1-1-2013

Energy efficiency performance improvements for ant-based routing algorithm in wireless sensor networks

A. M. Zungeru

Kah Phooi Seng

Li-Minn Ang
Edith Cowan University, li-minn.ang@ecu.edu.au

W.C. Chia

Hindawi Publishing Corporation
Journal of Sensors
Volume 2013, Article ID 759654, 17 pages
http://dx.doi.org/10.1155/2013/759654

Research Article

Energy Efficiency Performance Improvements for Ant-Based Routing Algorithm in Wireless Sensor Networks

Adamu Murtala Zungeru,[1] Kah Phooi Seng,[2] Li-Minn Ang,[3] and Wai Chong Chia[2]

[1] *Department of Electrical and Electronic Engineering, University of Nottingham, Jalan Broga, 43500 Semenyih, Selangor Darul Ehsan, Malaysia*
[2] *School of Computer Technology, Sunway University, 5 Jalan Universiti, Bandar Sunway, 46150 Petaling Jaya, Selangor, Malaysia*
[3] *School of Engineering, Edith Cowan University, Joondalup, WA 6027, Australia*

Correspondence should be addressed to Adamu Murtala Zungeru; adamuzungeru@ieee.org

Received 30 June 2012; Accepted 12 December 2012

Academic Editor: Xinyong Dong

The main problem for event gathering in wireless sensor networks (WSNs) is the restricted communication range for each node. Due to the restricted communication range and high network density, event forwarding in WSNs is very challenging and requires multihop data forwarding. Currently, the energy-efficient ant based routing (EEABR) algorithm, based on the ant colony optimization (ACO) metaheuristic, is one of the state-of-the-art energy-aware routing protocols. In this paper, we propose three improvements to the EEABR algorithm to further improve its energy efficiency. The improvements to the original EEABR are based on the following: (1) a new scheme to intelligently initialize the routing tables giving priority to neighboring nodes that simultaneously could be the destination, (2) intelligent update of routing tables in case of a node or link failure, and (3) reducing the flooding ability of ants for congestion control. The energy efficiency improvements are significant particularly for dynamic routing environments. Experimental results using the RMASE simulation environment show that the proposed method increases the energy efficiency by up to 9% and 64% in converge-cast and target-tracking scenarios, respectively, over the original EEABR without incurring a significant increase in complexity. The method is also compared and found to also outperform other swarm-based routing protocols such as sensor-driven and cost-aware ant routing (SC) and Beesensor.

1. Introduction

A sensor network is an infrastructure composed of sensing, computing, and communication elements that give a user or administrator the ability to instrument, observe, and react to events and phenomena in a specific environment [1, 2]. wireless sensor networks (WSNs) are collections of compact-size, relatively inexpensive computational nodes that measure local environmental conditions, or other parameters and forward such information to a central point for appropriate processing. Each node is equipped with embedded processors, sensor devices, storage, and radio transceivers. The sensor nodes typically have limited resources in terms of battery supplied energy, processing capability, communication bandwidth, and storage. WSN nodes can sense the environment, communicate with neighboring nodes, and in many cases perform basic computations on the data being collected. WSNs applications include commercial applications such as healthcare, target tracking, monitoring, smart homes, surveillance, and intrusion detection. Many applications of sensor networks deal with the static nature of nodes which in most cases sense their environment and then send the measured values to a central base station through hop-to-hop (multihop) routing, hence leading to rapid exhaustion of energy around the sink (base station). The issue is that sensor nodes around the sink tend to deplete faster in energy than those farther away. This is mainly because, besides forwarding their own traffic, they forward traffic on behalf of other sensor nodes that are located far away from the sink node. Therefore sensor nodes nearer to a sink tend to consume more energy than those farther away. Due to the high depletion in energy, the sensor nodes closer to the sink will drain their energy

resources faster than other nodes which will result in their death. Hence, the lifetime of sensor network can be improved upon if the energy spent in traffic relaying to the sink is reduced.

In recent years, several competitive efficient routing algorithms for WSNs have been developed and surveyed [3, 4]. Recent trends in WSN routing have been towards strengthening existing approaches by considering more detailed network properties. Early work sought to adapt only the network topology such as finding a shortest path. However, WSN environments are affected by many more factors than simply changes in topology. Additional factors may include traffic congestion, latency, link quality, relative node mobility, and most importantly minimum energy path. Swarm intelligence-based routing which utilizes the behavior of real biological species searching for food through pheromone deposition while dealing with problems that need to find paths to goals has been proposed to deal with some of the challenges as mentioned above. This biologically inspired approach is proposed to adapt to the aggregate effects of each of these phenomena by finding paths of maximum throughput.

Social insect communities have many desirable properties from the WSN perspective as surveyed in [4, 5]. These communities are formed from simple, autonomous, and cooperative organisms that are interdependent for their survival. Despite a lack of centralized planning or any obvious organizational structure, social insect communities are able to effectively coordinate themselves to achieve global objectives. The behaviors which accomplish these tasks are emergent from much simpler behaviors or rules that the individuals are following. The coordination of behaviors is also adaptive, flexible, and robust and is capable of solving real-world problems. No individual is critical to any operation, and task progress can easily be recovered from any setback. The complexity of the solutions generated by such simple individual behaviors indicates that the whole is truly greater than the sum of the parts [6–10]. The characteristics described above are desirable in the context of sensor networks. Such systems may be composed of simple nodes working together to deliver messages, while resilient against changes in its environment. The environment of sensor networks might include anything from its own topology to physical layer effects on the communications links to traffic patterns across the network. A noted difference between biological and engineered networks is that the former have an evolutionary incentive to cooperate, while engineered networks may require alternative solutions to force nodes to cooperate [11, 12]. The ability of social insects to self-organize relies on four principles: positive feedback, negative feedback, randomness, and multiple interactions. A fifth principle, stigmergy, arises as a product of the previous four [7]. In general, such self-organization is known as swarm intelligence. Research in this field of swarm intelligence has been focused on working principles of ant colonies as adopted in [13, 14], slime mold [15], and honey bees [16]. We propose a swarm intelligence based energy-aware routing algorithm for WSN considering the above constraints and social insect behaviors. In this paper, we propose several improvements for EEABR [17] to increase its energy efficiency. The improvements are based on the following: (1) a new scheme to intelligently initialize the routing tables giving priority to neighboring nodes that simultaneously could be the destination, (2) intelligent update of routing tables in case of a node or link failure, and (3) reducing the flooding ability of ants for congestion control. Comparison of the modified EEABR is made with some of the state-of-the-art routing protocols based on swarm intelligence.

The rest of the paper is organized as follows. Section 2 gives an overview of related work. Section 3 describes static, dynamic and mobility sink models in WSN. Section 4 presents a brief description of the social insect analogy. In Section 5, we describe our proposed algorithm. Section 6 evaluates the performance of the proposed algorithm and other routing protocols. Section 7 concludes the paper with comments for future work.

2. Related Work

The idea of using the swarm paradigm to establish routes in communication networks is not new. In [14], an ant-based algorithm was adopted to calculate the optimal paths among the nodes through an architecture called AntNet. Smaller agents, the virtual ants, migrate from a node to another, building the routing rules in a distributed way.

In SC (sensor-driven and cost-aware ant routing) [18], it is assumed that ants have sensors so that they can smell where there is food at the beginning of the routing process so as to increase in sensing the best direction that the ant will go initially. In addition to the sensing ability, each node stores the probability distribution and the estimates of the cost of destination of each of its neighbors. The routing suffers from misleading when there is an obstacle which might cause errors in sensing the best direction. In their extended work, FF (flooded forward) Ant routing [18] argues the fact that ants even augmented with sensors can be misguided due to the obstacles or moving destinations. The protocol is based on the flooding of ants from source node to the sink node. In the case where the destination is not known at the beginning by the ants or cost cannot be estimated, the protocol simply uses the broadcast method of sensor networks so as to route packets to the destination. Probabilities are updated in the same way as the basic ant routing, though FF reduces the flooding ants when a shorter part is traversed. However, the authors only focus on the building of an initial pheromone distribution, which is good at system startup, but bad when the system density is high.

The energy-efficient ant-based routing for WSN (EEABR) as proposed in [17] is an improved version of the ant based routing in WSN which does not only consider the nodes in terms of distance, but also in terms of the energy level of the path traversed by the ants. The authors in their work pointed out that, in the basic ant algorithm, the forward ants are sent to no specific destination node which means that sensor nodes must communicate with each other and the routing tables of each node must contain the identification of all the sensor nodes in the neighborhood and the corresponding levels of pheromone trail. In their work, much achievement

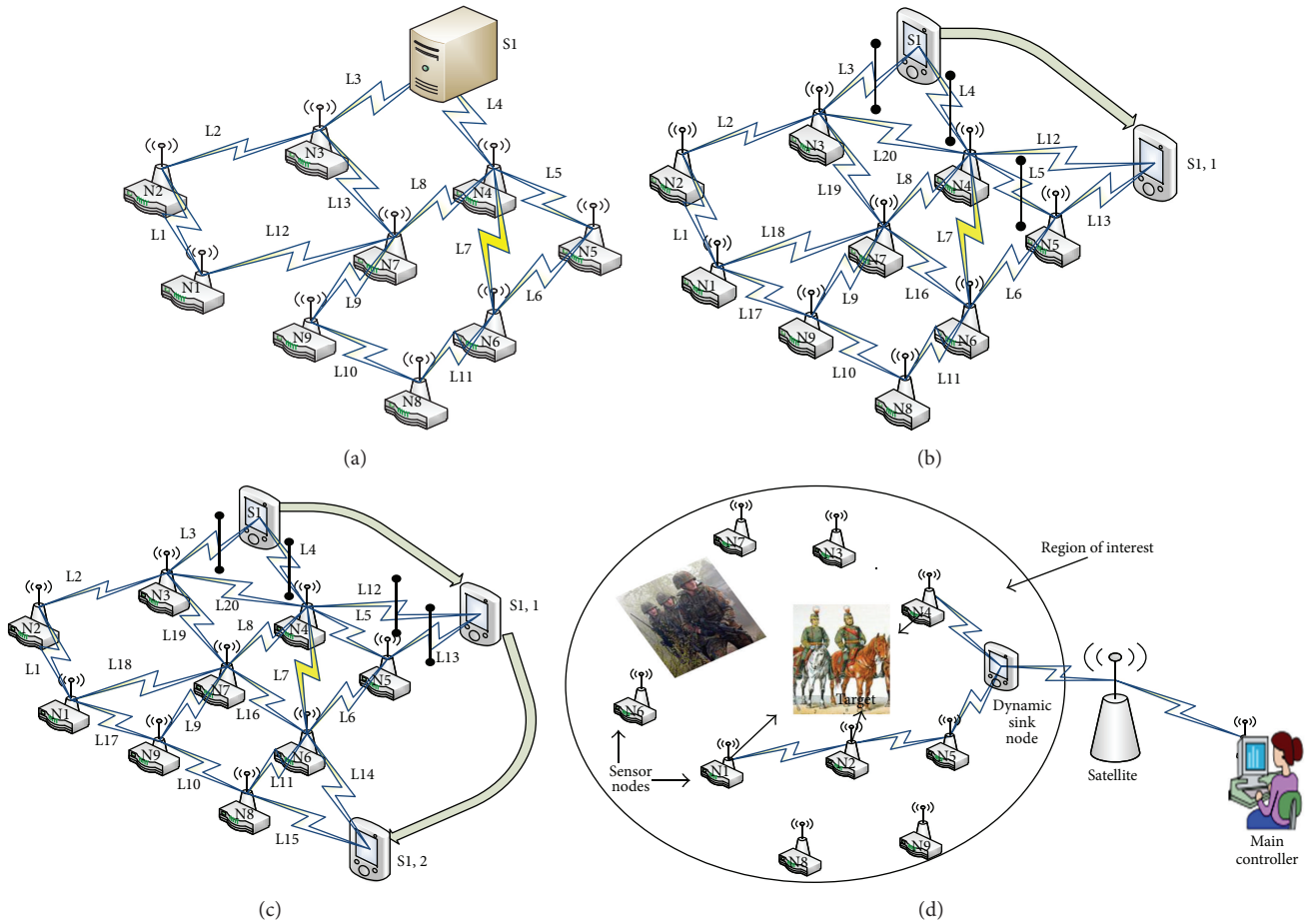


FIGURE 1: (a) Sink in a static position, (b) mobility of sink (S1 → S1, 1), (c) mobility of sink (S1 → S1, 1 → S1, 2), and (d) dynamic sink (target tracking) scenario.

was recorded in energy savings, but encounter difficulty in the mobility and dynamic scenario since much control traffic are generated hence consuming much energy with less reliability.

Beesensor [16] is an algorithm based on the foraging principles of honey bees with an on-demand route discovery. The algorithm works with three types of agents: packers, scouts and foragers. Packers locate appropriate foragers for the data packets at the source node. Scouts are responsible for discovering the path to a new destination using the broadcasting principle. Foragers are the main workers of Beesensor which follow a point-to-point mode of transmission and carry the data packets to a sink node. When a source node detects an event and does not have a route to the sink node, it launches a forward scout, puts the events in the payload of the scout, and broadcasts it to all its neighbors. When an intermediate node within the radius of the source node receives the forward scout, it updates its scout cache, increments the hop count, and rebroadcasts it. The approach is based on the interactions of scouts and source routing where small forwarding tables are built during the return of a scout. Its analysis was done and compared with several routing protocols in RMASE simulator. Since it is an on-demand-based protocol, it suffers in the area of security applications or an application where information needs updating at regular intervals of time. The routing process is reactive, the time used to search for the sink in a dynamic scenario of the routing process is high, and as such the algorithm is only good for static applications. For more explanation of the algorithms discussed above, interested readers are referred to [5].

Besides all the drawback of each of the related protocols, almost all the algorithms tend to sacrifice the network performance to gain an improvement of energy consumption of the nodes and vice versa for others.

3. Static, Dynamic, and Mobile Sink in WSNs

This section describes the main application scenarios of a typical WSN. In sensor networks, each sensor generates data packets at a fixed data rate. If a sensor node j is neither colocated with sink s nor directly connected with it, then data packets generated at node j have to be relayed through multiple hops to reach the sink. In a static sink scenario, the sink node is always in a fixed position. All the traffic destined to it must pass through the nodes closer to it, which will make them deplete in their energy resource faster. As seen in Figure 1, node 3 (N3) and node 4 (N4) are the closest and in the transmission range of the sink. Besides transmitting their

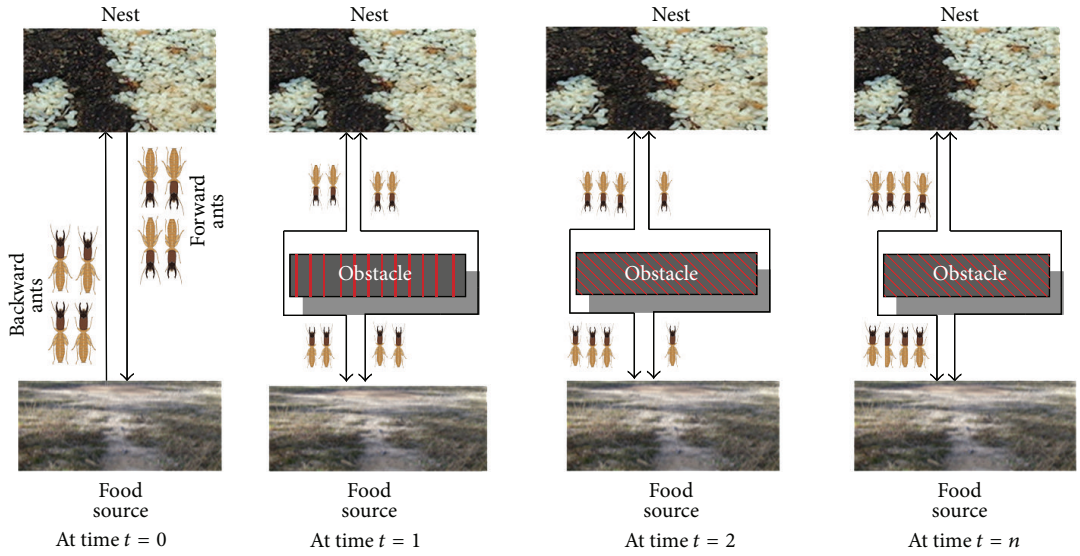


Figure 2: Food search in ants.

own information, they also transmit information belonging to other nodes, which will make them to deplete in their energy faster than other nodes. At time t, when their energy will get exhausted, there will be holes created in their spot as such; since other nodes are not in the transmission range of the sink, the information will no longer get to the sink, hence making the network useless. Due to this problem of energy holes or/and hot spots, it then calls for handling mobility of the sink. The mobile sink will be able to collect information from other nodes even when node 3 and node 4 are no longer active. It will also help in balancing the network energy due to its mobility, as nodes closer to it will keep on changing as can be seen in Figures 1(b) and 1(c). It should also be observed that as the sink moves, some of the links to it have to be broken due to communication range as can be seen with the communication links crossed with a bar. With this, sink mobility will not only improve on the network lifetime, but also helps in collecting the other useful information that could have been lost when node 3 and node 4 were no longer active. A typical routing protocol that involves sink mobility will do the following:

(a) inform the whole network of the topological changes during the sink mobility,

(b) notify a node when its link with the sink gets broken due to sink mobility.

This leads us to the social insect way of solving real world problems which are reviewed in Section 4.

Target tracking in WSNs is a process of estimating the location, trajectory, velocity, and/or acceleration of a mobile target. It often needs accurate estimation and prediction of the target state. In a typical-target tracking (dynamic scenario) application in WSN, the targeted event is mobile. Hence, the source nodes or sink node has to change its location at every interval of time to cope with its dynamic nature of the target event. Figure 2 shows a target-tracking scenario in a typical WSN.

In this scenario, the sink is responsible for forwarding the desired information from the WSN to the headquarters (i.e., main controller) through the Internet, via satellite or other wireless technology. The target can be a human being, moving vehicle, animal, tank, enemy, or any interesting object that needs to be tracked which is usually mobile. Targets can move in an unexpected manner and this causes loss of target sometimes. Locating the position of the target at any point in time is one of the main challenges for target tracking in WSNs. Each sensor node has a sensor device to sense or detect the presence of the target in the region of interest (ROI). Detection of the target is always handicapped when it is out of transmission range of the nodes. Therefore, using a mobile sink or nodes can lessen the burden of loss of the target, which will also in turn reduce energy consumption by the network nodes since less control traffic will be used to locate the position of the target. Due to the limited battery-supplied energy of the sensor nodes and the difficulty to physically access them, energy-efficient target tracking is a crucial aim. Additionally, hundreds or thousands of sensor nodes are deployed in the ROI. But with an energy-efficient routing algorithm and sink in dynamic scenario, the position of the target can easily be tracked and fewer hops will be needed by the sensor nodes to get the information across the sink node.

4. The Social Insect Abilities

The search for food in ants is organized in part by chemical trails laid while searching for food source. During foraging for food, ants communicate with the aid of the pheromone

laid on their way back to the nest. When food is discovered, ants return to the nest laying a trail to recruit nest mates to the food source. The difference between foraging and recruitment trails is attributed to different quantities of trail pheromone present on the path. Ants have been found to adapt to their environment and always find the most efficient path to their food source [14]. The optimization of path behavior of ant is now widely used to motivate applications such as routing in WSN. Considering Figure 2, the path from the nest to the food source at time $t = 0$, the ants find the food and bring it back efficiently, establishing a pheromone trail to it. At time $t = 1$, when there is an obstacle in their path such that there is one part that is shorter than the other, the ants can choose either path with equal probability, hence having the same number of ants on both sides. The path that is shorter will allow ants to gather food quickly and strengthen the pheromone trail on the way back faster than the ants on the longer path as seen when $t = 2$, causing the other batch of ants to move with higher probability towards the trail that is stronger. As the process continues till time $t = n$, it will be observed that all the ants will use the shortest path towards the food source. The obstacle in this analogy can be congestion, number of nodes on the path to the sink, latency, and so forth. This example illustrates the four principles of self-organization [7].

The following subsections explain how the principles of swarm intelligence interplay in the foraging of ants toward food sources.

4.1. Positive Feedback. Positive feedback represents general rules for a particular behavior. In ants foraging, an ant's attraction towards the pheromone gradient biases it to adding to large piles known as positive feedback. The more the piles accumulate, the more pheromone it is likely to have, and thus an ant is more biased to move towards it and potentially add to the pile on the path. The greater the biases to the food source, the more that ants are also likely to take the path to that food source, further increasing the pheromone content of the path.

4.2. Negative Feedback. Negative feedback is accomplished by pheromone evaporation. This happens so as to avoid premature convergence among ants (stagnation). For good communication among the ant individuals, pheromone must evaporate over the environment. The evaporation helps to weaken the pheromone and lower the concentration of the pheromone on that path. The path with lower pheromone concentration will have fewer ants as it will attract fewer ants towards that direction. Though this may seem contrary to the task of collecting all food to the nest, but it is important. Negative feedback is entirely useful in the removal of old or poor solutions from the collective memory of the system.

4.3. Randomness. The location and path taken by ants towards the food source is entirely determined by chance. A little drift in the behavior of ants may have a large influence on future events. Randomness is exploited to allow for new solutions to arise or to direct current solutions as they evolve to fit the environment.

4.4. Multiple Interactions. In the food collection of ants to their nest, it is a necessity that many individuals cooperate and work together to achieve their target; this is in accordance with neighboring nodes of a sensor network acting as routers to other source nodes. If not enough ants exist in a nest, then the pheromone would decay before any more food could be collected at the nest. Also, if we map this to a sensor network, if there are not many nodes on the path to the sink node, more packets will be dropped on the way to the sink. This might also be as a result of the low transmission distance of the nodes too. But if there exist more ants in the environment, the more food will be gathered fast to avoid complete pheromone decay in the shortest path, else ants would continue their random walk without building any strong solution as regards the best paths.

4.5. Stigmergy. This is the indirect communications between individuals of the social insect, generally through their environment. Complexity in stigmergic systems is due to the fact that individuals interact not with each other but with a common environment. They interact with the environment by making changes to it. These changes affect the way further changes are made. This gives rise to a positive feedback effect, where information feeds upon information (much the same effect as when conversations can take unpredictable directions according to the way people respond to each other's comments). Ants are directed to the path with a high pheromone gradient; it is not necessary for ants to directly communicate with each other or even to know of each other's existence. For this reason, ants are allowed to act independently of other individuals, which greatly simplifies the necessary rules.

5. Improved Energy-Efficient Ant-Based Routing Algorithm (IEEABR)

WSNs protocols are designed to meet some important functions like quality of service and improvement of the network lifetime. It is then important for each protocol designed for WSN to consider the energy efficiency of the underlying algorithm and its reliability. This is necessary for the network under consideration having limited power supply, constrained memory capacity, low processing capability, and constrained available bandwidth. To this end, we proposed some important improvements on an existing energy-efficient ant-based routing algorithm (EEABR).

5.1. Energy-Efficient Ant-Based Routing Algorithm (EEABR). energy-efficient ant based routing (EEABR) algorithm [17] is an improved version of the basic ant-based routing for WSN. The author's main idea is to reduce the communication load in the network related to the control packets (ants) and also save energy spent with communication. The proposal also includes new functions to update the pheromone tables on

the nodes. The basic algorithm of EEABR can be informally described as follows.

(1) At regular intervals, from every network node, a forward ant k is launched with the aim to find a path until the destination. The identifier of every visited node is saved onto a memory M_k and carried by the ant. Where k is any network node having a routing table and will have N entries, one for each possible destination, and d is one entry of the k routing table (a possible destination).

(2) At each node r, a forward ant selects the next hop node using the same probabilistic rule proposed in the ACO metaheuristic:

$$P_k(r,s) = \begin{cases} \dfrac{[\tau(r,s)]^{\alpha} \cdot [E(s)]^{\beta}}{\sum_{u \notin M_k} [\tau(r,u)]^{\alpha} \cdot [E(s)]^{\beta}}, & s \notin M_k \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where $P_k(r,s)$ is the probability with which ant k chooses to move from node r to node s, τ is the routing table at each node that stores the amount of pheromone trail on connection (r,s), E is the visibility function given by $1/(C-e_s)$ (C is the initial energy level of the nodes and e_s is the actual energy level of node s), and α and β are parameters that control the relative importance of trail versus visibility. The selection probability is a trade-off between visibility (which says that nodes with more energy should be chosen with high probability) and actual trail intensity (that says that if in connection (r,s) there has been a lot of traffic then it is highly desirable to use that connection).

(3) When a forward ant reaches the destination node, it is transformed to a backward ant whose mission is now to update the pheromone trail of the path it used to reach the destination and that is stored in its memory.

(4) Before backward ant k starts its return journey, the destination node computes the amount of pheromone trail that the ant will drop during its journey:

$$\Delta\tau = \frac{1}{C - \left[\left(E_{\min} - N_j\right) / \left(E_{\text{av}} - N_j\right)\right]}, \quad (2)$$

where C is the initial energy of the nodes, $E_{\min}$, E_{av} are the minimum and average energy, respectively, of the path traversed by the forward soldier as it moves towards the hill, N_j represent the number of nodes that the forward soldier has visited. The idea behind the calculation of $\Delta\tau$ is that it brings optimized routes, since it is a function of the energy level of the path, as well as the length of the path. For example, a path with 10 nodes can have the same energy average as the path with 4 nodes. Therefore, it is important to calculate the pheromone trail as a function of energy and the number of nodes as against the number of nodes as used in other ACO.

(5) Whenever a node r receives a backward ant coming from a neighboring node s, it updates its routing table in the following order:

$$\tau(r,s) = \left(1-\rho\right) * \tau(r,s) + \left[\frac{\Delta\tau}{\emptyset Bd_k}\right], \quad (3)$$

where ϕ is a coefficient and Bd_k is the distance travelled (the number of nodes visited) by the backward ant k until node r, in which the two parameters will force the ant to lose part of the pheromone strength during its way to the source node. ρ is a coefficient such that $(1-\rho)$ represents the evaporation of the pheromone trail since the last time $\tau(r,s)$ was updated. The idea behind the behavior is to build a better pheromone distribution (nodes near the sink node will have more pheromone levels) and will force remote nodes to find better paths. Such behavior is important when the sink node is able to move, since pheromone adaptation will be much quicker.

(6) When the backward ant reaches the node where it was created, its mission is completed and the ant is eliminated.

By performing this algorithm for several iterations, each node will be able to know which are the best neighbors (in terms of the optimal function represented by (2)) to send a packet towards a specific destination.

5.2. Improvements to Energy-Efficient Ant-Based Routing Algorithm. The improved version of EEABR algorithm considers the available power of nodes and the energy consumption of each path as the reliance of routing selection. It improves on memory usage and utilizes the selforganization, self-adaptive and dynamic optimization capability of an ant colony system to find the optimal path and multiple candidate paths from source nodes to sink nodes. The algorithm avoids using up the energy of nodes on the optimal path and prolongs the network lifetime while preserving network connectivity. This is necessary since for any WSN protocol design, the important issue is the energy efficiency of the underlying algorithm due to the fact that the network under investigation has strict power requirements. As proposed in [19] and adopted in [17], for forward ants sent directly to the sink node, the routing tables only need to save the neighbor nodes that are in the direction of the sink node, which considerably reduces the size of the routing tables and, in consequence, the memory needed by the nodes. The memory M_k of each ant is reduced to just two records, the last two visited nodes. Since the path followed by the ants is no more in their memories, a memory must be created on each node that keeps a record of each ant that was received and sent. Each memory record saves the previous node, the forward node, the ant identification, and a timeout value. Whenever a forward ant is received at any node, it searches for any possible loop with the aid of its identification (ID). For the situation where no record is found, the necessary information is retrieved and the timer is restarted, hence forwarding the ant to the next node, else, the ant is eliminated if a record containing the ant identification is found. When a backward ant is received, the source ID is searched so as to know where to send it to. In this section, we proposed some modifications on EEABR to improve the energy consumption in the nodes of WSNs and also in turn improve the performance. The improvements are based on a new scheme to (1) initialize the routing tables and give priority to neighboring nodes that simultaneously could be the destination, (2) intelligent update of routing tables in case of a node or link failure, and (3) reduce the flooding ability of ants for congestion control.

5.2.1. Initialization of Routing Tables. The EEABR improved ant-based routing does not specify an initialization method for the routing tables. For this reason, we propose an initial uniform distribution of probabilities in the routing table. Due to the situation of no a priori knowledge about the network topology, the proposed initialization of each routing table reflects a previous (initial) knowledge about the network topology as the routing process progresses. The initialization of the routing table is done with a uniform probability distribution as

$$P_{id} = \frac{1}{N_k}, \tag{4}$$

where P_{id} is the probability of jumping from node i to node d (destination) and N_k is the set of neighboring nodes of node k. This is done to reflect the previous (initial) knowledge about the network topology as the routing process progresses.

At a given time after network topology update, a greater probability value is assigned to the neighboring nodes that simultaneously could be the destination according to (5); this is in accordance with [20]. This saves network resources, because it is possible to reach the destination using just a link. If a destination node d for a table entry is at the same time a neighbor node, that is $d \in N_k$, then the initial probability in the routing table of k for the node d is given by

$$P_{dk} = \frac{9N_k - 5}{4N_k^2} \tag{5}$$

also, for the other neighboring nodes among the neighbors for which $i \neq d$ and $i \in N_k$ will then have their probability of selection in the routing table of k as

$$P_{ik} = \begin{cases} \dfrac{4N_k - 5}{4N_k^2}, & \text{if } N_k > 1 \\ 0, & \text{if } N_k = 1, \end{cases} \tag{6}$$

where N_k is the set of neighboring nodes of k and P_{ik} is the probability with which an ant or a data packet in k, jumps to a node i, $i \in N_k$ when the destination is $d (d \neq k)$. Then, for each of the N entries in the node k routing table, it will be N_k values of P_{id} subject to the condition:

$$\sum_{i \in N_k} P_{id} = 1; \quad d = 1, \ldots, N. \tag{7}$$

For example, if a source node has five neighbors, the probability of selecting a neighbor node which in turn serves as the destination according to previous knowledge of the route update is $P_{dk} = (9(5) - 5)/4(5^2) = 2/5$, and the other neighbors will be selected based on the probability $P_{ik} = (4(5) - 5)/4(5^2) = 3/20$ each. That is to say that the sum of probabilities of selecting all the five neighbors of the source node will be $P_{id} = (2/5) + (3/20) + (3/20) + (3/20) + (3/20) = 1$.

Of course we can see that (5) and (6) satisfy (7), (note: probability distribution table is maintained by the source nodes only). But for a source node whose neighbor is one, and the neighbor falls to be the destination, according to (5) and (6), $P_{dk} = 1$ and $P_{ik} = 0$. This also satisfies (7).

5.2.2. Reduction of the Network Control Packets. EEABR does not specify any method to maintain control of the total number of forward ants moving inside the network, which, under certain circumstances, could contribute to congestion. In order to control the number of ants, the total number of ants launched at each node was limited to an amount five times the number of network nodes $(5 * N)$, because this is an average number of links for each node in the networks used as seen in Figure 1. With this method, results were improved and this is also in accordance with [20].

5.2.3. Self-Destruction of Control Packets. There are situations when the forward ant sent from the source nodes does not get back to the source node following the reverse link. In that case, in order to avoid infinite loops when a loop occurs, that is when the forward ant keeps moving round the network, self-destruction of the forward ant $F_{s \to d}$ occurs if the amount of hops in a cycle is higher than half of the already accumulated number of hops. When a backward ant $B_{s \leftarrow d}$ cannot return to its source node because its return trip was interrupted, due to either a link or node failure, it is self-destroyed, because the information stored in its memory does not reflect any more the real state of the network. This method helps to reduce wrong update of information in the routing table.

5.2.4. Intelligent Update after Network Resources Failures. The original EEABR does not deal with situations of network resource failures. In the case of link failure, an automatic update is made on the routing tables in cases where a node k loses its link l_{mi} with its neighbor node i. It is assumed that if an ant is in m, the probability P_{id}, to a destination d through node i, is distributed uniformly between the remaining $N_k - 1$ neighbors for the entry d in the routing table of k where $P_{id} = 0$ during a link l_{mi} failure. Hence, it is not possible to travel from m to i for arrival to d. Hence, new probability values after link l_{mi} failure are introduced as P_{id}, and the probabilities will be proportional to their relative values before the failure instead of forgetting what it has learned until the moment of the failure and is updated according to

$$P_{id} = P_{id} * (1 + z) \quad i \neq m, \ i, m \in N_k, \tag{8}$$

where

$$z = \frac{P_{id}}{1 - P_{id}}. \tag{9}$$

This method reflects node knowledge about the network traffic and topology before the failure. With these improvements, the network converges faster, and better results were achieved. The step-by-step description of the algorithm is shown in the algorithm pseudocode in Algorithm 1. In the algorithm, we break down the prototype of procedures into different stages, and in line 6 to line 24, we describe the procedures of initializing the global variables. In that, we define most of the important variables and parameters. Lines 26 to 43 is a decision-making procedure, where decision to calculate pheromone, construction of backward ant, next-hop selection, and elimination of backward ant is decided

Destination \ Neighbor	A	S	F	E
C	$T_{A,C}$	$T_{S,C}$	$T_{F,C}$	$T_{E,C}$
B	$T_{A,B}$	$T_{S,B}$	$T_{F,B}$	$T_{E,B}$
E	$T_{A,E}$	$T_{S,E}$	$T_{F,E}$	$T_{E,E}$
D	$T_{A,D}$	$T_{S,D}$	$T_{F,D}$	$T_{E,D}$
S	$T_{A,S}$	$T_{S,S}$	$T_{F,S}$	$T_{E,S}$
A	$T_{A,A}$	$T_{S,A}$	$T_{F,A}$	$T_{E,A}$
F	$T_{F,A}$	$T_{S,F}$	$T_{F,F}$	$T_{E,F}$

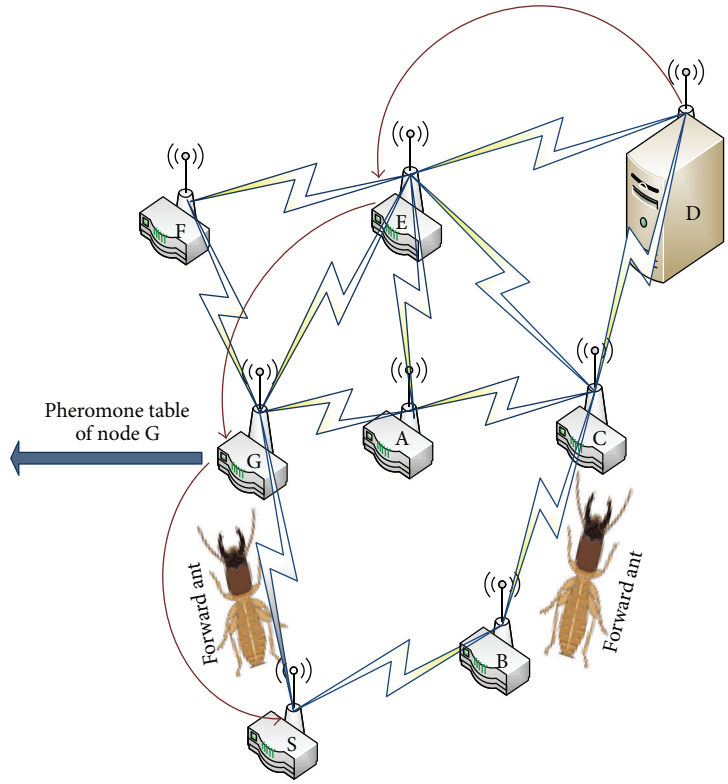


Figure 3: Description of pheromone table of node G.

upon. From line 44 to 74, we describe the selection for the next hop selection, which is in the decision of line 29. In this step, the calculation of the probability of selection is done, and the decision for the elimination of forward ant in the event of occurrence of a loop is also taken. Thereafter, the procedure of updating of the source probability distribution table is given in line 75 to 91. From line 92 are the procedures for calculating the pheromone of paths, which is in the first decision of line 26. With this description, all procedures of pheromone calculation, pheromone updating, and next-hop selection are achieved, and better routing procedure is also achieved.

We also describe in this work the pheromone table contents of a node in the network during the routing process. Below is a description of the pheromone table.

5.2.5. The Pheromone Table. The pheromone table keeps the information gathered by the forward ant. Each node maintains a table keeping the amount of pheromone on each neighbor path. The node has a distinct pheromone scent, and the table is in the form of a matrix with destination nodes listed along the side and neighbor nodes listed across the top. Rows correspond to destinations and columns to neighbors. An entry in the pheromone table is referenced by $T_{n,d}$ where n is the neighbor index and d denotes the destination index. The values in the pheromone table are used to calculate the selecting probabilities of each neighbor. When a packet arrives at node G from previous hop S, that is, the source, the source pheromone decays, and pheromone is added to link $\overrightarrow{SG}$. A backward ant on its way back from the sink node is more likely to take the path through G, since it is the shortest path to the destination that is, $\overrightarrow{SGED}$. The pheromone table of node G is shown in Figure 3 with nodes A, S, F, and E as its neighbor, A, B, C, D, E, F; and S are the possible destinations. It is worth noting that all neighbors are potential destinations in the route selection process of routing. At node G, the total probability of selecting links $\overrightarrow{ED}$, $\overrightarrow{FE}$, $\overrightarrow{AC}$, or $\overrightarrow{SB}$ to the destination node is equal to unity (1), that is, $\sum T_{ED} + T_{SD} + T_{AD} + T_{FD} = 1$. It will then be observed that, since link ED is shorter, more pheromone will be present on it and hence, ants are more likely to take that path.

The Pseudocode describing the algorithm description of our proposed methods is given in Algorithm 1.

6. Experimental and Simulation Results

We use a routing modeling application simulation environment (RMASE) [21] specifically designed for wireless sensor networks, which is a framework implemented as an application in the probabilistic wireless network simulator (Prowler) [22] written and running under Matlab.

Prowler is an event-driven simulator that can be set to operate in either deterministic or probabilistic mode; it provides a fast and easy way to prototype applications and has nice visualization capabilities. Prowler consists of a radio model as well as a medium access control (MAC) layer model. The MAC layer simulates the Berkeley motes'

Function 1: Initialization

(1) //Function: Initialization
(2) //prototype of Procedures
(3) Next-Hop-Selection;
(4) Calculate-Pheromone;
(5) //initialize the global variable
(6) S = Source ID;
(7) //initialize probability distribution table
(8) N = Number of nodes in network;
(9) N_k = the set of neighboring nodes of node k;
(10) $n = N_k$;
(11) dp = distribution probability;
(12) S_{PDT} [n][dp] = Source probability distribution table;
(13) $P_{\mathrm{id}} = \dfrac{1}{N_k}$; // P_{id} is the probability of jumping from node i to node d
(14) //Initiate the routing table
(15) For ($i = 0$; $i = n$; $i + +$){
(16) [dp]·$S_{\mathrm{PDT}} \leftarrow P_{\mathrm{id}}$;
(17) }
(18) D = Destination ID (sink ID);
(19) FA [S, M[2], D] = forward ant [source ID, memory of forward ant, Destination ID];
(20) Ph = amount of Pheromone;
(21) C = initial energy;
(22) BA [S, M[2], Ph, D] = Backward Ant [Source ID, Memory of backward ant, Pheromone value, Destination ID];
(23) E[] = visibility array;
(24) r = Intermediate node ID;

Function 2: Decision making

(25) //If the intermediate node is equal to the destination node, then calculate the pheromone and construct the backward ant
(26) L1: If ($r = d$)
(27) Calculate-Pheromone;
(28) Construct the BA [S, M[2], Ph, d];
(29) L2: Next-Hop-Selection;
(30) If ($r = s$)
(31) Eliminate BA [s, M[2], Ph, d];
(32) //update the routing table
(33) $\rho = 0.8$
(34) $\emptyset = 0.3$
(35) Bd_k = number of visited nodes by the backward ant
(36) $\tau = (1 - \rho) * \tau(r, s) \left[\dfrac{\Delta\tau}{\emptyset Bd_k} \right]$
(37) Goto L3;
(38) Else
(39) Goto L2;
(40) Else
(41) Next-Hop-Selection;
(42) Goto L1;
(43) L3: End

Function 3: Next hop selection

(44) //procedure Next-Hop-Selection
(45) Proc Next-Hop-Selection {
(46) s = next Intermediate node
(47) e = actual energy;
(48) N = Number of nodes;
(49) E_s = visibility of s;
(50) $E_s = \dfrac{1}{C - e_s}$;
(51) $E[] \leftarrow E_s$;
(52) τ = pheromone routing table;
(53) $P(r, s)$ = probability of r jump to s as a next hop;
(54) $\alpha = 0.9$
(55) $\beta = 0.2$
(56) $P(r, s) = \dfrac{[\tau(r, s)]^{\alpha} * [E(s)]^{\beta}}{\sum [\tau(r, u)]^{\alpha} * [E(s)]^{\beta}}$;

Algorithm 1: Continued.

(57) X = number of neighbors which are located in the destination direction;
(58) $P[X]$ = array for storing probability amount of neighbors;
(59) $P[X] \leftarrow P(r, s)$;
(60) $P_{\max} = 0$;
(61) For ($i = 0$; $i = X$; i++){
(62) If ($P[i] > P_{\max}$)
(63) $P_{\max} = P[i]$;
(64) }
(65) $r = s \cdot P_{\max}$;
(66) If ($r \in$ M[]·FA)
(67) Loop happens then eliminate FA;
(68) Else
(69) {LastV= Count the member of M[];
(70) If (LastV = 2)
(71) Delete M[i];
(72) M[i] ← M[$i + 1$];
(73) M[$i + 1$] ← r;
(74) }
(75) }

Function 4: Update of source probability distribution table

(76) //update the source probability distribution table
(77) For ($i = 0$; $i = n$; i++){
(78) Check the RoutingTable · S;
(79) Find nodes which could be D simultaneously Then
(80) $P_{\mathrm{dk}} = \dfrac{9N_k - 5}{4{N_k}^2}$;
(81) Update [dp]·$S_{\mathrm{PDT}} \leftarrow P_{\mathrm{dk}}$;
(82) For other neighbor
(83) $P_{\mathrm{ik}} = \dfrac{4N_k - 5}{4{N_k}^2}$;
(84) Update [dp]·$S_{\mathrm{PDT}} \leftarrow P_{\mathrm{ik}}$;
(85) //check the link failure
(86) If ([dp]·$S_{\mathrm{PDT}} = 0$) //it means the link is lost
(87) {
(88) $Z = \dfrac{P_{\mathrm{id}}}{1 - P_{\mathrm{id}}}$;
(89) $P_{\mathrm{id}} = P_{\mathrm{id}} * (1 + Z)$;
(90) Update [dp]·$S_{\mathrm{PDT}} \leftarrow P_{\mathrm{id}}$;
(91) }
(92)}

Function 5: Pheromone calculation

(93) // procedure Calculate-Pheromone
(94) Proc Calculate-Pheromone{
(95) $E_{\min} \leftarrow$ Min.E[];
(96) $E_{\mathrm{avg}} \leftarrow$ Avg.E[];
(97) N_j = number of visited nodes by forward ant
(98) $\Delta\tau = \dfrac{1}{C - \left[E_{\min} - N_j / E_{\mathrm{avg}} - N_j\right]}$;
(99) BA·Ph ← $\Delta\tau$
(100)}

Algorithm 1: IEEABR algorithm pseudocode.

CSMA protocol, including the random waiting and back-offs. Moreover, it also supports event-based structure similar to TinyOS/NesC and hence facilitates the implementation of algorithms on real sensor nodes. The radio propagation model determines the strength of a transmitted signal at a particular point of the space for all transmitters in the system. Based on this information, the signal reception conditions for the receivers can be evaluated and collisions can be detected.

Table 1: Simulation parameters.

Parameters	Values
Routing protocols	Beesensor, EEABR, SC, IEEABR.
X_dist, Y_dist, number of nodes	1, 1, 100
Source type, center type, radius, rate, random rate	Static, random, 1, 4, 0
Destination type, center type, radius, rate, random rate	Static, random, 1, 0.5, 0
Maximum hops, data traffic	Infinity, constant bit rate (CBR)
Data rate	250 Kbps
Simulation time	100 Sec.
Nodes energy	30 Joules each
Ant Ratio, AntStart, EEABRAntStart	2, 240000, 240000

The signal strength from the transmitter to a receiver is determined by a deterministic propagation function and by random disturbances.

We evaluated improved energy efficient ant-based routing (IEEABR) with three candidate algorithms: sensor-driven and cost-aware ant routing (SC), Beesensor, and energy-efficient ant-based routing (EEABR) algorithm using the metrics defined below (Section 6.1) based on the experimental results obtained. A preliminary and earlier version of this work has been presented and published in [20]. The evaluation of the protocols was done for two different types of application scenarios. The first application is a typical converge-cast scenario (static network) in which multiple source nodes communicate with a single global sink. The second application is the dynamic scenario which is a target-tracking application of WSNs. In this scenario, a sensor node in the vicinity of a moving target generates some arbitrary sequence of events. As the target moves out of the range of that node, it stops generating the events and another node takes over, and in this, the sink node moves randomly in the monitored area, hence some broken path is taken up by other paths so that the event can be delivered to the sink node via other alternate available paths. In both scenarios, the event has a length of 512 bits and was generated at a rate of four events per second at each source node. In our experiment, the initial network topology was a 9-sensor-node (3 × 3) grid with small random offsets. We later tested both applications on other topologies consisting of 12, 36, 49, 64, and 100 sensor nodes. The network of 36 nodes is generated by placing the nodes randomly in a square of 120 m × 120 m. The transmission radius of each node is set to 35 m, and the initial energy of each node is set to 5 J each for both application types. Other topologies are generated by scaling the square so that the average node density remains the same. Each experiment was performed for a duration of 100 seconds. The set of results recorded were averaged over ten different simulation results, and the table showing the simulation parameters is shown in Table 1.

6.1. Performance Evaluation Metrics. From several results obtained from our simulation experiments, we use the following performance metrics for clarity purpose.

Success Rate. It is defined as the ratio of total number of events received at the sink node to the total number of events generated by the nodes in the sensor network. We report it in percentage (%).

Energy Consumption. It is defined as the total energy consumed by the nodes in the network during the period of the experiment in Joules (J).

Energy Efficiency. it is a measure of the ratio of total packet delivered at the destination to the total energy consumed by the network's sensor nodes, that is, ((succes rate∗ total packet sent to the sink/total energy consumed) (Kbits/Joules).

Standard Deviation. This gives the average variation between energy levels of all nodes in the network in Joules (J).

Network Lifetime. It is defined as the difference of total energy of the network and the summation of average used energy of nodes and the standard deviation of their energy levels that is, Lifetime = (total network energy − ((used energy/total nodes) + energy deviation)). The basic motivation behind this definition, is that an algorithm should try to maximize average remaining energy levels of nodes but with a small standard deviation. We report it in percentage (%).

Latency. It is defined as the time delay of an event sent from the source node to the destination node (seconds). That is, it is the difference in time when an event is generated at the source node and when it eventually gets delivered at the sink node.

6.2. Discussion on Experimental Results

6.2.1. Comparison of IEEABR with SC, EEABR, and Beesensor. To better understand the differences between these four algorithms, we tested both algorithms using two typical applications of WSN. In all the application, each one tries as much as possible to represent real WSN deployment environments, and all nodes in both scenarios were deployed in a random fashion. This is because, in a real sensor network, node deployment cannot be controlled by an operator or human due to the environmental characteristics. In the converge-cast scenario, where the sensor nodes were randomly deployed with the objective to monitor a static event, all sources and sink are fixed, and the center of the circle is randomly selected at the start of the experiment. In this scenario, the location of the event and the position of the sink node are not known, and nodes are responsible to monitor the event and send the relevant sensed information to the sink node. The results of our experiments are shown in Table 2 and Figures 4(a)–4(g). In this scenario of fixed sensor nodes, nodes near the sink are more likely to use up their energy as they serve as

Table 2: Comparison of routing protocols in converge-cast scenario based on different metrics.

Comparison of the routing protocols based on different metrics					
Routing protocols	Latency (s)	Success rate (%)	Energy consumption (J)	Energy efficiency (Kb/J)	Standard deviation (J)
EEABR	0.0315	92.4240	16.6624	21.9656	2.6624
SC	0.0313	68.6870	14.9488	18.1955	2.9782
The proposed algorithm	0.0309	94.1920	15.5104	24.0484	2.7565
Beesensor	0.1229	90.9090	18.9696	18.9777	1.7042

Table 3: Comparison of routing protocols in target-tracking scenario for different metrics.

Comparison of the routing protocols based on different metrics					
Routing protocols	Latency (s)	Success rate (%)	Energy consumption (J)	Energy efficiency (Kb/J)	Standard deviation (J)
EEABR	0.0322	20.21	10.8032	0.7405	2.1284
SC	0.1941	12.63	40.3936	0.1238	3.0880
The proposed algorithm	0.0302	50.54	9.4064	2.1262	2.1230
Beesensor	0.0629	45.49	55.4075	0.3249	0.3463

sources and the same time as routers for the other source nodes, hence, they will be forced to periodically transmit information on behalf of other nodes. In Table 3 and Figure 5, we show the simulation results from our experiment using the target-tracking scenario. In this scenario, a sensor node in the vicinity of a moving target generates some arbitrary sequence of events; as the target moves out of the range of that node, it stops generating the events and another node takes over, and in this, the sink node move randomly in the monitored area, hence some broken paths are taken up by other paths so that the event can be delivered to the sink node via other alternate available path.

(1) *Energy Consumption.* Limited available energy which is the major problem of WSN needs more attention when designing an efficient protocol. Figure 4(a) shows the energy consumption of protocols for 9 nodes in the static network scenario, while Figure 4(c) is the energy consumption of protocols for different densities of the network for the variation from 9, 16, 36, 64, and 100 nodes. SC performs better in the lower-dense network of 9 nodes with 3% difference in performance as against the proposed algorithm, while the proposed algorithm performs better when the network grows larger. The percentage difference between IEEABR algorithm and SC when the network grows to 49 nodes is 25%. EEABR and Beesensor consumed 31%, and 29.3% of networks energy than the proposed algorithm, respectively. Lower energy consumption of SC protocol in the low density network is due to the assumption that each node in the network is equipped with a sensor to sense the location of the sink node at the beginning of the routing process, in this case, a GPS. This is not advantageous when it comes to the cost of purchasing the extra accessories for each node before implementation. It was also observed that in the densely populated network, the performance of SC protocol drops below that of IEEABR due to more nodes participating in the routing process and less flooding of ants by IEEABR. As also observed in the target-tracking scenario in Figures 5(a) and 5(c) for both fewer nodes and highly dense network of varying density, in Figure 5(a), SC protocol consumes 72.65% energy more than IEEABR algorithm, which shows a high performance in the converge-cast scenario. The poor performance of the SC protocol as compared to IEEABR is due to the dynamic network, and more control packets are flooded in the network to locate the sink position. SC finds it difficult to recalculate the location of the sink any time it changes position. While also the proposed algorithm shows a great improvement over the original EEABR with the percentage difference of 10.6%, though Beesensor has the poorest performance as it consumes 83% higher than the IEEABR. As can be seen in Figure 5(c), the percentage difference between IEEABR algorithm and SC when the network grows to 49 nodes is 60% which is a significant performance difference. The proposed algorithm with its predecessor at that point is 29.66%, and Beesensor is lower at that point but higher when the network grows to 100, with a difference of 88% as against IEEABR algorithm. Hence the proposed algorithm outperforms all the protocols in terms of low energy consumption. The high improvement is due to the reduced flooding of ants in the network and proper initialization of the routing table, while giving preference to the sink selection among the neighbors.

(2) *Energy Efficiency.* Figure 4(b) shows the energy efficiency of the protocols in static scenario, as energy consumption is an important metric to be considered when designing an efficient protocol. IEEABR and EEABR are the two best protocols in terms of energy efficiency in this scenario. IEEABR better performance is due to its low total energy consumption and high packet delivery ratio. If the loss rate is high or the packet delivery rate is low as in case of SC, it results in more route discovery processes which ultimately contribute to higher energy consumption. Another interesting observation is that Beesensor consumes far more energy than SC protocol. However, their energy efficiency figures show that SC is close to Beesensor which is clearly due to the poor packet-delivery rate of SC. In this scenario, the energy efficiency bars of IEEABR and EEABR are close to each other. On the other hand, in the target-tracking

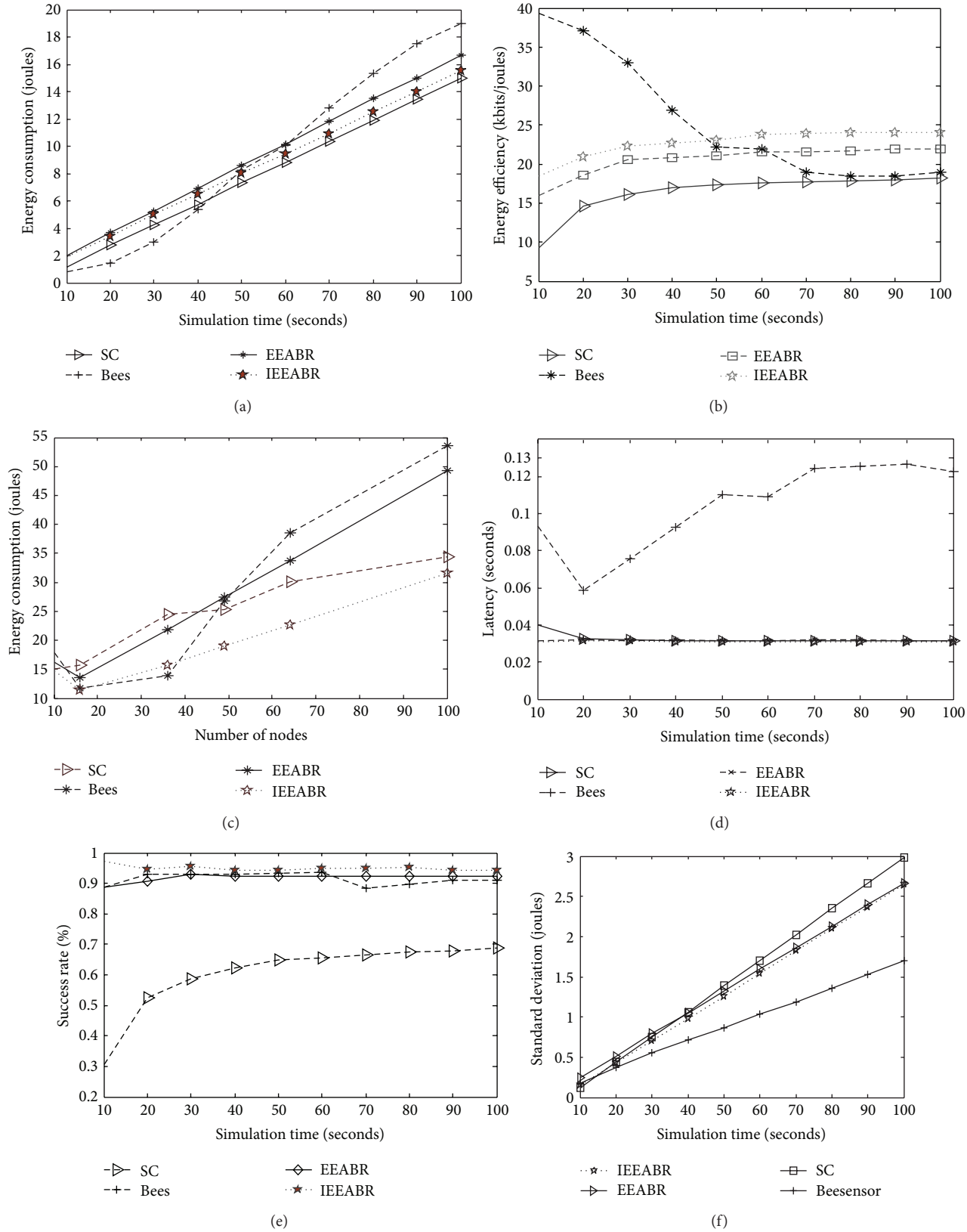
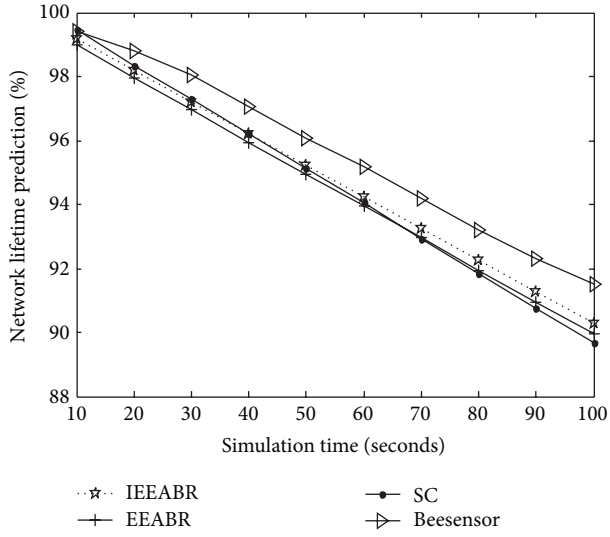


Figure 4: Continued.



(g)

Figure 4: Performance evaluation in Converge-cast scenario among four protocols: (a) energy consumption, (b) energy efficiency, (c) energy consumption for different network's densities, (d) latency, (e) success rates, (f) standard deviation and (g) network lifetime prediction.

(dynamic) application, IEEABR performs significantly better than EEABR. The reason is the ability of IEEABR to converge quickly in a dynamic scenario and achieve the high packet-delivery ratio. In the static scenario, the number of route discoveries is very small. Therefore, total energy consumption of both protocols is close to each other. However, when the number of route discoveries increases, the difference in the control overhead gets significant contributing negatively to the energy efficiency of EEABR in most of the nodes ran out of energy in Beesensor, which is the overshoot as seen in Figure 4(c). Also, if we compare the protocols in the target-tracking application, Figure 5(b) shows the energy efficiency of the protocols. It is clearly seen that, IEEABR not only having a high success rate, low energy consumption, but also is the most energy efficient among the protocols under consideration. In the converge-cast scenario, the energy efficiency bars of IEEABR and EEABR are close to each other. On the other hand, in this target-tracking (dynamic) application, IEEABR performs significantly better than EEABR. The reason is the ability of IEEABR to converge quickly in a dynamic scenario and achieve the high packet-delivery ratio. In the static scenario, the numbers of route discoveries are very small; therefore, total energy consumption of both protocols is close to each other. However, when the number of route discoveries increases, the difference in the control overhead gets significant contributing negatively to the energy efficiency of EEABR. IEEABR also outperform all the routing protocols in term of energy efficiency. The percentage difference in the target-tracking scenario between the proposed algorithm and its predecessor is 64.22%, 84.7% as against Beesensor and 93.2% for SC which is most costly in its algorithm implementation. Though all the protocols are energy aware protocols, the proposed algorithm still has a high success rate and the lowest end-to-end delay.

(3) *Success Rate.* The success rate of any protocol is the ability of the protocols to deliver successfully to the sink the packets generated at each node in the network. The success rate of the protocols in both applications is shown in Figures 4(e) and 5(d), respectively, for static and dynamic network. EEABR shows a high performance as it delivered 92.4% of all the packets generated in the network to the sink during the period of observation without much loss, whereas IEEABR algorithm having an average of 96% was the best among the protocols. SC has the poorest delivering rate in the experiment. The poor performance of the SC routing protocol is due to the path selection based on distance only without consideration of the energy of the path, in which some nodes of the paths might not be able to deliver the packets given to them for onward delivery. High packet-delivery ratio of IEEABR shows that information dissemination through multipath and link-failure management is robust in both applications of sensor networks. In the case of both converge-case and target-tracking scenario, the packet-delivery ratio of EEABR is significantly higher when compared with SC and Beesensor protocols, especially in large networks. Another important observation is the poor performance of the SC. The poor performance of the SC is due to the flooding of ants without consideration of energy of paths, and path selection is based on distance only, in which some nodes of the paths might not be able to deliver the packets given to them for onward forwarding.

(4) *Latency.* Figure 4(d) shows the end-to-end delay of the protocols under evaluation. As seen from the figure though not our priority in this work, the proposed algorithm has the lowest end-to-end delay (latency) followed by SC. Beesensor performance was worst throughout the period of observation as can be seen in the figure. The poor performance of Beesensor is due to the reactive method of route discovery of which routes have to be recalculated any time there exists an event to be sent to the sink. As IEEABR algorithm limits the number of ants flooding in the network to a fraction of 5 times the number of network nodes, while also assigning greater

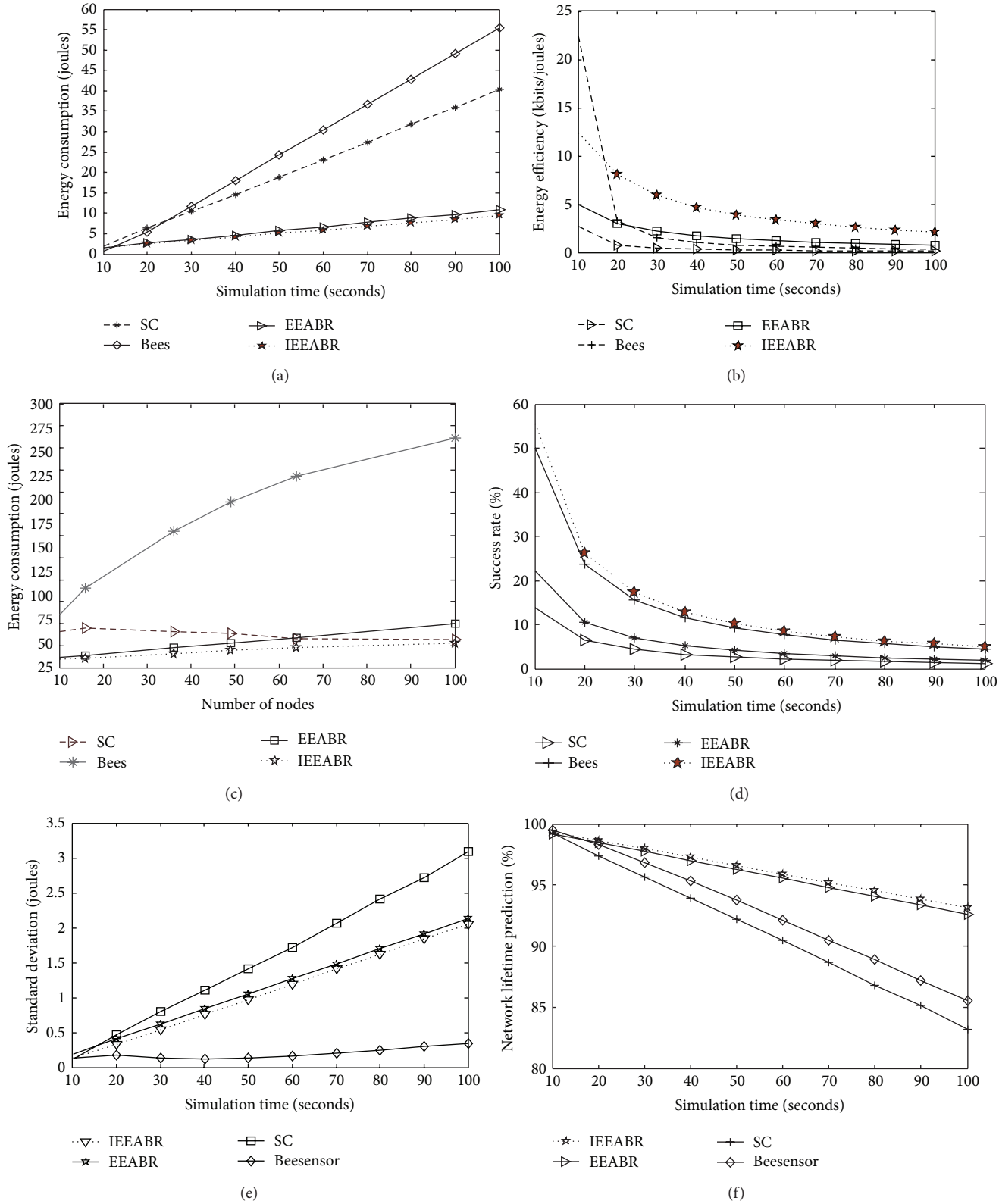


Figure 5: Performance evaluation in target-tracking scenario among four protocols: (a) energy consumption, (b) energy efficiency, (c) energy consumption for different network's densities, (d) success rates, (e) standard deviation and (f) network lifetime prediction.

probability to neighbor who falls at the same time as the sink, it performs better than all the protocols. It is also worth noting that Beesensor routes foragers throughout high-energy nodes and on demand, couple with small event cache maintained at each node which stores the events detected during the scouting process of the algorithm contributeing to its high latency.

(5) *Standard Deviation.* The standard deviation of the nodes gives the average variation between energy levels of all nodes in the network. An algorithm should try to maximize average remaining energy levels of nodes but with a small standard deviation so as to prolong the network lifetime. Figures 4(f) and 5(e) give the plots of results obtained for the four different algorithms in static and dynamic scenarios, respectively. It will be observed that Beesensor has the lowest standard deviation in both scenarios, but the performance of IEEABR as compared to SC and EEABR protocol is better in both scenarios. The lowest standard deviation attained by Beesensor is due to the fact that it is designed to operate as a reactive protocol, as such average variation between energy levels of all its nodes is significantly lower. The low standard deviation of Beesensor help, the protocol in the improvement of its lifetime even though it tends to consume high energy in the target-tracking scenario, which significantly drops it lifetime below that of IEEABR.

(6) *Lifetime Prediction.* Figures 4(g) and 5(f) show the network lifetime prediction of the algorithms in converge-cast and target-tracking applications, respectively. The result in Figure 4(g) shows that Beesensor has better lifetime than IEEABR in a static scenario but degrades in its lifetime in the dynamic scenario. This drop in performance in the target-tracking scenario is expected of Beesensor since it generates significantly higher control overhead due to its reactive path establishment and the dynamic nature of the topology in that scenario. As a result of the high control packets, much energy is consumed by the protocol and in turn decreases its lifetime, even as it has a lower standard deviation. Though, EEABR protocol has a relatively better network lifetime as compared to the SC protocol in both scenarios. The reason is because EEABR routes packets through the path with higher average energy and avoids using the shortest path regularly. Also, it should be noted that SC protocol unicasts the forward ants through the shortest path (least cost), which subsequently leads to fast depletion of energy on the shortest path and in turn results in a lower network lifetime of SC protocol. Generally, IEEABR has the highest lifetime in the target-tracking scenario and its performance is also comparable with Beesensor in the converge-cast application.

7. Conclusions and Future Work

In this paper, we have compared the performance of nature inspired state-of-the-art energy-aware routing protocols in wireless sensor networks that utilize the behavior of ant and bees in routing decisions. Our proposed routing protocol has shown and demonstrated good performance in terms of energy efficiency. The performance was not only on low energy consumption of its nodes in the network, but has low latency, high throughput, and success rate. The algorithm is capable of handling target tracking applications as well as QoS demanding applications. The proposed algorithm performs quite well in all the metrics used for evaluation purpose while also showing reasonable differences between it and its predecessor "EEABR." EEABR consumes 31% and 29.66% higher than the proposed algorithm in term of energy consumption of the nodes in the network for converge-cast and target-tracking scenario, respectively. SC which assumes that all sensor nodes have sensor to get the location of the sink did not do well as compared to the proposed algorithm. Beesensor consumes higher energy in both converge-cast and target-tracking scenario with 29.3% and 88%, respectively, due to its on-demand routing. The convergence time is low, and finds it difficult to locate sink whenever there is change in network topology. Compared with results from converge-cast scenario, the event mobility decreases the performance of the algorithm, which is understandable and expected since more nodes become sources of data packets, increasing the number of packets in the network. Once again the IEEABR protocol presents the best results when compared to the other protocols, but the results can easily be compared to scenarios where all the environment variables are static (converge-cast scenario). We are in the process of implementing the proposed algorithm on a real WSN hardware. We will improve on the algorithm based on the experience obtained from the real-time implementation and testing.

References

[1] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 325–349, 2005.

[2] 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.

[3] K. Akkaya and M. Younis, "An energy-aware QoS routing protocol for wireless sensor networks," in *Proceedings of the IEEE Workshop on Mobile and Wireless Networks*, pp. 710–715, 2003.

[4] F. Çelik, A. Zengin, and S. Tuncel, "A survey on swarm intelligence based routing protocols in wireless sensor networks," *International Journal of Physical Sciences*, vol. 5, no. 14, pp. 2118–2126, 2010.

[5] A. M. -Zungeru, L. M. Ang, and K. P. Seng, "Classical and swarm intelligence based routing protocols for wireless sensor networks: a survey and comparison," *Journal of Network and Computer Applications*, vol. 35, no. 5, pp. 1508–1536, 2012.

[6] A. M. Zungeru, L.-M. Ang, and K. P. Seng, "Termite-Hill: routing towards a mobile sink for improving network lifetime in wireless sensor networks," in *Proceedings of the 3rd International Conference on Intelligent Systems, Modelling and Simulation (ISMS '12)*, pp. 622–627, 2012.

[7] M. Roth and S. Wicker, "Termite: ad-hoc networking with stigmergy," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '03)*, pp. 2937–2941, December 2003.

[8] B. Hölldobler and E. O. Wilson, *The Ant*, Harvard University Press, 1990.

[9] A. M. Zungeru, L. M. Ang, and K. P. Seng, "Termite-hill: performance optimized swarm intelligence based routing algorithm for wireless sensor networks," *Journal of Network and Computer Applications*, vol. 35, no. 6, pp. 1901–1917, 2012.

[10] A. M. Zungeru, L.-M. Ang, and K. P. Seng, "A formal mathematical framework for modeling and simulation of wireless sensor network environments utilizing the hill building behaviors of termites," *Simulation*, 2012, Special issue: agent-based Modeling and Simulation of Complex Adaptive Communication Networks and Environments (CACOONS).

[11] L. Buttyan and J.-P. Hubaux, "Enforcing service availability in mobile ad-hoc WANs," in *Proceedings of the 1st ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 87–96, 2000.

[12] A. B. MacKenzie and S. B. Wicker, "Game theory in communications: motivation, explanation, and application to power control," in *Proceedings of the IEEE Global Telecommunicatins Conference (GLOBECOM '01)*, pp. 821–826, November 2001.

[13] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, London, UK, 1999.

[14] M. Dorigo and G. A. Di Caro, "AntNet: distributed stigmergetic control for communications networks," *Journal of Artificial Intelligence Research*, vol. 9, pp. 317–365, 1998.

[15] K. Li, C. E. Torres, K. Thomas, L. F. Rossi, and C. C. Shen, "Slime mold inspired routing protocols for wireless sensor networks," *Swarm Intelligence*, vol. 5, no. 3-4, pp. 183–223, 2011.

[16] M. Saleem, I. Ullah, and M. Farooq, "BeeSensor: a bee-inspired, energy-efficient and scalable routing protocol for wireless sensor networks," *Information Science*, vol. 200, pp. 38–56, 2012.

[17] T. C. Camilo, C. Carreto, J. S. Silva, and F. Boavida, "An energy-efficient ant based routing algorithm for wireless sensor networks," in *Proceedings of 5th International Workshop on Ant Colony Optimization and Swarm Intelligence*, pp. 49–59, Brussels, Belgium, 2006.

[18] Y. Zhang, L. D. Kuhn, and M. P. J. Fromherz, "Improvements on ant routing for sensor networks," in *Proceedings of the 4th International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS 2004)*,, M. Dorigo et al., Ed., vol. 3172, pp. 289–313, LNCS, 2004.

[19] K. Kalpakis, K. Dasgupta, and P. Namjoshi, "Maximum lifetime data gathering and aggregation in wireless sensor networks," in *Proceedings of the IEEE International Conference on Networking*, vol. 42, no. 6, 2003.

[20] A. M. Zungeru, L. M. Ang, S. R. S. Prabaharan, and K. P. Seng, "Ant based routing protocol for visual sensors," in *Proceedings of the Informatics Engineering and Information Science Communications in Computer and Information Science (ICIEIS '11)*, A. Abd Manaf et al., Ed., pp. 250–264, 2011.

[21] Routing Modeling Application Simulation Environment (RMASE), 2010, http://www2.parc.com/isl/groups/era/nest/Rmase/.

[22] Probabilistic wireless network simulator (Prowler), 2010, http://www.isis.vanderbilt.edu/Projects/nest/prowler/.