Colloq. Signal Process. Appl. (CSPA)*, 2009, pp. 186–190, doi: 10.1109/CSPA.2009.5069213.

[13] A. Ray and D. De, "Energy efficient cluster head selection in wireless sensor network," in *Proc. 1st Int. Conf. Recent Adv. Inf. Technol. (RAIT)*, Mar. 2012, pp. 306–311, doi: 10.1109/RAIT.2012.6194436.

[14] P. Hao, W. Qiu, and R. Evans, "An improved cluster-head selection approach in wireless sensor networks," in *Proc. 5th Int. Conf. Intell. Sens., Sensor Netw. Inf. Process. (ISSNIP)*, Dec. 2009, pp. 79–84, doi: 10.1109/ISSNIP.2009.5416783.

[15] D. Mandala, X. Du, F. Dai, and C. You, "Load balance and energy efficient data gathering in wireless sensor networks," *Wireless Commun. Mobile Comput.*, vol. 8, no. 5, pp. 645–659, 2008.

[16] G. Xin and L. Guan, "Data gathering algorithm based load balance for wireless sensor networks," in *Proc. 17th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Aug. 2008, pp. 1–5, doi: 10.1109/ICCCN.2008.ECP.143.

[17] Y. Mao, Z. Liu, L. Zhang, and X. Li, "An effective data gathering scheme in heterogeneous energy wireless sensor networks," in *Proc. Int. Conf. Comput. Sci. Eng. (CSE)*, vol. 1. 2009, pp. 338–343, doi: 10.1109/CSE.2009.16.

[18] Y. Wu, X.-Y. Li, Y. Liu, and W. Lou, "Energy-efficient wake-up scheduling for data collection and aggregation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 2, pp. 275–287, Feb. 2010.

[19] M. Zhao, M. Ma, and Y. Yang, "Efficient data gathering with mobile collectors and space-division multiple access technique in wireless sensor networks," *IEEE Trans. Comput.*, vol. 60, no. 3, pp. 400–417, Mar. 2011.

[20] M. Ma and Y. Yang, "Data gathering in wireless sensor networks with mobile collectors," in *Proc. IEEE Int. Symp. Parallel Distrib. Process. (IPDPS)*, Apr. 2008, pp. 1–9, doi: 10.1109/IPDPS.2008.4536269.

[21] C. Wang and P. Ramnanathan, "Energy efficient transmission scheme for data-gathering in mobile sensor networks," in *Proc. 3rd Annu. IEEE Commun. Soc. Sensor Ad Hoc Commun. Netw. (SECON)*, vol. 2. Sep. 2006, pp. 498–507, doi: 10.1109/SAHCN.2006.288506.

[22] S. Basagni, A. Carosi, and C. Petrioli, "Controlled vs. uncontrolled mobility in wireless sensor networks: Some performance insights," in *Proc. IEEE 66th Veh. Technol. Conf. (VTC Fall)*, Sep. 2007, pp. 269–273, doi: 10.1109/VETECF.2007.70.

[23] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme, "Robomote: Enabling mobility in sensor network," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw. (IPSN)*, Apr. 2005, pp. 404–409, doi: 10.1109/IPSN.2005.1440957.

[24] G. Xing, T. Wang, W. Jia, and M. Li, "Rendezvous design algorithms for wireless sensor networks with a mobile base station," in *Proc. 9th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2008, pp. 231–240.

[25] O. Chipara *et al.*, "Real-time power-aware routing in sensor networks," in *Proc. 14th IEEE Int. Workshop Quality Ser. (IWQoS)*, Jun. 2006, pp. 83–92, doi: 10.1109/IWQOS.2006.250454.

[26] S. Vupputuri, K. Rachuri, and C. S. R. Murthy, "Using mobile data collectors to improve network lifetime of wireless sensor networks with reliability constraints," *J. Parallel Distrib. Comput.*, vol. 70, no. 7, pp. 767–778, 2010.

[27] K. Lin, M. Chen, S. Zeadally, and J. J. P. C. Rodrigues, "Balancing energy consumption with mobile agents in wireless sensor networks," *Future Generat. Comput. Syst.*, vol. 28, no. 2, pp. 446–456, 2012.

[28] G. Xing, M. Li, T. Wang, W. Jia, and J. Huang, "Efficient rendezvous algorithms for mobility-enabled wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 11, no. 1, pp. 47–60, Jan. 2012.

[29] M. Ma and Y. Yang, "SenCar: An energy-efficient data gathering mechanism for large-scale multihop sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 10, pp. 1476–1488, Oct. 2007.

[30] M. Ghaleb, S. Subramaniam, M. Othman, and Z. Zukarnain, "Predetermined path of mobile data gathering in wireless sensor networks based on network layout," *EURASIP J. Wireless Commun. Netw.*, vol. 1, p. 51, Dec. 2014, doi: 10.1186/1687-1499-2014-51.

[31] K. Almi'ani, A. Viglas, and L. Libman, "Energy-efficient data gathering with tour length-constrained mobile elements in wireless sensor networks," in *Proc. IEEE 35th Conf. Local Comput. Netw. (LCN)*, Oct. 2010, pp. 582–589, doi: 10.1109/LCN.2010.5735777.

[32] E. Weingartner, L. H. vom, and K. Wehrle, "A performance comparison of recent network simulators," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2009, pp. 1–5, doi: 10.1109/ICC.2009.5198657.

[33] A. Rastegarnia and V. Solouk, "Performance evaluation of castalia wireless sensor network simulator," in *Proc. 34th Int. Conf. Telecommun. Signal Process. (TSP)*, Aug. 2011, pp. 111–115, doi: 10.1109/TSP.2011.6043761.

[34] A. Barberis, L. Barboni, and M. Valle, "Evaluating energy consumption in wireless sensor networks applications," in *Proc. 10th Euromicro Conf. Digit. Syst. Design Archit., Methods Tools (DSD)*, Aug. 2007, pp. 455–462, doi: 10.1109/DSD.2007.4341509.

[35] Y. Xue, H. S. Lee, M. Yang, P. Kumarawadu, H. Ghenniwa, and W. Shen, "Performance evaluation of NS-2 simulator for wireless sensor networks," in *Proc. Can. Conf. Elect. Comput. Eng. (CCECE)*, Apr. 2007, pp. 1372–1375, doi: 10.1109/CCECE.2007.345.

[36] S. Sinha, Z. Chaczko, and R. Klempous, "SNIPER: A wireless sensor network simulator," in *Proc. Comput. Aided Syst. Theory (EUROCAST)*, 2009, pp. 913–920.

[37] S. V. Mallapur and S. R. Patil, "Survey on simulation tools for mobile ad-hoc networks," *Int. J. Comput. Netw. Wireless Commun.*, vol. 2, no. 2, pp. 241–248, 2012.
[38] M. Stehlik, "Comparison of simulators for wireless sensor networks," M.S. thesis, Faculty Informatics, Masaryk Univ, Brno, Czech Republic, 2011.
[39] M. Z. Khan, B. Askwith, F. Bouhafs, and M. Asim, "Limitations of simulation tools for large-scale wireless sensor networks," in *Proc. IEEE Workshops Int. Conf. Adv. Inf. Netw. Appl. (WAINA)*, Mar. 2011, pp. 820–825, doi: 10.1109/WAINA.2011.59.
[40] I. Jawhar, "A flexible object-oriented design of an event-driven wireless network simulator," in *Proc. Wireless Telecommun. Symp. (WTS)*, Apr. 2009, pp. 1–7, doi: 10.1109/WTS.2009.5068942.
[41] D. Curren, "A survey of simulation in sensor network," Univ. Binghamton, Binghamton, NY, USA, Tech. Rep. CS580, 2005.
[42] T. Henderson, S. Roy, S. Floyd, and G. F. Riley, "NS-3 project goals," in *Proc. IP Netw. Simulator Workshop NS-2*, 2006, p. 13.
[43] P. Xie *et al.*, "Aqua-Sim: An NS-2 based simulator for underwater sensor networks," in *Proc. MTS/IEEE Biloxi-Marine Technol. Our Future, Global Local Challenges (OCEANS)*, Oct. 2009, pp. 1–7.
[44] T. Henderson. (2014). *NS-3 Overview*. [Online]. Available: http://www.nsnam.org/docs/ns-3-overview.pdf
[45] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla, "Glomosim: A scalable network simulation environment," Dept. Comput. Sci., UCLA, Los Angeles, CA, USA, Tech. Rep. 990027, 1999, p. 213.
[46] S. Tan and O. Wong, "Framework for event discrete simulation (FEDS)," in *Proc. 7th Asian–Pacific Ind. Eng. Manage. Syst. Conf. (APIEMS)*, Bangkok, Thailand, 2006, pp. 1–8.
[47] S. K. Dhurandher, M. S. Obaidat, and M. Gupta, "An acoustic communication based AQUA-GLOMO simulator for underwater networks," *Human-Centric Comput. Inf. Sci.*, vol. 2, no. 1, pp. 1–14, 2012.
[48] M. Köksal, "A survey of network simulators supporting wireless networks," Middle East Tech. Univ. Ankara, Turkey, Tech. Rep., Oct. 2008.
[49] I. S. Hammoodi, B. G. Stewart, A. Kocian, and S. G. McMeekin, "A comprehensive performance study of OPNET modeler for ZigBee wireless sensor networks," in *Proc. 3rd Int. Conf. Next Generat. Mobile Appl., Services Technol. (NGMAST)*, Sep. 2009, pp. 357–362, doi: 10.1109/NGMAST.2009.12.
[50] A. K. Dwivedi and O. P. Vyas, "An exploratory study of experimental tools for wireless sensor networks," *Wireless Sensor Netw.*, vol. 3, no. 7, pp. 215–240, 2011.
[51] S. K. Dhurandher, S. Misra, M. S. Obaidat, and S. Khairwal, "UWSim: A simulator for underwater sensor networks," *Simulation*, vol. 84, no. 7, pp. 327–338, 2008, doi: 10.1177/0037549708096606.
[52] K. Ovaliadis and N. Savage, "Underwater sensor network simulation tool (USNeT)," *Int. J. Comput. Appl.*, vol. 71, no. 22, p. 39, 2013.
[53] F. Guerra, P. Casari, and M. Zorzi, "World ocean simulation system (WOSS): A simulation tool for underwater networks with realistic propagation modeling," in *Proc. 4th ACM Int. Workshop UnderWater Netw.*, 2009, p. 4.
[54] T. H. Cormen, C. E. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2001.
[55] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. 33rd Annu. Hawaii Int. Conf. Syst. Sci.*, vol. 2. Jan. 2000, p. 10, doi: 10.1109/HICSS.2000.926982.
[56] R. Urick, *Principles of Underwater Sound*. New York, NY, USA: McGraw-Hill, 1983.
[57] R. Coates, *Underwater Acoustic Systems*. Sydney, NSW, Australia: Halstead Press, 1989.
[58] M. Stojanovic, "On the relationship between capacity and distance in an underwater acoustic communication channel," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 11, no. 4, pp. 34–43, 2007.
[59] M. A. Ainslie and J. G. McColm, "A simplified formula for viscous and chemical absorption in sea water," *J. Acoust. Soc. Amer.*, vol. 103, no. 3, pp. 1671–1672, 1998.
[60] M. Felemban and E. Felemban, "Energy-delay tradeoffs for underwater acoustic sensor networks," in *Proc. 1st Int. Black Sea Conf. Commun. Netw. (BlackSeaCom)*, Jul. 2013, pp. 45–49, doi: 10.1109/BlackSeaCom.2013.6623379.
[61] W. Marczak, "Water as a standard in the measurements of speed of sound in liquids," *J. Acoust. Soc. Amer.*, vol. 102, no. 5, pp. 2776–2779, 1997.
[62] J. P. C. Kleijnen, "Verification and validation of simulation models," *Eur. J. Oper. Res.*, vol. 82, no. 1, pp. 145–162, 1995.
[63] I. Bormann. (2010). *Digitizeit Version 1.5. 8B*. [Online]. Available: http://www.digitizeit.de/
[64] O. Balci, "Validation, verification, and testing techniques throughout the life cycle of a simulation study," *Ann. Oper. Res.*, vol. 53, pp. 121–173, Dec. 1994, doi: 10.1007/BF02136828.
[65] O. Balci, "Principles and techniques of simulation validation, verification, and testing," in *Proc. Winter Simulation Conf. Process.*, Dec. 1995, pp. 147–154, doi: 10.1109/WSC.1995.478717.
[66] A. M. Law, "How to build valid and credible simulation models," in *Proc. Winter Simulation Conf. (WSC)*, Dec. 2009, pp. 24–33, doi: 10.1109/WSC.2009.5429312.

MUKHTAR GHALEB received the B.S. degree in computer information system from Zarka Private University, Jordan, in 2004, and the M.S. degree in networking and distributed computation and the Ph.D. degree in computer networks from University Putra Malaysia, Malaysia, in 2008 and 2014, respectively. He began his pursuit in his career with the appointment at Sana'a University, Yemen. He has lectured various courses, such as computer networks, computer architecture, C++ programming language, and advanced distributed database. His research current interests are mobile data gathering, multihop data gathering, routing, power consumption, performance modeling and simulation, wireless sensor networks, and underwater acoustic sensor networks.

EMAD FELEMBAN received the M.Sc. and Ph.D. degrees from The Ohio State University, USA, in 2003 and 2009, respectively. He is currently an Assistant Professor with the Computer Engineering Department, Umm Al-Qura University, Makkah, Saudi Arabia. He is also leading the Transportation and Crowd Management Center of Research Excellence, Umm Al-Qura University, where he is involved in cutting edge research. His research interests include wireless sensor networks algorithm and protocols, smartcity applications, and smart antennas.

SHAMALA SUBRAMANIAM received the B.S., M.S., and Ph.D. degrees in computer science from University Putra Malaysia (UPM), Serdang, Malaysia, in 1996, 1999, and 2002, respectively. She is currently a Professor with the Department of Communication Technology and Networks, Faculty of Computer Science and Information Technology, UPM. Her research interests include computer networks, simulation and modeling, scheduling, and real time system.

ADIL A. SHEIKH received the B.E. degree from the National University of Sciences and Technology, Pakistan, with a Rectors Gold Medal, in 2001, and the double master's degree through the Erasmus Mundus majored in embedded systems from the University of Trento, Italy, and RWTH, Aachen, Germany, in 2007. He is currently pursuing the Ph.D. degree with the University of Grenoble, France. He is also a Researcher with the Science and Technology Unit, Umm Al-Qura University, Makkah, Saudi Arabia. His research interests include visual sensor networks, RFID applications, intelligent transportation systems, and e-health projects.

SAAD BIN QAISAR (SM'16) received the master's and Ph.D. degrees in electrical engineering from Michigan State University, USA, in 2005 and 2009, respectively, under the supervision of Dr. H. Radha. He is currently serving as an Assistant Professor with the School of Electrical Engineering Computer Science, National University of Sciences Technology (NUST), Pakistan. He is the Lead Researcher and the Founding Director of the CoNNekT Lab: Research Laboratory of Communications, Networks and Multimedia, NUST. He is also the Lead Researcher on a joint internet performance measurements study in Pakistan with the Georgia Institute of Technology, a Technical Consultant to United Nations Food and Agriculture Organization, and a Lead Architect of the Pakistan Laboratory Information Management System Project and actively involved with projects funded by the King Abdullah City of Science Technology. He has authored over 50 papers at reputed international venues with a vast amount of work in pipeline. He is also serving as the Chair for Mobile-Computing, Sensing and Actuation for Cyber Physical Systems Workshop in conjunction with the International Conference on Computational Science and Its Applications.