Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 172012, 14 pages
http://dx.doi.org/10.1155/2013/172012

Research Article

Bioinspired Load Balancing in Large-Scale WSNs Using Pheromone Signalling

Ipek Caliskanelli,[1] James Harbin,[2] Leandro Soares Indrusiak,[1] Paul Mitchell,[2] Fiona Polack,[1] and David Chesmore[2]

[1] *Department of Computer Science, University of York, UK*
[2] *Department of Electronics, University of York, UK*

Correspondence should be addressed to Ipek Caliskanelli; ic539@york.ac.uk

Received 16 February 2013; Revised 12 May 2013; Accepted 26 May 2013

Academic Editor: Danny Hughes

Wireless sensor networks (WSNs) consist of multiple, distributed nodes each with limited resources. With their strict resource constraints and application-specific characteristics, WSNs contain many challenging tradeoffs. This paper proposes a bioinspired load balancing approach, based on pheromone signalling mechanisms, to solve the tradeoff between service availability and energy consumption. We explore the performance consequences of the pheromone-based load balancing approach using (1) a system-level simulator, (2) deployment of real sensor testbeds to provide a competitive analysis of these evaluation methodologies. The effectiveness of the proposed algorithm is evaluated with different scenario parameters and the required performance evaluation techniques are investigated on case studies based on sound sensors.

1. Introduction

Wireless sensor networks (WSNs) consist of small, self-powered electronic nodes, each equipped with limited resources: embedded processors, memory, batteries, radio transceivers, and environmental sensors. WSNs are envisaged for industrial, civil, and military purposes to monitor, detect, and track events according to application requirements. Processing and energy restrictions are the major obstacles to achieving high performance in terms of service availability and quality of service (QoS). A vision to overcome the tradeoff between service availability and energy consumption uses distributed, self-organising algorithms to support the WSN hardware devices. For this purpose, we propose a bioinspired load balancing technique based on pheromone signalling mechanisms.

The highly dynamic nature of WSN applications requires self-organised, autonomous behaviour to overcome their fundamental resource challenges. Our research proposes a bioinspired solution to distribute workload evenly over the network components, balancing node energy levels. Classical load balancing techniques (in particular centralised algorithms focusing on optimal solutions) are inappropriate for WSNs, due to the changing workload dynamics and the energy costs of obtaining up-to-date state of the distributed WSN. To address those challenges, we rely on the lightweight and distributed nature of bioinspired mechanisms. In this research, a task mapping optimisation is used to manage the tradeoff between energy efficiency and event detection at run-time, maximising service availability while reducing energy consumption by restricting service times of the network components.

The main goal of this research is to demonstrate the effectiveness of a bioinspired load balancing technique based on pheromone signalling. The pheromone signalling (PS) mechanism proposed is inspired by biological processes: how social insects (bees) control and distribute responsibility to members of a hive [1, 2]. As abstract agents, individual bees have many similarities with sensor nodes (as do bee colonies with WSNs). The required similarities are in terms of individual wellbeing (bee/node) and collective welfare (colony/WSN). With that approach, we enable each node to decide whether it should be responsible for a given service request

using only information available locally. Ideally, such a distributed decision-making process would enable all services to be available when requested and balance the service load across the network to avoid excessive energy consumption by a few overloaded nodes.

The second goal of this research is to provide a clear comparison between evaluation methodologies. Finding the best experimental methodology to investigate and demonstrate the benefits of the work is always a key issue for researchers. For this purpose, evaluation of the proposed load balancing technique uses both a system-level simulation model and a real node hardware deployment, to exploit the beneficial points of each performance evaluation methodology. Advantages of simulation versus node deployment (and vice versa) are discussed and analysed for the proposed load balancing techniques.

The paper is organized as follows. Section 2 reviews the related work in the areas of task mapping and bioinspired routing protocols in WSNs. Our specific problem definition is presented in Section 3. Section 4 covers pheromone signalling based load balancing algorithms together with the required biological background. Section 5 describes the evaluation techniques and explains the objectives of system-level simulation and real sensor deployment. The paper is closed with the analysis of the experimental results in Section 6 and the main conclusion of this study in Section 7.

2. Related Work

The concept of task mapping refers to distributing responsibility for performing work across the entities of a distributed system such as a sensor network. Task mapping schemes can be static (offline) [3–5] or dynamic (at runtime). Some schemes are intended for homogeneous WSNs (with identical node hardware) and some for heterogeneous WSNs (taking advantage of the enhanced capabilities of some nodes). The control of the task mapping is also important, with some schemes requiring central coordinators to assign responsibility [6] and some allowing distributed decisions [7].

Pathak and Prasanna [8] present a static WSN task mapping solution that aims to minimise WSN energy consumption and balance energy usage across network nodes. The protocol operates using mixed-integer programming and exploits heuristics to find an acceptable solution. Zeng et al. [9] present a static mapping approach based on a Genetic Algorithm (GA), which aims to improve response time and limit energy usage. However, this approach results in overloading, as the mapping cannot adapt effectively to network conditions. Jin et al. [10] use a GA with a fitness function that considers network lifetime as well as the time taken to execute task sets. This dynamic approach balances energy usage while extending the network lifetime. Miorandi et al. [11] also present a genetic approach, involving genome mutation and crossover. BTMS [12] is a scheme for homogeneous networks inspired by zygote differentiation that aims to improve network lifetime and speed up task mapping and scheduling. Nodes begin in a default state and then dynamically differentiate to perform distinct tasks according to their location. DNRS [13] is an artificial immune system scheme which aims to limit energy consumption while retaining event detection reliability.

The problem of distributed task mapping in WSNs also has similarities to the research traditions of WSN cluster formation and dynamic topology control [14]. One key difference here is that the load balancing algorithm described in this paper is targeted at the application layer. Our load balancing nominates the nodes which respond to sensed events, rather than controlling routing (network layer) and MAC (link layer) activity.

Load balancing research in the WSN MAC layer has traditionally focused on clustering schemes, in which the protocol selects cluster head nodes as regional coordinators to bear responsibility for a system task. In LEACH [15], a form of dynamic cluster selection is presented in which nodes periodically rotate cluster head responsibilities to balance their energy consumption. Nodes probabilistically become cluster heads with probabilities governed by their remaining energy. Other nodes transmit data to the cluster head first, and cluster heads can be organised hierarchically to assist with delivery back to the sink. Therefore, nodes with the highest remaining energy assume more often the burden of routing and aggregating messages from their peers. In HEED [16] the residual energy of a node is also the primary factor in cluster nomination decisions; however, power levels upon cluster reception are also considered in order to improve the decisions made. PEGASIS [17] improves on LEACH by avoiding duplication of transmissions between cluster nodes and introduces aggregation of data at the cluster heads.

Firefly protocols propagate control messages to synchronise and coordinate network functions across a region, for example, clock synchronisation [18]. Heartbeat protocols are used to verify liveness and reachability of remote nodes (via a request-acknowledgment cycle) when dealing with distributed processes or nodes that may fail [19]. Gossip protocols are used for probabilistic data distribution at the network layer [20]. Our protocol incorporates propagation of control messages (hormone) across multiple hops, but in contrast to firefly protocols this distribution is limited in range. The decisions our protocol takes as a result of hormone reception are also fully deterministic, unlike in gossip protocols. Our protocol also does not require heartbeats for liveness verification, since due to hormone expiry remaining nodes will take on the queen function.

The organising metaphor of biological systems containing collective motion has been useful in developing general algorithms for distributed systems and searches over large problem sets. This comprises the field of swarm intelligence (SI). Cases studied include flocks of birds, shoals of fish [21], herds of sheep [22], and bacteria colonies [23]. These swarms are characterised by a large number of simple agents working together to collectively obtain useful solutions in terms of high performance efficiency. Collective motion changes the social network structures and establishes social ties between the individuals [24]. Groups of animals such as shoals of fish increase individual and group wellbeing by synchronising their motion. Conforming to this swarm metaphor, the artificial bee colony algorithm (ABC) has been explored in the solution of optimisation problems [25]. In this model, bees

represent search agents and their environment the space of potential solutions, with high-quality candidate solutions representing a pollen source that serves to encourage further exploration of the region by additional bee agents.

In the networking context, protocols have been developed in which network packets are treated as biologically inspired agents. In the Beehive protocol [26], packets search for efficient routes through an IP network in a process modelled after the foraging behaviour of bees. Similar work targeted specifically at WSNs is Beesensor in which routing is performed via classes of packets following different types of bee behaviour: for example as scouts and foragers. The redundancy introduced by Beesensor [27] is capable of increasing the proportion of delivered packets compared to AODV [28], although it experiences increased latency due to the possibility for bee packets to select suboptimal routes during exploration. A general framework through which a set of biological agents can attempt to simultaneously satisfy multiple possibly conflicting objectives (such as latency, energy efficiency, and delivery success in a WSN) is provided in MONSOON [29]. Previous work has also mapped the insect colony model more directly to WSN hardware, with individual nodes representing individual insects, status within the hive corresponding to node responsibilities, and signalling chemicals corresponding to data packets. Recent work has applied bee protocols specifically to WSN load balancing [30]. The protocol presented in this paper covers similar ground in allocating queen bees to fulfil sensing tasks, although it provides a different interaction process which features queen bees mating with workers in order to determine the future queens. The load balancing approach contained in this paper has several differences from the related work. Firstly, protocol behaviour is independent of the energy of particular nodes, depending only on their geographic placement and topology structure. Although the remaining energy parameter is transparently available in simulations, in hardware it is difficult to exactly assess available energy from the battery voltage obtained in low-cost WSN devices [31], so removing the dependence on it is an advantage. Secondly, our protocol also provides a stability property since a lone node (in the absence of peers) will always eventually activate and remain active providing service, unlike in protocols based upon probabilistic activation.

3. Problem Statement

This paper considers two main challenges. The first is to maximise service availability while minimising energy consumption, and to achieve that goal we propose a pheromone-based load balancing algorithm. The second challenge is to contrast different evaluation methodologies (simulation and test hardware deployment), providing a comparative analysis between those evaluation concepts. We now define the metrics and case study we will use throughout the paper.

Service availability is defined as the number of services that are successfully completed, over the total number of requested services within a period of time. A service is composed of a number of intercommunicating tasks; therefore, a service is completed only if all its tasks are executed by the WSN nodes. Task mapping therefore refers to balancing the load over network nodes, that is, deciding which node should execute the tasks of each requested service. Therefore, a service will not complete if at least one of its tasks is, firstly, not mapped to any node, or secondly, mapped to a node that runs out of energy whilst executing it.

In this paper, a sound-sensing network is used as a case study. Recording and processing every sound captured by a sensor node is considered a service, and the load balancing objective in this case study is to process all sounds in an energy-efficient way. The proposed dynamic load balancing technique introduces some redundancy in order to sustain a high level of service availability, but the level of redundancy is controlled in order to minimise energy dissipation.

4. Pheromone-Signalling-Based Load Balancing Algorithm

The load balancing mechanism proposed here allows nodes to decide whether they are willing to provide a service or not. To cope with the challenges listed in Section 1, the mechanism should be completely decentralised and have low computation and communication overheads. Research in social insects [2] has uncovered mechanisms that can achieve robust allocation of resources amongst large numbers of entities by making distributed decisions based on local information. Changes in pheromone levels are used by many social animals to orchestrate the colony by assigning responsibilities to each individual. For example, Roberts [1] reports that the pheromone produced by a dead ant causes the other ants to throw it out of the nest.

Roberts also covers a problem directly relevant to this paper, namely, the process of larvae differentiation in beehives [1]. Bees have developed a special hormonal system to ensure every beehive has a queen, which maintains the stability of the colony and orchestrates the behaviour of all other bees. Throughout its life, a queen bee stimulates a pheromone called Queen Mandibular Pheromone (QMP), which makes the worker bees aware of its presence as a queen. This hormonal mechanism works as follows: the worker bees lick the queen bee and pass the pheromone to the others. If there is no pheromone passed through the worker bees, they will then consider the queen as dead. In that case, workers will select a larva to be fed with large amounts of the royalactin protein. That protein induces the differentiation of honeybee larvae into a queen. If worker bees keep receiving the pheromone, they will be aware that there is a queen bee to orchestrate the colony and will take no action towards building a new queen.

The proposed load balancing technique is inspired by the behaviour described previously. According to Table 1, the role of queen bee denotes a sensor node that is responsible for managing the execution of all service requests it receives. Throughout this paper, we will refer to such a node as queen node (QN). They are dynamically differentiated from other nodes to indicate their duties, but this is a logical concept that does not make specific assumptions about the capabilities of the queen node (it can therefore be applied to both homogenous and heterogeneous platforms). Any nodes not differentiated into QNs are referred to as Worker Nodes (WNs).

Table 1: Correlation between bees pheromone stimulation and sensor networks.

Bees and pheromone stimulation	Sensor network
Queen bee	Sensor node responsible for task mapping and execution
Worker bees	Sensor node
Pheromone level	Parameter used for queen node selection
Lifetime of bee	Operation lifetime of the sensor node

```
1  every T_QN do
2      if (h_i < threshold_QN)
3          QN_i = true
4          broadcast hd = {0, h_QN}
5      else
6          QN_i = false
```

Listing 1: PS differentiation cycle.

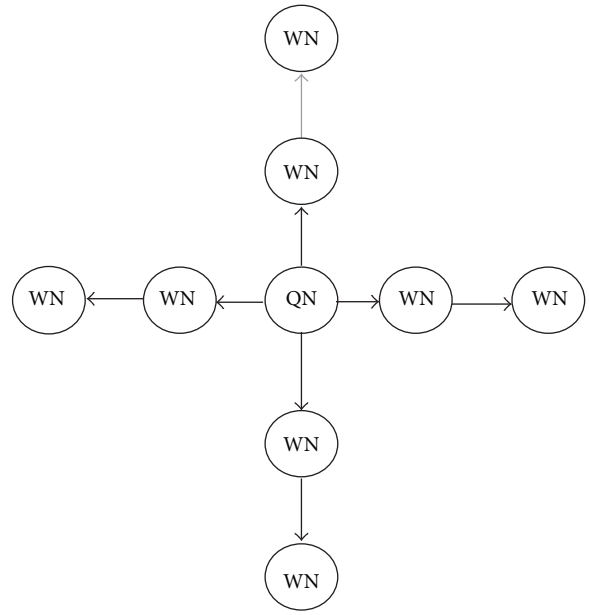

Figure 1: Pheromone propagation by workers receiving pheromone from the queen.

All QNs and WNs are capable of sensing the environment, queuing and executing tasks, and communicating with other nodes within range. However, in our approach, only QNs will voluntarily execute tasks (i.e., react to a sensed event or a service request). WNs will only execute tasks if QNs explicitly map those tasks to them, or if they differentiate themselves to become QN. The pheromone-signalling (PS) algorithm aims to enable node differentiation at a scale that produces sufficient QNs to handle all the required system functionality (e.g., service requests, event detection) either by executing those tasks themselves or mapping them to available WNs. Likewise, the algorithm should avoid unnecessary redundancy (e.g., several nodes sensing, processing, and notifying the same event multiple times).

The basic strategy of the algorithm is based on the periodic transmission of pheromone by QNs and its retransmission by recipients to their neighbours. The pheromone level at each node decays with time and with distance to the source. All nodes accumulate pheromone received from QNs, and if at a particular time the pheromone level of a node is below a given threshold this node will differentiate itself into a QN. This typically happens when this node is too far from other QNs, or when a WN exists for too long without receiving pheromone. The proposed PS algorithm consists of three parts which are executed on every node of the network: two of them are time-triggered (differentiation cycle and decay of pheromone) and one of them is event-triggered (propagation of received pheromone).

The first time-triggered part, referred to as the differentiation cycle (Listing 1), is executed by every node of the network every T_{QN} time units. On each execution, the node checks its current pheromone level h_i against a predefined level $threshold_{QN}$. The node will differentiate itself into QN (or maintain its QN status) if $h_i < threshold_{QN}$; otherwise it will become a WN. If the node is a QN, it then transmits pheromone to its network neighbourhood to make its presence felt. Each pheromone dose *hd* is represented as a two-position vector. The first element of the vector denotes the distance in hops to the QN that has produced it (and therefore is initialised as 0 in line 4 of Listing 1). The second element is the actual dosage of the pheromone that will be absorbed by the neighbours.

The event-triggered part of PS deals with the propagation of the pheromone released by QNs (as described previously in the differentiation cycle) and received at neighbouring nodes. The purpose of propagation is to extend the influence of QNs to nodes other than their directly connected neighbours. Propagation is not a periodic activity and happens every time a node receives a pheromone dose. Its pseudocode appears in Listing 2. Upon receiving a pheromone dose, a node checks whether the QN that has produced it is sufficiently near for the pheromone to be effective. It does that by comparing the first element of *hd* with a predefined $threshold_{hop\ count}$. If the *hd* has travelled more hops than the threshold, the node simply discards it. If not, it adds the received dosage of the pheromone to its own pheromone level h_i and propagates the pheromone to its neighbourhood. Before forwarding it, the node updates the *hd* vector element by incrementing the hop count and by multiplying the dosage by a decay factor $0 < K_{HOP\ DECAY} < 1$. This represents pheromone transmission decaying with distance from the source. This is shown in Figure 1, which illustrates the four WNs surrounding the QN retransmitting a lower dose of pheromone to their neighbours.

The second time-triggered part of the algorithm, shown in Listing 3 is a simple periodic decay of the pheromone level of each node. Every T_{DECAY} time units, h_i is multiplied by a decay factor $0 < K_{TIME\ DECAY} < 1$.

```
1  when hd is received
2    if (hd[1] < threshold_hop count)
3      h_i = h_i + hd[2]
4      broadcast hd = {hd[1] + 1, hd[2] · K_HOP DECAY}
5    else
6      drop hd
```

Listing 2: PS pheromone propagation cycle.

```
1  every T_DECAY do h_i = h_i.K_TIME DECAY
```

Listing 3: PS decay cycle.

It can be easily inferred from the PS differentiation cycle that each sensor node makes its own decision on whether and when it becomes a QN by referring to local information only: its own pheromone level h_i. This allows for a highly self-organised behaviour which fits the requirements for high-density networked embedded systems. The other parts of the algorithm exist to implement a simple dynamics for the propagation and decay of pheromone. The computational complexity of the algorithm is very low, as each of the parts is a short sequence of simple ALU operations. The communication complexity, which in turn determines how often the PS propagation step is executed, depends on the connectivity of the network and on the T_{DECAY}. The protocol also provides a stability property, in that a lone node with no peers will become and always remain a queen node after a given delay, unlike in other protocols where nodes may be probabilistically switched off for some intervals.

5. Evaluation Techniques

Experimentally evaluating the performance of novel algorithms is a fundamental focus of WSN research [32]. Existing evaluation concepts include system-level simulators, low-level simulators, and prototypes. Since WSNs are application-specific environments, researchers choose the best-fit concept to evaluate their target application area. Key criteria in choosing the best-fit performance evaluation technique are flexibility, scalability, complexity, implementation time, performance efficiency, financial cost, and accuracy.

Table 2 compares and contrasts three performance evaluation techniques. Design factors are marked with either L (low), M (medium), or H (high) for each performance evaluation criterion. As it is shown in Table 2, system-level simulation models are cost-efficient due to their flexibility. Financial costs of prototypes are high, whereas low-level simulation models are listed as being medium cost-efficient performance evaluation techniques. System-level simulation models are known to have short implementation duration, high scalability, and flexibility while providing high performance efficiency. Prototypes are considered as less flexible and scalable than alternatives, so are listed as low. Low-level simulation models are more accurate than system-level simulation models, because they abstract away less details than system-level simulation models. The implementation duration of low-level simulation is greater, compared to the prototypes and system-level simulation models due to their level of complexity. Although inefficient, prototypes provide the most accurate results, since they provide results from real WSN hardware and operating system environments [33–35].

In this research, we have decided to validate our approach via system-level simulation models and a small hardware prototype testbed. The important criteria that guided these decisions are cost, implementation duration, performance efficiency, and the level of accuracy. By validating our approach using different performance evaluation techniques, we aim to compare the implementation duration, performance efficiency, and the level of accuracy of the system level model versus prototype, as well as demonstrating the effectiveness of the proposed load balancing technique.

5.1. Case Study. A case study based on a surveillance multimedia target tracking application was used to evaluate the effectiveness of the proposed approach. The goal is to compare the proposed load balancing solution against existing solutions and then observe the impact of using different values for the parameters of the proposed algorithm. The case study uses a network of nodes equipped with acoustic sensors to capture sounds generated by the entities under surveillance (usually insects or birds) on a particular area [36]. Once a node captures a sound, it processes it aiming to identify its source and report its approximate location.

5.2. System-Level Simulation. The objective of evaluating the proposed work using a simulator is to investigate the long-term behaviour of the load balancing algorithm. Using simulation allows exploring the large parameter space of the load balancing technique, without the hardware and time consumption obstacles of the real sensor deployments. Unlike real sensor deployments, system-level simulation tools provide ease of use with broad applicability, which enables evaluation of long-term outcomes of the proposed technique on large scale deployments.

A three-tier WSN system model is designed to represent network components, the services that run over it and the functionality that assigns services to network nodes. Graph theory can be used to model a network and its services. The platform model, $NW = (N, L)$, consists of a set of N nodes and a set L of bidirectional wireless links between neighbouring nodes. Each node $n_m \in N$, is the tuple $n_m = \{m_m, bc_m, idr_m, cdr_m, wcdr_m\}$ representing its memory capacity in bytes, battery capacity in mAh, its battery discharge rate in μAs in idle mode, its battery discharge rate in μAs when performing a computation, and its battery discharge rate in μAs when transmitting a byte of data over its wireless interface. We use such parameters to determine, for a given assignment of services to nodes, how much energy is dissipated by each node and, over time, which nodes are still alive (i.e., have dissipated less than their battery capacity). As the network topology is represented by the

Table 2: Comparison between the performance evaluation techniques.

Models	Cost	Scalability	Flexibility	Accuracy	Complexity	Efficiency	Time
System-level simulator	L	H	H	L	L	H	L
Low-level simulators	M	M	M	M	H	M	H
Prototypes	H	L	L	H	M	L	M

set of links $l_{mn} \in L$, the cost of multihop transmission of each communication c_{ij} can also be taken into account if the routing algorithm used in the network is known. A service is a logical concept that denotes a particular subset of the systems functionality. A service is provided by one or more network nodes and can be requested or triggered by end users, other nodes or even the environment. For example, a simple service could be to provide the end-user with a temperature reading from a particular location within the area covered by the system. A complex service, on the other hand, could include a series of tasks that can be executed by multiple nodes, for example the calculation of maximum, minimum, and average temperatures of that particular area. In this paper, we focus on complex services that are composed of multiple tasks. We represent a service S as a directed acyclic graph (DAG), with nodes representing tasks and edges representing intertask communication: $S = (T, C)$. Each task, $t_i \in T$, is a tuple $t_i = \{n_i, mf_i, e_i, et_i\}$, where n_i is the supplier node, mf_i is its memory footprint in bytes, e_i is the energy consumption of the task, and et_i, is its execution time. Each intertask communication, $c_j \in C$, is also a tuple $c_j = \{s_j, r_j\}$, where $s_j \in T$ is the sender task and $r_j \in T$ is the receiver task of the communication. For the proposed framework, the mapping process is defined as a function from the application domain to the platform codomain and is represented as $F : T \rightarrow N$.

An event-driven simulator has been designed to implement this model. A UML diagram of its execution sequence is shown in Figure 2. It is controlled by the JavaSim library [37] and is validated with 30 different task sets. The simulator is sufficiently scalable as to be able to simulate hundreds of nodes, and some of the results have been obtained on a grid topology of 28×28 nodes.

5.3. TinyOS Experimental Testbed. The experimental testbed in Figure 3 is intended to evaluate the short-term behaviour of the protocol. It consists of 16 homogeneous nodes (MEMSIC Iris nodes with 2.4 GHz transceivers) together with a base station that serves to receive results and transfer them via USB to a monitoring computer.

These nodes run on the open-source TinyOS operating system version 2.1 [38]. A custom modular application was developed to perform multihop forwarding, sound detection and to implement the pheromone algorithm. Message delivery was performed using the TinyOS Active Message layer. Duty cycling MAC protocols are out of the scope of this work. However, the application layer also applies a randomised forwarding delay before packet dispatch, in order to reduce the impact of collisions when simultaneous detection would otherwise lead to a sudden burst of event generation.

Hardware nodes are arranged into a 4×4 regular grid and perform multihop routing in order to reach the sink node. Routing is preconfigured with nodes forwarding hop-by-hop on fixed multihop relaying chains towards the sink node with ID 1, as depicted in Figure 4. The intent of this shortest path routing in preconfigured chains is to simulate a standard forwarding protocol applied in a simple, known test deployment, in regular terrain, avoiding the complexities of route setup/teardown.

During the experimental deployment, nodes are located sufficiently close for packet transmission/reception to occur across the entire network. The simulation scenario assumes that nodes can only communicate directly with their immediate neighbours, to emulate the conditions of a real large-scale deployment. Although, since Iris nodes typically achieve transmission ranges up to 50 m indoors [36], it is necessary to restrict the communicating nodes in software so packets from nodes that are not one-hop neighbours within the topology are rejected.

To evaluate the performance of the protocol, a timer in the application layer originates a sequence of fixed trigger events, corresponding to detections of a periodic sound source in the environment. The pheromone algorithm is executed as described in Section 4, assigning statuses of QN and WN to nodes dynamically (changing as execution proceeds). Queen nodes respond to these periodic detections by transmitting an event notification, while worker nodes ignore these application layer events. Further down the protocol stack, multihop forwarding is then used at all nodes of the routing chain (queen and worker alike) to relay data on event detections and packet transmissions back to the sink node for analysis at the monitoring computer. A baseline experiment is also performed in which the pheromone protocol is not executed and all nodes report sensed events, in order to assess the advantages of the load balancing protocol.

5.4. Parameter Choices. Several important energy parameters of our experimental platform are shown in Table 3. Additionally, in the experimental results, we also consider the importance of pheromone propagation period and pheromone decay period.

6. Experimental Results

This section presents the experimental results. The goal of the hardware experiments in Section 6.1 is to demonstrate the behaviour of the system on a small scale, exhibiting performance advantages on a real sensor deployment on TinyOS-operated system. The intent of the system-level simulation results in Section 6.2 is to evaluate load balancing performance of the pheromone algorithm on a large scale, including lifetime issues and their effect upon performance.

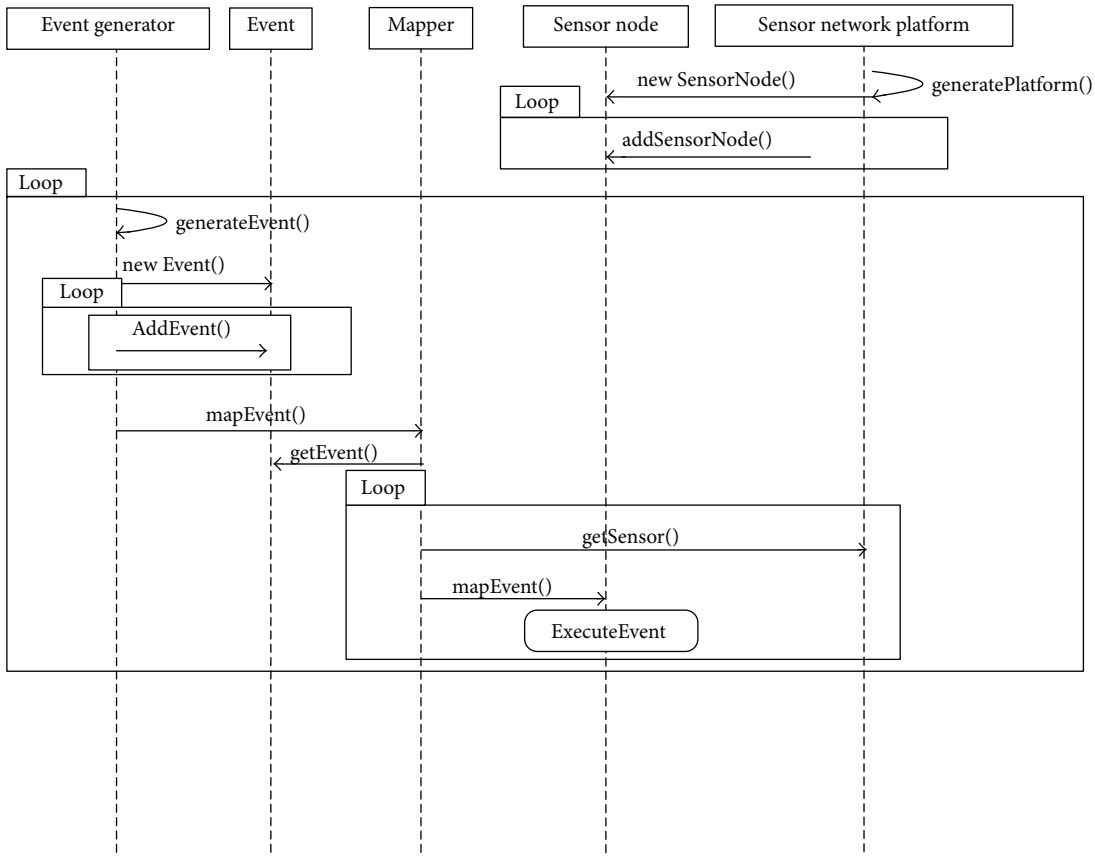

Figure 2: Execution sequence of system-level simulator.

Figure 3: Iris nodes used in the 4 × 4 grid hardware deployment.

Table 3: Energy-related parameters.

Configuration parameters	Platform model
Battery capacity (mAh)	1500
Idle discharge rate (μAs)	300
Task computation discharge rate (μAs)	3000
Wireless communication discharge rate per byte at 30 kbps (μAs)	0.6

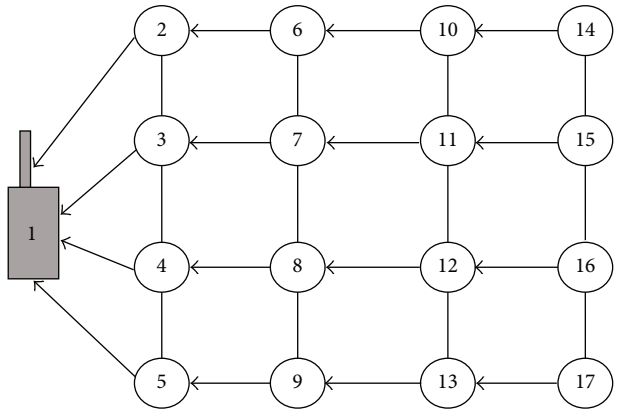

Figure 4: The network topology with preconfigured multihop routing chains.

6.1. TinyOS Hardware Testbed Experiment Results. Figure 5 shows the total number of event detections received over time and the number of packets transmitted in the network in total. Results are measured for the pheromone signalling algorithm, compared to a baseline case with no load balancing. An event covering the entire network occurs at 600 ms time intervals. Thus the smaller (nonzero) number of event detections the better, since this represents minimal duplication. The results demonstrate that following stabilisation (after 40 s) the load balancing algorithm produces a significantly smaller number of detections, reducing the

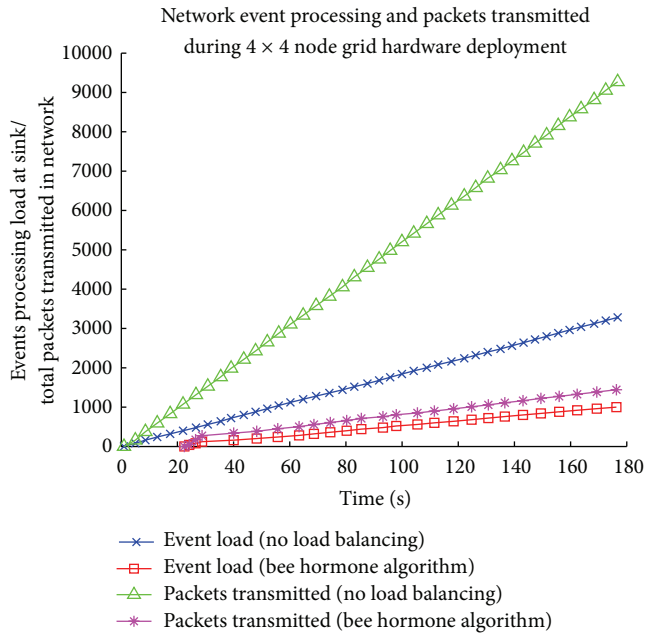

Figure 5: Event detections and packets for load-balancing pheromone algorithm.

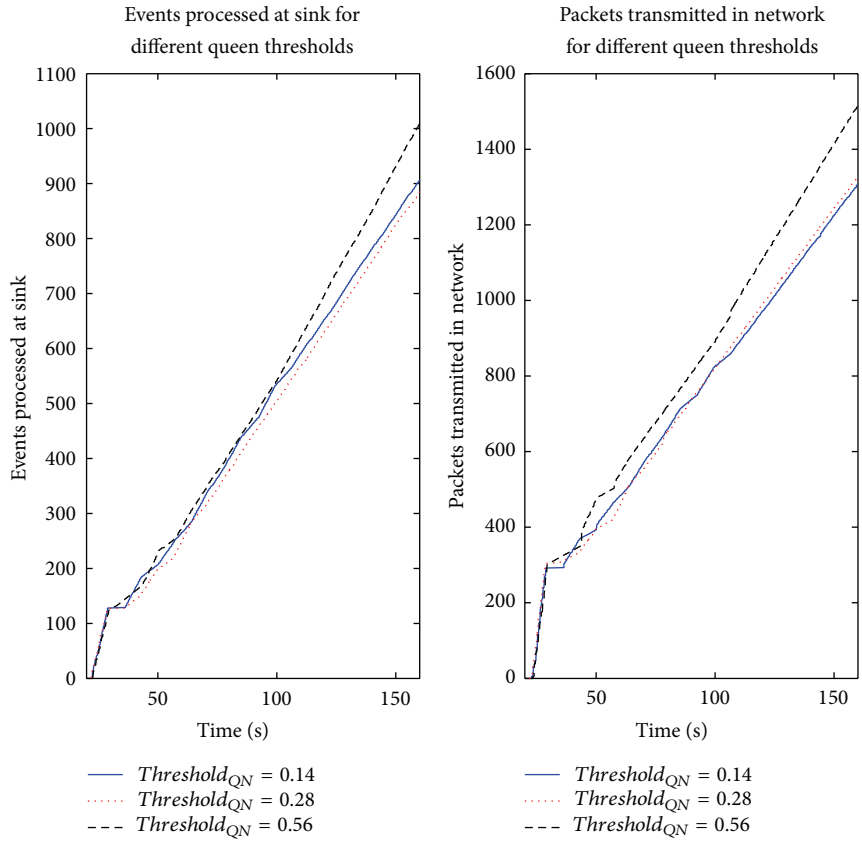

Figure 6: Event detections and packets for different queen thresholds.

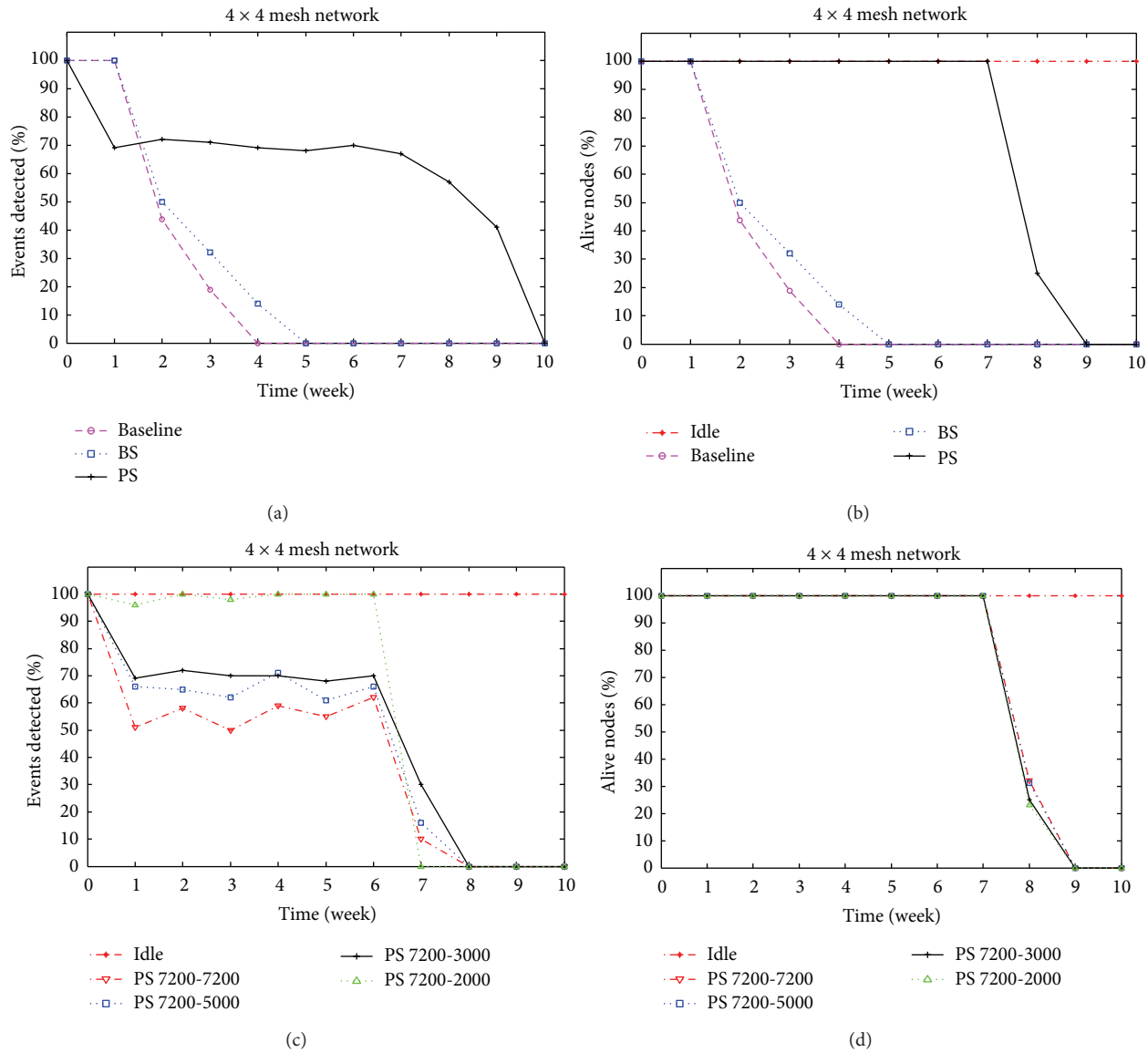

FIGURE 7: Experimental results: on 4 × 4 mesh network topology (a) % events detected, (b) % alive nodes, effects of different pheromone decay period (c) % events detected, (d) % alive nodes.

total event load to approximately a third. The reduction in packet transmission load is even more significant, given that preventing duplicate events being registered avoids the additional routing load that these events generate in other network nodes. There is an initial delay beginning event detection until after 20–40 seconds as the network stabilises and a suitable number of nodes become QNs.

Figure 6 shows the impact of queen pheromone threshold $threshold_{QN}$ upon the measured event processing and packet transmission load. These series are not greatly influenced by doubling the queen threshold from 0.14 to 0.28. However, if the queen threshold is doubled again to 0.56, then the pheromone algorithm tolerates a stable state with additional queens in the network. This leads to an approximately 10% increase in the total redundant event processing. Total packet transmissions also increase approximately by 16% due to processing these events, and the additional Pheromone propagations of the extra-queens. This illustrates that if aggressive load balancing for energy efficiency is the priority, then queen threshold should be minimised. However, if redundancy in event detection is preferred, then large queen thresholds are acceptable.

6.2. System-Level Simulation Results. This set of the experimental work aims to compare the proposed technique (PS) against three scenarios.

(1) *Idle* represents the absence of load, and all nodes of the system do not dissipate any energy on computation or communication with the neighbours. It is included as the maximum lifetime of the system if no surveillance is performed.

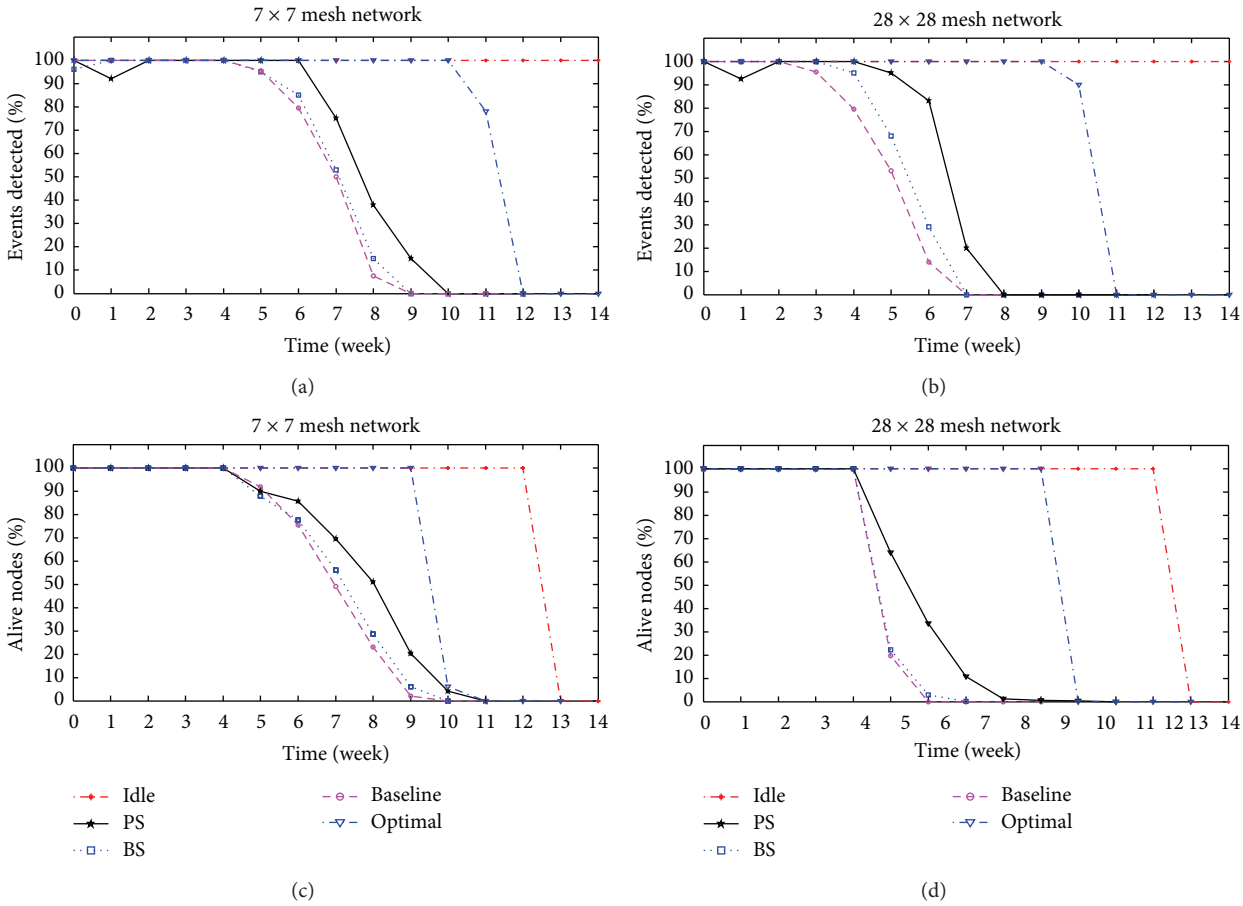

Figure 8: Experimental results: effects of PS algorithm on 7 × 7 and 28 × 28 mesh network topologies showing (a), (b) % events detected, (c), (d) % alive nodes.

(2) *Baseline* represents the execution of the case study without any load balancing support.

(3) *BS* represents the execution of the case study using load balance based on existing energy-aware task migration mechanisms as in [39, 40].

(4) *Optimal* represents an artificial scenario for WSNs where each service is executed by only one service provider to ensure that no redundant processing takes place and minimum number of network resources is used.

The comparison is based on 30 different soundscape scenarios running over ten different network configurations (4×4, 7×7 and 28×28 topologies and the five alternatives for load balancing listed previously), in a total of 450 simulation runs. Each run simulated the case study for 14 weeks, which is the point when all network nodes run out of energy even in the IDLE scenario. Figure 7 shows the results for service availability and alive nodes over time, plotting the average of the 30 runs for each configuration.

According to Figures 7(a) and 7(b), the percentages of detected events and alive nodes are lowest in the Baseline scenario. In the Baseline and BS scenarios all the nodes are allowed to execute events unlike the PS scenario. As a result of high redundant executions in both scenarios, network lifetime is short compared to the PS. BS scenario applies execution restriction only when nodes energy is lower than static threshold. In this case nodes apply task migration and as a result the percentage of alive nodes is increased, as well as the percentage of detected events. The major improvement is shown by the PS scenario. The detected event percentage remains constant after the first week; however, the percentage of the detected events dramatically drops in Figure 7(a) during the first week. PS algorithm limits the redundant network progress by allowing pheromone/QN procedure and the algorithm performs better in larger-scale networks where the number of the network resources is higher. In small networks like 4 × 4, pheromone stabilisation over the network takes more time due to the lower number of redundant network resources. Besides the redundant network resources, finding values for the PS parameters to provide higher service availability are more difficult to tune since pheromone propagation is limited to the small number of network resources. As a result of difficulties of bounded pheromone propagation (over the small networks) limits the network with low redundancy and it forces higher proportion of the network resources to be active in small network and makes PS more aggressive towards achieving high performance metrics.

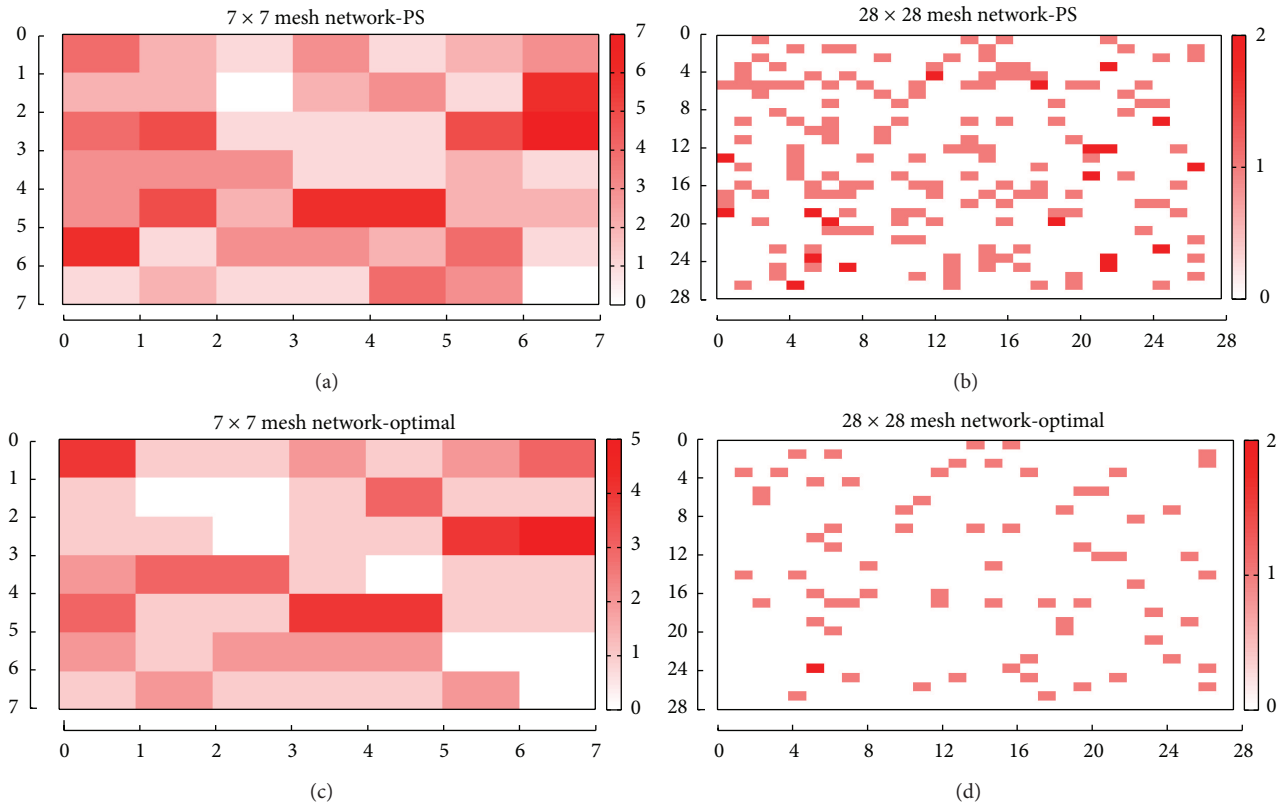

FIGURE 9: Experimental results: distribution of dropped events over 7 × 7 and 28 × 28 mesh network topologies showing (a), (b) on PS scenario, (b), (d) on Optimal scenario.

As a result of the low level of redundancy and less duplicated event executions on 4 × 4 network, there is a dramatic drop in percentage detected events after the first week illustrated on Figure 7(a). However, the required restriction on redundancy affects the network lifetime in a positive way and network lifetime improves dramatically according to Figure 7(b). As the nodes are balanced in terms of usage, their remaining energy levels are quite similar to each other. As a result, the percentage of alive nodes does not drop dramatically and remains steady until the end of week 7. At the end of week 7, 70% of the nodes run out of energy at the same time. This convincingly demonstrates that the load was successfully balanced over the nodes. The number of nodes is calculated at the end of each time interval, whereas detected events (service availability) are calculated for individual time intervals.

Sensitivity analysis is partially applied to the proposed technique to show the impacts of the PS parameters as mentioned in Section 5. A variety of pheromone decay interval is evaluated on our PS technique to show the effects of the required parameters on percentage of detected events and alive nodes on a 4 × 4 network in Figures 7(c) and 7(d). Simulation duration is also improved to illustrate the longer-term effects of the approach on the same topology. T_{DECAY} time unit plays an important role in performance. As the T_{DECAY} time unit shortens, the number of QNs increases. As a result, the number of detected events increases. Since the number of QNs affects energy use, the energy consumption of the network increase as well, whereas longer T_{DECAY} time parameters allow lengthening of the network lifetime.

Figures 7(c) and 7(d) show the number of alive nodes (i.e., with nonzero battery levels) over time for each network configuration. While there is an apparent correlation between these curves and the service availability curves discussed before, this is not necessarily the case. For instance, if events are concentrated in a particular area of the network, nodes in that area will be out of energy very quickly and service availability will drop, even if the number of nodes in other areas of the network is still high. By avoiding overloading and unnecessary redundancy, the proposed approach extended network lifetime, mean lifespan for every sensor node on average, by at least a week in both cases.

In Figure 8 the effects of PS algorithm on 7 × 7 and 28 × 28 mesh network topologies have been illustrated and results have been compared with the idle, Baseline, and BS scenarios as well as the Optimal scenario. Figure 8(a) shows the percentage of event detection and Figure 8(c) shows the percentage of alive nodes on 7 × 7 Mesh Network. Similarly, Figures 8(b) and 8(d) illustrate the same performance metrics on 28 × 28 Mesh Network. According to both Figures 8(a) and 8(b), the PS scenario achieves the highest percentage of event detection rate in both 7 × 7 and 28 × 28. In both networks during the first week the percentage of event detection rate of PS algorithm slightly drops. The reason behind the required performance drop is due to the pheromone stabilisation over

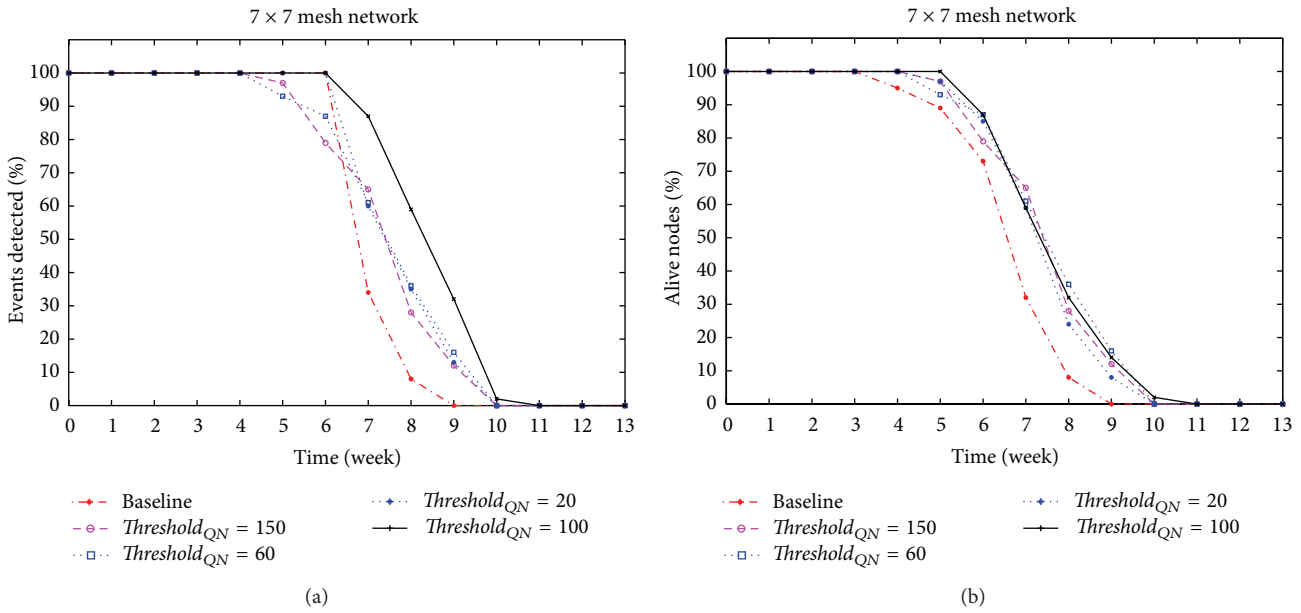

FIGURE 10: Experimental results: (a) % events detected, (b) % alive nodes for PS Load Balancing with different values for $threshold_{QN}$.

the network, which required some time. However, it recovers in the following week.

To show the network coverage of the PS technique, it is important to analyse the distribution of QNs and illustrate the number of dropped events. Figures 9(a) and 9(b) show the number of dropped events per node on PS technique, whereas Figures 9(c) and 9(d) show the number of dropped events per node on Optimal scenario. In PS technique, we allow some level of redundancy in order to ensure high level of service availability. This means that some services are allowed to be executed by more than one sensor node. In the case where a service is allowed to be executed on more than one service provider, an event is considered as detected if it has been captured by at least one service provider. On the other side, an event is considered as dropped if it has not been captured by any of the service providers responsible for the required service. In the Optimal scenario, each service is executed by best possible located in the sensor field to represent an optimistic, artificial case of sensor networks. According to Optimal scenario, if particular service provider is not able to detect, then event is considered as event dropped by that service provider.

In Figure 9(a) maximum number of the dropped events is higher Figure 9(b), which gives hints about the level of applied redundancy in PS technique. Results of the number of dropped events per node are as not low as Optimal scenario which is natural. However, the difference between the number of dropped events per node on Figures 9(a) and 9(c) is very small, which also shows the applied level of redundancy on PS algorithm as very restricted. On the other side, in both Figures 9(a) and 9(c), distribution of the dropped events over the network is almost equal, which then shows us the network coverage of PS algorithm.

Figures 9(b) and 9(d) show the number of dropped events per node on 28 × 28 mesh network. Both 7 × 7 and 28 × 28 mesh networks have executed the same number of events; however, redundancy in 28 × 28 mesh network is much than 7 × 7 mesh network. As mentioned earlier, PS algorithm performs better in large networks where it limits the redundant resource allocation low. Figures 9(a) and 9(b) support this argument, where in (b) the maximum number of dropped event per node is less than (a).

The second part of the experimental work aims to observe the impact of changes on the algorithm parameters defined in Section 5. Figure 10 shows a single scenario of the surveillance application managed by different configurations of the PS algorithm, each of them with a different value for $threshold_{QN}$. That parameter is used on PS differentiation cycle and determines whether a given node should change into (or stay as) queen node. With higher values for $threshold_{QN}$, nodes are more likely to become QNs, because they would have to receive significant amounts of pheromone from the neighbours to prevent the differentiation. With a lower threshold, few nodes will become QNs, as even small quantities of hormone from a single neighbour could prevent differentiation. However, since there is no guarantee that the differentiated QNs will be regularly distributed over the network and will match the pattern of sound events of a given scenario, it is not trivial to find the right value. Figure 10 shows that each alternative may produce different variations on service availability during the early part of the system lifeline and different degradation patterns during the end of life. To make matters worse, other parameters also have an impact on the metrics of interest, as shown in Figure 10. By changing T_{QN} and T_{DECAY}, it is possible to significantly extend network lifetime at the expense of service availability guarantees.

7. Conclusion

This paper had two major goals: solving the service availability versus energy consumption tradeoff with the proposed algorithm and demonstrating the good performance of our algorithm via the two evaluation methodologies of system-level simulation and hardware deployment. We demonstrated the long-term performance benefit of the proposed technique via a system-level simulation model. The short-term energy efficiency benefits of our load balancing technique have been evaluated on a real sensor deployment. The advantages and disadvantages of these two performance evaluation methodologies have been highlighted.

This paper has proposed a novel load balancing algorithm based on a pheromone-signalling mechanism. This distributed and asynchronous task mapping protocol has been shown to allow WSNs to balance service load across nodes, achieving increased energy efficiency without significantly sacrificing service availability. Extensive system-level simulation results have shown that our technique provides longer network lifetime, increasing the service availability over longer time scales consistent with a real deployment. Our proposed technique delivers 10% longer network lifetime on average and up to 85% higher service availability in later stages of the system lifetime. The experiments have also shown that the proposed algorithm is highly parameterisable, giving system designers the flexibility to choose different points over the tradeoff between service availability and network lifetime.

Hardware results for a 4×4 grid with multihop routing have demonstrated a corresponding reduction in duplicate event detection count (to approximately a third of the baseline event detections) and total packet transmissions. This equates to a substantial energy efficiency benefit. The impact of queen threshold levels has also been studied in hardware, verifying that a small threshold of 0.14 provides 10% fewer duplicate detections than 0.56. Moreover, it is important to compare the performance evaluation concepts used. As noted earlier, three important factors are cost, implementation duration, performance efficiency, and the level of accuracy provided. Costwise, it was expensive and time consuming to obtain, debug, and configure the sensor nodes for the real sensor deployment, whereas we used open source tools to develop a system-level simulation model that could be flexibly reconfigured to model different scenarios quickly.

Additional research is currently under way on mechanisms to automatically explore the parameter space of the algorithm, aiming to find for a particular network configuration what parameters can fulfil given requirements on lifetime or service availability. Another possibility is dynamic parameter tuning, aiming to increase the robustness of the algorithm, but we are aware that the communication and computation overheads of such approach will not be negligible and may affect its effectiveness. Another area of useful future research is the application of the pheromone-signalling algorithm to load balancing in networks with irregular topologies and mobile nodes, in which its lightweight, local decision making would likely prove advantageous. Finally, we would like to exploit this type of technique on upcoming large on-chip multicore systems, because in the near future they will present a level of core density that will pose the same challenges listed in Section 1.

Acknowledgment

The authors would like to thank Professor Leocadia Falkemberg Indrusiak for the insightful discussions on leader selection in animal groups, which provided the biological inspiration for the work presented here.

References

[1] M. B. V. Roberts, *Biology: A functional Approach*, Nelson, 1986, http://www.worldcat.org/isbn/0174480199.

[2] J. B. Free, *Pheromones of Social Bees*, Comstock Publication Associates, 1987.

[3] S. Shivle, H. J. Siegel, A. A. MacIejewski et al., "Static allocation of resources to communicating subtasks in a heterogeneous ad hoc grid environment," *Journal of Parallel and Distributed Computing*, vol. 66, no. 4, pp. 600–611, 2006.

[4] T. Braun, H. Siegel, and A. Maciejewski, "Static mapping heuristics for tasks with dependencies, priorities, deadlines, and multiple versionsin heterogeneous environments," in *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS '02)*, pp. 78–85, 2002.

[5] G. Martinovic, L. Budin, and Z. Hocenski, "Static-dynamic mapping in heterogeneous computing environment," in *Proceedings of the IEEE International Symposium on Virtual Environments, Human-Computer Interfaces and Measurement Systems (VECIMS '03)*, pp. 32–37, July 2003.

[6] H. S. Abdelsalam and S. Olariu, "Toward efficient task management in wireless sensor networks," *IEEE Transactions on Computers*, vol. 60, no. 11, pp. 1638–1651, 2011.

[7] M. Üney and M. Çetin, "Graphical model-based approaches to target tracking in sensor networks: an overview of some recent work and challenges," in *Proceedings of the 5th International Symposium on Image and Signal Processing and Analysis (ISPA '07)*, pp. 492–497, Istanbul, Turkey, September 2007.

[8] A. Pathak and V. K. Prasanna, "Energy-efficient task mapping for data-driven sensor network macroprogramming," *IEEE Transactions on Computers*, vol. 59, no. 7, pp. 955–968, 2010.

[9] Z. Zeng, A. Liu, D. Li, and J. Long, "A highly efficient DAG task scheduling algorithm for wireless sensor networks," in *Proceedings of the 9th International Conference for Young Computer Scientists (ICYCS '08)*, pp. 570–575, Hunan, China, November 2008.

[10] Y. Jin, D. Wei, A. Gluhak, and K. Moessner, "Latency and energy-consumption optimized task allocation in wireless sensor networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '10)*, pp. 1–6, Sydney, Australia, April 2010.

[11] D. Miorandi, L. Yamamoto, and P. Dini, "Service evolution in a bioinspired communication system," *International Transactions on Systems Science and Applications Journal*, vol. 2, no. 1, pp. 51–60, 2006.

[12] Y. E. M. Hamouda and C. Phillips, "Biological task mapping and scheduling in wireless sensor networks," in *Proceedings of the IEEE International Conference on Communications Technology and Applications (ICCTA '09)*, pp. 914–919, Beijing, China, October 2009.

[13] B. Atakan and Ö. B. Akan, "Immune system based distributed node and rate selection in wireless sensor networks," in *Proceedings of the 1st Bio-Inspired Models of Network, Information and Computing Systems (BIONETICS '06)*, pp. 1–8, Madonna di Campiglio Trentino, Italy, December 2006.

[14] P. Santi, "Topology control in wireless ad hoc and sensor networks," *ACM Computing Surveys*, vol. 37, no. 2, pp. 164–194, 2005.

[15] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Siences (HICSS-33 '00)*, p. 10, January 2000.

[16] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, 2004.

[17] S. Lindsey and C. Raghavendra, "Pegasis: power-efficient gathering in sensor information systems," in *Proceedings of the IEEE Aerospace Conference*, vol. 3, pp. 1125–1130, 2002.

[18] A. Tyrrell, G. Auer, and C. Bettstetter, "Fireflies as role models for synchronization in ad hoc networks," in *Proceedings of the 1st Bio-Inspired Models of Network, Information and Computing Systems (BIONETICS '06)*, Madonna di Campiglio Trentino, Italy, December 2006.

[19] M. G. Gouda and T. M. McGuire, "Accelerated heartbeat protocols," in *Proceedings of the 18th International Conference on Distributed Computing Systems*, pp. 202–209, May 1998.

[20] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-based ad hoc routing," *IEEE/ACM Transactions on Networking*, vol. 14, no. 3, pp. 479–491, 2006.

[21] D. Grünbaum, S. Viscido, and J. Parrish, "Extracting interactive control algorithms from group dynamics of schooling fish," in *Cooperative Control*, V. Kumar, N. Leonard, and A. Morse, Eds., vol. 309 of *Lecture Notes in Control and Information Science*, pp. 103–117, Springer, Berlin, Germany, 2005.

[22] J. Gautrais, P. Michelena, A. Sibbald, R. Bon, and J.-L. Deneubourg, "Allelomimetic synchronization in Merino sheep," *Animal Behaviour*, vol. 74, no. 5, pp. 1443–1454, 2007.

[23] V. Ramos, C. Fernandes, A. C. Rosa, and A. Abraham, "Computational chemotaxis in ants and bacteria over dynamic environments," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 1109–1117, Singapore, September 2007.

[24] J. Buhl, D. J. T. Sumpter, I. D. Couzin et al., "From disorder to order in marching locusts," *Science*, vol. 312, no. 5778, pp. 1402–1406, 2006.

[25] S. Okdem, D. Karaboga, and C. Ozturk, "An application of wireless sensor network routing based on artificial bee colony algorithm," in *Proceedings of the IEEE Congress of Evolutionary Computation (CEC '11)*, pp. 326–330, New Orleans, La, USA, June 2011.

[26] H. Wedde, M. Farooq, and Y. Zhang, "Beehive: an efficient fault-tolerant routing algorithm inspired by honey bee behavior," in *Ant Colony Optimization and Swarm Intelligence*, M. Dorigo, M. Birattari, C. Blum, L. Gambardella, F. Mondada, and T. Sttzle, Eds., vol. 3172 of *Lecture Notes in Computer Science*, pp. 83–94, Springer, Berlin, Germany, 2004.

[27] M. Saleem and M. Farooq, "Beesensor: a bee-inspired power aware routing protocol for wireless sensor networks," in *Applications of Evolutionary Computing*, M. Giacobini, Ed., vol. 4448 of *Lecture Notes in Computer Science*, pp. 81–90, Springer, Berlin, Germany, 2007.

[28] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, pp. 90–100, New Orleans, La, USA, February 1999.

[29] P. Boonma and J. Suzuki, "MONSOON: a coevolutionary multiobjective adaptation framework for dynamic wireless sensor networks," in *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS '08)*, p. 497, Waikoloa, Hawaii, USA, January 2008.

[30] J. Senthilkumar and M. Chandrasekaran, "Improving the performance of wireless sensor network using bee's mating intelligence," *European Journal of Scientific Research*, vol. 55, no. 3, pp. 452–465, 2011.

[31] A. B. da Cunha, B. R. de Almeida, and D. C. da Silva Jr., "Remaining capacity measurement and analysis of alkaline batteries for wireless sensor nodes," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 6, pp. 1816–1822, 2009.

[32] L. Shu, C. Wu, Y. Zhang, J. Chen, L. Wang, and M. Hauswirth, "NetTopo: beyond simulator and visualizer for wireless sensor networks," in *Proceedings of the 2nd International Conference on Future Generation Communication and Networking (FGCN '08)*, pp. 17–20, Hainan, China, December 2008.

[33] E. Egea-Lopez, J. Vales-Alonso, A. Martinez-Sala, P. Pavon-Marino, and J. Garcia-Haro, "Simulation tools for wireless sensor networks," in *Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS '05)*, 2005.

[34] E. Egea-Lopez, J. Vales-Alonso, A. Martinez-Sala, P. Pavon-Marñio, and J. Garcia-Haro, "Simulation scalability issues in wireless sensor networks," *IEEE Communications Magazine*, vol. 44, no. 7, pp. 64–73, 2006.

[35] D. Kotz, J. Liu, C. Newport, Y. Yuan, R. S. Gray, and C. Elliott, "Experimental evaluation of wireless simulation assumptions," in *Proceedings of the 7th ACM Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (ACM MSWiM '04)*, pp. 78–82, New York, NY, USA, October 2004.

[36] "Iris Wireless Measurement System," http://www.memsic.com//support/documentation/wireless-sensornetworks/category/7-datasheets.htmldownload=135%3Airis.

[37] M. C. Little, *JavaSim*, University of Newcastle upon Tyne, 2009.

[38] P. Levis and D. Gay, *TinyOS Programming*, Cambridge University Press, 1st edition, 2009, http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0521896061.

[39] H. Park and M. B. Srivastava, "Energy-efficient task assignment framework for wireless sensor networks," Tech. Rep. TR-UCLA-NESL 200309-03, UC Los Angeles, 2003, http://escholarship.org/uc/item/9q5244gn.

[40] V. Rafe, H. Momeni, and M. Sharifi, "Energy-aware task allocation in wireless sensor actor networks," in *Proceedings of the International Conference on Computer and Electrical Engineering (ICCEE '09)*, pp. 145–148, Dubai, United Arab Emirates, December 2009.