Runtime Optimisation in WSNs for Load Balancing Using Pheromone Signalling

Ipek Caliskanelli [#], James Harbin [*], Leandro Soares Indrusiak [#], Paul Mitchell [*], David Chesmore [*], Fiona Polack[#]

[#] *Department of Computer Science, University of York, UK*

[*] *Department of Electronics, University of York, UK*

Abstract— **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 trade-offs. This paper proposes a bio-inspired load balancing approach, based on pheromone signalling mechanisms, to solve the trade-off 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.**

Keywords: system-level simulation; prototype; load balancing; optimisation; WSN; bio-inspired

I. 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, track and detect events according to application requirements. Processing capabilities 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 trade-off between service availability and energy consumption uses distributed, self-organising algorithms to support the WSN hardware devices. For this purpose, we propose a bio-inspired load balancing technique based on pheromone signalling mechanisms.

In this research, a task mapping optimisation is used to solve the trade-off between energy efficiency and event detection, providing effective resource management by controlling service times of the network components. Task mapping is defined as the assignment of tasks to network nodes and the definition of the task set execution sequence, aiming to achieve specific performance objectives [1]. Unlike static task mapping optimisations, the proposed technique in this paper applies runtime optimisations in order to reflect the dynamic nature of WSNs.

This paper explores two important WSN issues. The main goal of this research is to show the effects of a bio-inspired load balancing technique based on pheromone signalling. By developing our proposed technique, we target to minimisation of the energy consumption and maximisation of service availability.

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

The highly dynamic nature of WSN applications requires self-organised, autonomous behaviour to overcome the fundamental resource challenges of WSNs. We propose a solution to distribute work load over the network components in an equal manner, balancing nodes energy levels. The pheromone signalling mechanism proposed is inspired by the biological knowledge on the behaviour of bees [2-4] when assigning responsibility to members of a hive for control and distribution purposes. 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 well-being (bee/node) and collective welfare (colony/WSN).

The paper is organized as follows. Section II reviews the related work in the areas of task mapping and routing protocols in WSNs, whereas our specific problem definition is presented in Section III. Section IV covers pheromone signalling based load-balancing algorithm together with the required biological background. Section V 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 VI and the main conclusion of this study in Section VIII.

II. 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 (or offline) [5-7], 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 [8] and some allowing distributed decisions [9].

Pathak and Prasanna [10] 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 [11-12] 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 [12] use a GA with a fitness

978-1-4673-4723-5/12/$31.00 ©2012 IEEE

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 [13] also present a genetic approach, involving genome mutation and crossover. BTMS [1] 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 [14] 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 systems also has similarities with the research traditions of WSN cluster formation and dynamic topology control [15]. One key difference here is that the load balancing algorithm described in this paper is targeted at the application layer. The 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 clusterhead nodes as coordinators of a region, bearing responsibility for a system task permanently or temporarily. In LEACH [16], a form of dynamic cluster selection is presented in which nodes periodically rotate cluster head responsibilities to balance their energy consumption. Nodes probabilistically become clusterheads with probabilities governed by their remaining energy. Other nodes transmit data to the clusterhead first, and the clusterheads can be organised hierarchically to assist with delivery back to the sink. Therefore nodes with the highest remaining energy take on the burden of routing and aggregating messages from their relevant workers more often. In HEED [17] 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 [18] improves on LEACH by avoiding duplication of transmissions between cluster nodes, and introduces aggregation of data at the cluster heads.

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 SI (swarm intelligence). Cases studied include flocks of birds, shoals of fish [19], herds of sheep [20] and, bacteria colonies [21]. 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 [22]. Groups of animals such as shoals of fish increase individual and group well-being by synchronising their motion. Conforming to this swarm metaphor, the artificial bee colony algorithm (ABC) has been explored in the solution of optimisation problems [23]. In this model, bees represent search agents and their environment's 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 [24] 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 [25] 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 is capable of increasing the proportion of delivered packets compared to AODV [26], 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 [27].

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 [28]. This protocol 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 is it difficult to exactly assess available energy from the battery voltage obtained in low-cost WSN devices [29], so removing the dependence on it is an advantage. 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.

III. PROBLEM STATEMENT

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

Service availability is measured by 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 inter-communicating tasks, so a service is completed only if all its tasks are executed by the WSN nodes. We use task mapping as a way to balance the load over network nodes, i.e. to decide which node should execute the tasks of each requested service. Therefore, a service will not complete if at least one of its tasks (i) is mapped to a node that runs out of energy whilst executing it, or (ii) is not mapped to any node.

In this paper, we use a sound sensing network as a case study. Recording and processing each 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.

IV. Pheromone Signalling Based Load Balancing Algorithm

In this framework, we propose our dynamic load balancing technique aiming at minimisation of the energy consumption and maximisation of service availability. In our technique we used as a biological metaphor the bee's pheromone stimulation, which is correlated with WSN concepts as shown in Table 1.

In Table 1, queen bees refer to the sensor nodes responsible for managing task mapping and execution (the *Queen Nodes*, or QNs) and they are differentiated from other sensors to indicate their duties. This differentiation is a logical concept that does not make specific assumptions on the underlying nodes, so it can be applied to homogenous or heterogeneous platforms. Worker bees refer to the non-queen sensor nodes of the network (the *Worker Nodes*, or WNs). 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). The level of a virtual pheromone (Queen's substance), which appears as *Pheromone Level* in Table 1 determines whether a node can differentiate itself as a *Queen*. Finally, the lifetime of a bee is related to the operational lifetime of the sensor nodes (e.g. how long they operate unattended before their battery dies).

TABLE 1: CORRELATION BETWEEN BEE'S 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

The Pheromone Signalling (PS) Algorithm is the most important part of the load balancing technique. The objective of the algorithm is to enable node differentiation at a scale that produces sufficient QNs to handle all the required system functionality (e.g. service requests, event detection) QN's can either execute tasks on themselves or map tasks to available WNs. The algorithm should also 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 premise that QNs periodically propagate pheromone to their network neighbourhood. The pheromone levels propagate with hop distance to the source, for example two hop neighbours receive less pheromone than direct neighbours. 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. The pheromone level at a node also decays over time, if no other pheromone is received to counteract this decay. This ensures nodes become a QN when they are too far (in a multi-hop topology) from other QNs, or when they have/ have not received pheromone for a given time. We now formalise the PS algorithm by describing its three parts which are executed on every node of the network - two of them periodically and one on demand.

The pseudo code for the first periodic stage, referred to as the differentiation cycle, appears in Listing 1. It executes on every node of the network every T_{QN} time units. On each iteration, a node checks its current pheromone level h_i against a predefined level $threshold_{QN}$ and differentiates itself into a QN (or maintains its QN status) if that level is below the threshold. If it is a QN, it propagates pheromone to its network neighbourhood. 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.

LISTING 1: PS DIFFERENTIATION CYCLE

```
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
```

The second part of PS deals with the propagation of the pheromone released by QNs as described above. It is not a periodic activity, and happens every time a node receives a pheromone dose. Its pseudo code appears in Listing 2. Upon receiving a pheromone dose, a node checks whether the QN that produced it is sufficiently near for the pheromone to affect it. It is achieved by comparing the first element of *pd* with a predefined $threshold_{hopcount}$.

LISTING 2: PS PHEROMONE PROPAGATION

```
when hd is received
    if (hd [1] < threshold_hopcount)
        h_i = h_i + hd [2]
        broadcast hd' = { hd [1] + 1, hd [2] . K_HOP DECAY }
```

If the *pd* has travelled over a greater number of 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 allows the propagation of the pheromone to its neighbourhood. The propagated dose must represent the fact that pheromone decays with the distance to the source, so it does that by incrementing the hop count and by multiplying the dosage by a decay factor $0 < K_{HOP\ DECAY} < 1$.

Finally, the third part of the algorithm, shown in Listing 3, is a simple periodic decay of the pheromone level of each node, which happens every T_{DECAY} time units and multiplies h_i by a decay factor $0 < K_{TIME\ DECAY} < 1$.

LISTING 3: PS DECAY CYCLE

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

V. EVALUATION TECHNIQUES

Designing and experimentally evaluating the performance of the algorithms pertaining to WSNs is a fundamental focus of the research in this area [30]. Existing evaluation concepts are categorised as system-level simulators, low-level simulators, and prototypes. Since WSNs are application-specific environments, researchers choose the best-fit concept depending on their target application area. Several design factors play an important role while choosing the best-fit performance evaluation technique and they can be listed as: flexibility, scalability, complexity, implementation time, performance efficiency, financial cost and accuracy.

TABLE 2: COMPARISON BETWEEN THE PERFORMANCE EVALUATION TECHNIQUES.

Models/ Metrics	Cost	Time Consumption	Scalability	Flexibility	Accuracy	Complexity	Efficiency
System- level Simulators	**L**	**L**	**H**	**H**	**L**	**L**	**H**
Low- level Simulators	**M**	**H**	**M**	**M**	**M**	**H**	**M**
Prototypes	**H**	**M**	**L**	**L**	**H**	**M**	**L**

In Table 2, three performance evaluation techniques are compared against each other. Design factors are marked with either L, M or H for each performance evaluation techniques, in which L refers to low, M refers to medium and H refers to high. As is shown in Table 2, system-level simulation models are cost-efficient and marked as low cost. Financial costs of prototypes are high, whereas low-level simulation models are listed as 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 not flexible and not scalable, so listed as low. Implementation duration of low-level simulation is greater, compared to the prototypes and system-level simulation models due to their level of complexity. In terms of accuracy, prototypes provide the most accurate results since they provide the results of real sensor deployments. Low-level simulation models are more accurate than system-level simulation models, because of the broader assumptions and abstraction away of detail in the system-level simulation models. In terms of evaluating protocol performance efficiency, prototypes are known to be inefficient although they provide the most accurate results, since they feature real WSN deployment hardware and operating system environments [31-33].

In this research, we have decided to validate our approach on system-level simulation model and a small hardware prototype test-bed. 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 the effect of the proposed load balancing technique.

A. 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. The reason behind the required selection is to explore the large parameter space of the load balancing technique without considering the hardware obstacles and time consumption 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. The platform model consists of a set of N nodes and a set L of links between the nodes. Each node $n_i \in N$, is the tuple $n_i = \{mc_i, e_i, h_i\}$, where mc_i is the available memory capacity of the node, e_i is its energy capacity and h_i is the pheromone level of the node. Each link $l_k \in L$ denotes the possibility of direct communication between two nodes $n_i, n_j \in N$. A service is designed as a Directed Acyclic Graph (DAG) and can be represented as $S = (T, C)$ which consists of a set of tasks T and set of inter-task communications 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 inter-task 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 co-domain and is represented as $F: T \rightarrow N$.

An event-driven simulator has been designed to implement this model. It is controlled by the JavaSim library [34] and is validated with 30 different task sets.

The level of accuracy is the only open discussion which is the major disadvantage of the system-level simulators. Duty cycling MAC protocols are out of the scope of this work, whereas other parameterization and discharge rates are taken into consideration to reflect the real world applications and achieve a high level of accuracy.

B. TinyOS Experimental Testbed

The experimental testbed Fig.1 is intended to evaluate the short-term behaviour of the protocol. It consists of 16 homogeneous nodes (MEMSIC Iris nodes with 2.4GHz transceivers) together with a base station that serves to receive results and transfer them via USB to a monitoring computer.

Nodes have MPR-400 sensor boards attached, which detect sound events. These nodes run on the open-source TinyOS operating system version 2.1 [35]. A custom modular application was developed to perform multi-hop forwarding, sound detection, and to implement the pheromone protocol. Message delivery was performed using the TinyOS Active Message layer. The application 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.

Fig. 1: The hardware testbed in operation.

C. Scenario, Topology and Routing Protocol

Hardware nodes are arranged into a 4x4 regular grid and perform multi-hop routing in order to reach the sink node. Routing is pre-configured with nodes forwarding hop-by-hop on fixed multi-hop relaying chains towards the sink node with ID 1, as depicted in Fig.2. The intent of this shortest path routing in pre-configured 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.

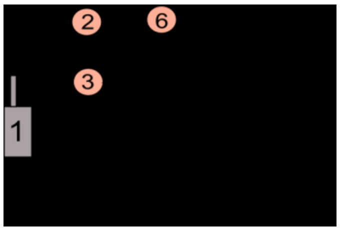


Fig. 2: The topology of multi-hop routing chains, shortest path.

During the experimental deployment, nodes are located sufficiently close for packet transmission/reception to occur across the entire network. The simulation scenario assumes nodes can only communicate directly with their immediate neighbours, to emulate the conditions of a real large-scale deployment. However, since Iris nodes typically achieve transmission ranges up to 50m 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 IV B, assigning statuses of QN and WN to nodes dynamically (they change as execution proceeds). Queen nodes respond to the periodic detections by transmitting an event notification, while worker nodes ignore these events at the application layer. Further down the protocol stack, multi-hop 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.

TABLE 3: ENERGY RELATED PARAMETERS

Configuration Parameters	Platform Model
Battery Capacity (mAh)	1500
Idle Discharge Rate (uAs)	300
Wireless Communication Discharge Rate per Byte at 30kbps (uAs)	0.6
Task Communication Discharge Rate (uAs)	3000

D. Parameter Choices

Several important parameters of the balancing technique are shown in Table 4. In the experimental results we have also inspected the importance of pheromone propagation period and pheromone decay period.

TABLE 4: PS RELATED PARAMETERS

Parameters/ Platforms	Real Deployment	Simulation
$K_{HOP\ DECAY}$	~ 0.25	~ 0.25
$\mathbf{threshold}_{QN}$	0.14,0.28,0.56	14
$K_{TIME\ DECAY}$	0.5	0.5
T_{DECAY} (s)	7.2	7200,5000,3000 ,2000
$threshold_{hopcount}$	2	2
h_{QN}	50	50

VI. EXPERIMENTAL RESULTS

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

A. TinyOS Hardware Testbed Experiment Results

Fig. 3 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 interval. Therefore the smaller the numbers of event detect the better, since this represents minimal duplication. The results demonstrate that following stabilisation (after 40s) the load balancing algorithm produces a significantly smaller number of detections, reducing the 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 generates in other network nodes. There is an initial decay in beginning event detection 20-40 at a second interval as the network stabilises and a suitable number of nodes become QNs.

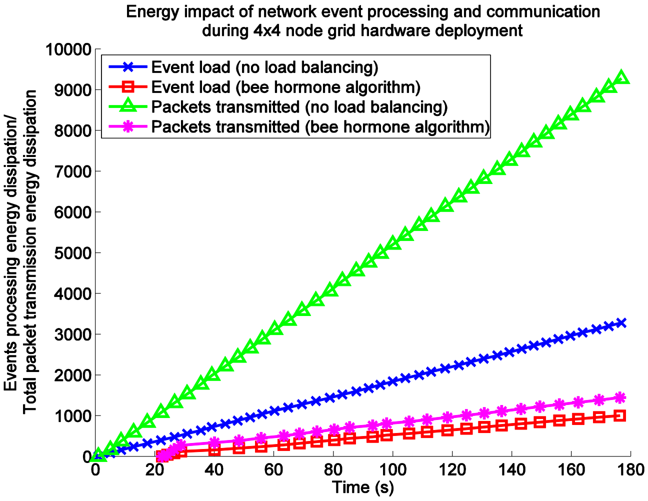


Fig. 3: Energy impact of the event processing on 4x4 mesh network topology.

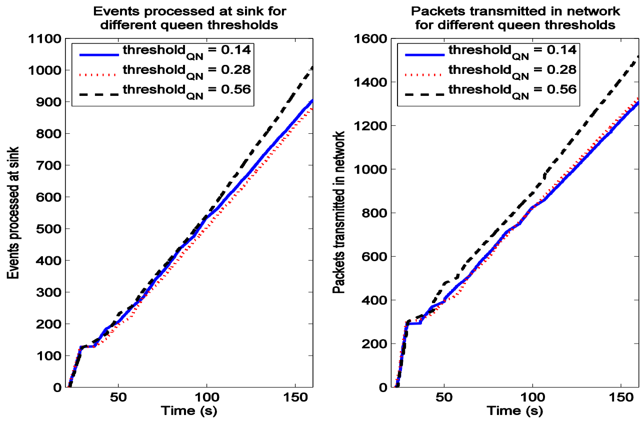


Fig. 4: The impact of queen threshold upon event detection and packet transmission load on a 4x4 mesh network topology.

Fig. 4 shows the impact of queen hormone 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 differentiation algorithm in Listing 1 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 16% due to processing these events, and the additional hormone 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.

B. System-level Simulation Model

Fig. 5 shows the percentage of the detected events (a), and the percentage of alive nodes (b).Idle, Baseline, BS and PS are the scenarios shown in Fig.5. Idle represents maximum lifetime of the nodes when the nodes are idle. Baseline is a scenario referred to as a dynamic mapping technique, without dynamic re-mapping. BS represents an existing self-organisation mechanism, which applies both dynamic mapping and re-mapping techniques [37-38]. In BS, nodes that have an energy level less than the defined threshold re-map their tasks to a neighbour which hold the maximum energy level. PS is our load balancing technique presented in Section IV.

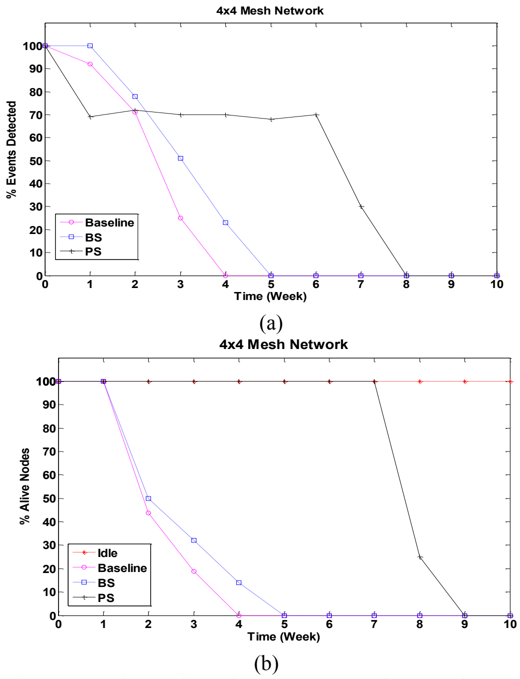
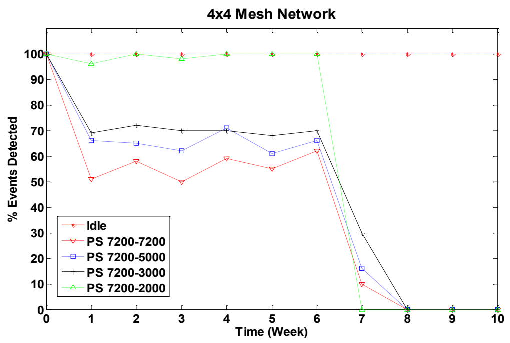


Fig. 5: Experimental results: On 4x4 mesh network topology (a) % Events detected, (b) % Alive nodes.

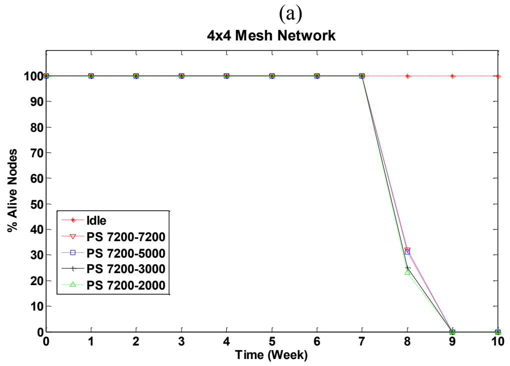
According to Fig. 5 (a) and (b), the percentages of detected events and alive nodes are lowest in baseline scenario. Due to the effect of the dynamic re-mapping technique, the percentage of alive nodes is increased as well as the percentage of detected events. The major difference is shown by the PS scenario. The detected event percentage remains constant after the first week, in which the number of QNs stabilises. 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 8. At the end of the week 8, 70% of the nodes run out of energy at the same time. This proves the balancing the load 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 V. 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 4x4 network in Fig. 6.

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 on 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.

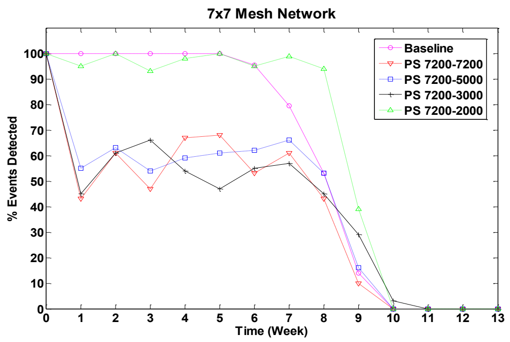


(a)

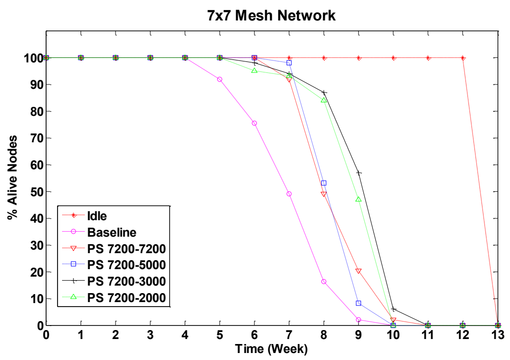


(b)

Fig. 6: Experimental results: effects of different phormone decay period on 4x4 Mesh Network topology using : (a) % Events detected, (b) % Alive nodes.



(a)



(b)

Fig. 7: % Events detected (a), % alive nodes (b) for 7x7 Mesh Network topology.

Fig. 7 shows the effects of our load balancing technique. Impacts of the pheromone decay interval are also shown on Fig. 7 where simulation duration is increased to 13 week. Larger scale and longer simulation duration allows to analyse the overall impact of the technique and effects of the T_{DECAY} time unit.

In every case, the PS algorithm provides an optimised solution in terms of service availability and energy consumption, regardless from the decay interval. Shorter decay interval means higher number of QNs, where PS 7200-2000 provides the highest percentage of detected events. Due to the high number of QNs, PS 7200-2000 do not improve the percentage of the alive nodes as much as percentage of detected events due to the redundant allocations.

VII. Conclusions

In this paper, we have proposed a load balancing algorithm based on a pheromone signalling mechanism. We showed the long term 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 real sensor deployment. The advantages and disadvantages of these two performance evaluation methodologies have been highlighted.

We have two major goals: solving the service availability versus energy consumption trade-off with the proposed algorithm, and demonstrating good performance via the two evaluation methodologies of system-level simulation and hardware deployment.

Simulation results show that our technique provides longer network lifetime, while increasing the service availability over longer time scales consistent with a real deployment. Hardware results for a 4x4 grid with multi-hop 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. Cost-wise, 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.

Acknowledgements

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

References

[1] Y. E. M. Hamouda and C. Phillips, “Biological Task Mapping and Scheduling in Wireless Sensor Networks,” in *Proc IEEE Int Conf Communications Technology and Applications (ICCTA),* 2009, pp. 914–919.

[2] M. B. V. Roberts, Biology: a functional approach, 1986.

[3] W. H. Johnson, "Essentials of biology," 1974.

[4] J. B. Free, Pheromones of social bees: Comstock Pub, Associates, 1987.

[5] S. Shivle, *et al.*, "Static allocation of resources to communicating subtasks in a heterogeneous ad hoc grid environment," *J. Parallel Distrib. Comput.,* vol. 66, pp. 600-611, 2006.

[6] T. D. Braun, *et al.*, "Static mapping heuristics for tasks with dependencies, priorities, deadlines, and multiple versions in heterogeneous environments," in *Proc Int Parallel and Distrib. Processing Symposium. (IPDPS 2002),* 2002, pp. 78-85.

[7] G. Martinovic, *et al.*, "Static-dynamic mapping in heterogeneous computing environment," in *IEEE Int Symp Virtual Environments, Human-Computer Interfaces and Measurement Systems (VECIMS),* 2003, pp. 32-37.

[8] H. S. AbdelSalam and S. Olariu, "Toward Efficient Task Management in Wireless Sensor Networks," *IEEE Trans on Computers,* vol. 60, pp. 1638-1651, 2011.

[9] M. Uney and M. Cetin, "Graphical Model-based Approaches to Target Tracking in Sensor Networks: An Overview of Some Recent Work and Challenges," *in 5th Int Symp Image and Signal Processing and Analysis (ISPA),* 2007.

[10] A. Pathak and V. Prasanna, "Energy-Efficient Task Mapping for Data-Driven Sensor Network Macroprogramming," in *Distrib Computing in Sensor Systems*. vol. 5067, S. Nikoletseas, *et al.*, Eds., ed: Springer Berlin / Heidelberg, 2008, pp. 516-524.

[11] Z. Zhiwen, *et al.*, "A Highly Efficient DAG Task Scheduling Algorithm for Wireless Sensor Networks," in *The 9th Int Conf Young Computer Scientists (ICYCS) 2008*, pp. 570-575.

[12] Y. Jin, *et al.*, "Latency and Energy-Consumption Optimized Task Allocation in Wireless Sensor Networks," in *IEEE Wireless Comm. and Networking Conf (WCNC), 2010*, pp. 1-6.

[13] D. Miorandi, *et al.*, "Service evolution in bio-inspired communication systems," *Int Trans on Systems Sciences and Applciations,* vol. 2, pp. 573–587, 2006.

[14] B. Atakan and Ö. Akan, "Immune system based distributed node and rate selection in wireless sensor networks," in *Proc of the 1st int conf on Bio inspired models of network, information and computing systems*, 2006, pp. 3-3.

[15] P. Santi, *Topology control in wireless ad hoc and sensor networks*: Wiley, 2005.

[16] W. R. Heinzelman, *et al.*, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc of the 33rd Annual Hawaii Int Conf System Sciences*, 2000, p. 10 pp. vol.2.

[17] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Trans Mobile Computing,* vol. 3, pp. 366-379, 2004.

[18] S. Lindsey and C. S. Raghavendra, "PEGASIS: Power-efficient gathering in sensor information systems," in *IEEE Aerospace Conference Proceedings,* 2002, vol.3, pp. 3-1125-3-1130.

[19] D. Grünbaum, *et al.*, "Extracting Interactive Control Algorithms from Group Dynamics of Schooling Fish," in *Cooperative Control*. vol. 309, V. Kumar, *et al.*, Eds., ed: Springer Berlin Heidelberg, 2005, pp. 103-117.

[20] J. Gautrais, *et al.*, "Allelomimetic synchronization in Merino sheep," *Animal Behaviour,* vol. 74, pp. 1443-1454, 2007.

[21] V. Ramos, *et al.*, "Computational chemotaxis in ants and bacteria over dynamic environments," in *IEEE Congress Evolutionary Computation (CEC),* 2007, pp. 1109-1117.

[22] J. Buhl, *et al.*, "From disorder to order in marching locusts," *Science,* pp. 1402–1406, 2006.

[23] S. Okdem, *et al.*, "An application of Wireless Sensor Network routing based on Artificial Bee Colony Algorithm," in *IEEE Congress Evolutionary Computation (CEC),* 2011, pp. 326-330.

[24] H. Wedde, *et al.*, "BeeHive: An Efficient Fault-Tolerant Routing Algorithm Inspired by Honey Bee Behavior," *Ant Colony, Optimization and Swarm Intelligence*, 2004, pp. 83-94 2004.

[25] Saleem, M., & Farooq, M., "BeeSensor: A bee-inspired power aware routing protocol for wireless sensor networks". In *Lecture Notes in Computer Science,* Vol. 4449. Applications of evolutionary computing , pp. 81–90.

[26] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proc of the 2nd IEEE Workshop Mobile Comp Systems and Applications (WMCSA)*, New Orleans, LA, USA, 1999, pp. 90-100.

[27] P. Boonma and J. Suzuki, "MONSOON: A Coevolutionary Multiobjective Adaptation Framework for Dynamic Wireless Sensor Networks," in *Proc of the 41st Annual Hawaii Int Conf on System Sciences,* 2008, pp. 497-497.

[28] J. Senthilkumar and M. Chandrasekaran, "Improving the Performance of Wireless Sensor Network using Bee's Mating Intelligence," *European Journal of Scientific Research,* vol. 55, 2011.

[29] A. B. da Cunha, *et al.*, "Remaining Capacity Measurement and Analysis of Alkaline Batteries for Wireless Sensor Nodes," *IEEE Trans Instrumentation and Measurement,* vol. 58, pp. 1816-1822, 2009.

[30] S. Lei, *et al.*, "NetTopo: Beyond Simulator and Visualizer for Wireless Sensor Networks," in 2nd *Int Conf Future Generation Communication and Networking (FGCN)*, 2008, pp. 17-20.

[31] C. Newport, *et al.*, "Experimental Evaluation of Wireless Simulation Assumptions," *Simulation,* vol. 83, pp. 643-661, 2007.

[32] E. Egea-Lopez, *et al.*, "Simulation scalability issues in wireless sensor networks," *IEEE Communications Magazine,* vol. 44, pp. 64-73, 2006.

[33] E. Egea-Lopez, *et al.*, "Simulation tools for wireless sensor networks," *Proc Int Symp on Performance Evaluation of Computer and Telecommunication Systems (SPECTS),* 2005.

[34] M.C. Little and D.L. McCue, C User's Journal Vol. 12, Issue 3, CMP Media/Miller Freeman Inc., 1994 ISSN 1075-2838.

[35] P. Levis and D. Gay, *TinyOS Programming*: Cambridge University Press, 2009.

[36] Memsic Corp. IRIS Wireless Measurement System Datasheet. 2010.

[37] H. Park and M. B. Srivastava, "Energy-Efficient Task Assignment Framework for Wireless Sensor Networks," UC *Los Angeles, TR-UCLA-NESL-200309-03*, 2003.

[38] V. Rafe, H. Momeni, and M. Sharifi, "Energy-Aware Task Allocation in Wireless Sensor Actor Networks," in *2nd Int Conf on Computer and Electrical Engineering (ICCEE),* 2009, vol. 1, pp. 145–148.