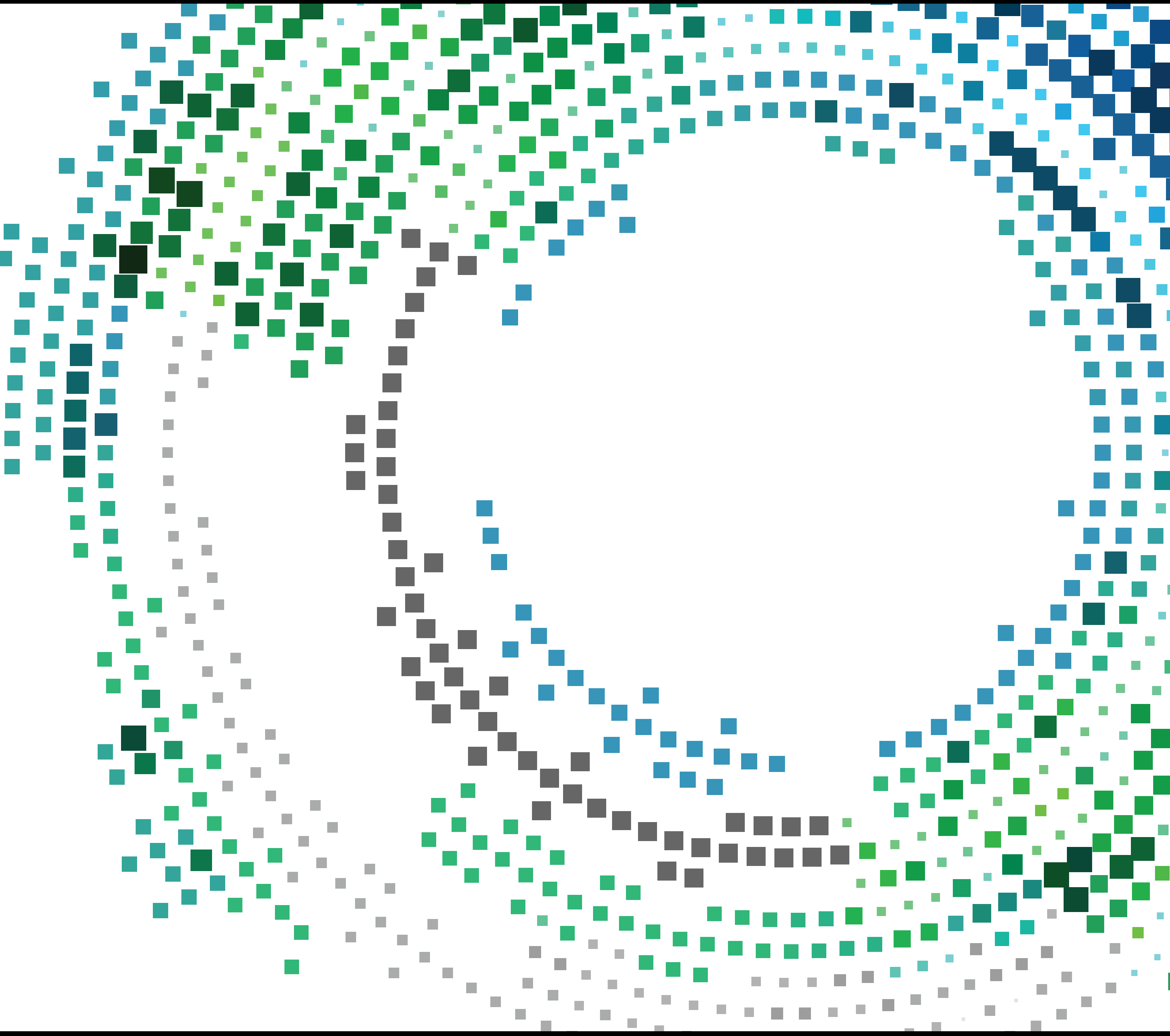
Mobile Information Systems

Mobile Sensor Networks Applications and Confidentiality

Guest Editors: Ansar-Ul-Haque Yasar, Haroon Malik, and Zahoor Khan



Mobile Sensor Networks Applications and Confidentiality

Mobile Sensor Networks Applications and Confidentiality

Guest Editors: Ansar-Ul-Haque Yasar, Haroon Malik, and Zahoor Khan

Editor-in-Chief

Editorial Board

Contents

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2015, Article ID 893438, 2 pages
http://dx.doi.org/10.1155/2015/893438

Editorial

Mobile Sensor Networks Applications and Confidentiality

Ansar-Ul-Haque Yasar,[1] Haroon Malik,[2] and Zahoor Khan[3]

[1] *Transportation Research Institute, Hasselt University, Hasselt, Belgium*
[2] *School of Computer Science, University of Waterloo, Waterloo, Canada*
[3] *Computer Information Sciences, Higher Colleges of Technology, Fujairah Campus, UAE*

Correspondence should be addressed to Ansar-Ul-Haque Yasar; ansar.yasar@uhasselt.be

Received 7 September 2015; Accepted 7 September 2015

This special issue is mostly based on the best papers from the 5th International Conference on Future Networks and Communications (FNC-2014), which was held in Niagara Falls, Ontario, Canada, on August 17–20, 2014. A large number of scientific papers from all around the world were presented in the conference. Many interested techniques were provided in order to enhance the growing demand of Future Networks and Communications technologies, including mobile broadband and all optical networks. All selected papers for this special issue underwent three rounds of rigorous peer-review process. Based on the reviewers' feedback, as well as the evaluations of the Guest Editors, the accepted papers cover remarkable works on new developments in future networked systems such as Wireless Sensor Networks (WSNs), Body Area Networks (BANs), and sensor network confidentiality.

The paper by E.-S. M. El-Alfy and F. Al-Obeidat is entitled "Detecting Cyber-Attacks on Wireless Mobile Networks Using Multicriterion Fuzzy Classifier with Genetic Attribute Selection." This paper investigates a novel methodology based on multicriterion decision making and fuzzy classification that can provide a viable second line of defence for mitigating cyber-attacks. The suggested approach has the advantage of dealing with various types and sizes of attributes related to network traffic such as basic packet headers, content, and time. Using three datasets covering a variety of network attacks, the performance enhancements due to the proposed approach are manifested in terms of detection errors and model construction times.

The paper by Z. Khan et al. is entitled "QPRD: QoS-Aware Peering Routing Protocol for Delay-Sensitive Data in Hospital Body Area Network." The paper proposes a routing protocol by considering the QoS requirements of the Body Area Network (BAN) data packets. A mechanism for handling delay-sensitive packets is provided by this protocol. The scalability of the protocol is demonstrated by simulating a 24-bed real hospital environment with 49 nodes. The experimental results illustrate that QPRD outperforms comparable protocols in terms of higher throughputs, lower overall network traffic, no packets dropped due to MAC buffer overflow, and fewer number of packet timeouts in both mobile and static patient scenarios. Moreover, linear programming based modelling along with graphical analysis is also done.

The paper by A. Dahane et al. is entitled "Energy Efficient and Safe Weighted Clustering Algorithm for Mobile Wireless Sensor Networks." The authors presented a new energy efficient and safe weighted clustering algorithm (ES-WCA) for mobile WSNs using a combination of five metrics. Among these metrics lies the behavioural level metric which promotes a safe choice of a cluster head in the sense where this last one will never be a malicious node. Extensive simulations are done for the performance evaluation of the proposed algorithm.

The paper by B. Al-Madani et al. is entitled "AVL and Monitoring for Massive Traffic Control System over DDS." The authors have proposed a real-time Automatic Vehicle Location (AVL) and monitoring system for traffic control of pilgrims coming towards the city of Makkah in Saudi Arabia based on Data Distribution Service (DDS) specified by the Object Management Group (OMG). The suggested DDS based middleware employs Real-Time Publish/Subscribe (RTPS) protocol that implements many-to-many communication paradigm suitable in massive traffic control applications. This middleware approach helps to locate and track huge number of mobile vehicles and identify all passengers in real time who are coming to perform annual Hajj. Various performance matrices are examined over WLAN using DDS for validation of the proposed framework. Results show that DDS based middleware can meet real-time requirements in large-scale AVL environment.

The paper by N. Jabeur et al. is entitled "Enabling Cyber Physical Systems with Wireless Sensor Networking Technologies, Multiagent Paradigm, and Natural Ecosystems." This paper proposes an agent-based architecture that migrates the complex processing loads outside the physical sensor network while incorporating missing characteristics such as autonomy, intelligence, and context awareness to the WSN. In addition, authors explore the ecosystem metaphor for WSNs with the aim of taking advantage of the efficient adaptation behaviour and strong communication mechanisms used by living organisms. Based on mapping these organisms onto sensors and ecosystems onto WSNs, authors highlight the shortcomings that would prevent WSNs from matching the capabilities of ecosystems at several levels, including structure, topology, goals, communications, and functions. In contrast to existing works, authors use software agents to bridge the gap between WSNs and natural ecosystems, achieve an optimal mapping between both systems, and enhance the capabilities of WSNs to take advantage of bioinspired algorithms.

It is our pleasure to thank all authors for their valuable contributions and their efforts of preparing high quality manuscripts. We would like to thank reviewers for providing their thoughtful and useful comments to authors.

Ansar-Ul-Haque Yasar
Haroon Malik
Zahoor Khan

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2015, Article ID 585432, 13 pages
http://dx.doi.org/10.1155/2015/585432

Research Article

Detecting Cyber-Attacks on Wireless Mobile Networks Using Multicriterion Fuzzy Classifier with Genetic Attribute Selection

El-Sayed M. El-Alfy[1] and Feras N. Al-Obeidat[2]

[1] *College of Computer Sciences and Engineering, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia*
[2] *IBM Research and Development Center, Markham, ON, Canada L3R 9Z7*

Correspondence should be addressed to El-Sayed M. El-Alfy; alfy@kfupm.edu.sa

Received 10 November 2014; Accepted 15 January 2015

Academic Editor: Zahoor Khan

With the proliferation of wireless and mobile network infrastructures and capabilities, a wide range of exploitable vulnerabilities emerges due to the use of multivendor and multidomain cross-network services for signaling and transport of Internet- and wireless-based data. Consequently, the rates and types of cyber-attacks have grown considerably and current security countermeasures for protecting information and communication may be no longer sufficient. In this paper, we investigate a novel methodology based on multicriterion decision making and fuzzy classification that can provide a viable second-line of defense for mitigating cyber-attacks. The proposed approach has the advantage of dealing with various types and sizes of attributes related to network traffic such as basic packet headers, content, and time. To increase the effectiveness and construct optimal models, we augmented the proposed approach with a genetic attribute selection strategy. This allows efficient and simpler models which can be replicated at various network components to cooperatively detect and report malicious behaviors. Using three datasets covering a variety of network attacks, the performance enhancements due to the proposed approach are manifested in terms of detection errors and model construction times.

1. Introduction

The number of wireless and mobile network subscribers is rapidly growing from day to day due to the flexibility of network access anywhere and anytime and the wide range of evolving capabilities that makes our lives easier. However, with these benefits a plethora of security threats also evolve as a result of the increased number of potentially exploitable vulnerabilities. The growth rate of malicious activities and botnets is jumping drastically to alarming levels according to recent security reports [1–3]. It is getting even worse for cross-network services with the emerging 4G/5G network technologies. The new era of information systems combines different environments including wireless ad hoc network, cloud computing, mobile applications, social networks, sensor networks, and smart grids [4].

There is a variety of passive and active cyber-attacks including eavesdropping or packet sniffing, attacks on wireless protocols, injection, port scanning, jamming and denial of service (DoS), fake authentication, address spoofing, session hijacking, man-in-the-middle, replay attacks, vulnerability exploits, traffic analysis, and unauthorized access [5–9].

To mitigate the anticipated risks resulting from various cyber-attacks on critical infrastructures and services, a number of algorithms and technologies have been proposed including encryption standards, digital signatures, antimalware packages, firewalls, and intrusion detection and prevention systems. These methods have been proven to be effective in securing privacy and integrity, controlling access to authorized users, and detecting malicious behaviors of known signatures. However, their performance fails to a great extent to handle sophisticated attacks, zero-day attacks, or attacks with varying signatures. A more flexible and adaptive set of approaches based on machine learning and data mining have been proposed to detect the stochastic deviation from normal behavior patterns. This category of methods is known as anomaly-based intrusion or outlier detection which provides a higher degree of automation and reduces the workload on

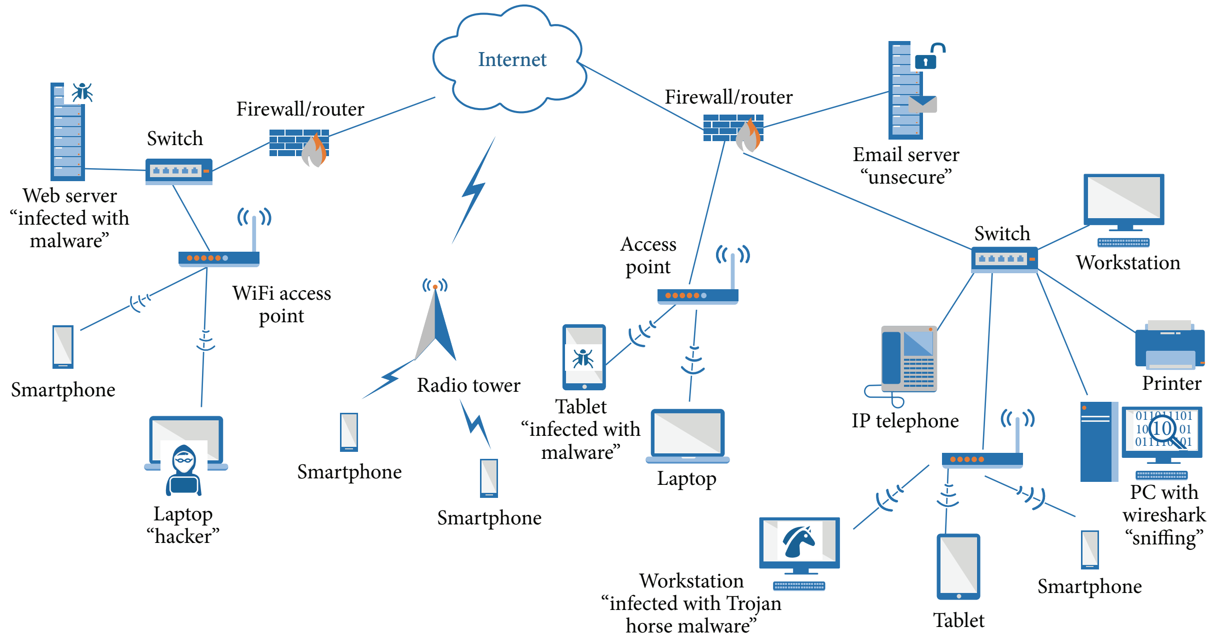


FIGURE 1: Illustration of a network topology with wireless and mobile devices where some devices are infected with malware or hacking.

security experts. Despite the variety of methods that have been proposed in the literature, the research on anomaly detection is still evolving to cope with uncertainties, improve the security, reduce false positive rate, and reduce computational costs [10, 11]. Additionally, since the performance to detect intrusive events is greatly influenced by type and number of attributes utilized [12], it is desirable to analyze and identify the most relevant and influential attributes from the large amount of available data.

Multicriterion decision making techniques were originally devised in the operations research field and have attracted attention of several researchers in various domains such as social psychology, business management, and health care [13, 14]. However, there is not much work done in the area of network security. In this paper, we investigate a new methodology for detecting cyber-attacks in wireless mobile networks based on multicriterion decision making fuzzy classification [15, 16]. The proposed approach is combined with an attribute selection strategy based on genetic algorithms [17]. With the minimum generalization error and the resulting simplicity and reduced computational complexity of the model, the proposed approach is practically feasible to be deployed in different network systems.

The rest of this paper is organized as follows. Section 2 gives a brief background on security in wireless and mobile information systems and Section 3 reviews related work. In Section 4, the proposed methodology is presented. Section 5 describes the adopted datasets and discusses the experimental evaluation and comparison of the proposed approach. Finally, Section 6 concludes the paper.

2. Background and Motives

In heterogeneous wireless mobile environments, there is no well-defined network perimeter; hence, the security administrator cannot enforce security policies even with the existence of firewalls and encryption. This can be attributed to its inherent nature resulting from device mobility, broadcast channels, pervasive use of multivender multidomain applications, and limited resources in wireless end-systems to implement sophisticated security countermeasures. Figure 1 illustrates a typical example of network topology where some machines are infected with malware and others are passively or actively hacking. Attackers only need to discover and exploit a single vulnerability to attack the entire system. Hence, the strength of the system security is as good as the strength of the least secure point in the system.

Wireless devices (such as smart phones, tablets, laptops, or sensors) can be communicating in an isolated environment or connected through a larger distribution network (such as a local area network, a wide area network, or the Internet) using access points. The former is called ad hoc network whereas the latter is known as infrastructure wireless network which is more common. Thus, cyber-attacks can target any of the software or hardware components in this environment including wireless end systems, wireless channels, access points, or the wired distribution network. It is highly important to detect and respond to these attacks to protect the entire system.

3. Related Work

Security of mobile information systems has been a core area in research and development. La Polla et al. in [18] surveyed

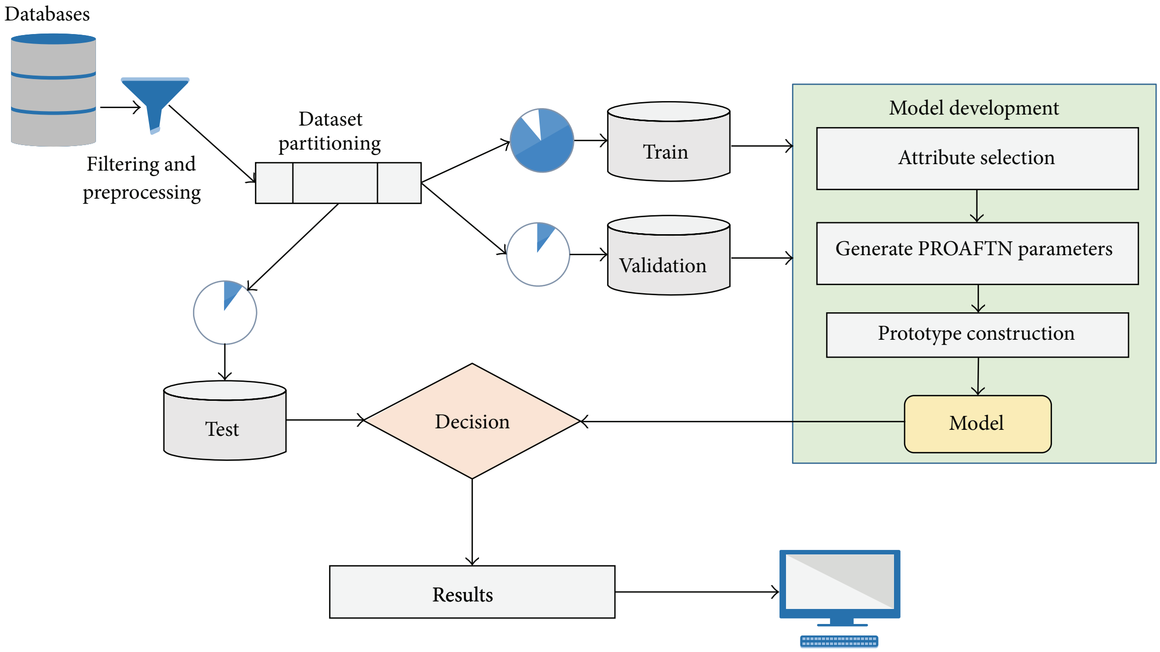


FIGURE 2: Block diagram for training and deploying the cyber-attack detection model.

the state of the art of high level attacks and vulnerabilities targeting mobile devices over the period from 2004 till 2011. They concisely reviewed and categorized known mobile malware including viruses, worms, rootkits, and botnets. They also discussed the proposed security solutions with focus on intrusion detection and trusted platforms. In [9], the authors reviewed the threats, vulnerabilities, and commonly available countermeasures for different components of a wireless network including clients, access points, and transmission medium.

Computational intelligence techniques have many characteristics such as adaption and fault tolerance that made them attractive for research on malware and intrusion detection. In [10], a review of 55 related studies between 2000 and 2007 is presented with focus on single, hybrid, and ensemble classifiers. Another extensive review is presented in [19]. Examples of these techniques include neural networks, fuzzy inference systems, evolutionary algorithms, artificial immune systems, and swarm intelligence. In [20], a naive Bayesian classifier is applied to identify potential intrusions. Trained on a small subset of KDD'99 dataset and tested on a larger subset, this approach showed superior identification rate. In [21], an evaluation of a number of existing machine learning classifiers is presented for dynamic Android malware detection. In [22], another approach for anomaly detection based on multicriterion fuzzy classification with greedy attribute selection is proposed and evaluated on KDD'99.

Combining security technologies can provide more solid multifaceted solutions against intrusion attempts [23]. A number of hybrid machine learning approaches have been proposed as well. For instance, in [24] a machine learning approach is introduced for classifying network activities as normal or abnormal. This approach combines support vector machines with clustering based on self-organized ant colony network. The authors demonstrated that this combination resulted in better classification rate and run time. Anomaly-based intrusion detection has attracted the interest of several researchers [10]. However, these methods can suffer from increased false positive rate. To gain advantage of misuse detection and anomaly detection, Depren et al. proposed a rule-based decision support system to combine the outcomes of decision tree for misuse detection and self-organizing map for modeling normal behavior [25].

Another important stage that can have significant impact on the accuracy and capability of intrusion detection systems is data preprocessing. A review of data preprocessing techniques for anomaly-based network intrusion detection is presented in [12]. During the preprocessing phase, various approaches can be applied such as discretization, normalization, and filtering of most relevant attributes. In [26], the impact of normalization techniques on the performance of support vector machines for intrusion detection is investigated. It has been found that min-max normalization leads to better results in terms of speed and accuracy than other normalization techniques. Another important related issue is attribute selection to reduce the high dimensionality and complexity [27].

Most of the work published in the literature is evaluated using the standard KDD Cup 99 dataset [20, 24, 26, 27]. Despite the fact that this dataset has some drawbacks, it is one of the largest datasets, covers a large number of attacks, and remains dominant to benchmark new techniques. Two more recent datasets have been recently collected and disclosed for the assessment of some attacks on IEEE 802.11 wireless channels [28].

4. Methodology

The overall block diagram for the cyber-attack detection system is shown in Figure 2. It starts with the database of

```
$i$ : prototype's index
$h$: class index
$m$: attribute's index
Select threshold $\beta$ for interval selection
Generate intervals using a discretization technique
Apply greedy hill climbing approach to select most relevant subsets
for each class do
    for each attribute $g$ do
        for every value in attribute $r$ do
            Recursively check all values in the next attribute $g_m$
            if Frequency of values $\geqslant \beta$ then
                Choose intervals for prototype $b_i^h$
            else
                Discard interval and go next (i.e., $I_{g_2 h}^{r_2}$)
            end if
        end for
    end for
end for
```

Algorithm 1: Composing of PROAFTN's prototypes (classification model).

captured traffic. After preprocessing and analyzing traffic records and log files, it performs feature extraction to represent each instance with a vector of relevant attributes. The dataset is then partitioned into train, validation, and test datasets. The train dataset is used to construct the detection model whereas the validation dataset is used during training to evaluate the model to avoid overfitting. The test dataset is used after training is over to evaluate the constructed model performance. The process of partitioning, training, and testing can be repeated if cross validation is required.

When datasets include attributes that are not relevant or may contain redundant attributes, this causes delay in building the classification model and accordingly degrades the classification accuracy. Hence, it is preferable to begin with selecting the most relevant attributes. In our case, we used a genetic algorithm attribute selection strategy. So, the target here is to reduce the hypothesis search space and improve the performance in terms of accuracy, scalability, and efficiency. The idea of genetic algorithms is to start with a random population of candidate solutions and then the population evolves by applying genetic operations, evaluation, and selection [17]. For attribute selection, each chromosome in the population is composed of a binary string with length equal to the total number of attributes where an attribute is selected if its corresponding bit is 1; otherwise, it is dropped. The fitness function depends on being "highly correlated with the class while having low intercorrelation" [29]. The evaluation function for a particular subset of attributes is defined mathematically as follows:

$$f(s) = \frac{k\overline{r}_{\mathrm{ca}}}{\sqrt{k + k(k-1)\overline{r}_{\mathrm{aa}}}}, \qquad (1)$$

where k is the size of the subset s, $\overline{r}_{\mathrm{ca}}$ is the mean of attribute-class correlations, and $\overline{r}_{\mathrm{aa}}$ is the mean of the attribute-attribute correlations. This function will have lower values for attributes that are irrelevant (small value for the numerator) and/or redundant (large value for the denominator).

Once the most relevant attributes are identified, a multicriterion fuzzy classification approach is applied to construct a decision model that can assign unknown behavioral patterns to predefined classes. This type of decision problems requires a comparison between alternatives or patterns based on the scores of attributes using absolute evaluations [30]. In this case, the evaluation is performed by comparing the alternatives to different prototypes of classes, where the category or class is assigned to patterns based on the highest score value. Each prototype is described by a set of attributes and is considered to be a good representative of its class [31]. The complexity of this approach is a function of the number of attributes. Thus, utilizing the smallest subset of relevant attributes greatly improves the time complexity and accuracy of classification. A graphical illustration of the methodology is shown in Figure 3.

To explain how it works, assume the network behavioral pattern is described by a set of m attributes $\{g_1, g_2, \ldots, g_m\}$ and a label c identifying its category which belongs to the k classes $\Omega = \{C^1, C^2, \ldots, C^k\}$. Given a set of N historical patterns P, it is required to construct a classification model $f : P \rightarrow \Omega$ that can accurately predict the target class of each pattern. Once the model is built, it can be used to assign the most relevant class to new unseen behavioral patterns. The model parameters are automatically determined from the training data examples. Then, the constructed model is used for assigning a category to the unseen cases (testing data). This automatic data-driven approach is common to the learning procedures in other machine learning classifiers [32, 33]. Algorithm 1 explains the proposed induction approach through a recursive process to generate the classification model. The tree is constructed in a top-down recursive divide-and-conquer manner, where each branch represents the generated intervals for each attribute. The branches

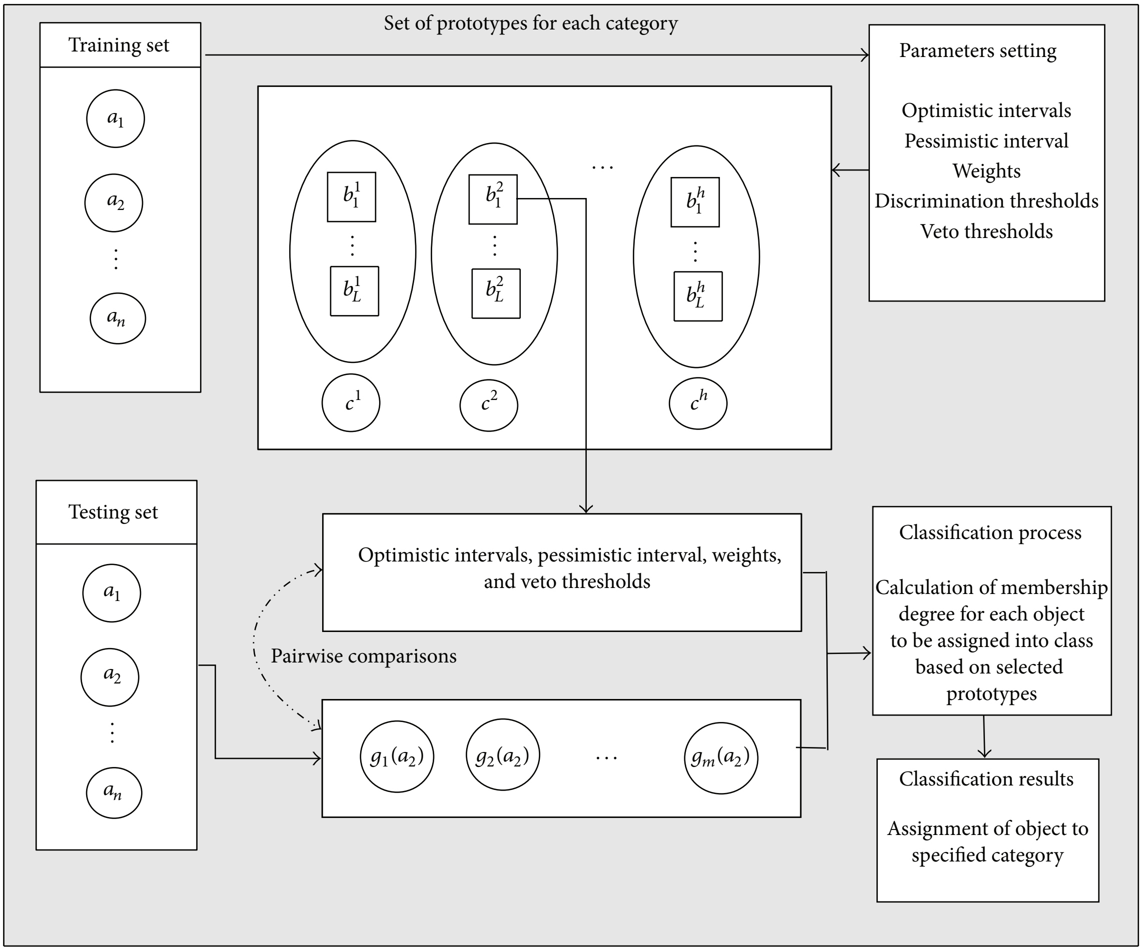


FIGURE 3: Graphical illustration of the multicriterion fuzzy classification procedure.

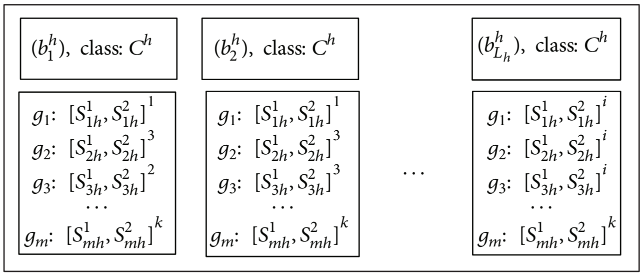

(b_1^h), class: C^h
g_1: $[S_{1h}^1, S_{1h}^2]^1$
g_2: $[S_{2h}^1, S_{2h}^2]^3$
g_3: $[S_{3h}^1, S_{3h}^2]^2$
g_m: $[S_{mh}^1, S_{mh}^2]^k$
(b_2^h), class: C^h
g_1: $[S_{1h}^1, S_{1h}^2]^1$
g_2: $[S_{2h}^1, S_{2h}^2]^3$
g_3: $[S_{3h}^1, S_{3h}^2]^3$
g_m: $[S_{mh}^1, S_{mh}^2]^k$
$(b_{L_h}^h)$, class: C^h
g_1: $[S_{1h}^1, S_{1h}^2]^i$
g_2: $[S_{2h}^1, S_{2h}^2]^i$
g_3: $[S_{3h}^1, S_{3h}^2]^i$
g_m: $[S_{mh}^1, S_{mh}^2]^k$

FIGURE 4: The prototype composition.

are selected recursively to compose the prototypes based on the proposed threshold. Using the generated tree from this algorithm, we can extract the prototypes and then the decision rules, respectively, to be used for classification. Figure 4 illustrates the prototypes' compositions process.

The learning strategy is based on utilizing the training set to compose a set of prototypes for each class. For class C^h, these prototypes are denoted by $B^h = \{b_1^h, b_2^h, \dots, b_{L_h}^h\}$, where L_h is the number of prototypes for this class. For each prototype b_i^h and each attribute g_j, a fuzzy partial indifference relation $C_j(a, b_i^h)$ is defined to measure the degree of resemblance of patterns a to b_i^h according to g_j. This fuzzy relation is characterized by four parameters: the interval $[S_j^1(b_i^h), S_j^2(b_i^h)]$ where $S_j^2(b_i^h) \geq S_j^1(b_i^h)$ and the thresholds $d_j^1(b_i^h)$ and $d_j^2(b_i^h)$. Figure 5 shows a typical example of a fuzzy relation

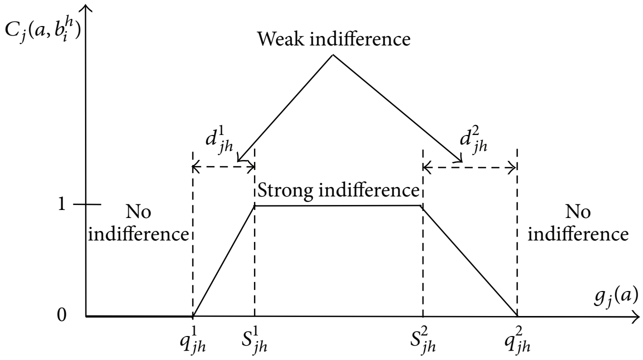


FIGURE 5: A typical example of the partial indifference fuzzy relation between the object a and the prototype b_i^h according to attribute g_j.

Table 1: Some characteristics of the adopted datasets for evaluation.

Dataset	Number of traffic samples			Number of attributes	Number of attack types
	Normal	Malicious	Total		
KDD Cup 99	97278	396743	494021	41	22
WEP/WPA Dataset	15000	9200	24200	15	4
WPA2 Dataset	6000	4000	10000	16	4

with the four parameters illustrated to divide the range of values of g_j into three regions: strong indifference, weak indifference, and no indifference.

In this work, the supervised discretization technique introduced by Fayyad and Irani [34], which is based on the calculation of entropy, is utilized to generate the interval $[S_j^1(b_i^h), S_j^2(b_i^h)]$ for each class prototype and each attribute. To determine the values for $d_j^1(b_i^h)$ and $d_j^2(b_i^h)$, an adjustment/tuning is applied on $S_j^1(b_i^h)$ and $S_j^2(b_i^h)$ to allow more flexibility in assigning patterns to the closest classes. The intervals adjustment can be expressed mathematically as follows:

$$d_j^1\left(b_i^h\right) = \beta S_j^1\left(b_i^h\right), \quad d_j^2\left(b_i^h\right) = \beta S_j^2\left(b_i^h\right); \quad \beta \in [0,1]. \tag{2}$$

The prototypes in this study are constructed based on the frequency of combined values from all attributes in the dataset. After implementing the supervised discretization technique, each attribute will have a set of intervals and nominal values. The learning strategy starts from the first attribute in the list and selects the first interval or nominal value from list of values that belong to the attribute. Then, it proceeds to the next attribute and selects the first interval/nominal value and then counts the frequency of the occurrences for these combined values in each class. If the frequency exceeds the preselected threshold (e.g., more than 15%) then these values are added to the first prototype. The learning continues until all intervals and nominal values are examined by the above discussed strategy. The target is to reach all values for value-attribute from the first attribute to the last one.

To classify a pattern a to the class C^h, PROAFTN calculates the membership degree $\delta(a, C^h)$ as follows:

$$\delta\left(a, C^h\right) = \max\left\{I\left(a, b_1^h\right), I\left(a, b_2^h\right), \ldots, I\left(a, b_{L_h}^h\right)\right\}, \tag{3}$$

where $I(a, b_j^h)$ is the fuzzy indifference relation which is computed as a weighted sum of the partial indifference relations as given by

$$I\left(a, b_i^h\right) = \sum_{j=1}^{m} w_{jh} C_j\left(a, b_i^h\right), \tag{4}$$

where w_{jh} is the weight that measures the importance of a relevant attribute g_j of a specific class C^h:

$$w_{jh} \in [0,1], \quad \sum_{j=1}^{m} w_{jh} = 1. \tag{5}$$

The last step is to assign the pattern a to the class C^h that has the maximum resemblance according to the following decision rule:

$$a \in C^h \Longleftrightarrow \delta\left(a, C^h\right) = \max\left\{\delta\left(a, C^i\right) i \in \{1, \ldots, k\}\right\}. \tag{6}$$

5. Experimental Work

For the sake of evaluation of the proposed methodology, we adopted three datasets in our experimental work. Table 1 shows some of the characteristics of these datasets and more detailed description is provided in the following subsection. Then, we describe the conducted experiments and discuss the results.

5.1. Datasets Description

5.1.1. KDD Cup 99 (KDD'99) Dataset. This dataset consists of processed dump traffic portions of normal and attack connections to a local area network simulating a military network environment [35]. It was prepared from the raw dataset collected and managed by MIT Lincoln Labs as part of the 1998 DARPA Intrusion Detection Evaluation Program. Its first use was in the third International Knowledge Discovery and Data Mining Tools Competition in 1999. Since then, it has become very popular and widely used by most researchers to evaluate and benchmark their research work [20, 24, 26, 27]. The dataset has 494021 traffic samples belonging to 22 different attack types in addition to the normal traffic. These attacks fall into the following four categories: Denial of Service (DoS) such as Syn floods, unauthorized access from a remote machine (R2L) such as password guesses, unauthorized access to local root privileges (U2R) such as rootkits, and probing such as port scanning and nmap. Each connection is described with 41 attributes, as described in Table 2, and has a label identifying the traffic type to be normal or one of the attack types. Three attributes are symbolic and five attributes are binary, whereas the remaining 33 attributes are numeric. As shown in the table, these attributes are divided into four groups: basic attributes of individual connections (9 attributes), content attributes within a connection suggested by domain knowledge (13 attributes), time-based traffic attributes computed using a two-second time window (9 attributes), and host-based traffic attributes computed using a window of 100 connections to the same host (10 attributes).

5.1.2. WEP/WPA Dataset. The traffic samples in this dataset have been recently collected from a controlled wireless home network with enabled WEP/WPA [28]. The network topology

Table 2: Summary of various attributes: category, notation, name, type (numeric, categorical, and binary), statistics, and description.

Cat.	Not.	Name	Type	Statistics		Description
				Min	Max	
Basic						
	a_1	Duration	Num.	0	58329	Connection length in seconds
	a_2	pro_type	Cat.	—	—	Prototype type which can be tcp, udp, or icmp
	a_3	srv	Cat.	—	—	Service on the destination; there are 67 potential values such as http, ftp, telnet, and domain
	a_4	Flag	Cat.	—	—	Normal or error status of the connection; there are 11 potential values, for example, rej, sh
	a_5	src_bytes	Num.	0	693 M	Num. of bytes from the source to the destination
	a_6	dst_bytes	Num.	0	52 M	Num. of bytes from the destination to the source
	a_7	Land	Binary	—	—	Whether conn. from/to same host/port or not
	a_8	wrng_frg	Num.	0	3	Number of wrong fragments
	a_9	urg	Num.	0	3	Number of urgent packets
Content						
	a_{10}	Hot	Num.	0	30	Number of hot indicators
	a_{11}	n_failed_lgns	Num.	0	5	Number of failed login attempts
	a_{12}	logged_in	Binary	—	—	Whether successfully logged in or not
	a_{13}	n_cmprmsd	Num.	0	884	Number of compromised conditions
	a_{14}	rt_shell	Binary	—	—	Whether root shell is obtained or not
	a_{15}	su_attmptd	Num.	0	2	Number of "su root" commands attempted
	a_{16}	n_rt	Num.	0	993	Number of accesses to the root
	a_{17}	n_file_crte	Num.	0	28	Number of create-file operations
	a_{18}	n_shells	Num.	0	2	Number of shell prompts
	a_{19}	n_access_files	Num.	0	8	Number of operations on access control files
	a_{20}	n_obnd_cmds	Num.	0	0	Number of outbound commands in an ftp session
	a_{21}	is_hot_lgn	Binary	—	—	Whether login belongs to hot list or not
	a_{22}	is_guest_lgn	Binary	—	—	Whether login is guest or not
t_traffic (using a window of 2 seconds)						
	a_{23}	cnt	Num.	0	511	Number of same-host connections as the current connection in the past 2 seconds
	a_{24}	srv_cnt	Num.	0	511	Num. of same-host conn. to the same service as the current connection in the past 2 seconds
	a_{25}	syn_err	Num.	0	1	Percentage of same-host conn. with syn errors
	a_{26}	srv_syn_err	Num.	0	1	Percentage of same-service conn. with syn errors
	a_{27}	rej_err	Num.	0	1	Percentage of same-host conn. with rej errors
	a_{28}	srv_rej_err	Num.	0	1	Percentage of same-service conn. with rej errors
	a_{29}	sm_srv_r	Num.	0	1	Percentage of same-host conn. to same service
	a_{30}	dff_srv_r	Num.	0	1	Percentage of same-host conn. to different services
	a_{31}	srv_dff_hst_r	Num.	0	1	Percentage of same-service conn. to different hosts
h_traffic (using a window of 100 connections)						
	a_{32}	h_cnt	Num.	0	255	Number of same-host connections as the current connection in the past 100 connections
	a_{33}	h_srv_cnt	Num.	0	255	Num. of same-host conn. to the same service as the current connection in the past 100 connections
	a_{34}	h_sm_srv_r	Num.	0	1	Percentage of same-host conn. to same service
	a_{35}	h_dff_srv_r	Num.	0	1	Percentage of same-host conn. to different services
	a_{36}	h_sm_sr_prt_r	Num.	0	1	Percentage of same-service conn. to different hosts
	a_{37}	h_srv_dff_hst_r	Num.	0	1	Percentage of same-service conn. to different hosts
	a_{38}	h_syn_err	Num.	0	1	Percentage of same-host conn. with syn errors
	a_{39}	h_srv_syn_err	Num.	0	1	Percentage of same-service conn. with syn errors
	a_{40}	h_rej_err	Num.	0	1	Percentage of same-host conn. with rej errors
	a_{41}	h_srv_rej_err	Num.	0	1	Percentage of same-service conn. with rej errors

Table 3: Comparisons of accuracy for different approaches using 10-fold cross validation (results are approximated to two decimal digits). All model constructions have taken reasonable time except SVM and MLP.

Approach	KDD'99 dataset		WEP/WPA dataset		WPA2 dataset	
	Acc (%)	Time (sec)	Acc (%)	Time (sec)	Acc (%)	Time (sec)
With attribute selection						
Proposed	99.92	7	85.70	6	90.10	4
NB	94.57	3	77.38	2	68.82	2
SVM	97.61	27	83.23	21	80.24	19
MLP	96.15	31	84.32	29	88.46	23
Without attribute selection						
Proposed	99.20	10	84.89	8	91.82	6
NB	93.28	3.8	77.24	4	76.41	3
SVM	97.58	33	83.02	28	83.26	22
MLP	96.24	48	78.73	34	90.44	29

Table 4: The KDD'99 per-class performance of the proposed method with and without attribute selection (approximated to three decimal digits).

Normal/attack	Count	With attribute selection				Without attribute selection			
		Precision	Recall	F_1	AUC	Precision	Recall	F_1	AUC
Normal	97278	0.999	0.999	0.999	1	0.990	0.989	0.989	0.990
Back	2203	1	0.999	0.999	0.999	0.987	0.987	0.987	0.989
Buffer_overflow	30	1	0.692	0.818	0.846	0.723	0.678	0.700	0.848
ftp_write	8	0.5	0.5	0.5	0.75	0	0.000	0.000	0.573
Guess_passwd	53	0.909	0.952	0.93	0.976	0.933	0.933	0.933	0.962
imap	12	1	0.4	0.571	0.7	0.157	0.240	0.190	0.670
ipsweep	1247	0.892	0.988	0.938	0.994	0.985	0.983	0.984	0.989
Land	21	0.8	1	0.889	1	0.847	0.937	0.890	0.919
Loadmodule	9	0.333	0.2	0.25	0.6	0	0.000	0.000	0.766
Multihop	7	0.25	0.333	0.286	0.667	0.276	0.323	0.298	0.847
Neptune	107201	1	1	1	1	0.990	0.990	0.990	0.990
nmap	231	0.906	0.358	0.513	0.679	0.938	0.981	0.959	0.981
Perl	3	0.5	0.5	0.5	0.75	0.323	0.990	0.490	0.980
phf	4	0.25	1	0.4	1	0.990	0.657	0.790	0.990
pod	264	0.986	0.973	0.98	0.987	0.990	0.986	0.988	0.990
Portsweep	1040	0.981	0.976	0.979	0.988	0.977	0.982	0.979	0.987
Rootkit	10	0.5	0.333	0.4	0.667	0	0.000	0.000	0.662
Satan	1589	0.986	0.989	0.988	0.995	0.981	0.984	0.982	0.987
Smurf	280790	1	1	1	1	0.990	0.990	0.990	0.990
Spy	2	1	1	1	1	0	0.000	0.000	0.443
Teardrop	979	0.994	0.997	0.996	0.999	0.989	0.990	0.989	0.989
Warezclient	1020	0.997	0.982	0.99	0.991	0.968	0.982	0.975	0.988
Warezmaster	20	0.333	0.5	0.4	0.75	0.790	0.752	0.770	0.910

is a single basic service set (BSS) consisting of one access point (AP) connected to the Internet and three stations: one generating real HTTP and FTP traffic (STA1), one running Wireshark to monitor the network and capture traffic (STA2), and one for generating attacks (STA3). In addition to normal traffic, four types of attacks are reported: ChopChop, deauthentication, duration, and fragmentation. There are a total of 24200 traffic samples; 15000 of them belong to normal traffic whereas the rest are divided equally for each attack type. The captured traffic from normal and attack processes is preprocessed using Tshark to extract 15 attributes from the MAC headers.

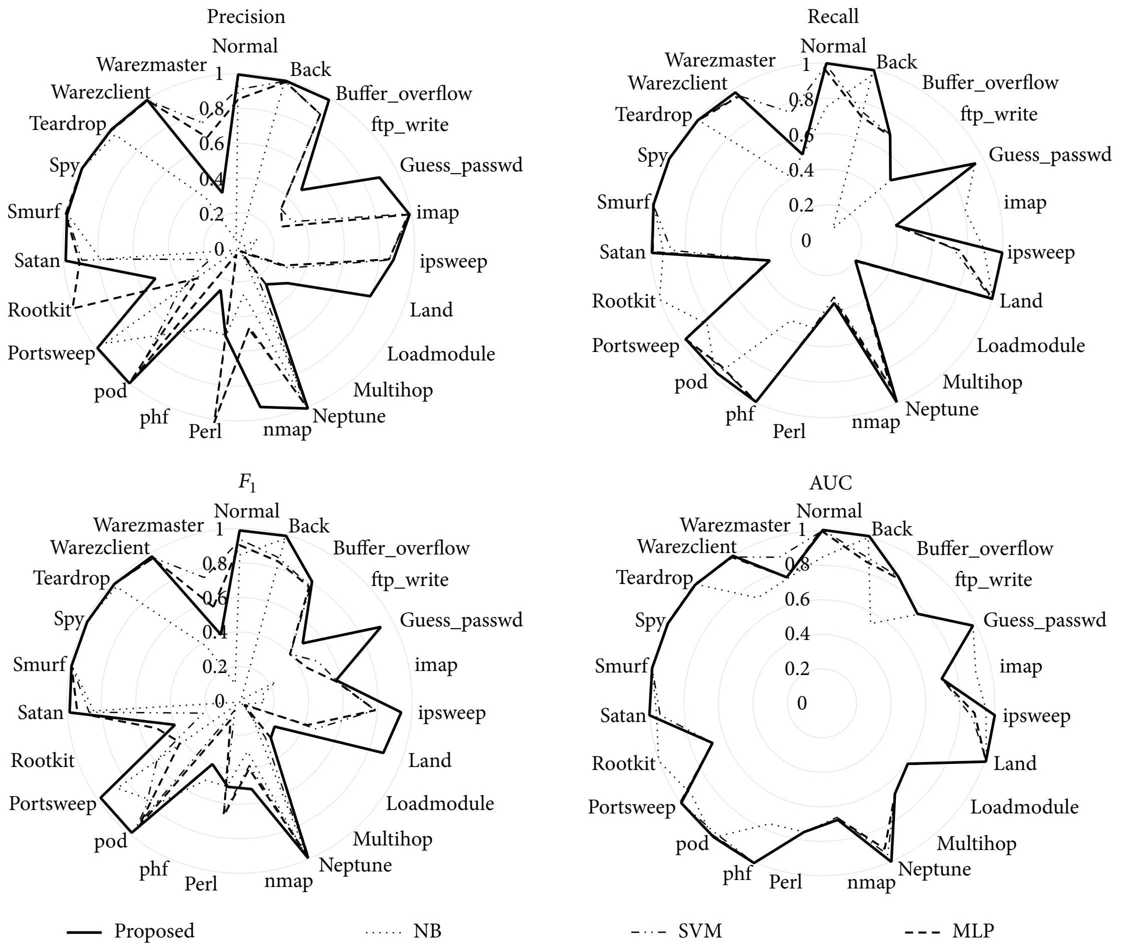


Figure 6: Comparing the per-class results for KDD'99 dataset using the reduced attribute vector (due to attribute selection) with various methods in terms of precision, recall, F_1 measure, and AUC.

5.1.3. WPA2 Dataset. The third dataset has been collected from a corporate network with enabled WPA2 encryption [28]. In this network, there are two access points connected to a local area network switch, which is connected to an authentication server (AS) and the Internet. In this scenario, there are five stations: three generating traffic, one monitoring the network, and one hacking. Here, there are four attack types: deauthentication, fake authentication, fake AP, and Syn flooding. The total number of traffic samples is 10000, where 6000 of them belong to normal traffic and the rest are distributed equally for each attack type. Each sample is processed as in the second dataset with Tshark and described with 16 attributes.

5.2. Performance Measures. We used 10-fold cross validation to evaluate and compare the performance of the proposed methodology. The performance is reported in terms of accuracy (Acc), recall (true positive rate), precision, and F_1 measure. These measures are computed as follows:

$$\text{Acc} = \frac{(\text{tp} + \text{tn})}{(\text{tp} + \text{tn} + \text{fp} + \text{fn})},$$

$$\text{Recall} = \frac{\text{tp}}{(\text{tp} + \text{fn})},$$

$$\text{Precision} = \frac{\text{tp}}{(\text{tp} + \text{fp})},$$

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{(\text{precision} + \text{recall})}, \tag{7}$$

where tp refers to true positive, tn refers to true negative, fp refers to false positive, and fn refers to false negative. We also compared the area under the receiver operating characteristic (ROC) curve (AUC) and the time to construct the attack detection model.

5.3. Experiments and Results. The proposed methodology was implemented in Java and ran in a Linux machine. We applied it to the datasets described above with and without attribute selection. For the first dataset, KDD'99, the application of the attribute selection strategy has resulted in only 17 out of the 41 attributes as relevant attributes. Referring to Table 2, the selected attributes are a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_{10},

Table 5: The WEP/WPA per-class performance of the proposed method with and without attribute selection (approximated to three decimal digits).

Normal/attack	Count	With attribute selection				Without attribute selection			
		Precision	Recall	F_1	AUC	Precision	Recall	F_1	AUC
Normal	15000	0.822	1	0.902	1	0.817	0.996	0.898	0.993
ChopChop	2300	1	0.326	0.491	0.627	0.855	0.130	0.226	0.521
Deauthentication	2300	0.945	0.971	0.958	0.984	0.938	0.981	0.959	0.989
Duration	2300	0.970	0.997	0.983	0.999	0.966	0.986	0.976	0.992
Fragmentation	2300	0.994	0.21	0.347	0.563	0.968	0.338	0.50	0.632

Table 6: The WPA2 per-class performance of the proposed method with and without attribute selection (approximated to three decimal digits).

Normal/attack	Count	With attribute selection				Without attribute selection			
		Precision	Recall	F_1	AUC	Precision	Recall	F_1	AUC
Normal	6000	0.906	0.935	0.920	0.916	0.906	0.970	0.937	0.961
Fake AP	1000	0.998	0.952	0.974	0.973	0.985	0.945	0.965	0.969
Fake authentication	1000	0.734	0.483	0.582	0.713	0.934	0.448	0.605	0.694
Deauthentication	1000	0.984	0.967	0.975	0.981	0.981	0.978	0.980	0988
Syn flooding	1000	0.822	0.997	0.901	0.998	0.874	0.997	0.931	0.998

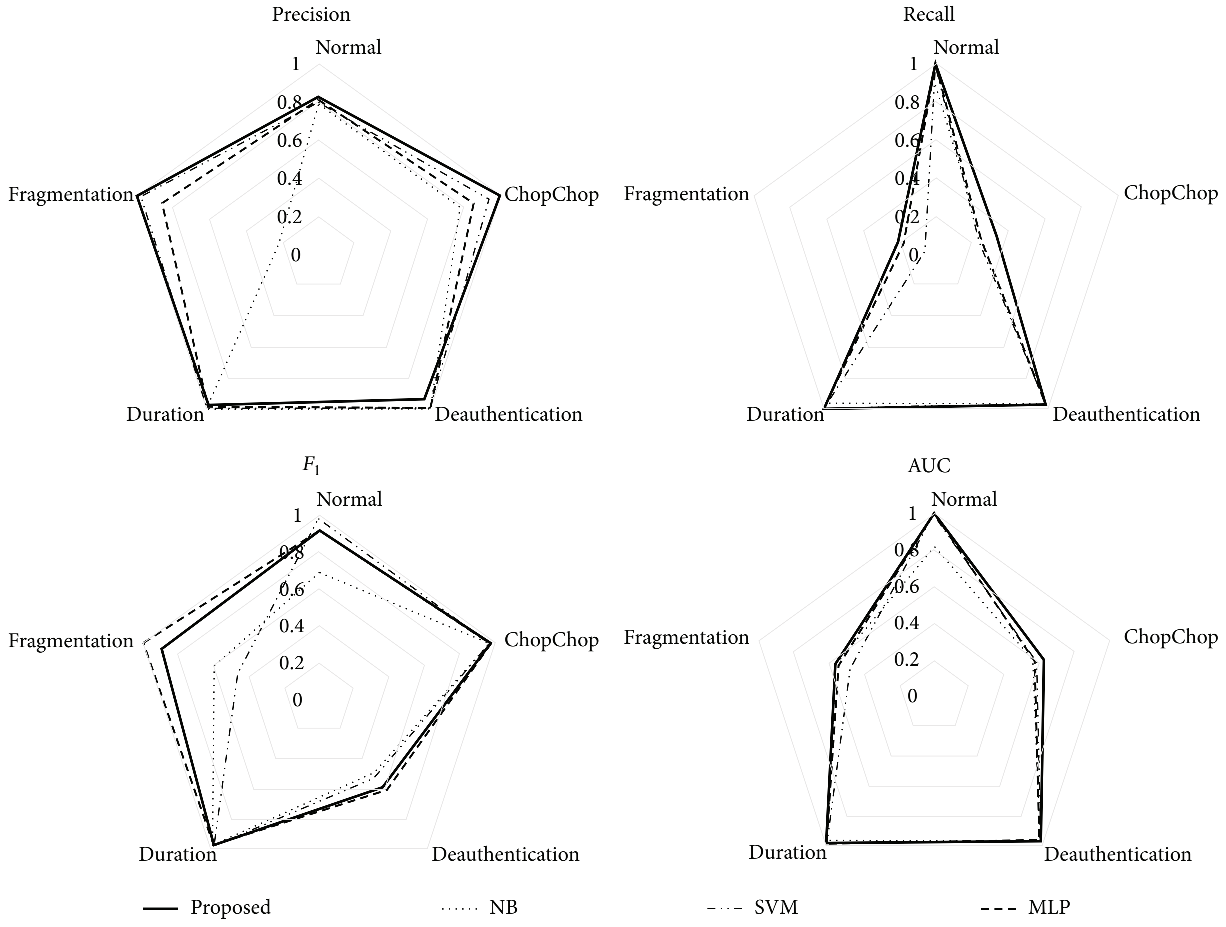


Figure 7: Comparing the per-class results for WEP/WPA dataset using the reduced attribute vector (due to attribute selection) with various methods in terms of precision, recall, F_1 measure, and AUC.

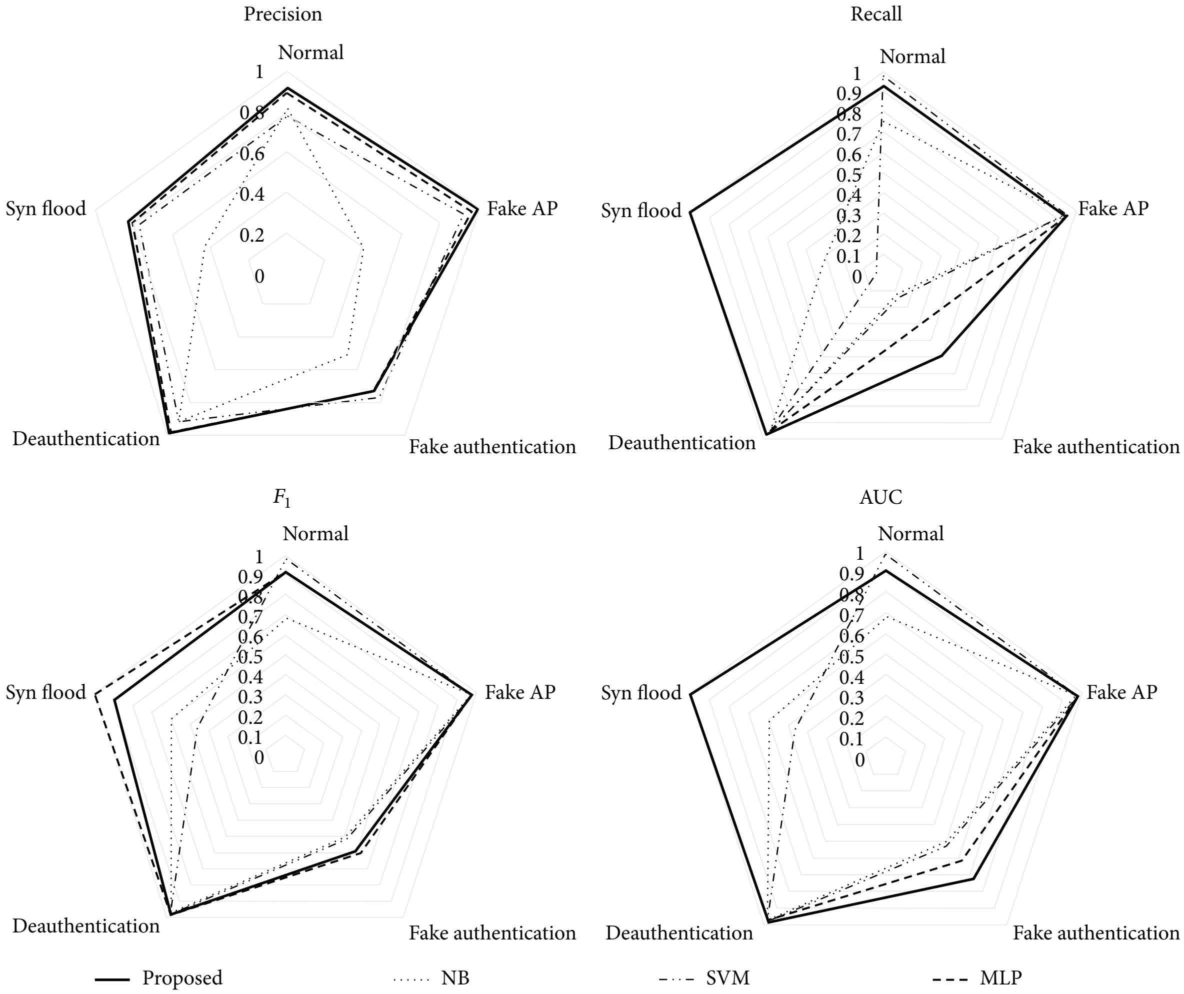


Figure 8: Comparing the per-class results for WPA2 dataset using the reduced attribute vector (due to attribute selection) with various methods in terms of precision, recall, F_1 measure, and AUC.

a_{12}, a_{19}, a_{23}, a_{29}, a_{30}, a_{31}, a_{33}, a_{34}, and a_{38}. For WEP/WPA dataset, only 7 attributes were selected whereas for the WPA2 dataset, only 5 attributes were selected.

We conducted a comparative study with three popular machine learning algorithms implemented in [36] with default settings using the stratified 10-fold cross validation. Table 3 summarizes the performance of the proposed method with and without attribute selection and compared it to the other classifiers: naive Bayes (NB), support vector machine (SVM), and multilayer perceptron (MLP). The reported time is the model construction time (in other words, it does not include the time for attribute selection). This table shows consistent results for the three considered datasets. All model constructions have taken reasonable times except for SVM and MLP. Although NB can take slightly less time than the proposed method, its accuracy is much lower. This demonstrates that the proposed methodology can outperform other techniques with improved accuracy and simpler models even with few selected attributes. In general, we observed that the performance for the KDD'99 dataset is much better than for the other datasets. This can be due to the size and nature of the dataset since KDD'99 has more samples and attributes covering larger parts of the search space.

For the proposed methodology, we also reported the performance for each class in the three datasets in terms of precision, recall, F_1 measure, and AUC. These results are shown in Tables 4, 5, and 6. For the first dataset, KDD'99, the distribution of traffic samples is skewed where some attacks are very rare. We can notice that the proposed methodology is very accurate when enough samples exist. For the other two datasets, the performance is very high except for two attack types. This can be attributed to incomplete attribute set to distinguish between all traffic types. The comparisons of the per-class performance with other methods are shown in Figures 6, 7, and 8. In these figures, it is desirable to cover larger area of the shape in each direction (class type). Similar conclusion can be drawn as above, where the proposed methodology is promising and can be effective for cyber-attack detection.

6. Conclusion

This paper presents a novel security mechanism for cyber-attack detection in wireless mobile networks. It uses historical data to build detection models with the most influential attributes. The proposed hybrid methodology is based on multicriterion fuzzy classification augmented with a meta-heuristic approach using a genetic algorithm for attribute selection strategy. The constructed predictive model is then deployed to classify unknown incoming traffic. After capturing, preprocessing, and analyzing traffic, the relevant attributes are then extracted and integrated with the model to decide whether the activity is normal or malicious. Three datasets with various natures and different cyber-attacks are utilized to evaluate and compare the effectiveness of the proposed methodology to detect cyber-attacks on different components of a mobile wireless network. Results showed that the proposed methodology behaved consistently for all datasets with promising detection accuracies and model construction times. In some attacks, the performance was relatively low. However, this can be due to the insufficient number of captured samples, imbalanced distribution of the dataset, or insufficient extracted attributes from the raw traffic. As future work, it is intended to explore more attacks and other datasets and subsequently improve our methodology further.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The first author would like to acknowledge the support provided by King Abdulaziz City for Science and Technology (KACST) through the Science & Technology Unit at King Fahd University of Petroleum & Minerals (KFUPM) for funding this work through Project no. 11-INF1658-04 as part of the National Science, Technology, and Innovation Plan.

References

[1] Cisco, *Cisco 2014 Annual Security Report*, 2014, https://www.cisco.com/web/offer/gist_ty2_asset/Cisco_2014_ASR.pdf.

[2] Sophos Security Threat Report 2014, http://www.sophos.com/en us/medialibrary/PDFs/other/sophos-security-threat-report-2014.pdf.

[3] *Mobile Security Review, AV-Comparatives*, 2014, http://www.av-comparatives.org/wp-content/uploads/2014/09/avc_mob_201407_en.pdf.

[4] M. Shiraz, A. Gani, R. H. Khokhar, and R. Buyya, "A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1294–1313, 2013.

[5] I. M. Chapman, S. P. Leblanc, and A. Partington, "Taxonomy of cyber attacks and simulation of their effects," in *Proceedings of the Military Modeling & Simulation Symposium*, pp. 73–80, Society for Computer Simulation International, 2011.

[6] G. Lehembre, "Wi-Fi security—WEP, WPA and WPA2," *Hackin9*, vol. 6, 2005.

[7] E. Tews and M. Beck, "Practical attacks against WEP and WPA," in *Proceedings of the 2nd ACM Conference on Wireless Network Security (WiSec '09)*, pp. 79–85, March 2009.

[8] A. H. Lashkari, M. M. S. Danesh, and B. Samadi, "A survey on wireless security protocols (WEP, WPA and WPA2/802.11i)," in *Proceedings of the IEEE International Conference on Computer Science and Information Technology*, pp. 48–52, Beijing, China, August 2009.

[9] M.-K. Choi, R. J. Robles, C.-H. Hong, and T.-H. Kim, "Wireless network security: vulnerabilities, threats and countermeasures," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 3, no. 3, pp. 77–86, 2008.

[10] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: a review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11994–12000, 2009.

[11] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: techniques, systems and challenges," *Computers & Security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[12] J. J. Davis and A. J. Clark, "Data preprocessing for anomaly based network intrusion detection: a review," *Computers & Security*, vol. 30, no. 6-7, pp. 353–375, 2011.

[13] N. E. Fenton and W. Wang, "Risk and confidence analysis for fuzzy multicriteria decision making," *Knowledge-Based Systems*, vol. 19, no. 6, pp. 430–437, 2006.

[14] C. Zopounidis and M. Doumpos, "Multicriteria classification and sorting methods: a literature review," *European Journal of Operational Research*, vol. 138, no. 2, pp. 229–246, 2002.

[15] F. Al-Obeidat, N. Belacel, J. A. Carretero, and P. Mahanti, "An evolutionary framework using particle swarm optimization for classification method proaftn," *Applied Soft Computing Journal*, vol. 11, no. 8, pp. 4971–4980, 2011.

[16] N. Belacel, "Multicriteria assignment method PRO AFTN: methodology and medical application," *European Journal of Operational Research*, vol. 125, no. 1, pp. 175–183, 2000.

[17] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine Learning*, vol. 3, no. 2-3, pp. 95–99, 1998.

[18] M. La Polla, F. Martinelli, and D. Sgandurra, "A survey on security for mobile devices," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 446–471, 2013.

[19] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: a review," *Applied Soft Computing Journal*, vol. 10, no. 1, pp. 1–35, 2010.

[20] H. Altwaijry and S. Algarny, "Bayesian based intrusion detection system," *Journal of King Saud University—Computer and Information Sciences*, vol. 24, no. 1, pp. 1–6, 2012.

[21] B. Amos, H. Turner, and J. White, "Applying machine learning classifiers to dynamic android malware detection at scale," in *Proceedings of the 9th International Wireless Communications and Mobile Computing Conference (IWCMC '13)*, pp. 1666–1671, Sardinia, Italy, July 2013.

[22] E.-S. M. El-Alfy and F. N. Al-Obeidat, "A multicriterion fuzzy classification method with greedy attribute selection for anomaly-based intrusion detection," *Procedia Computer Science*, vol. 34, pp. 55–62, 2014.

[23] K. Satpute, S. Agrawal, J. Agrawal, and S. Sharma, "A survey on anomaly detection in network intrusion detection system using particle swarm optimization based machine learning techniques," in *Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA '13)*, pp. 441–452, Springer, 2013.

[24] W. Feng, Q. Zhang, G. Hu, and J. X. Huang, "Mining network data for intrusion detection through combining SVMs with ant colony networks," *Future Generation Computer Systems*, vol. 37, pp. 127–140, 2014.

[25] O. Depren, M. Topallar, E. Anarim, and M. K. Ciliz, "An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks," *Expert Systems with Applications*, vol. 29, no. 4, pp. 713–722, 2005.

[26] W. Li and Z. Liu, "A method of SVM with normalization in intrusion detection," *Procedia Environmental Sciences*, vol. 11, pp. 256–262, 2011.

[27] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "Feature selection and classification in multiple class datasets: an application to KDD Cup 99 dataset," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5947–5957, 2011.

[28] D. W. F. L. Vilela, E. T. Ferreira, A. A. Shinoda, N. V. de Souza Araujo, R. de Oliveira, and V. E. Nascimento, "A dataset for evaluating intrusion detection systems in ieee 802.11 wireless networks," in *Proceedings of the IEEE Colombian Conference on Communications and Computing (COLCOM '14)*, pp. 1–5, Bogotá, Colombia, June 2014.

[29] M. A. Hall, *Correlation-based feature selection for machine learning [Ph.D. thesis]*, The University of Waikato, 1999.

[30] J. Léger and J.-M. Martel, "A multi-criteria assignment procedure for a nominal sorting problematic," *European Journal of Operational Research*, vol. 138, no. 2, pp. 349–364, 2002.

[31] K. Jabeur and A. Guitouni, "A generalized framework for concordance/discordance-based multi-criteria classification methods," in *Proceedings of the 10th International Conference on Information Fusion*, pp. 1–8, July 2007.

[32] P. W. Baim, "A method for attribute selection in inductive learning systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 888–896, 1988.

[33] D. M. Dutton and G. V. Conroy, "A review of machine learning," *The Knowledge Engineering Review*, vol. 12, no. 4, pp. 341–367, 1997.

[34] U. Fayyad and K. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI '93)*, pp. 1022–1029, 1993.

[35] KDD Cup 1999 dataset for network-based intrusion detection systems, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[36] H. Witten, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, 2005.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2015, Article ID 908315, 15 pages
http://dx.doi.org/10.1155/2015/908315

Research Article

Enabling Cyber Physical Systems with Wireless Sensor Networking Technologies, Multiagent System Paradigm, and Natural Ecosystems

Nafaâ Jabeur,[1] Nabil Sahli,[1] and Sherali Zeadally[2]

[1] *Computer Science Department, German University of Technology in Oman (GUtech), P.O. Box 1816, 130 Muscat, Oman*
[2] *College of Communication and Information, University of Kentucky, Lexington, KY 40506, USA*

Correspondence should be addressed to Nabil Sahli; nabil.sahli@gutech.edu.om

Received 10 November 2014; Accepted 15 January 2015

Academic Editor: Ansar-Ul-Haque Yasar

Wireless sensor networks (WSNs) are key components in the emergent cyber physical systems (CPSs). They may include hundreds of spatially distributed sensors which interact to solve complex tasks going beyond their individual capabilities. Due to the limited capabilities of sensors, sensor actions cannot meet CPS requirements while controlling and coordinating the operations of physical and engineered systems. To overcome these constraints, we explore the ecosystem metaphor for WSNs with the aim of taking advantage of the efficient adaptation behavior and communication mechanisms of living organisms. By mapping these organisms onto sensors and ecosystems onto WSNs, we highlight shortcomings that prevent WSNs from delivering the capabilities of ecosystems at several levels, including structure, topology, goals, communications, and functions. We then propose an agent-based architecture that migrates complex processing tasks outside the physical sensor network while incorporating missing characteristics of autonomy, intelligence, and context awareness to the WSN. Unlike existing works, we use software agents to map WSNs to natural ecosystems and enhance WSN capabilities to take advantage of bioinspired algorithms. We extend our architecture and propose a new intelligent CPS framework where several control levels are embedded in the physical system, thereby allowing agents to support WSNs technologies in enabling CPSs.

1. Introduction

Recent technological advances have been shifting computation to a wide variety of devices, including toys, home appliances, and phones. In addition to increasing their computing capabilities, advances are also enabling these devices to interact with each other in order to achieve individual or common goals which they are not able to achieve individually. These capabilities are bringing new research and development opportunities to a wide range of application domains, such as smart grid, healthcare, and intelligent road safety [1]. They are also bringing about new challenges with respect to the control of the physical environment of which computing capabilities have become an integral part. The concept of Cyber Physical System (CPS) has emerged as a promising tool where the operations of the physical and engineered systems are monitored, controlled, coordinated, and integrated by means of a computing and communication core [2]. In such system, sensors, actuators, and embedded devices are networked to sense, monitor, and control the physical world. The increasing pervasiveness of wireless sensor networks (WSNs) in many applications makes these technologies an important component of CPS designs [3]. WSNs are particularly deployed as interfaces through which *in situ* data are collected about/from the physical environment and then transferred to the cyber environment as well as interfaces through which new instructions/parameters are injected from the cyber environment to the physical environment.

Enabling Cyber Physical Systems with WSN technologies is not straightforward. Several challenges must be addressed, including the integration of appliances with different communication protocols, the mobility of sensor nodes, and the delivery of sensor data to the cyber system on time [3]. The solutions for such challenges will particularly depend on how researchers will deal with WSN recent emerging requirements, such as the ability to integrate spatial concepts, promote adaptability, support diversity and evolution, and allow low-cost, long-term evolutions when designing these systems [4]. These requirements cannot be fulfilled by simply adopting and adjusting traditional paradigms such as service-oriented architectures (SOA) [5] where services are triggered and coordinated according to predefined interaction patterns making self-adaptability and self-management hard to be integrated in a system [4].

In this context, a nature-inspired approach may be an important research direction. In such an approach, each service could behave as an autonomous organism in an ecosystem. An ecosystem can be defined as a dynamic compound formed by material circulation and energy flow, with reciprocity, interdependency, and self-organization functions within an interactive natural environment [6]. The physical, chemical, biological, and social interactions between system components are not determined by predefined centralized patterns but rather by a small set of natural laws [4] from which complex patterns of interactions dynamically emerge via self-organization.

Metaphors inspired by natural ecosystems, including digital [7], knowledge [8], and business ecosystems [9], have provided an important source of relevant knowledge, models, and algorithms thereby allowing efficient solutions in many fields. They are well suited for the development of new computing systems, particularly when these systems are complex, large-scaled, decentralized, open, and heterogeneous. This is the case with WSNs which commonly consist of spatially distributed nodes, operating unattended with severe restrictions on their computation capabilities, memory space, communication bandwidth, and battery lifetime. These nodes should self-organize while collaborating and/or competing for the limited resources in similar ways to the living organisms. We thus argue that an ecosystem metaphor would solve some of the current WSN problems and consequently enable the deployment of CPSs with WSN technologies. To fully exploit this metaphor, we propose a better mapping between WSNs and natural ecosystems. We adopt the multiagent system paradigm [10] which already has a well-defined set of formalisms, algorithms, and methodologies to bridge the gap between the WSNs and natural ecosystems. The multiagent system paradigm will particularly support the WSN in integrating the physical and the virtual environments of a CPS.

In the remainder of this paper, Section 2 describes the related works. Section 3 presents a mapping between sensors and living organisms as well as a mapping between natural ecosystems and WSNs. It also highlights the shortcomings of WSNs within this mapping. Section 4 describes our proposed agent-based architecture, called ABAMA, which aims to address the WSN shortcomings and improves the mapping between WSNs and natural ecosystems. Section 5 presents our new CPS architecture where software agents are used to intelligently bridge the cyber and physical worlds while integrating ABAMA concepts and capabilities. Section 6 presents few opportunities that this new architecture offers as well as the main challenges that the WSN community needs to investigate in the future.

2. Related Works

Cyber Physical Systems have special features in terms of their architectural design and operating mode. They connect distributed, potentially mobile and heterogeneous, devices which may collaborate or compete for resources for their optimal operations. In addition to providing these devices with the necessary data and instructions for their operations, the CPSs have to monitor these operations and ensure a fair, on-time, and convenient resource use and allocation to different parties. Thanks to their capabilities of remote and distributed sensing and their real-time data analysis and routing, WSNs are integral parts of CPSs. Lin et al. [3] summarized the research works that have addressed the fundamental role of WSN in CPS in terms of deployment (e.g., [11]), localization (e.g., [12]), coverage (e.g., [13]), data gathering (e.g., [14]), and communication (e.g., [15]). We argue that these benefits could be further enhanced if WSNs can use natural ecosystem concepts.

According to our literature review, no research work has identified the WSN itself as an ecosystem in the context of CPS. However, few researchers did explore this area in a pure WSN environment. In this context, Jones and colleagues [16] represented sensors as organisms in an ecosystem which are distributed throughout a geographic region. The proposed representation assumes that every sensor has exactly 8 neighbors and can only transmit to them. Barolli et al. [17] implemented a simulation system for WSN using an approach inspired by digital ecosystems. These systems use evolutionary computing to implement properties such as self-organization and scalability inspired by natural ecosystems. In spite of its good performance, the simulation did not highlight any similarities between WSNs and the natural ecosystem.

To the best of our knowledge, the only research work which has used the natural ecosystem as metaphor to model WSNs is presented by Antoniou and Pitsillides [18]. The authors proposed a bioinspired congestion control approach for streaming applications in WSNs and considered a WSN to be analogous to an ecosystem. In particular, sensors are compared to species which live and interact together to meet their needs for survival and coexistence. In WSNs, traffic flows are seen as species that compete with each other for resources through a multihop path leading to the sink. The network is divided into small groups of sensors, called subecosystems. Each subecosystem involves all nodes that send traffic to a particular one-hop-away node (parent node). We argue that the proposed mapping between natural species and WSNs is partial because it does not capture all the characteristics and behaviors of both systems in addition to being designed for congestion control problems only.

Furthermore, because of several WSN restrictions, including limited processing, storage, and context-awareness capabilities, many software agent-based approaches have been proposed to equip sensor nodes with the necessary autonomy and intelligence mechanisms for decision making, self-organization, and resource management processes. In this context, Malik and Shakshuki [19] proposed an approach where mobile agents are used to perform some of the required processing load instead of simply transferring the data to the sink. In this approach, each agent has to carry a code to a source node and bring back aggregated data to the sink, which reduces the communication cost. Garcia et al. [20] proposed to reduce the WSN energy consumption by using data aggregation algorithms whereby agents act as dynamic clustering points in the network. In addition to saving energy, agents can allow a more efficient use of sensor nodes' memories in addition to supporting code distribution among sensors [21]. In terms of conceptualization, sensor nodes have been modeled as software agents to achieve various objectives, such as data sampling [22], improving task assignment [10], and making data routing more efficient [23].

3. Natural Ecosystem Metaphor for Wireless Sensor Networks

To fully exploit the natural ecosystem metaphor for WSNs, in this section we describe the characteristics of both systems and discuss where they match and where WSNs have shortcomings.

3.1. Characteristics of Ecosystems and Wireless Sensor Networks. An ecosystem is a very complex entity that exhibits complex behaviors resulting from the mutual interactions between many components with common, individual, and/or antagonistic goals and their environment. Ecosystems are dynamic and may be defined using a wide range of scales of observation. They include large quantities of matter, energy, and information flowing within and between components, in a way that is not yet completely understood [24]. These flows depend on the ecosystem structure and could be controlled by different parties including top predators' feeding behavior (top-down control), primary producers (bottom-up control), some numerically abundant species (wasp-waist control), or a combination of some or all of these [24].

The functioning of an ecosystem stems from the organization of its species' populations which have their own dynamics in terms of abundance, survival, growth, production, reproductive, and other strategies. The ecosystems' structure, species composition, and functioning may change sometimes in uncontrolled and unpredictable ways [24]. Changes may consequently create uncertainty as to the future states and behavior of the system leading to potential risks for the ecosystem itself and its environment [24].

Wireless sensor networks are collections of spatially distributed nodes that commonly cooperate in order to achieve goals which are beyond their individual capabilities. These nodes may operate unattended in remote harsh areas wherein human interventions are often impossible. Due to a variety of causes including lack of support, spatiotemporal events, animals, and energy depletion of sensors, the topology of the network dynamically changes. Some sensor nodes may lose several of their neighbors and find themselves at the boundaries of physical, logical, malicious, and semantic holes [25], whereas others may be overloaded with data traffic due to the absence of alternative communication pathways.

To optimize the use of the limited resources and lengthen the lifetime of WSNs, several approaches [26, 27] have been proposed in recent years. Some of these approaches have provided the network with the capability to self-organize by creating clusters that may shrink or grow as sensors wake up, sleep, and/or move. The changes on every cluster are commonly controlled by a cluster head which is a sensor node generally selected for its extended capabilities, its residual energy, and/or its degree of connectivity.

3.2. Sensors as Living Organisms. In order to fully exploit the ecosystem metaphor, it is important to compare the low-level entities in natural ecosystems and WSNs, namely, living organisms and sensors. On the one hand, living organisms have 7 main characteristics [28]: (1) nutrition (provides the resources required to fulfill all the other functions of the organism); (2) excretion (set of chemical reactions to remove toxic materials, waste products, and substances in excess of requirements from the organism); (3) respiration (releases the energy from the nutrients); (4) sensitivity (ability to detect or sense changes in the environment and to respond); (5) reproduction (process that generates new organisms of the same species); (6) growth (concerns the increase in size and number of the living organisms); and (7) movement (action by which an organism changes its position). On the other hand, sensors are commonly deployed in closed or open spaces. They are capable of sensing some parameters of interest within their environments, processing and storing data, and communicating with neighboring peers within their communication ranges. In this communication, sensors can support each other (e.g., to heal voids or track intruders), compete (e.g., obtain the necessary resources for their own tasks), or show an antagonistic behavior (e.g., spy nearby peers or jam their communications). A sensor may also move (if equipped with appropriate actuators) to join or leave a subgroup of sensors. This is the case, for example, when a sensor may relocate to prevent any potential physical damage due to new environmental conditions such as fire or heavy rain. During such activity, the sensor may use its limited onboard memory to store new data and experience. It may also demonstrate a certain level of cognition by learning from its previous experiences [16].

Given the characteristics discussed above of both sensors and living organisms, we argue that the capabilities of sensors do not fully equate to those of living organisms. There is indeed a need to extend these capabilities with more flexibility, autonomy, intelligence, and context awareness as shown in Table 1.

3.3. Mapping Ecosystems onto Wireless Sensor Networks. Jones et al. [16] and Antoniou and Pitsillides [18] have argued that the WSN could be modeled based on observations of living systems which are likely to provide realistic models for sensor network design. Indeed, rather than adapting conventional techniques of centralized computer control, new techniques dependent on local cooperation among network

Table 1: Mapping living organisms' characteristics to sensors.

Characteristic	Sensor conformity	Sensor limitation
Nutrition	Supported: acquire information from the environment or from peers to fulfill tasks	Hardware and intelligence to get the right information
Excretion	Supported: clean memory from obsolete data	Intelligence to filter data
Respiration	Not supported, not necessary	—
Sensitivity	Supported: sense/detect changes in the environment	Context awareness
Reproduction	Supported: reproduce some capabilities with software components	Autonomy to reproduce
Growth	Supported: grow in terms of capabilities and knowledge	Processing capabilities
Movement	Supported: move when actuators are available	Intelligence to make the right move

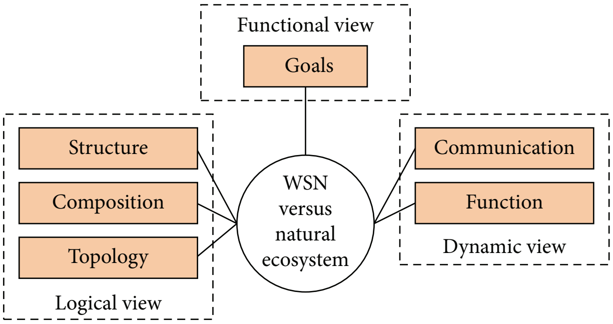


Figure 1: Mapping criteria between WSNs and natural ecosystems.

nodes will lead to self-sustaining communities of machines with emergent behavior that autonomously operate and adapt to changes in the environment [18]. According to this vision, Jones and colleagues [16] have perceived massively deployed motes as organisms which interact, learn, and make local decisions to achieve globally meaningful effects within their community. We also share this vision and propose, in Table 2, a mapping between WSN and ecosystems. Our mapping is basically done by emphasizing the three basic elements of an ecosystem which are [28] structure (represents a high level view of the ecosystem and refers to all of the living and nonliving physical components that make up that ecosystem), composition (refers to the variety of living entities found within an ecosystem as well as their types/roles), and function (reflects the dynamic behavior of the ecosystem and refers to the natural ecological processes of the ecosystem). Furthermore, we emphasize additional features by comparing the topologies resulting from the organization of the entities found in ecosystems and WSNs. We also highlight the aims behind organism organizations as well as the communication between the different components of the system. Based on our mapping (as shown in Figure 1), we argue that in terms of logical view (components, organizations, and their relationships) and functional view (aims), ecosystems and WSN match quite well. However, WSN presents several shortcomings for the dynamic view (behavior). This may be explained by the limited capabilities of sensors that do not usually allow for complex, efficient behavior of WSNs.

4. ABAMA: New Ecosystem-Oriented Architecture for Wireless Sensor Networks

In order to design effective, pervasive WSN services that exploit the benefits of ecosystems' features, we should define adequate methodologies and tools for the dynamic and decentralized control of the system. This control should support a tradeoff between top-down adaptation and a bottom-up one. We should also monitor the overall system by measuring its behaviors in order to make sure that the control is effective [4]. By taking into consideration these requirements, we propose a new ecosystem-oriented architecture for WSNs that we call ABAMA (agent-based architecture for mapping natural ecosystems onto wireless sensor networks). The use of the multiagent system technology in our architecture (Figure 2) is motivated by its proven flexibility, autonomy, and intelligence to solve complex problems within highly dynamic, constrained, and uncertain environments [29]. This technology is particularly used wherever and whenever the WSN fails and needs support to match the natural ecosystem. Following Zambonelli and Viroli's vision [4], ABAMA allows sensors to behave like natural organisms while keeping control of the overall network. ABAMA reflects the fact that sensors could be collaborating, competing, and even antagonistic. Several notations and acronyms on Figure 2 will be explained in the upcoming subsections.

As a WSN may be deployed to provide several services to end-users concurrently, subsets of sensors with each subset including a population of sensors can be created in response to one or more users' queries. The structure and composition of each subset (that we call here service sensor network (SSN)) may be dynamic particularly because users may request the same service from different areas with different quality of service parameters. SSNs may compete with each other to acquire/secure the necessary resources for their tasks. Sensors in each SSN along with the supporting software agents form a small-scale ecosystem that we call EcoSSN (as shown in Figure 2).

We describe in the next sections some important tasks carried out by our proposed architecture, namely, processing users' requests, creating service sensor networks (SSNs), controlling SSNs by agents, monitoring agents and monitoring the whole WSN, and multilevel collaborations. The different

Table 2: Mapping natural ecosystems onto Wireless Sensor Networks.

Ecosystem	WSN	Comments
	Structure (components making up the system)	
Contains living organisms	Contains sensors	Good match
Contains nonliving physical components	Contains only sensors	The space where the WSN is deployed could represent its nonliving physical component
	Composition (variety of active entities within the system)	
Organisms may be producers, consumers, or predators	Sensors may be data collectors (producers), sinks/gateways (consumers), intruders (predators), or relays	In both systems, roles could change depending on the environmental context and human interventions
	Topology (organization of entities that make up the system)	
Structured into populations (also called communities and colonies)	Commonly structured into clusters	Clusters could be predefined by human operators or result from the network self-organization
Populations have dynamic structures	Clusters have dynamic topologies	In both systems, topological changes are driven by internal and external factors
Populations may have different geographic scales	Clusters may have different geographic scales	In both systems, inheritance relationships may exist between populations/clusters
	Goals (aims of the system)	
Depends on the ecosystem; can be survival (nutrition and protection from predators) and/or growth (nutrition and reproduction)	Depends on the WSN but generally collecting, processing, and routing data while optimizing the use of the limited resources (survival)	The goals of WSNs are well known, whereas those of ecosystems are not always understood
	Communication (data flow between entities composing the system)	
Large quantities of matter, energy, and information flow, within and between components	Usually large quantity of data is exchanged between sensors	Sensors may not be able to support high data traffic because of energy restrictions
Flows of energy, matter, and information are in some cases controlled by one or more entities	Data traffic may be controlled by one or more entities, generally cluster heads/gateways	Communications between sensors are very costly and are generally controlled to reach the predefined aims while preserving energy
	Function (behavior of entities composing the system)	
Living organisms may be in a dormant state	Sensors usually have to sleep	Sensors are constrained to sleep to save energy
Organisms interact while exhibiting collaborative, competing, or antagonistic behaviors	Sensors interact while exhibiting collaborative, competing, or antagonistic behaviors	Much more restrictions on sensors' interactions compared to organisms' interactions (due to limited communication ranges and energy)
Populations self-organize to adapt to environmental changes	Clusters can partially self-organize to react to internal and external changes	Self-organization is usually a complex task for sensors because of their limited capabilities, lack of intelligence and autonomy
Populations may have unpredictable and uncontrolled changes/behaviors	Clusters generally have predicable and controlled behaviors unless unexpected events affect sensors	Sensors have limited context awareness
Ecosystem's operation results from the organization of its populations and the behavior of its organisms	WSN's operation results from the organization of its clusters and the behavior of its sensors	In both systems, complex functions result from simple behaviors of active entities which collectively achieve goals beyond their individual capabilities
Organisms have the important characteristic of evolution in terms of number, structure, and behavior	Sensors may be enhanced with mechanisms to learn and evolve thanks to artificial intelligence concepts (e.g., multiagent systems)	Evolution in WSNs takes much less time than in ecosystems but consumes a lot of energy and requires intelligence and autonomy from sensors

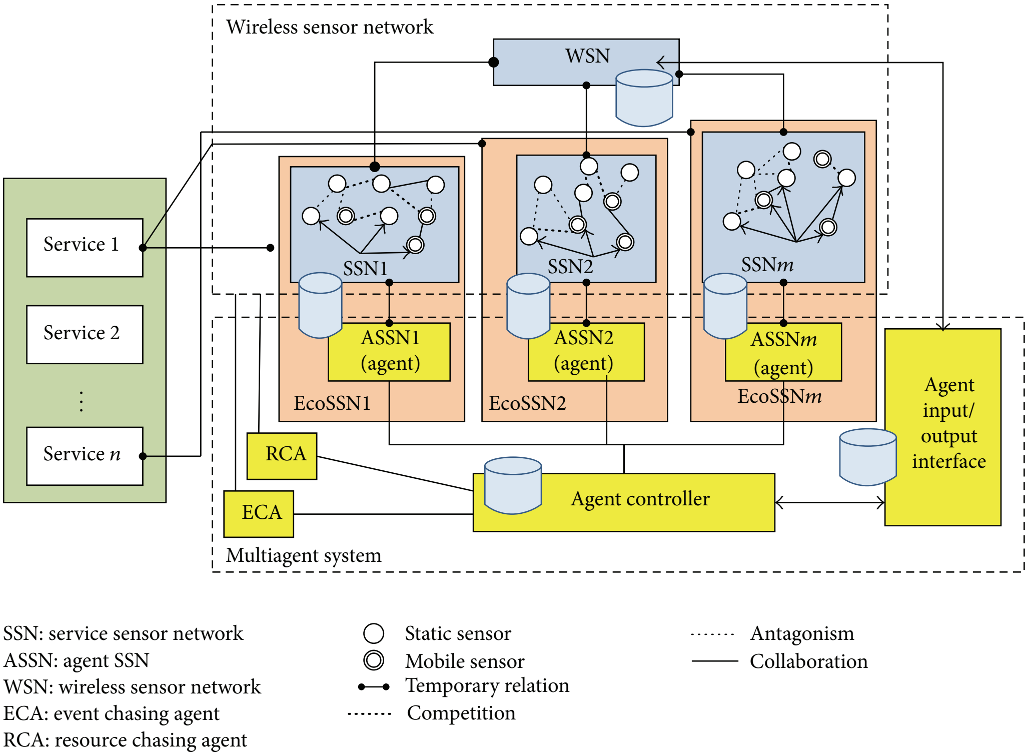


Figure 2: Proposed ABAMA architecture.

types of agents used in ABAMA to achieve these tasks are summarized in Figure 5.

4.1. Processing Users' Requests. ABAMA receives requests for services from end-users or from other WSNs through an agent input/output interface (AIOI). When a query is received, the agent AIOI compares the service requested with the services which were recently delivered by the CPS as well as with those currently in progress. This will allow for assessing the necessary processing within the right spatial areas. The agent AIOI will then make a filtered request to an agent controller (AC as shown in Figure 2) that monitors the different agents, the whole WSN, and the progress of delivering the requested services. The agent AC will request the necessary processing from the WSN based on the data collected from the WSN itself (i.e., from the sensor nodes) as well as the agents which continuously monitor specific aspects of the network such as the network connectivity, energy levels, communication pathways, and progress in supporting current services. The decision of the agent AC is also based on the available spatial data that helps in identifying the distributed areas of interest (AoIs) from which data will be collected.

If the service was already requested by a previous user and is currently being processed, then the agent AC assigns the request to an existing agent service sensor network (ASSN) which is in charge of monitoring the current requested service. Since the users might not necessarily request the same service from the same AoIs, the agent AC also informs the agent ASSN about the additional sensors that will be used to provide the service from the right spatial areas. Once the service is achieved, the agent ASSN notifies the agent AC which in turn replies back to the result to the agent AIOI. If the service was not requested before, then the agent AC passes the request along with the AoIs to the WSN in order to create the SSN necessary to provide the requested service. The agent AC also creates a new agent ASSN that will be in charge of controlling the new SSN. The algorithm of processing users' requests is depicted in Algorithm 1. The process of creating an SSN is described in Section 4.2.

4.2. Creating Service Sensor Networks (SSNs). In order to create a new SSN, the base stations in the WSN broadcast a message JoinService() within the AoIs (explicitly selected by the user or identified by the agent AC). If a given sensor A has already received this message, then it simply acknowledges it. Otherwise, before broadcasting this message, it sets up its role to be either a backbone sensor (BS) or a support sensor (SS). The sensor A is a BS if it is able to collect the requested type of data (e.g., sensor A is a BS if it is a pressure sensor and the requested service is to measure the current atmospheric pressure); otherwise, it is a SS which simply serves as relay to route the data collected by the BSs. Every BS that receives a JoinService() message directly from a base station or from a sequence of SS sensors only (i.e., does not include any other BS sensor) will be elected as a cluster head (CH). Every sensor will then promote the cluster head to which it belongs. All the clusters heads in a given AoI form the AoI board group (AoIBG). The AoIBG is responsible for monitoring the AoIsubnet which comprises all the sensors

```
ProcessRequests(service, AoI)
//check list of current and recently offered services
checkServices(service, QoS); //QoS: Quality of Service
if (service recently offered with the required QoS) {
    reply(AIOI, service, QoS);
}
else{
    //find the necessary areas of interest if not explicitly specified
    //by the user
    //AoI where service is being offered
    //AoIunav = AoI where service is not offered
    [AoIav, AoIunav] = findAoI(service);
    if (service being offered) {
        inform(ASSN, service, AoIav, QoS);
        inform(ASSN, service, AoIunav, QoS);
    }
    else{
        prepare message JoinService(); //Table 3
        inform(WSN, AoI, QoS, JoinService());
        create(ASSN, service, AoI);
        //inform Event Chasing Agent (Figure 5) for support
        inform(ECA, ASSN, service, AoI, Qos);
        //inform Resource Chasing Agent (Figure 5) for support
        inform(RCA, ASSN, service, AoI, Qos);
    }
}
```

Algorithm 1: Algorithm for processing users' requests.

Table 3: Messages exchanged between sensors.

Message	Description
JoinService(id_s, type_s, role, id_{CH})	It is initiated by a cluster head (CH) sensor and then forwarded by other sensors with each sensor indicating its ID, its type, its role (BS, SS), and the ID of the cluster head to which it belongs
RollBack(path, id_s)	It is sent by a sensor s to the neighbor from which it first received the JoinService() message. It contains the paths from s to all the leaf sensors (a leaf is a sensor that did not receive any reply for the JoinService() message that it has sent after a given period of time (timer expired))
ReplyJoinService(id_s, type_s)	It is sent by a sensor s to every sensor from which a JoinService() message was received

used to achieve the requested service in that particular spatial area. To this end, the CHs may, for example, mutually lend resources (basically mobile sensors) or route the data of each other in order to support their mutual operations.

In addition to the JoinService() message, any given sensor A may receive a ReplyJoinService() message or a RollBack() message (as shown in Table 3). In the first case, the sensor A stores the role, the ID, and the type of the sender sensor and marks it as a next hop. In the second case, the sensor A receives the paths leading to all the leaf sensors through the sender sensor. The sensor waiting time for incoming messages is delimited by a timer. Once this timer expires, it aggregates all the paths received from all its next hops and then sends a RollBack() message to its predecessor sensor. Every cluster head will aggregate all the paths received from its next hops. It will also nominate one of the members of its cluster as a cluster subordinate sensor (CSS) (as shown in Figure 3). The CSS sensor, which is selected based on its current energy and the number of hops it is away from the cluster head, will be delegated to carry out some processing (such as broadcasting updates within the cluster) ultimately freeing the cluster head and preserving its energy. Our algorithm for creating part of an SSN in a given area of interest is depicted in Algorithm 2.

4.3. Controlling Service Sensor Networks with Agents. An efficient functioning of an SSN should result from the mutual interactions among its sensors. However, unlike living organisms which can have dense interactions among each other, sensors usually have to run under limited resources and thus cannot have a similar flow of interaction. To prevent any misuse of the network resources, controlling mechanisms within every SSN are necessary. Control in WSN is often assigned to gateways which are specific sensor nodes with extended capabilities. Similar to the entities controlling colonies in ecosystems, gateways have limited awareness of

```
createSSN(AoI)
//AoIs is the set of areas of interest
∀ sensor s within an AoI, do
//the sensor checks the type of the message received
 if (JoinService() message is received) {
   save sender as previous hop
   save sender id, role and type
   if (JoinService() is received for the first time by s) {
     set role s
     send ReplyJoinService()
     broadcast JoinService()
     set timer to predefined value
   }
   else //message was received before
     send ReplyJoinService
 }
 else if(ReplyJoinService() message is received) {
     save sender as next hop
     save sender id, role and type
 }
 else //RollBack message is received
     aggregate paths from s to leaves
Wait for new messages and repeat steps
if no new message (Join/Rep/RollBack) after timer expired
         send RollBack(path, $id_s$) to the CH sensor to which s belongs
```

Algorithm 2: Algorithm for creating part of an SSN in a given area of interest.

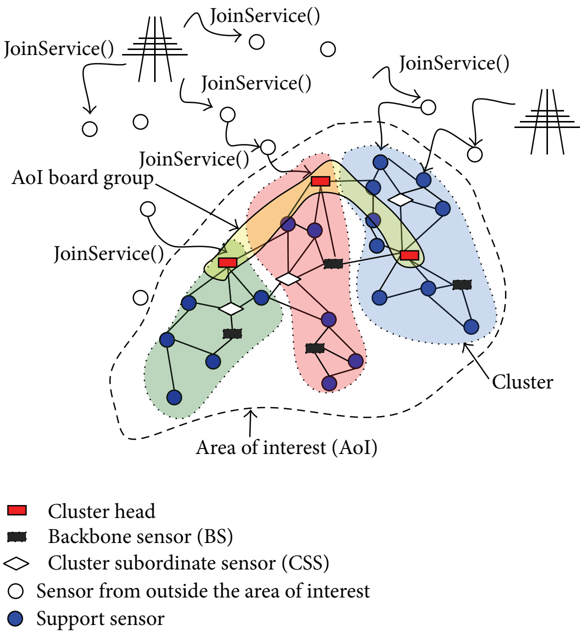


Figure 3: Creating part of the SSN in one area of interest (AoI).

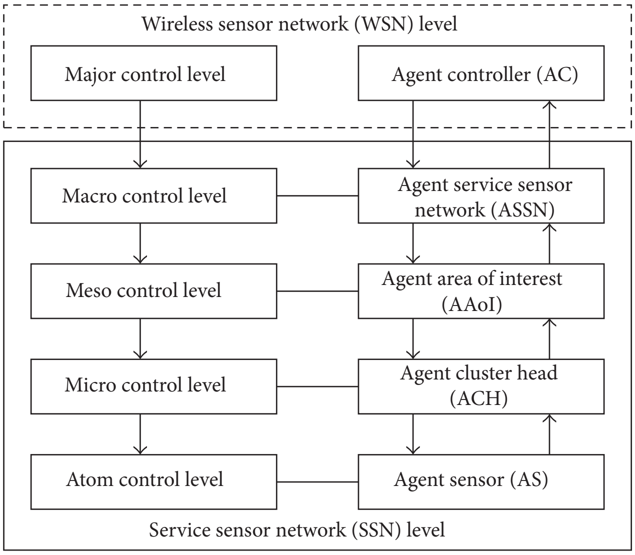


Figure 4: Control levels within a service sensor network.

the geographic space where the WSN is deployed. To overcome this shortcoming (as presented in Table 2), software agents are used to enhance the control of SSNs while adding flexibility to clusters to self-organize and increase their awareness of sensors' communications and mobility. As previously discussed, an SSN is essentially composed of sensors collecting and/or routing data to cluster head sensors located in distributed areas of interest. Based upon the SSN structure, we propose to monitor all SSN operations according to four control levels, namely, atom, macro, meso, and micro (Figure 4). The atom control level is implemented on each individual sensor by means of an agent sensor (AS) that manages the local resources, plans, and carries out local data processing. The micro control level is achieved by a set of

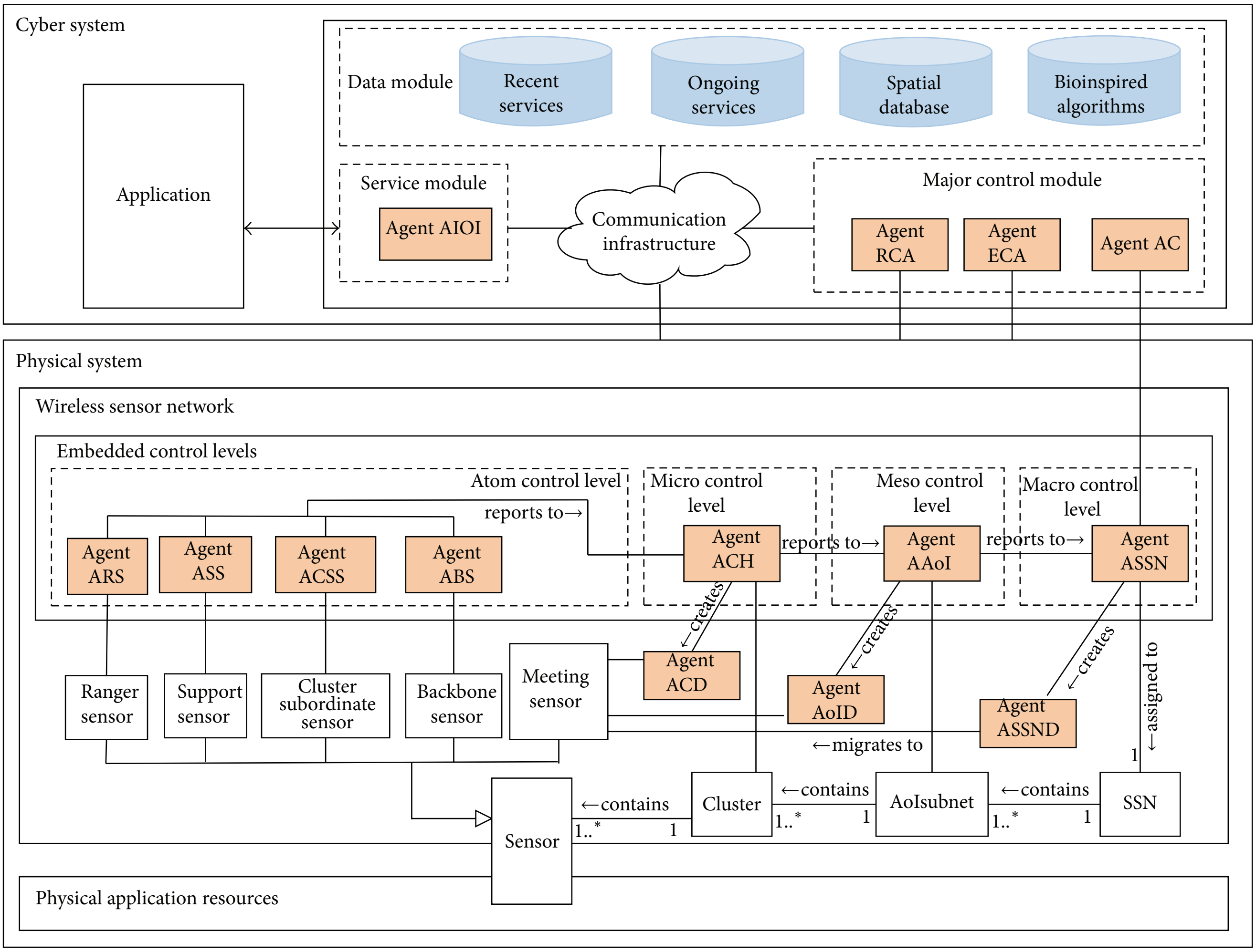


FIGURE 5: A new intelligent CPS architecture. AIOI: agent input/output interface, AC: agent controller, ASSN: agent service sensor network, ABS: agent backbone sensor, ACSS: agent cluster subordinate sensor, ASS: agent support sensor, ACD: Agent Cluster Delegate, AoIsubnet: area of interest subnet, ASSN: agent SSN delegate, RCA: resource chasing agent, ECA: event chasing agent, AoID: AoI Delegate, and ARS: agent ranger sensor.

agents, each of which is assigned to a specific cluster and hosted on the cluster head sensor. Each of these agents, called ACH (agent cluster head), collects information about the current processing activities, energy level, and connectivity of each sensor within its cluster. Some of this information is reported via a mobile agent created by the ACH and named Agent Cluster Delegate (ACD, shown in Figure 5). The mobile agent can migrate within the cluster to collect data on site and/or collaborate with other agents ACD from the same SSN. In this case, these agents agree and meet on a specific sensor node where they interact and make joint decisions on behalf of their agents ACH.

The agents ACH mutually report information on their own resource usage and processing activities. They may assign resources to each other as for example, when mobile sensors from one cluster move to support activities in a neighboring cluster. The coordination and the monitoring of interactions between the different agents ACH is achieved at the meso control level (as shown in Figure 4) by an agent called AAoI (agent area of interest). This agent is responsible for monitoring the progress of operations related to its corresponding service within its corresponding AoI. Thanks to their extended context-awareness, the agents AAoI commonly make recommendations to their agents ACH. Furthermore, several agents AAoI may coexist in the same AoI when more than one service is requested from this same area. In this case, these agents AAoI have to collaborate to ensure fair use of resources based on the priority of services being processed. All agents AAoI belonging to the same service are monitored at the macro control level by an agent ASSN (as described in Section 4.1). In addition to the micro, meso, and macro control levels within each SSN, a final major control level is achieved by the agent AC (as shown in Figure 2) which is in charge of monitoring all the agents ASSN.

The agents ASSN, AAoI, ACH, and AS perform several functions to achieve a better mapping of sensors into living organisms and therefore overcome the shortcomings of WSN (as shown in Table 2). For the sake of illustration, we enumerate below some of the functions which are carried out by an agent ASSN.

Collecting Data. Similar to living organisms which take substances from their surroundings for their vital nutrition, sensors need data from their immediate contexts as inputs for their operations. Because of their limited sensing capabilities, some of the required information is provided to sensors by the appropriate agents at the appropriate control levels. For example, an agent ASSN may exchange data about some events of interest (e.g., spreading wildfire) with other agents ASSN and then recommend better sampling rates to some of its agents AAoI based on processing priorities and available resources. The agent ASSN may also recommend specific data acquisition algorithms (e.g., Dendritic Cell Algorithm like in [30]) depending on particular ongoing circumstances such as sensors' connectivity and spatial events.

Balancing Energy. Based on the collected data, an agent ASSN should be able to predict and delimit the boundaries of potential energy holes [31]. Preventive actions can then be taken to ultimately balance energy over the SSN. For instance, certain nodes may be asked to take over the duties of some sensors with low energy levels while others may be asked to reduce their communication range in order to save energy. Similar behavior can be observed with certain living organisms which intentionally put their metabolisms or some of their functions on standby mode when they lack energy. To achieve the task of load balancing, the agent ASSN may use a bio-inspired load balancing technique such as one based on pheromone signaling (e.g., [32]). The agent ASSN may also use an approach benefiting from the mobility of its agents (ACHD, RSA, etc.) like in [33].

Dealing with Dynamism. Sensors are usually prone to failure thereby causing several unpredictable modifications in the topology of the WSN and leading to route changes. Transferring data through the network could then become a time and energy-consuming process. On their own, sensors are usually unable to establish and maintain communication pathways, particularly because of their limited (local) view of the SSN. In contrast, thanks to its global view of the topology of the SSN (e.g., sleep/wakeup cycles of sensors) and a better awareness about the environment's changing conditions (e.g., heavy rain), the agent ASSN may predict changes in the topology of the network and determines alternative communication pathways on time. The agent ASSN may also instruct some sensors to move in order not to lose connectivity in some areas. The relocation of mobile sensors can be implemented using, for example, a bio-inspired Digital Hormone Mode approach (as in [34]) or using other techniques as described in [34]. To optimize communication costs, an Ant Colony Optimization approach (e.g., [35]) could be used.

Secure Processing. Some sensors may exhibit malicious behaviors causing several problems, including communication jam, data loss, and energy depletion. For example, in a black hole attack, a malicious node attracts all the traffic by advertising that it has the shortest path in the network. Once it receives the packet from other nodes, it drops all the packets causing loss of critical information. Sensors could collectively identify malicious nodes at the expense of communication and energy-consumption overheads. Moreover, if the malicious nodes are moving from one cluster to another, the same processing efforts are more likely to happen to identify and prevent the effect of these nodes. To track malicious nodes efficiently while learning from previous experiences, an agent ASSN can analyze the data communication traffic in its SSN to identify antagonistic sensor nodes. This could be implemented, for example, with a bio-inspired machine learning approach (as in [36]). The agent ASSN can secure the data routing with an ant-based approach (e.g., [37]). The agent can then finally share its experience with other agents ASSN in order to prevent redundant processing (e.g., identifying malicious sensor nodes).

4.4. Monitoring Agents and the Wireless Sensor Network.

The agent controller (AC), which is physically hosted along with agent input/output interface (AIOI) on the cyber side of the cyber physical system (CPS), is in charge of monitoring the agents ASSN as well as the entire WSN. To achieve the first task (i.e., monitoring agents ASSN), different strategies can be implemented. The Agent AC may, for example, request, when needed, information from each agent ASSN on the progress of delivering the assigned service. Alternatively, agents ASSN may periodically notify agent AC about their progress. A hybrid strategy may combine both mechanisms whereby information on the progress of service delivery is reported to the AC whether upon its request or when initiated from any agent ASSN. The aim of the current work is not to discuss the details of agent AC functioning but mainly to explain the motivations for using this agent.

Actually, by monitoring the agents ASSN, the agent AC is also overseeing the overall performance of the WSN. Indeed, after collecting information about the progress of delivering a given service within a certain SSN, the agent AC has to take the best decisions which better fulfill the user requirements and improve the overall performance of the network. These decisions are communicated to the concerned agent ASSN and may consist in one of the following:

(1) Resume/Start Service. Based on its global view of the network, the running services, and pending requests, the agent AC may request a certain agent ASSN to start or resume a specific service.

(2) Stop Service. Agent AC may ask an agent ASSN to cancel the ongoing work. This may happen if agent AC infers that the resources of a given SSN should be assigned to tasks of higher priority. Cancelling the ongoing work may also be decided based on the request of an agent ASSN itself which discovered that the service cannot be achieved with the requested quality given the available resources.

(3) Update Service. As the environment is highly dynamic, agent AC may judge, at run-time, that the service assigned

to a given agent ASSN has to be updated because of the new environmental constraints. For instance, the agent ASSN may be asked to change some sensing parameters within its SSN. It is then the responsibility of the agent ASSN to plan the appropriate steps within the SSN in order to meet the new requirements.

(4) Void. If agent AC is satisfied with the service progression, no notifications are needed.

It is worth mentioning that the communication between agent AC and the different agents ASSN (regarding progression and decisions) is done via short messages and only when needed. These messages are used by agent AC and agents ASSN to monitor the entire WSN, oversee the quality of offered services, and optimize the use of resources. These message exchanges do not generate an extra overhead for the network. To the contrary, they reduce the overall load of the network by avoiding running services which would not have met the desired requirements.

4.5. Multilevel Collaboration. Similar to living organisms in ecosystem populations, sensors in WSN may move from one cluster to another freely or under the control of some monitoring nodes such as gateways. The movement of sensors may enforce the collaboration between clusters. This is the case, for example, when a given cluster is unable to maintain efficient data traffic because of communication holes and receives the support from neighboring clusters that relocate some of their mobile sensors to heal those holes. In a natural ecosystem, collaboration is achieved using several approaches such as Quorum Sensing [38]. In the present work, we allow every agent ACH to identify a list of Ranger Sensors (RSs) which are actually redundant mobile sensors that will be relocated to support other sensors inside or outside its cluster. The selection of RSs is based on their proximity to neighboring clusters and their degree of connectivity within the current cluster. When local RSs fail to address some deficiencies, the ACH may request help from neighboring clusters. The agents ACH of the clusters which have received the request determine the RS sensors to move and then respond back to the agent ACH which made the request. The latter compiles all responses and then confirms its needs to the selected supporting agents ACH. This approach may result in some communication overhead; however, it prevents relocating more RSs than what is actually needed.

To increase the collaboration among clusters, every agent ACH creates a mobile agent called Agent Cluster Delegate (ACD, as described in Figure 5). This agent maintains a list of redundant sensors that could be used as RSs when needed. It keeps an inventory of the available resources, processing capabilities, and routing paths. In case of severe resource deficiency or on a routine basis, the agents ACH may instruct their agents ACD to move to a meeting infrastructure (for example, a specific node in the sensor network) where they will negotiate the mutual needs of their respective clusters and cooperate on joint solutions to current and future problems. Every agent ACD reports the results of the negotiation to its agent ACH, which in turn reports the information to its agent AAoI.

Similar collaboration mechanisms can be implemented between neighboring areas of interest and SSNs where mobile agents (AoID (AoI Delegate) and ASSND (ASSN Delegate)) are created and used to collect specific information, including the number and locations of redundant sensors and the areas facing communication and/or sensing problems. Similar to agents ACD, the agents AoID and ASSND migrate when needed to a meeting infrastructure (i.e., gateway or remote sensor node) to make collaborative decisions and share knowledge and experience learned from previous experiences at low costs. The increased energy consumption resulting from the migration of mobile agents to a meeting infrastructure is irrelevant comparing to the benefits of the approach [39]. The information reported by the different mobile agents to their respective superior agents is then sent to a specific agent called the resource chasing agent (RCA in Figure 5). This agent is constantly updating information on available resources and making recommendations to the agent AC for a better use of the WSN resources and an enhanced optimization of collaborative processing.

5. Toward a New Cyber Physical System Architecture

CPSs are being implemented for a variety of applications, where processing capabilities located on a cyber system are using a communication infrastructure to monitor the physical world. In this configuration, sensors are being seen as an important component through which the access and control of physical, application-related resources can be achieved *in situ* and on time. Sensors can be either part of the application resources or an interface through which instructions are disseminated from the cyber system to the physical system and data are pushed in the reverse flow. In order to improve the limited processing and context-awareness capabilities of sensors, we are using software agents to particularly inject autonomy and flexibility in sensors and the whole sensor network. We believe that agents can also be efficiently used to strengthen the link between the cyber and physical worlds. We thus propose to extend ABAMA to a new CPS architecture (Figure 5) where agents are deployed on the cyber and physical sides of the CPS. Some of our agents (ASSN, AAoI, ACH, ASS, ACSS, ABS, and ARS) are being used to implement an embedded multilevel control over the WSN and the application resources. These agents are hosted on physical sensors and can be supported by some mobile agents (ASSND, AoID, and ACD) in collecting specific data (such as the availability and state of application resources, sensors' connectivity, and routing paths). At any moment, these mobile agents can move to the cyber system wherein extended communication and processing facilities and data are available. They can also be used to make soft copies of some capabilities of a given sensor and share or implement them on other sensors.

In addition to the communication infrastructure, the cyber side of our architecture includes a data module, a service module, and a major control module. The data module contains a spatial database for the identification of areas of interest, the location of sensors, and the application resources and a database containing several bioinspired algorithms that could be used as needed by the agent AC. It also contains a record of the recent services it has provided and the ongoing services to reduce user queries' response times. The service module contains the agent AIOI which is responsible for collecting and responding to the user requests. After checking the ongoing and recent services provided, the agent AIOI sends a request to the agent AC in the major control module. This agent is supported by two special agents: RCA (resource chasing agent) and ECA (event chasing agent). The goals and use of the AC and RCA were explained in the previous sections. The ECA is used to track and detect events of interest within the spatial areas where the WSN and the application resources are located. This agent is important especially since sensors as well as the application resources may be heavily affected by some events (e.g., heavy rain) or are tracking events of interest (e.g., level of water in specific areas). The details on the use of the ECA will be the objective of a future work. The three modules can communicate using the communication infrastructure. This infrastructure is also used by some of the agents on the WSN to exchange data and knowledge and migrate from one sensor to another.

6. Opportunities and Challenges

On its own, a WSN has limited capabilities in terms of self-organization, learning, processing, and detection of malicious nodes. However, with the help of software agents, it is possible to enhance the capabilities of the physical network and ultimately match them with those of a natural ecosystem. Indeed, the agents of ABAMA can implement appropriate bioinspired algorithms, including Ant Colony Optimization (ACO), particle swarm optimization (PSO), intelligent weeds optimization (IWO), and bee colony optimization (BCO) to optimize data traffic and carry out the right processing at the right time. With similar bioinspired approaches, software agents can strengthen the link between the cyber and the physical systems and enable a smooth, efficient, and timely communication.

To take full advantage of the natural metaphor and enjoy the opportunities presented above, there are several issues that must be addressed toward the implementation of our new CPS architecture. In what follows, we outline some of the challenges and opportunities related to the implementation, evaluation, scalability, and seamless integration of our proposed CPS architecture with Big Data and the Internet of Things.

Implementation. Within the specific context of our new CPS architecture, several bioinspired approaches can be used to deliver a given service. This is highlighted by the availability of a database containing bioinspired algorithms on the cyber side. In addition to the complexity of some of these algorithms that may not be suitable to run on sensors with limited resources, it is important to make sure that the appropriate algorithm is used for the right task at the right time. Although multiagent system approaches can bring relatively efficient solutions, this goal remains difficult to achieve because the efficiency of each algorithm may closely depend on the current sensors' requirements, the environmental conditions, and the required QoS (quality of service). Moreover, although sensor agent technology has become sufficiently reliable for operational use in the field [10], deploying agents on sensor nodes suggests additional research efforts that take into account the WSN constraints for the sake of increasing sensors' knowledge, competencies, and context awareness without consuming a lot of the limited energy of the network. Mobile agents allow for the reduction of energy consumption; however, they cannot carry out extended expertise while moving. Collaboration and negotiation algorithms should also be tailored to use as least interaction as possible. Furthermore, because of the variety of behaviors that a sensor may exhibit (e.g., collaborative, competitive, and antagonist) and the change of its capabilities and processing loads, several algorithms based on the theories and models of natural ecosystems have to be further adapted to coexist within the same WSN and eventually on the same sensor node.

Evaluation. Several metrics could be used in order to assess the performance of our CPS design in terms of data routing efficiency, energy consumption, security, reliability, and resource availability. These metrics can be computed using analytical techniques such as those based on continuous time Markov chains (CTMC) and reward functions. As our CPS architecture is targeting several functionalities with varying priorities and objectives, we may have opposing goals for some metrics, for some criteria. A tradeoff must then exist, for example, between data routing efficiency and energy consumption because efficiency is generally linked to communication overhead. Reducing this overhead may in turn lower the expectations in terms of QoS. Moreover, our architecture emphasizes the control of the WSN and the application resources to optimize operations and resource consumption. This control generally results in additional energy-consumption overheads, leading to unsatisfactory results for specific metrics measuring individual criteria. However, we argue that the overall performance of the CPS could be improved. Furthermore, performance of measures depends on specific conditions, including the WSN configuration, current spatiotemporal events, and the number and scope of services required. It is thus important to explore the possibility to adapt some metrics to WSN and CPS related features.

Scalability. We argue that the agents of our CPS architecture can handle the addition of new sensors and network topology changes with a reasonable operational, communication, time, power, and reliability cost overhead. This is because multiagent systems have proven good performance for the development of new computing systems, particularly when these systems are complex, large-scaled, decentralized, open, and heterogeneous [10]. In addition, the mobility of agents can push most of the processing load out of the network to the cyber system where extended capabilities are easy to be made available. This is particularly helpful in dealing with the additional data acquisition, processing, and routing requirements of any additional sensor. Our CPS architecture offers reasonable scalability and it can smoothly integrate and interact with other systems through the communication infrastructure.

Seamless Integration with Internet of Things. WSN technology is a fundamental component of the Internet of Things (IoT). WSN integration is expected to allow sensor nodes to join the Internet dynamically and use it to collaborate and accomplish their tasks. However, we must carefully investigate and analyze this integration as it is likely to open up further issues related to security, quality of services, increasing processing and communication loads, and interoperability. These challenges will need to be addressed in the context of our CPS architecture when the WSN is integrated with the IoT. Nevertheless, this integration would allow for several opportunities, especially resource availability and participatory data acquisition and processing. The first opportunity provides, in general, the CPS with the required resources to implement an efficient communication between the cyber and the physical systems while improving its control over the WSN and the application resources. Typically, CPS can select the appropriate additional resources for its processing without being limited to its own resources. The second opportunity allows sensors to mutually support each other by exchanging available data and available bioinspired algorithms. This exchange of data and algorithms could be extended to include sensors from other CPSs or tier systems (e.g., external WSNs). Sensors may also delegate some of their processing and sensing loads to other sensors from tier systems while respecting predefined trust and security mechanisms.

Seamless Integration with Big Data. Sensors are known to be able to collect real-time, *in situ* data on a variety of events of interest. The larger the WSN is, the bigger the quantity of data collected is. Such data could reveal important knowledge on the events of interest if it is archived and analyzed carefully. On the one hand, with the continuous use of sensors, huge amounts of collected data need to be stored, modeled, and handled with efficient theoretical and practical background of the Big Data field. Big Data technology could be seen in this case as an important enabler that helps toward making the best use of sensor data. On the other hand, sensors may allow enhanced processing of Big Data, particularly when real-time data streams coming from sensors are to be integrated with available data. In the context of our CPS architecture, the cyber system could be extended to host all the collected sensor data as well as techniques for Big Data management. Software agents can be used in order to filter and aggregate data as per the requirements of data processing.

7. Conclusion

Ecosystems and WSN exhibit several similarities, particularly in terms of structure and goals. They are indeed both composed of interactive components (organisms and sensors) which could self-organize, collaborate, and compete to achieve complex functions far more than what they are capable of. We found that a big gap exists between both systems in terms of behaviors due to the limited capabilities of sensors. We argued that the use of a multiagent system approach could bridge the gap between natural organisms and sensors and thus allows for an efficient use of an ecosystem-based metaphor for WSNs. To this end and while identifying the need for sensors to be more flexible, autonomous, and intelligent, we proposed the ABAMA architecture where software agents can ensure a better use of the limited WSN resources by implementing a multilevel control (over the entire network and over individual sensors). These agents are located either on sensor nodes or on a virtual platform where the heavy processing tasks of the WSN are migrated to so that we can increase sensors' context awareness and save energy. We extended our ABAMA architecture to a new CPS architecture where the cyber system incorporates a major control level over the physical system while enforcing the multilevel control embedded on WSN.

Our CPS architecture offers enough flexibility to integrate with new hardware and software capabilities. It is also open to a seamless integration with current hot research and development fields, including IoT and Big Data. In addition to a more in-depth investigation of the roles of the resource chasing agent (RSA) and the event chasing agent (ECA), our future works will focus on selecting the appropriate criteria to adapt some available metrics to the contexts of WSN and CPS.

Disclosure

This paper is an extended version of an invited paper which previously appeared in the Proceedings of the 9th International Conference on Future Networks and Communications in 2014.

Conflict of Interests

The authors, Nafaâ Jabeur, Nabil Sahli, and Sherali Zeadally, declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors thank the anonymous reviewers for their useful comments and suggestions which helped us to improve the quality and presentation of this paper. They would also like to thank the guest editors for providing them with the opportunity to extend the paper into a journal paper for consideration for the special issue.

References

[1] O. Bello and S. Zeadally, "Intelligent device-to-device communication in the internet of things," *IEEE Systems Journal*, 2014.

[2] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: the next computing revolution," in *Proceedings of the 47th Design Automation Conference (DAC '10)*, pp. 731–736, June 2010.

[3] C.-Y. Lin, S. Zeadally, T.-S. Chen, and C.-Y. Chang, "Enabling cyber physical systems with wireless sensor networking technologies," *International Journal of Distributed Sensor Networks*, vol. 2012, Article ID 489794, 21 pages, 2012.

[4] F. Zambonelli and M. Viroli, "A survey on nature-inspired metaphors for pervasive service ecosystems," *International Journal of Pervasive Computing and Communications*, vol. 7, no. 3, pp. 186–204, 2011.

[5] M. N. Huhns and M. P. Singh, "Service-oriented computing: key concepts and principles," *IEEE Internet Computing*, vol. 9, no. 1, pp. 75–81, 2005.

[6] Z. Bai, "Thinking about the ecosystem characteristics based on systems science," *Journal of Central South University of Forestry & Technology*, vol. 12, no. 6, pp. 174–178, 2007.

[7] Digital-ecosystems, 2014, http://www.digital-ecosystems.org/.

[8] D. Bray, "Knowledge ecosystems: a theoretical lens for organizations confronting hyperturbulent environments," in *Proceedings of the International Federation for Information Processing*, Manchester, UK, 2007.

[9] J. F. Moore, "Predators and prey: a new ecology of competition," *Harvard Business Review*, vol. 71, no. 3, pp. 75–86, 1993.

[10] A. Rogers, N. R. Jennings, and D. D. Corkill, "Agent technologies for sensor networks," *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 13–17, 2009.

[11] C.-Y. Chang, J.-P. Sheu, Y.-C. Chen, and S.-W. Chang, "An obstacle-free and power-efficient deployment algorithm for wireless sensor networks," *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, vol. 39, no. 4, pp. 795–806, 2009.

[12] Y. Weng, W. Xiao, and L. Xie, "Total least squares method for robust source localization in sensor networks using TDOA measurements," *International Journal of Distributed Sensor Networks*, vol. 2011, Article ID 172902, 8 pages, 2011.

[13] M. Li, W. Cheng, K. Liu, Y. Liu, X. Li, and X. Liao, "Sweep coverage with mobile sensors," *IEEE Transactions on Mobile Computing*, vol. 10, no. 11, pp. 1534–1545, 2011.

[14] O. D. Incel, A. Ghosh, B. Krishnamachari, and K. Chintalapudi, "Fast data collection in tree-based wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 1, pp. 86–99, 2012.

[15] S. C. Ergen and P. Varaiya, "PEDAMACS: power efficient and delay aware medium access protocol for sensor networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 7, pp. 920–930, 2006.

[16] K. H. Jones, K. N. Lodding, S. Olariu, L. Wilson, and C. Xin, "Communal cooperation in sensor networks for situation management," in *Proceedings of the 9th International Conference on Information Fusion (FUSION '06)*, pp. 1–8, IEEE, Florence, Italy, July 2006.

[17] L. Barolli, T. Yang, G. Mino, A. Durresi, and F. Xhafa, "A simulation system for WSNs as a Digital Eco-System approach considering goodput metric," in *Proceedings of the 4th IEEE International Conference on Digital Ecosystems and Technologies (DEST 2010)*, pp. 375–380, Dubai, United Arab Emirates, April 2010.

[18] P. Antoniou and A. Pitsillides, "Congestion control in autonomous decentralized networks based on the Lotka-Volterra competition model," in *Artificial Neural Networks—ICANN 2009*, vol. 5769 of *Lecture Notes in Computer Science*, pp. 986–996, Springer, Berlin, Germany, 2009.

[19] H. Malik and E. Shakshuki, "Data dissemination in wireless sensor networks using software agents," in *Proceedings of the 21st International Symposium on High Performance Computing Systems and Applications (HPCS '07)*, IEEE Computer Society, May 2007.

[20] M. S. Garcia, D. Carvalho, O. Zlydareva et al., "An agent-based wireless sensor network for water quality data collection," in *Ubiquitous Computing and Ambient Intelligence*, vol. 7656 of *Lecture Notes in Computer Science*, pp. 454–461, Springer, Berlin, Germany, 2012.

[21] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pp. 81–94, ACM, November 2004.

[22] J. Kho, L. Tran-Thanh, A. Rogers, and N. R. Jennings, "Decentralised control of adaptive sampling and routing in wireless visual sensor networks," in *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '09)*, pp. 1208–1209, Honolulu, Hawaii, USA, May 2009.

[23] J. Kho, L. Tran-Thanh, A. Rogers, and N. R. Jennings, "Distributed adaptive sampling, forwarding, and routing algorithms for wireless visual sensor networks," in *Proceedings of the 3rd International Workshop on Agent Technology for Sensor Networks*, pp. 63–70, Budapest, Hungary, May 2009.

[24] P. Cury, L. Shannon, and Y. J. Shin, "The functioning of the marine ecosystems: a fisheries perspective," in *Responsible Fisheries in the Marine Ecosystem*, M. Sinclair and G. Valdimarsson, Eds., pp. 103–123, 2003.

[25] N. Jabeur, N. Sahli, and I. M. Khan, "Survey on sensor holes: a cause-effect-solution perspective," in *Proceedings of the 4th International Conference on Ambient Systems, Networks and Technologies, and the 3rd International Conference on Sustainable Energy Information Technology*, vol. 19, pp. 1074–1080, June 2013.

[26] W. Liang, "Constrained resource optimization in large-scale wireless sensor networks with mobile sinks," *Journal of Communications*, vol. 7, no. 7, pp. 494–499, 2012.

[27] K. Lee and H. Lee, "Energy-efficient self-organized clustering with splitting and merging for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 487846, 11 pages, 2013.

[28] T. Vold and D. A. Buffett, Eds., *Ecological Concepts, Principles and Application to Conservation*, 2008, http://www.biodiversitybc.org/.

[29] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *PRoceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications (INFOCOM '03)*, vol. 3, pp. 1713–1723, IEEE Societies, March-April 2003.

[30] F. Gu, J. Greensmith, and U. Aicklein, "The dendritic cell algorithm for intrusion detection," in *Biologically Inspired Networking and Sensing: Algorithms and Architectures*, pp. 84–102, IGI Global, 2012.

[31] I. M. Khan, N. Jabeur, and S. Zeadally, "Hop-based approach for holes and boundary detection in wireless sensor networks," *IET Wireless Sensor Systems*, vol. 2, no. 4, pp. 328–337, 2012.

[32] I. Caliskanelli, J. Harbin, L. S. Indrusiak, P. Mitchell, F. Polack, and D. Chesmore, "Bioinspired load balancing in large-scale WSNs using pheromone signalling," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 172012, 14 pages, 2013.

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

[34] S. Kataria and A. Jain, "Bio inspired optimal relocation of mobile sink nodes in wireless sensor networks," in *Proceedings of the IEEE International Conference on Emerging Trends in Communication, Control, Signal Processing & Computing Applications (IEEE-C2SPCA '13)*, pp. 1–6, Bangalore, India, October 2013.

[35] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.

[36] H. Rathore and S. Jha, "Bio-inspired machine learning based wireless sensor network security," in *Proceedings of the 5th World Congress on Nature and Biologically Inspired Computing (NaBIC '13)*, pp. 140–146, IEEE, Fargo, ND, USA, August 2013.

[37] N. A. Alrajeh, M. S. Alabed, and M. S. Elwahiby, "Secure ant-based routing protocol for wireless sensor network," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 326295, 9 pages, 2013.

[38] M. B. Miller and B. L. Bassler, "Quorum sensing in bacteria," *Annual Review of Microbiology*, vol. 55, pp. 165–199, 2001.

[39] N. Sahli, N. Jabeur, and M. Badra, "Agent-based framework for sensor-to-sensor personalization," *Journal of Computer and System Sciences*, vol. 81, no. 3, pp. 487–495, 2015.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2015, Article ID 475030, 18 pages
http://dx.doi.org/10.1155/2015/475030

Research Article

Energy Efficient and Safe Weighted Clustering Algorithm for Mobile Wireless Sensor Networks

Amine Dahane,[1] Abdelhamid Loukil,[1] Bouabdellah Kechar,[2] and Nasr-Eddine Berrached[1]

[1] *Intelligent Systems Research Laboratory, University of Sciences and Technology of Oran, Algeria*
[2] *Laboratory of Industrial Computing and Networking, Ahmed Ben Bella Oran University, Algeria*

Correspondence should be addressed to Abdelhamid Loukil; abdelhamid.loukil@univ-usto.dz

Received 10 November 2014; Accepted 15 January 2015

Academic Editor: Zahoor Khan

The main concern of clustering approaches for mobile wireless sensor networks (WSNs) is to prolong the battery life of the individual sensors and the network lifetime. For a successful clustering approach the need of a powerful mechanism to safely elect a cluster head remains a challenging task in many research works that take into account the mobility of the network. The approach based on the computing of the weight of each node in the network is one of the proposed techniques to deal with this problem. In this paper, we propose an energy efficient and safe weighted clustering algorithm (ES-WCA) for mobile WSNs using a combination of five metrics. Among these metrics lies the behavioral level metric which promotes a safe choice of a cluster head in the sense where this last one will never be a malicious node. Moreover, the highlight of our work is summarized in a comprehensive strategy for monitoring the network, in order to detect and remove the malicious nodes. We use simulation study to demonstrate the performance of the proposed algorithm.

1. Introduction

After the success of theoretical research contributions in previous decade, wireless sensor networks (WSNs) have become now a reality [1–3]. Their deployment in many societal, environmental, and industrial applications makes them very useful in practice. These networks consisted of large number of small size nodes which sense ubiquitously some physical phenomenon (temperature, humidity, acceleration, noise, light intensity, wind speed, etc.) and report the collected data to the sink station by using multihop wireless communications. Although the nodes are able to self-organize and collaborate together in order to establish and maintain the network, they are battery powered, limited in terms of processing, storage, and communication capabilities [4]. WSNs are considered in many cases as stationary, but topology changes can happen due to a weak mobility (new nodes join the network and existing nodes experience hardware failure or exhaust their batteries) [5]. In other scenarios, the mobility can occur when nodes are carried by external forces such as wind, water, or air [6] so that the network topology can be affected accordingly and can be changed slowly. This second kind of mobility, known as one form of strong mobility in the literature in the sense where nodes are forced to move physically in the deployment area, has been considered in this paper. Clustering means grouping nodes which are closed to each other and it has been widely studied in ad hoc networks [3, 7–14]. More recently, it has been used in WSNs [14–21] where the purpose in general is to reduce useful energy consumption and routing overhead. Figure 1 illustrates how inside the cluster two kinds of nodes can be found: one node called cluster head (CH) or coordinator (in Figure 1: CH1, CH2, and CH3) which is responsible for coordinating the cluster activities and several ordinary nodes called cluster members (CMs) (in Figure 1: CM1 and CM2) that have direct access only to one CH. An ordinary node which is able to hear two or more CHs is called a gateway (G) (in Figure 1: the gateway G2 can hear CH1, CH2, and CH3, while the gateway G1 can hear CH1 and CH2) instead. So, each communication initiated by a cluster member to a destination inside the cluster must pass by CH. If the destination is outside the cluster, the communication must be forwarded by a gateway. Recent research studies recognize that organizing mobile WSNs, in the sense defined above, into clusters by using a clustering mechanism is a challenging task [9, 19]. This is due to the fact that CHs carry out extra

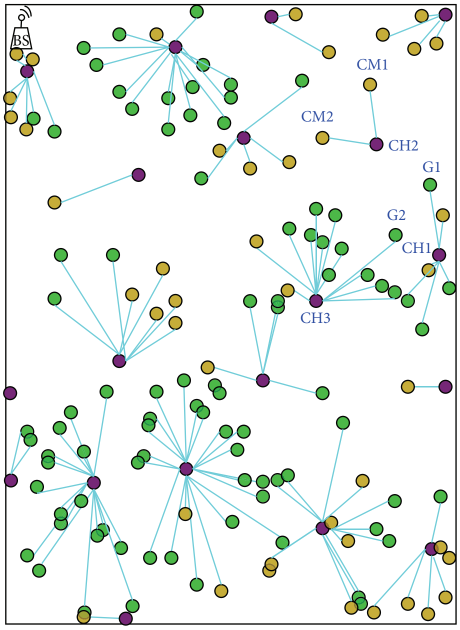


Figure 1: Clustering formation of WSNs composed of 150 sensor nodes deployed in a 570 m × 555 m space area with a radio range = 100 m.

work and consequently consume more energy compared to CMs during the network operations and this will lead to untimely death causing network partition and therefore failure in communication link. For this reason, one of the most frequently encountered problems in this mechanism is to search for the best way to elect CH for each cluster. Indeed, a CH can be selected by computing the quality of nodes. This may depend on several metrics: connectivity degree, mobility, residual energy, and the distance of a node from its neighbors. Significant improvement in performance of this quality can be achieved by combining these metrics [3, 9, 10, 12, 19, 21].

In this paper, we propose an energy efficient and safe weighted clustering algorithm for mobile WSNs using a combination of the above metrics to which we added a behavioral level metric. The latter metric is decisive and allows the proposed clustering algorithm to avoid any malicious node in the neighborhood to become a CH, even if the remaining metrics are in its favor. The election of CHs is carried out using weights of neighboring nodes which are computed based on selected metrics. So this strategy ensures the election of legitimate CHs with high weights. The preliminary results obtained through simulation study demonstrate the effectiveness of our algorithm in terms of the number of equilibrate clusters and the number of reaffiliations, compared to WCA (Weighted Clustering Algorithm) [3], DWCA (Distributed Weighted Clustering Algorithm) [9], and SDCA (Secure Distributed Clustering Algorithm) [21]. These results also reveal that our approach is suitable if we plan to use it in network layer reactive routing protocols instead of proactive ones once the clustering mechanism is launched.

We can enumerate the contributions of our paper as follows:

(i) maintaining stable clustering structure and offering a better performance in terms of the number of reaffiliations using the proposed algorithm ES-WCA (Energy Efficient and Safe Weighted Clustering Algorithm);

(ii) detecting common routing problems and attacks in clustered WSNs, based on behavior level;

(iii) showing clearly the interest of the routing protocols in energy saving and therefore maximizing the lifetime of the global network.

The remaining part of this paper is organized as follows. Section 2 briefly surveys the related works on clustering algorithms proposed for ad hoc networks and in particular those developed for WSNs. In Section 3, we emphasize on the security problems in WSNs. Section 4 introduces and explains the selected metrics for the proposed approach of clustering. More details on the proposed algorithm are given in Section 5. Section 6 presents the simulation tool developed for evaluation. Simulation results are provided to show the effectiveness of the proposed algorithm. Section 7 concludes the paper and outline directions of future works.

2. Related Works

In this section, we outline some approaches of clustering used in ad hoc networks and WSNs. Research studies on clustering in ad hoc networks involve surveyed works on clustering algorithms [11, 22] and cluster head election algorithms [10, 16]. Abbasi and Younis [17] presented taxonomy and classification of typical clustering schemes, then summarized different clustering algorithms for WSNs based on classification of variable convergence time protocols and constant convergence time algorithms, and highlighted their objectives, features, complexity, and so forth. A single metric based on clustering as in paper [23] shows that the node with the least stability value is elected as CH among its neighbors. However, the choice of CH which has a lower energy level could quickly become a bottleneck of its cluster. Er and Seah [8] designed and implemented a dynamic energy efficient clustering algorithm (DEECA) for mobile ad hoc networks (MANETs) that increases the network lifetime. The proposed model elects first the nodes that have a higher energy and less mobility as cluster heads, then periodically monitors the cluster head's energy, and locally alters the clusters to reduce the energy consumption of the suffering cluster heads. The algorithm defines a yellow threshold to achieve some sort of local load balancing and a red threshold to trigger local reclustering in the network. However, the cluster formation in this scheme is not based on connectivity so the formed clusters are not well connected; consequently, this increases the reaffiliation rate and maximizes reclustering situations. Jain and Reddy [24] have proposed a novel method of modeling wireless sensor network using fuzzy graph and energy efficient fuzzy based k-Hop clustering algorithm which takes into account the dynamic nature of network, volatile aspects of radio links, and physical layer uncertainty. They have defined a new centrality metric, namely, fuzzy k-hop centrality. The proposed centrality metric considers residual energy of individual nodes, link quality, hop distance between the prospective cluster head, and respective member nodes to ensure better cluster head selection and cluster quality, which results in better scalability, balancing of energy consumption of nodes, and longer network lifetime. Other proposals use a strategy based on computed weight in order to elect CHs [3, 9, 10, 12]. The main strategy of these algorithms is based mainly on adding more metrics such as the connectivity degree, mobility, residual energy, and the distance of a node from its neighbors, corresponding to some performance in the process of electing CHs. Although the algorithms which use this strategy allow us to ensure the election of better CHs based only on their high computed weight from the considered metrics, they unfortunately do not ensure that the elected CHs are legitimated nodes, that is, whether the election process of CHs is safe or not. Safa et al. [13] propose a novel cluster based trust-aware routing protocol (CBTRP) for MANETs to protect forwarded packets from intermediary malicious nodes. The proposed protocol ensures the passage of packets through trusted routes only by making nodes monitor the behavior of each other and update their trust tables accordingly. However, in CBTRP all nodes monitor the network which lead to rapid drainage of node energy and therefore minimize the lifetime of the network. In Section 3, we show that WSNs are vulnerable to various types of attacks [24, 25]. In the last decade, several studies proposed solutions to solve attacks in WSNs by using cryptography, such as SPINS [26]. However, cryptography alone is not enough to prevent node compromise attacks and novel misbehavior in WSNs [27]. Little effort has been made to include the security aspect in the clustering mechanism. Yu et al. [4, 28] try to secure the clustering mechanism against wormhole attack in ad hoc networks (communication between CHs). However, this is done after forming clusters, not during the election procedure of CHs. Liu [4, 29] surveyed the clustering algorithms available for WSNs but that was done from the perspective of data routing. Hai et al. [30] propose a lightweight intrusion detection framework integrated for clustered sensor networks by using an overhearing mechanism to reduce the sending alert packets. Elhdhili et al. [31] propose a reputation based clustering algorithm (RECA) that aims to elect trustworthy, stable, and high energy cluster heads but during the election procedure, not after forming clusters. Benahmed et al. [21] used clustering mechanism based on weighted computing as an efficient solution to detect misbehavior nodes during distributed monitoring process in WSNs. However, they focused only on the misbehavior of malicious nodes and not on the nature of attacks, the formed clusters are not homogeneous, the proposed algorithm SDCA is not coupled with a routing protocols, and it does not give much importance to energy consumption.

In this paper, the proposed approach focuses around strategy of distributed resolution which enables us to generate a reduced number of balanced and homogeneous clusters in order to minimize the energy consumption of the entire network and prolong sensors lifetime. The introduction of a new metric (the behavioral level metric) promotes a safe choice of a cluster head in the sense where this last one will never be a malicious node. Thus, the highlight of our work is summarized in a comprehensive strategy for monitoring the network, in order to detect and remove the malicious nodes.

The fact that WSNs include limited energy resources (batteries) due mainly to their small size, our algorithm shows clearly the interest of the routing protocols in energy saving which therefore maximize the lifetime of the network by coupling it with AODV and then DSDV protocols [5, 32, 33].

3. Security in WSNs

The typical attacks in WSNs include Sinkhole attack, Black Hole attack, Hello Flood attack, and Node Outage which are the most common network layer attacks on WSNs [30, 34–38]. These selected attacks have been summarized in the following sections.

3.1. Sinkhole. Sinkhole attack is one of the most devastating ones: it is very hard to protect against [36, 39]. In a Sinkhole attack, the adversary's goal is to redirect nearly all the traffic from a particular area through a compromised node, creating a metaphorical sinkhole with the adversary at the center so that all traffic in the surrounding will be absorbed by the malicious node. Because nodes, on or near the path followed by transmitted packets, have many opportunities to tamper with application data. Sinkhole attacks can enable many other attacks such as selective forwarding, for example [40].

3.2. Black Hole. In this attack, malicious nodes advertise very short paths (sometimes zero-cost paths) to every other node, forming routing black holes within the network [41]. As their advertisement propagates, the network routes more traffic in their direction. In addition to disrupting traffic delivery, this causes intense resource contention around the malicious node as neighbors compete for limited bandwidth. These neighbors may themselves be exhausted prematurely, causing a hole or partition in the network.

3.3. Hello Flood Attack. Many routing protocols use "Hello" broadcast messages to announce themselves to their neighbor nodes. The nodes that receive this message assume that source nodes are within range and add source nodes to their neighbor list. The Hello Flood attacks can be caused by a node which broadcasts a Hello packet with very high power, so that a large number of nodes even far away in the network choose it as the parent node [14]. These nodes are then convinced that the attacker node is their neighbor, so that all the nodes will respond to the Hello message and waste their energy.

3.4. Node Outage. If a node acts as an intermediary, an aggregation point, or a cluster head, what happens if the node stops working? Protocols used by the WSNs must be robust enough to mitigate the effects of failures by providing alternate routes [34].

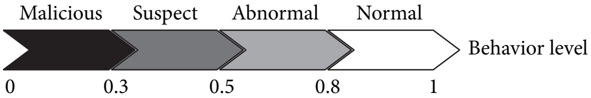


Figure 2: Behavior level $\mathrm{BL}_i \in [0, 1]$.

4. Metrics for CHs Election

This section introduces the different metrics used for cluster head election by focusing on behavior level metric.

4.1. The Behavior Level of Node n_i (BL_i). The behavioral level of a node n_i is a key metric in our contribution. Initially, each node is assigned an equal static behavior level "$\mathrm{BL}_i = 1$." However, this level can be decreased by the anomaly detection algorithm if a node misbehaves. For computing the behavior level of each node, nodes with a behavior level less than threshold behavior will not be accepted as CH candidates even if they have the other interesting characteristics such as high energy, high degree of connectivity, or low mobility. Nevertheless, abnormal nodes and suspect nodes may belong to a cluster as CM but never as CH. So, we define the behavior level of each sensor node n_i, noted BL_i, in any neighborhood of the network as illustrated in Figure 2.

BL_i is classified by the following mapping function:

$$\mathrm{Mp}\left(\mathrm{BL}_i\right) = \left\{\begin{array}{c} \text{Normal node:} \quad 0.8 \leq \mathrm{BL}_i \leq 1 \\ \text{Abnormal node:} \quad 0.5 \leq \mathrm{BL}_i < 0.8 \\ \text{Suspect node:} \quad 0.3 \leq \mathrm{BL}_i < 0.5 \\ \text{Malicious node:} \quad 0 \leq \mathrm{BL}_i < 0.3 \end{array}\right\}. \quad (1)$$

The values in formula (1) are chosen on the basis of several reputed models of WSNs adopted by numerous researchers like Shaikh et al. [42] and Lehsaini et al. [43]. The monitor node watches its neighbors to know what each one of them does with the messages it receives from another neighbor. If the neighbor of the monitor changes, delays, replicates, or simply keeps a message that should be retransmitted, the monitor counts a failure. Number of failures have influence on the behavior of neighbors; for instance, if the monitor counts one failure from a neighbor, its behavior will decrease by 0.1 units. This allows the monitor (cluster head) to differentiate malicious nodes (that make much failure) of a legitimate node (that make fewer failure) in case there are collisions.

4.2. The Mobility of Node n_i (M_i). Our objective is to have stable clusters. So, we have to elect nodes with low relative mobility as CHs. To characterize the instantaneous nodal mobility, we use a simple heuristic mechanism as presented in the formula below (2) [4, 44]:

$$M_i = \frac{1}{T}\sum_{t=1}^{T}\sqrt{\left(x_t - x_{t-1}\right)^2 + \left(y_t - y_{t-1}\right)^2}, \quad (2)$$

where (x_t, y_t) and (x_{t-1}, y_{t-1}) are the coordinates of node n_i at time t and $t-1$, respectively. T is the period for which this parameter is estimated.

In our previous paper [4], the considered mobility has a particular sense by the fact that a mobile node does not move from one location to another in the space area of its own will, but in our case, it moves through the forces acting from the outside. These external forces can act from time to time sporadically. In contrary, the malicious node can use its own ability to move freely in the space area. The behavior of the malicious node by moving frequently inside the same cluster (case illustrated by Figure 3) or from a cluster to another is a normal behavior to not attract attention of the neighborhood and therefore be detected. The idea of our algorithm to ensure the choice of a legitimate CH is to never elect a node that moves frequently and even it has the best performance metrics, but this malicious node does nothing just mobility, so in this paper our algorithm (ES-WCA) detects the internal misbehavior of nodes during distributed monitoring process in WSNs by the follow-up of the messages exchanged between the nodes. ES-WCA is based on the ideas proposed by da Silva et al. [45] used in his efficient and accurate IDS in detecting different kinds of simulated attacks.

4.3. The Distance between Node n_i and Its Neighbors (D_i). This is likely to reduce node detachments and enhance cluster stability. For each node i, we compute the sum of the distance D_i with all its neighbors j. This distance is given, as in [3, 4, 9], by

$$D_i = \sum_{j \in N(i)} \left\{\mathrm{dist}\left(i, j\right)\right\}. \quad (3)$$

4.4. The Residual Energy of Node n_i (Er_i). The residual energy of a node n_i, after transmitting a message of k bits at distance d from the receiver, is calculated according to [4, 16]

$$\mathrm{Er}_i = E - \left(E_{Tx}\left(k, d\right) + E_{Rx\,\mathrm{elec}}\left(k\right)\right), \quad (4)$$

where

(i) E: the node's current energy;

(ii) $E_{Tx}(k, d) = k \cdot E_{\mathrm{elec}} + k \cdot E_{\mathrm{amp}} \cdot d^2$: it refers to the required energy to send a message, where E_{amp} is the required amplifier energy;

(iii) $E_{Rx\,\mathrm{elec}}(k) = kE_{\mathrm{elec}}$: it refers to the energy consumed while receiving a message.

4.5. The Degree of Connectivity of Node n_i at Time t (C_i). It represents the number of n_i's neighbors given by (5) according to [4]

$$C_i = \left|N\left(i\right)\right|, \quad (5)$$

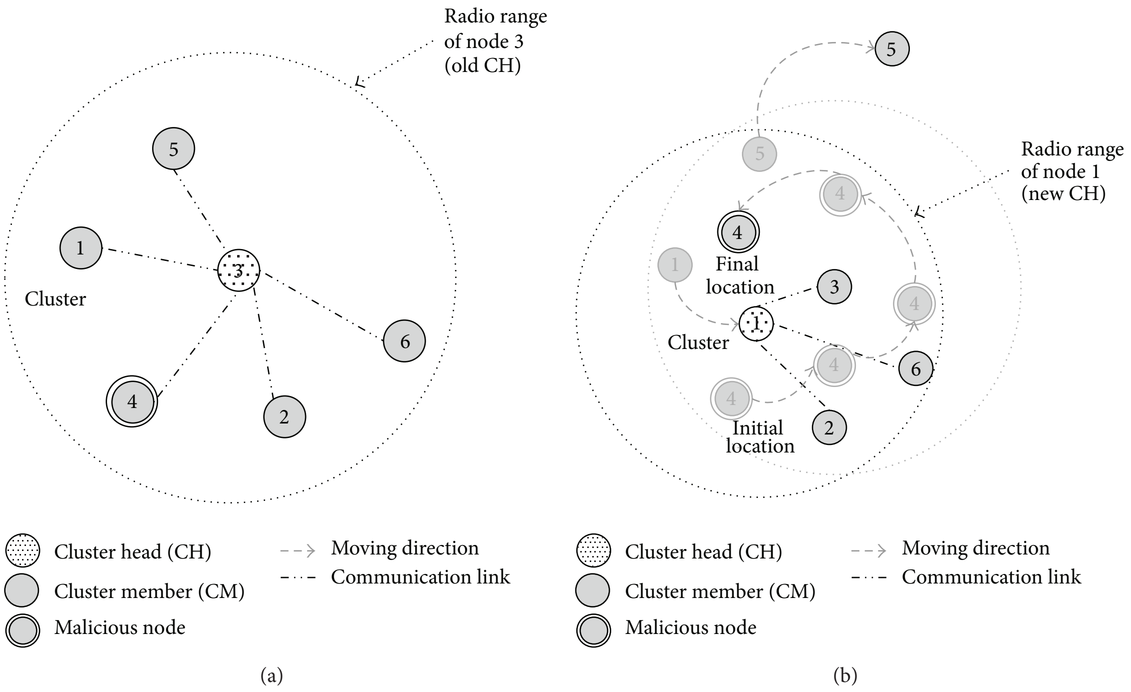


FIGURE 3: (a) Clustering mechanism in mobile WSNs before moving nodes and (b) after moving nodes 1, 5, and 4.

where

(i) $N(i) = \{n_i/\text{dist}(i, j) < tx_{\text{range}} \text{ with } i \neq j\}$,

(ii) dist(i, j): outdistance separating two nodes n_i and n_j,

(iii) tx_{range}: the transmission radius.

For each node, we must calculate its weight P_i, according to the equation:

$$P_i = w_1 * \text{BL}_i + w_2 * \text{Er}_i + w_3 * M_i + w_4 * C_i + w_5 * D_i, \quad (6)$$

where w_1, w_2, w_3, w_4, and w_5 are the coefficients corresponding to the system criteria, so that

$$w_1 + w_2 + w_3 + w_4 + w_5 = 1. \quad (7)$$

We propose to generate homogeneous clusters whose size lies between two thresholds: $Thresh_{Upper}$ and $Thresh_{Lower}$.

These thresholds are arbitrarily selected or they depend on the topology of the network. Thus, if their values depend on the topology of the network, they are calculated as follows according to [43]:

(i) u: the node that has the maximum number of neighbors with one jump:

$$\delta_{12}(u) = \min(\delta_{12}(u_i) : u_i \in U), \quad (8)$$

(ii) v: the node that has the minimal number of neighbors with one jump:

$$\delta_{12}(v) = \min(\delta_{12}(v_i) : v_i \in U). \quad (9)$$

We denote AVG by the average cardinal of the groups with one jump of all the nodes of the network:

$$\text{AVG} = \frac{\sum_{i=1}^{n} \delta_{12}(u_i)}{N}, \quad (10)$$

where N represents the number of nodes in the network. Thus, the two thresholds are calculated as follows:

$$Thresh_{Upper} = \frac{1}{2}(\delta_{12}(u) + \text{AVG}),$$
$$Thresh_{Lower} = \frac{1}{2}(\delta_{12}(v) + \text{AVG}). \quad (11)$$

The calculated weight for each sensor is based on the above parameters ($\text{BL}_i, M_i, D_i, \text{Er}_i$, and C_i). The values of coefficients w_i should be chosen depending on the basis of the importance of each metric in considered WSNs applications. For instance, it is possible to assign a greater value to the metric BL_i compared to other metrics if we promote the safety aspect in the clustering mechanism. It is also possible to assign the same value for each coefficient w_i in the case where all metrics are considered as having the same importance. An approach based on these weight types will enable us to build a self-organizing algorithm which forms a small number of homogenous clusters in size and radius by geographically grouping close nodes. The resulting weighted clustering algorithm reduces energy consumption and guaranties the choice of legitimate CHs.

5. Weighted Clustering Algorithm (ES-WCA)

In this section, we first present some assumptions of the proposed algorithm: Energy Efficient and Safe Weighted

Clustering algorithm (ES-WCA). Then we present in detail an extended version of ES-WCA [4] followed by an illustrative example.

5.1. Assumptions. This paper is based on the following assumptions.

(i) The network formed by the nodes and the links can be represented by an undirected graph $G = (U, E)$, where U represents the set of nodes ni and E represents the set of links ei [3, 4].

(ii) All sensor nodes are deployed randomly in a 2-dimension (2D) plane.

(iii) A node interacts with its one-hop neighbors directly and with other nodes via intermediate nodes using multihop packet forwarding based on a routing protocol such as ad hoc on demand distance vector [5, 32] or DSDV [33].

(iv) The radio coverage of sensor nodes is a circular region centered on this node with radius R.

(v) Two sensor nodes cannot be deployed in exactly the same position x, y in a 2D space.

(vi) All sensor nodes are identical or homogeneous. For example, they have the same radio coverage radius R.

(vii) Each node can determine its position at any moment in a 2D space.

(viii) Each cluster is monitored by only one CH.

(ix) Each CM communicates directly with its CH for the transmission of security metrics.

(x) A CH communicates directly with the base station for the transmission of security information and possible alerts.

5.2. Proposed Algorithm. The ES-WCA algorithm that we present below is based on the ideas proposed by Chatterjee et al. [3], Lehsaini et al. [43], and Zabian et al. [10], with modifications made for our application. This algorithm runs in three phases: the setup phase, the reaffiliation phase, and the monitoring phase. ES-WCA combines each of the above system parameters with certain weighting factors chosen according to the system needs.

5.2.1. The Setup Phase. ES-WCA uses three types of messages in the setup phase (Algorithm 1). The message CHmsg is sent in the network by the sensor node which has the greatest weigh. The second one is the JOINmsg message which is sent by the neighbor of CH if it wants to join this cluster. Finally, a CH must send a response ACCEPTmsg message as shown in Figure 4.

The node which has the greatest weight begins the procedure by broadcasting CH message to their 1-hop neighbors to confirm its role as a leader of the cluster. The neighbors confirm their role as being member nodes by broadcasting a JOINmsg message. In the case when nodes have the same maximum weight, the CH is chosen by using the best parameters ordered by their importance. If all parameters of nodes are equal, the choice is random.

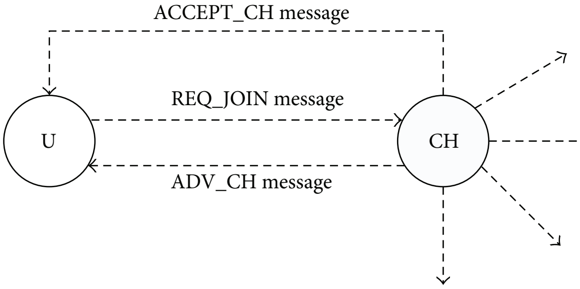


Figure 4: Procedure of affiliation of node "U" to a cluster.

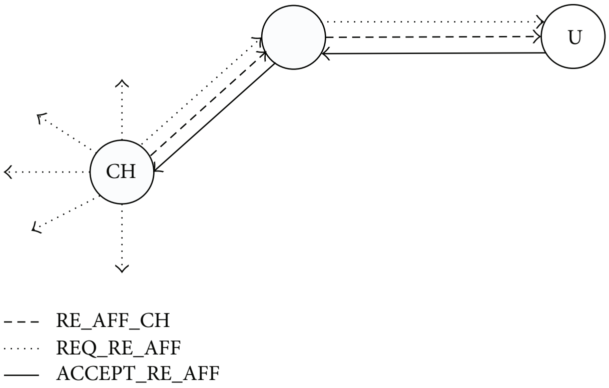


Figure 5: Procedure of reaffiliation of node "U" to a cluster.

Table 1: Values of the various criteria of normal nodes.

Ids	BL_i	Er_i	C_i	D_i	M_i	P_i
1	0.86	3842.12	3	1.15	1.20	769.632
4	0.81	4832.54	5	2.30	0.30	968.133
5	0.88	4053.25	3	1.30	0.55	811.829
6	0.85	4620.43	0	0.00	0.20	924.361
8	0.81	4816.80	4	1.05	1.40	964.753
10	0.95	3650.25	2	0.55	0.10	730.805
11	0.91	4819.60	1	0.70	2.20	964.753

5.2.2. The Reaffiliation Phase. ES-WCA uses four types of messages in the reaffiliation phase (Algorithm 2). The message RE_AFF_CH is sent in the network by the CH whose cluster size is less than $Thresh_{Upper}$. The second one is the REQ_RE_AFF message which is sent by the neighbors of CH if it wants to join this cluster. Finally a CH must send a response ACCEPT_RE_AFF message or DROP_AFF message as illustrated by Figure 5. Accordingly, in this phase we propose to reaffiliate the sensor nodes belonging to clusters that have not attained the cluster size $Thresh_{Lower}$ to those that did not achieve $Thresh_{Upper}$ in order to reduce the number of clusters formed and organize them so as to obtain homogeneous and balanced clusters.

With the help of 3 figures (Figures 6, 7, and 8), our algorithm setup phase is demonstrated. Table 1 shows the quantitative results of the different criteria applied on the normal nodes ($BL_i \geq 0.8$). Table 2 shows the weights P_i of neighbors for each node which has behavior BL_i higher

```
Begin
(1) Assign values to the coefficients $w_1, w_2, w_3, w_4, w_5$;
(2) For any node $n_i \in G$ make:
(3)     $n_i$ forms a list of its neighbors $N(i)$ through the Message who_are_neighbors;
(4)     $N(i) = \emptyset$;
(5)     Calculate its weight $P_i$:
(6)         $P_i = w_1 * \mathrm{BL}_i + w_2 * \mathrm{Er}_i + w_3 * M_i + w_4 * C_i + w_5 * D_i$;
(7)     Initialize Time Cluster and the state vector of all
        nodes $n_i \in G$ Vector_State (Id, CH, Weight, List_Neighbors, Size, Nature)
(8)     CH = 0, Size = 0;
(9)     Nature = "None";
(10)    Repeat
(11)        Any node $n_i \in G$ Broadcasts a message "Hello";
(12)        If $N(i) <> \emptyset$ Then
(13)            Choose $v \in N(i)$;
(14)            $Weight(v) = \max\{weight(w) / \ w \in N(i)\}$;
(15) the node that have the same maximum weight, the CH is
the node that has the best criteria ordered by their
importance ($\mathrm{BL}_i$, $\mathrm{Er}_i$, $C_i$, $D_i$ and $M_i$) if all criteria of
nodes are equal, the choice is random.
(15)        Else $n_i$ is a CH of itself.
            EndIf
(16)        Update the state vector of the elected CH;
(17)        CH = ID;
(18)        Size = 1;
(19)        Nature = CH;
(20)        Send the message "CHmsg" by CH to its neighbors $N(\mathrm{CH})$;
(21)        $J$ = Count ($N(\mathrm{CH})$);
(22)        For $I = 1$ to $J$ Do
(23)          If ($n_i \in N(\mathrm{CH})$ receives the message &&$n_i \rightarrow \mathrm{CH} = 0$)
(24)           Then $n_i$ sends a message "JOINmsg" to CH
(25)          If (CH $\rightarrow$ Size < $Thresh_{Upper}$)
(26)           Then CH sends a message "ACCEPTmsg" to Node $n_i$;
(27)                CH executes the accession process;
(28)                CH $\rightarrow$ Size = CH $\rightarrow$ Size + 1;
(29)                $n_i$ executes the accession process;
(30)                $n_i \rightarrow$ CH = CH $\rightarrow$ Id;
(31)           Else go to (10);
              EndIf
          EndIf
      End For
(32) Until expired (TimeCluster);
End.
```

Algorithm 1: Algorithm setup phase.

Table 2: Weights of neighbors.

Ids	1	4	5	6	8	10	11
1	769.632				**964.753**		**964.753**
4		**968.133**	811.829		964.753		
5		**968.133**	811.829			730.805	
6				**924.361**			
8	769.632				**964.753**		
10		**968.133**	811.829			730.805	
11	769.632						**964.753**

```
Inputs: $Thresh_{Upper}$, $Thresh_{Lower}$;
Outputs: set of clusters
Begin
(1)  For num_cl = 1 to Count (Cluster) Do
(2)      If (Size (Cluster [num_cl]) < $Thresh_{Upper}$)
             Then
(3)          CH sends a message “RE_AFF_CH” to its neighbors
             ($N$(CH));
(4)          $J$ = Count ($N$(CH));
         EndIf
(5)  For $I$ = 1 to $J$ Do
(6)      If ($n_i \in N$(CH) receives the message)
             && ($n_i$ ∈ (Size (Cluster [num_cl]) < $Thresh_{Lower}$)
             Then
(7)          $n_i$ sends a Select message “REQ_RE_AFF” to the CH;
(8)      If (Size (Cluster [num_cl]) < $Thresh_{Upper}$)
             Then
(9)          CH sends a message “ACCEPT_RE_AFF” to $n_i$;
(10)         CH updates its state vector;
(11)         CH → CH → Size = Size + 1;
(12)         $n_i$ updates its state vector;
(13)         $n_i$ → CH → ID = ID;
(14)         Else CH sends a “FIN_AFF” message to $n_i$;
(15)         Go to (2);
         EndIF
(16)   Else $n_i$ sends a “DROP_AFF” message to CH;
       EndIf
     End For
   End For
End.
```

Algorithm 2: Algorithm reaffiliation phase.

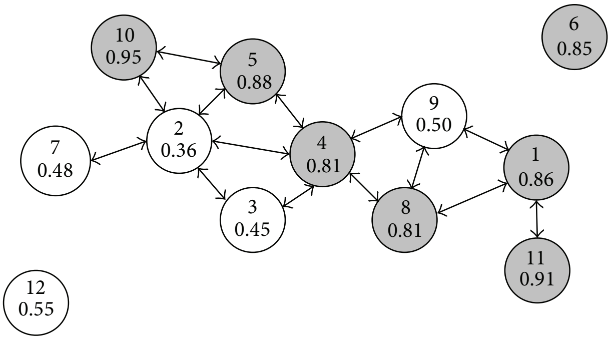


Figure 6: Topology of the network.

than 0.8. The circles in Figure 6 represent the nodes, their identity Ids are at the top, and their levels of behavior are at the bottom. According to Table 2, node 1 could be attached to either CH11 or CH8 (since they have the same weight). However, the behavior level of node 11 is greater than that of node 8 ($BL_{11} > BL_8$). So, node 1 will be attached to CH11. For the other nodes, we have various conditions. Node 4 declares itself as a CH. Node 5 will be attached to CH4. Node 6 declares itself as a CH, because it is an isolated node. Node 8 will be attached to CH4. Node 10 is connected to CH5, but node 5 is attached to CH4. Thus, node 10 declares itself as a CH. Node 11 declares itself as a CH. These results give us the representation shown in Figure 7. Node 2 is connected to CH4 and CH10. Node 2 will be attached to CH4, because CH4 has the maximum weight (968.133). Node 3 is connected to CH4, which implies that node 3 will be attached to CH4. Node 7 is not connected to any CH, so node 7 declares itself as CH. Node 9 is connected to CH4, and then node 9 will be attached to CH4. Node 12 is not connected to any CH, which implies that node 12 declares itself as a CH. These results give us the representation shown in Figure 8. We propose to generate homogeneous clusters whose size lies between two thresholds: $Thresh_{Upper} = 9$ and $Thresh_{Lower} = 6$. For that, we suggest to reaffiliate the sensor nodes belonging to the clusters that have not attained the cluster size $Thresh_{Lower}$ to those that did not reach $Thresh_{Upper}$. Node 4 has the highest weight and his size is less than $Thresh_{Upper}$. Nodes 1, 7, and 10 are neighbors of node 4 with 2 hops and belong to the clusters that have not attained the cluster size $Thresh_{Lower}$, so these nodes get merged to cluster 2. Clusters 1, 3, and 4 will be homogeneous with cluster 1 when the network becomes densely.

At the end of this example, we obtain a network of four clusters (as shown in Figure 9).

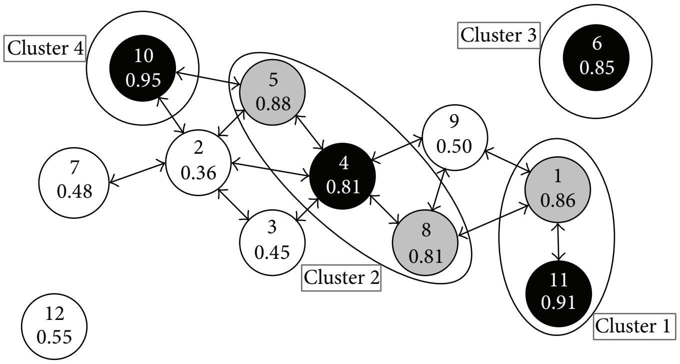


Figure 7: Identification of clusters node.

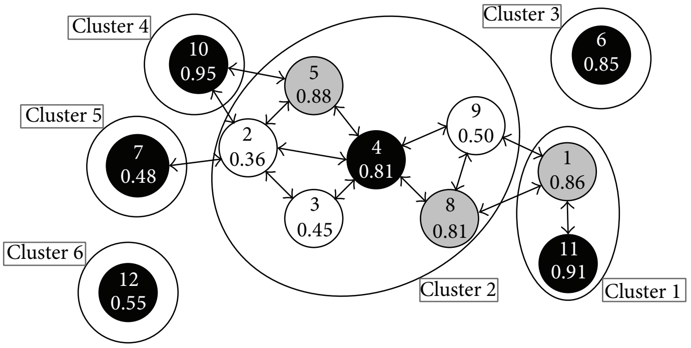


Figure 8: The final identification of clusters.

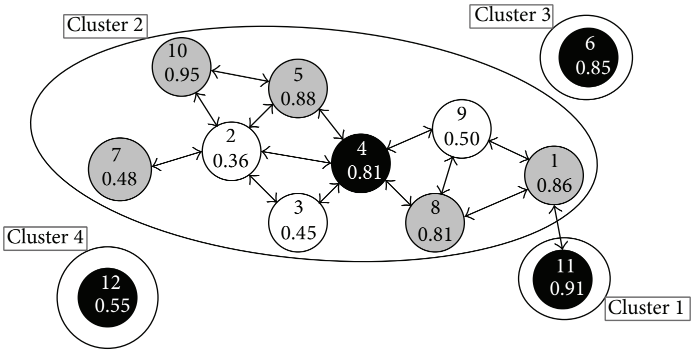


Figure 9: Final cluster structure (reaffiliation phase).

There are five situations that require the maintenance of clusters:

(i) battery depletion of a node,

(ii) behavior level of a node less than or equal 0.3,

(iii) adding, moving, or deleting a node.

In all of these cases, if a node n_i is CH then the setup phase will be repeated.

5.2.3. The Monitoring Phase. Monitoring in WSNs can be both local and global. The local monitoring can be with respect to a node and the global monitoring can be with respect to the network, but in sensor networks, for detecting some types of errors and security anomalies, the local monitoring would be insufficient [46]. For this reason, we adopt in this paper a hybrid approach that is global monitoring based on distributed local monitoring. The general architecture of our approach is illustrated in Figure 10. Our simulator, baptized "Mercury," detects the internal misbehavior nodes during distributed monitoring process in WSNs by the follow-up of the messages exchanged between the nodes. We assume that the network has already a mechanism of prevention to avoid the external attacks. By using a set of rules, all the received messages are analyzed. A similar approach is used by da Silva et al. [45] and Benahmed et al. [21].

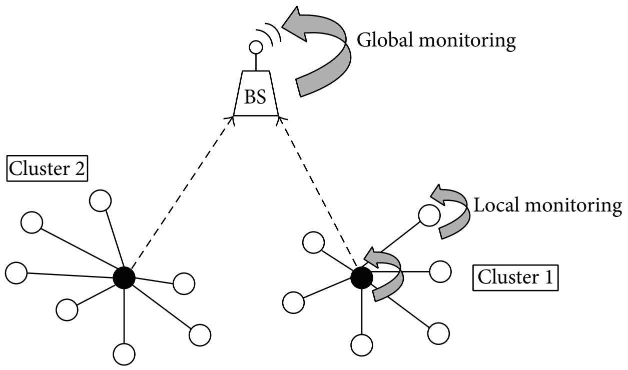


Figure 10: Monitoring phase architecture.

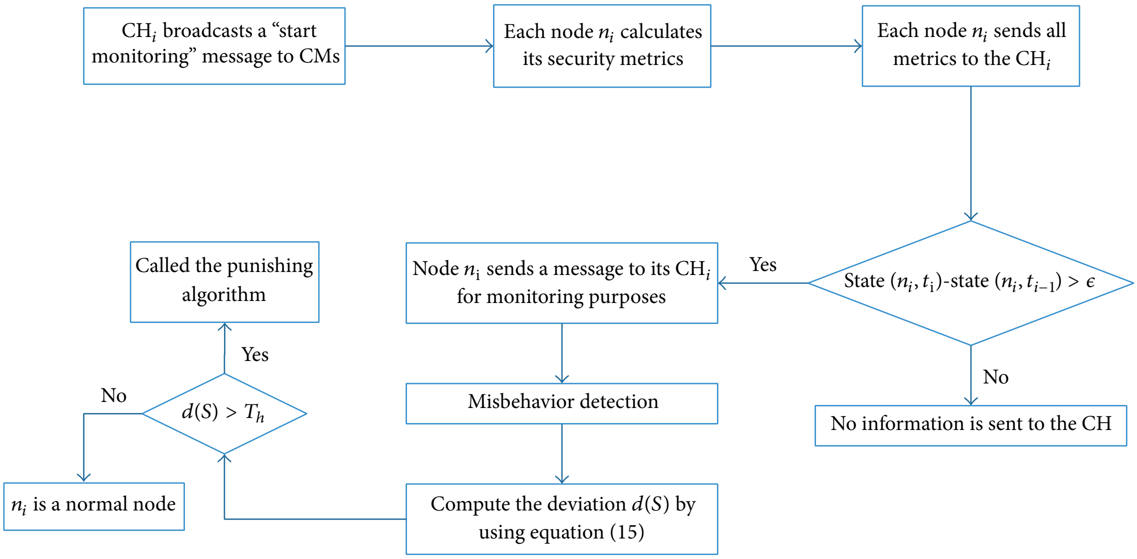

CH$_i$ broadcasts a "start monitoring" message to CMs
Each node n_i calculates its security metrics
Each node n_i sends all metrics to the CH$_i$
State (n_i, t_i)-state $(n_i, t_{i-1}) > \epsilon$
Yes
No
No information is sent to the CH
Node n_i sends a message to its CH$_i$ for monitoring purposes
Misbehavior detection
Compute the deviation $d(S)$ by using equation (15)
$d(S) > T_h$
Yes
No
Called the punishing algorithm
n_i is a normal node

Figure 11: Monitoring phase.

Algorithm 4 (monitoring phase algorithm). The monitoring process involves a series of steps as illustrated by the flowchart in (Figure 11).

Step 1 (this step runs in each CH$_i$). Each CH$_i$ becomes the monitor node of its cluster members and broadcasts a "*Start Monitoring*" message with its Id$_i$ to its entire cluster CMs.

Step 2 (calculation of security metrics performed by each member n_i of the cluster i). Each node n_i $(i <> j)$ receives the message "*Start Monitoring*" and calculates its security metrics as follows.

(i) Number of packets sent by n_i at time interval is $\Delta t = [t_0, t]$: $Nbp\_Send(ni, \Delta t)$.

(ii) Number of packets received by node n_i at time interval is $\Delta t = [t_0, t_0]$: $Nbp\_Received(n_i, \Delta t)$.

(iii) Delay between the arrivals of two consecutive packets is

$$Delay\_BP\,(n_i, t) = Arrival\_PT_i - Arrival\_PT_{i-1}. \quad (12)$$

(iv) Energy consumption: the energy consumed by the node j in receiving and sending packets is measured using the following equation:

$$Ec\,(n_i, \Delta t) = \mathrm{Er}\,(n_i, t_0) - \mathrm{Er}\,(n_i, t_1)\,, \quad (13)$$

where Δt is the time interval $[t_0, t_1]$; $\mathrm{Er}(n_i, t_0)$ is the residual energy of node n_i at time t_0; $\mathrm{Er}(n_i, t_1)$ is the residual energy of node n_i at time t_1 and $Ec(n_i, \Delta t)$ is the energy consumption of node n_i at time interval Δt.

Step 3 (sending all metrics to the CH). After each consumption of the security metrics, the state of a node n_i at time t is denoted by state (n_i, t_i). For storage volume economy, each node keeps only the latest calculation state.

(i) In the initial deployment, each CM in cluster "i" sends some states (state(n_i, t_i)) to the CH$_i$ for making a normal behavior model of node n_i by using a learning mechanism.

(ii) Each state contains the following information:

$$\left(Id, Nbp_{Send(ni,\Delta t)}, Nbp_{Received(n_i,\Delta t)}, Delay_{BP(n_i,t)}, Ec\left(n_i, \Delta t\right)\right). \tag{14}$$

(iii) If (state (n_i, t_i) − state $(n_i, t_{i-1}) > \epsilon$)

then node n_i sends a message (ϵ a given threshold):

$Msg = (Id, Nbp_{Send(ni,\Delta t)}, Nbp_{Received(n_i,\Delta t)}, Delay_{BP(n_i,t)}, Ec(n_i, \Delta t))$ to its CH_i for monitoring purposes.

Otherwise, no information is sent to the CH.

(iv) The message received by CH_i will be stored in a table Tmet for future analysis.

(v) If a sensor node n_i does not respond during this monitoring period, it will be considered as misbehaving.

(vi) The behavior level of sensor node n_i is computed using the following equation:

$$\mathrm{BL}_i = \mathrm{BL}_i - \text{rate}. \tag{15}$$

The "rate" is fixed on the basis of the nature of the application. For example, if it is fault tolerant or not. In our case, we took rate = 0.1.

Step 4 (misbehavior detection, which is performed by CH_i).

(i) For each node n_i in the cluster "i," the state in time slot "t" is expressed by the three-dimensional vector:

$$S = \left(S_{t1}, S_{t2}, S_{t3}\right), \tag{16}$$

where

(a) S_{t1} is the number of packets dropped by n_i, defined as follows:

$$S_{t1} = \sum Ps_{Received\ by\ n_i} - \sum Ps_{Sent\ by\ n_i} - \sum Ps_{destined\ by\ n_i}, \tag{17}$$

with

$$\sum Ps_{Received\ by\ n_i} = \sum Ps_{Sent by\ n_i} + \sum Ps_{destined by\ n_i} + \sum Ps_{lost\ by\ n_i}. \tag{18}$$

For a normal node, $S_{t1} \approx 0$.

(b) S_{t2} is the delay between the arrival of two consecutive packets:

$$S_{t2} = Delay\_BP\left(n_i, t\right). \tag{19}$$

(c) S_{t3} is the energy consumption:

$$S_{t3} = Ec\left(n_i, \Delta t\right). \tag{20}$$

Here, $t \in [t_0, \mathrm{t}] = \Delta t$.

(ii) In our case, the first interval is used for the training data set of n time slots. We calculate the mean vector $\overline{S}$ of S by using

$$\overline{S} = \frac{\sum_{t=t_0}^{t_{n-1}} S_t}{n}. \tag{21}$$

(iii) After modeling a normal behavior model for each sensor node, the behaviors of all nodes are sent to the base station for further analysis. We then compute the deviation $d(S)$ by using

$$d\left(S\right) = \left|S - \overline{S}\right|. \tag{22}$$

(iv) When the deviation $d(S)$ is larger than threshold T_h (which means that it is out of the range of normal behavior), it will be judged as a misbehaving node. In this case, the level of behavior is $\mathrm{BL}_i \approx 0$. This is called the punishing algorithm:

$$\begin{aligned} &d\left(S\right) > T_h\text{: } n_i \text{ is an abnormal node} \\ &d\left(S\right) \leq T_h\text{: } n_i \text{ Is a normal node.} \end{aligned} \tag{23}$$

The punishing algorithm is presented in Algorithm 3.

6. Simulation Results

This section presents the implementation of the proposed approach using the Borland C++ language and the analysis of the obtained results.

6.1. The Simulator "Mercury". We try to complete the theoretical study by implementing our own wireless sensor network simulator "Mercury." On the other hand, a bit of simulators for WSNs such as TOSSIM [47] and Power-TOSSIM [48] are irrelevant with our goal and purpose and in order to avoid many complications we established our own mercury simulator. It is established on an object-oriented design and a distributed approach such as self-organization mechanism which is distributed at the level of each sensor; it provides a set of interfaces for configuring a simulation and for choosing the type of event scheduler used to drive the simulation. A simulation script generally begins by creating an instance of this class and calling various methods to create nodes and topologies and configure other aspects of the simulation. Mercury uses two routing protocols for delivering data from sensor nodes to the Sink station: a reactive protocol AODV (ad hoc on demand distance vector) [5] and a proactive protocol DSDV (destination sequenced distance vector) [6]. To determine and evaluate the results of the execution of algorithms that are introduced previously; the number of sensors to deploy must be inferior or equal to 1000. There are two types of sensor nodes deployment on the sensor field: random and manual. Mercury offers users the ability to select a sensor type from 5 types of existing sensor, each of them has its proper characteristics (radius, energy, etc.).

```
Begin
(1)  $I := 0$;
(2)  $I := I + 1$;
(3)  If (($I$ = Rate) && ($BL_i <= 0.1$))
        // Rate: parameter of maximum number of faults
           defined by the administrator
        $BL_i = BL_i -$ Rate;
(4)     // Classification of the node according to its $BL_i$
(5)     $Mp(BL_i) = \begin{cases} \text{Normal node: } 0.8 \le BL_i \le 1 \\ \text{Abnormal node: } 0.5 \le BL_i < 0.8 \\ \text{Suspect node: } 0.3 \le BL_i < 0.5 \\ \text{Malicious node: } 0 \le BL_i < 0.3 \end{cases}$
(6)     If ($BL_i \le 0.3$) Then
(7)      If ($n_i$ is CM) Then
(8)         Suppression of the node of the list of the members;
(9)       Addition of the node to the blacklist;
         EndIf
(10)     If ($n_i$ is CH) Then            // CH: Cluster Head
(11)        Addition of the node to the blacklist;
(12)        Set up Phase;
         EndIf
        EndIf
     EndIf
End.
```

Algorithm 3: Punishing algorithm.

Unity of the energy used is as Nanojoules: (1 Joule = 10^9 NJ). Mobility has influence on energy and the behavior of sensors; for instance, if the sensor moves one meter away from its original location, its energy will diminish by 100,000 NJ and its behavior will also decrease by 0.001 units. This allows users to differentiate a malicious node (that moves frequently) of a legitimate node (that can changes position with reasonable distances). Since sensors nodes move due to the forces acting from the outside, no power consumption for mobility must be taken into consideration in all simulations that we have carried for evaluation [4].

6.2. Discussion and Results. To evaluate our ES-WCA algorithm, we have performed extensive simulation experiments. This section provides our experimental results and discussions. In all the experiments, N varies between 10 and 1000 sensor nodes. The transmission range (R) varies between 10 and 175 meters (m) and the used energy (E) is equal to 50000 NJ. The sensor nodes are randomly distributed in a "570 m × 555 m" space area by the following function:

```
for (int n = 0; n < node_tobe_deployed; n + +).

        {
        X_ = rand()  %  image_Field_Of_Collecting
        → width;
        Y_ = rand()  %  image_Field_Of_Collecting
        → Height;
        }
```

The performance of the proposed ES-WCA algorithm is measured by calculating (i) the number of clusters, (ii) number of reaffiliations, (iii) choice of ES-WCA with AODV or DSDV, and (iiii) detection of misbehavior nodes and the nature of attacks during the distributed monitoring process.

In our experiments, the values of weighting factors used in the weight calculation are as follows: $w_1 = 0.3$, $w_2 = 0.2$, $w_3 = 0.2$, $w_4 = 0.2$, and $w_5 = 0.1$. It is noted that these values are arbitrary at this time and for this reason they should be adjusted according to the system requirements. To evaluate the performance of the proposed ES-WCA algorithm by comparing it with alternative solutions, we studied the effect of the density of the networks (number of sensor nodes in a given area) and the transmission range on the average number of formed clusters. Then we compared it with a WCA proposed in [3], DWCA proposed in [9], and SDCA proposed in [21].

Figure 12 illustrates the variation of the average number of clusters with respect to the transmission range. The results are shown for N which varies between 200 and 1000. We found that there is opposite relationship between clusters and transmission range. This is on the grounds that a cluster head with a considerable transmission range will cover a large area.

Figure 13 depicts the average number of clusters that are formed with respect to the total number of nodes in the network. The communication range used in this experiment is 200 m. From Figure 13, it is seen that ES-WCA consistently provides about 61.91% less clusters than DWCA and about 38.46% than SDCA, when there were 100 nodes in the network. When the node number is equal to 20 nodes,

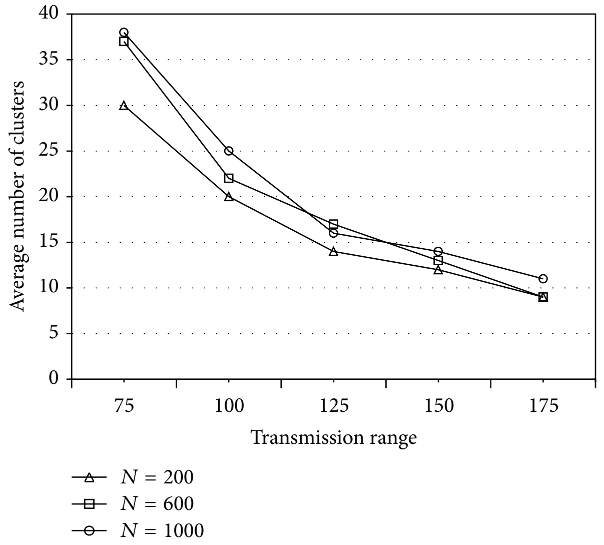


Figure 12: Average number of clusters versus transmission range (R).

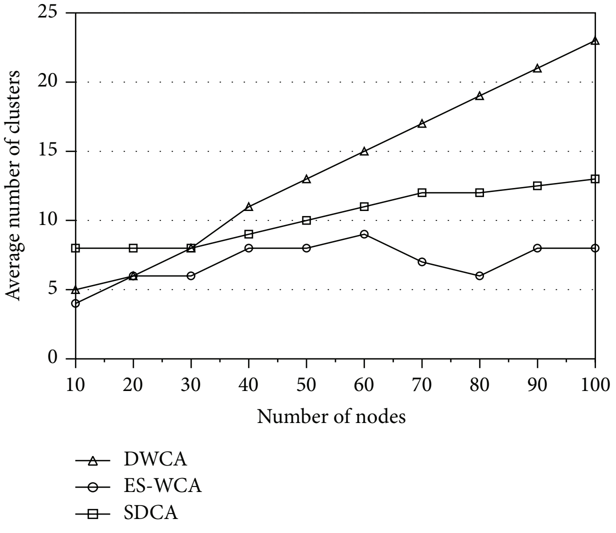


Figure 13: Average number of clusters versus number nodes (N) for ES-WCA, DWCA, and SDCA.

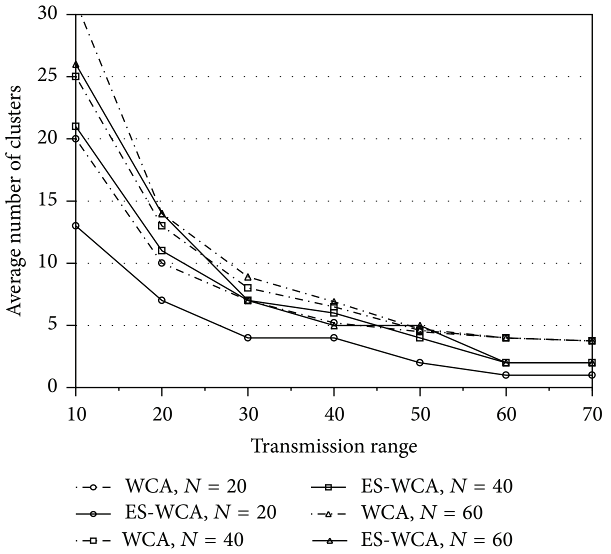


Figure 14: Average number of clusters versus transmission range ES-WCA and WCA.

the performance of ES-WCA is similar to DWCA in terms of number of clusters; however, if the node density had increased, ES-WCA would have produced constantly less clusters than SDCA and DWCA, respectively, regardless of the node number. Because of the use of a random deployment, the result of ES-WCA is unstable between 60 and 90. So, the increase in the number of clusters depends on the increase of the distance between the nodes. As a result, our algorithm gave better performance in terms of the number of clusters when the node density in the network is high, and this is due to the fact that ES-WCA generates a reduced number of balanced and homogeneous clusters, whose size lies between two thresholds: $Thresh_{Upper}$ and $Thresh_{Lower}$ (reaffiliation phase) in order to minimize the energy consumption of the entire network and prolong sensors lifetime.

Figure 14 shows the variation of the average number of clusters with respect to the transmission range. The results are shown for varying N. We notice an inverse relationship, and the average number of clusters decreases with the increase in the transmission range. As shown in Figure 14, the proposed algorithm produced 16% to 35% fewer clusters than WCA [3] when the transmission range of nodes was 10 m. When the node density increased, ES-WCA constantly produced less clusters than WCA regardless of the node number. With 70 nodes in the network, the proposed algorithm produced about 47% to 73% less clusters than WCA. The results show that our algorithm gave a better performance in terms of the number of clusters when the node density and transmission range in the network are high.

Figure 15 interprets the average number of reaffiliations that are established with esteem to the total number of nodes in the network. The number of reaffiliations incremented linearly when there were 30 or more nodes in the network for both WCA and DWCA. But for our algorithm, the number of reaffiliations increased starting from 50 nodes. We submit to engender homogeneous clusters whose size is between two thresholds: $Thresh_{Upper} = 18$ and $Thresh_{Lower} = 9$. According to the results, our algorithm presented a better performance in terms of the number of reaffiliations. The benefit of decreasing the number of reaffiliations mainly comes from the localized reaffiliation phase in our algorithm. The result of the remaining amount of energy per node for each protocol AODV and DSDV is presented in Figure 16 such as R equal to 35 m. As shown in the above-mentioned figure, the remaining energy for each node in AODV protocol is greater than that in DSDV protocol such as AODV which consumes 22, 74% less than DSDV. According to the results, the network consumes 19, 23% of the total energy when we use an AODV protocol (192322.091 NJ). However, it consumes

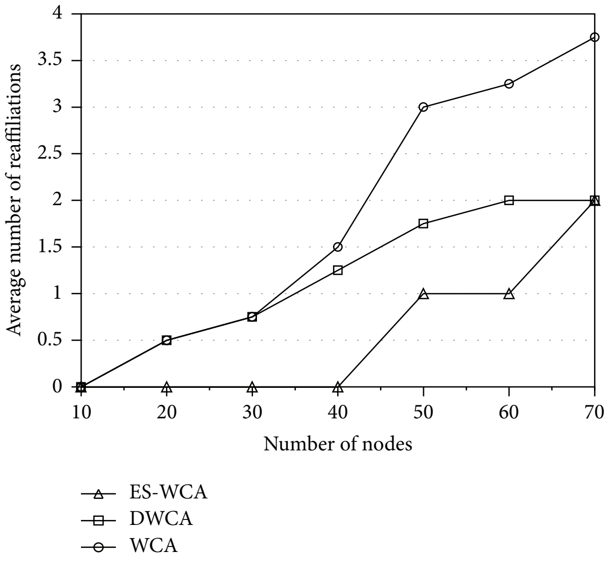


Figure 15: Average number of reaffiliations.

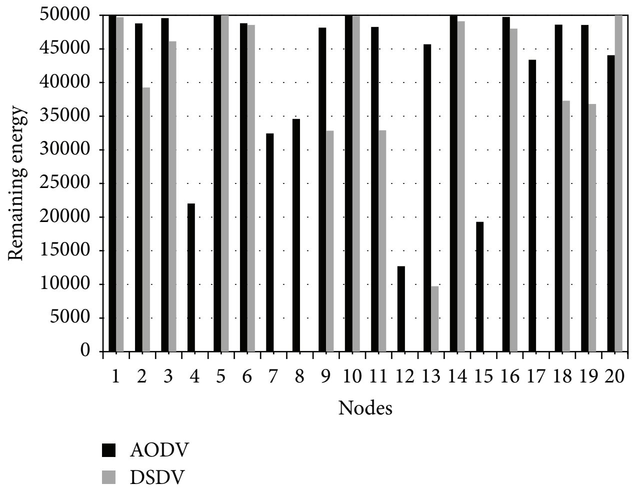


Figure 16: Remaining energy per node using ES-WCA.

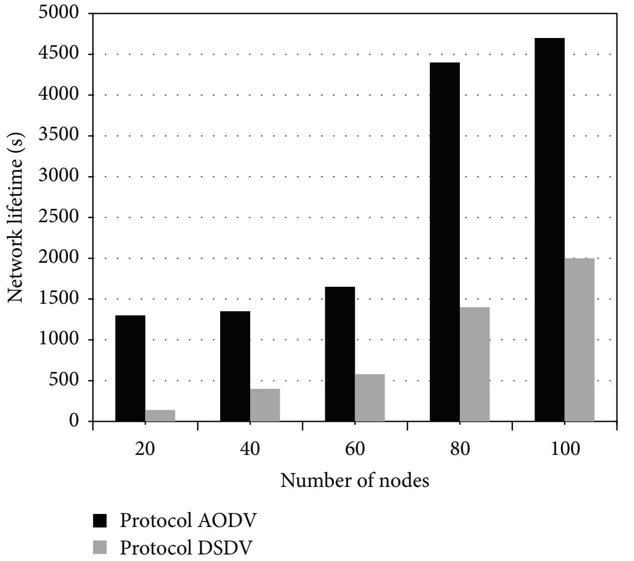


Figure 17: Network lifetime depending on number of nodes using ES-WCA.

Table 3: Detection of the nature of attacks.

IDs	Packets_Sent	Packets_Received	Attack
41	(19, 13)	(16, 14)	Node Outage
71	(24, 152)	(20, 34)	Hello Flood
162	(15, 8)	(22, 112)	Sinkhole
181	(16, 179)	(26, 42)	Hello Flood
190	(58, 32)	(50, 51)	Black Hole

41, 97% with a DSDV protocol (419740.129 NJ). We also observe that the network lost 6 nodes with DSDV but only one node with AODV because of the depletion of its battery. This result clearly shows that AODV outperforms DSDV. This is due to the tremendous overhead incurred by DSDV when exchanging routing tables and the periodic exchange of the routing control packets. So, our algorithm gave a better performance in terms of saving energy when it is coupled with AODV.

We consider that the network will be inoperative when the nodes of the neighborhood of the sink exhaust their energy as exemplified. In Figure 17, we appraise the network lifetime by changing the number of nodes such as R equal to 70 m. When there were 20 nodes in the network, AODV increases the network period about 88, 47% compared to DSDV and about 57,9% for $N = 100$. Also, this is for the reason that in a DSDV protocol each node must have a global view of the network. This in turn raises the number of the exchanged control packets (overhead) in the full network and it decreases the residual energy of each node which has a direct effect on the network lifetime. There are 9 nodes in an active state but the network is inoperative. We discover that the increase in the total of nodes does not have a powerful factor on the network lifetime except between $N = 60$ and $N = 80$.

To illustrate the effect of abnormal behavior in the network, in our experiments we propagated 200 nodes with 5 malicious nodes. The cases of the malicious nodes will pass from a normal node with a yellow color to an abnormal node with a blue color, to a suspicious node with a grey color, and lastly, to a malicious node with a black color. All the cases of the CMs are discovered by their CH. Malicious CHs are disclosed by the base station.

Figure 18(b) displays the total of clusters established according to the transmission range. Figures 19(a), 19(b), and 19(c) display the measure results for a scenario with malicious nodes which are achieved by the generator of bad behavior. The generated attacks are explained in Section 3. We can identify that these nodes migrate from a normal case to an abnormal or suspicious state and finally to a malicious state as expected. Table 3 presents the Ids of malicious nodes and

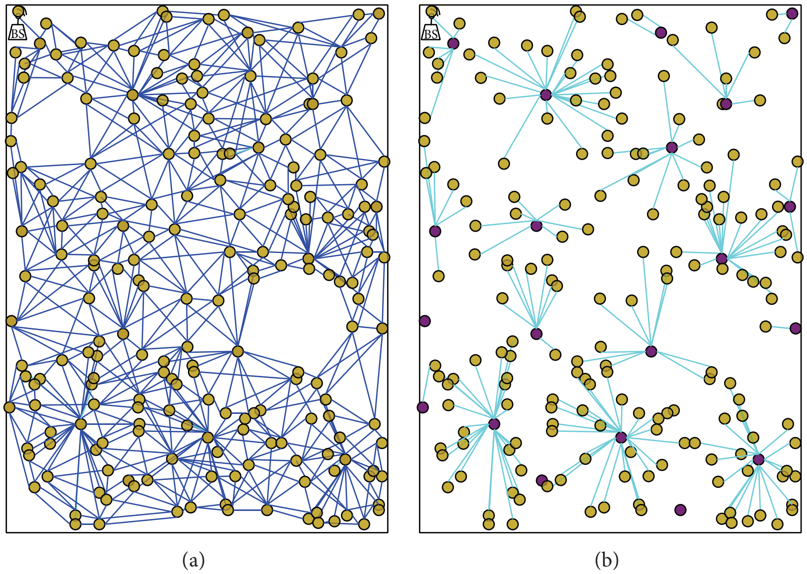


Figure 18: (a) Graph connectivity of 200 nodes. (b) Network after clustering formation.

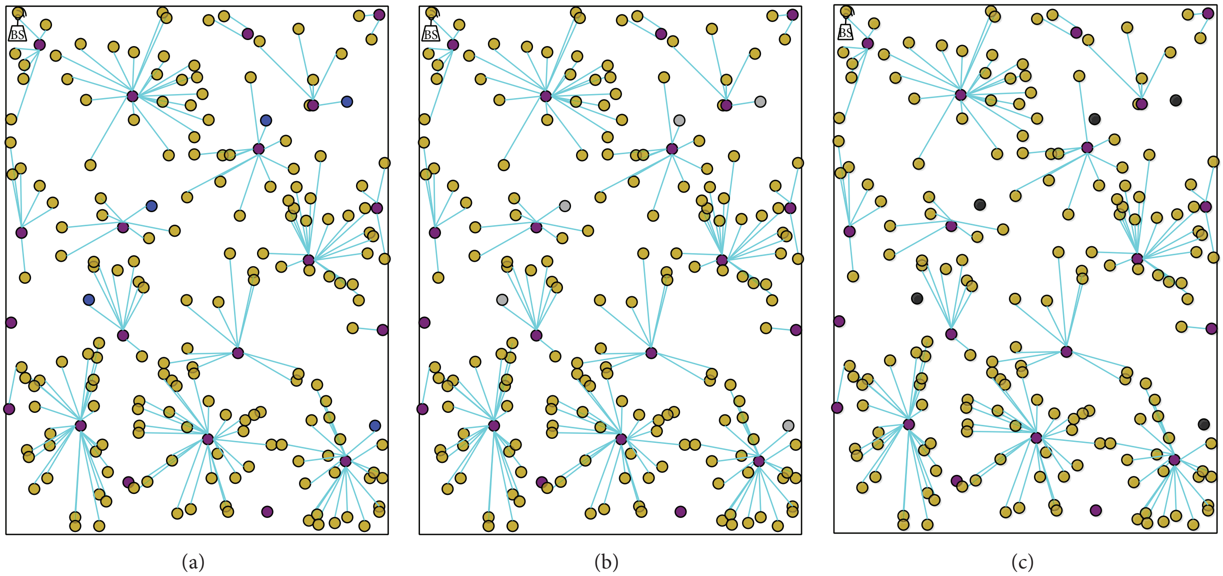


Figure 19: (a) Sensors with a blue color are abnormal but not malicious. (b) The grey sensors have a suspect behavior. (c) The sensors with a black color are compromised and are exhibiting malicious behavior.

their categories of attacks in the course of the dissemination of a monitoring mechanism in the network by the follow-up of the messages exchanged between the nodes. When Packets_sent [$N1, N2$], Packets_received [$N3, N4$]. Thus, $N1$ is the total of packets sent before attacks, and $N2$ is the total of packets sent after attacks, while $N3$ is the total of packets received before attacks and $N4$ is the total of packets received after attacks. We regard that these malicious nodes increment $N1$, as the sensors (71, 181), reduce $N1$, like the sensor (190), increment $N3$, as the sensor (162), and lastly break sending data like node (41). From Figure 20 it is observed that the sensor nodes (3, 17) are malicious and have a behavior level less than 0.3, its behavior decreased by 0.1 units, and when the monitor (CH) counts one failure an alarm is raised. However, packets from malicious nodes are not processed and no packet will be forwarded to them. The sensor node (11) has the behavior level less then threshold behavior so it will not be accepted as a CH candidate even if it has the other interesting characteristics (Er_i, C_i, D_i, and M_i). On the other side the behavior level in Figure 21 decreased by 0.001 units in our first work [4] when the malicious node moves frequently. We note that sensor (6) is suspicious so if it continues to move frequently its behavior will gradually be decreased until it reaches the malicious state; in this case this node will be

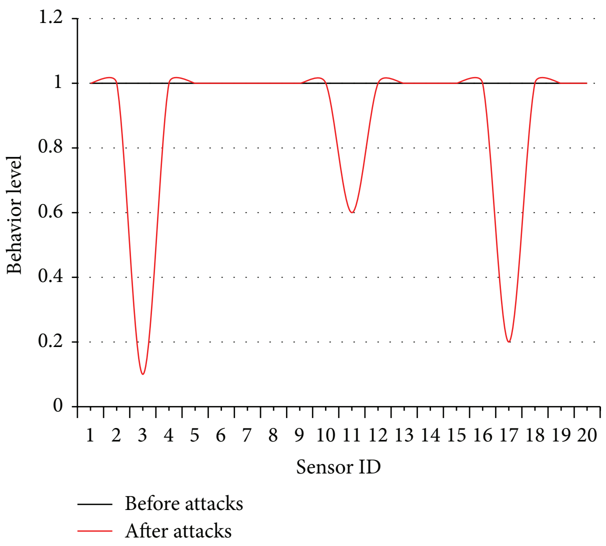


Figure 20: Behavior level of some sensors (moves frequently).

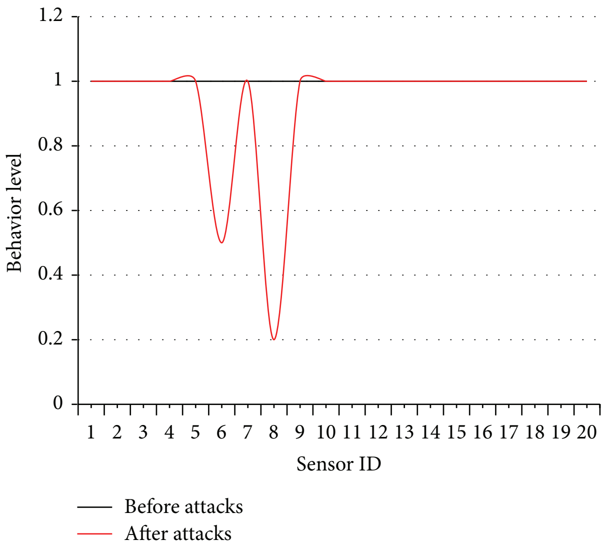


Figure 21: Behavior level of some sensors before and after attacks.

deleted from the neighborhood and finally it will be added to the black list.

7. Conclusion and Future Works

In this paper, we have presented a new algorithm called "ES-WCA" for promoting the self-organization of mobile sensor networks. This algorithm is fully decentralized and aims at creating a virtual topology with the purpose to minimize frequent reelection of the cluster head (CH) and avoid overall restructuring of the entire network. Simulations result attest of the outperformance of our algorithm compared to WCA and DWCA in every sense. It yields a low number of clusters and it preserves the network structure better than WCA and DWCA by reducing the number of reaffiliations. The proposed algorithm selects the most robust and safe CHs with the responsibility of monitoring the nodes in their clusters and maintaining clusters locally. Our third algorithm analyses and detects specific misbehavior in the WSNs. The results show that in scenarios in which mobile WSNs are with a low density or with a small size, the choice of ES-WCA with AODV is comparable to ES-WCA with DSDV to show clearly the interest of the routing protocols in energy saving. However, the difference in favor between ES-WCA and AODV becomes very important in case of a high node density. This is due to the tremendous overheads incurred by ES-WCA with DSDV when exchanging routing tables and exchanging routing control packets. Future work includes considering further the concept of redundancy by using the "sleep" and "wakeup" mechanism in case of node failure, providing in-network processing by aggregating correlated data in order to reduce both the energy consumption and the congestion issue.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The authors are grateful to the anonymous referees for their insightful comments and valuable suggestions, which greatly improved the quality of the paper.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[3] M. Chatterjee, S. Das, and D. Turgut, "WCA: a weighted clustering algorithm for mobile ad hoc networks," *Journal of Cluster Computing*, vol. 5, no. 2, pp. 193–204, 2002.

[4] A. Dahane, N. E. Berrached, and B. Kechar, "Energy efficient and safe weighted clustering algorithm for mobile wireless sensor networks," in *Proceedings of the 9th International Conference on Future Networks and Communications (FNC '14)*, vol. 34, pp. 63–70, Procedia Computer Science (Elsevier), Niagara Falls, Canada, August 2014.

[5] Q. Dong and W. Dargie, "A survey on mobility and mobility-aware MAC protocols in wireless sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 88–100, 2011.

[6] M. Ali, T. Suleman, and Z. A. Uzmi, "MMAC: a mobility-adaptive, collision-free MAC protocol for wireless sensor networks," in *Proceedings of the 24th IEEE International Performance, Computing, and Communications Conference (IPCCC '05)*, pp. 401–407, IEEE, April 2005.

[7] Y. Yu and L. Zhang, "A secure clustering algorithm in mobile ad-hoc networks," in *Proceedings of the IACSIT Hong Kong Conferences*, vol. 29, pp. 73–77, 2012.

[8] I. I. Er and W. K. G. Seah, "Mobility-based d-hop clustering algorithm for mobile ad hoc networks," in *Proceedings of the*

IEEE Wireless Communications and Networking Conference (WCNC '04), pp. 2359–2364, March 2004.

[9] W. Choi and M. Woo, "A distributed weighted clustering algorithm for mobile ad hoc networks," in *Proceedings of the IEEE Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW '06)*, p. 73, February 2006.

[10] A. Zabian, A. Ibrahim, and F. Al-Kalani, "Dynamic head cluster election algorithm for clustered Ad-Hoc networks," *Journal of Computer Science*, vol. 4, no. 1, pp. 42–50, 2008.

[11] M. Chawla, J. Singhai, and J. L. Rana, "Clustering in mobile ad- hoc networks: a review," *International Journal of Computer Science and Information Security*, vol. 8, no. 2, pp. 293–301, 2010.

[12] R. Agarwal, R. Gupta, and M. Motwani, "Review of weighted clustering algorithms for mobile ad-hoc networks," *Computer Science and Telecommunications*, vol. 33, no. 1, pp. 71–78, 2012.

[13] H. Safa, H. Artail, and D. Tabet, "A cluster-based trust-aware routing protocol for mobile ad hoc networks," *Wireless Networks*, vol. 16, no. 4, pp. 969–984, 2010.

[14] Sikander, M. Zafar, A. Raza, M. Babar, S. Mahmud, and G. Khan, "A survey of cluster-based routing schemes for wireless sensor networks," *Smart Computing Review Networks*, vol. 3, no. 4, pp. 261–275, 2013.

[15] X. Liu, "A survey on clustering routing protocols in wireless sensor networks," *Sensors*, vol. 12, no. 8, pp. 11113–11153, 2012.

[16] S. Soro and W. B. Heinzelman, "Cluster head election techniques for coverage preservation in wireless sensor networks," *Ad Hoc Networks*, vol. 7, no. 5, pp. 955–972, 2009.

[17] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer Communications Journal*, vol. 30, no. 14-15, pp. 2826–2841, 2007.

[18] K. A. Darabkh, S. S. Ismail, M. Al-Shurman, I. F. Jafar, E. Alkhader, and M. F. Al-Mistarihi, "Performance evaluation of selective and adaptive heads clustering algorithms over wireless sensor networks," *Journal of Network and Computer Applications*, vol. 35, no. 6, pp. 2068–2080, 2012.

[19] V. Geetha, P. Kallapur, and S. Tellajeera, "Clustering in wireless sensor networks: performance comparison of LEACH & LEACH-C protocols using NS2," *Procedia Technology*, vol. 4, pp. 163–170, 2012.

[20] Y. Wang, X. Wu, J. Wang, W. Liu, and W. Zheng, "An OVSF code based routing protocol for clustered wireless sensor networks," *International Journal of Future Generation Communication and Networking*, vol. 5, no. 3, pp. 117–128, 2012.

[21] K. Benahmed, M. Merabti, and H. Haffaf, "Distributed monitoring for misbehaviour detection in wireless sensor networks," *Security and Communication Networks*, vol. 6, no. 4, pp. 388–400, 2013.

[22] J. Y. Yu and P. H. J. Chong, "A survey of clustering schemes for mobile ad hoc networks," *IEEE Communications Surveys and Tutorials*, vol. 7, no. 1, pp. 32–47, 2005.

[23] J. Y. Yu and P. H. J. Chong, "A survey of clustering schemes for mobile ad hoc networks," *IEEE Communications Surveys and Tutorials*, vol. 7, no. 1–4, pp. 32–47, 2005.

[24] A. Jain and B. V. R. Reddy, "A novel method of modeling wireless sensor network using fuzzy graph and energy efficient fuzzy based k-hop clustering algorithm," *Wireless Personal Communications*, vol. 82, no. 1, pp. 157–181, 2015.

[25] T. Kavita and D. Sridharan, "Security vulnerabilities in wireless sensor networks: a survey," *Journal of Information Assurance and Security*, vol. 5, pp. 31–44, 2010.

[26] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.

[27] S. Ganeriwal and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '04)*, pp. 66–77, October 2004.

[28] Y. Yu and L. Zhang, "A secure clustering algorithm in mobile ad-hoc networks," in *Proceedings of the 2012 IACSIT Hong Kong Conferences*, vol. 29, pp. 73–77, Hong Kong, 2012.

[29] X. Liu, "A survey on clustering routing protocols in wireless sensor networks," *Sensors*, vol. 12, no. 8, pp. 11113–11153, 2012.

[30] T. H. Hai, E.-N. Huh, and M. Jo, "A lightweight intrusion detection framework for wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 10, no. 4, pp. 559–572, 2010.

[31] M. E. Elhdhili, L. B. Azzouz, and F. Kamoun, "Reputation based clustering algorithm for security management in ad hoc networks with liars," *International Journal of Information and Computer Security*, vol. 3, no. 3-4, pp. 228–244, 2009.

[32] S. Taneja and A. Kush, "A survey of routing protocols in mobile ad-hoc networks," *International Journal of Innovation, Management and Technology*, vol. 1, no. 3, pp. 279–285, 2010.

[33] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proceedings of the Conference on Communications Architectures, Protocols and Applications (SIGCOMM '94)*, pp. 234–244, ACM, London, UK, September 1994.

[34] D. G. Padmavathi and D. Shanmugapriya, "A survey of attacks, security mechanisms and challenges in wireless sensor networks," *International Journal of Computer Science and Information Security*, vol. 4, no. 1-2, pp. 1–9, 2009.

[35] P. Li, L. Sun, X. Fu, and L. Ning, "Security in wireless sensor networks," in *Wireless Network Security*, pp. 179–227, Higher Education Press, Springer, Berlin, Germany, 2013.

[36] W. Stallings, *Cryptography and Network Security, Principles and Practice*, Prentice Hall, 5th edition, 2010.

[37] M. Safiqul-Islam and S. Ashiqur-Rahman, "Anomaly intrusion detection system in wireless sensor networks: security threats and existing approaches," *International Journal of Advanced Science and Technology*, vol. 36, pp. 1–8, 2011.

[38] P. Berwal, "Security in wireless sensor networks: issues and challenges," *International Journal of Engineering and Innovative Technology*, vol. 3, no. 5, pp. 192–198, 2013.

[39] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," *Ad-Hoc Networks Journal*, vol. 1, no. 2-3, pp. 293–315, 2003.

[40] S. Dai, X. Jing, and L. Li, "Research and analysis on routing protocols for wireless sensor networks," in *Proceedings of the International Conference on Communications, Circuits and Systems*, vol. 1, pp. 407–411, May 2005.

[41] M. Tripathi, M. S. Gaur, and V. Laxmi, "Comparing the impact of black hole and gray hole attack on LEACH in WSN," in *Proceedings of the 4th International Conference on Ambient Systems, Networks and Technologies (ANT '13) and the 3rd International Conference on Sustainable Energy Information Technology (SEIT '13)*, vol. 19, pp. 1101–1107, June 2013.

[42] R. A. Shaikh, H. Jameel, S. Lee, S. Rajput, and Y. J. Song, "Trust management problem in distributed wireless sensor networks," in *Proceedings of the 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA '06)*, pp. 411–414, August 2006.

[43] M. Lehsaini, H. Guyennet, and M. Feham, “An efficient cluster-based self-organisation algorithm for wireless sensor networks,” *International Journal of Sensor Networks*, vol. 7, no. 1-2, pp. 85–94, 2010.

[44] Y. Li, F. Wang, F. Huang, and D. Yang, “A novel enhanced weighted clustering algorithm for mobile networks,” in *Proceedings of the IEEE 5th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM ’09)*, pp. 2801–2804, IEEE, Beijing, China, September 2009.

[45] A. da Silva, M. Martins, B. Rocha, A. Loureiro, L. Ruiz, and H. Wong, “Decentralized intrusion detection in wireless sensor networks,” in *Proceedings of the 1st ACM International Workshop on Quality of Service & Security in Wireless and Mobile Networks*, pp. 16–23, 2005.

[46] K. Benahmed, H. Haffaf, and M. Merabti, “Monitoring of wireless sensor networks,” in *Sustainable Wireless Sensor Networks*, Y. K. Tan, Ed., chapter 3, InTech, 2010.

[47] P. Levis, N. Lee, M. Welsh, and D. Culler, “TOSSIM: accurate and scalable simulation of entire TinyOS applications,” in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys ’03)*, pp. 126–137, November 2003.

[48] V. Shnayder, M. Hempstead, B.-R. Chen, G. W. Allen, and M. Welsh, “Simulating the power consumption of large-scale sensor network applications,” in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, pp. 188–200, November 2004.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2015, Article ID 187548, 9 pages
http://dx.doi.org/10.1155/2015/187548

Research Article

AVL and Monitoring for Massive Traffic Control System over DDS

Basem Almadani,[1] Shehryar Khan,[1] Muhammad Naseer Bajwa,[1] Tarek R. Sheltami,[1] and Elhadi Shakshuki[2,3]

[1] *Department of Computer Engineering, KFUPM, Dhahran 31261, Saudi Arabia*
[2] *Jodrey School of Computer Science, Acadia University, Wolfville, NS, Canada B4P 2R6*
[3] *King Faisal University, Al-Ahsa 31982, Saudi Arabia*

Correspondence should be addressed to Shehryar Khan; g201303590@kfupm.edu.sa

Received 10 November 2014; Accepted 15 January 2015

Academic Editor: Ansar-Ul-Haque Yasar

This paper proposes a real-time Automatic Vehicle Location (AVL) and monitoring system for traffic control of pilgrims coming towards the city of Makkah in Saudi Arabia based on Data Distribution Service (DDS) specified by the Object Management Group (OMG). DDS based middleware employs Real-Time Publish/Subscribe (RTPS) protocol that implements many-to-many communication paradigm suitable in massive traffic control applications. Using this middleware approach, we are able to locate and track huge number of mobile vehicles and identify all passengers in real-time who are coming to perform annual Hajj. For validation of our proposed framework, various performance matrices are examined over WLAN using DDS. Results show that DDS based middleware can meet real-time requirements in large-scale AVL environment.

1. Introduction

Applications of distributed mobile networks exist in everyday life in the form of transportation systems, healthcare systems, weather and environment monitoring systems, and so forth. Such systems require their mobile nodes to communicate and share data among them in real-time. Mobile nodes in these scenarios may be hand held devices, vehicles [1, 2], or robots [3, 4]. With the advancement in embedded systems, it is now possible to allow thousands of mobile nodes to communicate and share huge amount of data. It is also possible to collect the data at the sensor level and forward it to the application level for processing and analysis, all in real-time. These nodes need to share their context updates regularly. In AVL applications, vehicles assume the role of distributed mobile nodes and they require sharing their information such as their locations, vehicles identification, and passenger status. The passengers' information can be extracted using Radio Frequency Identification (RFID) system.

The city of Makkah in Kingdom of Saudi Arabia hosts millions of Hajj pilgrims every year. Hajj consists of number of rituals, spanning over five designated days that are to be performed in specific locations in and around the city of Makkah. The number of pilgrims is increasing each year and about 3.1 million pilgrims performed Hajj in 2012 [5]. Some of these people get lost in huge crowd and, therefore, there is dire need to identify them, trace their position, and inform their families. There are many models that have been proposed to facilitate the organizers such as in [6] three security requirements are detailed and lattice model is proposed for flow of information. In [7] a bus transportation system within the city of Makkah is proposed and validity of this system is tested using simulation and experiments.

Addressing this challenge necessitates formulation of a framework that identifies and monitors the pilgrims as well as their road transport. Such a large-scale system calls for a vast and scalable infrastructure that supports reliable and instant context updates for sharing among the mobile nodes [8] and, at the same time, is able to dynamically adjust to the load demand. In distributed mobile communication environments, the nodes are limited in power and resources.

The network connections for these mobile nodes are also fluctuating and may suffer from frequent disconnections.

The publish/subscribe (PS) model is by far the most suitable for mobile distribution environment. Although many researchers have attempted to develop publish/subscribe model [9–13], yet only a few of them are able to support mobile networks. The publish/subscribe model has two distinct characteristics. First, it efficiently distributes large amount of data to large number of users. Secondly, the publisher and the subscriber are not required to connect simultaneously in order to distribute data. In PS model, both communicating participants do not know about each other's existence. Nowadays, industrial automation, aerospace, and defense applications use Data Distribution Service (DDS) based middleware [14, 15]. The work presented in this paper also uses DDS based middleware for our application of Automatic Vehicle Location.

DDS is specification for real-time scalable middleware. Its architecture is decentralized and it realized an asynchronous communication model. It specifies many Quality of Service (QoS) policies such as reliability, data flow prioritization, data persistence, and other optimization schemes that are used to control various aspects of data transmission. The unique property of DDS based middleware is that the efficiency of network resources and latency can be controlled by fine tuning some of the QoS policies like latency budget, deadline, and transport priority. In our application, we chose DDS based middleware because of the following characteristics that support mobile distributed environments.

(A) Asynchronous Interaction. Network connections in distributed mobile environment have high error rate, high disconnection frequency, and bandwidth limit due to the limited power or lack of spectrum availability. Therefore, asynchronous communication is better than synchronous communication in this type of applications.

(B) Data Sharing. The data to mobile devices must be available at all times even when disconnected from the main server. This capability of data sharing and disconnected operation is significant feature in DDS based middleware. It should be noted that consistent, reliable, and efficient accessibility to the database should be provided by the mobile distributed information system. This implies that global distributed database should also be synchronized among all nodes even after disconnection.

(C) Reflection and Dynamic Reconfiguration. A heterogeneous environment of dynamic context is encountered by the devices in mobile communication scenarios. In many heterogeneous systems, devices may exhibit different behaviors due to diversity in their network protocols, I/O interfaces, and OS. Thus, mobile nodes/devices should be adaptive to the available resources, which require techniques to optimize the behavior of nodes. Therefore, the middleware should detect the changes in the availability of resources to support dynamic reconfiguration. It also has to notify the application for the reallocation of resources to different nodes.

2. Related Work

For large-scale mobile system, a middleware called Scalable Context-Aware Middleware for Mobile Environments (SALES) is developed in [16]. It is a tree based classified model of nodes for performance evaluation, load balancing, and calculating communication cost among the following four types of nodes: (1) central node, (2) base node, (3) coordinator user node, and (4) simple user. SALES does not take advantage of real-time DDS and uses UDP. Two main terminologies are used: Quality of Context (QoC) and Context Data Distribution Level Agreement (CDDLA). QoC is associated with context information distributive service whereas CDDLA is quality agreement between consumer and producer imposed by the middleware. This SALES architecture lacks the functionality of fault tolerance, QoS support, and context updates by mobile nodes.

For ubiquitous computing, a middleware named Solar is developed in [17]. It uses two protocols: TCP for interplanetary communication and Distributed Hash Table (DHT) for routing and discovery. It is built on the basis of self-organizing peer-to-peer network and is designed for scalability among the set of communicating nodes. It employs filter and pipe programming model. Each filter has group of entry and exit points as well as sources and sinks. Each node in the solar architecture is viewed as planet and each planet has a number of satellite nodes. Solar has the reliability of TCP that may not be suitable for some real-time applications. It also lacks the functionality of fault tolerance. A limited research is done in the implementation of mobile distributed applications using DDS based middleware. Among few of them is [4]. Its architecture supports mobile nodes and provides reliable data delivery. It also supports handover by switching the wireless access points. The mobile nodes in the proposed middleware execute light version of DDS, whereas the fixed nodes execute full version and are responsible for routing and data delivery among all nodes. Due to the architecture of this middleware, mobile nodes have to run in a single domain and stable wireless connectivity is necessary. This proposed middleware, too, lacks the functionality of fault tolerance.

REliable and VErsatile News delivery support for aGEncies (Revenge) is a DDS based middleware, which serves as news dispatching service among mobile nodes [18]. It efficiently disseminates the data among the mobile nodes in DDS domain. It is also based on self-organizing peer-to-peer network and is fault tolerant. Revenge efficiently detects the crashed nodes and reroutes the paths from any source to any sink. This architecture, however, lacks the functionality of handover. Another DDS based middleware is proposed in [19] for real-time communication between mobile nodes using proxy approach. A proxy DDS client is used for managing, coordinating, and forwarding all the data to mobile nodes. This architecture provides both reliable and unreliable data delivery. Due to the Firewall/NAT restrictions, mobile nodes have to run within a single domain with continuous connectivity.

A DDS based middleware called Scalable Data Distribution Layer (SDDL) [20, 21] is proposed for real-time tracking of mobile nodes. This middleware connects the stationery

DDS nodes in a wired network to the mobiles nodes with IP based wireless connection. Two protocols are used in this middleware: RTPS wire protocol for communication among the stationery nodes and mobile reliable UDP protocol for communication among the mobile nodes. In wired core network, there are three types of stationery nodes with unique roles: (1) gateway that is responsible for maintaining separate connections with mobile nodes through Mobile Reliable-(MR-) UDP connections; it is also responsible for notifying other core network nodes when a mobile node is connected or disconnected; (2) Point of Attachments (PoA) Manager that distributes the list of points of attachment/gateways to mobile nodes and it may later switch to different gateway; mobile nodes may also switch the gateway when they detect weak connection with the current gateway; and (3) group definers, which are responsible for evaluation of group-membership of mobile nodes. They subscribe to a specific DDS topic and map each node to one or more groups corresponding to application specific group membership logic. When a new message is sent to a group, a gateway asks for group-to-mobile-node mapping to know which mobile node is ready to send messages.

An RFID system based on publish/subscribe middleware is introduced in [22]. Since different applications require different data, middleware has to adapt to all applications. When more applications need to be incorporated, it is necessary to adjust middleware functionalities to satisfy them. This, however, costs time and effort. Using publish/subscribe mechanism applications can subscribe to events from the RFID reader dynamically. In this model filtering of messages can be topic based or content based. In topic based filtering, the subscriber receives messages published to a specific topic. All subscribers will receive the same messages for a particular topic. In content based filtering, subscribers will receive messages if the content of the messages match the constraints set by the subscribers. Four components are proposed for publish/subscribe model in this RFID middleware: (1) list of publications maintained by the message manager, (2) list of subscriptions maintained by the message manager, (3) message manager that acts as a controller and responds to the requests of the reader manager for maintenance; it also responds to the client queries for the list of publication and maintains the list of subscribers, and (4) an API with a set of routines and protocols to allow the clients to subscribe and unsubscribe.

3. DDS Architecture

DDS specifies a communication model that is data centric publish/subscribe for various computing and distributed environments. This data centric publish/subscribe (DCPS) uses a Global Data Space (GDS) where publishers post and the DCPS model disseminates this information to all the interested subscribers.

Figure 1 illustrates all the basic constructs in a simple DCPS model. A domain is a virtual network area and all the publishers and subscribers can send and receive messages within a domain. A publisher is an object that, according to the publisher's QoS policies, distributes data and publishes different types of data objects. Data writers are used by the publishers to write data to the GDS. A subscriber receives the publishing data using data readers and, according to subscriber's QoS policies, makes them available to the receiving application. It reads the topics from GDS for which there exists a matching subscription and data readers are informed that data is received.

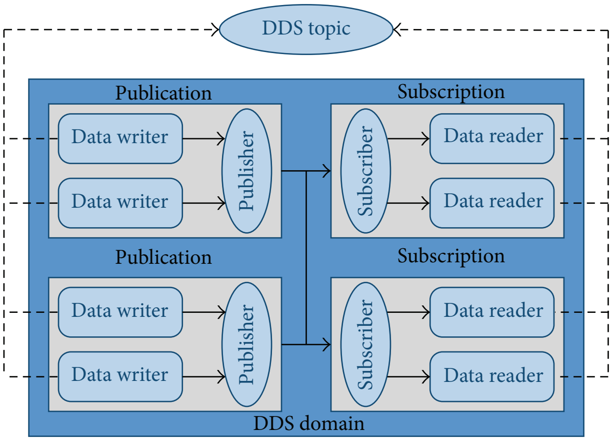


Figure 1: DDS architecture [19].

The DDS has two types of discovery protocols: the Participant Discovery Protocol (PDP) and the Endpoint Discovery Protocol (EDP). Through these protocols participants can dynamically discover the existence of other participants and also inform other participants about the end points such as data readers and data writers and so forth. The basic architecture of DDS is described as the unidirectional data communication in a fixed network. In this network a publisher pushes the data to the subscriber which is saved in its history caches. Topic is the data type that links the publishers and subscribers. Implementation requires this discovery protocol to identify the existence, presence, and absence of the endpoints when the network is joined or left by them. Another key distinguishing feature of DDS as compared to another publish/subscribe middleware is that it has rich QoS support. The behavior and performance of the DDS service and how it performs the various coordination tasks depend upon how its QoS policies are configured. For mobile environment the useful QoS policies are durability, history, and reliability. Below is a brief description of each of them.

(i) For mobility support, RELIABILTY QoS policy is used indicating the reliability level between publisher and subscriber. If RELIABILTY is set to BEST_EFFORT then the publisher will push the samples without retransmissions or expecting acknowledgements. If RELIABILITY is set to RELIABLE then publisher will guarantee the delivery of samples to the subscriber. If the subscriber is disconnected or unavailable and samples are unacknowledged then the publisher can retransmit the samples using the DURABILITY QoS policies.

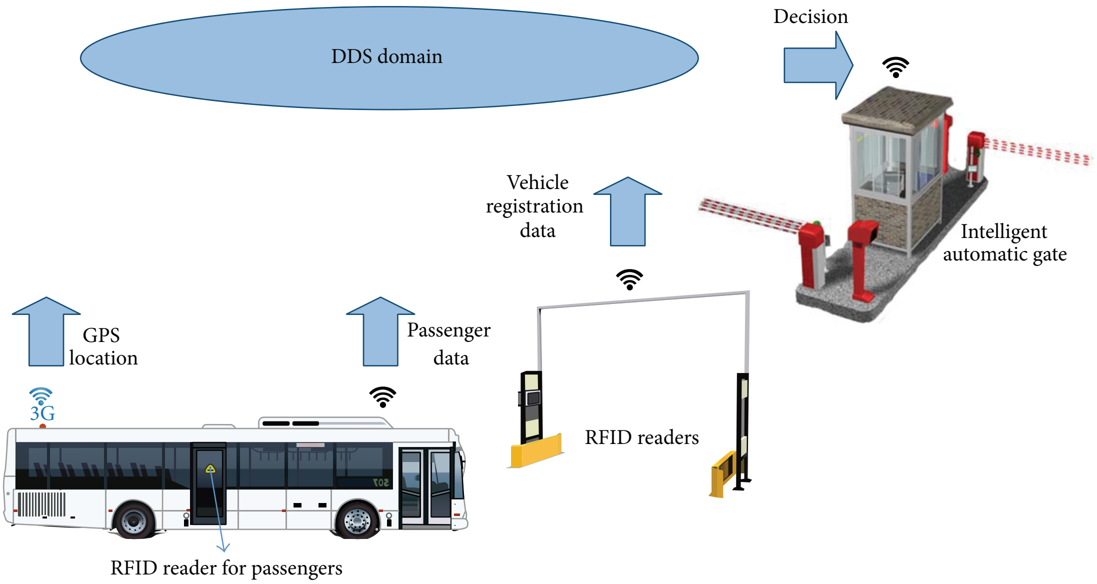

DDS domain
Decision
Vehicle registration data
Intelligent automatic gate
GPS location
3G
Passenger data
RFID readers
RFID reader for passengers

Figure 2: AVL and monitoring system architecture.

(ii) DURABILITY QoS indicates whether previous data samples will be made available to late joining subscriber by the publisher or not. By setting the DURABILITY policy to PERSISTENCE, the past data samples are stored in the permanent memory such as FLASH, HARD DISK. Due to this the subscriber can have this data when it reconnects to the system at any time. The DURABILITY policy depends upon the HISTORY QoS policy and RESOURCE_LIMITS QoS policy.

(iii) HISTORY QoS specifies the maximum number of data samples that are stored in the History Cache of publisher for reconnecting or late joining subscribers.

(iv) RESOURCE LIMIT specifies maximum number of samples and instances which publisher or subscriber can manage.

4. Proposed Solution

In this section we propose a framework for location and monitoring system based on DDS based middleware especially for pilgrims of Hajj. To provide control and location services of this huge crowd, each pilgrim can be provided with a GPS transmitter; however, implementing this solution is too expensive. Similarly, another possibility is that each person can be provided with a radio transmitter with unique identifier and the transmitter can update pilgrim's location by sending update to the database via base station. Although it may be useful, signal coverage can be a problem. Our solution addresses these challenges during journey of pilgrims and offers crowd management and prevention of accidents using the architecture depicted in Figure 2.

Proposed approach can manage a huge system consisting of about 50,000 buses as mobile nodes as well as the passengers inside these vehicles. This system is especially useful for ensuring that all pilgrims have valid Hajj permits. Assume there are various entry points to the city of Makkah and on each entry point there is check posts almost 10 Km from the city. Three layers of security are implemented on each check post. These include RFID readers to read electronic license plates, distributed database server, and automatic gates access based upon previous two security layers. The detailed layout of the framework is given below.

4.1. Bus or Mobile Node

(i) *GPS and 3G Device.* Every bus has a 3G device that periodically updates its current locations using GPS.

(ii) *Thermal Camera.* These cameras count number of passengers in the bus and send this information to local database in the bus.

(iii) *RFID Readers.* Each passenger has RFID wrist bands with their personal information. This information is collected using RFID readers inside the bus and sent to local database.

(iv) *WiFi Device.* Local database is synchronized with DDS cloud using WiFi.

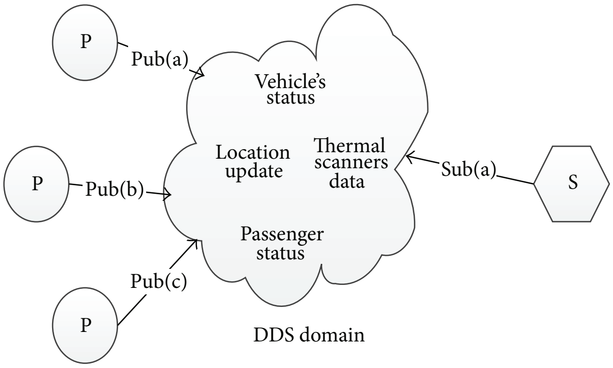
The message format for RFID passenger data is shown in Figure 3.

4.2. The Check Post. The check posts are established at different entry points of the city. Each check post has various parallel lines. They have distributed servers and databases which are part of the DDS cloud. Also, multiple RFID readers are installed for vehicle's identification through passive RFID license plates.

4.3. The Automatic Gate. At the end of each check post there is an automatic gate that will only open after a positive

Pilgrim ID	Name
Country	Passport number
ID expiration	Group
Agency	

Figure 3: Message format in RFID message.



S: subscriber
P: publisher

Figure 4: DDS domain.

decision is made inside the cloud for a particular vehicle based on different factors such as vehicle identification, passenger counting through thermal cameras, and passenger identification and status through RFID data. When a bus arrives at a check post it has to pass through all three stages before it is finally allowed to enter the city.

In the first part of this clearance, there are RFID readers for the verification and authentication of license plates and a decision is made whether this bus is allowed to operate for Hajj pilgrims or not. At this stage it is also verified whether this mobile node is physically present at the checkpoint or not using its GPS location update. This cross checking can deal with identity theft of vehicles. In second part the stored RFID data of passenger identification and status inside the bus is passed to the cloud via WiFi for authentication and decision making. If a mobile node/bus has more than a certain number of unauthorized passengers then the whole bus is rejected; otherwise manual checking is done. In third stage the thermal cameras installed on the bus will do the head counting and will convey this data to the DDS cloud over WiFi. At this point the passenger count data from RFID readers and the thermal cameras are cross checked for further verification in case there are multiple readings for RFID tags.

At the end when a mobile node reaches the automatic gate, a decision is made based on the data provided to the cloud during the previous stages. In our case the number of publishers is in thousands whereas subscriber is one. There are four topics for which publishers are publishing as shown in Figure 4. These topics are vehicle location update, vehicle registration information, passenger head count, and RFID passenger's data.

5. Experimentation Setup

Before going to experimentation and results, some performance parameters are defined below to validate the timeliness of our proposed solution.

(i) The time taken by a data packet to reach the receiver side is known as the latency. This includes both the propagation delay as well as the queuing delay and can be calculated by dividing Round Trip Time (RTT) by 2:

$$\text{Latency} = \frac{\text{RTT}}{2}. \quad (1)$$

(ii) Variation in latency is known as the jitter. The smaller the value of the jitter, the smaller the variation in the delay of packets. It means that with small values of jitter we can be sure that delay of the packets will be same for most of the time during their journey from sender to receiver. Jitter can be calculated as given in

$$\text{Jitter} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_i - \overline{x}\right)^2}, \quad (2)$$

where the total number of delay samples is represented by N and mean value of N delay samples is represented by $\overline{x}$.

(iii) The average rate of successful data transmission is known as the throughput. This includes the payload as well as the protocol overhead. Following equation is used to calculate the throughput:

$$\text{Throughput} = \frac{\text{Packet Size} \times \text{Number of Packets}}{\text{Total Time}}. \quad (3)$$

We provide hardware platform used in our experiment as well as software tools needed in Tables 1 and 2, respectively.

To support mobility and ensure real-time behavior of the system, QoS policies used in our experimentation are shown in Table 3.

Table 1: Platform specifications for WLAN.

	Host A publisher	Host B subscriber
CPU	Intel(R), Core (TM)2 Duo CPU P8800 @ 2.66 GHz	Intel(R), Core (TM)2 Duo CPU P8800 @ 2.66 GHz
OS	Windows 7 64 bit operating system	Windows 7 64 bit operating system
Memory	4 GB	4 GB
Network connection	WLAN IEEE 802.11, 54 Mbps	WLAN IEEE 802.11, 54 Mbps

Table 2: Software specifications for WLAN.

Tool	Version	Purpose
RTI Connext- RTI DDS Latency Test	5.0.0	Measure latency (one way delay)
Wire-shark	1.2.3	Measure throughput

Table 3: QoS policies for mobility.

QoS policy	Publisher	Subscriber
DURABILITY	PERSISTENCE	PERSISTENCE
RELIABILITY	RELIABLE	RELIABLE
HISTORY	KEEP_ALL	KEEP_LAST
RESOURCE_LIMIT	LENGTH_UNLIMITED	1

Table 4: Latency and jitter for many-to-one model.

Number of publishers	Minimum (m sec)	Maximum (m sec)	Avg. (m sec)	Avg. [23] (m sec)	Jitter (m sec)
1	1.28	63.51	2.41	12.18	2.35
2	1.38	57.29	2.47	13.17	1.96
4	1.77	112.44	4.00	12.45	4.68
6	2.06	454.14	4.23	12.71	5.16
8	2.22	707.40	5.95	12.93	7.53

Table 5: Latency and jitter for many-to-many model.

Number of publishers	Number of subscriber	Minimum (m sec)	Maximum (m sec)	Avg. (m sec)	Jitter (m sec)
2	2	1.42	131.93	2.69	2.95
4	2	1.64	237.24	3.43	3.36
4	4	1.55	106.48	2.40	3.39
8	4	2.43	291.53	6.32	4.28

Table 6: Throughput for many-to-one model.

Number of publishers	Total packets (×1000)	Total time (sec)	Throughput (Mbps)	Throughput [23] (Mbps)
1	4585	5536	6.810	0.842
2	2794	3277	6.958	1.598
4	3330	3807	7.168	3.386
6	3780	3986	7.772	3.515
8	5040	5544	7.498	2.91

Table 7: Throughput for many-to-many model.

Number of publishers	Number of subscribers	Total packets (×1000)	Total time (sec)	Throughput (Mbps)
2	2	3560	3706	7.87
4	2	4032	4360	7.67
4	4	4032	2767	12.37
8	4	4032	2730	12.59

6. Results and Analysis

The bottleneck in our proposed framework can be data transmission over WiFi at check posts. Therefore, we conducted experiments to measure various performance matrices for this wireless medium. Latency and jitter can validate if the proposed solution can fulfill real-time requirements in AVL application for Hajj pilgrims. The values of throughput can shed light on whether WiFi can withstand high data-rate demand of such scenario. The obtained results for latency, jitter, and throughput for Wifi over DDS are tabulated in Tables 4–7. There is significant improvement in the results compared to [23] because of the introduction of QoS policies as well as controlled environment for experimentation making sure that only DDS applications are running over WiFi.

For latency and jitter, 1024-byte payload is used. First, one subscriber and multiple publishers scenario is examined. In each run 20,000 to 50,000 packets are sent and the minimum, maximum, and average values are calculated. Tests are repeated up to 10 times and overall average is taken to account for any random fluctuation in performance measures.

Table 4 records measurements obtained from experiments. Jitter is calculated using (2). We can see that both latency and jitter increase linearly as the number of publishers grow. Figures 5 and 6 show latency and jitter graphs corresponding to Table 4. In Figure 5 we can observe that there is considerable improvement in latency as compared to results in [23] because specific QoS policies for mobility are introduced in our scenario and Wifi experimentation was carried out in a controlled lab setup to avoid other applications running on the same WiFi.

Latency and jitter are also calculated for many-to-many communication scenario. Tests are run to examine the effect of multiple publishers and multiple subscribers on the latency and jitter. As expected, both performance measures have higher values when multiple participants try to transmit the data over a single channel. These results are tabulated in Table 5. Figures 7 and 8 show the results graphically.

The data packets' size and the frequency of the transmission affect the throughput. Data size used in our case is up to few hundred bytes except for the local database updates that are synchronized with DDS cloud. These updates are sent in

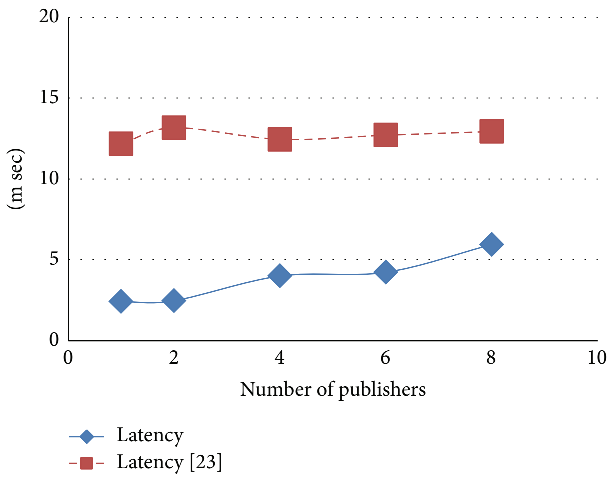


Figure 5: Average latency in many-to-one model.

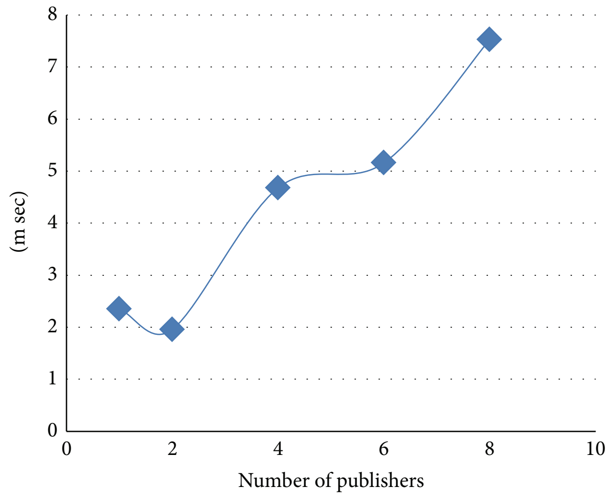


Figure 6: Jitter in many-to-one model.

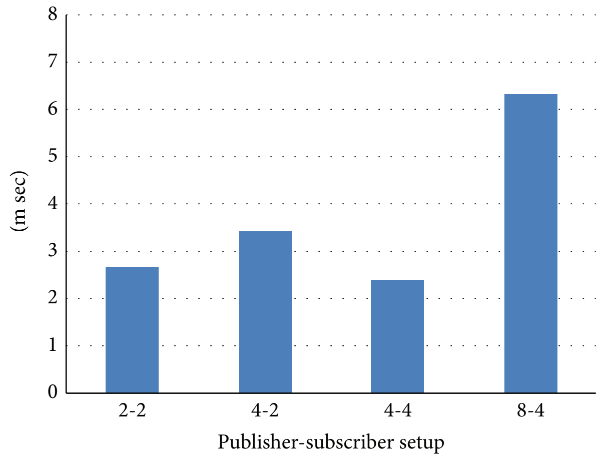


Figure 7: Average latency in many-to-many model.

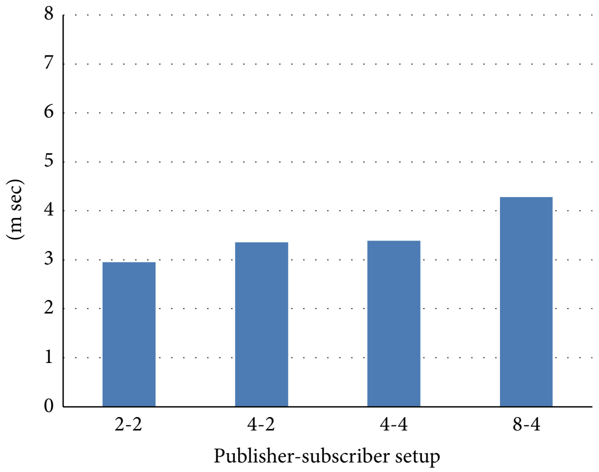


Figure 8: Jitter in many-to-many model.

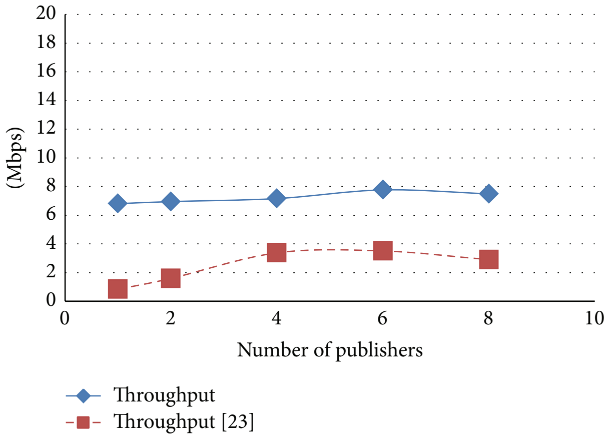


Figure 9: Throughput in many-to-one model.

packets up to 1 KB. Hundreds of thousands of samples are sent from publisher side and received on the subscriber side in each iteration. Total time for this communication is observed and throughput is calculated using (3). The tests are repeated at least 10 times to get more accurate results.

Similar to the latency results we can see in Table 6 and Figure 9 that there is considerable difference in the throughput. The reason is again judicious selection of QoS values in mobile applications. However, we can observe that with the increase in number of participants there is negligible difference in the throughput.

Like many-to-many latency and jitter experiments, we also conducted tests to measure average WiFi in many-to-many mode. Various configurations of publishers and subscribers are examined. Table 7 summarizes the results obtained. We can observe from Figure 10 that throughput is not significantly affected by the number of participants in many-to-many scenarios as well.

Here, it is worth pointing out that the maximum numbers of publishers in our experiments are limited to only eight whereas the total numbers of vehicles are in thousands. The

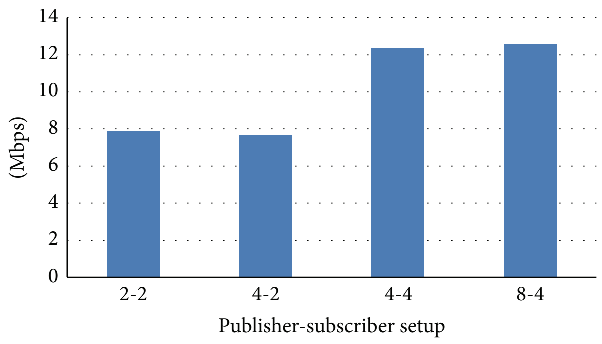


FIGURE 10: Throughput many-to-many model.

reason is that this experimentation shows the data communication over WiFi at checkpoints. If, on each highway, there are four lanes for vehicles then this framework can deal with eight buses simultaneously. And the small values of latency points out that the buses are cleared almost instantaneously within any let or hindrance.

7. Conclusion

Monitoring and tracking vehicles during Hajj have been a tough task for Saudi Arabian authorities for a long time. Illegal Hajj pilgrims also pose a challenge in organization of annual event. In this paper, we introduced a framework to solve this problem of vehicle tracking during Hajj by using OMG's DDS middleware specification. We discussed pilgrims' difficulties and transport congestion problems faced during this period. An automated solution based on Real-Time Publish/Subscribe middleware is proposed for tracking and monitoring of vehicles as well as pilgrims. The use of DDS based middleware is motivated by its data centric and asynchronous communication paradigm along with rich set of QoS policies. Experiments are performed for WLAN over DDS to validate real-time characteristics of our proposed framework. Obtained results for various performance parameters such as latency, jitter, and throughput show that the proposed approach can withstand stringent real-time requirements in AVL application under consideration. These preliminary results are encouraging enough to serve as bases for further development of the framework, and ultimately the infrastructure.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors are thankful to RTI Inc. for granting license for this research. The authors are also thankful to College of Computer Science and Engineering (CCSE), KFUPM, Saudi Arabia, for providing necessary lab equipment and setup for experimentation.

References

[1] D. Stojanović, B. Predić, I. Antolović, and S. Dordević-Kajan, "Web information system for transport telematics and fleet management," in *Proceedings of the 9th International Conference on Telecommunications in Modern Satellite, Cable, and Broadcasting Services (TELSIKS '09)*, pp. 314–317, October 2009.

[2] J. Rybicki, B. Scheuermann, W. Kiess, C. Lochert, P. Fallahi, and M. Mauve, "Challenge: peers on wheels—a road to new traffic information systems," in *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking (MobiCom '07)*, pp. 215–221, September 2007.

[3] G. T. Sibley, M. H. Rahimi, and G. S. Sukhatme, "Robomote: a tiny mobile robot platform for large-scale ad-hoc sensor networks," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1143–1148, May 2002.

[4] A. Herms, M. Schulze, J. Kaiser, and E. Nett, "Exploiting publish/subscribe communication in wireless mesh networks for industrial scenarios," in *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA '08)*, pp. 648–655, Hamburg, Germany, September 2008.

[5] F. Abdessemed, "An integrated system for tracking and control pilgrims shuttle buses," in *Proceedings of the 14th IEEE International Intelligent Transportation Systems Conference (ITSC '11)*, pp. 384–389, IEEE, Washington, DC, USA, October 2011.

[6] MakkahGIS, http://www.MakkahGIS.net.

[7] HajjHousing, http://www.hajjhousing.net.

[8] M. Grossmann, M. Bauer, N. Hönle, U.-P. Käppeler, D. Nicklas, and T. Schwarz, "Efficiently managing context information for large-scale scenarios," in *Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications (PerCom '05)*, pp. 331–340, March 2005.

[9] Y. Huang and H. Garcia-Molina, "Publish/subscribe in a mobile environment," *Wireless Networks*, vol. 10, no. 6, pp. 643–652, 2004.

[10] M. Castro, P. Druschel, A.-M. Kermarrec, and A. I. T. Rowstron, "Scribe: a large-scale and decentralized application-level multicast infrastructure," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1489–1499, 2002.

[11] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf, "Design and evaluation of a wide-area event notification service," *ACM Transactions on Computer Systems*, vol. 19, no. 3, pp. 332–383, 2001.

[12] W. W. Terpstra, S. Behnel, L. Fiege, A. Zeidler, and A. P. Buchmann, "A peer-to-peer approach to content-based publish/subscribe," in *Proceedings of the 2nd International Workshop on Distributed Event-Based Systems (DEBS '03)*, pp. 2–9, San Diego, Calif, USA, June 2003.

[13] P. R. Pietzuch and J. M. Bacon, "Hermes: a distributed event-based middleware architecture," in *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops*, pp. 611–618, Vienna, Austria, 2002.

[14] OMG Data Distribution Service for Real-time Systems Specifications, 2012, http://www.omg.org/spec/.

[15] N. Wang, D. C. Schmidt, H. van't Hag, and A. Corsaro, "Toward an adaptive data distribution service for dynamic large-scale network-centric operation and warfare (NCOW) systems," in *Proceedings of the IEEE Military Communications Conference (MILCOM '08)*, pp. 1–7, November 2008.

[16] A. Corradi, M. Fanelli, and L. Foschini, "Adaptive context data distribution with guaranteed quality for mobile environments,"

in *Proceedings of the IEEE 5th International Symposium on Wireless Pervasive Computing (ISWPC '10)*, pp. 373–380, May 2010.

[17] G. Chen, M. Li, and D. Kotz, "Data-centric middleware for context-aware pervasive computing," *Pervasive and Mobile Computing*, vol. 4, no. 2, pp. 216–253, 2008.

[18] A. Corradi, L. Foschini, and L. Nardelli, "A DDS-compliant infrastructure for fault-tolerant and scalable data dissemination," in *Proceedings of the 15th IEEE Symposium on Computers and Communications (ISCC '10)*, pp. 489–495, June 2010.

[19] K.-J. Kwon, C.-B. Park, and H. Choi, "A proxy-based approach for mobility support in the DDS system," in *Proceedings of the 6th IEEE International Conference on Industrial Informatics*, pp. 1200–1205, Daejeon, Republic of Korea, July 2008.

[20] D. Lincoln, V. Rafael, A. Lucas, A. Rafael, B. Gustavo, and E. Markus, "A large-scale communication middleware for fleet tracking and management," in *Proceedings of the 30th Brazilian Symposium on Computer Networks and Distributed Systems*, pp. 964–971, May 2012.

[21] L. David, R. Vasconcelos, L. Alves, R. André, and M. Endler, "A DDS-based middleware for scalable tracking, communication and collaboration of mobile nodes," *Journal of Internet Services and Applications*, vol. 4, no. 1, pp. 1–15, 2013.

[22] J. Y. J. Yu and S. L. S. Lai, "Message publish/subscribe system of RFID middleware," in *Proceedings of the International Conference on New Trends in Information and Service Science (NISS '09)*, pp. 288–292, Beijing, China, June-July 2009.

[23] B. Almadani, S. Khan, T. R. Sheltami, E. M. Shakshuki, M. Musaddiq, and B. Saeed, "Automatic vehicle location and monitoring system based on data distribution service," *Procedia Computer Science*, vol. 37, pp. 127–134, 2014.

Hindawi Publishing Corporation
Mobile Information Systems
Volume 2015, Article ID 153232, 16 pages
http://dx.doi.org/10.1155/2015/153232

Research Article

QPRD: QoS-Aware Peering Routing Protocol for Delay-Sensitive Data in Hospital Body Area Network

Zahoor A. Khan,[1] Shyamala Sivakumar,[2] William Phillips,[3] Bill Robertson,[3] and Nadeem Javaid[4]

[1] *CIS, Higher Colleges of Technology, Fujairah Campus, P.O. Box 4114, Fujairah, UAE*
[2] *Saint Mary's University, Halifax, NS, Canada B3H 3C3*
[3] *Internetworking Program, FE, Dalhousie University, Halifax, NS, Canada B3H 4R2*
[4] *COMSATS Institute of Information Technology, Islamabad 44000, Pakistan*

Correspondence should be addressed to Zahoor A. Khan; zahoor.khan@dal.ca

Received 10 November 2014; Accepted 15 January 2015

Academic Editor: Haroon Malik

Consistent performance, energy efficiency, and reliable transfer of data are critical factors for real-time monitoring of a patient's data, especially in a hospital environment. In this paper, a routing protocol is proposed by considering the QoS requirements of the Body Area Network (BAN) data packets. A mechanism for handling delay-sensitive packets is provided by this protocol. Moreover, linear programming based modeling along with graphical analysis is also done. Extensive simulations using the OMNeT++ based simulator Castalia 3.2 illustrate that the proposed algorithm provides better performance than other QoS-aware routing protocols in terms of higher successful transmission rates (throughputs), lower overall network traffic, no packets dropped due to MAC buffer overflow, and fewer numbers of packet time outs in both the mobile and static patient scenarios. The scalability of the protocol is demonstrated by simulating a 24-bed real hospital environment with 49 nodes. It is shown that, even in the larger real hospital scenario requiring the transmission of delay-sensitive data packets with stringent delay requirements, QPRD outperforms comparable protocols.

1. Introduction

A patient's real-time health-related data monitoring is possible with the help of a new emerging field, Body Area Networks (BANs). Body Area Network is a small wireless network which consists of sensors placed inside or outside of the human body. The body implant or wearable sensors transmit the data to a central device called Body Area Network Coordinator (BANC). BANC is computationally more powerful device then the body sensors. BANC is responsible for transferring the sensors' data to the next node or destination reliably.

Some important issues of BAN data transmission are to ensure the high reliability, low latency, compatibility with movable sensors, and low energy consumption. The specific need of BAN communication is not fulfilled by the existing Personal Area Network (PAN) standards [1]. IEEE task group 6 was assigned a job in November 2007 to suggest a BAN communication standard 802.15.6 by the consideration of short range transmission, reliability and latency requirements of QoS, and less energy consumption [2]. The real-time monitoring of patients requires the transmission of delay-sensitive data such as video imaging, motion sensing, and Electromyography (EMG) using BAN. Some projects like SMART [3], CareNet [4], AID-N [5], and ALARM-NET [6] provide different methods to monitor the patient data. In these methods, the transmission of BAN data from body sensors to the central database is considered and then BAN data is downloaded and monitored from the central database. However, these techniques do not monitor or display in real-time BAN data in hospital environment. The advantages of using a centralized system are to have better control and maintain the data privacy of the patient. However, traffic congestion, server failure, or link failure can cause considerable delays in monitoring the patient data which can badly affect treatment. On the other hand, distributed data approaches help to reduce the traffic load and can better accommodate patient mobility. The ZK-BAN peering framework proposed in [7] suggests a semicentralized system for reliably monitoring BAN data. The hybrid ZK-BAN uses both centralized and distributed techniques.

The routing protocols EPR, proposed and discussed in [7], resolves the problem of handling ordinary data packets. The QoS-aware peering routing protocol for reliability-sensitive data (QPRR) [8, 9] provides a mechanism of handling the reliability-sensitive packets in addition to the ordinary data packets. The requirement of real-time display for delay-sensitive packets is different from those of ordinary and reliability-sensitive packets. Hence, a new QoS-aware routing protocol is required to handle delay-sensitive packets. A novel routing protocol that addresses the issue of handling delay-sensitive data and displaying in real-time delay-sensitive BAN data is proposed in this paper. The proposed QoS-aware peering routing protocol for delay-sensitive packets (QPRD) is designed for the ZK-BAN peering framework discussed in [7]. QPRD provides an innovative approach to the reliable transmission of ordinary packets (OPs) and delay-sensitive packets (DSPs). The initial results and architecture of QPRD were presented in IEEE conference proceedings [10].

This paper is organized as follows: Section 2 provides the related work; Section 3 discusses the problem formulation and modeling; Section 4 provides the proposed QoS-aware peering routing protocol for delay-sensitive data (QPRD); Section 5 describes the performance evaluation; Section 6 discusses the scalability test of QPRD and Section 7 presents the conclusions.

2. Related Work

A smart monitoring system of BAN data in hospital environment can resolve the challenges related to the management of patients' medical information [11]. The Scalable Medical Alert and Response Technology (SMART) [3] is designed to monitor the patient's data in hospital emergency area. The data from sensors is transferred to the PDA and then the PDA sends it to the next tier by using wireless standard 802.11b. CareNet [4] provides an integrated wireless sensor based solution to monitor the patient's data from remote hospitals. The two-tier wireless communication is used in the projects [3, 4]. A GPS system is used in [5] to monitor the patient's data only in outdoor BAN communication. A wireless sensor network for assisted-living and residential monitoring system with a query based protocol is provided in ALARM-NET [6]. A three-tier communication approach is used in [12] to store the BAN data on the server and then make this data available for the physician to analyze the patient's data. The projects [3–6, 12] used a centralized approach to monitor the patient's data. However, the real-time display of data by considering the delay requirements of delay-sensitive packets is not considered. To access the data from a centralized server may cause delay and even a simple link failure can completely disconnect the healthcare system from the central server.

In [7], an energy-aware peering routing protocol (EPR) was presented which considers the energy level and geographic information of the neighbor nodes for choosing the best next hop. The EPR only considers ordinary packets. It was shown that EPR has an overall lower energy consumption than comparable protocols [11, 13–16] and provides better results in terms of reduced overall network traffic, reduced number of packets forwarded by intermediate nodes, and higher successful data transmission rates. However, EPR does not provide a mechanism for dealing with delay-sensitive packets (DSPs). In this paper, delay-sensitive packets are considered by the proposed QoS-aware peering routing protocol for delay-sensitive data (QPRD) and their performance is investigated by comparing it to the existing DMQoS protocol [13]. In [13], DMQoS categorizes the data packets into four types: ordinary packets (OPs), critical packets (CPs), reliability-driven packets (RPs), and delay-driven packets (DPs). The DMQoS [13] provides better results for delay-driven packets than several previously investigated methods [11, 14–16] in terms of end-to-end path delay. However, DMQoS employs a *hop-by-hop* approach to determine the next hop. DMQoS considers the neighbor device with the lowest delay, and the next hop then determines the best next upstream hop with least delay. The disadvantage of this *hop-by-hop* delay-driven approach employed in DMQoS is that only neighboring nodes delay information is considered by source node. The source node forwards the packet to a particular neighbor node which has lower node delay than the required delay. The neighbor node sends the acknowledgement of the successfully received packet to the source node. Now, the packet receiving neighbor node determines its best upstream node in terms of delay requirement and forwards the packet to the upstream node if the node delay of upstream node is less than the required delay. In case the neighbor node does not find any upstream node with node delay less than required delay, then the packet is dropped. In this case, the packet does not reach the destination, but the source node assumes that the packet has been successfully received by the destination. Furthermore, the *hop-by-hop* approach used in DMQoS causes an increase in overall network traffic, and the required end-to-end latency may not be guaranteed. In this paper, the proposed QPRD addresses these shortcomings by selecting and choosing the next hop device based on the lowest end-to-end path delay from the source node to the destination.

3. Problem Formulation and Modeling

The motivation that BAN consists of nodes connected with each other via wireless links leads us to model it as a directed graph. This section focuses on two points: (i) to maximize throughput, and (ii) to minimize the end-to-end delay. These two problems are modeled via linear programming [17, 18] because requirements to these problems could easily be represented by linear relationships.

3.1. Throughput Maximization. We consider BAN as a directed graph $G = (S, L)$, $|S| = s$, and $|L| = l$; S is the set of nodes and L is the set of directed graphs (links). If the network operations are divided into rounds, each round r is the duration from the network establishment till the death of all nodes; then linear programming based mathematical formulation for throughput maximization is as follows:

$$\text{Max} \sum_r T(r), \quad \forall r \in R, \tag{1}$$

where

$$T(r) = \sum_i l_{(i,\text{Dst})} \cdot T_{(i,\text{Dst})} \quad \forall i \in S, \tag{2}$$

$$l_{(i,\text{Dst})} = \begin{cases} 1 & \text{if packet delivery is guaranteed} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

such that

$$B \leq B_{\max}, \tag{4a}$$

$$\sum_i E_i \leq E_0, \quad \forall i \in S, \tag{4b}$$

$$\sum_i \text{DL}_{\text{path}(i,\text{Dst})} \leq \sum_i t_{\text{out}(i,\text{Dst})}, \quad \forall i \in S, \tag{4c}$$

$$\sum_i d_{(i,\text{Dst})} \leq \sum_i R_i^{tx}, \quad \forall i \in S, \tag{4d}$$

$$\sum_i f_{(i-1,i)} + \sum_i \lambda_i t \leq \sum_i f_{(i,\text{Dst})}, \quad \forall i \in S. \tag{4e}$$

The objective function in (1) aims to maximize throughput T during each round r such that (2) associates packet delivery from source i to destination Dst with link flag l. Equation (3) provides details about the status of l being raised ($l = 1$) if packet delivery through that link is guaranteed else not ($l = 0$). Constraint in (4a) provides the upper bound for the allocated bandwidth B as $B_{\max}$. Similarly, constraint in (4b) deals with limited energy constraint; that is, each node i is equipped with an energy source E_i such that $\sum_i E_i$ is upper bounded by E_0. Node ceases transmission whenever its battery is drained out, so, energy efficient utilization is very important (routing and MAC layer protocols play a critical role here). Constraint in (4c) comes into consideration if and only if $\exists(m/P) \in P_{\text{QoS}}$; m path(s) out of total P satisfies the given quality of service P_{QoS} where DL_{path} is the end-to-end delay and t_{out} is the timeout period. This means that, as a first priority, QoS needs to be satisfied. Afterwards, if there is more than one QoS path, then as a second priority end-to-end delay is checked. Transmission range R^{tx} constraint in (4d) demonstrates that packet delivery is successful if a source node transmits data to an in-range destination node where $d_{(i,\text{Dst})}$ is the distance between source and destination. For data generation rate λ, (4e) constraint entails flow conservation such that the incoming data flow $f_{(i-1,i)}$ plus the data generated during time t should not exceed the outgoing data flow $f_{(i,\text{Dst})}$. Violation of (4c), (4d), and (4e) leads to packets being dropped which ultimately results in decreased throughput.

3.2. Delay Minimization. The delay minimization problem, while routing dynamically such that path for each request is selected to prevent routing latency for future demands, is addressed here. The linear programming problem is formulated as follows:

$$\text{Min} \sum_i \text{DL}_{\text{path}(i,\text{Dst})}, \quad \forall i \in S, \tag{5}$$

where

$$\text{DL}_{\text{path}(i,\text{Dst})} = \begin{cases} \text{DL}_{\text{node}(i)} + \text{DL}_{\text{path}(j,\text{Dst})} & j \neq \text{Dst} \\ \text{DL}_{\text{node}(i)} & j = \text{Dst}, \end{cases} \tag{6}$$

$$\text{DL}_{\text{node}(i)} = \text{DL}_{\text{trans}(i)} + \text{DL}_{\text{queue+channel}} + \text{DL}_{\text{proc}}, \quad \forall i \in S \tag{7}$$

such that

$$l_{\text{path}(i,\text{Dst})} = 1, \quad \forall i \in S, \tag{8a}$$

$$R_{\text{bit}} \leq R_{\text{bit}}^{\max}, \tag{8b}$$

$$\sum_{t=1}^{t_{\max}} n \leq n_{\text{cap}}, \tag{8c}$$

$$\sum_i \lambda_i^{\text{arrival}} < \sum_i \lambda_i^{\text{departure}}, \quad \forall i \in S, \tag{8d}$$

$$0 \leq \sum_i \text{node}(i) \leq s, \quad \forall i \in S, \tag{8e}$$

$$N_{\text{bit}}^{e} \leq N_{\text{bit}}^{\text{proc}}. \tag{8f}$$

The objective function in (5) aims to minimize the end-to-end path delay $\text{DL}_{\text{path}(i,\text{Dst})}$, where (6) depicts the two possible cases: communication via intermediate node and without intermediate node, and (7) defines the node delay $\text{DL}_{\text{node}(i)}$ calculated at the network layer as the addition of packet delays due to transmission $\text{DL}_{\text{trans}(i)}$, queuing DL_{queue}, channel capturing $\text{DL}_{\text{channel}}$, and processing DL_{proc}. Constraint (8a) clearly says that the link through which data is routed must be established where l_{path} is the link flag. Constraint in (8b) provides the upper bound of data rate R_{bit} as $R_{\text{bit}}^{\max}$ such that R_{bit} is inversely proportional to DL_{trans} according to (15) explained in later Section 4.4. Constraint in (8c) says that the number of transmitted packets n in 4 seconds ($t_{\max} = 4$ sec) should not exceed the packet handling capacity n_{cap} because, at negligible load, there is constant small delay. However, queuing delays on each node are added as soon as the network load increases and when n exceeds n_{cap} delays increase without bound. Similarly, constraint in (8d) deals with queue stability meaning that the packet arrival rate $\lambda_i^{\text{arrival}}$ should be always less than the packet departure rate $\lambda_i^{\text{departure}}$. Violation of (8d) results in nonavailability of buffer space leading to queue instability which in turn leads to congestion and thus causing increased delay. Constraint in (8e) states that the total number of nodes $\sum_i \text{node}(i)$ in the given network is limited such that i has an inverse relation with $\text{DL}_{\text{channel}}$. In other words, increasing the number of nodes means that there are more chances that one of the noncapturing nodes might have a lower backoff time as compared to that of the capturing node, thereby increasing the idle time due the backing off of noncapturing nodes. Constraint in (8f) deals with $\mathrm{DL}_{\mathrm{proc}}$. It is obvious that the total bit level errors N_{bit}^{e} should not exceed a certain threshold $N_{\mathrm{bit}}^{\mathrm{proc}}$; otherwise increased erroneous bits would increase $\mathrm{DL}_{\mathrm{proc}}$ and performance of the network would degrade in terms of processing delays at the nodes.

3.3. Graphical Analysis. Consider the simplified path B_3-B_1-NSC where B_3 can directly send data to NSC as well as via B_1 (refer to Figure 3 in Section 4.4). Moreover, NSC can also send data to itself. For typically assumed delay values, path delay is maximum when B_1 is involved in forwarding the data of B_3 intended for the destination NSC (i.e., 50 ms), and path delay is minimum when B_3 directly communicates with NSC (i.e., 20 ms). Let $z = \mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})}$, $x = \mathrm{DL}_{\mathrm{node}(i)}$, and $y = \mathrm{DL}_{\mathrm{path}(j,\mathrm{Dst})}$. The objective function in (5) can be reformulated as

$$\mathrm{Min}\,(z), \tag{9}$$

where

$$z = x + y \tag{10}$$

such that

$$x + y \geq 20, \tag{11a}$$

$$x + y \leq 50, \tag{11b}$$

$$\begin{gathered} 0 \leq x \leq 20, \\ 0 \leq y \leq 30. \end{gathered} \tag{11c}$$

The objective function in (9) aims to minimize end-to-end delay regarding the selected path, whereas (10) illustrates nature of the objective function, that is, two-dimensional linear programming problem. Constraints in (11a) and (11b) provide lower and upper bounds for the selected path, respectively, whereas constraint in (11c) deals with similar bounds for x as well as y. For simplicity in calculation, we replace the inequalities in (11a), (11b), and (11c) with equalities, respectively. In subject to the given constraints, Figure 1 shows the set of feasible solutions which is obtained by the intersection of lines, L_1, L_2, L_3, and L_4, and is indicated by coloured region such that each point in the feasible region satisfies each constraint. We can find the minimum value of z by testing it at each of the vertices (refer to $P1$, $P2$, $P3$, and $P4$ in Figure 1) as follows:

$$\begin{aligned} &\text{at } P1(0,0)\text{: } z = 0\,\mathrm{ms}, \\ &\text{at } P2(20,0)\text{: } z = 20 + 0 = 20\,\mathrm{ms}, \\ &\text{at } P3(0,30)\text{: } z = 0 + 30 = 30\,\mathrm{ms}, \\ &\text{at } P4(20,30)\text{: } z = 20 + 30 = 50\,\mathrm{ms}. \end{aligned}$$

The minimum value of z is 0 ms at $x = 0$ and $y = 0$. However, this value indicates self transmission or communication within NSC. The next minimum value is $z = 20$ ms showing the case of direct communication between B_3 and NSC. Similarly, $z = 30$ ms when B_1 directly communicates with NSC. On the other hand, end-to-end path delay is maximum when B_3 communicates with NSC via an intermediate B_1; $z = 50$ ms.

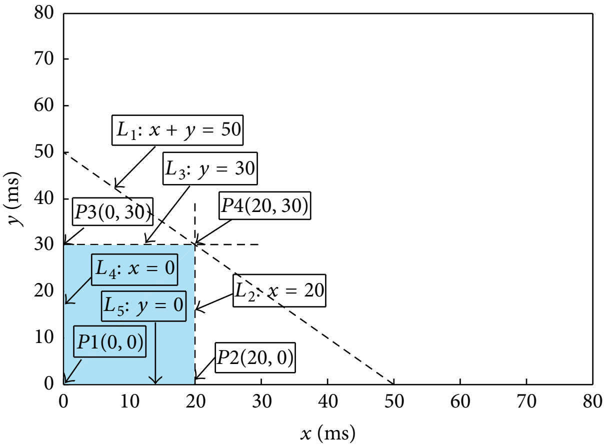


Figure 1: Feasible region.

4. QoS-Aware Peering Routing Protocol for Delay-Sensitive Data (QPRD)

Based on the mathematical analysis in Section 3, the proposed QoS-aware routing protocol used in an indoor hospital ZK-BAN peering framework [7] is discussed in this section. The proposed QPRD provides a mechanism to (1) calculate the node delays and path delays of all possible paths from the source node to the destination, (2) determine the best path, and (3) choose the best next hop NH_D based on the delay requirements of the packet. For each destination, the routing table contains information about the next hop device connected to the path with the least end-to-end latency. For any DSP, if the path delay ($\mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})}$) is less than or equal to the delay requirement, the source node sends the DSP through that path.

The architecture of proposed QPRD, based on the mathematical formulation of the end-to-end path delay problem, is shown in Figure 2. It consists of seven modules: MAC receiver, delay module (DM), packet classifier (PC), Hello protocol module (HPM), routing services module (RSM), QoS-aware queuing module (QQM), and MAC transmitter. The modules are discussed below.

4.1. MAC Receiver. The MAC receiver receives the data or Hello packets from other nodes (BAN, MDC, or NSC). It checks the MAC address of the packet. It only forwards the broadcast packets or the packets which have the same node's MAC address as destination address to the network layer.

4.2. Delay Module (DM). The delay module monitors the time required to capture the channel ($\mathrm{DL}_{\mathrm{channel}(i)}$), MAC layer queuing delay ($\mathrm{DL}_{\mathrm{MAC\_queue}(i)}$), and transmission time

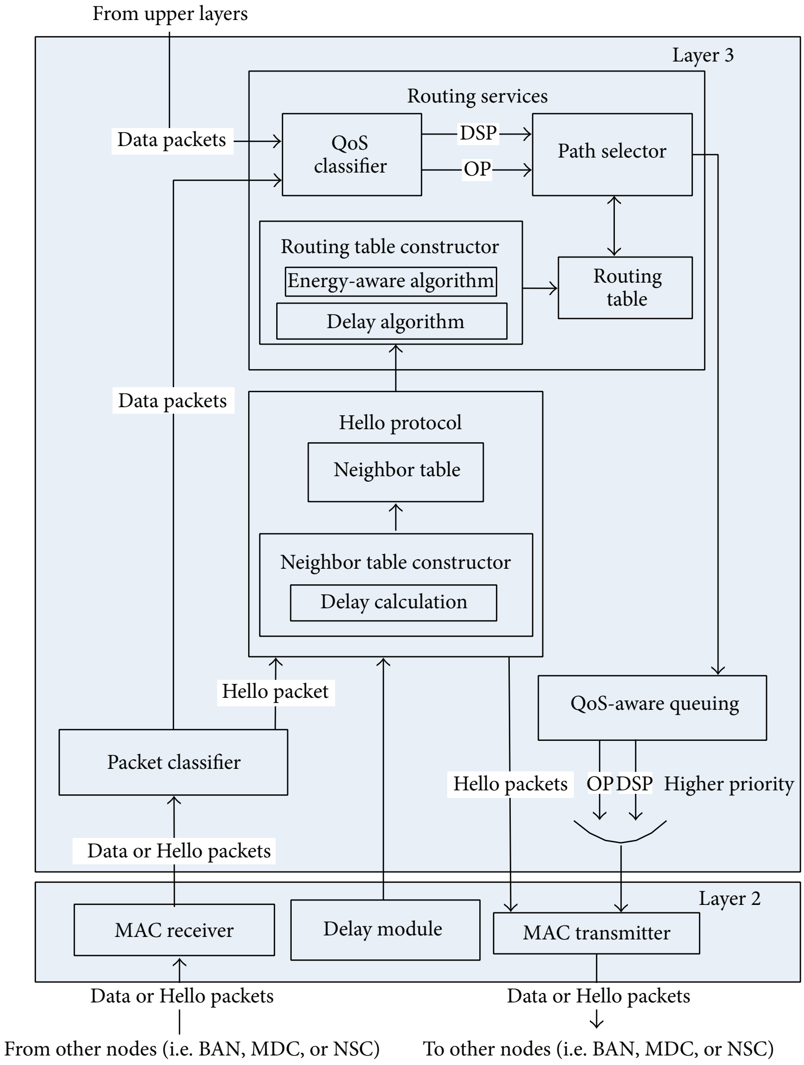


Figure 2: QPRD protocol architecture.

($\mathrm{DL}_{\mathrm{trans}(i)}$) of a packet. The delay module sends this information to the network layer. The network layer uses this information to calculate the node delay ($\mathrm{DL}_{\mathrm{node}(i)}$).

4.3. Packet Classifier (PC). The packet classifier (PC) receives all the packets from the MAC receiver. The data packets and Hello packets are differentiated by the PC. The PC forwards the data and Hello packets to the routing services module and Hello protocol module, respectively.

4.4. Hello Protocol Module (HPM). The neighbor table constructor and the neighbor table are the two submodules of Hello protocol module. The information received from the delay module of the MAC layer and the Hello packets is used by the neighbor table constructor to construct the neighbor table. Initially, Hello packets are broadcasted by each of type 1 (NSC) and type 2 (MDC) devices. The node i receives the Hello packet. The neighbor table constructor of node i calculates its own $\mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})}$ based on the information in the Hello packets. The Hello packet is updated and forwarded by node i to the other nodes. The Hello packet fields of node j are shown as follows.

Hello Packet Structure. Consider

$$\boxed{\mathrm{ID}_{\mathrm{Dst}} \mid L_{\mathrm{Dst}} \mid \mathrm{ID}_j \mid L_j \mid D_{(j,\mathrm{Dst})} \mid E_j \mid T_j \mid \mathrm{DL}_{\mathrm{path}(j,\mathrm{Dst})}} \quad (12)$$

The commonly used notations in this paper and their descriptions are summarized in notations section.

The neighbor table contains fields for both hop-by-hop delay ($\mathrm{DL}_{\mathrm{node}(i)}$) and end-to-end path delay ($\mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})}$). The neighbor table constructor updates the neighbor table periodically after receiving every new Hello packet. The neighbor table structure of node i is shown as follows.

The Neighbor Table Structure. Consider

$$\boxed{\mathrm{ID}_{\mathrm{Dst}} \mid L_{\mathrm{Dst}} \mid \mathrm{ID}_j \mid L_j \mid D_{(j,\mathrm{Dst})} \mid D_{(i,j)} \mid C_j \mid T_j \mid \mathrm{DL}_{\mathrm{node}(i)} \mid \mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})}} \quad (13)$$

The node delay ($\mathrm{DL}_{\mathrm{node}(i)}$) can be found by adding the packet delays due to transmission, queuing, processing, and channel capturing:

$$\mathrm{DL}_{\mathrm{node}(i)} = \mathrm{DL}_{\mathrm{trans}(i)} + \mathrm{DL}_{\mathrm{queues+channel}} + \mathrm{DL}_{\mathrm{proc}}. \quad (14)$$

The node updates its Hello packets periodically; 4 seconds are used in QPRD for simulation purposes. The time interval 4 seconds is used because the delay module sends the delays of MAC queue and channel capture after every 4 seconds. The average transmission delay ($\mathrm{DL}_{\mathrm{trans}}$) before sending the Hello packets is calculated by using

$$\mathrm{DL}_{\mathrm{trans}} = \frac{1}{R_{\mathrm{bit}}} \frac{\sum_{z=1}^{n} N_{\mathrm{bit}(z)}}{n}, \quad (15)$$

where R_{bit} is the data rate and as per BAN requirement 250 kbps is used in the simulations. N_{bit} is the total number of bits in each packet. n is the number of packets transmitted in 4 seconds.

The delay due to the MAC and network layers' queues and capturing the channel can be calculated by using the Exponentially Weighted Moving Average (EWMA) formula and is given in

$$\begin{aligned}\mathrm{DL}_{\mathrm{queues+channel}} = {} & (1-\rho) * \mathrm{DL}_{\mathrm{queues+channel}} + \rho \\ & * \mathrm{DL}_{\mathrm{queues+channel}},\end{aligned} \quad (16)$$

where queues are both network and MAC layers' queues.

Initial values of $\mathrm{DL}_{\mathrm{queues+channel}}$ are the delay of the first packet sent by the node. ρ is the average weighting factor that satisfies $0 < \rho \leq 1$. The selection of ρ value is heuristic and was chosen based on simulations experience. The recommended values are $0.2 \leq \rho \leq 0.3$. The best suited value of ρ found for QPRD simulations is 0.2.

The path delay between node i and destination node Dst ($\mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})}$) is calculated by using

$$\mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})} = \mathrm{DL}_{\mathrm{node}(i)} + \mathrm{DL}_{\mathrm{path}(j,\mathrm{Dst})}, \quad (17)$$

where initial value of $\mathrm{DL}_{\mathrm{path}(j,\mathrm{Dst})}$ is zero when $j = \mathrm{Dst}$.

An example of finding the path delay from node i (B_3) to Dst (NSC) is shown in Figure 3. The delay calculation of two paths B_3-B_1-MDC_2-NSC (path 1) and B_3-MDC_3-B_2-MDC_1-NSC (path 2) is given for illustrative purposes. The typical assumed values are chosen for illustrated purposes. The individual node delays used in this example are given below:

$$\mathrm{DL}_{\mathrm{node(NSC)}} = 20\,\mathrm{ms}, \quad (18a)$$

$$\mathrm{DL}_{\mathrm{node}(\mathrm{MDC}_2)} = 40\,\mathrm{ms}, \quad (18b)$$

$$\mathrm{DL}_{\mathrm{node}(\mathrm{B}_1)} = 30\,\mathrm{ms}, \quad (18c)$$

$$\mathrm{DL}_{\mathrm{node}(\mathrm{B}_3)} = 20\,\mathrm{ms}, \quad (18d)$$

$$\mathrm{DL}_{\mathrm{node}(\mathrm{MDC}_1)} = 20\,\mathrm{ms}, \quad (18e)$$

$$\mathrm{DL}_{\mathrm{node}(\mathrm{B}_2)} = 30\,\mathrm{ms}, \quad (18f)$$

$$\mathrm{DL}_{\mathrm{node}(\mathrm{MDC}_3)} = 10\,\mathrm{ms}. \quad (18g)$$

The path delay of destination ($\mathrm{DL}_{\mathrm{path(Dst,Dst)}}$) is approximately zero, because the time required to receive the packet from MAC to network layer is negligible. So, in this example initial path delay is given below:

$$\mathrm{DL}_{\mathrm{path(NSC,NSC)}} = 0\,\mathrm{ms}. \quad (19)$$

Each node calculates the path delay from itself to the NSC. First, the calculations of the path delay for path 1 (B_3-B_1-MDC_2-NSC) are considered.

The path delay of MDC_2 ($\mathrm{DL}_{\mathrm{path}(\mathrm{MDC}_2,\mathrm{NSC})}$) is calculated by using (17):

$$\mathrm{DL}_{\mathrm{path}(\mathrm{MDC}_2,\mathrm{NSC})} = \mathrm{DL}_{\mathrm{node(MDC2)}} + \mathrm{DL}_{\mathrm{path(NSC,NSC)}}. \quad (20)$$

Using the values from (18a) and (19) in the above equation, we get

$$\mathrm{DL}_{\mathrm{path}(\mathrm{MDC}_2,\mathrm{NSC})} = 40 + 0 = 40\,\mathrm{ms}. \quad (21)$$

The path delay of BAN B_1 is calculated below:

$$\begin{aligned}\mathrm{DL}_{\mathrm{path}(\mathrm{B}_1,\mathrm{NSC})} &= \mathrm{DL}_{\mathrm{node}(\mathrm{B}_1)} + \mathrm{DL}_{\mathrm{path}(\mathrm{MDC}_2,\mathrm{NSC})}, \\ \mathrm{DL}_{\mathrm{path}(\mathrm{B}_1,\mathrm{NSC})} &= 30 + 40 = 70\,\mathrm{ms}.\end{aligned} \quad (22)$$

The node B_3 determines the path delay by using the values from (18d) and (22):

$$\begin{aligned}\mathrm{DL}_{\mathrm{path}(\mathrm{B}_3,\mathrm{NSC})} &= \mathrm{DL}_{\mathrm{node}(\mathrm{B}_3)} + \mathrm{DL}_{\mathrm{path}(\mathrm{B}_1,\mathrm{NSC})}, \\ \mathrm{DL}_{\mathrm{path}(\mathrm{B}_3,\mathrm{NSC})} &= 20 + 70 = 90\,\mathrm{ms}.\end{aligned} \quad (23)$$

In the same manner, the path delay of path 2 (B_3-MDC_3-B_2-MDC_1-NSC) can be calculated as follows:

$$\begin{aligned}\mathrm{DL}_{\mathrm{path}(\mathrm{MDC}_1,\mathrm{NSC})} &= 20 + 0 = 20\,\mathrm{ms}, \\ \mathrm{DL}_{\mathrm{path}(\mathrm{B}_2,\mathrm{NSC})} &= 30 + 20 = 50\,\mathrm{ms}, \\ \mathrm{DL}_{\mathrm{path}(\mathrm{MDC}_3,\mathrm{NSC})} &= 10 + 50 = 60\,\mathrm{ms}, \\ \mathrm{DL}_{\mathrm{path}(\mathrm{B}_3,\mathrm{NSC})} &= 20 + 60 = 80\,\mathrm{ms}.\end{aligned} \quad (24)$$

Equations (23) and (24) show that the path delays of path 1 and path 2 are 90 ms and 80 ms, respectively. It is quite possible that the path with less delay is longer (has more hops) than the other paths. As it is observed from the above example, path 2 includes five devices and path 1 has four devices. However, the path delay of path 2 is lower than the path delay of path 1.

4.5. Routing Services Module (RSM). The routing services module is responsible for constructing the routing table, categorizing the data packets into delay-sensitive packets (DSPs) and ordinary packets (OPs). It also chooses the best path(s) for each category (DSPs or OPs) of traffic. QoS classifier, routing table constructor, path selector, and routing table are the submodules of routing services module. The routing table structure for node i is shown as follows.

The Routing Table Structure for QPRD. Consider

$$\boxed{\mathrm{ID}_{\mathrm{Dst}} \mid L_{\mathrm{Dst}} \mid \mathrm{NH}_E \mid \mathrm{NH}_D \mid \mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})}} \quad (25)$$

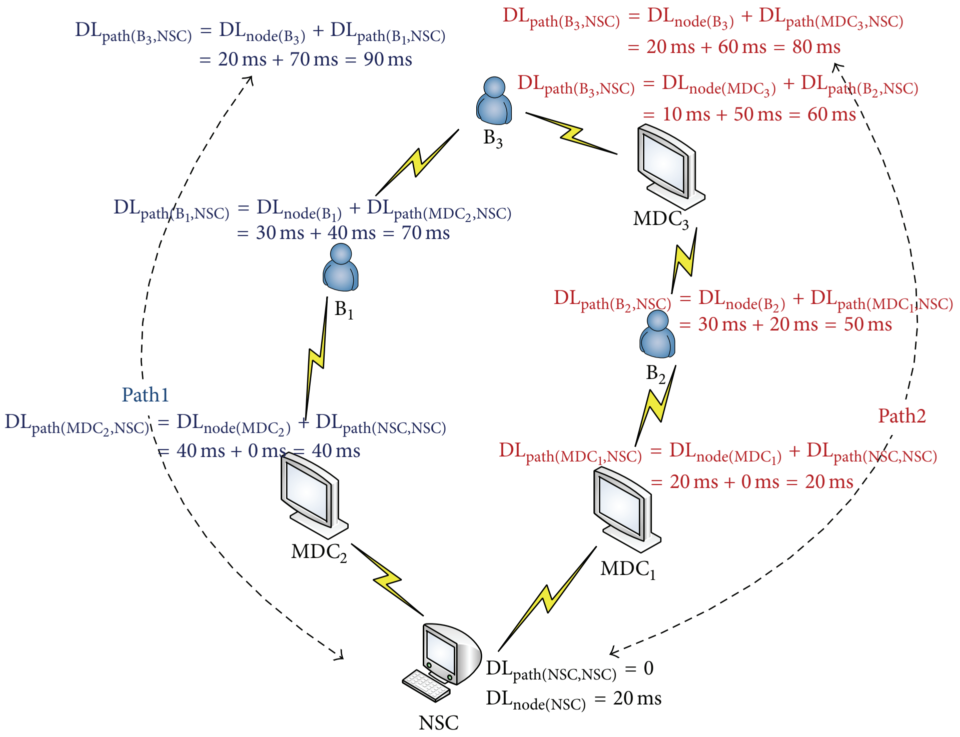

$DL_{path(B_3,NSC)} = DL_{node(B_3)} + DL_{path(B_1,NSC)}$
$= 20\,ms + 70\,ms = 90\,ms$
$DL_{path(B_3,NSC)} = DL_{node(B_3)} + DL_{path(MDC_3,NSC)}$
$= 20\,ms + 60\,ms = 80\,ms$
B_3
$DL_{path(B_3,NSC)} = DL_{node(MDC_3)} + DL_{path(B_2,NSC)}$
$= 10\,ms + 50\,ms = 60\,ms$
$DL_{path(B_1,NSC)} = DL_{node(B_1)} + DL_{path(MDC_2,NSC)}$
$= 30\,ms + 40\,ms = 70\,ms$
MDC_3
B_1
$DL_{path(B_2,NSC)} = DL_{node(B_2)} + DL_{path(MDC_1,NSC)}$
$= 30\,ms + 20\,ms = 50\,ms$
B_2
Path1
$DL_{path(MDC_2,NSC)} = DL_{node(MDC_2)} + DL_{path(NSC,NSC)}$
$= 40\,ms + 0\,ms = 40\,ms$
Path2
$DL_{path(MDC_1,NSC)} = DL_{node(MDC_1)} + DL_{path(NSC,NSC)}$
$= 20\,ms + 0\,ms = 20\,ms$
MDC_2
MDC_1
$DL_{path(NSC,NSC)} = 0$
$DL_{node(NSC)} = 20\,ms$
NSC

Figure 3: Example of finding the path delay.

The notations and their descriptions are listed in notation section. Two next hop entries NH_E and NH_D are given for each destination Dst in routing table. The routing table constructor contains the energy-aware and delay algorithms. The energy-aware algorithm discussed in [7] is used to find next hop NH_E for OPs. Residual energy and geographic location of the neighbor nodes are considered for choosing NH_E. For DSPs, the new proposed algorithm finds the best possible path to ensure the minimum required path delay. The routing table is constructed by using the neighbor table entries. Neighbor table contains multiple records for each destination. For example, Figure 3 shows that there are many paths from B_3 to NSC. Some of these paths are B_3-B_1-MDC_2-NSC, B_3-MDC_3-B_2-MDC_1-NSC, and so forth. For each destination, the routing table constructor stores the next hop (NH_D) which has the lowest latency.

Algorithm 1 shows that node i identifies the next hop candidates by searching the records which have the same ID_{Dst} in the neighbor table. The path delay has been calculated by using the neighbor table constructor and stored in neighbor table for each next hop candidate, using (17). The node stores the neighbor nodes' IDs in the variable NH (line 2). If NH has only one entry, this means there is only one path available. The node stores this entry to NH_D (line 4).

Otherwise, the node sorts the NH entries in ascending order of delay and then stores the first entry which has the lowest path delay in NH_D (lines 6-7). The next hop candidate NH_D is then stored with its path delay value ($DL_{path(i,Dst)}$) in the routing table. The data packets from both upper layers and packet classifier are received by QoS classifier. The QoS classifier classifies the packets into DSP and OP data. For each data packet, the path selector (PS) checks the QoS requirement and chooses the most appropriate next hop(s) by using Algorithm 2.

The path selector compares the delay requirement (DL_{req}) with the path delay ($DL_{path(i,Dst)}$) of NH_D which is stored in the routing table. If the path delay ($DL_{path(i,Dst)}$) is lower than required delay (DL_{req}), the packet is sent to NH_D (lines 3-4). Otherwise, the packet is dropped (line 6).

For ordinary packets, the PS returns the next hop NH_E which is discussed by the EPR (lines 8-9); else the packet is dropped.

4.6. QoS-Aware Queuing Module (QQM). The routing services module passes the data packets to the QoS-aware queuing module (QQM) after choosing the appropriate next hop(s). The QQM receives the data packets and separates these packets in two classes (DSP and OP). An individual queue is used for each class of packets. QQM functions are the same as discussed in [13]. The priority of the DSP queue is higher than that of the OP queue. By default, the DSP queue with higher priority sends the packets first. The packets from lower priority OP queue will be sent only when the DSP queue is empty. However, for fair treatment of OP data, a timeout is used by all the queues. A queue sends the packets to the MAC layer within the period specified by the timeout for that queue. QQM changes the control from higher priority queue to lower priority queue after the queue timeout occurs or when the higher priority queue is empty whichever is earlier.

4.7. MAC Transmitter. The MAC transmitter receives the data and Hello packets from the network layer and stores it in

```
INPUT: Neighbor table, i's neighbor table records $NH_{(i,Dst)}$, $\forall Dst \in \{MDC, NSC, BAN\}$
for each destination $Dst \in \{NSC, MDC, BAN\}$ do
  $NH = \{$All neighbor nodes $j \in NH_{(i,Dst)}\}$
  if $(|NH| == 1)$ then
    $NH_D \leftarrow NH$
    else if $(|NH| > 1)$ then
    Sort NH in ascending order of $DL_{path(i,Dst)}$
    $NH_D$ = first neighbor node $j \in NH$;
  end if
end for
```

Algorithm 1: Routing table construction algorithm for delay-sensitive packets.

```
INPUT: Routing table, i's routing table records $NH_{(i,Dst)}$, $\forall Dst \in \{MDC, NSC, BAN\}$
for each data packet do
  if data packet is delay-sensitive packet (DSP)
    if $(DL_{path(i,Dst)} \le DL_{req})$ then
     send to $NH_D$
    else
        Drop the packet immediately
    end if
  else if data packet is Ordinary Packet (OP)
        send to $NH_E$
          else
        drop the packet immediately
  end if
end for
```

Algorithm 2: Path selector algorithm for delay-sensitive packets.

the queue. The queue works in a First-In-First-Out (FIFO) fashion. It transmits the packets after capturing the channel by using CSMA/CA algorithm.

5. Performance Evaluation

Simulations are performed on OMNeT++ based simulator Castalia 3.2 [19]. In this section, the proposed QPRD algorithm is compared with the DMQoS [13] and noRouting protocols. In noRouting, the delay-sensitive data packets are forwarded to random next hop devices instead of algorithm's next hop based on end-to-end path delay routes. The network parameters used in simulations are shown in Table 1.

Three scenarios are considered for simulation. All the nodes used in scenario 1 are static, whereas the source node B_4 is moving in scenario 2. Scenario 3 is used for the scalability test of the protocol. The transmit power used in the simulations is −25 dBm. The performance of the QPRD is measured by calculating the throughput, number of packets forwarded by the intermediate nodes, overall network traffic, packets timeout due to not fulfilling the required delay condition, and packets dropped due to the buffer overflow. The better results provided by QPRD are in accordance with the equations used in Section 4. The higher throughput is due to the use of objective function in QPRD, as described in (1), and the least violations of (4c), (4d), and (4e). The simulation results show that the end-to-end path delay mechanism, as discussed in Sections 3.2 and 4.4, used in QPRD helps to reduce the packets forwarded by intermediate nodes and the packets dropped due to the buffer overflow, which results in higher throughput and lower overall network traffic. To achieve a 97% confidence interval for the illustrative results, the average of three runs is simulated in every experiment which may introduce a maximum error of 3×10^{-3}, based on the error calculation done by Castalia 3.2 simulator [20]. The results obtained for first two scenarios are discussed below.

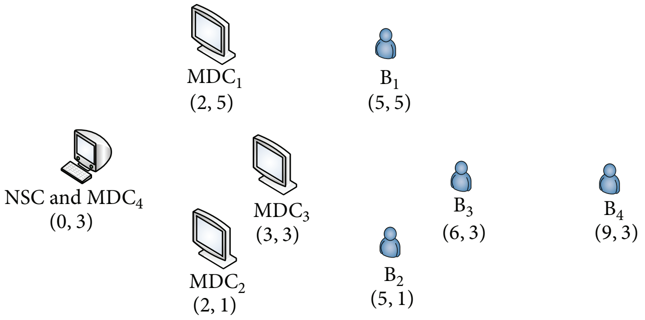
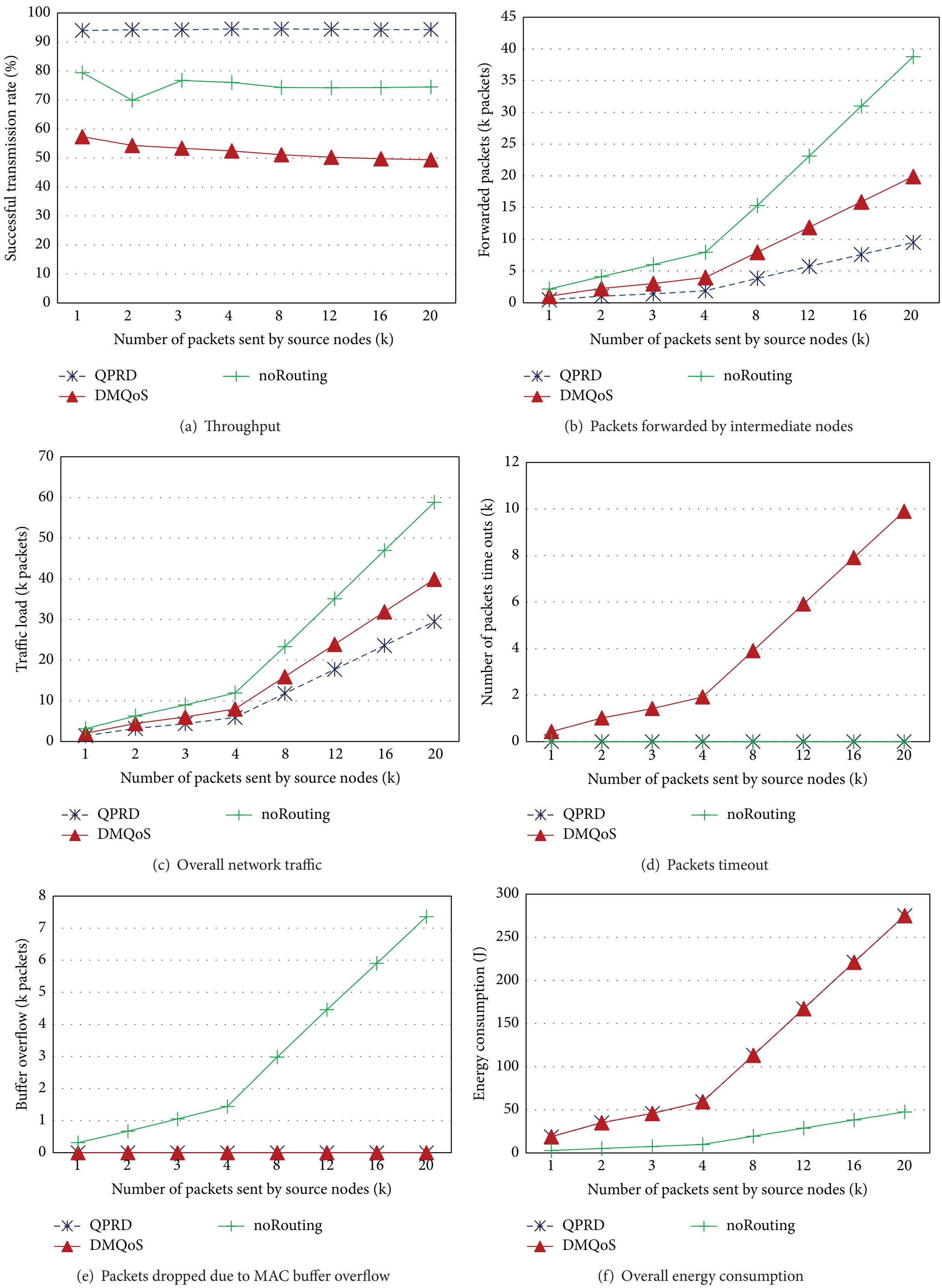


Figure 4: Node deployment for scenario 1.

5.1. Scenario 1: Static Nodes. Figure 4 shows the deployment of the experimental network for scenario 1. All the nodes are



(a) Throughput

(b) Packets forwarded by intermediate nodes

(c) Overall network traffic

(d) Packets timeout

(e) Packets dropped due to MAC buffer overflow

(f) Overall energy consumption

Figure 5: Performance comparison for different parameters when source nodes are static.

Table 1: Parameters information.

Deployment	Area	Scenarios 1 and 2: 9 m by 9 m Scenario 3: 21 m × 16 m
	Deployment type	Scenario 1: all nodes are static Scenario 2: movable source node B_4 Scenario 3: hospital environment
	Number of nodes	Scenarios 1 and 2: 8 nodes (4 BANs, 3 MDCs, 1 NSC) Scenario 3: 49 nodes (24 BANs, 24 MDCs, 1 NSC)
	Initial nodes locations	Scenarios 1 and 2: NSC(0,3), MDC_1(2,5), MDC_2(2,1), MDC_3(3,3) B_1(5,5), B_2(5,1), B_3(6,3), B_4(9,3) Scenario 3: shown in Figure 8
	Initial node energy	18720 J (= 2 AA batteries)
	Buffer size	32 packets
	Link layer trans. rate	250 Kbps
	Transmit power	−25 dBm
Task	Application type	Event-driven
	Max. packet size	32 bytes
	Traffic type	CBR (Constant Bit Rate)
MAC	IEEE 802.15.4	Default values
Simulation	Time	2003 seconds (3 seconds are setup time. Simulation results are the average of three rotations.)

static in this scenario. The type 1 devices (BANCs: B_1, B_2, B_3, and B_4) are considered as source nodes, and type 2 devices (NSC and MDCs) are the destination nodes. B_1 sends packets to MDC_1, B_2 sends packets to MDC_2, B_3 sends packets to MDC_3, and B_4 sends packets to NSC. The data of B_4 has to go through the other devices to reach NSC. The source nodes send a total of 20 k delay-sensitive packets. The throughput, packets forwarded by intermediate nodes, overall network traffic, number of packets timeout, packets dropped due to MAC buffer overflow, and overall energy consumption are calculated after every 1000 packets until 4 k and then every 4000 packets sent by all BANCs.

From Figure 5(a), it is seen that QPRD consistently provides throughput of 94% or more. In comparison, noRouting provides an average of 74% transmission rate, whereas DMQoS has a throughput ranging from 49% to 57%. For low offered data loads of 1 k, DMQoS has a throughput of 57% that continues to decrease especially for high offered data loads of 20 k, when the throughput is 49%. The low throughput in DMQoS may be explained by the way it selects the next hop using the energy-aware geographic forwarding scheme. Because the best next hop does not guarantee that it has the smallest latency connection to the destination, the packet may timeout when it is sent using the "best" next hop. Moreover, the energy-aware geographic forwarding scheme used in DMQoS prefers the nearest next hop candidate in terms of hop count and ignores next hop nodes having a lower delay. As a result, the network traffic is increased and the packets are dropped due to timeout before reaching the destination. QPRD resolves these issues by using the end-to-end path delay.

B_2 is the closest node to the destination nodes (i.e., NSC or MDCs) as shown in Figure 4. In DMQoS [13], B_2 is responsible for forwarding the data packets from other nodes to NSC or MDCs. This results in more energy consumption for B_2 and increased traffic congestion experienced by B_2. EPR resolves these problems by choosing the most appropriate next hop. In the proposed QPRD scheme, the BAN coordinator does not send data to another BAN coordinator unless it is absolutely necessary. Figure 5(b) shows the number of packets forwarded by the intermediate nodes. It is seen from Figure 5(b) that number of data packets forwarded by intermediate nodes before reaching the destinations in QPRD are on average 0.5 times and 3 times lower than DMQoS and noRouting, respectively.

The lower number of forwarded packets by intermediate nodes helps to reduce the overall network traffic. Figure 5(c) shows the total network traffic generated by QPRD, DMQoS, and noRouting as a function of the offered traffic load. From this Figure, it is seen that QPRD generates about an average of 26% and 99% less traffic in the network compared to DMQoS and noRouting, respectively. The path calculation in QPRD considers the delay of all the nodes and uses the best path delay information to select the next hop to send the data from source to destination.

In contrast to the method used in DMQoS which decides on the immediate next hop based merely on next hop delay instead of overall path delay, each upstream hop in DMQoS sends the packet to its next hop and resultant path in DMQoS may not be the most optimal.

From Figure 5(d) it is observed that QPRD and noRouting have no packets that were timed out for all offered

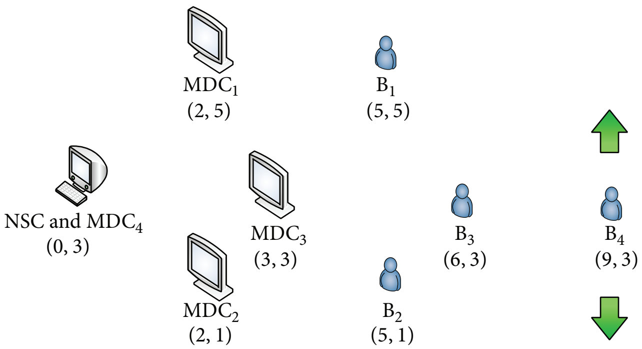


Figure 6: Node deployment for scenario 2.

traffic loads (number of data packets sent by source node range from 0 k to 20 k). QPRD has better performance in terms of reduced overall network traffic and fewer numbers of dropped packets due to timeout, because the clear end-to-end path delay information helps the packet to reach the destination within the requested delay requirement. Moreover, the path calculation in QPRD considers the delay of all the nodes in the network and chooses only those paths which can guarantee delivering the packet to the destination before it times out.

Figure 5(e) shows that there is no packet dropped due to the MAC buffer overflow in QPRD protocol. This is due to no violation of the model constraints as explained in Sections 4.1 and 4.2. The source node chooses the path which provides the maximum throughput and minimum end-to-end path delay as described in [16, 18]. Only few packets are dropped in DMQoS, whereas 7.5 k packets are dropped in noRouting.

It is seen from Figure 5(f) that the end-to-end path delay mechanism used in QPRD does not affect the overall energy consumption when compared with DMQoS. QPRD and DMQoS consume the same 18 Joules to 275 Joules of energy when 1 k to 20 k packets are sent by source nodes. On the other hand, the energy consumption of noRouting protocol is 2.6 Joules to 47.7 Joules when 1 k to 20 k packets are sent by source nodes. The data packets in noRouting are randomly forwarded to three neighbor nodes without considering the delay requirements. The additional computations for delay in QPRD consume on average 6 times more energy than noRouting. However, it must be noted that noRouting results in on average a 99% higher overall network traffic. This may be attributed to the 3 times more packets forwarded by intermediate nodes in noRouting resulting in a 20% lower throughput as compared to QPRD.

In summary, QPRD outperforms DMQoS and noRouting when the source node is static.

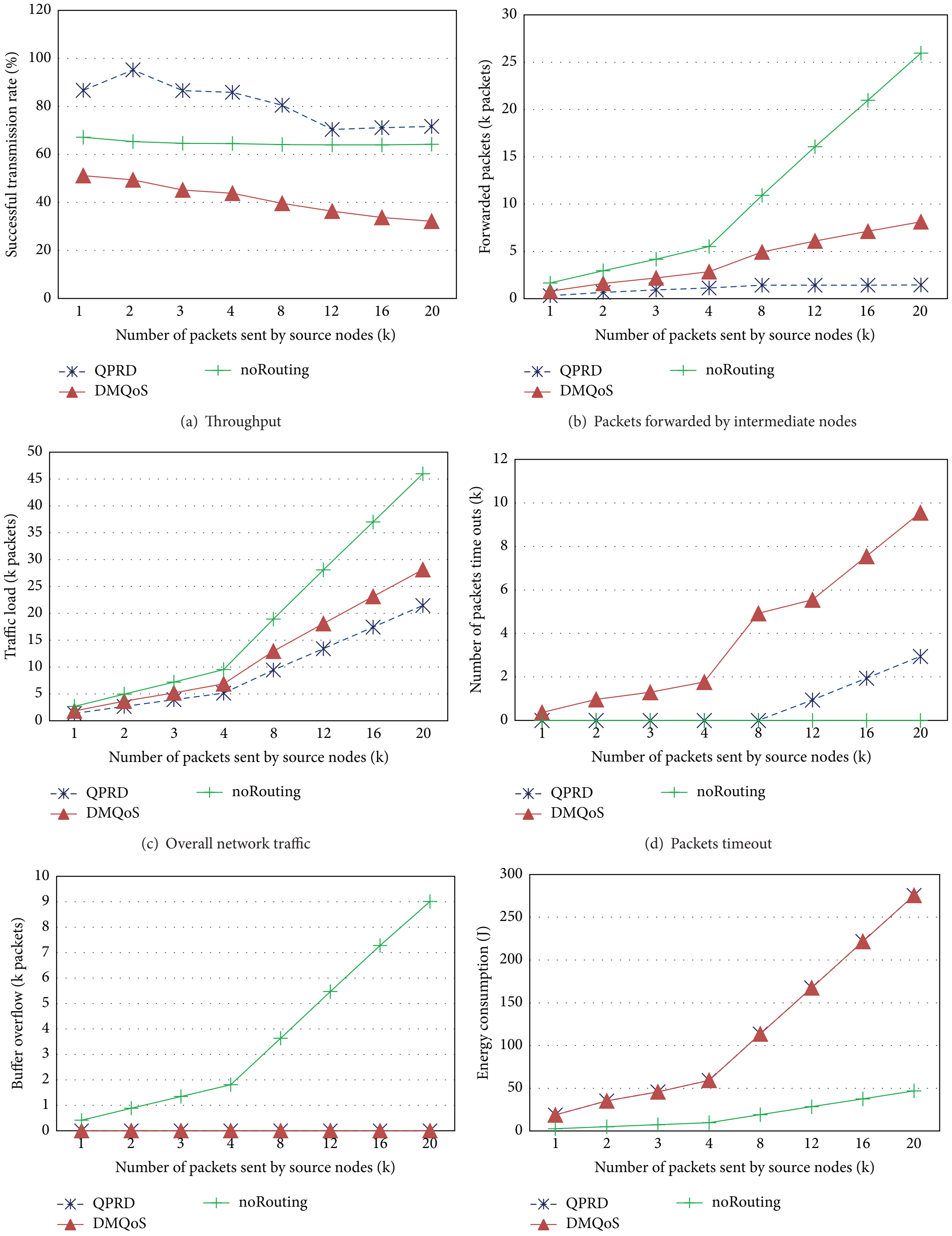
5.2. Scenario 2: Mobile Source Node. In the second scenario, the source node B$_4$ is moving at the speed of 1 meter per second vertically as shown in Figure 6. It is assumed that the speed of a fast walking patient is 1 meter per second.

Once again, it is observed that QPRD provides better results than DMQoS and noRouting in case of mobile source node scenario. Figure 7(a) shows that the throughput is in excess of 80% in QPRD for offered data packet rates less than 8 k. The throughput reduces slightly at higher offered data packet rates of 8 k and more and reduces to 71% when total offered packets sent by the source are 20 k. In contrast with DMQoS, it is observed that when the offered data packet load is increased, DMQoS suffers from a much lower successful data transmission rate that reduces from 50% to 32% with resultant low throughput. Due to node mobility, the source node moves away from its neighbor nodes resulting in a connection loss which results in more packets being lost. QPRD handles this situation much more gracefully than DMQoS. In QPRD, the mobile nodes resume the connection more rapidly once the nodes come back into the range of neighbor node. The overall lower throughput in this scenario is due to the packet lost when the mobile node is out of range. Equation (4d) in Section 3.1 also supports this behavior. According to (4d), the packet delivery is successful only if a source node transmits data to an in-range destination node. The packets are dropped when the movable nodes go out of range. The noRouting provides the lower throughput with an average of 64%.

Figure 7(b) shows that the number of packets forwarded by the intermediate nodes in QPRD is on average 0.75 times and 9 times lower when compared to the number of packets forwarded by intermediate nodes in DMQoS and noRouting protocols, respectively. The routing mechanism used in the QPRD protocol helps to send the data directly to the destination without transferring the packets to the intermediate nodes in case the destination is in range. It can be seen in Section 3.3 that use of intermediate node results in larger delay and in Section 3.2 that the backoff of other noncapturing nodes also contributes to exacerbating the problem. The performance of noRouting for this parameter is worst as it forwards up to 26 k packets which increases the overall network traffic.

It is observed from Figure 7(c) that the overall network traffic in QPRD is about 25% and 50% less than DMQoS and noRouting protocols, respectively, for all offered network data loads considered. This is due to the end-to-end path calculation mechanism used in QPRD. The delay of all the nodes is considered and QPRD algorithm selects the best next hop, on the basis of end-to-end path delay information, to send the data from source to destination.

From Figure 7(d), it is seen that QPRD has no packets that were timed out for data packet transmissions at 8 k or less. The selection of minimum end-to-end path delay, given in [18], helps QPRD to send the data through a path where lower packets time out occurs. For high data packets (above 8 k), the source node moves out of the neighbors' radio range which causes more packets to time out. On the other hand, DMQoS has more timed out packets than QPRD. Initially for low offered data packet rates below 4 k, about 40% of data packets were timed out, and for higher offered data packets (above 4 k) the 40% of data packet time outs increase to 50% (approximately). This is because the packets travel through different nodes by using hop-by-hop delay calculation as discussed in detail in scenario 1. Equation (9) in Section 4.2 shows that the delay on each node is the summation of four different delays (i.e., transmission (DL$_{\text{trans}}$), MAC and network queues



(a) Throughput

(b) Packets forwarded by intermediate nodes

(c) Overall network traffic

(d) Packets timeout

(e) Packets dropped due to MAC buffer overflow

(f) Overall energy consumption

Figure 7: Performance comparison for different parameters when source node is mobile.

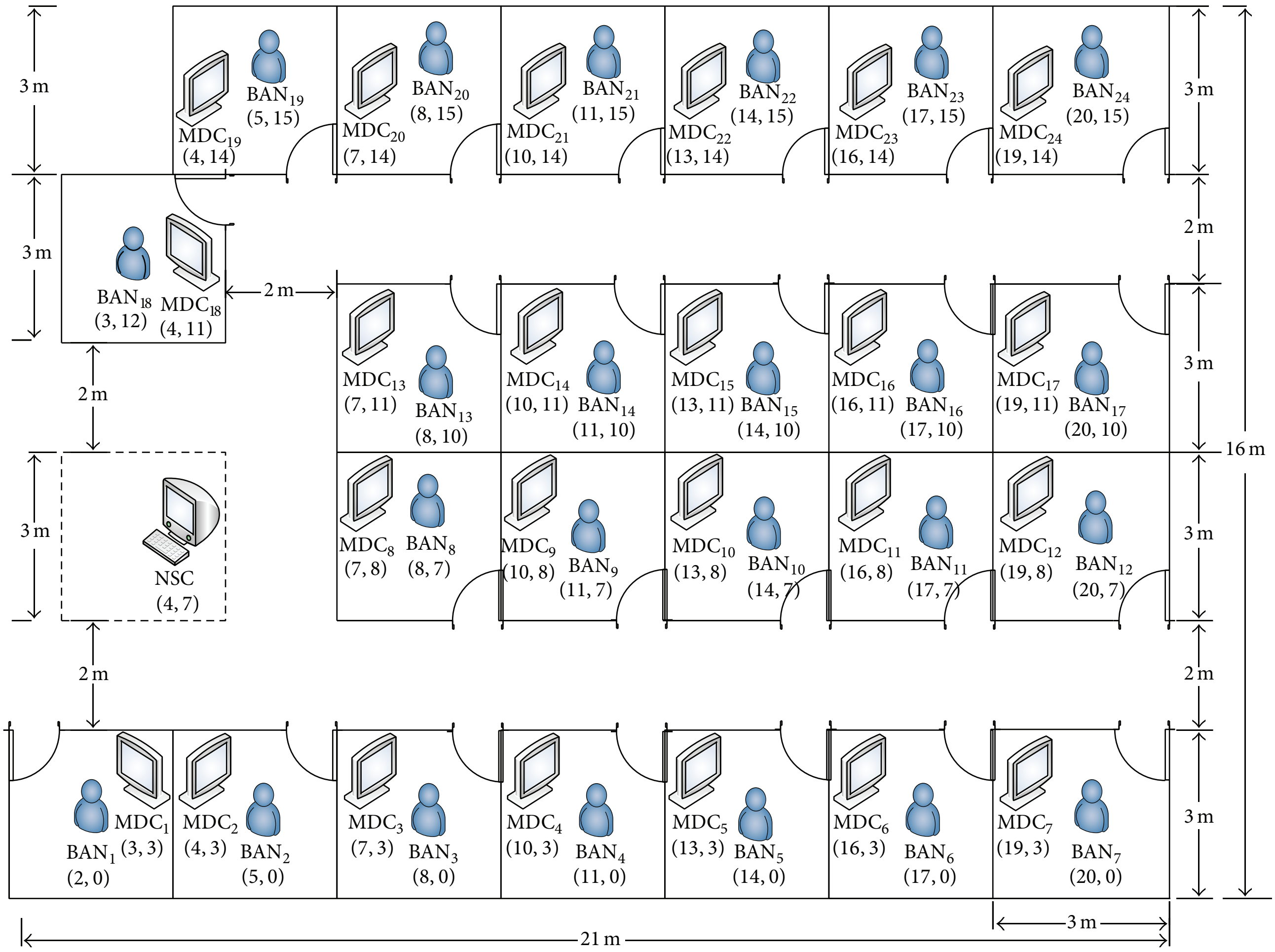


Figure 8: Scenario 3. Node deployment for 24 patient beds in hospital environment.

(DL_{queue}), channel ($DL_{channel}$), and processing (DL_{proc}). The calculations done by DMQoS on each node increase the processing delay which causes the increase of overall node delay. The higher node delay results in packet time out. The source node mobility makes the packet time out worse than scenario 1 of Figure 5(d).

Figure 7(e) shows that there are no packet drops due to MAC buffer overflow in QPRD and DMQoS protocols, whereas 9 k packets are dropped in noRouting. The performance of DMQoS is similar to QPRD in terms of MAC buffer overflow; however, DMQoS has on average 39% lower throughput and an average of 25% higher overall network traffic.

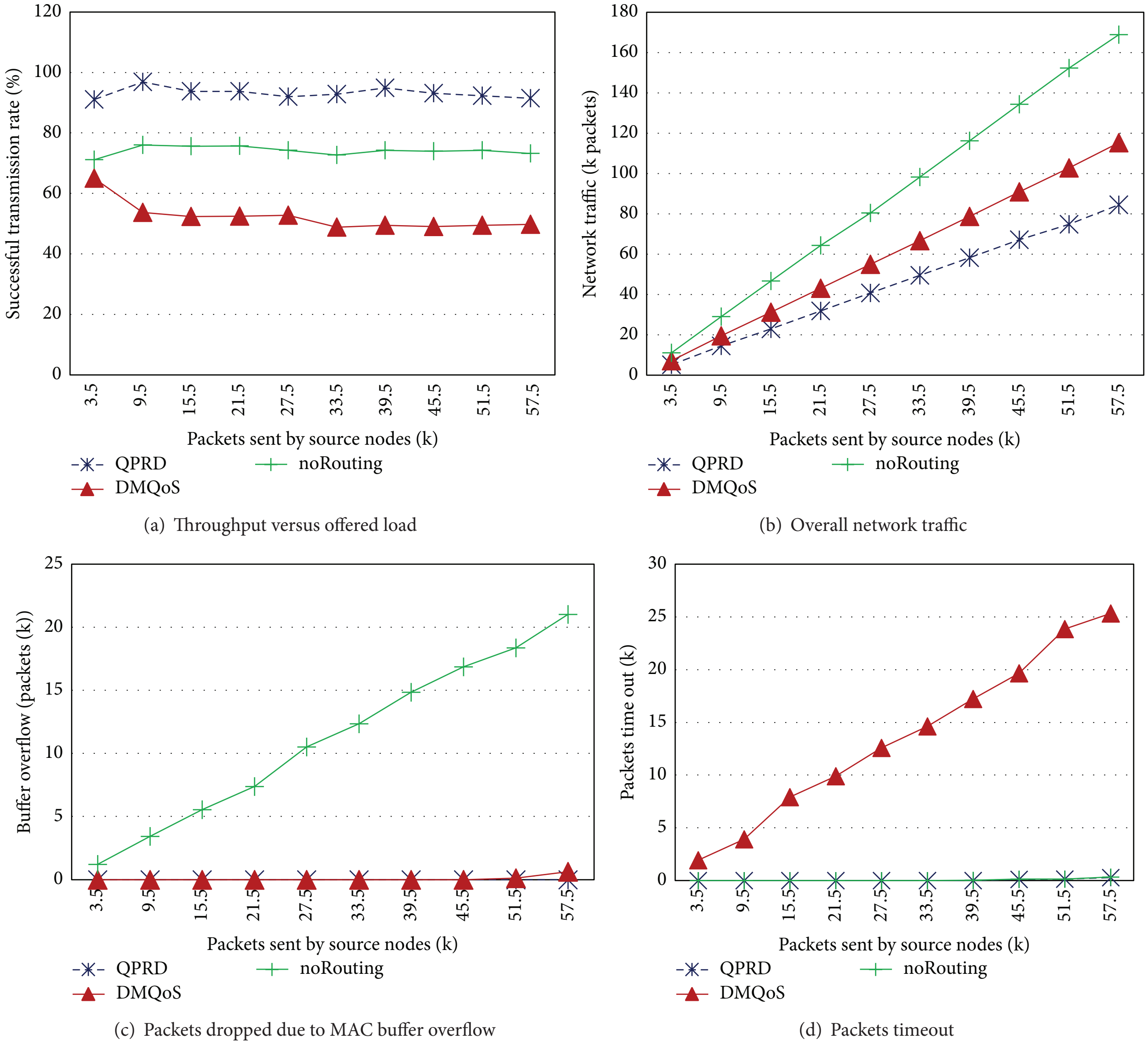
From Figure 7(f), it is observed that the overall energy consumptions of QPRD and DMQoS are 18.9 Joules to 275.7 Joules when 1 k to 20 k packets are sent by source nodes. The noRouting consumes 2.6 Joules to 47 Joules when 1 k to 20 k packets are sent by source nodes. The computations for delay in QPRD are almost similar to the DMQoS but QPRD provides on average 25% lower overall network traffic, 73% fewer packets forwarded by intermediate nodes, and, more importantly, a 40% higher successful data transmission rate (throughput) as compared to DMQoS.

In summary, the overall performance of QPRD is better than DMQoS and noRouting when the source node is mobile.

6. Scalability Test: Real Hospital Environment with 24 Beds (49 Nodes)

A real 24-patient-bed hospital is considered for the scalability test of QPRD routing protocol, as shown in Figure 8. The approximate area covered is 16 m by 21 m which is similar to the Hematology-Oncology Unit of the Children Hospital named IWK Health Centre Halifax, Canada. The distance between two beds is 3 meters which is in accordance with the recommended transmission range for BAN communication in hospital environment. The total nodes used in the deployment area are 49 (24 BANs, 24 MDCs, and 1 NSC). Each BAN transmits the data to its peer MDC. All the BANs and MDCs are sending or receiving Hello protocols to/from other nodes and the NSC.

Both MDCs and BANs are movable. Generally, BANs can move freely anywhere and the movement of a MDC is only within the room where it is placed. It is assumed that the MDC of one room has a connection with the MDC of the next room.



(a) Throughput versus offered load

(b) Overall network traffic

(c) Packets dropped due to MAC buffer overflow

(d) Packets timeout

Figure 9: Performance comparison for different parameters of scenario 3.

The simulation results show that QPRD performs better than DMQoS and noRouting even when the number of nodes is increased to 49. From Figure 9(a) it is seen that the throughput provided by QPRD is in excess of 91%, whereas the throughput of noRouting and DMQoS protocols is on average 74% and 52%, respectively. From Figure 9(b), it is observed that the overall network traffic of QPRD is 50% and 25% less than noRouting and DMQoS protocols, respectively. Figure 9(c) shows that the packet drops due to MAC buffer overflow in QPRD and DMQoS protocols are negligible, whereas 9 k packets are dropped in noRouting. Figure 9(d) shows that there are no packets timeouts due to not fulfilling the delay requirements in QPRD and noRouting. On the other hand 25 k packets are timed out in DMQoS. From these results it is shown that QPRD is equally effective when the deployment area is larger, and number of nodes has been increased to simulate a real hospital scenario with 24 patient beds.

7. Conclusion

The paper models the wireless BAN as a directed graph and derives conditions for throughput maximization and end-to-end delay minimization. It is shown that efficient energy utilization is critical to the proper design of the routing and MAC layer protocols. Similarly, delay is minimized by formulating the BAN end-to-end path delay as a linear programming problem with multiple constraints to be satisfied simultaneously.

Based on the mathematical analysis, a novel modular QoS-aware routing protocol for hospital BAN communication is proposed in this paper. The architecture of the new protocol consists of seven modules: the MAC receiver, the delay module (DM), the packet classifier (PC), the Hello protocol module (HPM), the routing services module (RSM), the QoS-aware queuing module (QQM), and the MAC transmitter. The proposed routing protocol provides a mechanism for the end-to-end path delay calculation of all possible paths from a source to destination and then decides the best possible path by considering the path delay requirements of the delay-sensitive packets.

OMNeT++ based simulator Castalia 3.2 is used to test the performance of the proposed protocol (QPRD) and compare it with DMQoS and noRouting. The simulations are performed for both the movable source and stationary scenarios. A scalability test is done with larger deployment area and by using higher number of nodes. The results show that the QPRD offers over 94% successful data transmission rates for delay-sensitive packets in a stationary patient scenario. QPRD provides about 35% better results in terms of successful transmission rate than DMQoS in the movable patient scenario. The simulation results show that the QPRD improves the reliability of Body Area Networks by 40% on average for each scenario by decreasing the number of packet time outs with zero and averaging 729 packets for the static and mobile patient scenarios, respectively. In addition, QPRD results in an average of 25% lower overall network traffic for each mobile and static patient scenarios as compared to similar protocols. The scalability test results prove that QPRD outperforms DMQoS and noRouting even when a higher number of nodes are employed in the BAN. QPRD provides on average 93% throughput without any packet being timed out and any packet being dropped due to MAC buffer overflow.

Notations for the Proposed Algorithm

Node i:	Source node
Node j:	Neighbor node of source node
Node Dst:	Destination node (i.e., NSC, MDC, BAN)
$\mathrm{ID}_{\mathrm{Dst}}$:	Destination ID
L_{Dst}:	Destination location
ID_j:	Neighbor node j ID
L_j:	Neighbor node j location
$D_{(j,\mathrm{Dst})}$:	Distance between neighbor node j and destination Dst
E_j:	Residual energy of node j
T_j:	Device type of node j
$D_{(i,j)}$:	Distance between node i to neighbor node j
$\mathrm{NH}_{(i,\mathrm{Dst})}$:	Next hop between node i and destination Dst
NH_E:	Energy-aware next hop
NH_D:	Next hop for delay-sensitive packets
$\mathrm{DL}_{\mathrm{path}(i,\mathrm{Dst})}$:	Path delay from node i to destination Dst
$\mathrm{DL}_{\mathrm{node}(i)}$:	Time delay within the node i
$\mathrm{DL}_{\mathrm{req}}$:	Required path delay for delay-sensitive packets.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] B. Zhen, M. Patel, S. Lee, E. T. Won, and A. Astrin, "15-08-0644-09-0006-tg6-technical-requirements," IEEE Project: IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs), 2011, https://mentor.ieee.org/802.15/dcn/11/15-11-0307-00-0006-tg6-closing-report-march-2011.ppt.

[2] IEEE, "IEEE 802.15 WPAN task group 6 (TG6) body area networks," IEEE 802.15, 2007, http://ieee802.org/15/pub/TG6.html.

[3] D. Curtis, E. Shih, J. Waterman et al., "Physiological signal monitoring in the waiting areas of an emergency room," in *Proceedings of the ICST 3rd International Conference on Body Area Networks (BodyNets '08)*, Tempe, Ariz, USA, 2008.

[4] S. Jiang, Y. Cao, S. Iyengar et al., "CareNet: an integrated wireless sensor networking environment for remote healthcare," in *Proceedings of the 3rd International ICST Conference on Body Area Networks (BodyNets '08)*, Tempe, Ariz, USA, March 2008.

[5] T. Gao, T. Massey, L. Selavo et al., "The advanced health and disaster aid network: a light-weight wireless medical system for triage," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 1, no. 3, pp. 203–216, 2007.

[6] A. Wood, G. Virone, T. Doan et al., "ALARM-NET: wireless sensor networks for assisted-living and residential monitoring," Tech. Rep. CS-2006-11, Department of Computer Science, University of Virginia, Charlottesville, Va, USA, 2006.

[7] Z. Khan, S. Sivakumar, W. Phillips, and N. Aslam, "A new patient monitoring framework and Energy-aware Peering Routing Protocol (EPR) for Body Area Network communication," *Journal of Ambient Intelligence and Humanized Computing*, vol. 5, no. 3, pp. 409–423, 2014.

[8] Z. Khan, S. Sivakumar, W. Phillips, and B. Robertson, "A QoS-aware routing protocol for reliability sensitive data in hospital body area networks," *Procedia Computer Science*, vol. 19, pp. 171–179, 2013.

[9] Z. Khan, S. Sivakumar, W. Phillips, and B. Robertson, "QPRR: QoS-aware peering routing protocol for reliability sensitive data in body area network communication," *The Computer Journal*, 2014.

[10] Z. Khan, S. Sivakumar, W. Phillips, and B. Robertson, "QPRD: QoS-aware peering routing protocol for delay sensitive data in hospital body area network communication," in *Proceedings of the 7th International Conference on Broadband, Wireless Computing, Communication and Applications (IEEE BWCCA '12)*, pp. 178–185, University of Victoria, Victoria, Canada, November 2012.

[11] X. Huang and Y. Fang, "Multiconstrained QoS multipath routing in wireless sensor networks," *Wireless Networks*, vol. 14, no. 4, pp. 465–478, 2008.

[12] S. Agarwal, Divya, and G. N. Pandey, "SVM based context awareness using body area sensor network for pervasive healthcare monitoring," in *Proceedings of the 1st International Conference on Intelligent Interactive Technologies and Multimedia (IITM '10)*, pp. 271–278, Allahabad, India, December 2010.

[13] A. Razzaque, C. S. Hong, and S. Lee, "Data-centric multiobjective QoS-aware routing protocol for body sensor networks," *Sensors*, vol. 11, no. 1, pp. 917–937, 2011.

[14] E. Felemban, C.-G. Lee, and E. Ekici, "MMSPEED: multipath multi-SPEED protocol for QoS guarantee of reliability and timeliness in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 6, pp. 738–754, 2006.

[15] M. Chen, T. Kwon, S. Mao, Y. Yuan, and V. C. M. Leung, "Reliable and energy-efficient routing protocol in dense wireless

sensor networks," *International Journal of Sensor Networks*, vol. 4, no. 1-2, pp. 104–117, 2008.

[16] M. Razzaque, M. Alam, M. Rashid, and C. Hong, "Multi-constrained QoS geographic routing for heterogeneous traffic in sensor networks," in *Proceedings of the 5th IEEE Consumer Communications and Networking Conference (CCNC '08)*, Kyung Hee University, Las Vegas, Nev, USA, January 2008.

[17] J. Elias and A. Mehaoua, "Energy-aware topology design for wireless body area networks," in *Proceedings of the IEEE International Conference on Communications (ICC '12)*, pp. 3409–3410, Ottawa, Canada, June 2012.

[18] N. Ababneh, N. Timmons, J. Morrison, and D. Tracey, "Energy-balanced rate assignment and routing protocol for body area networks," in *Proceedings of the 26th IEEE International Conference on Advanced Information Networking and Applications Workshops (WAINA '12)*, pp. 466–467, Fukuoka, Japan, March 2012.

[19] NICTA, "Castalia," National ICT Australia, 2011, http://castalia.npc.nicta.com.au/.

[20] A. Boulis, "Castalia, wireless sensor network simulator, NICTA," 2011, https://forge.nicta.com.au/docman/view.php/301/592/Castalia+-+User+Manual.pdf.