A Bio-inspired Load Balancing Technique for Wireless Sensor Networks

İpek Çalışkanelli

Doctor of Philosophy

University of York

Computer Science

March 2014

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 thesis is concerned with the load balancing of Wireless Sensor Networks (WSNs). We present an approach, inspired by bees' pheromone propagation mechanism, that allows individual nodes to decide on the execution process locally to solve the trade-off between service availability and energy consumption. We explore the performance consequences of the pheromone-based load balancing approach using a system-level simulator. The effectiveness of the algorithm is evaluated on case studies based on sound sensors with different scenarios of existing approaches on variety of different network topologies. The performance of our approach is dependant on the values chosen for its parameters. As such, we utilise the Simulated Annealing to discover optimal parameter configurations for pheromone-based load balancing technique for any given network schema. Once the parameter values are optimised for the given network topology automatically, we inspect improving the pheromone-based load balancing approach using robotic agents. As cyber-physical systems benefit from the heterogeneity of the hardware components, we introduce the use of pheromone signalling-based robotic guidance that integrates the robotic agents to the existing load balancing approach by guiding the robots into the uncovered area of the sensor field. As such, we maximise the service availability using the robotic agents as well as the sensor nodes.

Contents

List of Figures

List of Tables

Acknowledgements

Above all, most importantly I would like to thank my supervisor, Dr Leandro Soares Indrusiak for his great effort and time; he has always been a great technical guide with his expertise. He turned a new graduate into a researcher and without him I would not have succeeded. Many thanks to my examiners Dr Utz Roedig and Dr Iain Bate for taking time to read my thesis. Their feedback helped me to improve this thesis.

Thanks a lot for my family for their boundless love, care and support. Although we have lived apart for years now, they were there when I needed them and I am very lucky to have them in my life. I also want to thank to my lovely partner, James, for being a constant pillar of support through hard times.

I would like to thank all the RTS group members for the technical discussion and for raising many interesting points in my research that I had not thought off. I particularly want to thank Dr Iain Bate, who has shown great interest in my thesis, made many useful suggestions during my PhD, and cared about my well-being. Special thanks to Dr James Harbin, he has been a good friend and a colleague who supported me technically and motivated emotionally when I needed.

During my studies in York I met with Onder Fahrioglu, Feriha Bilgekul, Oytun Sozudogru and Sevi Horoz whose became the most valuable friends to me. As well as my degree and the expertise that I gained in York, the friends that I made during this time are important and I am glad to have them in my life. Special thanks to Onder Fahrioglu and James Williams who also helped me with proofreading this thesis.

Declaration

This thesis has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree other than Doctor of Philosophy of the University of York. This thesis is the result of my own investigations, except where otherwise stated. Some parts of this thesis have been published in conference proceedings and journals; where items were published jointly with collaborators, the author of this thesis is responsible for the material presented here.

- Ipek Caliskanelli, James Harbin, Leandro Soares Indrusiak, Paul Mitchell, David Chesmore, Fiona Polack. Runtime Optimisation in WSNs for Load Balancing Using Pheromone Signalling. In *Proceedings of the 3rd IEEE International Conference on Network Embedded Systems for Every Application (NESEA 2012), 2012* [21].

- Ipek Caliskanelli, James Harbin, Leandro Soares Indrusiak, Paul Mitchell, David Chesmore, Fiona Polack. Bio-inspired load balancing in large-scale WSNs using pheromone signalling. In *International Journal of Distributed Sensor Networks, vol. 2013, Article ID 172012, 14 pages, 2013 doi:10.1155/ 2013/ 172012* [24].

- Ipek Caliskanelli, Leandro Soares Indrusiak. Search-Based Parameter Tuning on Application-Level Load Balancing For Distributed Embedded Systems. In *Proceedings of the 11th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC 2013), 2013* [22].

- Ipek Caliskanelli, Leandro Soares Indrusiak. Using Mobile Robotic Agents to Increase Service Availability and Extend Network Lifetime on Wireless Sensor

and Robot Networks. *Proceedings of the 12th IEEE International Conference on Industrial Informatics (INDIN), 2014* [23]

Chapter 1

Introduction

1.1 Background And Motivation

Wireless Sensor Networks (WSNs), also known as *sensornets*, consist of a large number of small distributed, autonomous, 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.

As the technology advances, the resource capacity of WSNs improves, whilst the cost of the sensors decreases. However, these small, capacity-limited devices still restrict the performance of the applications in which they are used. One approach to maintain the required performance, without introducing additional resources from outside the network, is to place an excessive number of node deployments in the network. This, however, causes greater computational redundancy, and thus, increases the network overheads.

In short, resource limitations (e.g. processing and energy) and scalability issues of WSNs are the two unique characteristics that differentiate them from traditional wireless ad hoc networks. Therefore, managing large scale WSNs is even more difficult as they require solutions to overcome not only the limited resources of these environments, but also the computational redundancy that results. In particular, large scale WSNs need to carefully allocate tasks over sensors in such a way that communicating tasks are physically close enough to each other to avoid large net-

work latencies, but at the same time are well spread across the network to avoid network congestion and communicational energy consumption. Furthermore, classic resource scheduling and load balancing techniques cannot handle this type of problem due to the size and the distributed nature of the given large scale problem. This, together with time constraints to perform load balancing are some of the key factors on performance. Optimising the performance of WSNs is a challenging problem, in particular the issue of load balancing and resource allocation in WSNs is NP-complete. In a networked system of N processing nodes and M communicating tasks, there are M^N possible task allocations for the system. It is implausible for a central node to be aware of the complete state of the system to make the best allocation decisions: communication latencies mean that the state may be stale by the time global allocation decisions are made and disseminated. The complexity of the problem in WSN domain together with the distributed, dynamic nature of the resource limited WSNs result in tight timing constraints which prevent the use of computationally heavy algorithms. As a result, there is a great need to find a computationally lightweight load balancing algorithm that can cope with large scale WSNs without introducing high level of redundancy. Moreover, among the resource challenges, managing the energy usage and prolonging the network lifetime without sacrificing much from the network performance in terms of service availability and quality of service (QoS) has a major criticality [1].

To contextualise the given characteristics of the WSNs, we describe an exemplar scenario of the use of WSNs: Wild-Life Sound-Based Detection Systems that are used for bird identification, or intruder detection have a dense wireless network of sound sensors. Each sound is sensed by many of the network nodes simultaneously. In such systems, some techniques will allow many network nodes to process the same sound at the same time resulting in repercussions such as computational redundancy, shorter lifespan of nodes and low performance efficiency especially once nodes started to run out of energy. In such techniques, where there is no attempt to distribute the load over the network components (load balancing), the sensor nodes in the more

[1] Military-purpose applications and safety-critical systems are exempt.

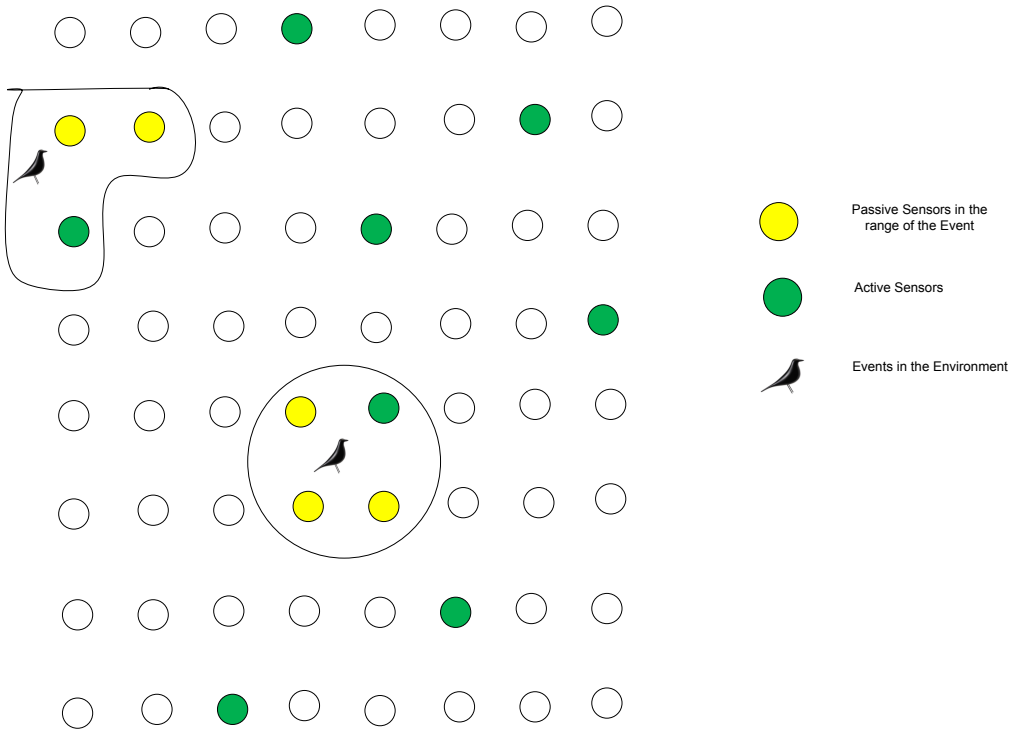


Figure 1.1: Wild-Life Sound-Based Detection on Sensor Field

active areas of the sensor network (e.g. where birds appear more) will run out of energy almost simultaneously as a result of computational redundancy. This will lead to a shorter network lifespan and low performance efficiency.

To cope with such scenarios, an effective runtime mechanism should

- use a distributed control mechanism
- be in place to optimise the allocation of load in a way that it avoids unnecessary computational redundancy by guaranteeing that relevant events will not go undetected
- spread the computation and communication load across the network (and therefore distribute the energy consumption about the network)
- increase the network lifespan.

1.2 Challenges and Key Design Issues

The emergence and subsequent rapid growth of WSNs has led to many open issues for researchers. Beside the limited resource structure and the scalability issues of WSNs that are mentioned in the previous section, WSNs have other challenges related to the location in which they are deployed. WSNs are typically composed of a small number of battery-powered sensor nodes are often deployed in unfriendly and unattended remote environments. Network maintenance (i.e. changing, replacing or reconfiguring the failed nodes) can be costly or even impractical (e.g. in cases where the network is isolated) and as a result network lifetime and performance is highly restricted and dependent on the survivability of the nodes. From the performance and financial standpoint, it is more beneficial to produce a network structure which can cope with the changes in the environment without maintenance.

The need to adapt to changing environments has always been associated with self-adaptation, self-organisation, and self-configuration. The dynamic and unpredictable nature of adaptive systems like WSNs, require self-adaptation and self-organisation protocols. A way to enable self-adapting and self-organising behaviours is to take inspiration from nature which is the biggest and the greatest dynamic and adaptive system. Organic computing, or bio-inspired computing as it is often called, takes inspiration from living objects of nature and mimics their behaviours to achieve more robust systems in the fields of engineering and computer science. In both 2003 and 2013 the MIT Technology Review [149] identified biology and computational modelling/simulation of chemical and biological systems, and WSNs as two of the ten emerging technologies that will change the world. Therefore, this thesis focuses on finding a solution to the fundamental resource challenges of WSNs, using bio-inspired techniques on large scale WSNs.

1.3 Research Question

The aim of this thesis is to investigate the problem of load balancing in WSNs with a focus on adaptivity to the dynamic environment. Instead of addressing this aim from the perspective of a small scale problem, this research will approach it from

the perspective of large scale problem in order to inspect scalability issues of WSNs as well as load balancing. Therefore, the research question that this thesis aims to address is as follows:

Can a distributed bio-inspired load balancing technique be used to improve service availability and network lifetime using adaptive and dynamic resource management for large scale WSNs with redundant coverage?

1.4 Research Objectives

This thesis will investigate the use of biologically-inspired techniques for load balancing large-scale WSNs. The proposed bio-inspired load balancing solution in this thesis distributes workload evenly over the network components, and balances node energy levels. Classical load balancing techniques (in particular centralised algorithms that focus 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 bio-inspired mechanisms. In this research, we present a task mapping optimisation algorithm that addresses the trade-off between energy efficiency and event detection at run-time, maximising service availability while reducing energy consumption by restricting service times of the network components.

In detail, the research objectives of the experimental work in this thesis are:

- RO1: To implement a lightweight bio-inspired load balancing technique that can satisfy the dynamic and adaptive requirements of large scale WSNs.
- RO2: To establish an experimental test bed to investigate the characteristics of load distribution over network resources and the aspect of dynamic and adaptive bio-inspired load balancing in large scale WSNs.
- RO3: To investigate the feasibility of accelerating the experimental test bed without sacrificing accuracy to improve the performance of the experimental test bed.

- RO4: To inspect the effects of a static parameter tuning on our bio-inspired load balancing algorithm from a large-scale perspective, and to propose a corresponding parameter tuning method for the parameter optimisation.

- RO5: To extend the load balancing technique that is provided in *RO1* to incorporate mobile robotic agents that help distribute load more effectively about the network.

1.5 Thesis Structure

The structure of this thesis is as follows. Chapter 2 inspects the related work and analyses the current problems encountered in WSNs development. Chapter 3 presents our bio-inspired load balancing technique, based on the pheromone signalling mechanism of bees, as a potential solution to these problems. Chapter 3 also establishes an experimental test bed to investigate the proposed algorithm. Following that Chapter 4 inspects the experimental test bed and introduces a new experimental test bed that aims to improve the performance of the simulations. This chapter also presents a search-based parameter tuning approach for our developed pheromone signalling-based load balancing algorithm to automatically discover optimal parameter configurations for any given network topology. Chapter 5 argues the benefits of using additional mobile robotic agent to increase the network performance that corporates with the fixed sensor nodes. Chapter 6 concludes with a summary of findings developed in this research.

Chapter 2

Literature Survey

In the previous chapter we described the wide application areas, characteristics and challenges of WSNs. Many of the challenges of WSNs are tackled by optimisation techniques. The chapter gives an overview of relevant literature that has attempted to describe, analyse, or efficiently exploit optimisation techniques on WSNs. This chapter is split into four main parts of the problem targeted in this research:

- Section 2.1 defines **Load Balancing** and describes why load balancing is crucial for restricted embedded systems like WSNs explicitly. We classify existing load balancing techniques based on their implementation area on network (OSI) stack on two levels. This section also covers task mapping and bio-inspired task mapping techniques, which are widely used to perform load balancing on WSN. Task mapping techniques are explained in detail with well-known algorithms and categorised in section 2.1.1. In section 2.1.2, bio-inspired techniques that are in the field of computer science are introduced, defined, and categorised based on their inspirations and area of application. As most of the proposed bio-inspired techniques improve load balancing directly or indirectly, this section covers all the explored with a taxonomy briefly. Section particularly focuses on the bio-inspired load balancing techniques.

- Section 2.2 describes **Performance Evaluation** and starts with the importance of the performance evaluation on WSNs. Section continues describing the performance evaluation metrics and different evaluation techniques in sec-

tion 2.2.2. Definitions of the evaluation metrics are given, categorised and their differences are explained in section 2.2.1. Section 2.2.2 explains the importance of choosing a well-suited performance evaluation technique to incorporate with the metrics.

- Section 2.3 categorise **Optimisation Techniques** and introduces examples of the relevant literature. In 2.1.1 we described task mapping optimisation. In this section, we present traditional optimisation techniques in Section 2.3.1, and metaheuristic search techniques in Section 2.3.2 briefly. Section 2.3.2.1 particularly covers the research that implies Simulated Annealing metaheuristic search as it is widely used to optimise WSNs.

- Section 2.4 explains **Network Coverage**. Section starts with gives brief information on the network coverage and connectivity, expalins the different method on how to improve connectivity and provides relevant approaches exist in the literature. Section 2.4.1 introduces variety of techniques that uses mobile entities to increase the network coverage in WSNs. Some of the significant research examples on increasing the network coverage using mobile entities given.

2.1 Load Balancing

The concept of load balancing in WSNs refers to distributing work load over the network component and is analogous to Bin Packing problem [45]. The Bin Packing problem is one of the well-known object distribution algorithms, which is often applied to tackle optimisation problems specially in resource allocation, and scheduling issues [190]. Given N items, of sizes S1, S2, S3, S4, ..., SN. The goal is to pack items into as few bins as possible. The Bin Packing can be applied in two ways; offline and online [77], [98]. Online techniques occur at run time, they are difficult to implement. On the other hand online techniques are more promising and they provide (near) optimal results. In WSNs, classic load balancing techniques are often applied offline before runtime (static) and/or use a centralised control mechanism to

manage the network load distribution. Both static techniques and approaches that implement centralised control mechanisms do not cope with the dynamic nature of WSNs and so much of the recently proposed load balancing techniques WSNs are applied online (dynamic at run time). In this section, we overview some of the essential research in WSN based on the applied area in different partitioning layers of the communication model (OSI stack) [207].

The concept of load balancing has been applied at both the network level, and the application-level of the OSI stack, and it has significant impact on low power consumption and network performance. Both network and application-level load balancing target efficient utilisation of resources to extend the network lifetime. At the network-level, work-load refers to packet transfer and communication, whereas at the application-level it refers to execution and processing the data (e.g. monitoring the environment, sensing the temperature).

Network-Level Load Balancing Research that has focused on network-level load balancing inspects successful packet delivery ratio, error rate, latency, link failures and bandwidth usage in context of routing particularly and it has traditionally focused on clustering schemes, in which the protocol selects clusterhead nodes as regional coordinators to bear responsibility for a system task. In LEACH [67], a form of dynamic cluster selection is presented in which nodes periodically rotate clusterhead 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 clusterheads 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 [202] 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 [107] improves on LEACH by avoiding duplication of transmissions between cluster nodes, and introduces aggregation of data at the clusterheads. Firefly protocols propagate control messages to synchronise and coordinate network functions across a region, for exam-

ple clock synchronisation [179]. Heartbeat protocols are used to verify liveness and reachability of remote nodes (via a request-acknowledgement cycle) when dealing with distributed processes or nodes that may fail [60]. Gossip protocols are used for probabilistic data distribution at the network layer [63]. In [36] a reliability-based distributed routing algorithm is implemented to achieve low-power consumption. This dynamic approach keeps the network responsive to link dynamics by sending broadcast beacon packets periodically and updating route information. Trumler et al. [175] present *AMUN* – self-organising middleware that distributes the work load of service-oriented applications. Their proposed technique is inspired from the human hormone system and provides near optimal resource usage for large-scale systems. The distributed algorithm uses organisational information about hormones information to piggy-back on top of the messages that are exchanged between nodes.

Application-Level Load Balancing Research that has focused on application-level load balancing distributes the work load by deciding which node should execute the requested application requirement among network entities. A static technique presented by Zeng et al. [205] aims to improve network response time and limit energy usage. However, this approach results in overloading the network, as the mapping cannot adapt effectively to network conditions. Miorandi et al. [123] present a genetic approach, involving genome mutation and crossover. DNRS [7] limits energy consumption while improving reliable event detection. BTMS [65] uses zygote differentiation to extend the network lifetime whilst speeding up task mapping and scheduling. Homogeneous nodes begin in a default state and within time nodes differentiate themselves dynamically to perform distinct tasks according to their location.

Summary

In this section, we described the purpose of load balancing for WSNs, and categorised the application area on OSI stack. A wealth of approaches on both the network and application-level of OSI have been given as examples of load balancing in WSNs. Each of the approaches have their own drawbacks although they improve the network

performance. In the next section, we focus on task mapping techniques which are used as a way of optimising WSN performance.

2.1.1 Task Mapping Techniques

Task mapping is an optimisation technique that has been widely used in WSNs. *Tasks* are often considered as the smallest parts of the system requirements and functionalities that the application model consists of [199]. System functionality can be defined as describing the hardware components on an abstract platform models. An example of a system functionality in WSNs can be monitoring, collecting and processing the real- time data based on the sensor readings.

Task mapping is defined by Hamouda et al. [65] as assigning tasks to network resources and determining the task execution sequence to achieve the performance objectives. In this research, the concept of task mapping refers to distributing responsibility for performing work across the entities of a distributed system such as a sensor network. In WSNs, mapping techniques have been used in multi-model approaches, where different models used in mapping process (unrelated from the OSI layers) represent different aspects of a WSN system: platform model, application model and mapping model. The platform model consists of the hardware structure of the networks, mainly called as sensors. Application model consists of system requirements and functionalities. System functionality can be defined as describing the hardware components using abstract platform models, such as monitoring, collecting and processing the real- time data. In this research tasks are the system requirements and functionality of the application model is computation of the tasks and inter-task communications. Mapping model, or mapper, which is the link between the application model, and the platform model, has been defined as the system component which plays a critical role of handling mapping-related functionalities that enable explorations of mapping influence in terms of performance metrics [14]. Only the mapping heuristics plays a critical role of maintaining the low energy consumption, longer network lifetime and the resource availability where platform layer and application layer remains the same.

In order to explain the task mapping process clearly, we give an example using

Set Theory and provide a mathematical model of the concept of task mapping. The platform model, $PM = (N, L)$ consists of a set of nodes N and a set of bidirectional wireless links between neighbouring nodes L. Each node $n_m \epsilon$ 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 tasks 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 set of links $l_{mn} \epsilon$ L, each communication c_{ij} contains a sender and a receiver, which is also represented as $l_{mn} = s_m, r_n$. An application model $AM = (T, C)$ consists of a set of tasks T and set of inter-task communications C. Each task $t_i \epsilon$ T, is the tuple $t_i =< id_i, mf_i, e_i, et_i >$, where id_i is the name of the task, mf_i is its the memory footprint in bytes. e_i is the energy consumption of the task, and et_i is its execution time. Each inter-task communication $c_ij \epsilon$ C, is also a tuple $c_j =< s_j, r_j >$, where $s_j \epsilon$ T is the sender task and $r_j \epsilon$ T is the receiver task of the communication. The mapping process can be define as a surjective function $f(t)$ where t is the tasks of the mapping function $f(t)$. $f(t) = AP$, where all the tasks in AM are being mapped onto nodes of PM, but not all the nodes in the co-domain need to be allocated tasks. The mapping function can also be explained separately as computation scheduling and communication scheduling as;

$mapTask(t) = mapComputation(co) + mapCommunication(c)$

where $mapComputation(co) = TSandmapCommunication(c) = CL$.

Ost el al. [133] categorise the task mapping techniques for Network on Chips (NoC's) as Figure 2.1 illustrates. Due to the similarities of NoC's and WSN's in terms of their resource constrained, and energy-hungry nature, their categorisation is valid for WSNs and also relevant to this research.

According to Figure 2.1 mapping techniques are categorised into four as moment of task mapping, number of the tasks per sensor node, mapping control system and the architecture model. *Moment when task is mapped* is divided into three as offline,

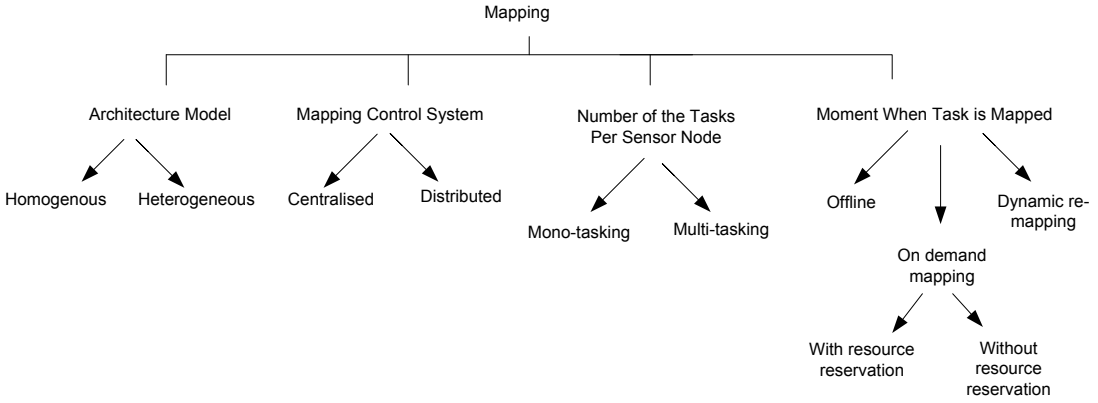


Figure 2.1: Categorisation of mapping techniques [133].

on demand and dynamic re-mapping. Offline mapping techniques as often referred as static, occur in advance, whereas on demand mapping approaches refer to the dynamic techniques which take place during execution time of the application. Since on demand dynamic mapping approaches use fast and simple algorithms, execution time reduces. On demand dynamic techniques can be separated into two; with resource reservation and without resource reservation. In the resource reservation approach, sensor nodes are being reserved before the task mapping applied, so that this method is a guaranteed way of mapping, however, it is more time consuming and execution time of the tasks are longer. Tasks are queued and wait until the processing element finishes the execution of the previous allocations. Resource reservation approaches are known as the guaranteed way of task mapping, whereas resource reservation approaches may result in higher execution time due to the long queues on the processing element. On demand dynamic mapping without resource reservation applies a mapping heuristic, if available processing element exists. This method is not a guaranteed way of resource management but speed ups the system [133]. Mapping the task to one of the available processing element avoids queues and this speeds up the system. Although sensornet applications use multi-tasking in many cases *number of the task mapped per sensor node* may differ according to the used heuristic; either in the form of mono- tasking or multi-tasking. In mono-task approach one task can be mapped to a processing element, on the other hand in

multi-task approach several tasks can be mapped to a specific sensor node. Tasks are mapped to sensors for the task execution on different time durations depends on the selected mapping heuristic. In order to maintain the desired performance network control system plays an important role. *Mapping control system* defines whether the mapping process is done in a centralised or distributed manner. Distributed mapping requires specific control mechanisms in each region (region here can be defined as part of the platform) to control and perform the mapping, whereas in centralised mapping methods only one network element is responsible for the mapping. Required network element may or may not have the same physical characteristics as the nodes. As the network *architecture model* (platform model as we referred previously) may consist of either homogeneous or heterogeneous network elements, the mapping technique should be able to cope with the difference in the network elements to maintain the desired service availability, QoS demands and performance efficiency [133].

Summary

This section overviews task mapping techniques, and categorises and defines the task mapping process. In order to avoid repetition, this section has not presented many examples of existing techniques. In the next section, we define bio-inspired task mapping techniques and discuss significant examples that apply bio-inspired task mapping.

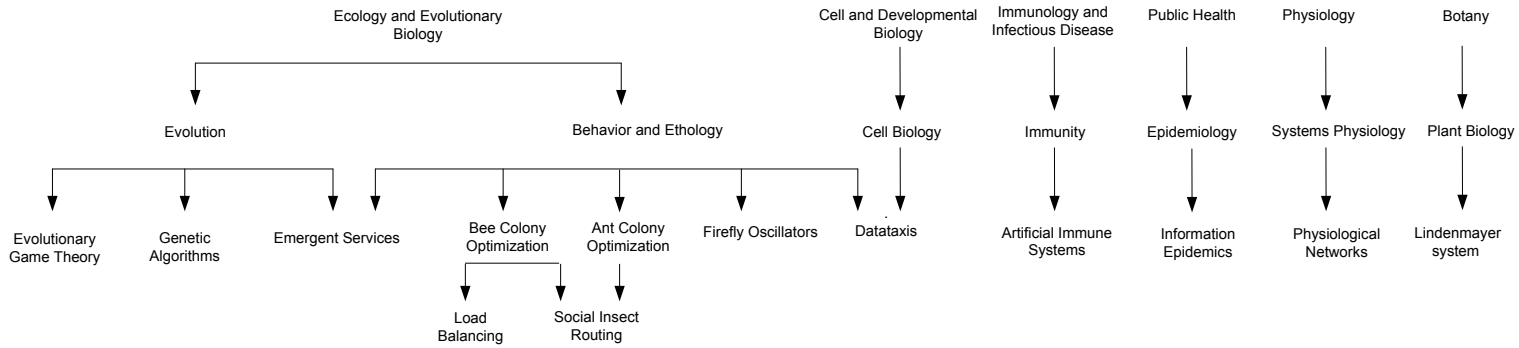
2.1.2 Bio-inspired Task Mapping Techniques

"We believe that the challenges faced by future network applications, such as scalability, adaptability, and survivability/availability, have already been overcome by large scale biological systems and that future network applications will benefit by adopting key biological principles and mechanisms" [184].

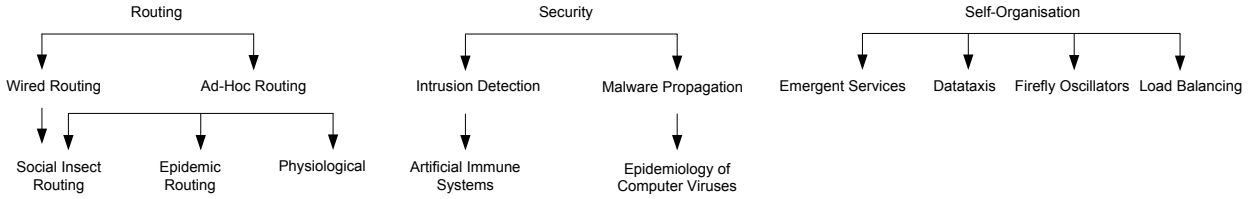
Different kinds of optimisation techniques will be explained in the next section in detail, however, we begin by stating the advantages of bio-inspired techniques over other optimisation techniques. Many of the traditional optimisation techniques,

particularly analytical techniques, require high computational effort and long working hours. Additionally, required time and effort grows exponentially as the size of the problem increases. Although some traditional optimisation techniques are very effective, they require high computational processing which cause sensor node to run out of energy even faster. A lightweight optimisation method that requires moderate memory and computational resources and yet produces good results is desirable, especially for implementation on an individual sensor node [96]. Bio-inspired optimisation methods are lightweight, so they are computationally efficient alternatives to analytical methods and other traditional optimisation techniques. Moreover, most of the bio-inspired approaches aim to overcome scalability, adaptability and survivability issues of WSNs using the certain characteristic of the particular inspiration. The categorisation of inspirations and their area of application are defined below, but first we give an overview of what is organic computing and bio-inspired behaviours.

Organic computing is defined by Dressler [42] as covering the bio-inspired mechanisms in engineering and computer science related fields, that attempt to build high-scalable architectures, which are self-organising, self-maintaining and self-healing [42]. Biologically-inspired techniques are a field of study of observation of the social, environmental and physiologic behaviours of complex biological systems. Biologically inspired networks are the distributed, autonomous sensor networks which have inspirations from biology [42], [43]. Biological inspirations are used to make systems more reliable, efficient and self-organised. There are many different ecological groups in different categories in biology to observe. The main concern is to make the link between the computer networks or computing in general and the ecological systems. In complex systems such as human beings, bees, ants, termites and birds, there are many trade-offs in those systems as well as computer networks. The challenging trade-offs in WSN depends on limited network resources, QoS, cost and energy consumption.



(a) Classification of organic computing based on inspiration



(b) Classification of organic computing based on area of application

Figure 2.2: (a) Classification of organic computing based on inspiration, (b) Classification of organic computing based on area of application [115].

Moreover bio-inspired computing is described as a class of algorithms focusing on efficient computing, e.g., for optimisation processes and pattern recognitions [43]. Categorisation of organic computing together based on their inspiration and area of applications are shown by the Figure 2.3 based on Meisel et al.'s [115] research and we have extended Meisel et al's model to include latest research on the bio-inspired techniques. As Figure 2.3 illustrates, organic computing is divided into categories such as cell and developmental biology, ecology and evolutionary biology, immunology, epidemiology and physiology. The biological inspirations are presented in 2.2b whereas the area of application of the given biological fields are illustrated in 2.2b.

Inspirations based on ecology and evolutionary biology are also divided into two main subgroups; evaluation and behaviour, and ethology [115]. Evolutionary game theory, and genetic algorithms are categorised under the evolutionary subgroup, whereas ant colony optimisation, bee colony optimisation, and firefly oscillation are categorised under behavioural subgroup [115]. Emergent services are inspired from both evolutionary and behavioural subgroups. The rest of this section overviews each category based on the biological inspiration and provides some of the significant examples on the application area of WSNs.

We would like to start with the **Emergent Services** as most of the research represented in this section contains emergent behaviour. Emergent behaviour from a biological point of view is the behaviour observed from a colony/ group, flock or shoal that arises out of the product of relatively simple interactions by its individuals. The patterns of behaviour observed by the group (colony, flocks or shoal) are totally distinct compared to the exhibited behaviours of the individuals. We give an example for better understanding. In a flocks of birds, birds fly in a certain shape (V-shape) during their migration, and each bird changes its place in that shape from time to time based on their interaction: move along side the flock, go to the back or to the front. The simple interaction between the birds allow them to fly faster, and consume minimum energy during the flight, which benefits to the entire flock. The emergent behaviour observed in this example is the special V-shape that the flock is formed. Individuals alone cannot fly in this form and gain the same bene-

fits. Bio-inspired computing and techniques are widely used to benefit more from the simple interactions, just like the flock example. However, evolving desirable emergent behaviour in our man-made systems is not easy and as we may benefit from applying a bio-inspired technique, we may also face with unexpected/unwanted emergent behaviour that the system is generation. From the computer science point of view, emergent behaviour is the consequences of our bio-inspired protocols on a system. An emergent service is the positive consequence (expected benefits) towards achieving the main contribution on the system performance. On the other side, observed repercussions of the applied bio-inspired technique (if any applicable) is still a subset of the emergent behaviour of the required bio-inspired technique, however, negatively unexpected/unwanted emergent behaviour is not identified as an emergent service and it is a sort of property that we want to minimise.

Evolutionary Game Theory was devised by Borel [39] in 1921 to evaluate multi-objective problems. Game theory represents multi-objective optimisation with multiple decision makers, each controlling certain design variables for the given problem [112]. Game theory is used to tackle sensor network problems in the terms of energy efficiency [20], [53] and security issues such as denial of service (DoS) attacks [114], [3]. Most of the studies focus on energy conservation of routing [120], [83], [35] however, only a few researches particularly focus on load balancing [203], [153]. Game theory in WSN load balancing and especially in task mapping, is mainly used for offline approaches like parameter tuning [83].

In 1975 Holland [68] introduced **Genetic Algorithms (GA)**. They are considered as global optimisation techniques based on population rather than individuals. GA loosely parallel biological evolution based on Darwin's theory of natural selection. Populations represent a subset of possible solution space, whereas generations represent the algorithmic iterations. Genes in the gene set (chromosome) is the design factor and alternate to find (near) optimal solutions. GAs can be easily applied for single-objective optimisation. Adapting GAs for multi-objective optimisation needs tailoring, but has been commonly used since VEGA in 1989. Zeng et al. [205] present a static task mapping approach based on a GA for WSNs, 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. Most of the GA approaches use static optimisation techniques like [205] due to their computational cost. On the other hand, very few dynamic approaches also use GA at runtime [74]. Jin et al. [74] 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. [123] also present a genetic approach, involving genome mutation and crossover to increase the performance of the large scale WSNs.

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 [61], herds of sheep [57] and, bacteria colonies [147]. Although not all the comprises fileds of SI is yet applied into WSNs, we present existing examples as bio-inspired approaches. 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 [19]. Groups of animals such as shoals of fish increase individual and group well-being by synchronising their motion. One of the well-explored and frequently used branch of swarm optimisation is known to be the **Ant Colony Optimisation (ACO)**. ACO is based on the observation of the collective foraging behaviour of ants [16], [43]. With the use of ACO, many research studies are held on social insect routing based on the ability of ants to converge on the shortest path from their nest to a food source. Ant Colony Routing Algorithm (ARA) [62] aims to reduce routing overhead for mobile ad Hoc networks. The algorithm enable of route discovery, maintenance and failure handling to increase the successful transmission rate whilst reduced network traffic, but does not consider energy-usage. Similarly, AntHocNet [38] does not consider the energy issue but proposes efficient network routing with high successful transmission rate and short delay time by providing alternative routing paths in case of route failures. Energy Efficient Ant Colony Based Routing (EEABR) [25] aims to prolong network lifetime by increasing successful communication ratio with

a centralised approach. Each node stores limited information in their memory for the routing tables and the minimal memory consumption enforced by the approach leads to minimal power consumption. The algorithm considers not only the length of the path but also the energy level of that path because it is always preferable to select the shorter path with high energy level than the longer path with low energy level. The biggest disadvantage of the approach is the centralised control mechanism, which limits the entire usage of the proposed technique. Yingzhuang Liu et al. [201] designed an effective ant colony based routing algorithm. Ant Colony Multiple QoS Constrained Routing (AMQRA) the authors proposed extended EEABR for mobile ad-hoc network to improve the packet delivery ratio, and reduce the end to end delay. Unlike EEABR, AMQRA decides the path based on a cost function to reduce the delay and communication loss rate while increasing the QoS and optimised bandwidth usage. Oktem and Karaboga [131] proposed advanced EEABR target to improve network lifespan by finding the shortest route using BCO with a decentralised approach. Their algorithm proposes a decentralised approach based on based on the nodes energy level and the pheromone value. ARO [195], ANT-E [161], EARA [6], and FACO [59] are some of the other ACO-based approaches that target efficient routing. Only ARP and FACO are energy-aware among these protocols, whereas most of them concern about reduced network overheads and high delivery ratios. Although PERA [10] is a ACO inspired protocol, unlike ARP and FACO, it is based on probabilistic decisions.

Conforming to this swarm metaphor, **Bee Colony Optimisation (BCO)** was introduced by Karaboga et al. [85], [84]. In their artificial bee colony algorithm (ABC) model, bees represent search agents and their environment the space of potential solutions, to a given problem 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 by Karaboga. Karaboga et al. improved their technique in [4] by tuning its parameters and modifying their initial work [88]. In the Beehive protocol [186] 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 [156] 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 [140], 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 [17]. Previous work has also mapped the bee colony model more directly to WSN hardware, with individual nodes representing individual bees, 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 [160] which is inspired by the bees mating procedure. This approach focuses on cluster set-up communication overheads by restricting the communications with bee mating election algorithm. Removing the redundant communications inside the cluster increase the successful delivery ration whilst decreasing the latency.

Firefly oscillations aim to develop self-organisation mechanisms commonly to enable robustness on massively distributed complex system particularly for clock synchronisation. Firefly synchronisation is based on pulse-coupled oscillations, the basic mathematical model of which has proposed by Richmond [150]. Many other mathematical models are developed on firefly osculations in the last decade, whereas researchers started to use firefly osculations to improve self-organisation based on Strogatz and Mirollo's M&S model [124]. Allen et al. proposed Reachback Firefly Algorithm (RFA) [189], that improves the MAC layer timestamps and response mechanism. In [105], Cui and Wang improved the RFA technique by considering message delays with a late sensitivity window to reachback firing scheme (RFA with LSW). By neglecting the delay messages, they optimised the network load and coverage rate. Dynamic changes on the network scheme will bring many disadvantages to this method, where the algorithm heavily depends on the parameters selected for the window range. Similarly in 2012, Sun et al. [200] developed their firefly

synchronization for clustering in WSN based on RFA. Enabling virtual clusters and synchronizing clusters individually lead mitigation in the network load.

Datataxis (or datacabs) have inspirations from both behavioural and cell biology that are used to develop self organisation mechanisms. The inspirational point comes from the bacteria, where phenomenon of a bacteria naturally moves towards a higher concentration (of phenomenon) [2], [125]. Datataxis are mainly used for vehicle routing (VANETs) to allow larger area coverage and maximise data harvesting. Since vehicle devices are limited in storage, losing the accumulated data have always been a problem to tackle. Datataxis have started to be used to gather the stored information from the vehicle especially in concentrated areas. Ensuring application safety and time constraints are some of the key concepts of mobility of WSNs, which most of the research held on. ADCD [64] uses datataxis to manage information harvesting, and distribution.

Artificial Immune System (AIS) which is listed under Immunology and Infectious Diseases according to Meisel et al. Figure 2.3, is inspired by the human/-mammalian immune system. Sensitivity to detecting environmental change, and identifying the foreign/infectious agents is used in WSNs, particularly for security purposes in anomaly detections. SASHA [15] implements a self-healing fault-detection mechanism for faulty sensor reading. Sarajanovic and Le Boudec present their incremental work on anomaly detection for mobile ad Hoc networks based on AIS in [159], [158]. In both papers the authors are using immune-based mechanism to improve the self-learning and adaptation on an existing routing algorithm; DSR. Simulation results show that by improving introducing the self-adaptation mechanism to an existing protocol perform better in terms of increased successful transmission rate and less dropped packets. DNRS [7] is an artificial immune system scheme which aims to limit energy consumption while retaining event detection reliability by changing the signal frequency of the nodes. By enabling dynamic voltage change, DNRS reduces the energy consumption in an autonomous way.

Information Epidemics is listed under public health and inspiration is used to improve epidemic routing particularly in WSN. As the diseases/viruses spread in humans, animals or plants, self-replicating malicious programs in computers (dis-

eases of computers/computer viruses) also spread the same way and based on this inspiration information epidemics are used to increase the level of security in computer science, especially in WSNs. Epidemic routing in WSN (and in mobile ad hoc networks) propose robust algorithms that minimise the delay and remove the redundancy. In [180] any information from any sensor node is replicated and forwarded to every network entity (flooding) which results with minimum data loss but duplications of unnecessary redundant information. PREP [146] prioritises the packets on the buffer, however, still spreads the information based on [180]; with high network overhead.

Lindenmayer Systems are inspired from botany and initially modelled by A. Lindenmayer in 1968 [106]. Later on mathematical models are combined by genetic algorithms started to be used for self-organisation, self-healing and self-adaptation mechanism. Although this technique is not commonly used in WSNs it is important to cover this category of biological inspiration. Ponnusamy et al. [143] used the botany inspired self-healing mechanism. In [143] an energy efficient routing algorithm is initiated where the dynamic nature of the algorithm avoids to have gaps on the routes.

We have presented different categories of biological inspirations and gave some significant examples of each category together with their fields of application. However, there are many important protocols which we could not present in this section in details. Table 3.1 summaries the application fields of each biological category and provides some other interesting existing work in the literature.

Summary

In this section we have provided background knowledge for load balancing in WSNs, and categorised the existing literature based on the area of implementation in the OSI stack. In Section 2.1.1, we defined and categorised task mapping techniques which are often used as a way to load balance in the WSNs. Later on, we focused on bio-inspired task mapping techniques in Section 2.1.2. We categorised existing biological inspirations and give examples for each category from the literature.

In short, the need of collaboration makes network components autonomous and

Table 2.1: Biological inspirations and their fields of application in WSNs

Biological inspiration	Application fields in WSN	References
Game Theory	Used to improve the security and reduce the energy consumption	[203], [153], [120], [83], [35], [114], [3]
Genetic Algorithms	Used to optimise algorithms / parameters	[97], [13], [54], [79], [206]
ACO	Used to improve energy efficiency and QoS in routing	[201], [10], [59], [62], [25]
BCO	Used to improve energy efficiency and QoS in routing and load balancing	[87], [86], [131], [89], [5]
Firefly Oscillators	Used to improve self-organisation and clock synchronisation	[189], [105], [152], [200], [128], [50], [135], [196], [179], [118], [178], [197]
Datataxis	Used to allow larger area coverage and maximise data harvesting for vehicle networks	[99], [134], [99], [64], [2], [125]
AIS	Used to identify foreign molecules and produce beneficial cells for removing the foreign molecules	[37], [171], [8], [7], [104], [155], [154], [129], [48], [15]
Info. Epidemics	used to maximise the successful message transmission rate and reduce the latency	[92], [78], [75], [71], [66], [170]

self-organised, so that each piece of collection may rely on its own decisions based on its local interactions. Developing autonomous, and self-organised system based on the biological inspirations make systems more robust to changes in the network operational state, independent from the node deployment and its deterministic structure. In the given examples of existing researches in almost all the distinct bio-inpired

fields, we show that the presented techniques encourage emergent behaviour and improve performance objectives. In the next section, we will provide the essentials on performance evaluation. We will describe performance metrics used in WSNs, categorise them and discuss the evaluation techniques for those metrics.

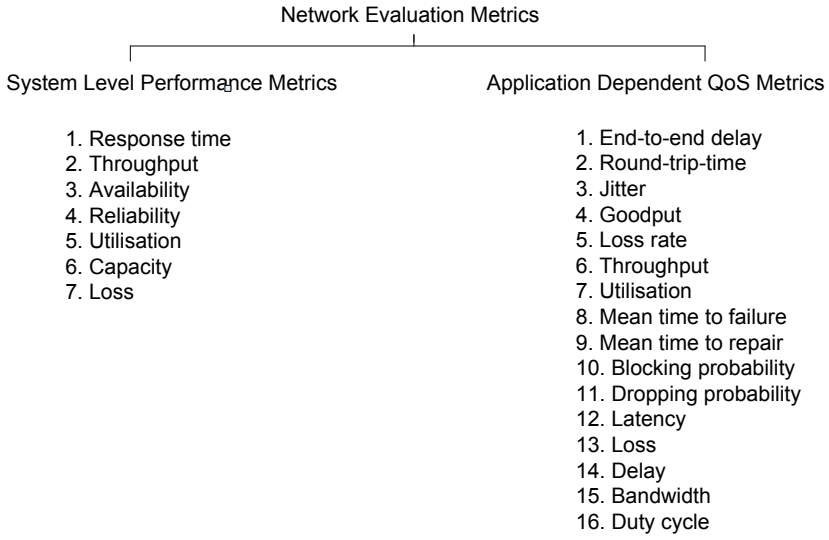
2.2 Performance Evaluation of WSNs

We have defined WSNs as small, autonomous, self-powered devices that are envisaged for industrial, civil and military purposes to monitor, detect and track events according to application requirements. We have motivated the researches on WSN with the limited resource capacity of the WSNs, and the need to improve their performance by developing self-adaptation, self-healing, and self-organisation mechanisms. The only way to critique our proposed techniques is to apply them on WSNs and compare them against each other in the most fairly way. In the rest of this section, we will provide information of performance evaluation metrics and the evaluation techniques these metrics.

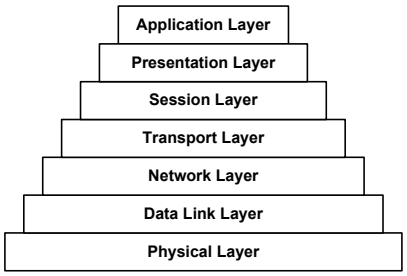
2.2.1 Performance Evaluation Metrics

Pehcevski and Piwowarski [138] defined evaluation metrics as following: *"An evaluation metric is used to evaluate the effectiveness of information retrieval systems and to justify theoretical and/or pragmatical developments of these systems. It consists of a set of measures that follow a common underlying evaluation methodology."*.

In [29], and [108] commonly used network performance metrics in WSNs are inspected and defined. Here, in this section we will focus on how these network performance metrics are different than each other and what are the possible categories of these metrics. It is essential to specify the network performance metrics and select the network evaluation technique and the tools to use based on the specified metrics. In Figure 2.3a we present our categorisation of network performance metrics and in Figure 2.3b OSI model is illustrated. We divide network evaluation metrics into two based on the level at which they are measured: system level performance metrics and application-dependent QoS metrics. System level performance



(a) Categorisation of Network Evaluation Metrics



(b) OSI Model

Figure 2.3: (a) Categorisation of network evaluation metrics, (b) The OSI model [207].

metrics have high level aspects of the networks, whereas QoS metrics has low level aspects of the system which are often application dependent and subjective. System level performance metrics heavily depends on the QoS metrics. For instance, an application specific metric would be associated with the application protocols' performance or a specific application goal, for example in the habitat monitoring protocol "packet loss rate" or "proportion of wildlife detected", or "proportion of energy spent on packets". Whereas system level performance metrics are more general and are shared with all elements of WSN deployments, e.g. "time to first battery exhaustion". Most of the system level performance metrics exist in the top layers

of the OSI model; in application, presentation or session layer. In order to apply system-level performance metrics, the technique under analysis has to take place in one of the top layers, and the performance is evaluated from the systems' point of view. Similarly, application dependent QoS metrics exist in lower layers of OSI, and evaluation of these metrics needs the application load on the lower layers of OSI, such as transport, network or MAC layer [207].

In the last decade, with the growing interest in WSNs, improving application dependent QoS metrics is becoming more important. Network performance can be increased by increasing goodput, bandwidth usage, throughput, utilisation and reducing delay, jitter, time to repair and packet loss rate. Little research has focused on system-level performance metrics. Not only is there an absence of research focusing on system-level performance metrics, but there is also a big gap in evaluating the network performance for the entire system. One of the important points in evaluating application dependent QoS performance metrics is evaluating the systems performance based on the provided technique. Imagine a COFFEE protocol that increases successful packet transmission by 20%, and reduces the packet loss rate significantly. For this example, we can not claim that COFFEE protocol increases the system performance 20%, especially if the proposed techniques combine different methods. Interactions on different layers of OSI can easily have detrimental impact on the performance, which in such a case also has to be reported. Scientifically, it is also very important to evaluate the COFFEE protocol's effect on the overall network performance, not only based on the successful packet transmission or loss rate.

2.2.2 Performance Evaluation Techniques

In the previous section we categorised the performance metrics used in WSNs into two and discussed the importance of evaluating approaches from the point of application dependent QoS metrics and from the overall system point of view. In this section, we will define, describe and classify the existing performance evaluation techniques of WSNs, together with the existing related research on WSNs.

The selection of performance evaluation metrics is a significant factor in showing

the effectiveness of the proposed techniques. In some domains this selection might be very obvious. For example, most of the routing protocols analyse the successful transmission rate and packet loss rate as a primary or at least as a secondary performance evaluation metrics; these two metrics are commonly used and will be questioned and/or asked if one does not considers them in this context. In some cases, when the application domain is not well-known, the appropriate evaluation metrics may be less obvious, but some can be guided based on existing work. On the other hand, in the context of selecting the performance evaluation technique can be more open to personal decisions that it allows researchers to choose the evaluation tools they will use based on their personal choices. It is absurd to select a high-level performance evaluation technique to evaluate a performance metric that relates to lower levels of OSI system. However, there are plenty of different kinds of tools in different categories so it allows researchers to have personal decisions on tools they will to use. The rest of this section, we categorised the performance evaluation techniques based on their types, and present significant tools used in each category.

Experimentally evaluating the performance of novel algorithms is a fundamental focus of WSN research [163]. Existing evaluation concepts include system-level simulators, low level simulators, and prototypes. Since WSNs are application specific environments, researchers choose the most appropriate concept to evaluate their target application area. Egea-Lopez defined the key points on the performance evaluation techniques as follows; *"A 'good' model based on solid assumptions is mandatory to derive trustful results. One of the challenging parts is to choose the suitable simulation tool to represent the nature of the WSNs. The fundamental trade-off is: accuracy and necessity of details versus performance and scalability" [47].* From our perspective the key criteria in choosing the most appropriate performance evaluation technique are: flexibility, scalability, complexity, implementation time, performance efficiency, financial cost and accuracy. Table 2.2 compares and contrasts three performance evaluation techniques based on the key design factors. Design factors are marked with either low, medium or high for each performance evaluation criterion.

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

Models	Cost	Scalability	Flexibility	Accuracy	Complexity	Efficiency	Time
System-level Simulators	Low	High	High	Low	Low	High	Low
Low-level Simulators	Medium	Medium	Medium	Medium	High	Medium	High
Prototypes	High	Low	Low	High	Medium	Low	Medium

Table 2.2: Comparison Between The Performance Evaluation Techniques.

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. The implementation duration of low-level simulations takes more time, 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 the fact that they abstract away 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 [46], [47], [94]. Sensor emulations are hybrid approaches which combine components of real hardware and simulated results; they are combination of low-level simulators and hardware prototyping. Emulators run at either in the bit-level or in MAC-level. Emulators financially cost more than low-level simulators, and they have low scalability. In terms of accuracy, emulators are as accurate as the prototypes and they provide more accurate results than system and low-level simulators. Testbeds are designed for emulators and simulators to prevent unrealistic assumptions and inaccurate results. So we consider testbeds mainly part of low-level simulators like emulators.

In the discussion above, we compared different types of performance evaluation techniques based on their cost, scalability, flexibility, accuracy, complexity, efficiency and implementation time. Some of the existing tools related to WSNs evaluation in

Table 2.3: Comparison of Performance Evaluation Techniques

System-level simulator	ETSSI [1], Fast, SuperFast
Low-level simulators	NS2 [113], OMNET++ [181], J-SIM [165], GloMoSim [204], OPNET, Ptolomy II, Jist [11], SSFNet, Castalia [137], VisualSense, Viptos [30], Sidh, Prowler, sens.
Emulators	TOSSIM [101], ATEMU [142], Avrora [172], Sense [168], Emstar [58], MINT [182], Salt, EWANT.
Testbeds	Motelab [188], SensorScope [73], Gnomes, emulab [139], Signetlab [34], ORBIT.

the literature are listed in Table. 2.3 based on the categorisation of the performance evaluation techniques. Each of the provided simulator/emulator/ testbed has very specific characteristics. For further information on the features of each simulator [44], [93], [47], [46], [193], [90], [166], [72], [187].

Summary

In this section we have described the importance of performance evaluation in WSNs, categorised the performance evaluation metrics, and discussed the advantages and disadvantages of different types of the performance evaluation techniques. We have listed the existing work on types of performance evaluation techniques, and focused on the system-level simulators. The number of system-level simulators that exist in the literature is significantly less than low-level simulators/ emulators/ testbeds. On the other hand, prototyping is also commonly used in many research studies. As we mentioned, the main disadvantage of the system-level simulators are their level of accuracy. Considering the amount of the research that focused on low-levels of OSI stack, it is clear that few research studies and natural for scientists to prefer low-level simulators rather than system-level simulators. However, we underlined the importance of providing the overall system influence on the evaluation protocol,

and highlighted the gap in the literature on this topic. In the next section, we will discuss the optimisation techniques for WSNs.

2.3 Optimisation Techniques

Previously, in Section 2.1.1, we discussed task mapping optimisation and why optimisation is needed in WSNs. In this section we discuss different types of optimisation in WSNs, give examples of some of the applied techniques used for WSNs optimisation, and discuss the kinds of problem tackled by these techniques. The resource limited structure of WSNs forces researchers to manage network resources more efficiently. One way to improve resource management and solve the WSN trade-off is to use optimisation techniques. Many of the design decisions found in WSN development relate to meeting application requirements that target to optimise WSNs. Based on the application-specific design decisions optimisation techniques might target challenges in:

- managing network resources, from the system point of view
- managing node resources, from an individual's point of view
- managing operating environments.

Managing network resources from the system point of view involves decisions like scheduling, sampling, computing, and communication. These approaches use the system perspective to guide sensor nodes take certain actions at certain times independent from the control mechanism (centralised or distributed control). Network dynamics (e.g. mobility of nodes, sink) play an important in network management from the system point of view, which affects scheduling [65], sampling, computing and communications [63]. Managing node resources from functionality of an individual node focuses on the maximum usage of the node capacity by adjusting the bandwidth, signal frequency, processor voltage and frequency, memory, computational resources, or energy. Operating environments consist of dynamic topology changes [33], mobility [159], localisation [162], deployment density and spatial distribution.

In WSNs, many research studies exist in the literature that try to improve the concepts presented above using different kinds of optimisation techniques. In the next section, we present some of the traditional optimisation techniques. Section 2.3.2 overviews metaheuristic search techniques that have been applied in WSNs, focusing on Simulated Annealing.

2.3.1 Traditional Optimisation Techniques

Task mapping techniques are presented in Section 2.1.1, as an optimisation technique that is commonly used in WSNs. In this section, we describe some other optimisation techniques and give examples of each technique from the existing literature. Linear/ non-linear programming, integer programming, Pareto optimisation, heuristic and metaheuristic optimisation are some of the other optimisation techniques that are used in WSNs to address problems in addition to task mapping and resource allocation. Many heuristic techniques are presented in Section 2.1.2 about bio-inspired task mapping, and we present metaheuristic optimisation techniques separately in the next section. Here, we focus on traditional optimisation techniques briefly.

Linear Programming (LP) provides (near) optimal results based on a given mathematical model of the problem under scrutiny. [80] formalise the network model to maximise the network lifespan and focuses on schedule flows within the network. In [76], link capacity optimisation in WSNs is applied using LP to simultaneous optimise the of channel partitioning on code-division multiple access (CDMA) of routing and power allocation.

Non-linear Programming (NLP) provides (near) optimal results based on the given mathematical of the problem; they are composed of an objective function, general constraints and variable bounds, just like LP. The difference between linear and non-linear programming is the fact that a non-linear program includes at least one non-linear function, which could be the objective function, or some/all of the constraints. Many real systems are non-linear so optimisation methods can cope with them. [132] uses formal methods by developing non-linear optimisation models, that aims to minimise the energy usage whilst maximising the data gathering

(from sensing, transmission and processing). Similarly, in [95] fundamental limits on the performance of information routing are modelled using non-linear optimisation technique and maximum information extraction, whilst involving minimum total energy usage. Fairness constraints on the provided routing algorithm are also analysed based on some design factors that play a role in the topology, number of nodes, energy levels, source rates, reception power. [76], [194] considers the physical channel constraints in the optimisation models.

Ciciriello et al. [31] presents an efficient routing schema for decentralized WSNs that adapts the topology by maximising the overlapping among source-sink paths. This, therefore minimises the overall number of network links exploited. The range assignment problem for heterogeneous sensornets are tackled in [27] by using integer programming. The proposed algorithm selects the (near) optimal transmission range for each node such that the energy consumption is minimised on the multi-hop communication, between the nodes and sink. Meaning algorithm minimises the maximum transmission power consumed at each sensor nodes.

2.3.2 Metaheuristic Optimisation Techniques

Clark et al. [32] define metaheuristic optimisation techniques as, "*a set of generic algorithms that are concerned with searching for optimal or near optimal solutions to a problem within a large multi-modal search space and have been used to find acceptable approximations to the solution of many NP complete problems.*" They divide metaheuristic search techniques into two; local search techniques and evolutionary search using genetic algorithms. Hill-climbing, simulated annealing and tabu search are categorised as local search, whereas evolutionary search using genetic algorithms (GA) is categorised separately as population based search metaheuristic. All the descriptions can be found in [111]. As Simulated Annealing is a robust and widely used technique in WSNs, we describe Simulated Annealing in detail and give some of the significant examples on WSNs.

2.3.2.1 Simulated Annealing on WSNs

Alike to all the other local metaheuristic search techniques, the moves applied in simulated annealing are also based on relative desirability/ undesirability of partic-

ular local information. They are easy to program and time efficient due to their low level of complexity. Unlike *GAs*, *SA* is applied to the single individuals rather than a population. Since the solution domain is sampled through all the population in GA, evaluation of the fitness function might be costly and take a long time without achieving to the global optima. Although *exhaustive search* algorithms cover the whole search space, they are impractical, computationally unaffordable and time inefficient for multi criteria metrics with several parameters. They are also infeasible if the search space is infinite.

SA is a stochastic optimization procedure for obtaining approximate solutions to combinational optimisation problems. The idea originated from a thermal process by obtaining low energy states of a solid metal in a heat bath [119]. The temperature of the heat bath is increased until the metal melts and then cooled carefully and slowly until the atomic rearrangements of the metals turns into solid state. *SA* is an iterative process where the system starts rearranging itself until an improved configuration of particular solution is found. Once the solution is found, then that particular solution becomes the new starting point for the further rearrangements. The iterative process will continue until the system achieves the stopping criteria, where no further improvements can be found. This simple idea is often used to find feasible solutions which can converge to an optimal solution.

Kirkpatrick et al [91] took the idea of annealing process and applied it to optimisation problems, and since then *SA* has been successfully used in many diverse fields of computer science; artificial neural network [56], pattern detection [70], NoC [110], mobile ad-hoc networks [177], model-driven engineering [145] and software verification [174], as well as WSNs. Some of the applied work on WSNs in this section is as follows. Kannan et al [81], [82] present a simulated annealing metaheuristic for WSNs that aims to localise network nodes accurately for centralised architectures. Their technique reduces large scale localisation errors (flip ambiguity) significantly by applying static accurate position determination by SA. The fitness function measures the sum of squared distance over all pairs. Slijepcevic and Potkonjak [164] present a search heuristic which aims to find the optimal number of network nodes. Deterministic node placement in clusters is also implemented to maintain high net-

work coverage with minimum energy consumption. The fitness function subtracts approximate measurement of minimally constraining heuristic from the most constrained heuristic in terms of the number of nodes. Wang at el [185] exploits a novel fault-tolerant distributed multiclass classification fusion approach using error correcting codes (DCFECC) that provides excellent fault-tolerance capability in WSN. Park and Srivastava [136] present centralised task decomposition, transformation and assignment solution using SA to maximize the lifetime of the network and/or minimize the latency. Their work also includes a distributed task migration algorithm at run-time which occurs based on the results of the SA search. The fitness function measures total energy consumption, which is sum of communicational and computational consumption for all tasks, weight of latency, weight of maximum energy consumption and penalty. RUGGED [51] energy efficient, fully distributed and reactive routing protocol based on information gradient uses simulated annealing concepts to discover single and multiple path explorations. Montemanni et al [126] combine mixed integer programming with SA metaheuristic to achieve minimum power consumption for broadcasts. Euclidean distance between nodes, channel loss exponent, the power required to reach from source to destination calculated and used as a parameter to measure the sum of the transmission powers of all the nodes.

More details about these techniques can be found in Table 2.4.

Table 2.4: Literature Review about Simulated Annealing on WSNs

No	Aim	Tuned Parameters and Fitness Function	Results
[81] [82]	First phase is to estimate the location of nodes accurately using simulated annealing, whereas second phase is to optimise the nodes which are likely to have flip ambiguity due to large scale localization errors.	N: number of non-anchor nodes, N_i: set of neighbours of node i, $d'_i j$ and d_{ij}: estimated and measured distance of node i with its neighbour j respectively Permutation distance (Δd): Distance between one hop neighbours using known coordinates of anchors (x_i, y_i) Fitness Function: CF=$\sum_{i=1}^{N}\sum_{jN_i}(d'_{ij} - d_{ij})^2$.	Simulations performed on a total of 200 uniformly distributed nodes in a square region of 10X10. Location of anchor node, original location of nonanchor node, estimate location of nonanchor node, error offset between original and estimated location of nonanchor node are analysed.
[164]	Finding an optimal number of network nodes and implementing deterministic placement in clusters to maintain a high network coverage with minimum energy consumption using simulated annealing	A: set of all the fields in the sensor located area. C: set of sensor nodes.C_i: sub collections of nodes (clusters). N: number of deployed sensors. k: number of covers. U: unmarked elements of A. V: set of available elements (current members of C). e_{min}(critical element): member of the smallest number in the available sets from V, which is selected among U. m: uncovered elements of A. Fitness Function: Subtracting approximate measurement of minimally constraining heuristic from the most constrained heuristic in terms of number of nodes. CF=$a\sum_{i=1}^{kj} 1 + b\sum_{i=k+1}^{nj} 1/m$	Simulation results analyse average number of covers, runtime duration is analysed to see the effects of the number of the sensors, number of the fields (clusters) and sensing radius.
[185]	Classification fusion via error correcting codes to incorporate fault-tolerance capability while reducing energy consumption, computation time and memory requirements.	T: code matrix to find minimum Hamming distance between vectors. $(T^(o)) = t^(0), 1), t^(0), 2), t^(0), N))$. $t^(0), j)$: jth column vector of code matrix$T^(o))$. P_e: probability of error (misclassification in the code matrix). Fitness Function: The probability of decision error = $lim_{N\to\infty} inf - 1/N log P_e \geq min_{\{0\leq i\leq M-1\}} D(\delta||\beta_i) > 0$	Probability of misclassification with three code matrices with different minimum Hamming distances Probability of misclassification of DCFECC and FCA with variety channel transmission error.

Table 2.4: Literature Review about Simulated Annealing on WSNs

No	Aim	Tuned Parameters and Fitness Function	Results
[136]	1) centralised task decomposition/transformation and assignment for design-time, and 2) distributed task migration for run-time support. The remainder of this paper is organized as follows.	Task Decomposition/Transformation and Assignment phase: Finding task transformation T, task assignment A. Task Scheduling Phase: radio range assignment and task scheduling. Fitness Function: Total energy consumption; sum of communicational and computational consumption for all tasks, weight of latency, weight of maximum energy consumption and penalty.	Energy consumption of communication, computation, maximum and total discharge rates are shown in details at certain conditions.
[51]	Energy efficient, fully distributed and reactive routing protocol based on information gradient is designed. Single path and multiple path exploration to discover routes is achieved by simulated annealing concepts.	d_i: distance of the location from peak information point, $f(d_i)$: gradient information of the location with environmental noise, f_{max}: peak information, $f^*(d_i)$: gradient information of the location without environmental noise. Fitness Function: $f(d_i) = f(d_i) fEN(f^a(d_i))$, $fEN(f^a(d_i)) \infty (fmax f^a(d_i))$	Simulation are validated on 1)100x100 grid of 10000 sensor nodes; 2) Sensor field of dimension $225375m^2$. Analysis on effect of at information region nodes, effect of malfunctioning nodes, query failure rate, average energy dissipation without lter to avoid malfunctioning nodes, average energy dissipation with lter, percentage of sources found vs number of sources, average energy dissipation has been explicitly illustrated.
[126]	Mixed integer programming formulation is combined with a new heuristic approach for the minimum power broadcast problem in wireless networks using simulated annealing algorithm.	d_{ij}: Euclidean distance between nodes i and j, k: channel loss exponent, $p_{ij}(The power required to reach from i to j) = (d_{ij})$ Fitness Function: Sum of the transmission powers of all the nodes.	Simulations held on 5x5 grid with 25, 50,75,100,150 and 200 nodes. Percentage improvements (%) in mean tree power of SA + sweep over BIP. Percentage improvements (%) in mean tree power over BIP Algorithm are analysed and comparison between other approaches are also illustrated in the result section.

Summary

In this section, we define and present some examples of the traditional and metaheuristic optimisation techniques. Traditional optimisation techniques have been effectively applied on WSNs, however, due to the complexity of the problem it is not easy to apply traditional optimisation techniques to cases such as WSN. It is important to underline that task mapping and resource allocation is orthogonal to traditional optimisation techniques. Task mapping is an application of optimisation, the traditional optimisation techniques we mentioned in this section are approaches to optimisation. Metahuristic search techniques in WSNs have been used to optimise algorithms offline for the fine tuning. In the next section we discuss about the network coverage and provide some of the examples of increasing the network coverage in WSNs.

2.4 The Network Coverage

Network connectivity and coverage are widely researched topics that focus on how well the sensor field is monitored and tracked by sensors [69]. Approaches improving network coverage can be categorised into three classes [127]: area coverage, point coverage and barrier coverage. Area coverage deals with effective coverage the entire sensor field [9], [176], whereas solutions to point coverages deal with the coverage issues between a set of selected points. On the other side, solutions to barrier coverage deal with minimising the probability of undetected penetration through the barrier [130]. Most of the existing research on network connectivity and coverage is developed at the lower levels of OSI stack and they deal with the lower level complexities of networks. Gage [55] on the other hand, defines system-level functionality of network coverage as the measure of the quality of system (QoS).

One of the ways that assure the network connectivity is to apply topology control based on mathematical calculations. These calculations are based on physical characteristics and properties of the sensor field and the network components. Santi in [157] presents ways of topology control in wireless ad hoc and sensor networks for various network sizes, components and topologies. Santi specify the critical trans-

mission range of nodes based on the number of nodes deployed, size of the sensor field, and network topology. Specifying critical transmission range is one of the most important points in his work that ensures a connected with minimum radio power is used to save energy. Similarly, Poe and Schmitt [141] propose a node deployment scheme for large scale network to challenge the coverage issue.

Another common way of increasing network coverage and connectivity is to use power management mechanisms such as duty-cycling protocols where nodes are in sleeping mode most of time and only wake up asynchronously, to conserve energy and increase the life time of the network. However, the network performance can be affected negatively by the low-duty cycle of the network as redundant nodes switch between the sleep/wake cycles [103], although network has increased coverage. Duty-cycling protocols for WSNs are also well-studied and some of the examples are as follows [18], [41], [167], [183].

Alternatively, mobile devices are also used to address the connectivity, coverage, and network lifetime obstacles in WSNs. Increasing network connectivity, coverage, also benefits on performance efficiency in terms performance measures as the network lifetime increase [14]. There are some research that have been applied on Mobile ad-hoc networks (MANETs), Sensor and Actuator Networks (SANETs) and Wireless Sensor and Robot Networks (WSRNs) that particularly focus on increasing the network connectivity and coverage to increase the network performance. In any of these application domains mobile devices are incorporated into the fixed/stable nodes of the WSN architecture on their vehicles through airborne or ground.

In this thesis, we focus on using mobile devices/techniques to increase the network coverage. In the next section we provide more information on using the mobile techniques on the network coverage.

2.4.1 Mobile Techniques On The Network Coverage

Plenty of research exists on mobile entities on the network coverage (on MANETs, SANETs) that focus on localisation, effective data transmission on routing protocols, and power management protocols. We inspect some of the significant research briefly to show how different aspects of the mobile network coverage and connectivity

applied.

Effective data transmissions and routing protocols for mobile entities are one of the highly inspected aspects that increase the network coverage. Wu and Dai [192] present an efficient broadcasting algorithm that guarantees network coverage for MANETS. As MANETs are constrained by the mutual interference of concurrent transmissions between nodes, Wu and Dai focus on a protocol that guarantees the coverage of mobile nodes based on nodes local information about its neighbourhood which is updated periodically. The authors provide a robust protocol that resolves the connectivity issues of MANETs, however, their approach is suffers from the need of the GPS information used in the technique. Lowe et al. [122] focus on improving fault tolerance of embryonic algorithms in mobile networks. According to Tong et al. [173] self-healing by means of mobile nodes still remains a greatly unstudied area. One potential problem with this protocol with this is that one single mobile node is used to full routing holes. This will not solve the problem of energy depletion at static sensor nodes as single node is not sufficient to maintain energy at sensor nodes.

Localisation is also a key factor in terms of mobility. Melodia et. al [116] use integer linear programming-based solutions that constructs data-aggregation trees for Sensor-Actuator networks (SANETs), especially aims to address low energy consumption and high reliability to maintain the localisation problem for the mobile entities. The author believes ensuring reliability with minimum energy consumption is the key to path from a sensor to an actuator, and then improve this path by optimising energy while any delay is controlled.

One of the latest trends is to merge sensornet platforms with the robots as a cooperating objects and to use a distributed resource scheduling technique for managing the robot/sensor platform in the most resource effective way [102]. This is because robots are physically more capable of executing heavier workload (as opposed to sensor nodes) due to their less strict hardware constraints, it will only be beneficial to introduce the robotic agents in WSNs. In [102], robotic agents are used to increase the network coverage together with fixed sensor nodes. By Using effective task allocation applied on the robotic agents the network coverage and

performance is increased in WSRNs. Cyber-physical system (CPS) has emerged as a promising research direction to solve the communication issues due to the heterogeneity. Increasing the system performance by increasing the connectivity between the heterogeneous networks elements. So far not much research focus on combining different subclasses of embedded systems that challenges the trade-off caused by the heterogeneity of the CPS.

Summary

In this section, we discussed network coverage in WSNs, and inspected different ways of achieving it. Network coverage plays an important role in the performance of WSNs. Some of the common ways to increase network coverage focus on different aspects of WSNs such as improving the power management schemes like duty-cycling protocols or using mobile entities. We focused on using mobile entities by briefly overviewing the different use of technologies, and the heterogeneous use of the network entities on several cases in the of mobile network elements.

2.5 Summary

In this chapter we have present load balancing in WSNs, categorising it into two as system-level and MAC-level in terms of the level of the OSI stack at which the techniques are implemented in WSNs. Task mapping techniques, which are one of the most common way to achieve load balancing, are presented in Section 2.1. The analogy behind task mapping techniques is explained over the Bin Packing algorithm, which is one of the oldest techniques in the fields of computer science. The common point between the Bin Packing algorithm and task mapping techniques are underlined as resource allocation, scheduling and execution sequence which are certainly some of the key issues in WSNs apart from the improvements on the hardware technologies. Solutions to these key issues of WSNs, bio-inspired algorithms particularly bio-inspired task mapping techniques are shown.

In the last decade, bio-inspired approaches are have started to be used commonly as they mimic the robustness of the nature in variety of different aspects of WSNs.

To show the wide range of biological inspirations and the their area of applications we inspect bio-inspired load balancing techniques in Section 2.1.2. We looked for answers to the question: what sort of bio-inspired techniques can be used for effective resource allocation for large scale WSNs? We found out that there are plenty of different inspirational categories each considers different area of application and has slightly different focuses. In short, we specify the major focus of the research that have been applied on WSNs as low-level aspects of WSNs to resolve the communication problems. Improving QoS or reducing the energy consumption of any communication protocol has also been researched heavily.

We then looked at how researchers evaluate their proposed algorithms and focused on different evaluation techniques that exist in WSNs. In Section 2.2.2 we define the existing categories of performance evaluation in WSNs, and inspect the evaluation performance metrics for each of the performance evaluation category. We found that as most of the research focuses on the low-level aspects of the communication protocols, the performance evaluation metrics are also selected accordingly, that contains low-level information. We specify the performance evaluation tools mainly as real hardware components (prototyping), or low-level simulators (NSII, OPNET). As reproducibility of the research on hardware components can be difficult, commonly experimental testbeds are used in order to satisfy validity for the proposed techniques in this area. We underline the existing techniques, tools, and the metrics in this section and the absence of system-level research that can provide high-level of aspects of WSNs such as techniques for reducing computational and communicational redundancy or distributed scheduling of execution sequence of WSNs.

In the third part of this chapter, we explain some of the optimisation techniques beside task mapping optimisation. We divide this section into two and discuss traditional optimisation techniques in the first part, and focused on the metaheuristic search optimisation on the second half of this section. We concluded that metaheuristic techniques, especially Simulated Annealing metaheuristic search algorithms focus on offline optimisation for the fine tuning in WSNs, whereas traditional optimisation techniques can be used either at run-time or offline depending on the

technique itself.

In order to understand how variety of different optimisation techniques affect network coverage and how each technique affects the overall systems performance rather than particular metrics that is evaluated by, we briefly inspected the existing techniques that have an influence of network coverage in Section 2.4. Although there are plenty of techniques that involves low-level aspects of WSNs, we focused on using mobile techniques that aims to improve network performance and gave brief examples on couple of existing technologies that use mobile entities.

Most of the research techniques presented in this section improves network performance in WSNs either minor or major, whilst using low-level techniques or system-level specifications. It is clear that not much research studies focus on the system-level aspects of WSNs and there are plenty of research gaps in this aspect of WSNs.

In Chapter 3, we will explain our load balancing technique as a solution to the problems introduced that covers the gap in literature.

Chapter 3

Load Balancing Using Pheromone Signalling

The highly dynamic nature of WSN applications requires self-organised, autonomous behaviour to overcome their fundamental resource challenges. Our research argues a bio-inspired solution to distribute workload evenly over the network components and balancing node energy levels to extend the network lifetime and availability. Classical task mapping and 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 as we discuss in Chapter 2. To address those challenges, we use bio-inspired mechanism that are computationally lightweight and able to cope with a size and is the complexity of the problem. This research implements a task mapping optimisation to manage the trade-off between energy efficiency and event detection at run-time technique. We present a bee-inspired pheromone signalling mechanism to create self-organising WSNs that maximise service availability whilst reducing energy consumption. Our bee-inspired algorithm restricts service times of the network components and distributes the network workload to achieve desired high network performance. To evaluate network performance, we use service availability and energy consumption as two performance metrics. We now define the performance evaluation metrics.

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 inter-communicating tasks, therefore a service is completed only if all its tasks are executed by the WSN nodes. Therefore, a service will not be completed 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. Task mapping therefore refers to balancing the load over network nodes, i.e. deciding which node should execute the tasks of each requested service. A sound sensing network is used as a case study presented in Chapter 1. Recording and processing sounds detected 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.

Chapter Contributions The contributions of this chapter are listed below:

- The definition of a lightweight, bee-inspired algorithm to optimise load in WSNs.
- The implementation of the system-level simulation infrastructure to evaluate the long-term effects of the PS algorithm.
- The deployment of real-sensor nodes to analyse the short-term effects of the PS algorithm.

Chapter Structure Section 3.1 defines the biological background of this research, explains the underlying biological inspiration of bee colonies and explicitly establishes a link between bees/sensor nodes and bee colonies/sensor networks. Section 3.2 describes our *Pheromone Signalling-based Load Balancing* resource management algorithm. Section 3.3 explains the evaluation infrastructure in details. In Section 3.3.1 an underlying case study is defined. Section 3.3.2 illustrates the real sensor infrastructure on a TinyOS testbed, whereas system-level simulation model infrastructure is explained in Section 3.3.3. The section ends with the obtained

parameters choices that are presented in Section 3.3.4. Section 3.4 demonstrates the effectiveness of PS algorithm in comparison to several other scenarios. In Section 3.4.1 experimental setup defines the scenarios used for the experiments in this chapter. Section 3.4.2 illustrates the results of the real sensor deployment to show the short-term effects of PS algorithm, whereas Section 3.4.3 shows the system-level simulation results in order to analyse the long-term effects of PS algorithm.

3.1 Biological Background

In the previous chapter we reviewed a number of biologically inspired optimisation approaches. Many of the interactions among social animals are still unknown to mankind and are considered as innate knowledge and we avoid using an inspiration that we cannot explain scientifically. Bees are one of the most studied social animals and many of their behaviours, such as pollen collection, mating, and foreign behaviours have previously been applied to WSNs, as explained in Chapter 2. After examining many different aspects of bees, we recognised that there are obvious parallels between their pheromone-based communication mechanism and load balancing. This communication mechanism is used by bees to orchestrate a bee colony by assigning responsibilities to individual bees in order to produce an effective bee colony; this is, in essence, load balancing. This process has been well studied and scientifically proven, unlike some other natural processes, giving our research a solid foundation to build upon. In this thesis, therefore, we investigate whether an algorithm inspired by the pheromone-based communication mechanism used by bees can effectively distribute work over a network.

Research in social insects [151] has uncovered innate 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 [151] 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 research, namely the process of larvae differen-

tiation in beehives [151]. 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 queen. This phenomenal mechanism works as follows: the worker bees lick the queen bee and pass the pheromone to their surrounding neighbours and pheromone substance will move gradually through neighbours by licking each other. 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 [151]. 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.

Table 3.1: Correlation between bee's pheromone stimulation and sensor networks

Bees/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 load balancing technique presented in this chapter takes inspiration from the behaviour of bees. Table 3.1 describes the analogy we make between bee colonies and sensor networks. In this research, we consider that our network consists of more than one bee colony, so that there can be many queen bees in the network at any given time. Accordingly, the role of a queen bee denotes a sensor node that is responsible for managing the execution of all service requests it receives. Throughout this chapter we will refer to such a node as a Queen Node (QN). They are dynamically differentiated from other nodes to indicate their duties; this behaviour is conceptual and does not make specific assumptions about the capabilities of the

queen node meaning that it can be applied to both homogeneous and heterogeneous platforms. Any nodes not differentiated into QNs are referred to as Worker Nodes (WNs). All QNs and WNs are capable of sensing the environment, queuing tasks, and communicating with other nodes within range. However, in our approach only QNs will execute tasks based only on their local information (i.e. react to a sensed event or a service request). By allowing many nodes to become queen nodes, we are effectively populating the network with multiple bee colonies - where each colony is responsible for processing work in that area of the network. Multiple bee colonies are similar to dynamic clusters where cluster-heads and members change on a periodic basis. This cluster-like concept allows PS to increase the coverage of the network.

3.2 Load Balancing Using Pheromone Signalling

In this section we describe our load balancing algorithm. As we argue the need of using the dynamic, decentralised, and computationally lightweight algorithm to distribute the network load onto the network elements in the beginning of this chapter, the *Pheromone Signalling* algorithm (PS) 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 from the transmitting pheromone source. All nodes accumulate of each 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 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).

Listing 3.1: PS Differentiation Cycle

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

The first time-triggered part, referred to as the differentiation cycle (Listing 3.1), is executed by every node of the network every T_{QN} time units. On each execution, the node checks its current pheromone level p_i against a predefined level $threshold_{QN}$. The node will differentiate itself into QN (or maintain its QN status) if $p_i < threshold_{QN}$, 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 *pd* 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 3.1). The second element is the actual dosage of the pheromone that will be absorbed by the neighbours.

Listing 3.2: PS Propagation Cycle

```
1 when hd is received
2   if (pd[1] < threshold_hopcount)
3     p_i = p_i + pd[2]
4     broadcast pd = {pd.distance[0] + 1, pd.distance[2] * K_HOPDECAY}
5   else
6     drop hd
```

The event-triggered part of PS deals with the propagation of the pheromone released by QNs (as described above 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 pseudo-code appears in Listing 3.2. 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 *pd* with a predefined $threshold_{hopcount}$. If the *pd* 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 *pd* vector element by incrementing the hop count, and by multiplying the dosage by a decay factor $0 < K_{HOPDECAY} < 1$. This represents pheromone transmission decaying with distance from the source.

Listing 3.3: PS Decay Cycle

1 every T_{DECAY} do $h_i = h_i * K_{TIMEDECAY}$

The second time-triggered part of the algorithm, shown in Listing 3.3 is a simple periodic decay of the pheromone level of each node. Every T_{DECAY} time units, p_i is multiplied by a decay factor $0 < K_{TIMEDECAY} < 1$.

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 p_i. This allows for a highly self-organised behaviour which fits the requirements for high-density networked embedded systems. The computational complexity of the algorithm is very low, as each of the parts is a short sequence of simple algorithmic 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 decay cycle time 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.

All three cycles of the PS are shown in Figure 3.1, where circles represent the nodes and the written values inside the circles represent the pheromone level of each node. Nodes shaded green indicate queen nodes. Arrows represent the propagation of pheromone where thick arrows represented high level of pheromone propagation, thin arrows represent low level of pheromone propagation. Figure 3.1 illustrates the QN differentiation in iteration 1.1, and it continues with QN pheromones propagation and WN pheromone retransmission a lower dose of pheromone to their neighbours in iteration 1.2. QN propagates 60 and 15 amount of pheromone to first hop

and second hop neighbours respectively. The dramatic change in the pheromone numbers represented in iteration 1.2 is as a result of summed up pheromone levels. Pheromone decay is demonstrated in iteration 1.3 to indicate the elapsed time and the pheromone decay within time. Iteration 2.1 shows the pheromone levels for each node after all the cycles of PS is completed. Although we make the analogies with bee colonies, the names of the nodes are misnomer. We make it more clear the unfortunate naming of queen and worker nodes as such the queen nodes respond to

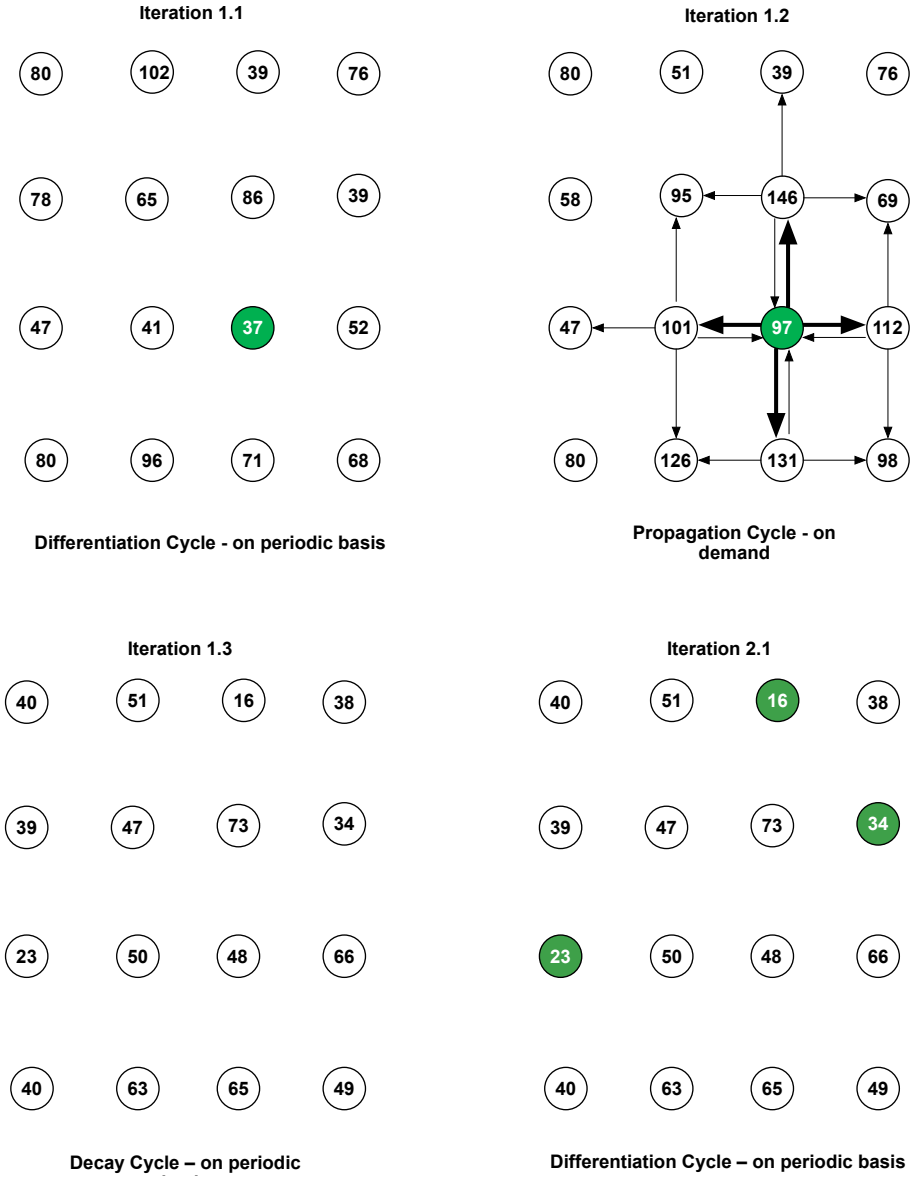


Figure 3.1: The three important stages of the PS algorithm: differentiation, propagation and decay cycles.

these periodic detections by transmitting an event notification, while worker nodes ignore these application events.

Summary

In this section we have presented a bio-inspired algorithm to address the trade-off between the service availability and energy consumption of WSNs. We took inspiration from bees and developed an algorithm that mimics the pheromone signalling process in bee colonies. Our aim was to balance the application load using a lightweight, robust protocol whilst achieving the highest network performance with minimum redundancy. In the rest of this chapter, we explain the evaluation infrastructure and illustrate the experimental results of the *PS* technique.

3.3 Evaluation Infrastructure

In the previous section, we explained *Pheromone Signalling-Based Load Balancing* technique. In this section, we explain the evaluation infrastructures that *PS* is applied on. In Section 3.3.1 we define the case study that we evaluate *PS* with throughout this thesis. Section 3.3.2 describes an *TinyOS*-based experimental testbed for evaluating PS on real sensor deployments. In Section 3.3.3 we describe a system-level simulator we have implemented. Configuration parameters related to the evaluation infrastructures are presented in Section 3.3.4.

3.3.1 Case Study

A case study based on a surveillance multimedia target tracking application is used to evaluate the effectiveness of the PS approach. The goal is to compare the load balancing solution against existing solutions, and then observe the impact of using different values for the parameters of the 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 [117] as we explained in Chapter 1 with Figure 1.1. Once a node captures a sound, it processes it aiming to identify its source and report its approximate location.

3.3.2 TinyOS Experimental Testbed

The experimental testbed (see Figure 3.2) is intended to evaluate the short-term behaviour of the protocol. It consists of 16 homogeneous nodes (MEMSIC Iris nodes with 2.4GHz transceivers [117]) together with a base station that serves to receive results and transfer them via USB to a monitoring computer.

Figure 3.2: Iris nodes used in the 4x4 grid hardware deployment

These nodes run on the open-source TinyOS operating system version 2.1 [100]. A custom modular application was developed to perform multi-hop 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 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 Figure 3.3. 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

and illustrated with arrows in the figure. Undirected links between nodes represent possible communication links that are not totally separated from the routing paths.

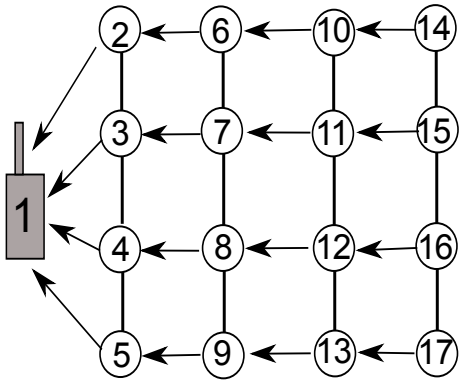


Figure 3.3: The network topology with preconfigured multihop routing chains

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 [117], 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 3.2, 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, 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. In order to explore *PS* algorithm and it's real-life effects we conduct some experiment on deployed sensors that will represent

in Section 3.4.2. By applying *PS* on the real sensor deployment we analysed the short term effects of the *PS* and compared them with the long term aspects of the algorithm on the system-level simulation infrastructure.

3.3.3 System-Level Simulation Infrastructure

The objective of evaluating *PS* 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 PS technique on large scale deployments. Moreover, the short implementation duration and the cost effective nature of system-level simulators makes them suitable for this research.

A three-tier WSN system model is designed to represent network components, the services that run over it and the function that assigns services to network nodes. Graph theory can be used to model a network and its services. The platform model, $PM = (N, L)$ consists of a set of N nodes and a set L of bidirectional wireless links between neighbouring nodes.

Each node $n_m \epsilon$ N, is the tuple: $n_m =< m_m, bc_m, idr_m, cdr_m, wcdr_m >$

m_m memory capacity in bytes;

bc_m battery capacity in mAh;

idr_m battery discharge rate in μ as in idle mode;

cdr_m battery discharge rate in μ as when performing a computation;

$wcdr_m$ discharge rate in μ as 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 set of links $l_{mn} \epsilon$ L, the cost of multi-hop transmission of each communication c_{ij} can also be taken into account if the routing algorithm used in the network is known.

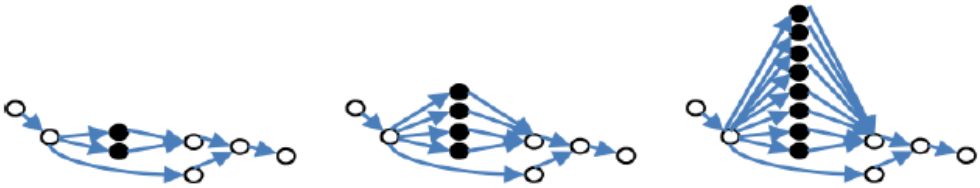


Figure 3.4: Services composed of (a) 8 tasks, (b) 10 tasks, (c) 14 tasks. *White shaded circles indicate the skeleton of the DAG whereas black shaded circles are used to populate the tasks of the DAG.

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 research, we focus on complex services that are composed of multiple tasks. We represent a service S as a directed acyclic graph (DAG) as Figure 3.4 illustrates, with nodes representing tasks and edges representing inter-task communication: $S = (T, C)$. Each task $t_i \epsilon$ T, is a tuple: $t_i =< n_i, mf_i, e_i, et_i >$,

n_i supplier node;

mf_i memory footprint in bytes;

e_i energy consumption;

et_i execution time;

Each inter-task communication $c_j \epsilon$ C , is also a tuple $c_j =< s_j, r_j >$, where $s_j \epsilon$ T is the sender task and $r_j \epsilon$ T is the receiver task of the communication. For

this framework, the mapping process is defined as a function from the application domain to the platform co-domain and is represented as F:T→ N.

To contextualise the services, we refer back to the wild life bird monitoring case study. Tasks of each service indicate the pitch of the bird, and any sensor nodes located in the same neighbourhood hear the same bird with same delay due to the delay in the sound propagation. This is represented as follows: among the sensor nodes that detect the same service get assign a DAG (containing 8, 10 or 14 tasks as Figure 3.4 illustrates) of the same service. The sensor node where the event occurs is assigned a DAG with 14 tasks, whereas other nearby nodes that also detect the same event receives DAGs with fewer tasks to simulate the delay in the sound propagation.

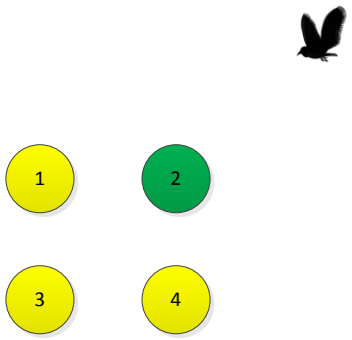


Figure 3.5: Worked example on the event generation used for simulation.

Figure 3.5 illustrates a bird and four nodes labelled from one to four. Node 2, which is coloured in green is a QN, whereas nodes 1, 3 and 4 are WNs. A bird in this worked example lands near sensor number 2 and performs a short birdsong. In such a scenario, the event generation used in our simulation runs as follows:

1. *Real Life:* Sensor number 2 detects the birdsong before the other sensors as initially the birdsong only occurs in the detection range of sensor number 2.
   Simulation: Sensor number 2 starts executing tasks related the detected bird as the node is a QN. The DAG assigned for this node will be populated like Figure 3.4 (c).

2. *Real Life:* Sensors 1, 3 and 4 detects the bird shortly after sensor number 1 as the takes longer to reach them. They detect a smaller amount of the bird

pitch in comparison to sensor number 2 due to their increased distance from the bird.

Simulation: Sensors 1, 3 and 4 start detecting the sound when it appears in their detection range, however, these sensor nodes are assigned fewer tasks in comparison to the sensor number 1 (as in Figure 3.4 (a) or (b), dependent on the distance). This is to simulate the direction of the birdsong and to show which node detects the bird first. As these nodes are WNs, they do not execute the tasks that they have detected and they drop all their tasks.

In this section, we present the system-level simulation infrastructure that PS has been implemented on and thus have only described how the application model and platform model are simulated. An event-driven simulator has been designed to implement both of these models. Figure 3.6 presents a UML sequence diagram illustrating the execution of the simulator. It is controlled by the JavaSim library [109]. The sensor network platform is created once in the beginning of the simulation and each sensor node is created by the generatePlatform() function. The event generator starts executing once the platform has been initialised and periodically produces events in the network. The event generator generates one event at a time,

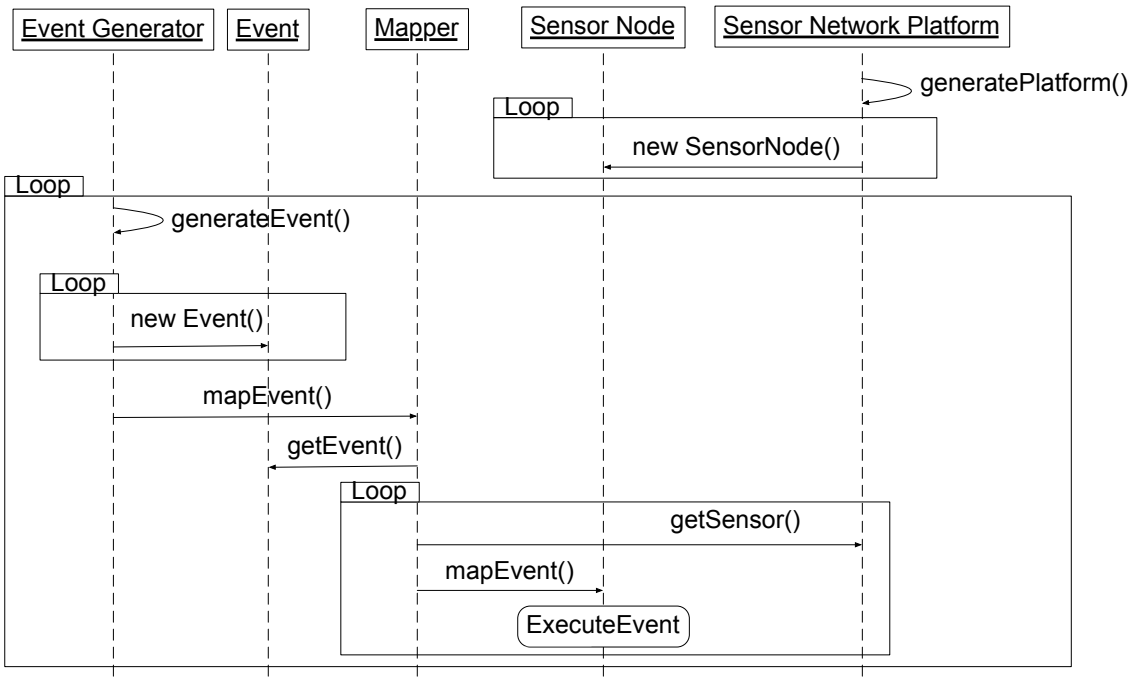


Figure 3.6: Execution sequence of system-level simulator.

at a location randomly selected in the sensor field. Each event consists of a number of DAGs as explained previously in Figure 3.5. The mapper maps the event onto the sensor nodes by assigning of the tasks of each DAG onto nearby sensor nodes within a predefined range. Once an event has been assigned to a sensor node, the PS algorithm decides whether or not that node will execute the event. The simulator is sufficiently scalable as to be able to simulate hundreds of nodes, and some of the results have been obtained on a large sparse topology and grid topology of 28x28 nodes.

3.3.4 Parameter Choices

The parameters used for the evaluation infrastructures are distinct from the parameters of the *PS* algorithm. In this section, we focus on describing the parameters used for the evaluation infrastructures. As we used MEMSIC Iris nodes for the real sensor deployment, we also use the MEMSIC Iris datasheets to configure the parameters for our system-level simulator. This allows us to analyse the validity of the simulation results by comparing them with the real hardware. Some of the several important energy parameters of our experimental platform are shown in Table 3.2.

Table 3.2: Energy related parameters

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

3.4 Experimental Results

In this research, we validate our bio-inspired load balancing technique via system-level simulation models and a small hardware prototype test-bed. In Section 2.2, we define, explain and compare system-level simulators, low-level simulators and prototypes. 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 gain insight on the implementation duration, performance efficiency and the level of accuracy of the system level model versus the prototype, as well as demonstrating the effectiveness of the load balancing technique. Moreover, by evaluating *load balancing using pheromone signalling* algorithm on real sensor nodes we demonstrate the behaviour of the system on a small scale, and exhibiting performance advantages of PS on a real sensor deployment on TinyOS operating system. Evaluating the system-level simulator helps us to analyse the long-term effects upon performance of PS in particular on a large scale where *PS* is more beneficial. We now define the experimental setup in detail to explain the objective of the applied scenarios, used approaches and their results.

3.4.1 Experimental Setup

We have created four other scenarios to compare with *PS* as follows:

Idle Represents the absence of load, and all nodes of the system do not dissipate any energy on computation or communication with the neighbours.

Objective It is included as the maximum lifetime of the system if no surveillance is performed.

Baseline Represents the execution of the case study without any load balancing support.

Objective It represents an approach that allows high amount of redundant computation as it does not apply any load balancing.

BS Represents the execution of the case study using load balance based on existing energy-aware task migration mechanisms as in [136] and [144].

Objective It includes a basic self-organising character that involves task migration to show the impact of it.

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

Objective It is designed to show maximum possible network performance by creating a scenario which never can occur in real life. The role of load balancing is to have no redundancy, but this is unlikely. It is almost impossible to have a perfect knowledge of where the event is occurring so closest we can get in real life is to achieve minimum redundancy. However, Optimal scenario assumes that nodes have the perfect knowledge on where events are occurring and as a result it provides maximum possible performance without any redundancy.

Probabilistic Represents a simple random policy for WSNs where nodes execute events with a given probability.

Objective It is created to ensure that *PS* provides better performance than a random probabilistic selection.

PS Represents the execution of the case study using *Pheromone Signalling Load Balancing* algorithm.

Objective It is developed to distribute network load to solve the service availability and energy consumption using computationally lightweight bio-inspired approach.

3.4.2 TinyOS Hardware Testbed Experiment Results

Note: The work described in this section is joint work and developed in collaboration with James Harbin, Leandro Soares Indrusiak, Paul Mitchell, David Chesmore and Fiona Polack, and has been published in [21, 24]. Harbin was responsible for the hardware implementation and I was responsible for definition of *PS* and implementation of it on the simulation infrastructure.

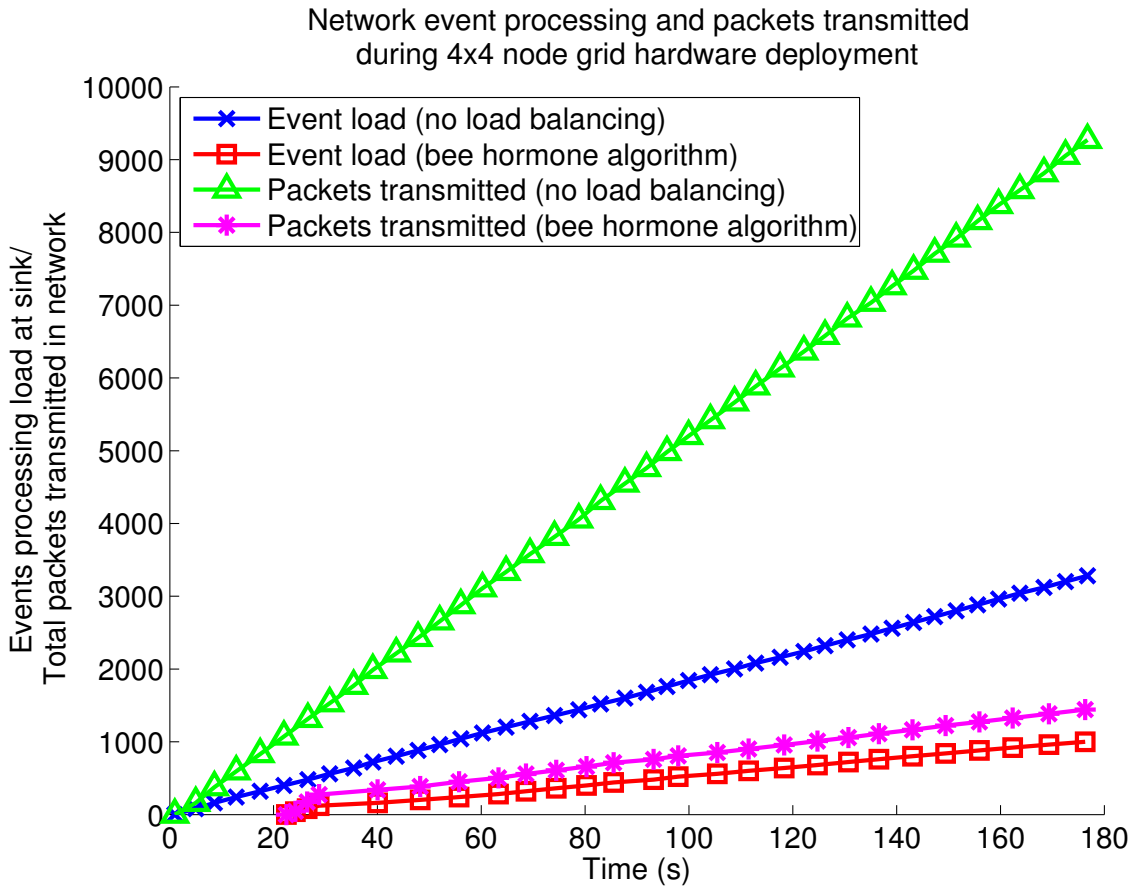


Figure 3.7: Event detections and packets for load-balancing pheromone algorithm and baseline

This set of experiments only use two scenarios: *Baseline* and *PS*, to gain some insight about non load balanced approach versus load balancing using pheromone signalling due to the high cost of implementing all on hardware.

The results of our experiments are shown in Figure 3.7 illustrates 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 (non-zero) number of event detections 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

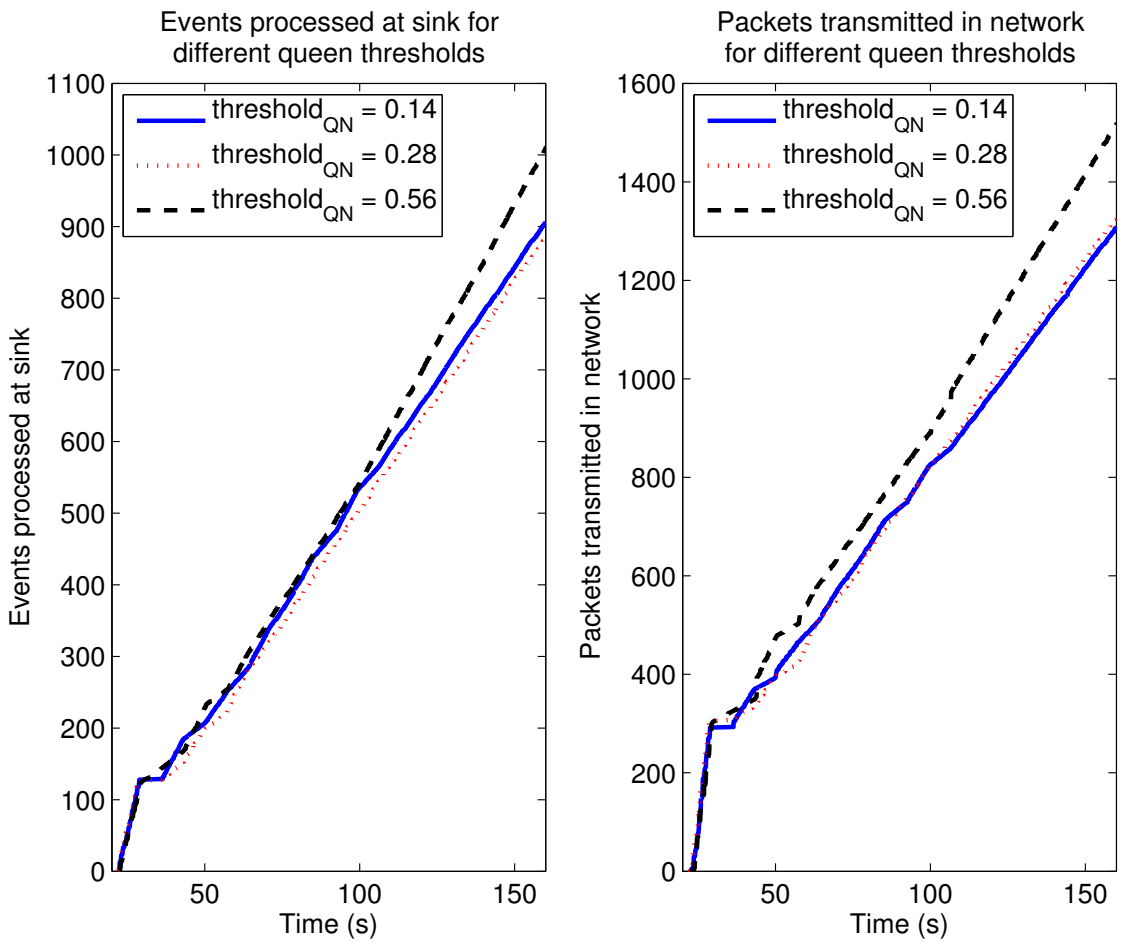


Figure 3.8: Event detections and packets for different queen thresholds

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.

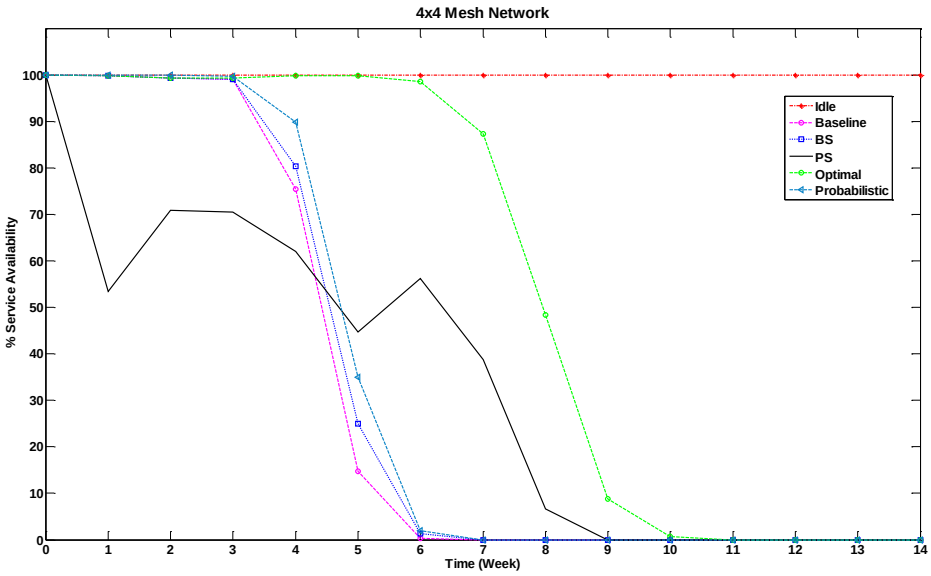
Figure 3.8 shows the impact of the queen pheromone threshold $threshold_{QN}$ upon the measured event processing and packet transmission load. These series are not greatly influenced by doubling the queen threshold from 0.14 to 0.28. However, if the queen threshold is doubled again to 0.56, then the 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 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.

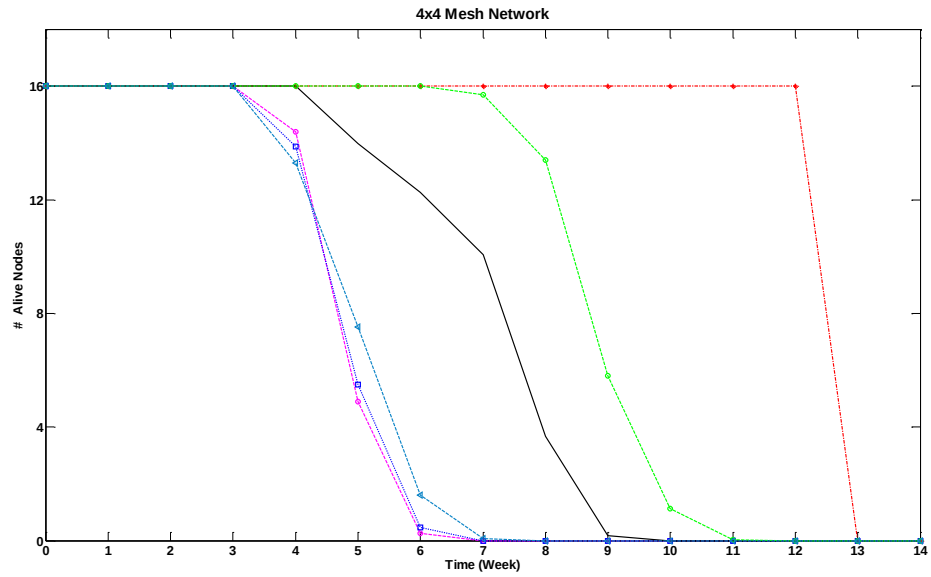
3.4.3 System-Level Simulation Results

The experimental work presented in this section aims to compare the *PS* technique against five scenarios: *Idle, Baseline, BS, Optimal* and *Probabilistic*. We also analyse effects of different values of execution probabilities for the *Probabilistic* scenario. Moreover, we analyse the effects of T_{DECAY} and $threshold_{QN}$ using a variety of different values for these parameters for the *PS* technique. We also explore the QN distribution on the network layout using heatmaps to show the uniform use of network resources. The comparison in each category of experimental results are based on 30 different soundscape scenarios running over the given network configurations (4x4, 7x7 and 28x28 topologies) to ensure statistical significance. Each run simulates the case study for 14 weeks, which is the point at which all network nodes run out of energy even in the *Idle* scenario.

Comparison against other scenarios Figure 3.9 (a) and (b), shows the percentages of detected events and the number of alive nodes. In x-axis values presented in the figure illustrate the results that are captured by the end of each week (this applies to all the other figures represented in this chapter). According to Figure 3.9 (a) and (b), the percentages of detected events and the number of alive nodes are lowest in the *Baseline* scenario. In *Baseline* and *BS* scenarios all the nodes are allowed to execute events unlike *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 a nodes energy is lower than a predefined threshold. In this case, nodes apply task migration and as a result the number of alive nodes is increased, as well as the percentage of detected events as opposed to *Baseline*. In the *Probabilistic* scenario, nodes choose to ignore detected events with a given probability, otherwise they execute them. This allows a node to conserve energy. The *Baseline* scenario, from the probabilistic perspective, is equivalent to the



(a)

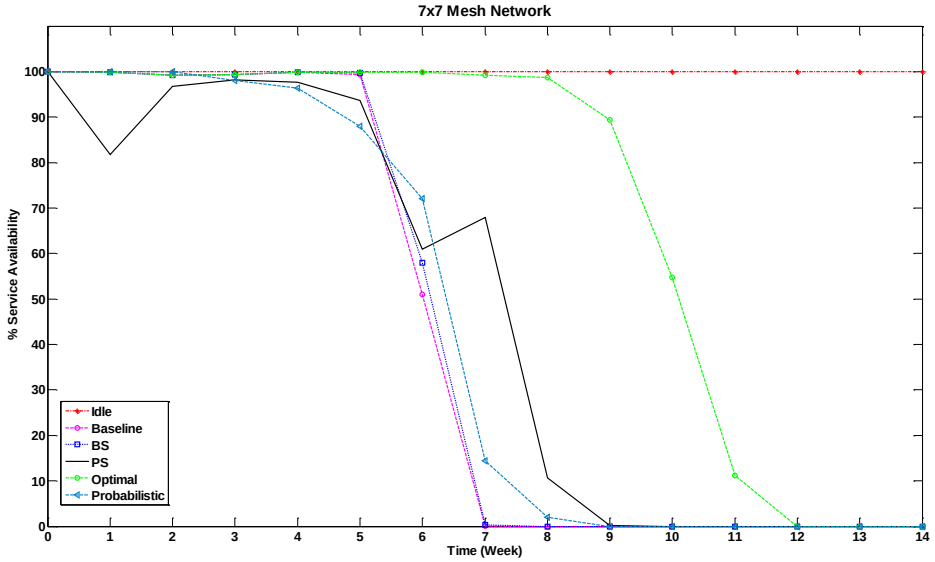


(b)

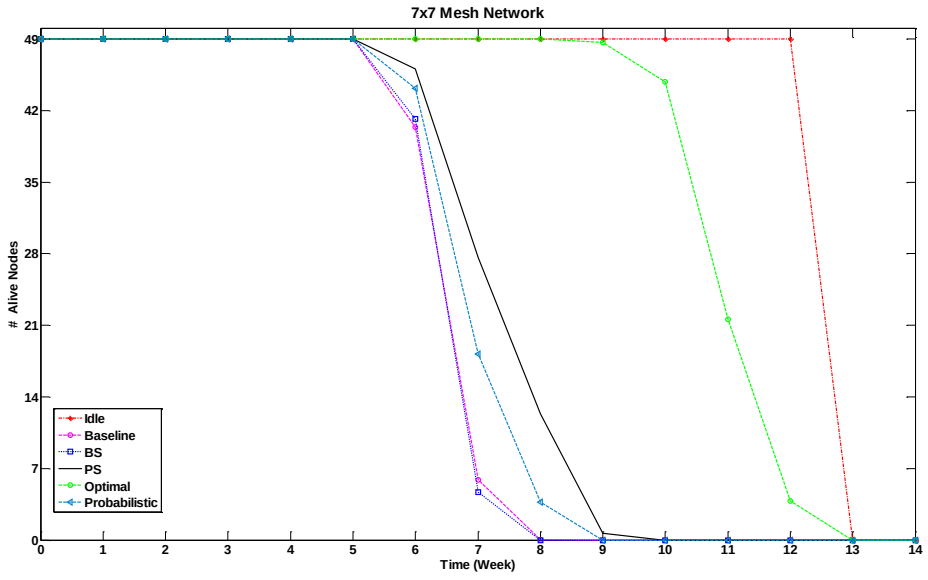
Figure 3.9: Experimental results: effects of PS algorithm on 4x4 mesh network topologies showing (a) % service availability, (b) # alive nodes.

probabilistic scenario with the probability of ignore events set to 0. In Figure 3.9, the results illustrate the Probabilistic scenario with a probability of 0.8 of ignoring detected events (in the next section we investigate different probabilities; 0.8 scored the highest). The results show that the *Probabilistic* scenario saves more energy and

improves the network lifetime whilst increasing service availability in comparison to *Baseline* and *BS*. The major improvement, however, is shown by the *PS* scenario. The number of alive nodes remains constant until the fourth week in Figure 3.9 (b), however the percentage of the detected events dramatically drop in Figure 3.9 (a) during the first week. This occurs due to the pheromone stabilisation over the



(a)



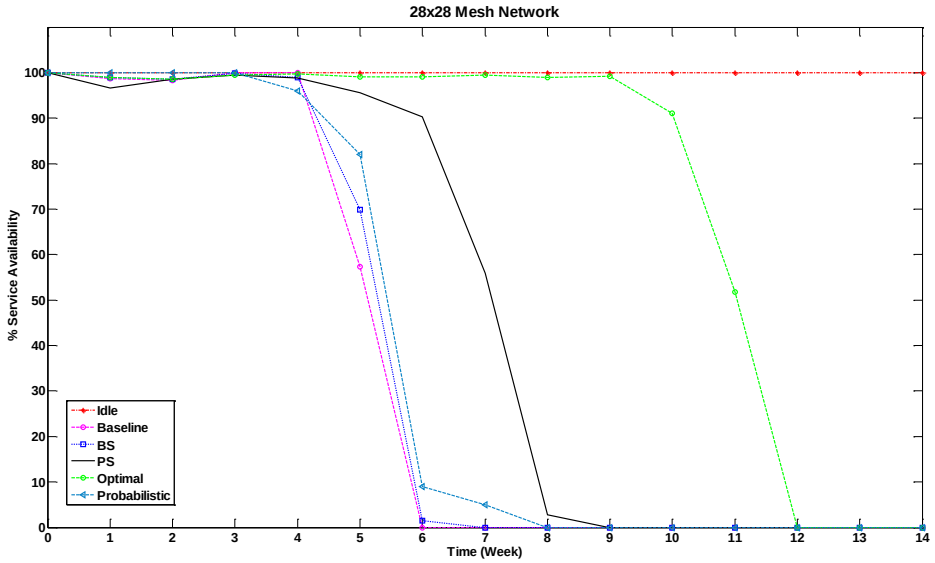
(b)

Figure 3.10: Experimental results: effects of PS algorithm on 7x7 mesh network topologies showing (a) % service availability, (b) # alive nodes.

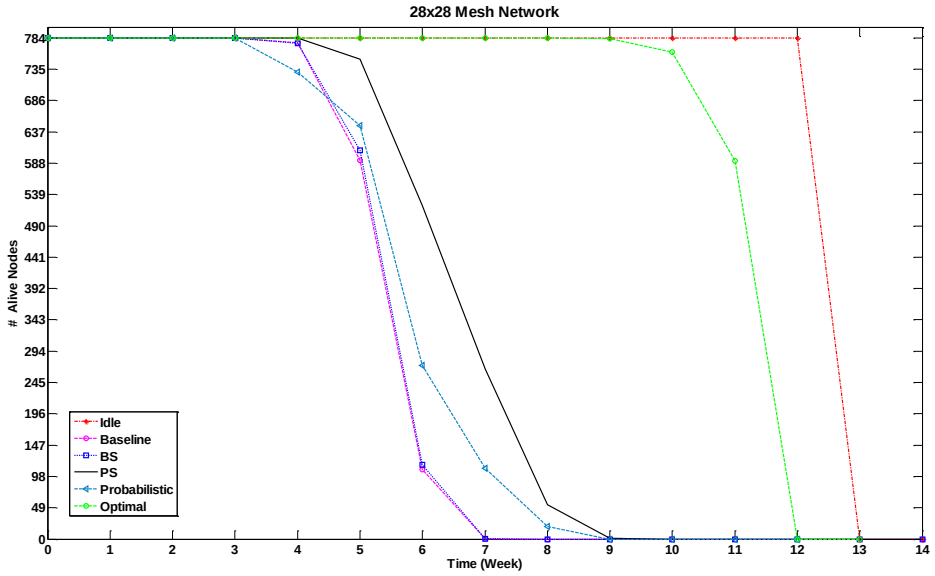
network. As *PS* limits the redundant network progress by allowing pheromone/QN procedure, a significant amount of the percentage of the detected events occurs due to the pheromone stabilisation. In small networks like 4x4, pheromone stabilisation over the network takes more time due to the lower number of redundant network resources. The positive effects of pheromone propagation on large scale networks are shown in the rest of this section as follows:

Figure 3.10 illustrates the experimental results on 7x7 mesh topology. Although there is a quite noticeable drop in the percentage of service availability during the first week in Figure 3.10 (a) for *PS* technique, network stabilisation occurs faster than the 4x4 mesh network. This is due to the higher number of the network resources on 7x7 network that allows pheromone propagation to occur more smoothly. As the network stabilises faster, the percentage of service availability remains high in the second, third and fourth weeks. After the fourth week, the nodes are starting to run out of energy in *PS* as Figure 3.10 (b) shows and by the end of fourth week the percentage service availability starts decreasing. In *PS* technique nodes live longer than the *Baseline*, *BS* and *Probabilistic* scenarios accordingly as a result of uniform distribution of network loads over the network nodes. The *Probabilistic* scenario with 0.8 probability of staying idle performs better than *Baseline* and *BS*, however, *PS* is still performs remarkably better than *Probabilistic* scenarios.

Similarly, Figure 3.11 also illustrates (a) the percentage of service availability, (b) number of alive nodes for a 28x28 mesh network topology. As with Figure 3.9 and Figure 3.10 the *Baseline* scenario achieves the lowest detected event percentage among all the scenarios, whereas *BS* improve the network performance in terms of service availability and network lifetime slightly. The *Probabilistic* scenario saves more energy than both the *Baseline* and *BS* scenarios and increases the service availability whilst achieving longer network lifetime. There are two important and related reasons to explain why *PS* technique outperforms the other scenarios. First, and the most importantly, in 28x28 mesh network the number of redundant network resources are the highest among other network topologies. This allows pheromone propagation and QN selection easily and more successfully although the nodes are strictly limited with the $Threshold_{QN}$ just like small networks. Evenly distributing



(a)



(b)

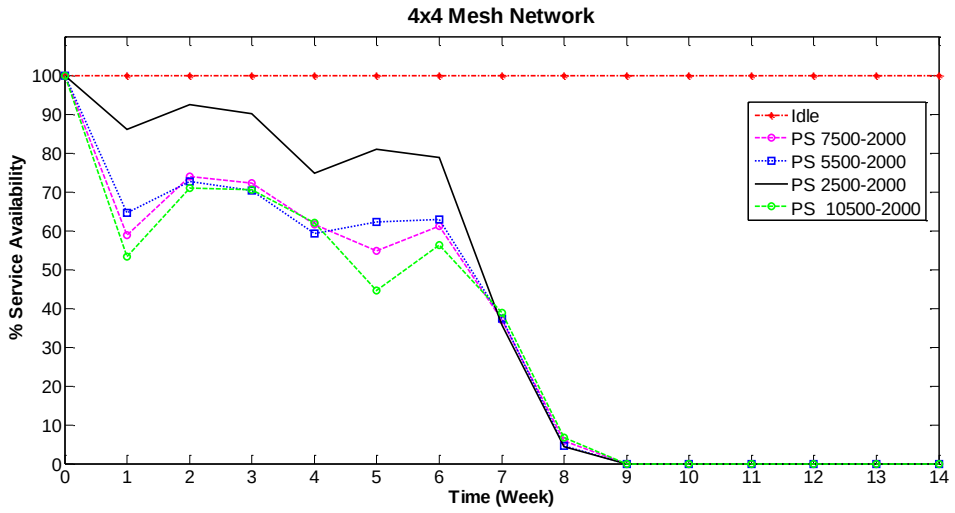
Figure 3.11: Experimental results: effects of PS algorithm on 28x28 mesh network topologies showing (a) % service Availability, (b) # alive nodes.

the network load over time is important in *PS* technique, however, the sequence of the distribution is not important. As a result, the more redundant network resources exist in the network, more the pheromone propagation occurs successfully.

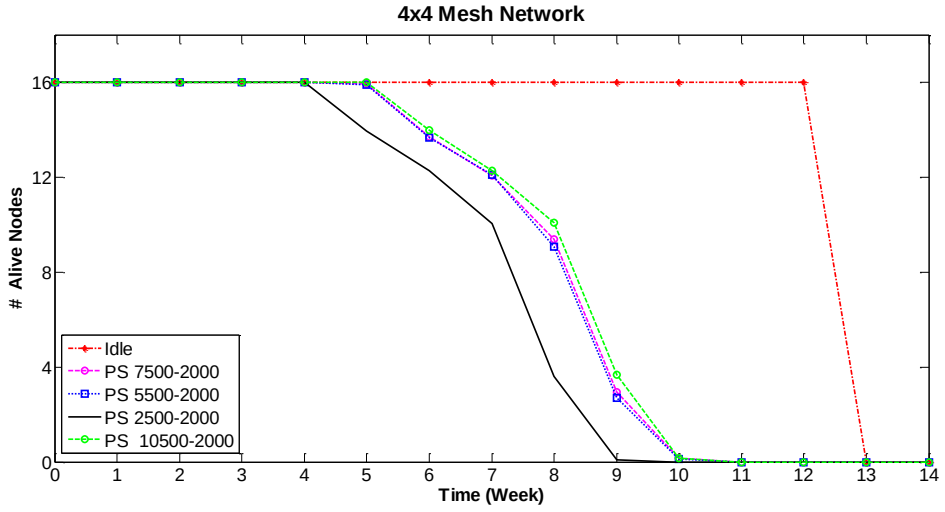
In summary, we have compared four different approaches to load balancing different size mesh networks - *PS*, *BS*, *Baseline*, and *Probabilistic*. *PS* has been shown

to be the most effective at increasing network lifetime whilst maintaining good service availability. The *Probabilistic* scenario presented in this section used 0.8 as the probability for a node to ignore detected events. A short investigation looking to the effects of different probabilities at the end of this section, and shows that 0.8 was the best of those analysed.

Analysing the effects of T_{DECAY} Sensitivity analysis is partially applied to the

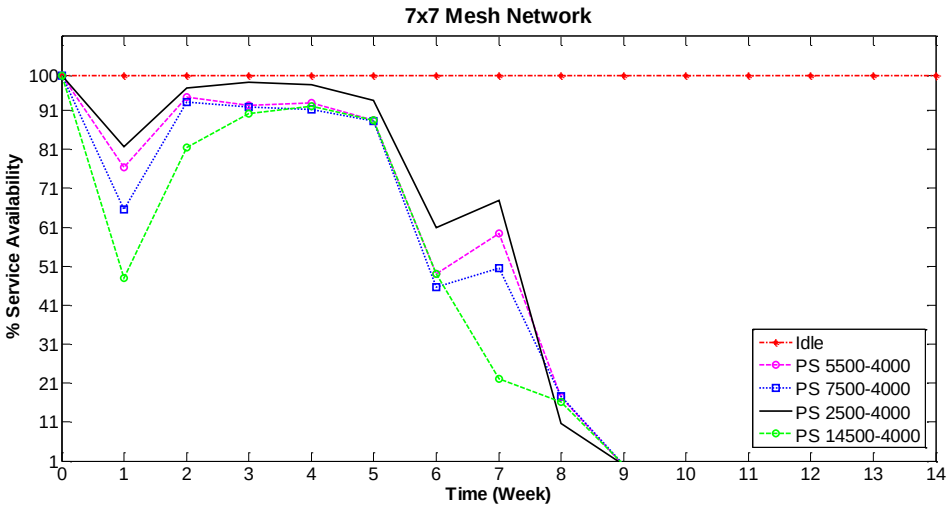


(a)

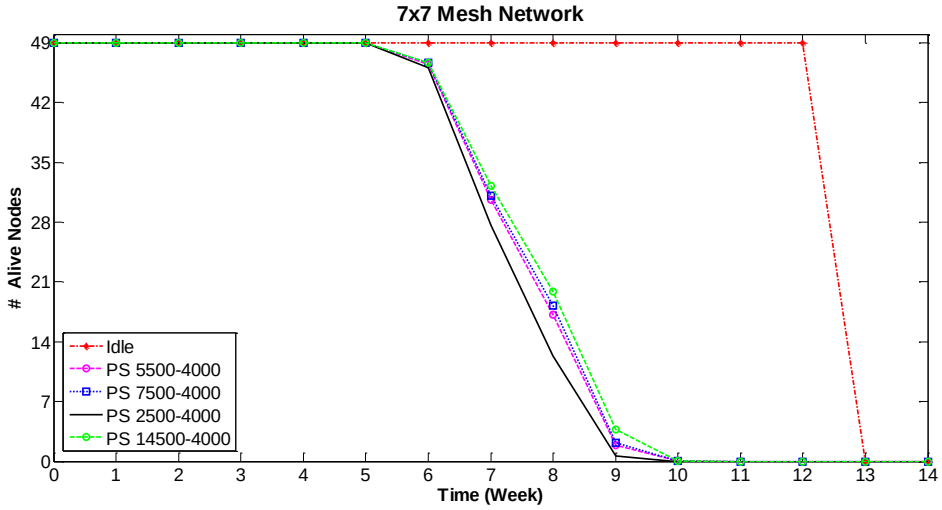


(b)

Figure 3.12: Experimental results: (a) % service availability, (b) # alive nodes for PS load balancing with different T_{DECAY} on 4x4 mesh network topology.



(a)



(b)

Figure 3.13: Experimental results: (a) % service availability, (b) # alive nodes for PS load balancing with different T_{DECAY} on 7x7 mesh network topology.

PS technique to show the impacts of the *PS* parameters as mentioned in Section 3.3.4. A variety of pheromone decay intervals are evaluated on our *PS* technique to show the effects of the parameters on the percentage of service availability and alive nodes on 4x4, 7x7 and 28x28 mesh network topologies. Figure 3.12 (a), Figure 3.13 (a) and 3.14 (a) shows the experimental results of the percentage service availability on 4x4, 7x7 and 28x28 with variety of T_{DECAY} values. Based on our understanding of the biological background, and how the technique reflects on implementation,

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 increases as well, whereas longer T_{DECAY} time parameters allow lengthening of the network lifetime.

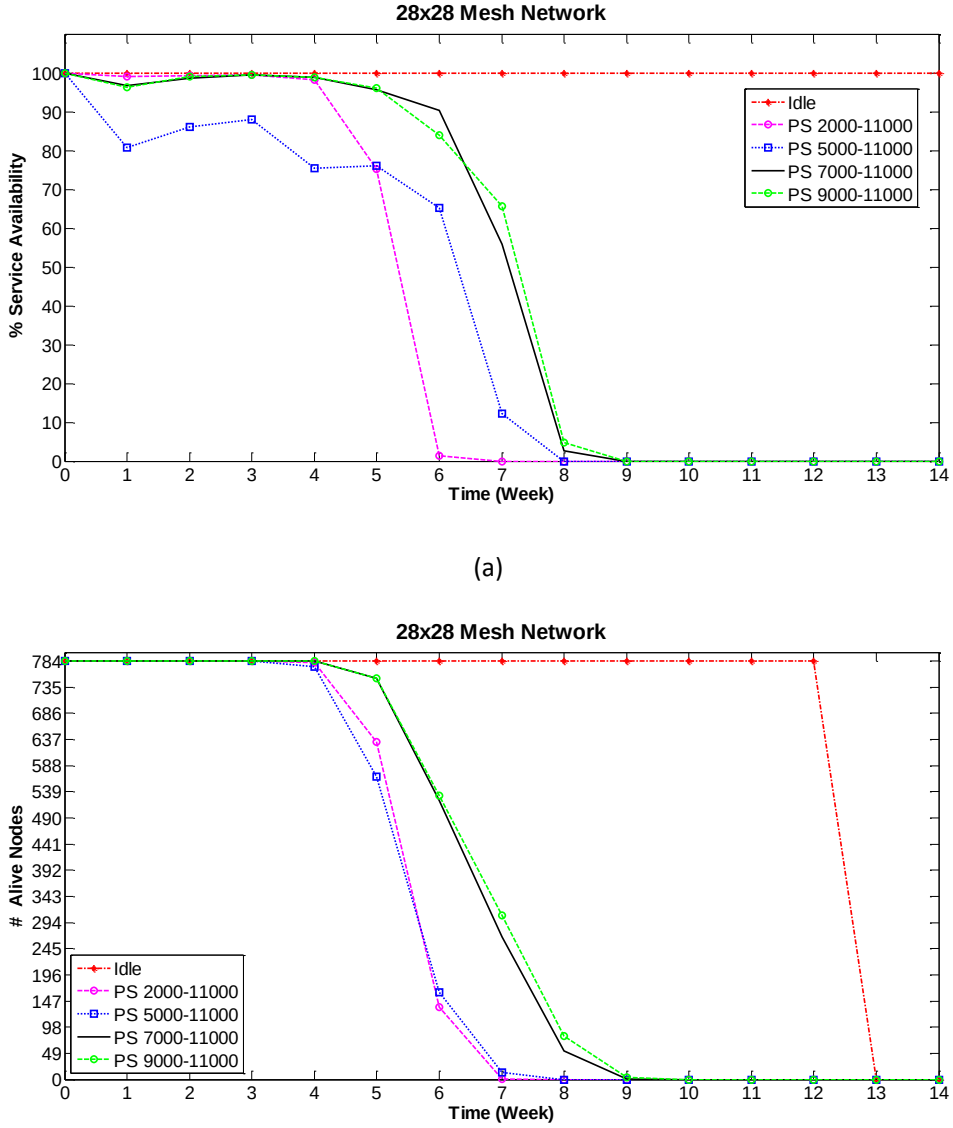


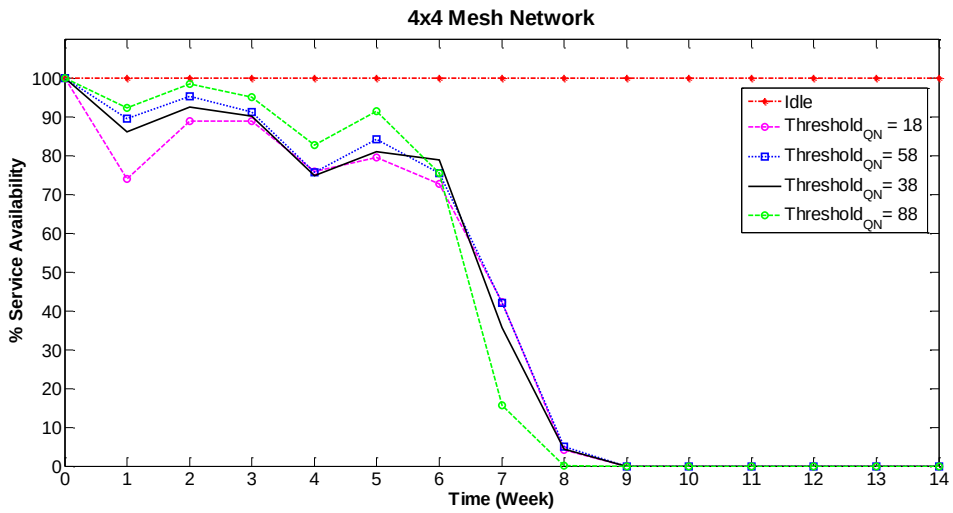
Figure 3.14: Experimental results: (a) % service availability, (b) # alive nodes for PS load balancing with different T_{DECAY} on 28x28 mesh network topology.

On the other hand Figure 3.12 (b), Figure 3.13 (b) and 3.14 (b) illustrates the number of alive nodes (i.e. with non-zero battery levels) over time for each

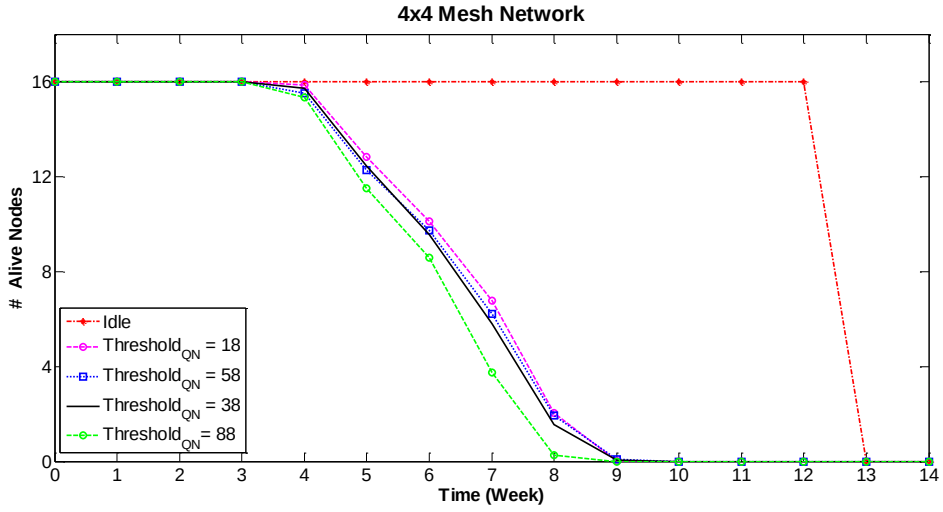
network configuration on 4x4, 7x7 and 28x28 respectively. While there is an apparent correlation between these curves and the service availability curves discussed before, this is not necessarily the case. For instance, if events are concentrated in a particular area of the network, nodes in that area 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 *PS* approach extended network lifetime, mean lifespan for every sensor node on average, by at least a week in both cases.

Analysing the effects of $Threshold_{QN}$ $Threshold_{QN}$ is another important parameter in *PS* that is used to control the QN selection. We explored all three network topologies and the effects of the $Threshold_{QN}$ on the percentage of service availability and the network lifetime. Figure 3.15, Figure 3.16 and 3.17 shows scenarios 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 in the *PS* differentiation cycle, and determines whether a given node should change into (or stay as) a 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 pheromone 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. The number of alive nodes presented in Figure 3.15 (b), Figure 3.16 (b) and 3.17 (b) change based on the $Threshold_{QN}$ too. There is not a dramatic difference in terms of network lifetime in any of the network topologies as the figures illustrate. From this, we can conclude that $Threshold_{QN}$ values has an higher impact on service availability rather than the network lifetime.

Accordingly, Figure 3.15 (a), Figure 3.16 (a) and 3.17 (a) presents the service availability for each network configuration on 4x4, 7x7 and 28x28 mesh network respectively. Depending on the network size and the value applied for the



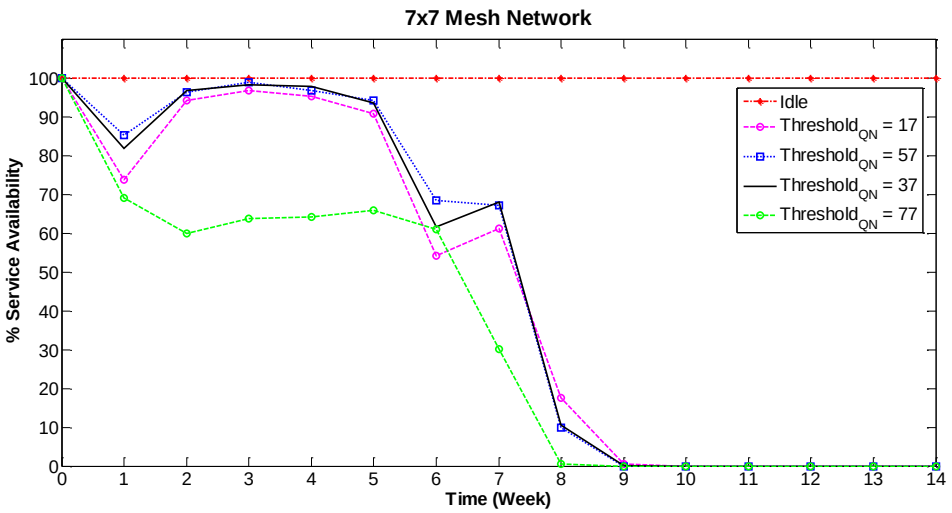
(a)



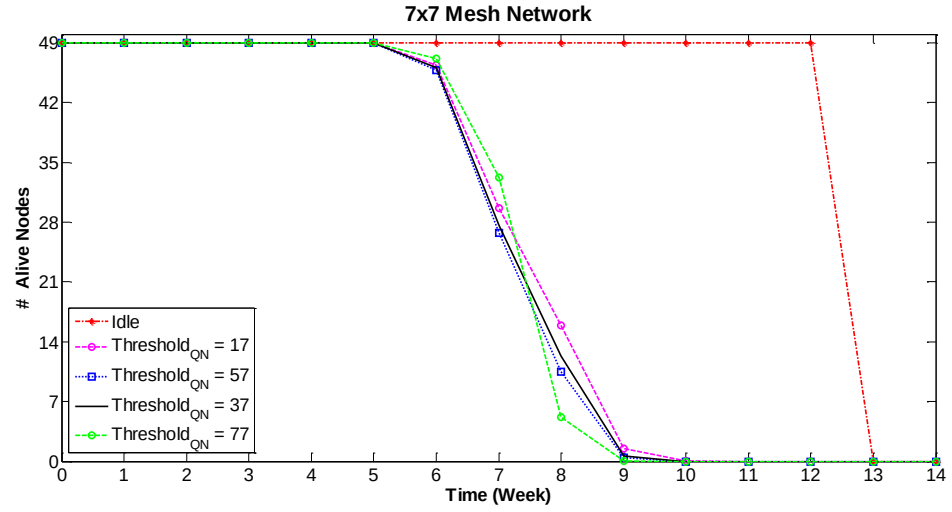
(b)

Figure 3.15: Experimental results: (a) % service Availability, (b) # alive nodes for PS load balancing with different $Threshold_{QN}$ on 4x4 mesh network topology.

$Threshold_{QN}$, the performance of the *PS* changes. In all the cases, $Threshold_{QN}$ affects the pheromone stabilisation period over network changes, and this effects the percentage of service availability during the first week. Only in the 28x28 network, pheromone stabilisation takes longer when the selected value for the $Threshold_{QN}$ is not suitable. Although, *PS* technique is beneficial in large network, and network stabilisation is faster and more successful, the time required for the stabilisation increase as the values gets more and more inappropriate.



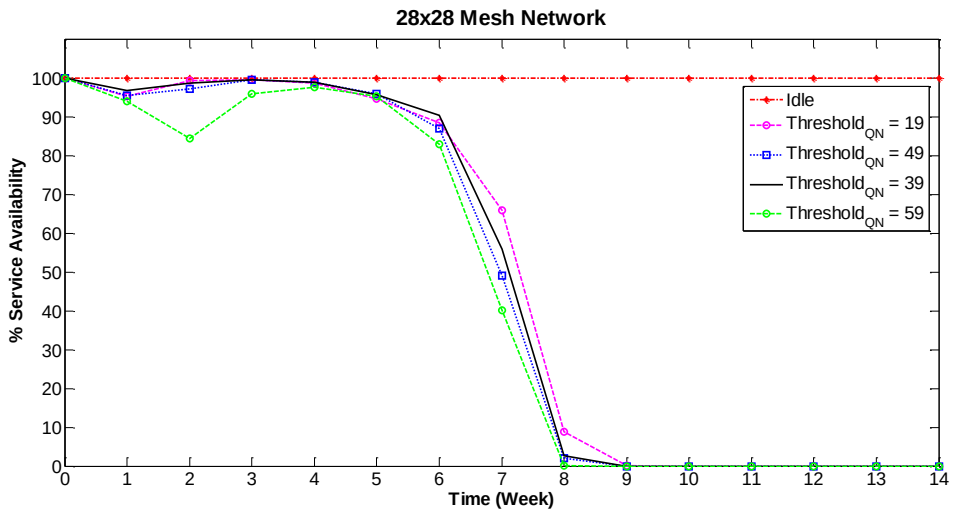
(a)



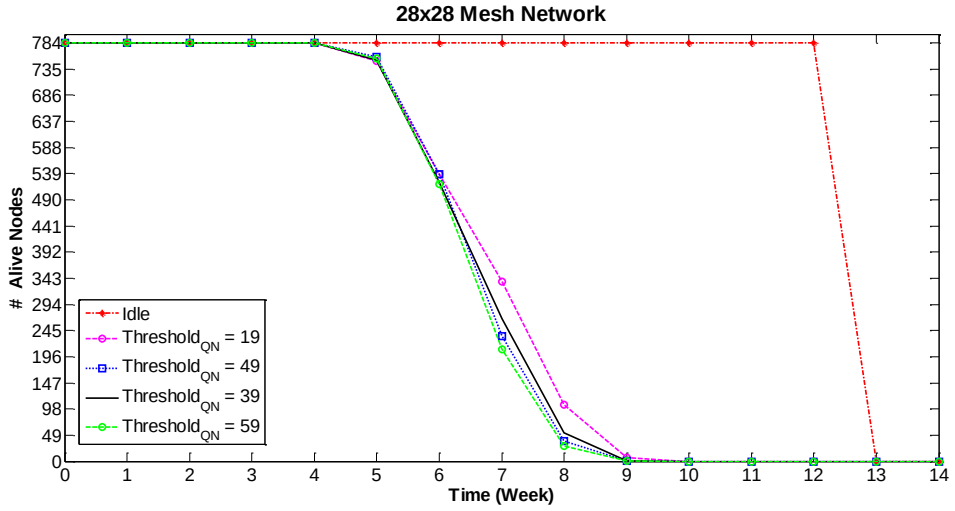
(b)

Figure 3.16: Experimental results: (a) % service availability, (b) # alive nodes for PS load balancing with different $Threshold_{QN}$ on 7x7 mesh network topology.

The figures presented above that analyses the effects of $Threshold_{QN}$ and T_{DECAY} shows that each alternative may produce different variations on service availability during the early part of the system lifetime, and different degradation patterns during the end of life. To make matters worse, other parameters also have an impact on the metrics of interest. By changing T_{QN} and T_{DECAY}, it is possible to significantly extend network lifetime at expense of service availability guarantees.



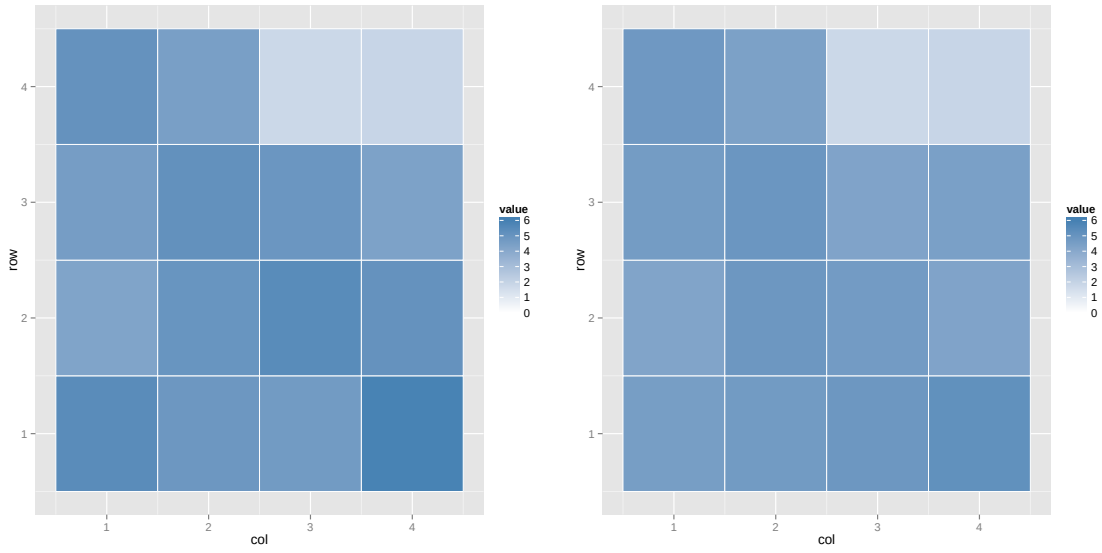
(a)



(b)

Figure 3.17: Experimental results: (a) % service availability, (b) # alive nodes for PS load balancing with different $Threshold_{QN}$ on 28x28 mesh network topology.

Analysing the distribution of QNs Furthermore, we want to show the high network coverage of *PS* technique. To do this, it is important to analyse the distribution of QNs by looking at the number of dropped events. We define dropped events as a service that has failed to be detected by all the service providers (nodes) that are in range of that event. Figure 3.18 (a), Figure 3.19 (a) and Figure 3.20 (a) shows the number of dropped events per node on *PS* technique, whereas Figure 3.18 (b), Figure 3.19 (b) and Figure 3.20 (b) shows the number of dropped events

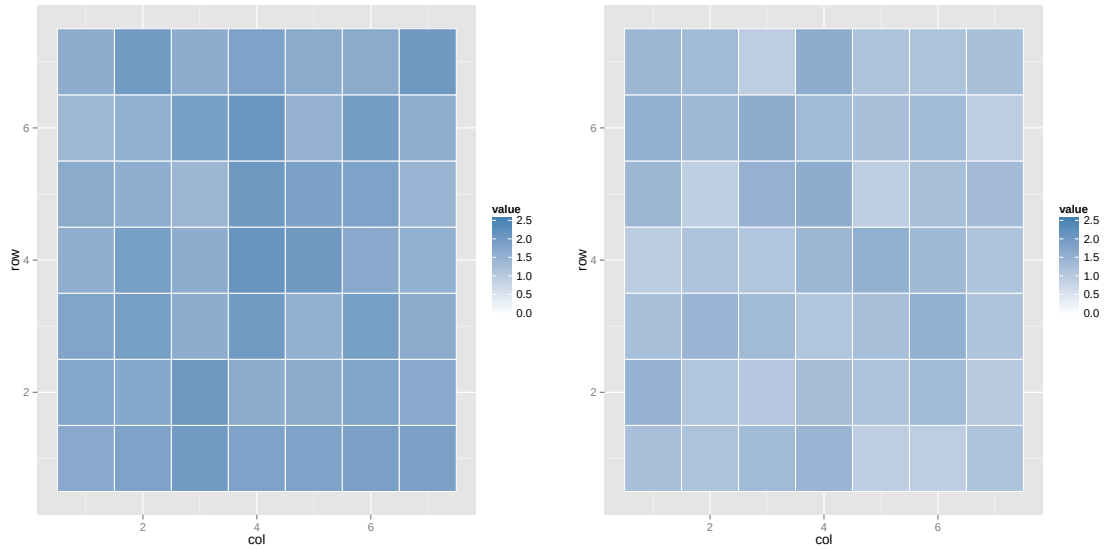


(a) (b)

Figure 3.18: Experimental results: distribution of dropped events over 4x4 mesh network topologies showing (a) on *PS* scenario, (b) on *Optimal* scenario.

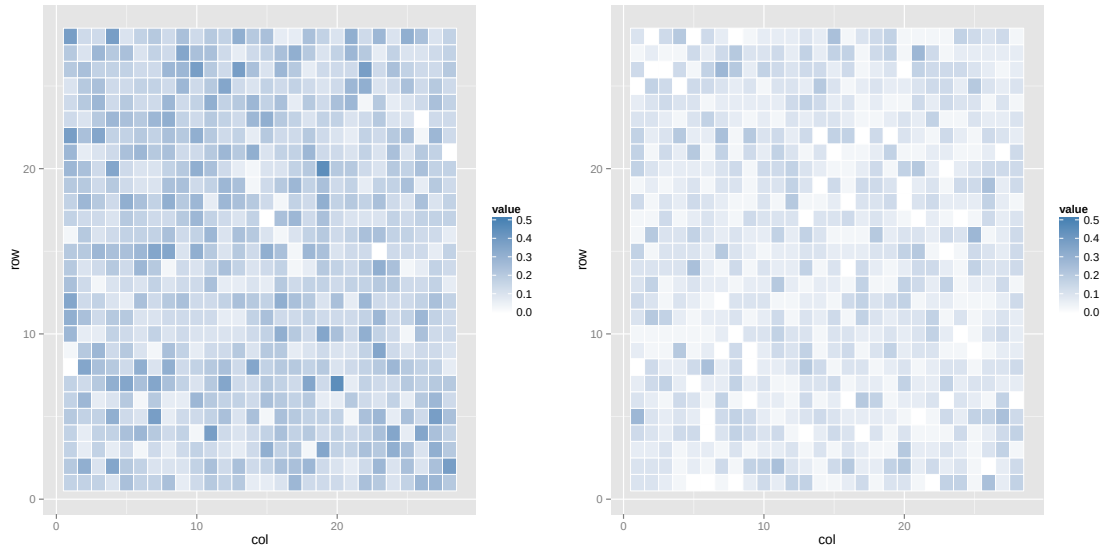
per node on the *Optimal* scenario. In all the figures display a heat map where each cell represents one node in the mesh network. In *PS* technique, we allow some level of redundancy in order to ensure high level of service availability. This means, some services are allowed to be executed by more than one sensor node. In a case where a service is allowed to be executed on more than one service provider, an event is consider 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 an event is considered as an event dropped by that service provider.

In Figure 3.18 (a) maximum number of the dropped events is higher Figure 3.19 (a) or Figure 3.20 (a), which gives hints about the level of applied redundancy in PS technique. Results of the number of dropped events per node is as not low as Optimal scenario which is natural. However, the difference between number of



(a) (b)

Figure 3.19: Experimental results: distribution of dropped events over 7x7 mesh network topologies showing (a) on *PS* scenario, (b) on *Optimal* scenario.



(a) (b)

Figure 3.20: Experimental results: distribution of dropped events over 28x28 mesh network topologies showing (a) on *PS* scenario, (b) on *Optimal* scenario.

dropped events per node on the same figures is very small, which also show that applied level of redundancy on PS algorithm as very limited. On the other side,

in both Figure 3.19 (a) and Figure 3.20 (a), the distribution of the dropped events over the network is almost equal, which then shows us the network coverage of *PS* algorithm. *PS* algorithm performs better in large networks where it limits the redundant resource allocation low. Figure 3.19 (a) and Figure 3.20 (a) supports this argument, where in Figure 3.19 (a) maximum number of dropped event per node is less than Figure 3.20 (a).

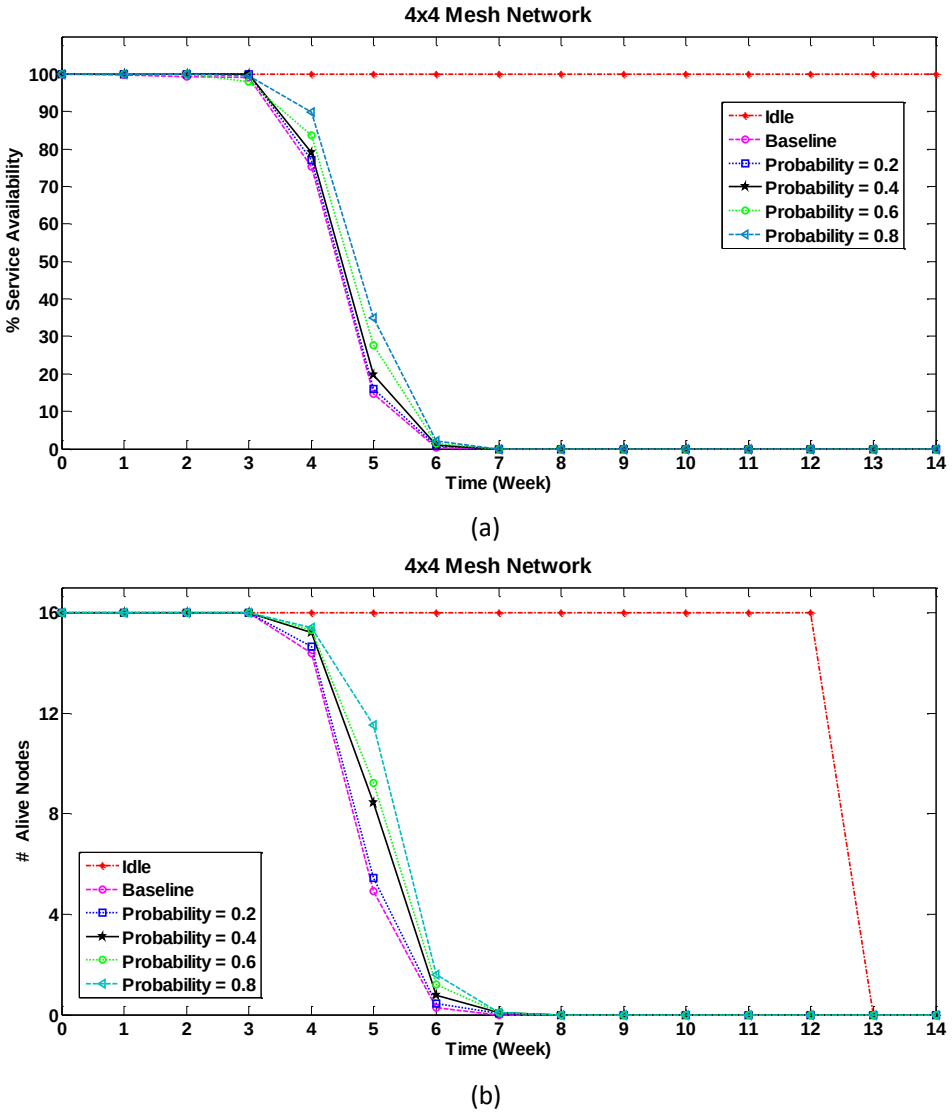


Figure 3.21: Experimental results: effects of different probabilities for the *Probabilistic* scenario on 4x4 mesh network topologies showing (a) % service availability, (b) # alive nodes.

Analysing Effects of Probability Values for the *Probabilistic* Scenario Among the four other scenarios that we compare *PS* against, the *Probabilistic* scenario is the nearest competitor of the *PS* as shown in the Figure 3.9, 3.10 and 3.11. As the nodes decide whether to execute an event based on their pheromone level in *PS*, we now analyse the effects of different values of probabilities for the *Probabilistic* scenario which also makes nodes active or passive depending on the given probability value.

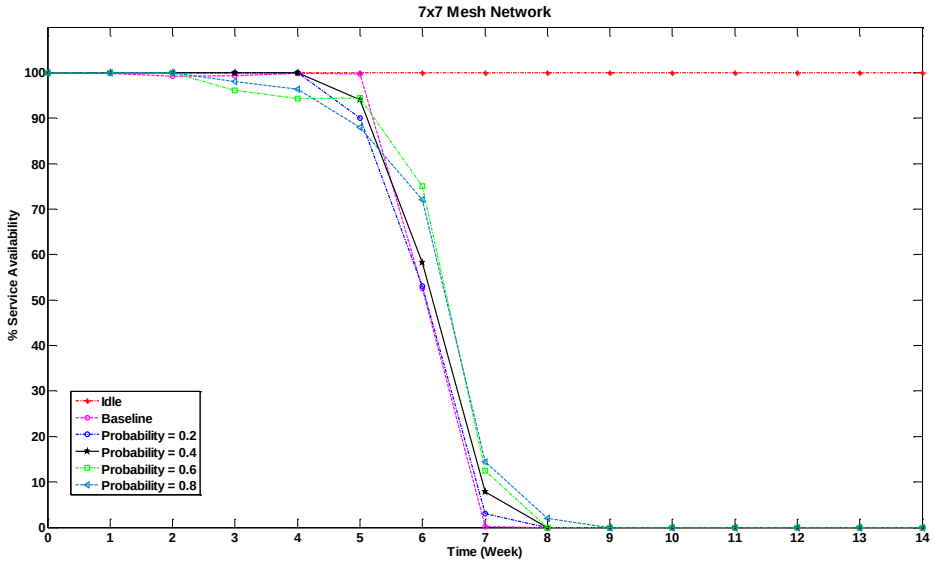
Figure 3.21, 3.22 and 3.23 shows the simulated results of five different probability choices on 4x4, 7x7 and 28x28 mesh network. To reiterate, the *Baseline* scenario represents a zero probability of staying idle when an event is detected, whereas here we analyse four different probability levels (0.2, 0.4, 0.6 and 0.8) of staying idle when an event is detected. The *Idle* scenario is also plotted to indicate the maximum lifespan of the network.

As one would expect, the simulated results shown in the three figures demonstrate that the lower the probability of staying idle, the less amount of energy is saved. Moreover, as the probability of ignoring detected events increases, the network lifetime improves and service availability improves. None of these probabilistic scenarios outperform *PS*, meaning that we can conclude that PS is more effective than probabilistic scenarios (as shown in Figure 3.9, 3.10 and 3.11). To make it clear, the referred plots show the results of the most effective probability choice against *PS* which is 0.8.

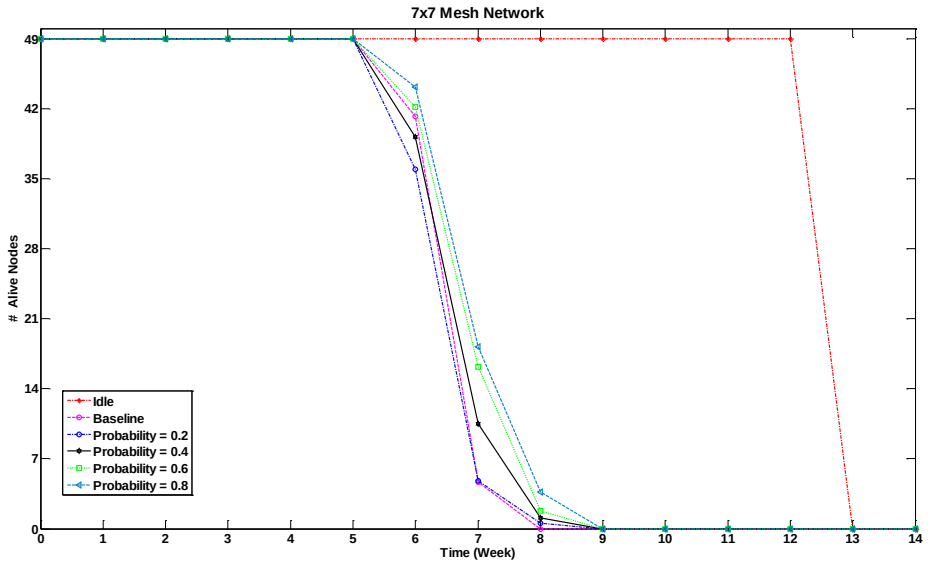
Analysing the Message Overhead of the *PS* Scenario One thing we need to analyse is the message overhead of the *PS* scenario in order to show weather the *PS* algorithm is overloading the network with pheromone packages.

Figure 3.24, 3.25 and 3.26 illustrate the total amount of network overhead found in PS on 4x4, 7x7 and 28x28 mesh network topologies using a stacked bar chart. The network overhead caused by the pheromone signalling process is shown in blue, whereas the network overhead caused by the data dissemination is shown with pink.

Data dissemination in this set of experiments refers to the number of packets injected in the network to transmit the successful event detections from QNs to the



(a)

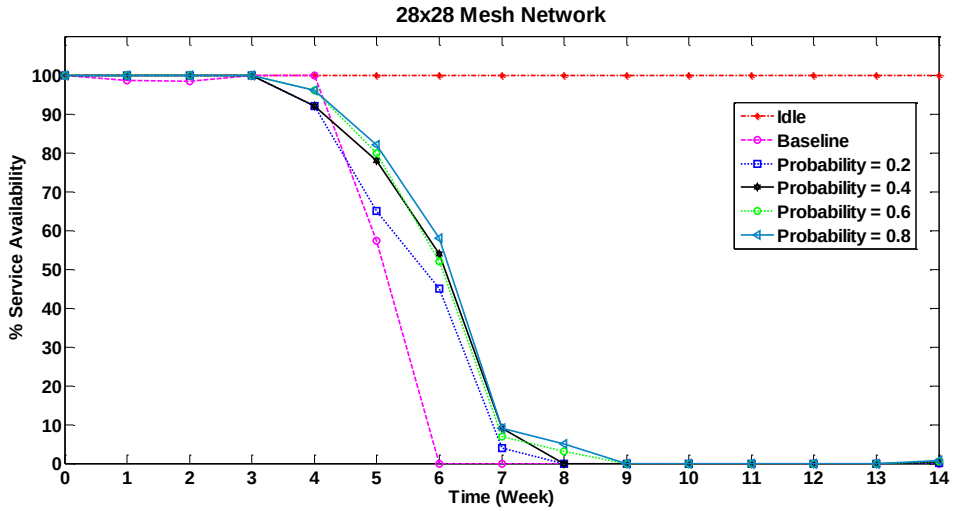


(b)

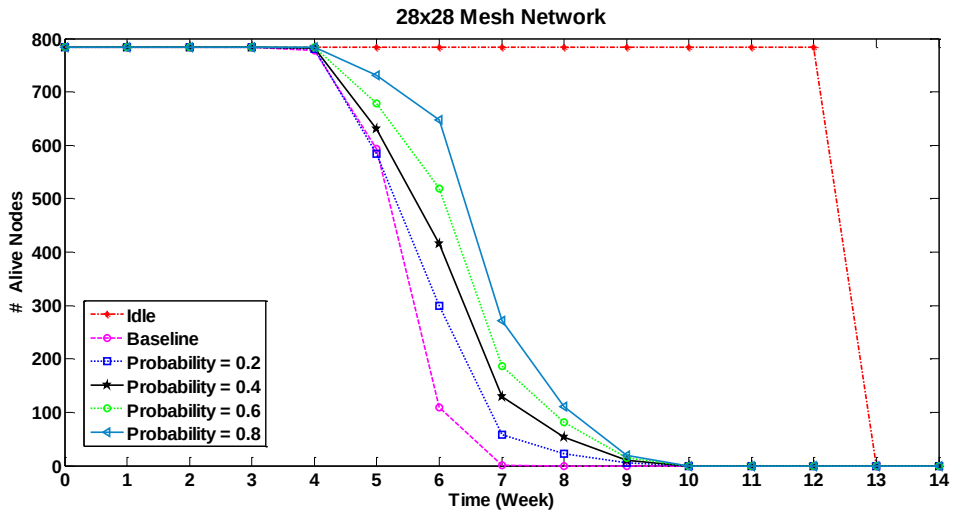
Figure 3.22: Experimental results: effects of different probabilities for the *Probabilistic* scenario on 7x7 mesh network topologies showing (a) % service availability, (b) # alive nodes.

sink. The number of message packets used for pheromone signalling is periodic based on T_{QN} of the network – 2000, 4000 and 11000 for the three networks respectively. The propagation cycle occurs 539 times for 4x4 mesh network, 269 times for the 7x7 mesh network and 98 times for 28x28 mesh network. To recall our previous

explanation, the pheromone signalling process starts from the QNs and pheromones are propagated only up to two hop neighbours. Therefore, in one propagation cycle one QN (the QN itself and it's one hop neighbours who also propagate pheromone packets) can inject maximum of 20 packets into the network in a mesh topology. This means that there are three important factors that influence the network overhead of the pheromone signalling process: the propagation cycle occurrence amount, the number of QNs and the fact that pheromones are propagated a maximum two hop



(a)



(b)

Figure 3.23: Experimental results: effects of different probabilities for the *Probabilistic* scenario on 28x28 mesh network topologies showing (a) % service availability, (b) # alive nodes.

distant neighbours. In a 4x4 network topology, if all the nodes are QNs and each QN can transmit up to 20 pheromone packets we should expect the pheromone signalling overhead in the 14 weeks to be $\bar{1}6$ x 20 x 539 = 172480 packets, in the worst case. This worst case should never occur as the nodes at the edge of the network have fewer neighbours. The results shown in Figure 3.24, 3.25 and 3.26 are based on the given TQN periods (2000, 4000 and 11000). The same T_{QN} values used in Figure 3.9, 3.10 and 3.11 to illustrate that *PS* algorithm provides high network performance in comparison to other scenarios. In case of changing these T_{QN} values, results will definitely change as message propagation only depends on its period which is represented with T_{QN}. The results shown in Figure 3.24, 3.25 and 3.26 are much lower than worst case pheromone signalling analysis which means: not every node is QN as we claim, and the *PS* algorithm is not overloading the network with redundant information. Instead, the *PS* algorithm propagates pheromones across the network without causing the network to be overloaded even though pheromone signalling packets are dramatically higher than data dissemination packets.

Interestingly, the number of data dissemination packages increases as the network

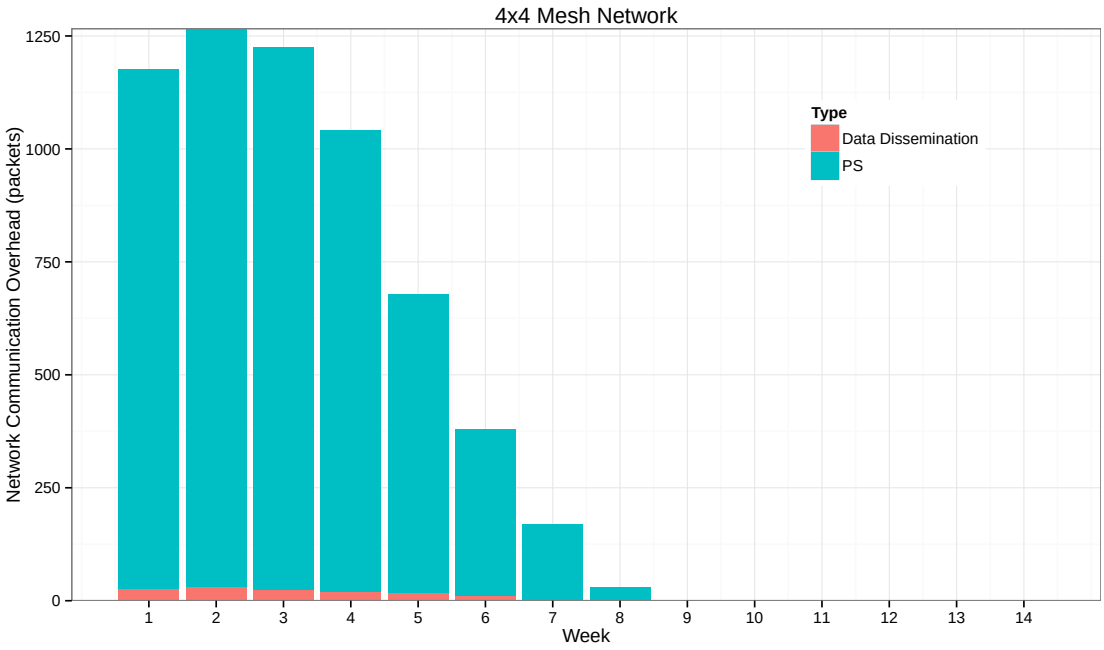


Figure 3.24: Experimental results: analysing data dissemination overhead and pheromone signalling overhead on 4x4 mesh network topology.

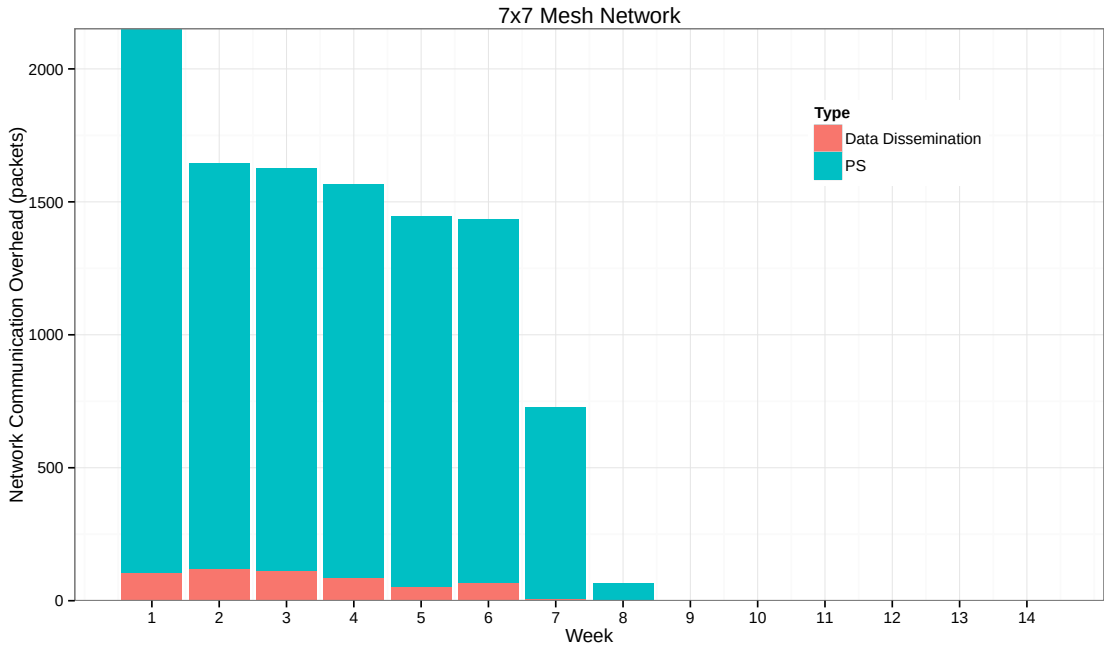


Figure 3.25: Experimental results: analysing data dissemination overhead and pheromone signalling overhead on 7x7 mesh network topology.

scales up as a result of routing from service provider to the sink. Considering the fact that we use a single sink in our experiments, the number of hops to route a package increases in larger network topologies which directly increases the network overhead of the data dissemination packages which is also very reasonable.

On the other hand, it is important to keep in mind that data dissemination packages are larger packages than pheromone signalling packages. The volume of the data dissemination packages are much higher, however, Figure 3.24, 3.25 and 3.26 only shows the numbers of the packages not the volumes that these packages worth in the network.

3.4.4 Long-term Effects of *PS* using the Sonoran Framework

Note: The work described in this section is based on Andrew Faulkner's research and has no input from myself. It is presented in this thesis only for the completeness.

In addition to the short-term analysis of PS that was performed on a TinyOS

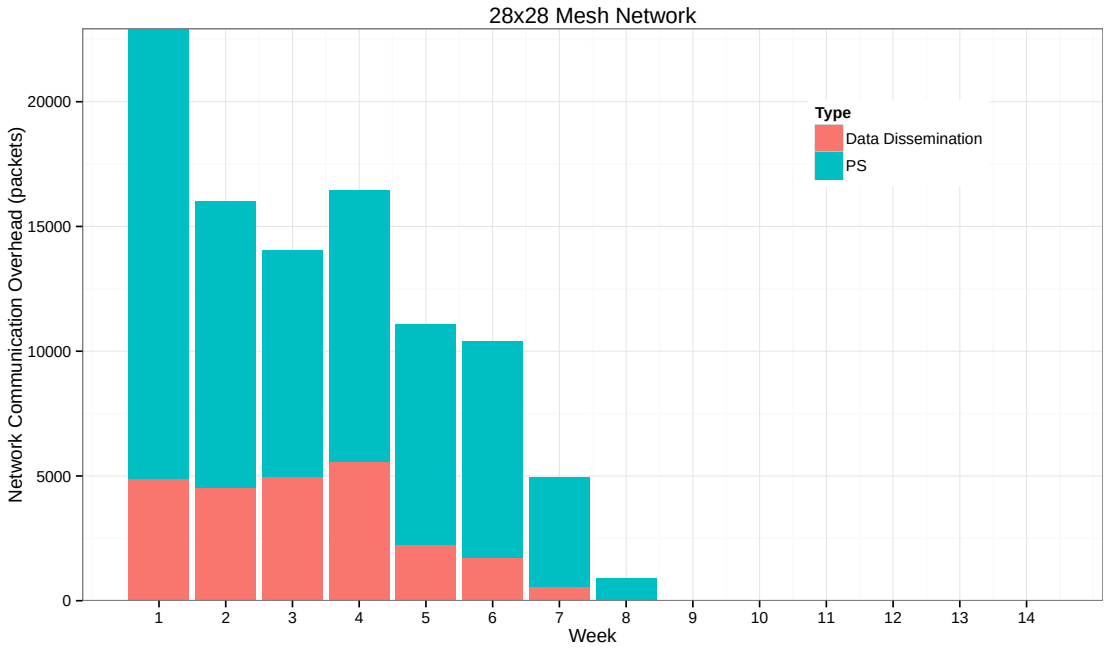


Figure 3.26: Experimental results: analysing data dissemination overhead and pheromone signalling overhead on 28x28 mesh network topology.

experimental testbed (Section 3.4.2) and the long-term analysis of PS on a system-level simulator (Section 3.4.3), this section provides further validation of the simulation results by presenting a study of the long-term effects of PS using the Sonoran Framework built on top of the Mote Runner platform [26].

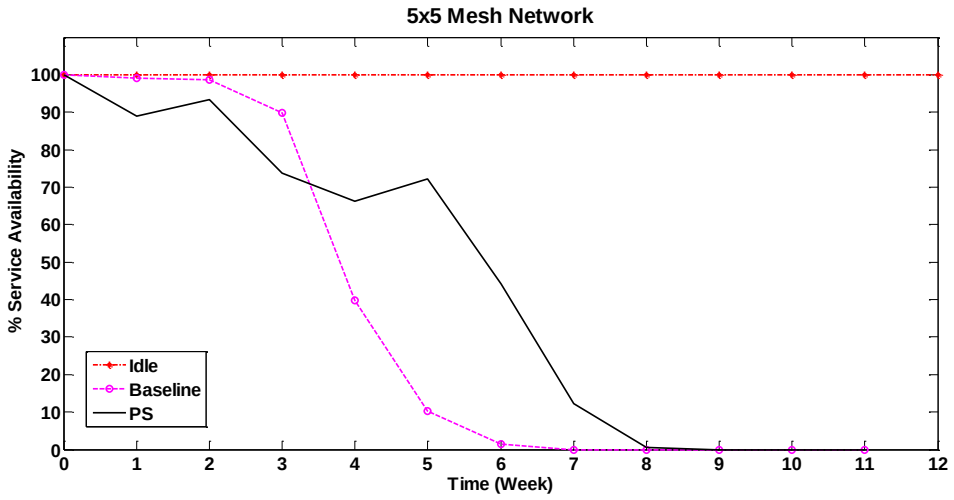
Faulkner [52] analysed the *Idle*, *Baseline*, and *PS* scenarios as described in Section 3.4.1, as well as his own energy-aware extension to PS, PSE. PSE introduces the ability for nodes to exhibit altruistic and egoistic behaviours. The implementation of PSE extends the behaviour of sensor nodes such that a particular mote makes decisions based on its current energy level, displaying selfless behaviour when it feels that its energy level is high and selfish behaviour when it has low energy. This means that QNs can relinquish their queen status if their energy level is lower than their neighbours, and WNs can choose to differentiate into a QN if their energy is greater than their neighbours.

All of Faulkners simulations were executed using Mote Runner beta 11.0 on a system running Microsoft Windows 8.1 x64 operating system, Intel Core i5-760 Quad Core @ 4 GHz CPU and 4GB Dual Channel DDR3 memory. To validate

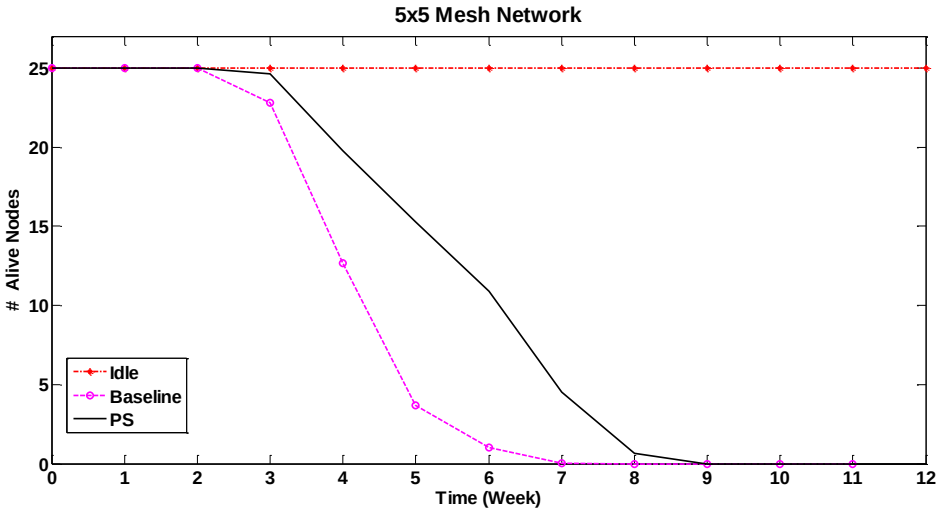
the correctness of the Fast simulator, experiments were run for 90 days, which is 12 weeks. A 5x5 mesh network topology is used instead of 4x4, 7x7, or 28x28 mesh network topologies to simplify the analysis (due to the excessive time required for the experiments of all three network topologies), which can still be considered a large network as it consists of 25 nodes. To increase confidence in the results, experiments were repeated 25 times and the results presented are mean values over the 25 runs, which was the maximum perceived to be achievable given the amount of time available to perform the evaluation.

Based on the results shown in [52], Faulkner demonstrates the similar behaviour of PS on his testbed as has been seen with the Fast simulator. Faulkners analysis of the long-term (12 weeks) prototype using the Sonoran Framework and the Mote Runner platform shows that *PS* outperforms both the *Idle* and *Baseline* (referred to as ON in [52]) scenarios. Moreover, Faulkner's own energy-aware extension of PS, PSE, shows an increase in network lifetime and improves the network performance of PS. Faulkner also evaluated the effects of adjusting the number of hops from the queen that pheromone is propagated and found that increasing the number of hops increases the percentage of detected events (service availability) ([52], Figure 6.3) and the network lifetime ([52], Figure 6.5) for PSE.

To provide a direct comparison between Faulkners Mote Runner implementation and the Fast simulator, and therefore evaluate the correctness of Fast with respect to a near-hardware implementation, we have performed the same experiment on a 5x5 network. Figure 3.27 illustrates simulated results on 5x5 mesh network and compares *PS* against *Idle* and *Baseline* scenarios. Figure 3.28 shows Faulkners results for *PS* against the *Fast* simulator, comparing the number of nodes alive (as this is the only data available for PS in [52]). As you can see the results are similar, although Faulkners implementation has a higher network lifetime than Fast. The number of alive nodes decreases more quickly in *Fast* than Faulkners implementation. This could be due to Faulkner using a different decay period (the selected value is not presented in [52]). However, considering that these are two distinct implementation of PS, the behaviours in both implementations are very similar and share common patterns with respect to the other scenarios. Incidentally, the performance of the



(a)



(b)

Figure 3.27: Experimental results: effects of different probabilities for the *PS* scenario on 5x5 mesh network topologies showing (a) % service availability, (b) # alive nodes.

Fast simulator is orders of magnitude greater than Mote Runner, with a single 5x5 experiment execution taking on average 14 seconds in Fast, but 29 minutes 20 seconds in Mote Runner ([52], Table 6.1).

As the work presented in this section does not belong to myself, and it is presented only for the completeness of this research to validate the correctness of Fast simulator, please find further details in [52].

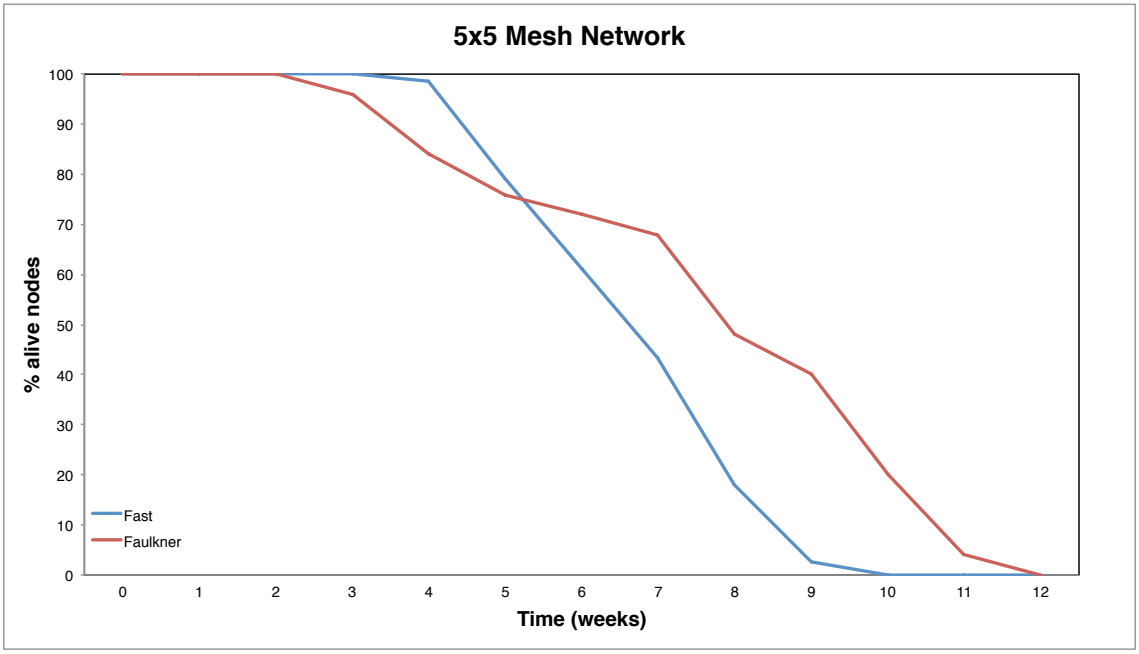


Figure 3.28: Experimental results: Fast and Faulkner (Sonoran) frameworks side-by-side on 5x5 mesh network topology.

3.5 Summary

In summary, this chapter presented a bio-inspired solution to address the challenging trade-offs of WSNs that are due to their limited processing capabilities. We developed a bee-inspired algorithm that mimics the dynamic and distributed characteristics of the nature. To improve the survivability of WSNs against the network conditions, we correlated the sensor nodes as bees, and bee colony as the network in our *PS* algorithm.

In this chapter had two major goals: addressing the service availability versus energy consumption trade-off with the PS algorithm, and demonstrating the 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 *PS* 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 on the experimental results.

This chapter has presented 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. 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.

We run extensive system-level simulation experiments, we analysed effects of *PS* on three different size of mesh network topologies to show the effectiveness of the algorithm on scale. We compared *PS* against other scenarios to illustrate the benefits of load balancing, and *PS* performs. 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 *PS* technique delivers 10% longer network lifetime on average, and up to 85% higher service availability in later stages of the system lifetime.

Analysis show that *PS* performs better on large scale networks. The main contribution of *PS* is to remove the redundancy. As the the network scales up, more redundant computation is involved and therefore, *PS* performs better in removing the redundancy in large scale WSNs.

We also analysed various values of for some key parameters ($Threshold_{QN}$ and T_{DECAY}) are tested compared against each other on *PS*. The experiments have also shown that the performance of *PS* algorithm is highly depends on its parameters, giving system designers the flexibility to choose different points over the trade-off

between service availability and network lifetime.

In the next chapter, we inspect the importance of the parameters in a systematic manner rather than trial-and-error.

Chapter 4

Search-Based Parameter Tuning on the Pheromone Signalling Based Load Balancing Algorithm

In the previous chapter we devised a bio-inspired load balancing algorithm that addresses the trade off between service availability and network lifetime. We performed a rigorous analysis of the algorithm in real sensor deployment and system-level simulation on different network topologies. As part of the process, we found that, like many load balancing techniques, *PS* has numerous parameters and its performance depends on the values chosen for its parameters. Finding appropriate parameter values for a given scenario is not trivial. Furthermore, the parameter values that are appropriate for one scenario are rarely applicable to another that has, for instance, a different application profile or network topology.

In this chapter we present a search-based approach to selecting a set of optimal parameters for *PS* for any given network scenario as Figure 4.1 illustrates. The approach considers the service availability and energy dissipation figures obtained by each configuration of the load balancing technique, and uses those values to explore the parameter space in search of optimal solutions. To accelerate the search, we also define a number of improvements to the simulator used to evaluate each parameter configuration. The proposed parameter tuning approach is then evaluated by analysing the best configurations it can find for several scenarios, and we use

Principle Component Analysis to identify which of the parameters have the most critical effect on the quality of the solutions.

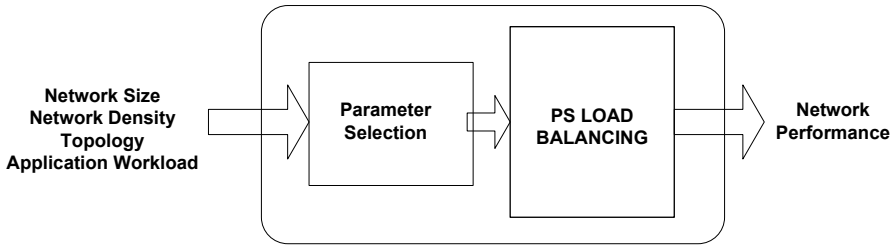


Figure 4.1: Automating the parameter selection for the PS algorithm

This chapter considers two main challenges. The first objective of this research is to automatically tune the parameters of the *PS* algorithm for a given network configuration in order to maximise service availability and minimise energy consumption. The second challenge is to accelerate the evaluation of the search technique towards finding optimal parameter configurations quickly by speeding up the simulation infrastructure. We now define the fitness function we will use to evaluate parameter selection throughout the chapter. As with the previous chapter, our performance metrics have been defined as:

1. *service availability*: the number of services that are successfully completed divided by the total number of requested services within a period of time;

2. *total energy consumption*: the sum of communication and computation energy consumption within a period of time.

Our definition of service is the same with the previous chapter, however, to remind what the service is to the reader we would like to restate it here. A *service* is defined as the composition of a number of inter-communicating tasks, and therefore a service is considered to be successfully *detected* only if all of its tasks are executed by the nodes just the same way as we defined in the previous chapter. Accordingly, our fitness function considers both the service availability and the number of alive

nodes. We target high service availability and aim to minimise the number of time intervals (weeks) without service (based on the number of the alive nodes) whilst also minimising the total energy cost.

Chapter Contributions The contributions of this chapter are listed below:

- the automation of parameter selection of the *PS* algorithm, through a Simulated Annealing search metaheuristic.

- The analyses of parameters of the PS algorithm using Principle Component Analysis (PCA) and showing their effects individually.

- The development of a new simulation infrastructure to evolve the search metahueristic faster.

Chapter Structure Section 4.1 starts by describing the general characteristic of Simulated Annealing metaheuristic technique, and explains how the algorithm is applied to the challenge of discovering a set of optimal parameters for the *PS* algorithm. Section 4.2 defines a new simulation infrastructure which focuses on performance over accuracy. We measure the level of accuracy and execution time of newly developed simulator (SuperFast) using our network performance measurements which are service availability and network lifetime. We compare SuperFast and Fast simulation infrastructures and illustrate the performance effectiveness of both simulators. Section 4.3.1 demonstrates the experimental results on pre-evaluated network topologies in the Chapter 3 to analyse the effectiveness of the search-based parameter tuning on PS. *Principal Component Analysis* is applied on PS to show the effects on the parameters in Section 4.3.2. Additionally, new network topologies are evaluated to illustrate parameter tuning on several untested network topology in Section 4.3.3.

4.1 Search-based Parameter Tuning Using Simulated Annealing

Relating back to the Chapter 2, we discuss a variety of different search metaheuristics. We underline that the *Simulated Annealing* metaheuristic algorithms have low level of complexity: they are easy to program and are time efficient. As they do not cover the whole search space like exhaustive search techniques, they are practical, computationally affordable and time efficient for multi-criteria metrics with several parameters. Due to cost and performance effective nature of *Simulated Annealing* metaheuristic search algorithms, we decided to use them for our *PS* parameter tuning approach.

In *Simulated Annealing (SA)* algorithms positive improvement is always accepted, whereas negative improvements may be accepted probabilistically and depending on the temperature T. According to *SA* theory; the worse the move the less likely it is to be accepted. A negative move is less likely to be accepted the cooler the temperature is. The temperature T starts with a high value and gradually cooled as the algorithm progresses. A typical pseudocode of the *Simulated Annealing* is given in the Listing 4.1.

Listing 4.1: :SA Algorithm

```
1  Initialise (S_0, T_0, L_0)
2  k=0;
3  S=S_0
4  repeat
5    for l=1 to L_k
6        S_new = generateNeighbours(S)
7        Δ = f(S_new) − f(s)
8      if(Δ > 0)
9          S = S_new (accept)
10       else if(exp^(Δ/T_k) > U(0,1))
11         S = S_new (accept)
12       else reject
13   k + k + 1;
14   Calculate_Lenght(L_k)
```

```
15    Calculate_Temperature(T_k)
16 Until Stopping_criteria
17 Solution is best so far
```

Certain design decisions must be taken in order to run the simulated annealing algorithms:

1. ***SA configuration values***: SA configuration values are temperature related control parameters like the initial temperature T_0 and the temperature is a virtual temperature introduced only by analogy to the cooling process of metals at any time during SA process T_k. The *cooling rate*, which is often selected between 0 and 1, indicates how much the temperature be cooled. The frequency at which cooling is applied also affects the search. In some SA algorithms cooling is applied once in every neighbourhood generation, whereas researchers who prefer to explore larger search spaces applies the cooling less frequently, only once in several neighbourhood generation.

2. ***Initial solution***: Often in SA algorithms, the parameters for the initial solution are the optimisation parameters and they are set to default values, such as the minimum values within the search space.

3. ***Neighbourhood generation***: Neighbourhoods are the part of the search space that are prospective solutions of the problem that are produced after altering a solution. Neighbourhood selection varies depending on size of the search space based on number of parameters and user preference. It is more likely that SA will find a good set of parameters with highly populated neighbourhood set, however, a highly populated neighbourhood set will also increase the cost of the search as each neighbour needs to be evaluated by the fitness function.

4. ***Fitness function***: The definition of the fitness function depends on the application and varies. Researches show that fitness functions have to correlate closely to the algorithm's goal and have to be carefully defined in order to produce usable results that are non-trivial to the problem [121]. Plenty of

research studies particularly focus on methods on the importance of the defining successful fitness functions and how to specify effective fitness functions to achieve the most significant results.

To apply SA to our *PS* technique, we make the following choices for each of the previously mentioned design decisions:

1. ***SA configuration values***: We set the initial temperature T_0 to 100, and define the stopping criteria as $T_k < 1$. The cooling rate is applied once in every twenty neighbourhood generations in order to explore the search space wider.

2. ***Initial solution***: Each solution is a set of assignments to the key parameters of the load balancing technique together with temperature which is represented as a tuple $S =< threshold_{QN}, T_{QN}, T_{DECAY}, QN_{INITIAL}, T_k >$. Parameters of each solution $S_i \in S$ are tuned according to the given range in Table 4.1 with the provided step values. Parameters of the initial solution are set to minimum default values presented in Table 4.1.

Table 4.1: Parameters setting for the Simulated Annealing algorithm used for 7x7 and 28x28 mesh network

Parameters	7x7 Parameter range	28x28 Parameter range	Step value
$threshold_{QN}$	3-40	3-40	1
T_{QN}	2000-40000	2000-40000	1000
T_{DECAY}	2000-10000	2000-10000	1000
$QN_{initial}$	2-30	12-120	1

3. ***Neighbourhood generation***: 16 neighbours are generated in every neighbourhood generation. We have evaluated the three different scenarios to show their effects on *PS* which will be explained in details in Section 3.4. Once the fitness evaluation of the entire neighbourhood is complete, candidate solutions are selected according to the applied scenario and it becomes new starting

point for the further rearrangements. To reduce the complexity of the fitness function evaluation, normalisation is applied to the performance metrics.

4. ***Fitness function***: We defined our fitness function as combination of total service availability and the minimum number of intervals without service detections. For a better understanding, we give an example to visualise our performance metrics first. The simulation results of one solution are shown in Figure 4.2. Simulation time is set to 14 weeks and we evaluate the percentage of service availability and the number alive nodes weekly for each solution.

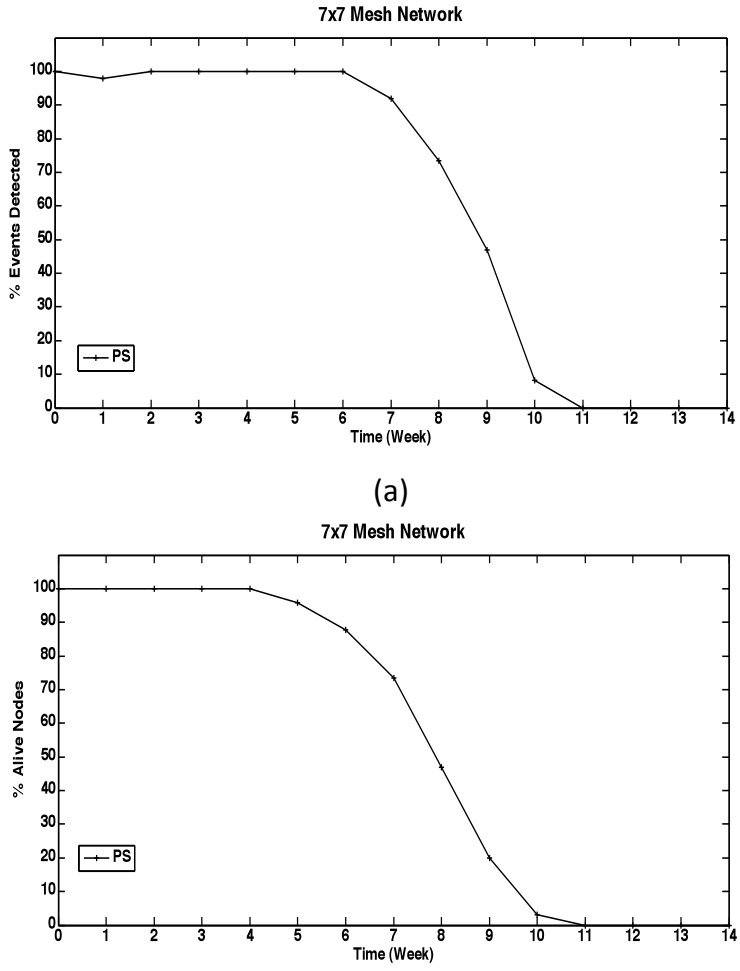


Figure 4.2: Fitness is derived from the sum of the values in (a) and the size of the zero tail in (b).

Illustrated in Figure 4.2, the definition of the fitness function is the combination

of total service availability (the sum of % detected events), which is 917 and the number of intervals without service detections, 4. The best solution is the one that has the highest total service availability and lowest number of intervals without service detections. Demonstrated by the given example in Figure 4.2, our proposed algorithm targets maximum service availability and minimum energy consumption by defining a simple, but effective, fitness function.

Summary

In this chapter we aim to optimise the *PS* algorithm by automating its parameter selection for any given network topology. For this purpose, we tailored *Simulated Annealing* search metaheuristics and defined multi-objective fitness function to provide maximum service availability and minimum energy consumption. In the rest of this chapter, we explain the evaluation infrastructure and show the experimental results.

4.2 Evaluation Infrastructure

Previously in Chapter 3 Section 3.3.3 we explain our system-level, abstract simulator and defined its design goals. To remind *Fast* simulator to reader, we summaries its design objective and characteristics as follows:

Our design objectives was to have short implementation time, high performance and cost efficiency over level of accuracy. To achieve our design objectives, we created the *Fast* simulator which is an event-driven simulator. *Fast* uses the JavaSim library [109] to synchronise the events of the multi-threaded simulation engine. It has been developed in Java to use the advantage of the encapsulation of object-oriented programming and runs the application and platform models in parallel. The *PS* algorithm runs on top of the platform model and is integrated with the simulation infrastructure.

Our idea was to apply search-based parameter tuning - to run *SA* metaheuristic search on the *Fast* simulator. We designed the search metaheuristic and conduct some experiment on the *Fast* simulator. The main issue about running *SA* meta-

heuristic search on *Fast* was the execution time. The shortcomings of our experiments conclude as the search space increase, execution time of *Fast* exponentially increase. The motivation behind creating the *SuperFast* simulator that accelerates the simulation infrastructure developed because of the execution time required for the *SA* search metaheuristic to evolve.

Table 4.2: Comparison of simulators

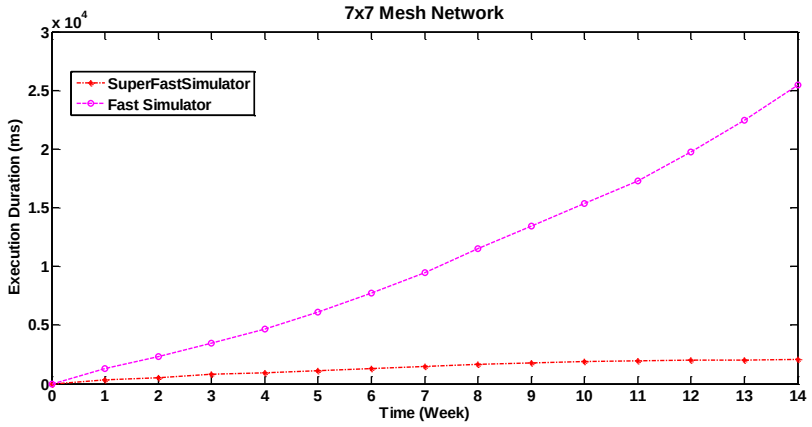
	Features	Fast	SuperFast
Considered Engines	multi-hop	✓	X
	scheduling queues	✓	X
	contention on wireless channel	X	X
	energy consumption while idle	✓	✓
	energy consumption while processing	✓	✓
	energy consumption while remapping tasks	✓	✓
	energy consumption for communication with sink	✓	X
Search Engine	multi-threaded	✓	X
	JavaSim library	✓	X

The multi-threaded nature of the *Fast* simulator allow us to simulate multi-hop relations between nodes, task scheduling queues and energy consumption in terms of idle, processing and communication of the nodes. Although, the multi-threaded nature of the *Fast* is an advantage in terms of high level of accuracy, the same characteristic appears to be disadvantage for the *SA* search metaheuristic to evolve. Due to the complexity of the search process, the required memory space for the hardware components and most importantly the time factor, we decided to create a more abstract simulator than *Fast* which will allow us to work faster. Removing the complexity of concurrent programming, and lowering the accuracy in certain limits reduced the computational time and the cost of the simulation work, without sacrificing significant accuracy. In Table 4.2 features of simulators are analysed

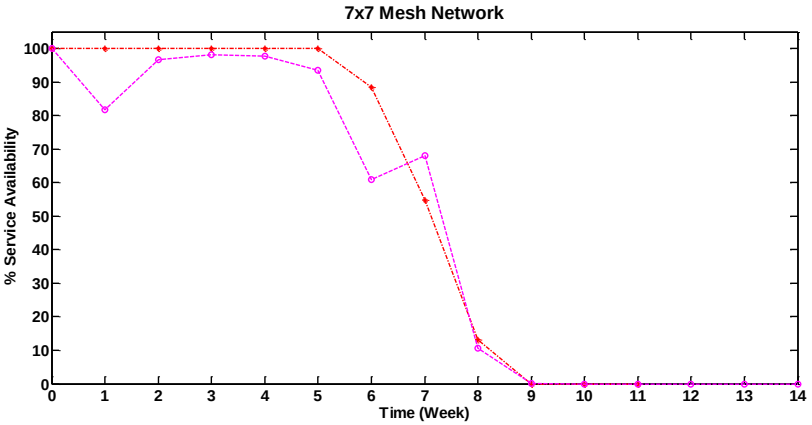
from component considerations and search engine features of the two simulators. Disabling multi-threaded nature of the *Fast* simulator and implementing sequential programming instead of concurrent make *SuperFast* simulator time efficient. Since most of the energy consumption kept and taken under consideration in *SuperFast*, level of accuracy is preserved. Before we start presenting the results of SA, we show a detailed comparison between the Fast and SuperFast simulators in terms of execution time and performance accuracy.

In order to evaluate where *SuperFast* maintains an acceptable level of accuracy compared to *Fast*, we compare two simulators on 7x7 and 28x28 mesh topologies. Experiments presented using the *Fast* simulator on the percentage of service availability and the number of alive nodes are identical with the results presented in the previous chapter. We repeat the same experiments on the *SuperFast* simulator using same parameters selections for the *PS* algorithm and using the same platform specifications. Figure 4.3 illustrates comparisons between the *Fast* and *SuperFast* simulators on 7x7 mesh network topology, where Figure 4.4 compares the two simulators on 28x28 mesh network.

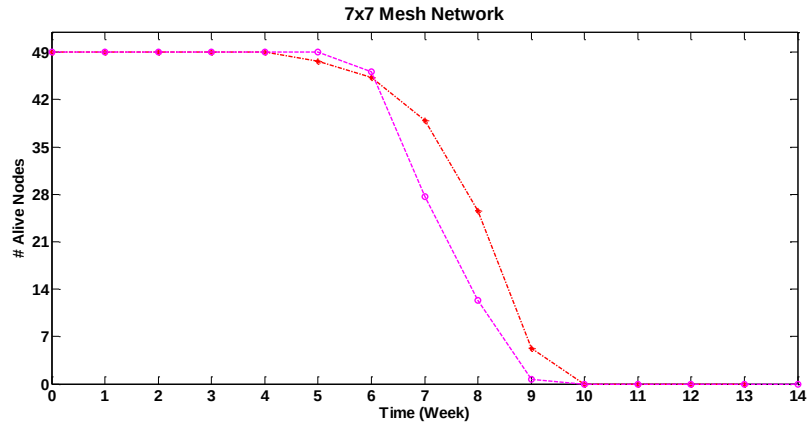
In Figure 4.3 (b) shows the pheromone stabilisation on 7x7 mesh network during the first and the sixth week. On the other side, in Figure 4.4 (b) network stabilisation only once during the first week. In small networks, pheromone stabilisation take longer and it is more difficult as opposed to large networks. As a result we exhibit sort of behaviour as Figure 4.3 (b) and Figure 4.4 (b) shows on different scale. In the *SuperFast* simulator, the pheromone propagation implemented by adding up the estimate amount of received pheromone to the nodes pheromone level because of the absence of the multi-threaded nature of the simulator. By implementing the pheromone propagation precess sequentially, we enable faster simulation. As a results, we exhibit slightly less accurate, optimistic performance in terms of the percentage service availability shown in both figures. The number of alive nodes illustrated in Figure 4.3 (c) and Figure 4.4 (c) also shows that *SuperFast* the number of nodes decrease sharper than *Fast*, and many nodes run out of energy at the same time. This also validates that estimated processing time, energy consumption and cost of the pheromone propagation is cut at the same time from the nodes in



(a)



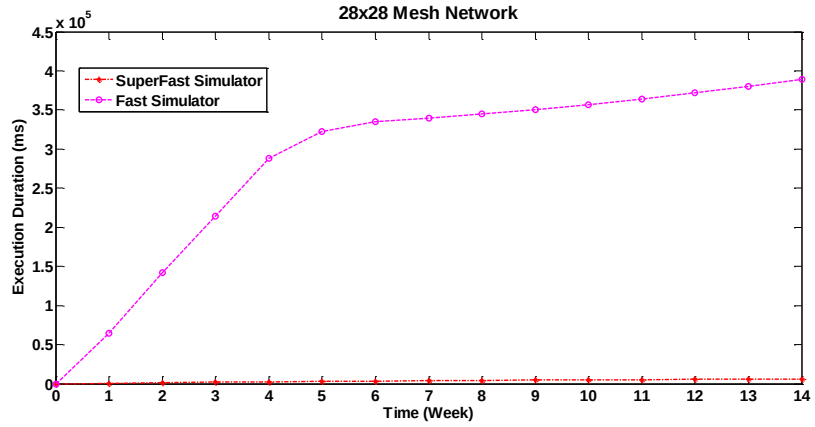
(b)



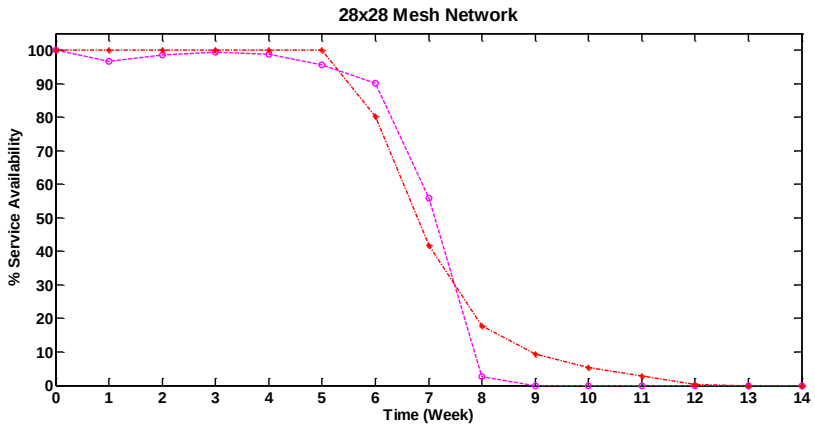
(c)

Figure 4.3: Simulator comparison of execution duration, % service availability and # alive nodes on 7x7 mesh network.

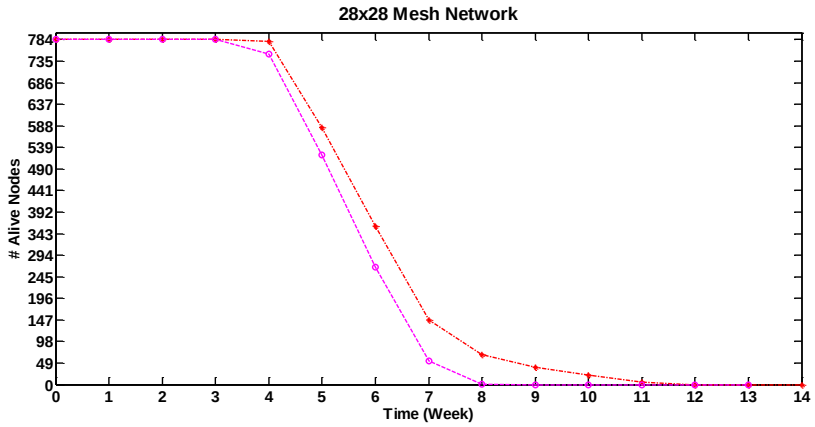
SuperFast. When nodes calculate the energy costs, they do not do it synchronized, instead they do it on-demand. As a results, due to the nature of milti-threaded



(a)



(b)



(c)

Figure 4.4: Simulator comparison of execution duration, % service availability and # alive nodes on 28x8 mesh network.

versus sequantial implemented simulators Figure 4.3 and Figure 4.4 exhibits more realistic simulator with *Fast* and more optimist simulator in *SuperFast*.

In short, the advantage of using *SuperFast* is the short execution time, whereas the disadvantage of it is the lower level of accuracy and being less precise in terms of the service availability and network lifetime. The simulations need to be repeated many times in *SA* search metaheuristic like any other metaheuristic technique. As a result, the desired simulation infrastructure for the metaheuristic search is the simulation infrastructure with the shortest execution duration together with the highest performance efficiency. In this regard, we decided to use the *SuperFast* simulator over *Fast* where we achieve the desired characteristics of the simulation infrastructure in terms of performance and execution time.

4.3 Experimental Results

In this section, we present experimental results of the parameter selection tool in three stages. In Section 4.3.1, we show the effectiveness of SA-based parameter selection tool on known schemas: 7x7 and 28x28 mesh network topologies to show the effectiveness of the parameter selection tool. We achieved high network performance on the 7x7 and 28x28 mesh network topologies by tuning the parameters manually as we illustrated in the previous chapter. The aim of this section is to compare the performance of our manually tuned parameters over the performance of the parameter selection tool that uses the *SA* metaheuristic search. By doing this set of analysis, we show whether the parameter tool is effective or not. At the same time, we also analyse how much effective was our manual parameter tuning for the given mesh networks. Later in Section 4.3.2, we present experimental results on *Principle Component Analysis* to show the importance of the parameters on the performance of the *PS* algorithm. By applying *PCA*, we show the how much each parameter affects the *PS* algorithms results. Using the results of the *PCA*, we modify the search metaheuristic that will allow us to achieve successful set of parameters faster by guiding the search. Last, in Section 4.3.3, we apply the parameter selection tool onto untested/ new network schemas to analyse the effectiveness of the *PS* on the new network schemes without spending time to tune the *PS* algorithm manually.

4.3.1 Verifying the Effectiveness of Parameter Selection Tool on a Known Schema

This section presents the initial set of experimental results. The goal of the first set of experiments is to demonstrate the behaviour of the *SA* algorithm towards selecting parameters that achieve high service availability and low energy consumption, and analyse how good our intuitive parameter (which we used in the previous chapter) search is based on small (7x7) and large (28x28) mesh network topologies. Three different scenarios have been prepared for this purpose.

1. *WS*: Represents the behaviour of the SA when the search uses a fixed small step size for the parameter values and the selection of the first encountered fitter neighbour;

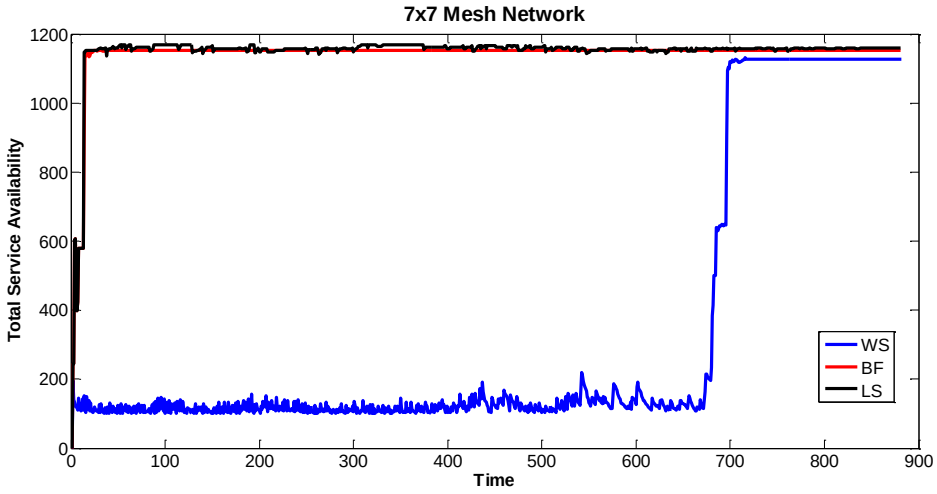
2. *BF*: Represents the behaviour of the SA when a search changes step size dynamically (between large and small) depending on the improvement on fitness and selects randomly among fit neighbours;

3. *LS*: Represents the behaviour of the SA when a search changes step size dynamically (between large and small) depending on the improvement on fitness, ranks the fit neighbours and selects the fittest neighbour;

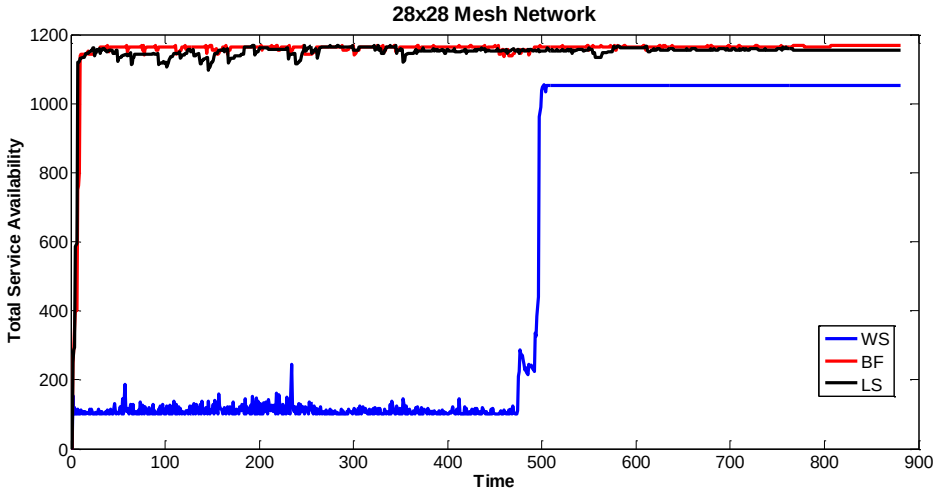
We mention how important to define the fitness function in Section 4.1. We also mentioned that defining the neighbourhood and step sizes are also very important and depends on the application. The desired step sizes should be in the range of not jumping too far that will cause missing good solutions, but at the same time not a tiny amount that will have very little effect on the *PS* performance and *SA* evolving. In order to specify the most appropriate step sizes for the solution space of the *PS* algorithm, we used static (WS scenario) or dynamic (BF and LS scenarios). We have not used random step sizes to make the parameter selection tool more systematic.

Comparison against three scenarios on total service availability We initially implemented the WS scenario and we found out that there is not only one

optimal and fittest solution for the PS algorithm. As a result, we have decided to explore the larger search space within the shortest time as much as possible.



(a)

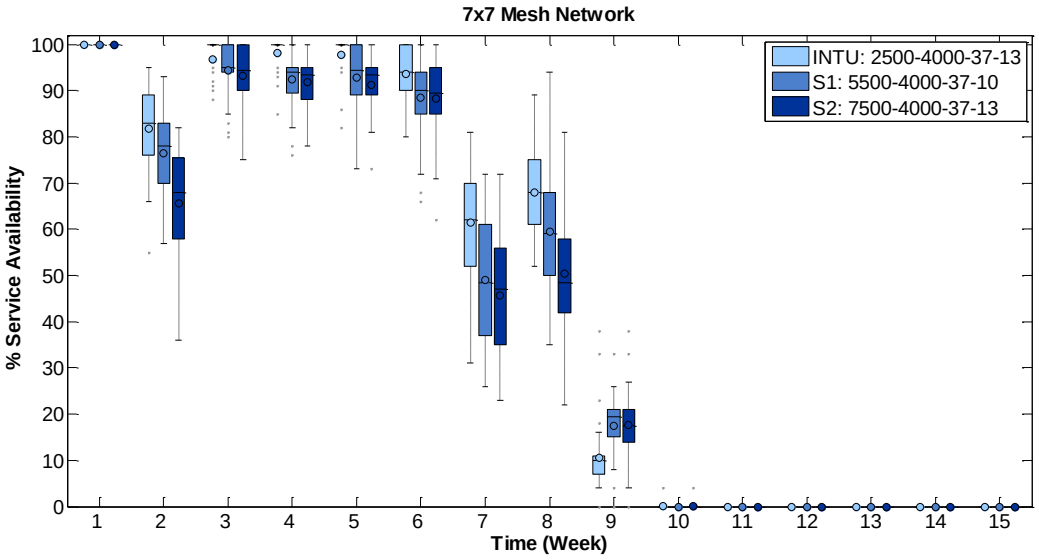


(b)

Figure 4.5: Experimental results: improvements on total service availability showing (a) 7x7 mesh network topology, (b) 28x28 mesh network topology.

Figure 4.5 (a) on 7x7 Mesh Network, (b) on 28x28 Mesh Network show improvements on cumulative total service availability over acceptable steps of *SA* for WS, BF and LS. In the 7x7 Mesh Network initial implementation, WS achieves high total service availability in very low temperatures and the algorithm does not converge fast due to small step sizes and neighbour selection. Whereas, LS and BF search

algorithms converge much faster as these scenarios allow larger step sizes when there is no improvement. In both networks, the LS scenario performs the best in terms of achieving higher service availability in a shorter time, since the algorithm ranks neighbours and picks the fittest neighbour. Although the WS scenario converges faster in a large network in comparison to a small network, all three scenarios behave similarly on both networks in terms of achieving higher service availability and time consumption.



(a)

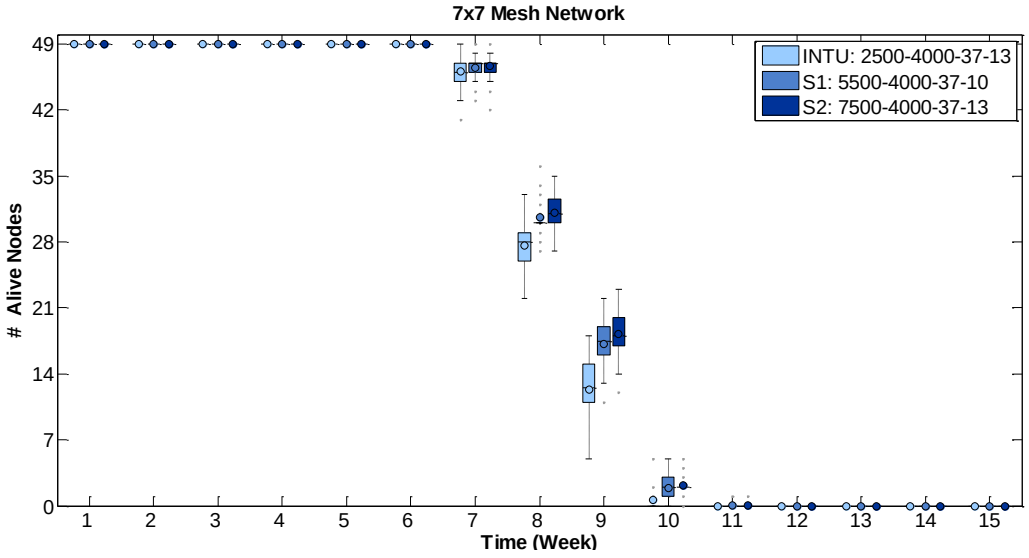


Figure 4.6: Visualisation of sample solutions on 7x7 mesh network.

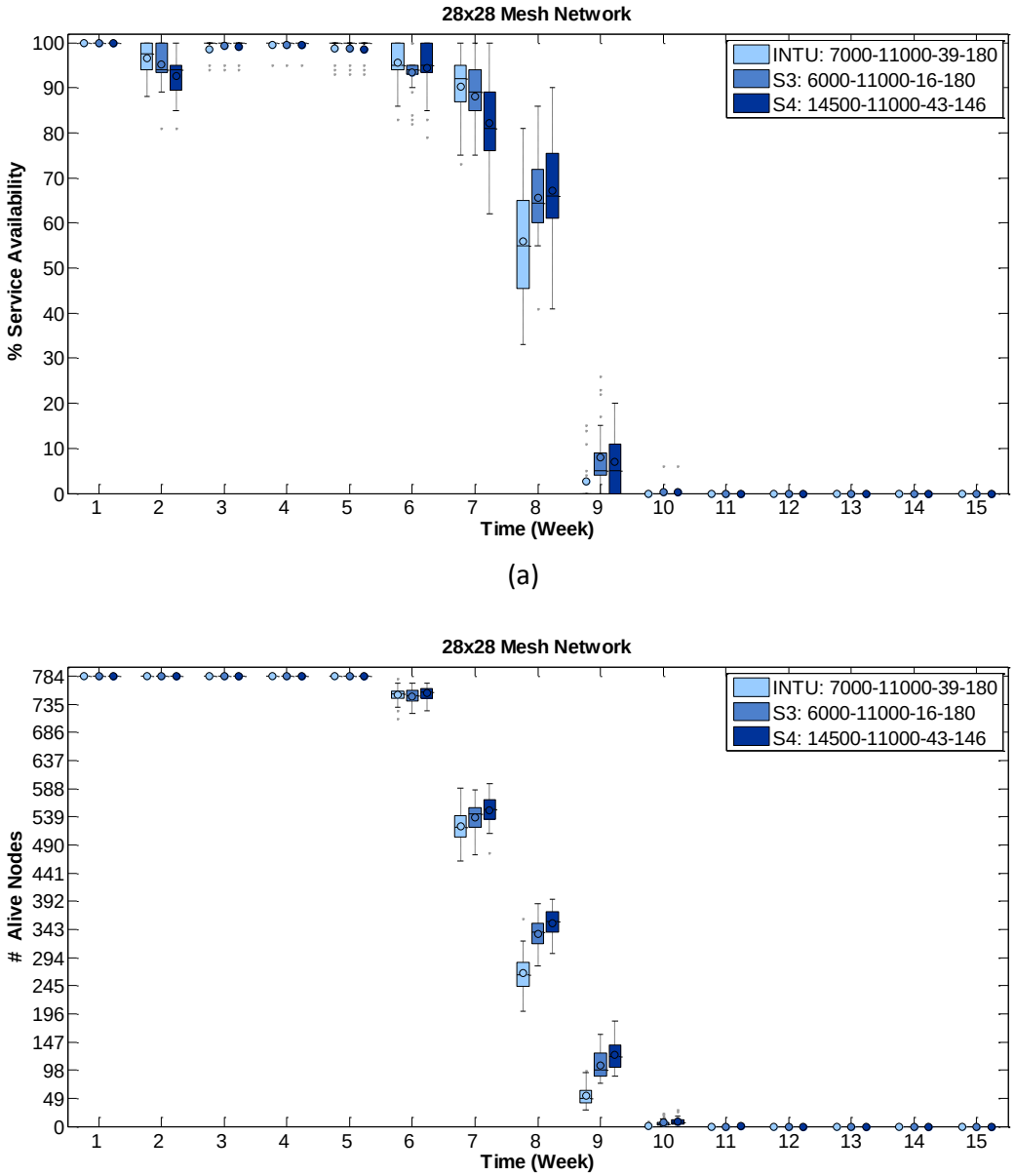


Figure 4.7: Visualisation of sample solutions on 28x28 mesh network.

Analysing the effectiveness of parameter selection tool over intuitive, manually tuned parameters In Figure 4.6 shows three sample solutions on 7x7 mesh network, and whereas Figure 4.7 illustrates three sample solutions on 28x28 mesh network. For both networks, INTU represent the intuitive parameters that we showed in the previous chapter, where S2, S3, S4, S4 are encountered successful solutions by LS scenario on *SA* algorithm. Box plots illustrates the set of results gained from the 30 repetition that satisfies the statistical significance, whereas the

circles inside the box plots represent the mean values. Although the encountered parameter sets do not outperform INTU, the experiments show that 1) the *SA* process is capable of producing parameters achieving performance close to our intuitive parameter selection, which is a strong baseline as it was extensively tuned during the previous work; 2) our intuitive parameter selection is near-optimal.

4.3.2 Principle Component Analysis

Knowing the biological background and how the PS algorithm works, we suspect that not all the parameters have an equal effect on the results. In order to understand the importance of each parameter, we have decided to apply *Principle Component Analysis (PCA)*. The main purpose of PCA is to maximise the variance of a linear combination of the variables in order to scale and rank them. PCA is an effective tool that performs dimensionality reduction in which the original data is projected to the lower dimension spanned by leading eigenvectors of the covariance matrix of data [148]. We used the data that we gathered from SA (results of over 880 solutions), and together with the results of randomly selected parameters (results 300 solutions) to ensure that the results of PCA are not only based on the guided search metaheuristics but also contains some samples to cover the entire search space. Table 4.3 illustrates the impact of the four parameters of PS and highlights that only T_{DECAY} and T_{QN} play a role in determining the outcome of PS. The other two parameters depend on T_{DECAY} and T_{QN}.

Table 4.3: PCA on PS.

Parameters	PCA results
T_{DECAY}	4.1285
T_{QN}	0.2564
$threshold_{QN}$	0.0000
$QN_{initial}$	0.0000

4.3.3 Applying Parameter Selection Tool Onto New Schemas

This section presents the second set of experimental results. The goal of this set of experiments is to show the effectiveness of the parameter selection onto a new schema and to analyse the effectiveness of the *PS* on the new network schemes. This section is divided into two as experimental design in Section 4.3.3.1 and experimental results in Section 4.3.3.2

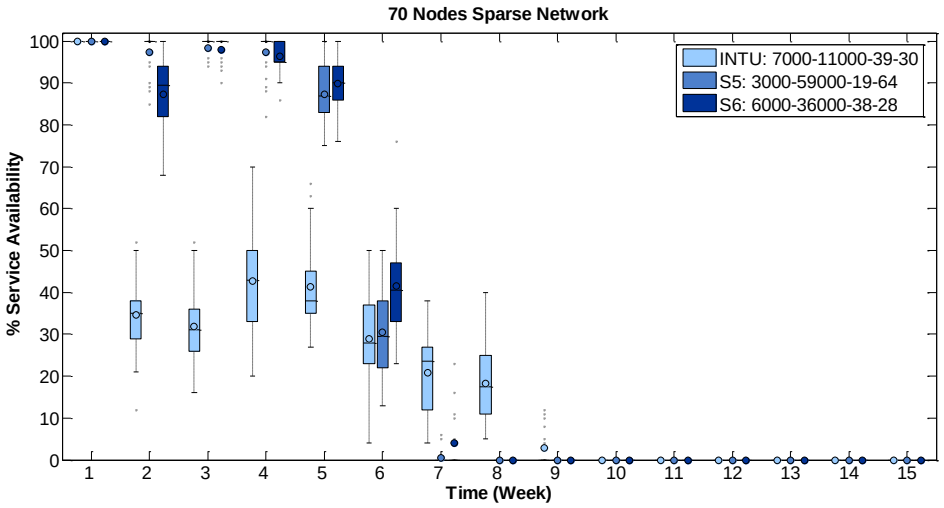
4.3.3.1 Experimental Setup

For our second set of experiments, we apply *PS* algorithm on three different sparse network topologies with 70, 200 and 700 nodes in a 30m x 30m grid as a test case, similar to [169]. We decide to work on sparse network topologies to provide more of a challenge for the algorithm compared to the original regular grid topology. In a mesh network, all the nodes are equally spaced and each node is connected to its adjacent neighbours. Sparse networks, however, are not evenly distributed and each node may have a different number of neighbours. The transmission range of network elements are selected as 6m, 5m and 3m respectively to establish a connected network. In order to specify the minimum transmission range that ensure connectivity, we use the topology control on sparse networks by Santi [157].

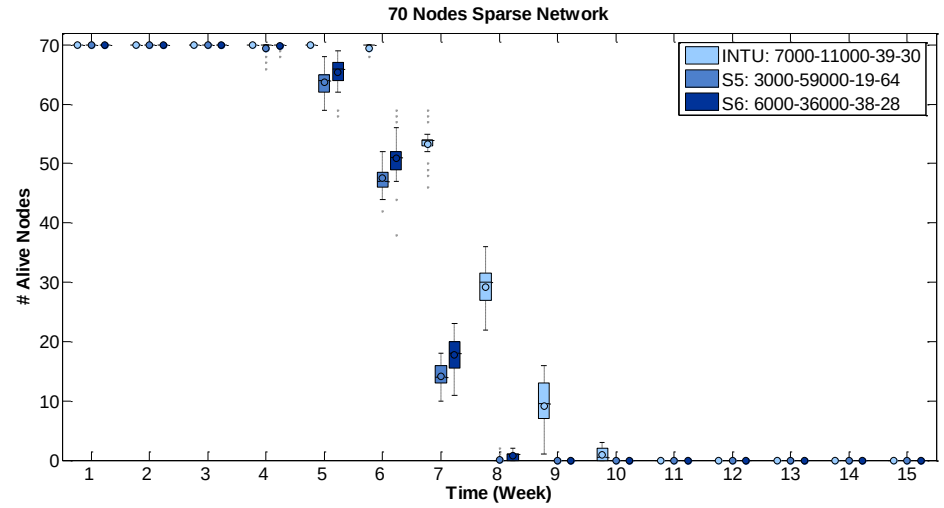
Moreover, among the three *SA* scenarios (WS, BF and LS) the results presented in this section uses only LS scenario as it is the quickest scenario to evolve with highest total service availability. In addition to the LS scenario used in Section 4.3.1, we modified the step size according to the *PCA* results. More explicitly, we used *PCA* results to *guide* the *SA* search metaheuristic towards achieving the fittest solutions by modifying the step size of the LS scenario. The shortcomings of the *PCA* results shows that T_{DECAY} and T_{QN} plays an important role on the performance of the *PS*, whereas $threshold_{QN}$ and $QN_{initial}$ varies based on the values of the T_{DECAY} and T_{QN}. Based on this information, we increased the neighbourhood generation by allowing more modifications on the key parameters (T_{DECAY} and T_{QN}) without changing the other two parameter's neighbourhood generation. Therefore, the modified LS allow us to explore *PS* search space in more effective way. In the next section, we present the experimental results of the search-based parameter

tuning.

4.3.3.2 Experimental Results



(a)



(b)

Figure 4.8: Visualisation of sample solutions on sparse network with 70 nodes.

Figure 4.8 illustrates the experimental results on the new network configurations and compares the intuitive parameter set, INTU with SA suggestions of successful solutions that provides high network performance, S5 and S6. Set of parameters presented in INTU are the intuitive parameters that we used for 28x28 mesh network.

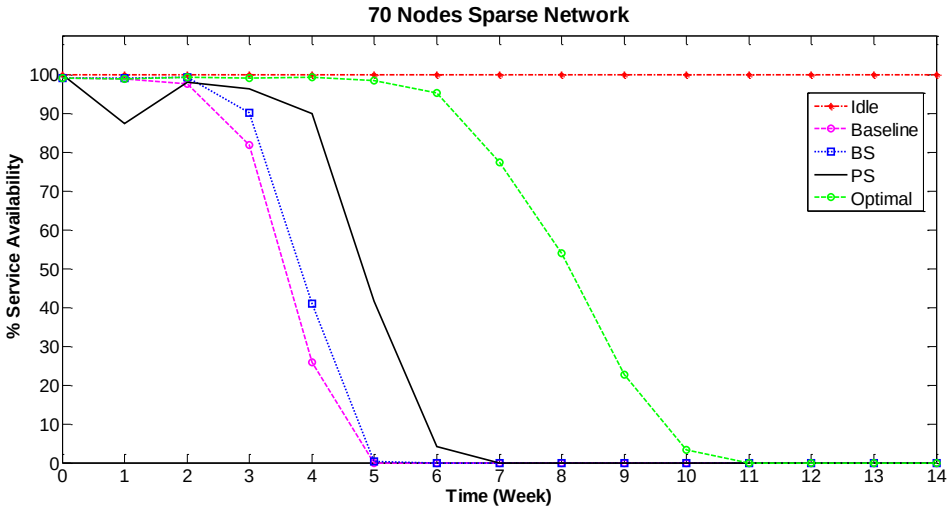
Instead of using parameters used for 4x4 or 7x7 mesh topologies, we used the set of parameters used in 28x28 as it is the largest network we intuitively tuned, and we thought this set of parameters may give good result. Although INTU performs well with a known topology (as shown in the previous section), it performs badly with the new network configuration as Figure 4.8 illustrates. On the other side, both S5 and S6 deliver some service availability until the 7th week when the network nodes run out of energy in both S5 and S6. In Figure 4.8 (b), the number of alive nodes for INTU outperforms, where the S5 and S6 does not improve the energy consumption as much, due to the high weekly resource usage to increase the service availability. This shows that the S6 parameter set balances network load better than S5 parameter set.

After tuning the parameters of sparse network with 70 nodes, we illustrate the effectiveness of the *PS* technique by showing the effects of the Baseline, BS and Optimal scenarios like we presented in the previous chapter. As shown in Figure 4.9, *PS* outperforms both Baseline and BS scenarios. Readers can easily notice that Optimal scenario does not perform as well as 4x4, 7x7 and 28x28 mesh topologies that we presented in Chapter 3.4.3. This occurs as a result of the use of the large sensor field (30mx30m) with a few sensor nodes. Although network performance is limited with 70 nodes in a large area, *PS* performs and balances the network load over the limited network resources. In the rest of this chapter, we introduce denser networks by adding more sensor nodes in the same sensor filed (30mx30m) to see the effects of *PS* technique in various density levels of sparse network topologies.

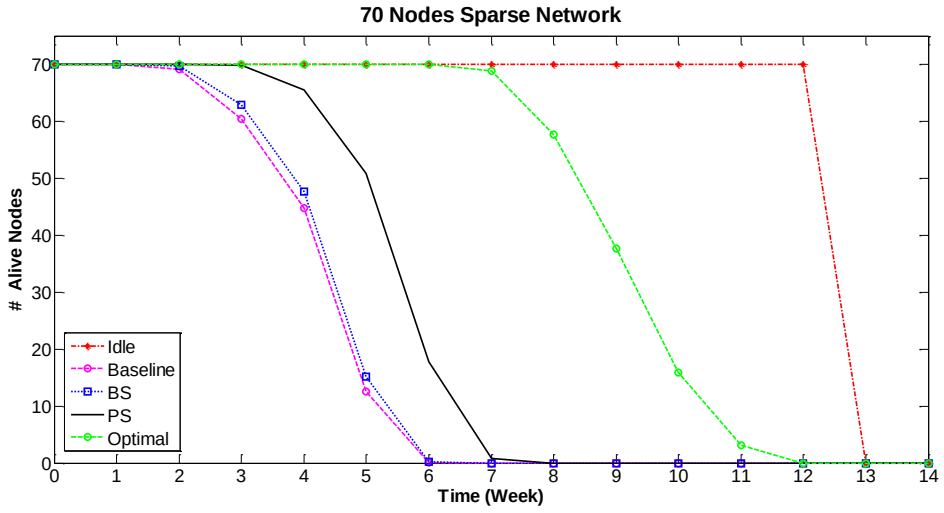
Figure 4.10 illustrates the experimental results on a sparse network configuration with 200 nodes and compares the intuitive parameter set, INTU, with SA suggestions of successful solutions that provides high network performance, S7 and S8. The set of parameters presented INTU is the same with Figure 4.8 and 28x28 mesh network as we explained previously. Figure 4.10, S7 and S8 performs better than INTU as expected. SA based parameter tuning balances the network load over the sensor nodes that increases the the network performance in terms of service availability and extends the network lifetime. In the S7, percentage of service availability drops dramatically at the end of 4th week which we have not experienced before. This

occurs as a result of network stabilisation, similar to what is shown at the end of 2nd week. As T_{QN} is a very large number (49000), network stabilisation happens for the second time in this instance unlike any other experiments.

Figure 4.11 demonstrates the effectiveness of the *PS* technique by showing the effects of the Baseline, BS and Optimal scenarios. The Optimal scenario performs better than the sparse network with 70 nodes as presented in Figure 4.9. The denser the network is, better the Optimal scenario performs because there are more



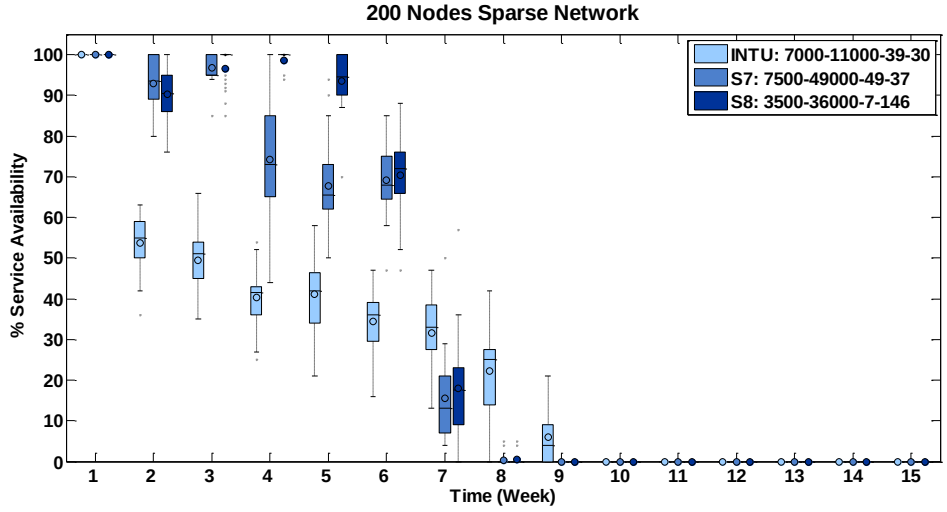
(a)



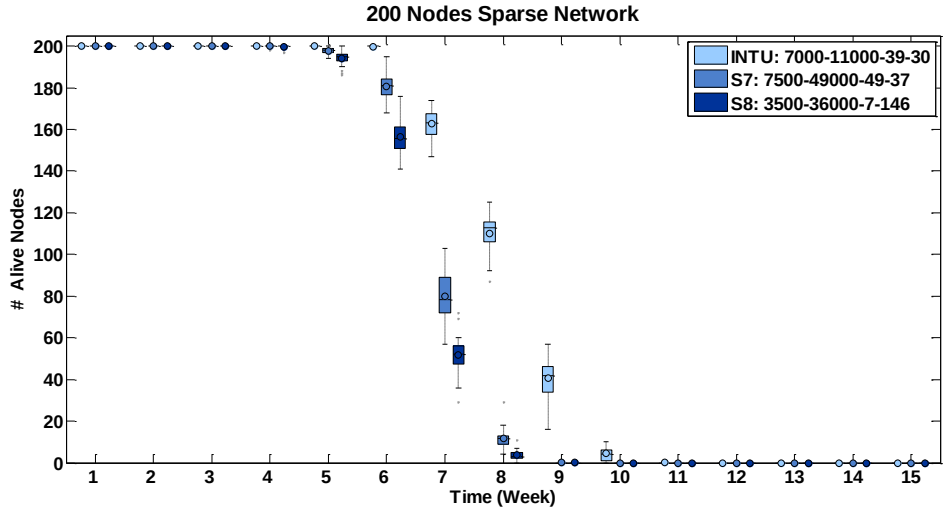
(b)

Figure 4.9: Experimental results: effects of PS algorithm on sparse network topology with 70 nodes showing (a) % service availability, (b) # alive nodes.

nodes available to execute tasks. Although network lifetime is the same with sparse network with 70 nodes, the total service availability (area shown under the green curve) dramatically improves. As we increase the number of the sensor nodes in the same sensor area, we increase the network resources and this affects the network performance positively. The Optimal scenario illustrates this hypothesis very clearly. *PS* outperforms to Baseline and BS scenarios by reducing the redundant use of the network resources and balances the network performance. On the other side, Baseline and BS scenarios runs out of energy by the end of 6th week and both do



(a)



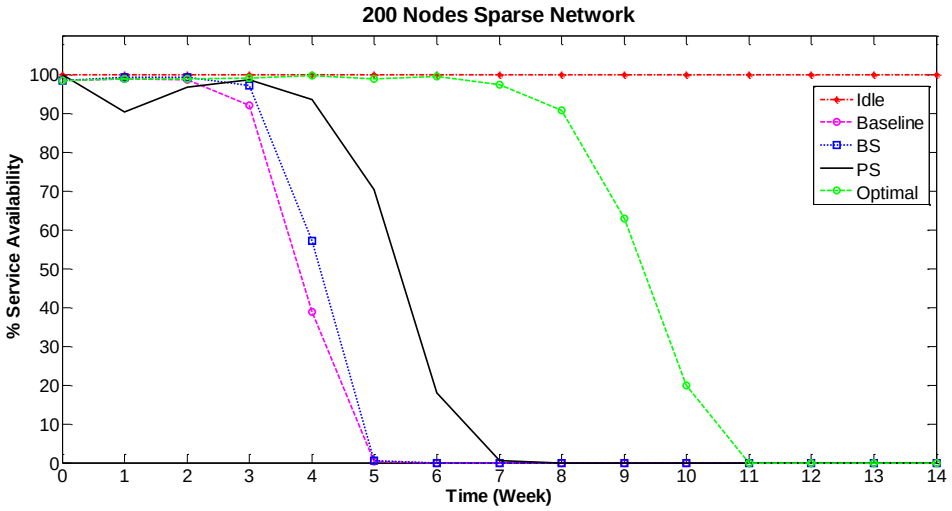
(b)

Figure 4.10: Visualisation of sample solutions on sparse network with 200 nodes.

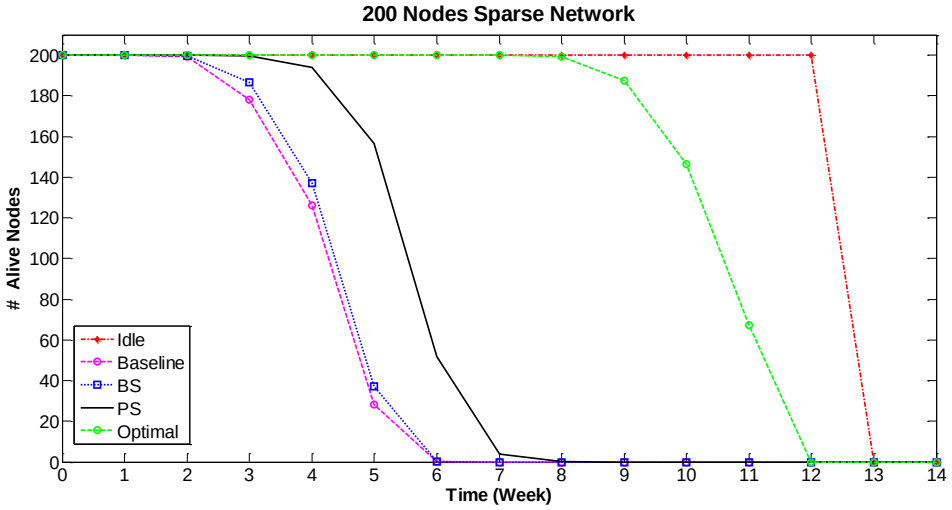
not improve the service availability after the 5th week.

Figure 4.12 illustrates the SA-based parameter tuning on a sparse network topology with 700 nodes. Three sets of solutions are compared to each other: INTU is the intuitive one we used previously, whereas S9 and S10 are SA suggested set of parameters. Although S9 and S10 are very different to each other, both perform similarly.

Comparisons between Baseline, BS, Optimal and *PS* scenarios are presented in



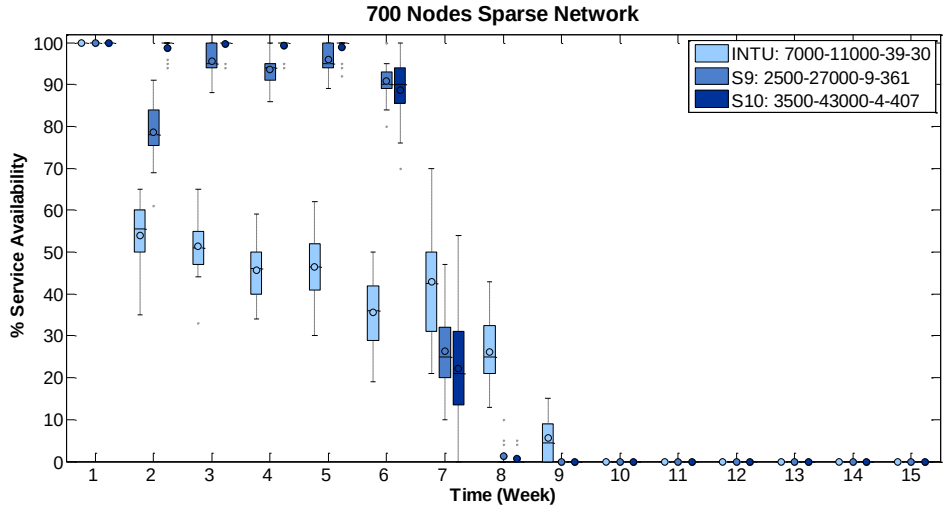
(a)



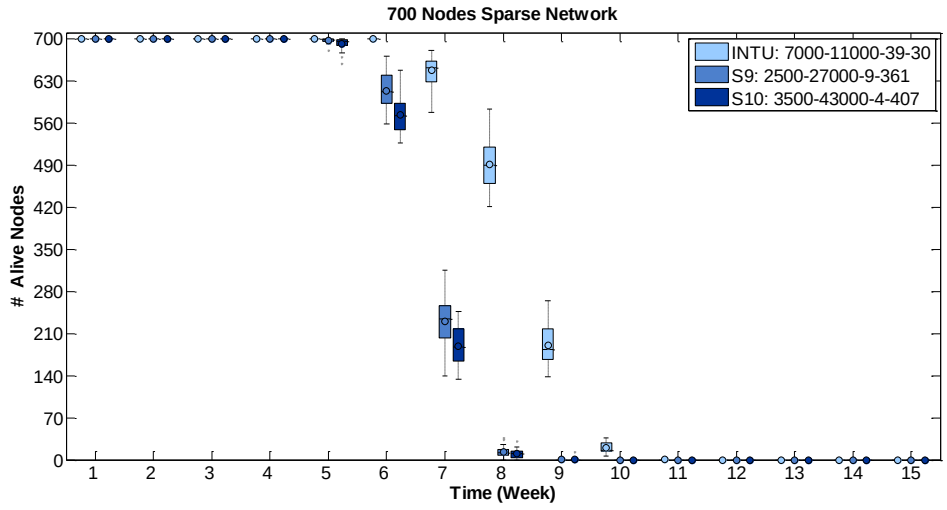
(b)

Figure 4.11: Experimental results: effects of PS algorithm on sparse network topology with 200 nodes showing (a) % service availability, (b) # alive nodes.

the Figure 4.13 to show the effectiveness of the and *PS* technique. The improvements on the optimal scenario are more dramatic for the 700 node network as opposed to 70 or 200. This is due to it being more densely populated, enabling more resources to be available to provide service. Similarly, *PS* technique also improves the service availability more in denser networks. In Figure 4.9, the total service availability (area under the black curve) is the minimum. Total service availability increase in Figure 4.10 slightly and achieves the maximum on Figure 4.12 among three sparse network configurations. As the number of the nodes increases, the total service availability

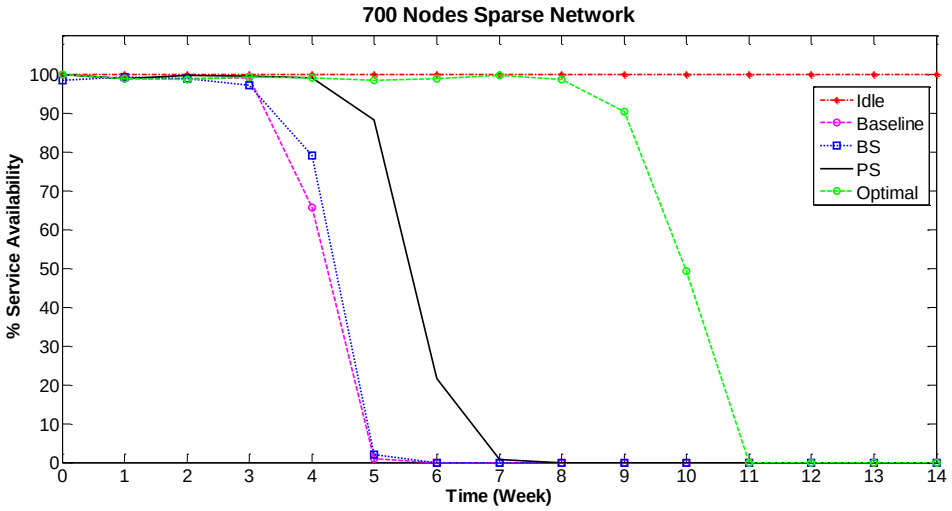


(a)

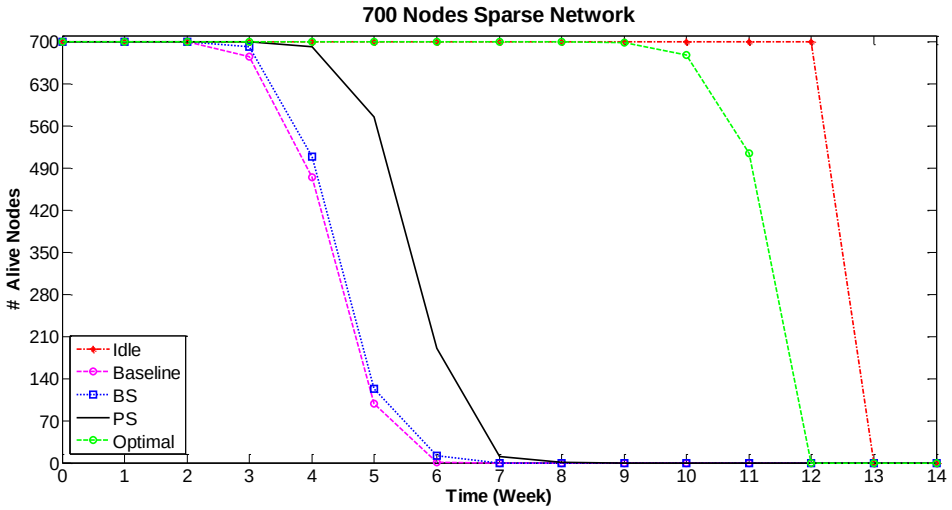


(b)

Figure 4.12: Visualisation of sample solutions on sparse network with 700 nodes.



(a)



(b)

Figure 4.13: Experimental results: effects of PS algorithm on sparse network topology with 700 nodes showing (a) % service availability, (b) # alive nodes.

increases and in all three network topologies *PS* balances the resource usage and increases the network performance compared to Baseline and BS scenarios.

4.4 Summary

This chapter had two major goals. Firstly, we aimed to automate the parameter tuning process needed to optimise *PS* for a given scenario, balancing service availability

against energy consumption. The second goal of this chapter was accelerating the evaluation method towards finding fit solutions quicker by evolving the simulation infrastructure. We implemented a less accurate but very fast system level simulator, SuperFast, and analysed its accuracy and execution time as compared to our previous simulator. The advantages and disadvantages of these two simulators have been highlighted with the experimental results.

The need for systematic parameter selection on our load balancing technique to solve the trade-off between service availability and energy consumption in WSNs leads us to inspect search metaheuristics. We presented a search-based technique that uses *SA* to automate parameter tuning for our PS algorithm. In the first set of experiments, we have analysed the effectiveness of *Simulated Annealing* by creating three different scenarios on the known network topology. The experimental results verify that there is more than one near-optimal solution. Based on the first set of experimental results, *Principle Component Analyses* has been applied to inspect the importance of each parameter of the parameter set. The results show that only two of the parameters are important for the PS technique. The search technique has been modified accordingly so that *SA* converges faster and is applied onto an untested network topology. The second set of experimental results compare our intuitive parameter set and sets of parameters found out by the *SA* on three different sparse network topologies. SA-based parameter atomisation is applied on sparse networks with 70, 200 and 700 nodes. The tuned parameters by *SA* outperform the our manual 'intuitive' set of parameters and consequently increase the network performance. Moreover, performance efficiency of *PS* are shown on these three sparse networks and compared with Baseline, BS and Optimal scenarios. In all scenarios, *PS* outperforms and manages to balance the effective use of the network resources.

In short, *PS* parameter atomisation is successfully implemented and a parameter selection tool has been created to tune the parameters of *PS* algorithm to any given network schema and size. A user looking to deploy *PS* on their WSN could tailor the simulator to their node specifications, define their topology and estimated load, and use the tool to generate the optimal set of parameters for *PS*. We showed

that effective network coverage increases the network performance. The way we illustrated the network coverage in this chapter was to increase the network density. In the next chapter, we will target to increase the network coverage by not only adding more network nodes but by introducing some mobile network elements.

Chapter 5

Using Mobile Robotic Agents to Increase Service Availability and Extend Network Lifetime on WSRNs

We have explained the resource constraints, characteristics and nature of WSNs in the previous chapters. In this chapter we extend our perception of load balancing into WSRNs to increase network performance in terms of service availability, network coverage and lifetime using different domains in addition to WSNs.

Wireless Sensor and Robot Networks (WSRNs) are heterogeneous collections of sensor nodes and robotic vehicles that communicate wirelessly. In the last decade many research studies have attempted to address the challenging trade-offs of Wireless Sensor Networks (WSNs) that arise due to their resource limitations, however, much work is still to be done. A recent trend is to merge different subclasses of cyber-physical systems together to achieve the desired performance goals by benefiting from their heterogeneity. This chapter presents a bio-inspired robot guidance technique that is used to improve network coverage, service availability and energy consumption of WSNs using robotic agents on vehicles.

The main goal of this research is to *effectively* guide the robots to increase the network coverage, which will directly increase the service availability and extend

the network performance. *Effective* network coverage in this research is defined as achieving the highest service availability by moving less. To achieve the desired effective network coverage, we have extended our *PS* technique to guide robots towards the areas of the sensor field where the sensor nodes have run out of battery and are unable to provide service. Some of the performance metrics are reused in this chapter from Chapter 3 and only one of the performance metrics are particularly defined for mobile robotics agents. To remind the reader of the metrics that we use in Chapter 3, we now relist them and also define the performance metrics that we particularly use for the robotic agents as:

- service availability: the number of services that are successfully completed divided by the total number of requested services within a period of time;
- total energy consumption: the sum of communication and computation energy consumption within a period of time;
- total distance travelled by robots: the total distance that a robot has travelled during the simulation.

We have defined a *service* in Chapter 3 as the composition of a number of inter-communicating tasks. The readers may remember that a service is considered to be successfully detected only if all of its tasks are executed by the nodes. Here in this chapter, we newly introduce the minimum total distance travelled by robots. Ideally, it is desired to achieve the minimum total distance travelled with highest service availability indicates efficient guiding. Therefore, we target maximising service availability whilst minimising the total energy dissipation together with the shortest total distance travelled by robots.

Chapter Contributions The contributions of this chapter are listed below:

- The definition of a lightweight, *robot guidance* algorithm based on *PS* to increase network coverage and load balancing in WSRNs.
- The integration of robotic elements into the system-level infrastructure to evaluate the long- term effects of the PS algorithm on WSRNs.

- The effects of the event distribution on the *robot guidance* algorithm.

Chapter Structure Section 5.1 describes our *Pheromone Signalling-based Load Balancing (PS) Robot Guidance* resource management algorithm. We explore the performance consequences of the Pheromone Signalling-based Load Balancing (PS) Robot Guidance on an abstract level simulator to present a system perspective. Section 5.2 illustrates how does the robotic agents is integrated to our system-level simulation model infrastructure that we explained in Chapter 3. The effectiveness of the algorithm is evaluated with different network topologies and investigated on various scenarios. Section 5.3 demonstrates the effectiveness of *PS-based Robot Guidance* algorithm on simulated experimental results on sparse topologies validate that robot guidance based on *PS* increases network lifetime using a uniformly distributed events in Section 5.3.3 and using a non-uniformly distributed events in Section 5.3.3.

5.1 Pheromone Signalling-based Load Balancing (PS) Robot Guidance Technique

This section explains the integration of robot guidance into the *PS* technique. In the *PS* technique, the level of pheromone indicates the resource usage in a particular area of the network. Areas in the sensor field that have lower level of pheromone at a given time demonstrate less resource usage as opposed to other parts of the network. Less resource usage may due to: (1) no events occurring in the neighbourhood; (2) events occur in the neighbourhood that are not in the detection range of particular node; or (3) nodes are already out of energy in that part of the sensor field. In our previous work, we apply our *PS* load balancing solution to cases (1) and (2), and show that by distributing the network load evenly we balance service availability and energy consumption. In this paper, we propose a robot guidance algorithm based on PS to solve case (3) by guiding robots into the areas where the sensor nodes are already out of energy. Incorporating additional robotic agents on vehicles and guiding the agents based on *PS* not only balances the network load (1),(2) but also improves the network coverage (3).

The three existing cycles of *PS* are explained in Chapter 3 as the differentiation, propagation and decay cycles. In addition to these three cycles that *PS* applies, we have extended *PS* with the robot behaviour to solve the given problem above (3). We now describe the robot behaviour algorithm and define how it occurs. The standard differentiation, propagation and decay cycles apply only on the sensor nodes, whereas the robot behaviour only occurs on robotic agents.

While the sensor nodes are deciding whether to provide a service or not to reduce the computational redundancy, it is essential to underline that the robotic agents are willing to provide service at all times to increase the network coverage. As with sensor nodes, robotic agents can receive pheromone from QNs if they are in communication range. Robotic agents act as QNs – they execute all tasks assigned to them. However, robotic agents do not propagate pheromone to other nodes in their communication range, so as to stop the robots interfering with the standard pheromone signalling mechanism. Listing 5.1 presents the robot behaviour in pseudocode. A robotic agent moves under two conditions: 1) if a robotic agent receives pheromone from a QN; 2) if the agent arrives at its destination without receiving any pheromone.

Listing 5.1: Pseudocode of the PS-based robot guidance algorithm.

```
1 if(pheromone received)
2   PS–guided moving decision
3 else if (arrived at destination without receiving pheromone)
4   randomly move
5 else
6   broadcast communication link request
7   establish local communication links
```

If a robotic agent receives pheromone it makes a moving decision and selects a target destination in the opposite direction of the received pheromone based on *PS*. The moving decision of the robotic agent is based on vector addition and its pseudo code appears in Listing 5.2. Given the mathematical formulation in the pseudocode and assuming all the network elements (sensor nodes and robotics agents) know their location as x and y coordinates, we calculate the angle of the received pheromone

with the use of the sender's x and y coordinates. To do this, we resolve the horizontal and vertical components based on the amount of received pheromone level, h_i, and the coordinates of the sensor node. In order to find the magnitude, we sum up all the horizontal and vertical components. In order to determine the direction of the magnitude, we take arctangent of the magnitude and resolve x and y coordinates. This process happens on-demand as the robotic agents receive pheromone from QNs.

Listing 5.2: Robot Moving Decision

```
1  if (h_i > 0)
2   for all the received pheromones (p) of the node
3      diff_X = p_Sender_X − currentCoordinate_X
4      diff_Y = p_Sender_Y − currentCoordinate_Y
5      θ = ArcTangentQuadrant(diff_Y, diff_X)
6      component_X = p.hd ∗ cos θ
7      component_Y = p.hd ∗ sin θ
8      Sum_X+ = component_X
9      Sum_Y+ = component_Y
10 magnitude = √(Sum_X^2 + Sum_Y^2)
11 θ_destination = ArcTangentQuadrant(Sum_Y, Sum_X)
12 apply 180 degrees shift to θ_destination
13 clear all received pheromones
```

If a robotic agent does not receive any pheromone by the time it arrives to its destination then the robotic agents picks a new destination at random and moves towards the new destination to increase the service availability by helping the sensor nodes. If a robotic agent does not receive any pheromone and has not yet arrived at its destination that means it is currently moving. In this case, the robotic agent continues to move towards its calculated destination whilst periodically broadcasting communication requests and updates its communication links nearby nodes.

Summary

In this section we have presented a robot guidance algorithm that builds upon our bio-inspired *PS* algorithm to address the trade-off between the service availability and energy consumption of WSRNs. We extended the *PS* algorithm for robotic

agents to increase the network coverage using the higher resource capacities of robots on vehicles. We aim to guide the robotic agents effectively into the areas of the sensor field where sensor nodes are not able to provide service so as to increase the network coverage and therefore improve the service availability and network lifetime. In the rest of this chapter, we explain the extended evaluation infrastructure and illustrate the experimental results of the *PS robotic guidance* technique.

5.2 Integration of Mobile Robot Agents into *Fast*

We described our system-level simulation infrastructure *Fast* in Chapter 3, Section 3.3.3. In this chapter, we *extend* our simulation infrastructure *Fast* as such to evaluate the performance efficiency of using mobile robotic agents.

In WSRN, three-tier system model is designed to represent network components, the services that run over it and the function that assign services to network elements as follows.

The platform model in WSRNs, $PM = (R, N, L)$ consists of a set of R robots, a set of N nodes and a set L of bidirectional wireless links between neighbouring nodes and robots. Each robotic agent $r_m \epsilon$ R, is the tuple:

$$r_m =< m_m, bc_m, idr_m, cdr_m, wcdr_m, mdr_m, loc_m, tr_m >$$

whereas each node $n_m \epsilon$ N in WSRNs, is the tuple:

$$n_m =< m_m, bc_m, idr_m, cdr_m, wcdr_m, loc_m, tr_m >$$

m_m memory capacity in bytes;

bc_m battery capacity in mAh;

idr_m battery discharge rate in μ as in idle mode;

cdr_m battery discharge rate in μ as when performing a computation;

$wcdr_m$ discharge rate in μ as when transmitting a byte of data over its wireless interface;

tr_m transmission range that specify the communication distance in meter with its neighbouring network element;

loc_m location in x and y coordinate system;

As the network topology is represented by the set of links $l_{mn} \epsilon$ L, the cost of multi-hop transmission of each communication c_{ij} can also be taken into account if the routing algorithm used in the network is known. Communication links establish between the network elements (robot and sensor, sensor and sensor, robot and robot) based on the specified transmission range that varies according to the applied network topology.

There are no applied changes in the application model: definition of the service and its mathematical representation in WSNS; the information given in Chapter 3, Section 3.3.3 is applied to WSRNs without any changes.

In this chapter, the mapping process is applied in a slightly different way. We have not changed the definition of the mapping process, and used it as we defined in Chapter 3 which is a function from the application domain to the platform co-domain. However, as the entire idea of introducing robotic agents is to force them to help sensor nodes and increase the network coverage, the mapping is *additionally* applied to the robotic agents *if* there are some existing robotic agents where the events occur. This means, that mapping is applied to sensor nodes as we did previously in Chapter 3 whereas the mapping process is extended for the robotic agents given that the robotic agents are physically in events range.

Figure 5.1 illustrate the UML sequence diagram of how robotic agents are integrated to the sensor nodes and the application domain using *Fast*. The simulation environment does not change much: it still is an abstract simulator that controls event-driven nature by the JavaSim library [109] with multi-threaded nature as shown in the Figure 5.1.

5.3 Experimental Results

This section will start with the definition of the experimental setup in Section 5.3.1 and will continue presenting the experimental results on the effects of PS

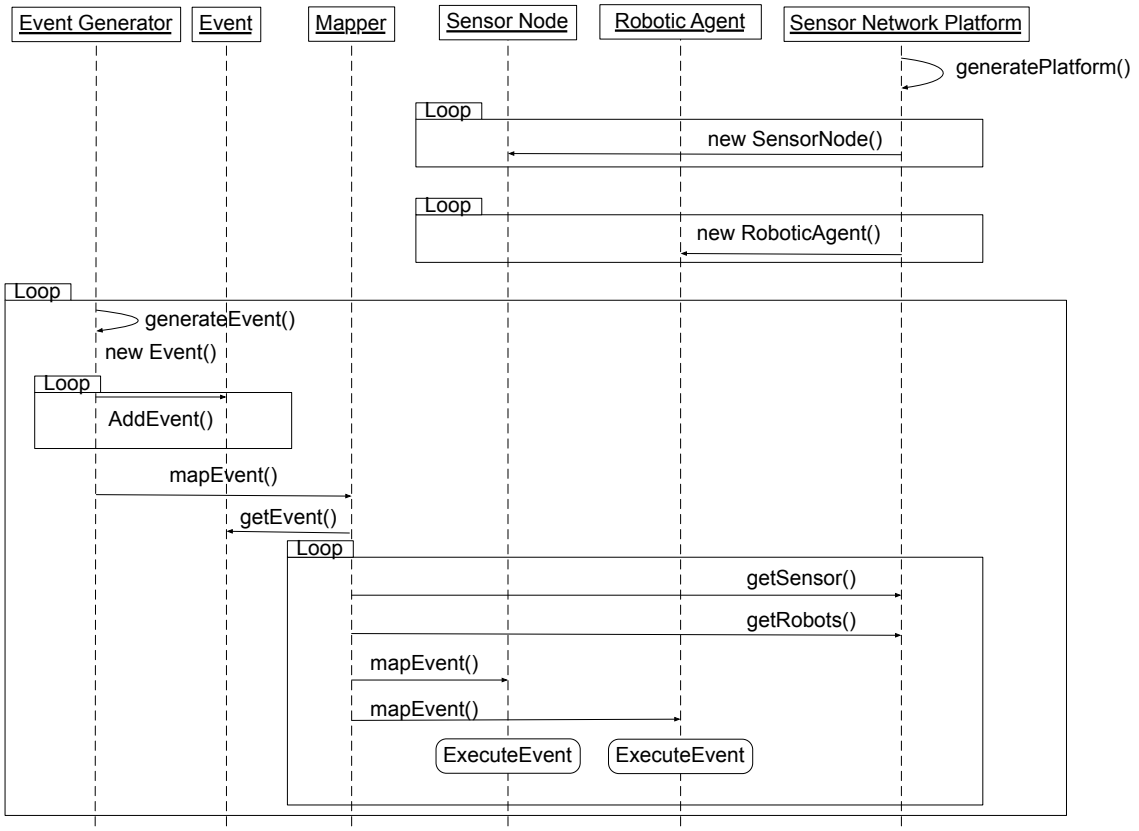


Figure 5.1: Execution sequence of system-level simulator applied on WSRNs.

Robotic Guidance technique on uniformly distributed events in Section 5.3.3 and non-uniformly distributed events in Section 5.3.3 to show the effectiveness of PS Robotic Guidance technique on event distribution.

5.3.1 Experimental Setup

We have created three other scenarios to compare with *PS-based Robot Guidance Technique*. Some of the scenarios are the same with Chapter 3 Section and reused in this chapter. To remind the scenarios to the reader, we relist them as follows:

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

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

Objective It is design to show maximum possible network performance by creating a scenario which never can occur in real life.

PS with Random Moving Robotic Agents represents a WSRN that robotic agents move randomly in the sensor field without interacting with the sensors where sensor nodes apply *PS* load balancing technique separately.

Objective It combines basic random moves that robots present and applies *PS* which sensor nodes execute.

PS Robotic Guidance represents a WSRN that robotic agents move according to PS in the sensor field where sensor nodes apply *PS* load balancing technique at the same time in parallel.

Objective It is developed to guide robotic agents to where sensor nodes are not able provide service to increase the network coverage.

5.3.2 Effects of PS Robotic Guidance Technique on Uniformly Distributed Events

In this section, we present the effectiveness of the PS Robotic Guidance technique on uniformly distributed events similar to previous chapters: Chapter 3 and Chapter 4. We use location-aware sensor nodes and mobile robotic agents (i.e. each GPS equipped). As the most significant part of this chapter is the mobile coverage of *PS Robotic Guidance* technique which involves location-awareness, it is worth specifying the type of the distribution and the probability of events occurring in a given location. When events are uniformly distributed every location in the sensor filed has the same probability of the event occurring at it. To contextualise uniformly distributed events we use the same case study that we explain in Chapter 1 Figure 1.1 sound-based bird detection.

In Figure 5.2 the percentage of the detected events and the number of alive nodes are shown for 70 nodes sparse network topology for *Idle*, *PS*, *Optimal*, *PS-Guided*

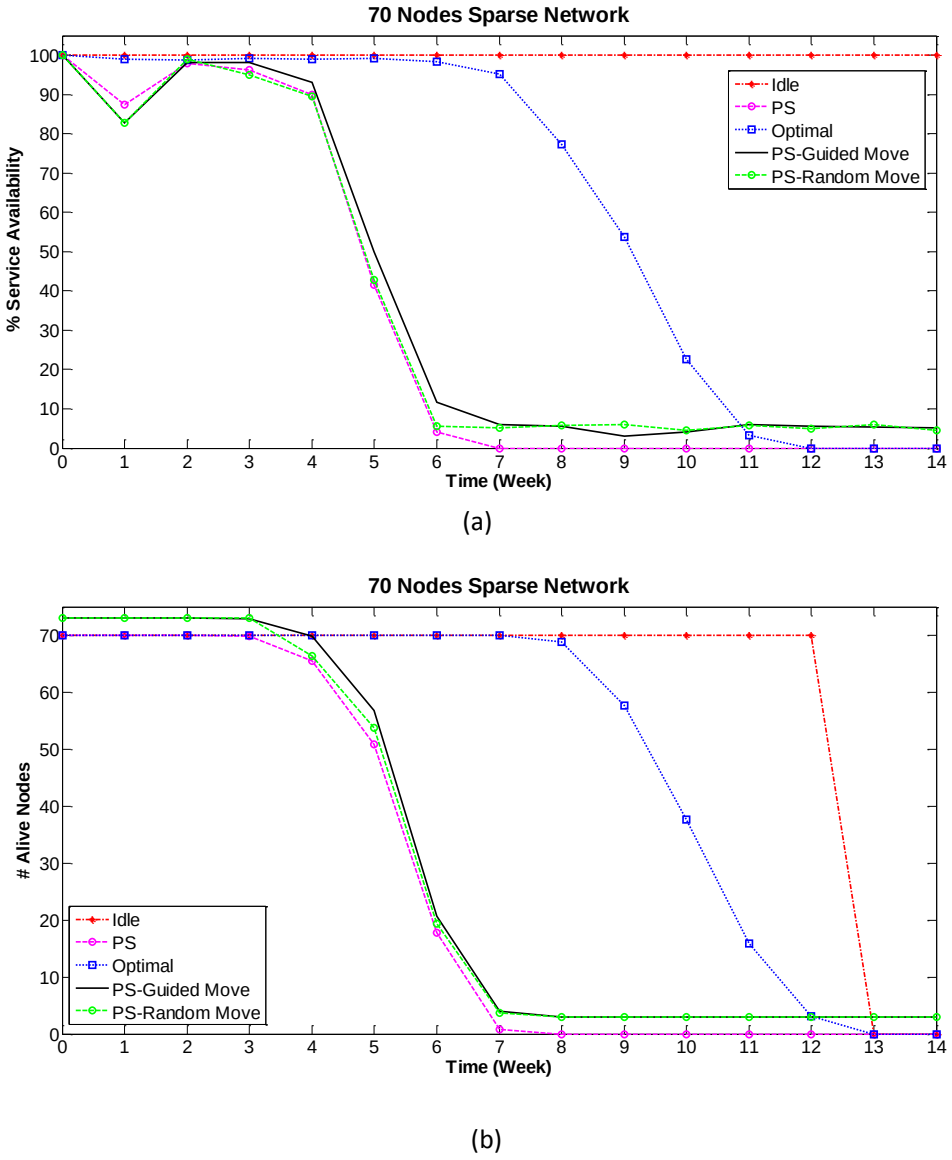


Figure 5.2: Experimental results: effects of *PS Robot Guidance* algorithm on 70 nodes sparse network topology on a uniform event distribution showing (a) % service availability, (b) # alive nodes.

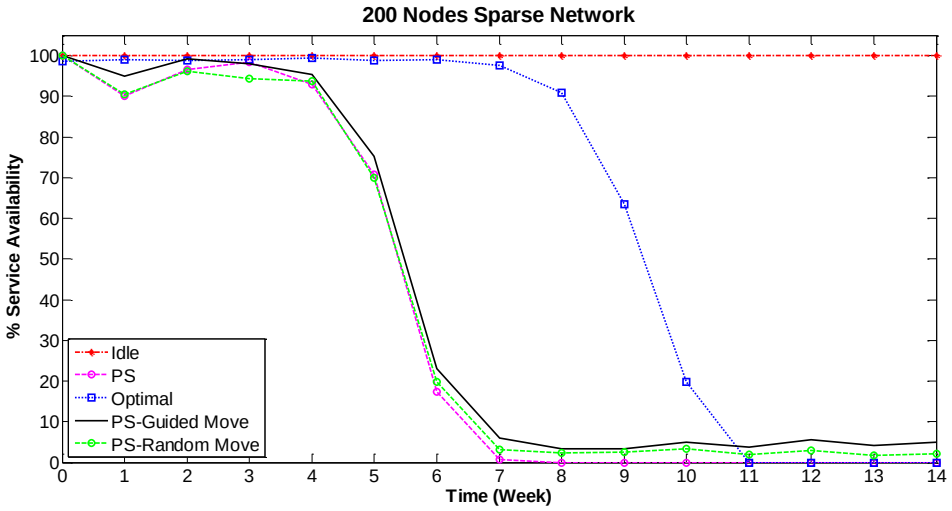
Move and *PS-Random Move* scenarios. A sparse network with 70 nodes in a 30m x 30m grid as a test case [169]. The transmission range of network elements (both sensors and robots) are selected as 6m to establish a connected network. In order to specify the minimum transmission range that ensure connectivity, we use the topology control on sparse networks by Santi [157]. *Idle*, *PS* and *Optimal* scenarios are only applied on the 70 sensor nodes, whereas *PS-Guided Move* and *PS-Random*

Move are applied both sensors and robots in the environment (70 sensors and 3 robots; 73 pieces of hardware device). *PS-Random Move* takes actions based on the given probabilities. As shown in Figure 5.2 (a) and (b), the percentages of service availability and alive nodes are lowest in the *PS* scenario on WSNs (no robotic agents involved), and the highest is *Optimal* scenario. This means, using additional robotic agents on the WSNs is beneficial: robots do not interfere with the sensors, and improve network performance.

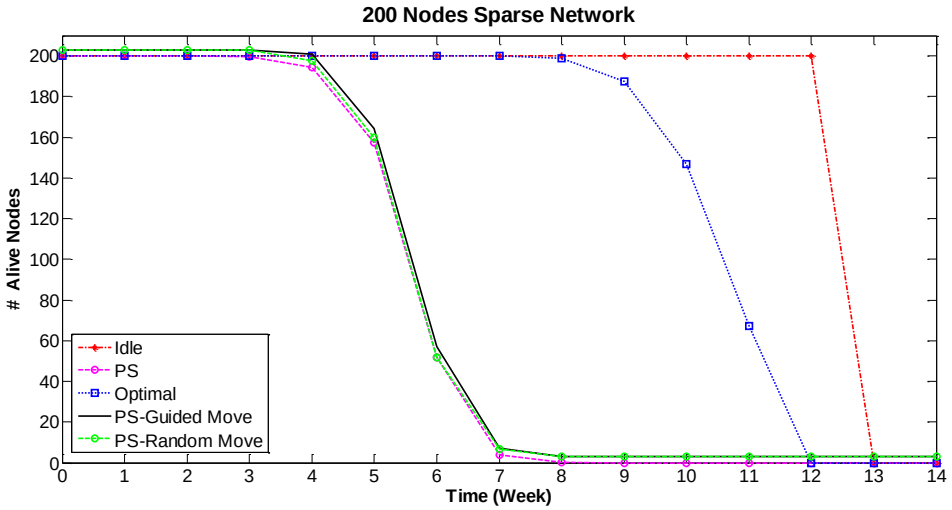
The *PS-Guided Move* scenario performs better, in terms of service availability, than the *PS-Random Move* scenario between week 3 and week 7 as this is the time that nodes begin to run out of energy. Starting from the week 7 there are no alive network nodes in the sensor field in any of the *PS*, *PS-Random Move* and *PS-Guided Move* scenarios, and as can be expected as there is no alive node, we do not observe any advantage of using *PS-Guided Move*.

Similarly, in Figure 5.3, *Idle*, *PS*, *Optimal*, *PS-Guided Move* and *PS-Random Move* scenarios are illustrated on a sparse network with 200 nodes in a 30m x 30m grid is inspected together with 3 robotic agents and the transmission range of network elements (both sensors and robots) are selected as 5m to establish a connected network [157]. In Figure 5.3 *PS-Guided Move* also performs slightly better than *PS-Random Move* between week 4 to week 7. Although the difference between the two scenarios (70 and 200 nodes sparse networks) are not significant, the sparse network with 200 nodes performing better than the sparse network with 70 nodes in terms of service availability. This is due to the fact that the pheromone propagation is more effective in large scale networks and thus is why we claim that *PS* brings more advantages for large scale networks. The affects of the mobile robotic agents, and *PS-Guided Move* supports this claim with minor benefit.

Idle, *PS*, *Optimal*, *PS-Guided Move* and *PS-Random Move* scenarios in Figure 5.4 on a sparse network with 700 nodes in a 30m x 30m grid is inspected together with 3 robotic agents and the transmission range of network elements (both sensors and robots) are selected as 3m to establish a connected network [157]. Although using mobile robotic agents increase the performance in Figure 5.4, the performance difference between no robotic elements (*PS*) and *PS-Guided Move* and *PS-Random*



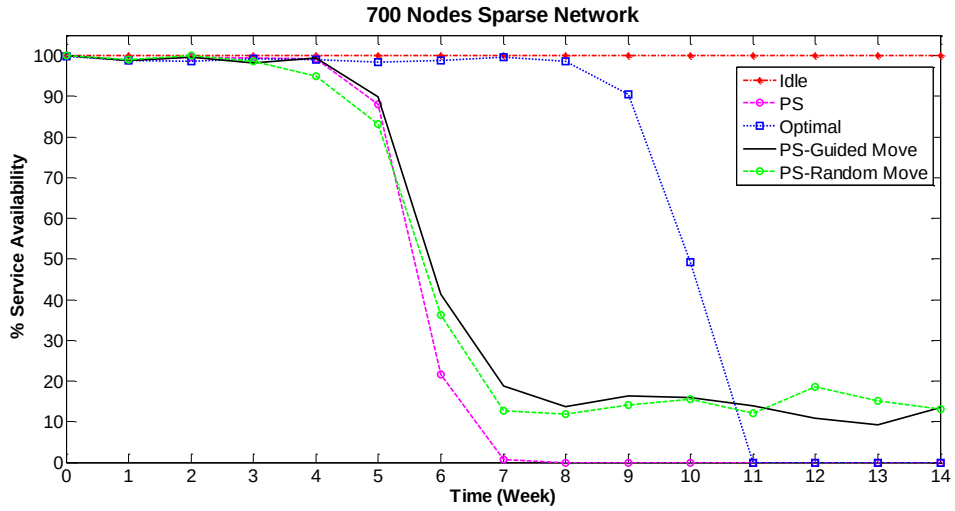
(a)



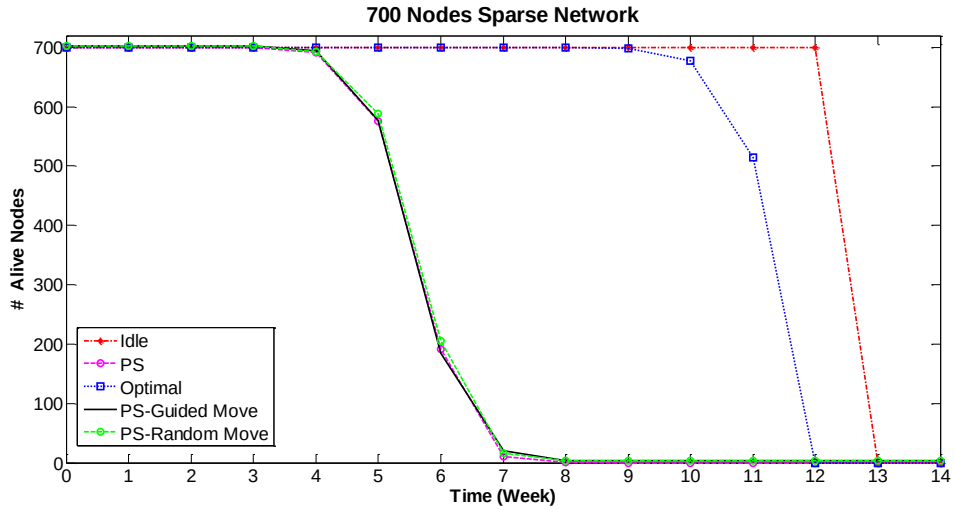
(b)

Figure 5.3: Experimental results: effects of *PS Robot Guidance* algorithm on 200 nodes sparse network topology on a uniform event distribution showing (a) % service availability, (b) # alive nodes.

Move are very minor. We believe, this occurs as a result of high level of pheromone in the environment where there are many nodes unlike the other scenario, and the fact that *PS* algorithm improves the network performance particularly in large scale networks. As a results, although mobile agents increase the network coverage, both *PS-Guided Move* and *PS-Random Move* scenarios do not exhibit major benefits over *PS*.



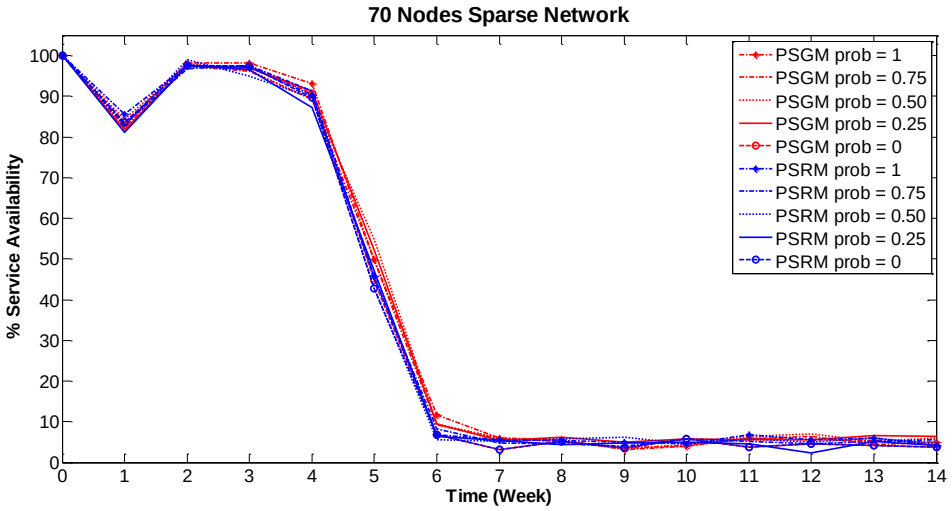
(a)



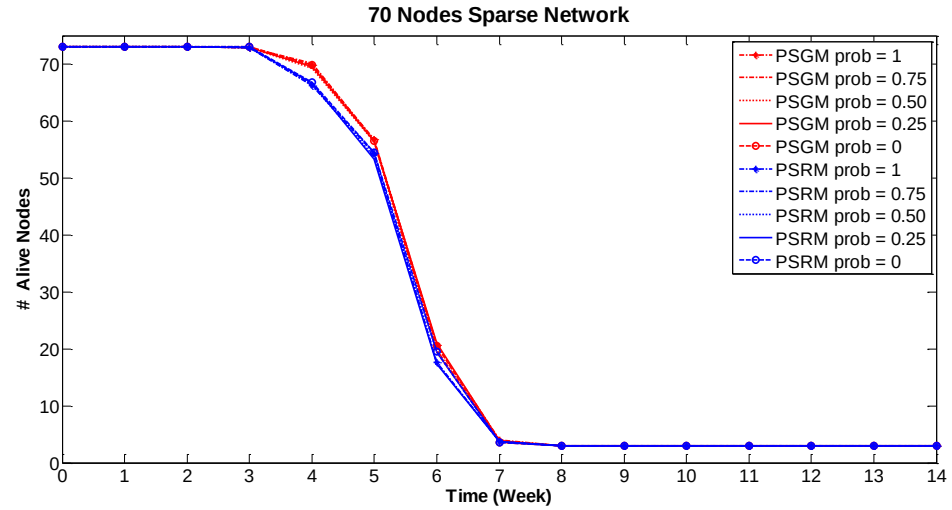
(b)

Figure 5.4: Experimental results: effects of *PS Robot Guidance* algorithm on 700 nodes sparse network topology on a uniform event distribution showing (a) % service availability, (b) # alive nodes.

In Figure 5.5, Figure 5.6 and 5.7, the effects of probabilities on deciding whether to move is illustrated for both the *PS-Guided Move* and *PS-Random Move* scenarios on sparse networks. 'PSGM' represents *PS-Guided Move*, whereas 'PSRM' represents *PS-Random Move* scenarios. Based on the given probabilities, each robot decides to move or not when the moving decision occurs. As we mention in Section 5.1, a robotic agent moves under two conditions: (1) if it receives some pheromone from



(a)

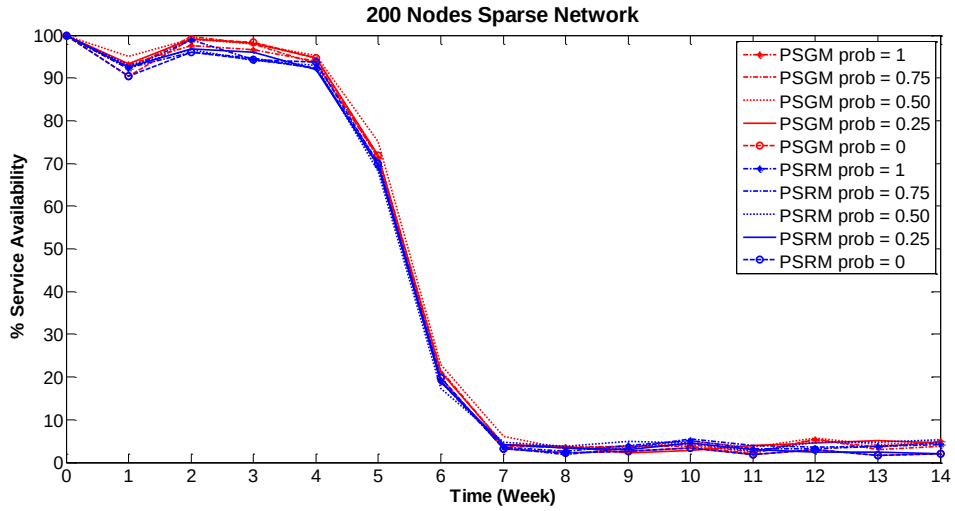


(b)

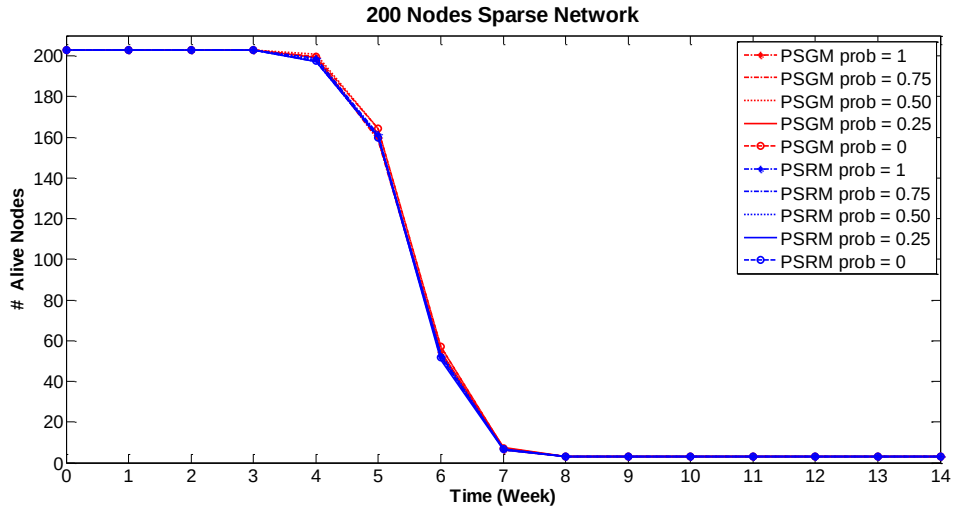
Figure 5.5: Experimental results: effects of *PS Robot Guidance* algorithm on 70 nodes sparse network topology on a uniform event distribution showing various probabilities on move (a) % service availability, (b) # alive nodes.

a QN; or (2) if the agent arrives to its destination without receiving any pheromone. In *PS-Guided Move*, robots move based on *PS Guidance* when (1) occurs, and randomly move when (2) takes place. The probabilistic moves only occur when robots do not receive pheromone, when they are moving randomly.

Given the probabilities on both scenarios and efficiency of the various probabilities in terms of service availability and number of alive nodes shown in Figure 5.5,



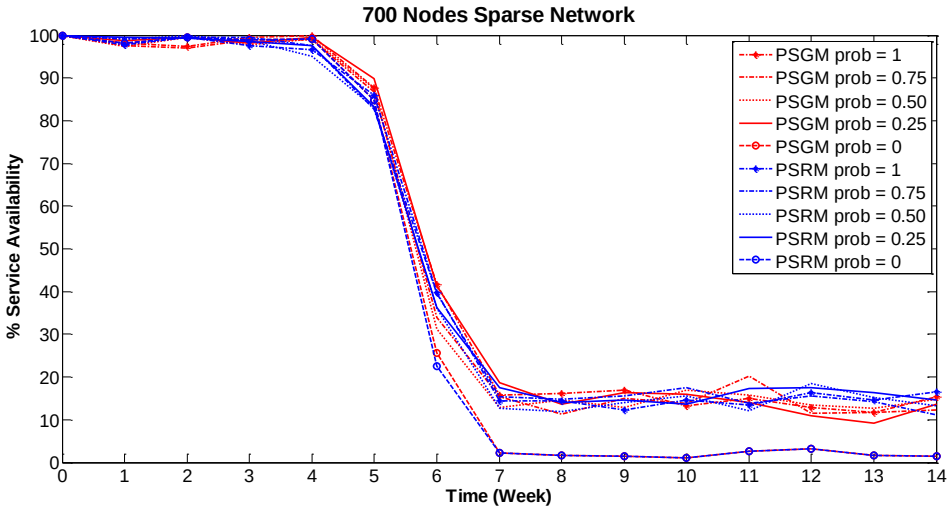
(a)



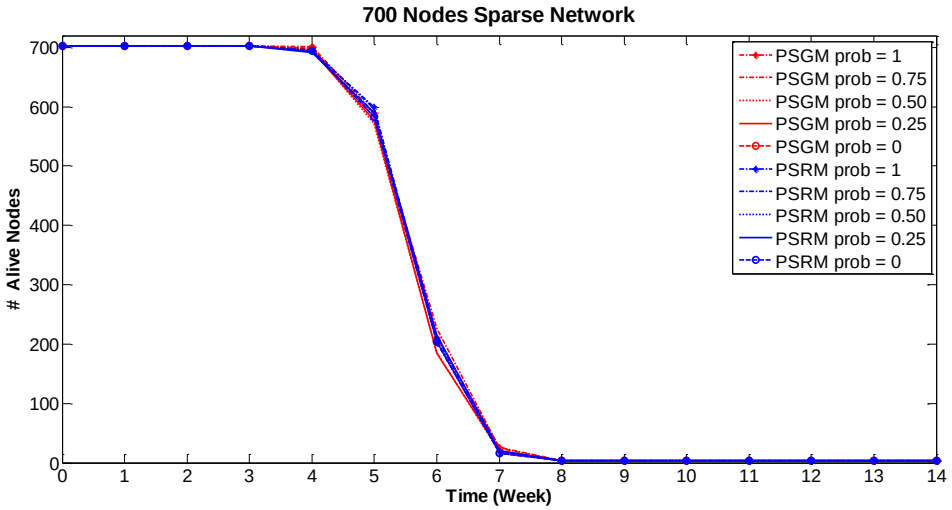
(b)

Figure 5.6: Experimental results: effects of *PS Robot Guidance* algorithm on 200 nodes sparse network topology on a uniform event distribution showing various probabilities on move (a) % service availability, (b) # alive nodes.

Figure 5.6 and Figure Fig:various700, we can conclude that fixed robots (probability 0 = no move) brings very small benefit but still improves the network performance. On the other hand, when robots constantly move they achieve the highest level of service availability and lowest level of sensor energy consumption. It is an interesting point to notice that probabilities 0.25, 0.5 and 0.75 do not have much difference on service availability and sensor energy consumption, however, the total distance



(a)



(b)

Figure 5.7: Experimental results: effects of *PS Robot Guidance* algorithm on 700 nodes sparse network topology on a uniform event distribution showing various probabilities on move (a) % service availability, (b) # alive nodes.

taken by robots are directly affected by the given probabilities. This then begs a question: is it better to sacrifice a little from the service availability and sensor energy usage but conserve the robots' energy by moving less? The answer of this question is based on the application domain and depends on how critical it is. By implementing *PS Guidance* we successfully direct robots into the areas of the sensor field where nodes are out of energy or much network activities occur. We will try

to solve the trade-off between service availability and the total distance taken by a robot in the future as it is not the scope of this paper. However, we have done some basic analysis on the total distance travelled by a robot in Figure 5.8 to get an idea of the technique on the total distance taken.

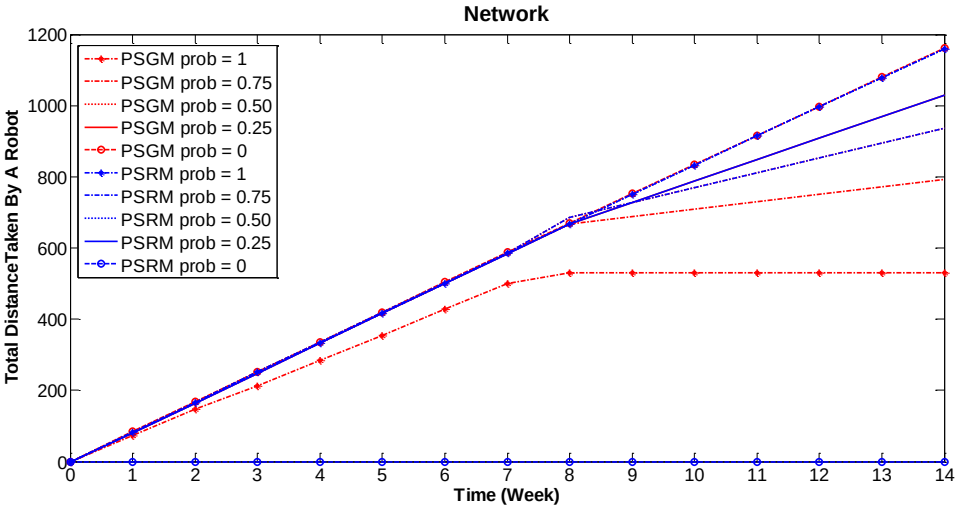


Figure 5.8: Experimental results: effects of *PS Robot Guidance* algorithm on a network topology on a uniform event distribution showing total distance travelled by a robotic agent.

Figure 5.8 illustrates the effects of the moving decision probability of *PS-Guided Move* and *PS-Random Move* on the total distance travelled by a robotic agent during the simulation. Due to the large number of experiments performed, each having 3 robots, we limit the results to the total distance travelled by a single robot on a network. The numbers in this figure are not important as they change with the network topology. However, the behaviour of the scenarios and ratio of the results are similar to each other in all cases.

5.3.3 Effects of PS Robotic Guidance Technique on Non-Uniformly Distributed Events

PS Robotic Guidance technique focuses on improving the network coverage and increasing the network performance in terms of service availability and network life-

time particularly where sensor nodes do not provide any service as we explained. In this section, we explore the performance consequences of *PS Robotic Guidance* technique on non-uniformly distributed events. We defined uniformly distributed events in the previous section as every location in the sensor filed has the same probability of the event occurring at it and as and we gave wild-life habitat monitoring as an example. We now define non-uniformly distributed events for this section and will contextualise it with an example based on the a case study. When events are non-uniformly distributed, not every location in the sensor filed has the same probability of the event occurring at it but some locations has higher probability of the event occurring at them, some locations has lower probability of the event occurring at them. To contextualise, we focus on the wild-life bird detection example once again, however, this time considering that there are two water fountains for the birds. In this case, although we use the same case study, the birds will not uniformly be distributed in the required area. Instead, the birds will fly around the water fountains more than the other parts of sensor field.

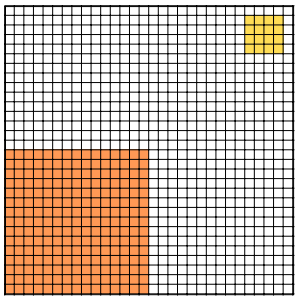
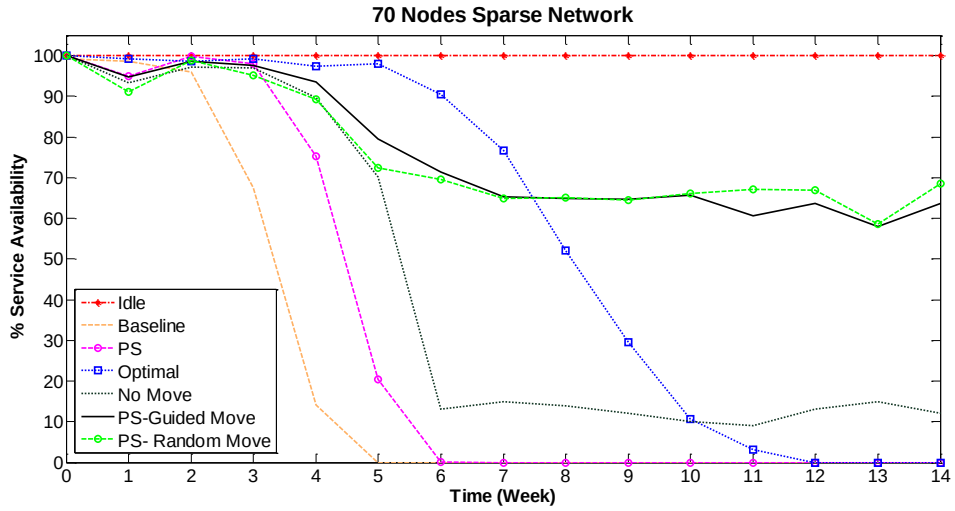


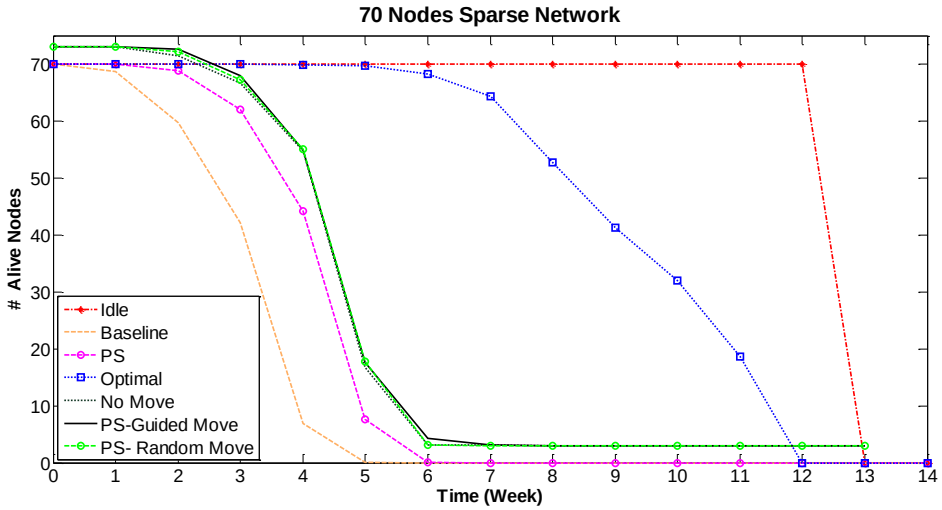
Figure 5.9: Non-uniform event distribution on sensor field.

Figure 5.9 illustrates the sensor field layout of non-uniform event distribution on 30m x 30m grid. Brown and yellow areas are coloured to indicate the water fountains located in the sensor field under assumption that water fountain located in the brown area is bigger and attracts more birds. Each square represents 1m x 1m, whereas blue, yellow and brown coloured areas indicate the different probability rates. Event distribution probabilities are set as 0.1, 0.3 and 0.6 respectively for this set of experiments. To ensure statistical significance, we have repeated each experiment 30 times.

Figure 5.10 shows the experimental results of the effects of *PS Robot Guidance*



(a)



(b)

Figure 5.10: Experimental results: effects of *PS Robot Guidance* algorithm on 70 nodes sparse network topology on a non-uniform event distribution showing (a) % service availability, (b) # alive nodes.

technique on a sparse network topology with 70 nodes (a) on the percentage of service availability, (b) the number of alive nodes. Events are non-uniformly distributed according to the Figure 5.9 with the respective probabilities. Until now, we have not focused on non-uniformly event distributions and therefore have not compared *PS* load balancing with no load balancing for this kind of event distribution. As a result, Figure 5.10 illustrates the *Baseline*, *PS* and *Optimal* scenarios as

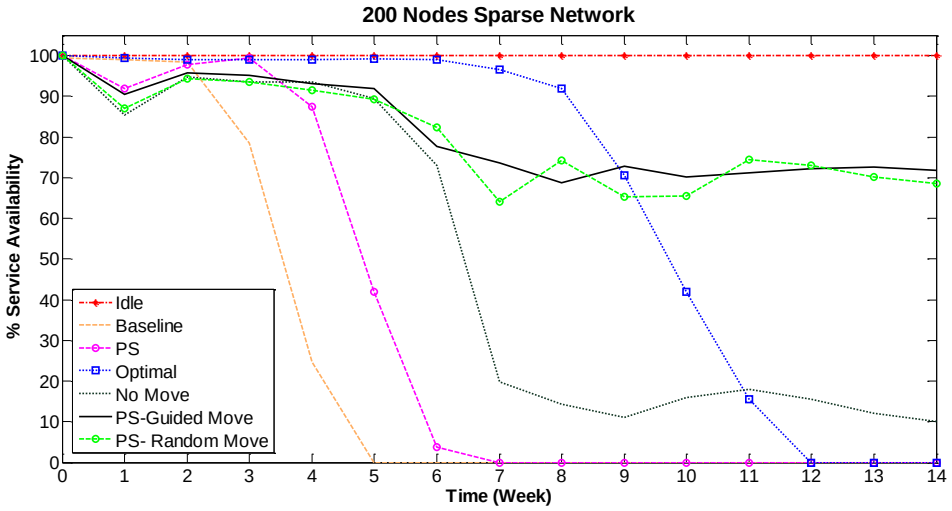
well as the scenarios contains mobile robotic agents such as *No Move*, *PS-Guided Move*, and *PS-Random Move*. The *Baseline*, *PS* and *Optimal* scenarios do not contain any mobile entities and their objectives are the same as explained in Section 3.4.1. The *Baseline* scenario, where no load balancing is applied, also performs worse than the *PS* load balancing technique in non-uniformly distributed events as well as uniformly distributed events. The *Optimal* scenario, an artificial scenario where no computational redundancy exists, performs better than both *Baseline* and *PS*, as expected. All three scenarios where no mobile agents are included (*Baseline, PS* and *Optimal*) perform visibly worse when events are non-uniformly distributed as opposed to uniformly distributed (Figure 4.9). When events are uniformly distributed, the probability of executing one event is same for all the network loads, however, when events are non-uniformly distributed certain parts of the network are forced to use their network resources more. It is natural to consume the network resources on the areas of the network where events occurs with high probability. Introducing load balancing increases the network performance in terms of service availability and extends the network lifetime as shown in the Figure 5.10, however, no load balancing technique can stop the unequal use of network resources when events are distributed non-uniformly. This is clearly shown in *Optimal* scenario in Figure 5.10. Even the *Optimal* scenario in Figure 5.10 has lower service availability when events are non-uniformly distributed as opposed to Figure 4.9 where events are uniformly distributed.

In terms of mobile aspects, three scenarios are used for this set of experiments: *No Move*, *PS-Guided Move*, and *PS-Random Move*. In Section , we illustrate the effectiveness of using mobile agents by comparing a variety of different probabilities of move. In our analysis, one of the conclusions is that using mobile robotic agents improves network performance in terms of service availability and network lifetime even when mobile agents have zero probability of move. This means that when the mobile agents are randomly deployed into the network, they do not move as their move is specified as probability zero. Although random move with zero probability performed the worst among the other mobile scenarios, it showed us that additional network resources increases the network performance as expected. In this set of

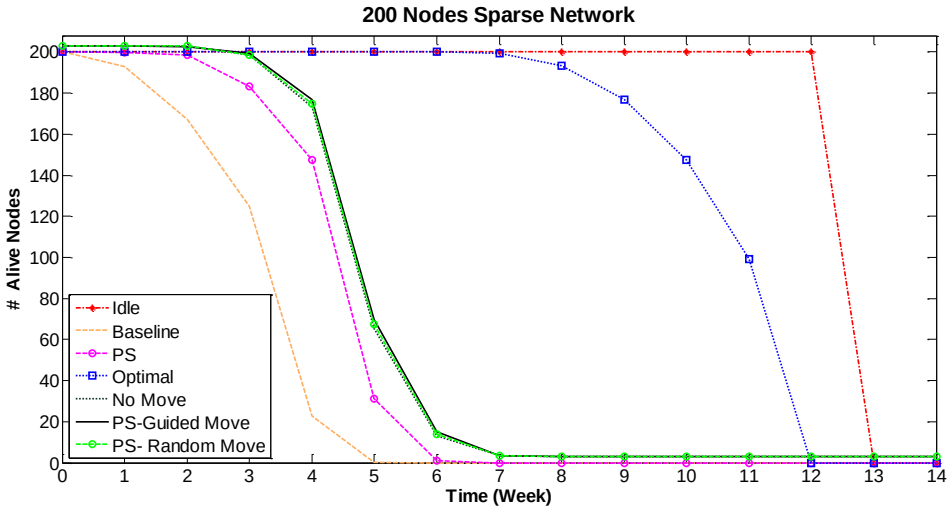
experiments, we used random move with zero moving probability (which we rename as *No Move*), *PS-Guided Move* and *PS-Random Move*. The *PS-Guided Move* and *PS-Random Move* have similar results especially after week 7. This is a result of no pheromone signalling occurring as all the nodes have already run out of energy, and the fact that when nodes are out of energy both *PS-Guided Move* and *PS-Random Move* scenario move around randomly. The benefit of using the *PS-Guided Move* over *PS-Random Move* occurs when the nodes start to run out of energy until all the nodes are out of energy and shown in by the end of week 3 to week 6. Although *No Move* scenario has a minor effect on the number of alive nodes over *PS*, it has significant effect on the percentage of service availability. This is due to the possibility of random deployed mobile agents in the areas where events occur more. Moving increases the network performance dramatically since there is a significant difference in terms of percentage of service availability in *No Move* and *PS-Guided Move* and *PS-Random Move* which clearly is shown after week 6. Our analysis on the *No Move*, *PS-Guided Move* and *PS-Random Move* scenarios on a sparse network with 70 nodes show that mobile network resources increases the network coverage and therefore network performance in terms of service availability in Figure 5.10 (a) and the network lifetime in Figure 5.10 (b) improved.

Figure 5.11 presents the simulated experimental results on a sparse network topology with 200 nodes using a non-uniform event distribution (a) the percentage of service availability, (b) the number of alive nodes. Similar to the previous figure, the *Baseline*, *PS* and *Optimal* scenarios are shown as well as the scenarios contains mobile robotic agents such as *No Move*, *PS-Guided Move*, and *PS-Random Move*. The *Baseline* scenario performs badly in terms of service availability and it also has the shortest network lifetime when the events are non-uniformly distributed. *PS* load balancing improves the network performance in both the percentage of service availability and it also increases the network lifetime over *Baseline*. The *Optimal* scenario plotted in Figure 5.11 also shows that the network performance in terms of service availability and the network lifetime is reduced over uniformly distributed events just like Figure 5.11.

The use of the robotic agents in *No Move*, *PS-Guided Move* and *PS-Random*



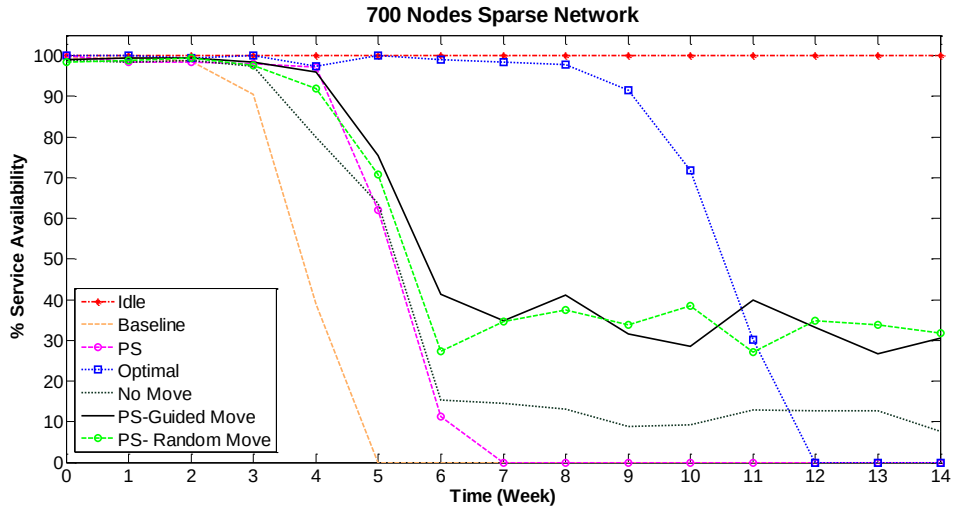
(a)



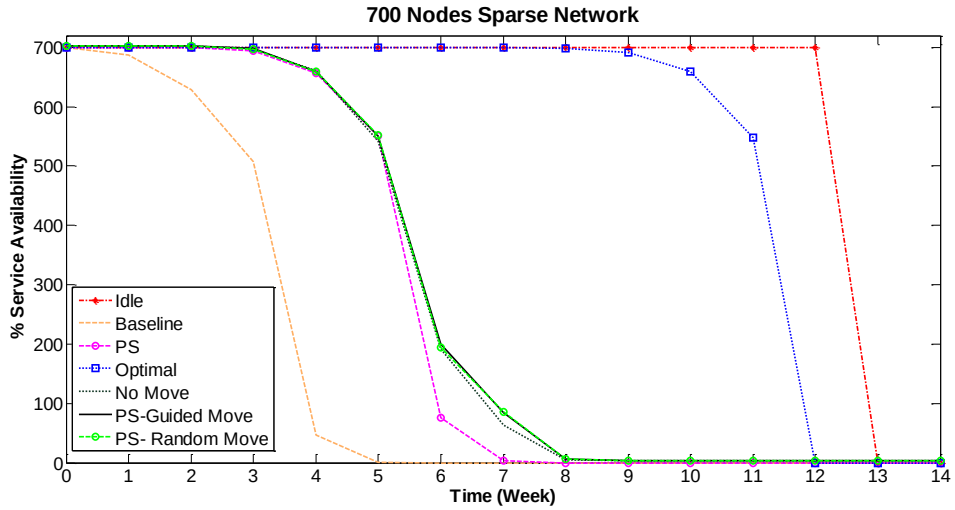
(b)

Figure 5.11: Experimental results: effects of *PS Robot Guidance* algorithm on 200 nodes sparse network topology on a non-uniform event distribution showing (a) % service availability, (b) # alive nodes.

Move scenarios illustrates that inserting additional network resources increases the network performance regardless although agents do not move as *No Move* scenario shows. The advantage of using *PS-Guided Move* over *PS-Random Move* is shown between week 3 to week 6, although the difference between the two scenarios are minor. The benefits of using mobile agents are clearly shown after the week 7, where *PS-Guided Move* and *PS-Random Move* scenarios over *No Move* scenario.



(a)



(b)

Figure 5.12: Experimental results: effects of *PS Robot Guidance* algorithm on 700 nodes sparse network topology on a non-uniform event distribution showing (a) % service availability, (b) # alive nodes.

Figure 5.12 illustrates the simulated results on a sparse network topology with 700 nodes using a non-uniform event distribution (a) the percentage of service availability, and (b) the number of alive nodes. Seven different scenarios are plotted including the *Idle*, *Baseline*, *PS* and *Optimal* scenarios which has not inspected in the previous chapters, as well as the scenarios contains mobile robotic agents such as *No Move*, *PS-Guided Move*, and *PS-Random Move*. The *Baseline* scenario that

applies no load balancing performs the worst among all other scenario due to the redundant computational consumption. *PS* load balancing improves the network performance both in Figure 5.12 (a) and (b) as the algorithm reduces the computational redundancy according to the pheromone signalling mechanism. Although the *Optimal* scenario performs better than any other scenarios represented in Figure 5.12, it does not perform as good when events are uniformly distributed shown in Figure 5.12.

Effectiveness of using the additional mobile network resources are shown in *No Move*, *PS-Guided Move*, and *PS-Random Move* scenarios. The advantage of using *PS-Guided Move* over *PS-Random Move* between the end of week 5 to the beginning of week 8. Moving robotics agents perform better in terms of the percentage of service availability and the number of alive nodes over mobile agents that do not move represented in the *No Move* scenario especially by the beginning of the week 8. Using additional network resources increase the network performance on the sparse network with 700 nodes although they are not moving as the *No Move* scenario illustrates as well as other sparse topologies. This therefore also shows the mobility increases the network coverage.

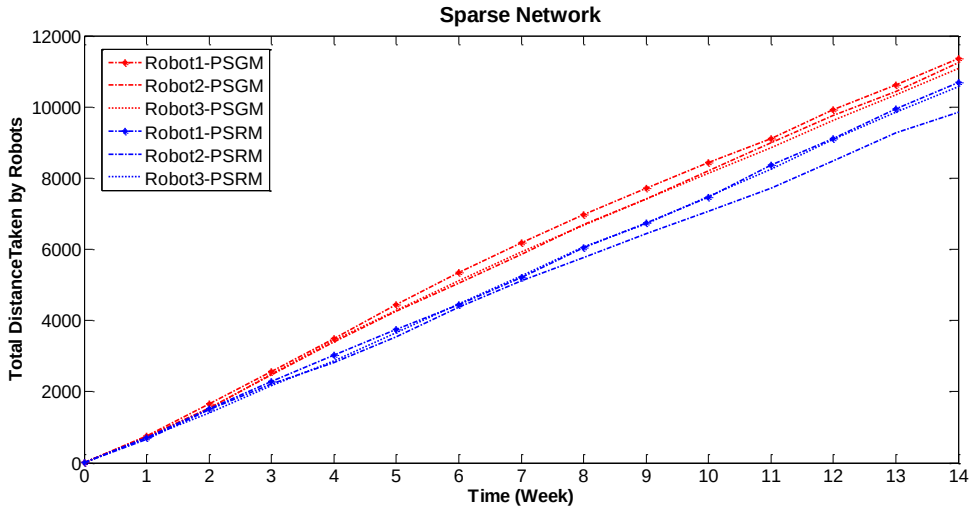


Figure 5.13: Experimental results: effects of *PS Robot Guidance* algorithm on a network topology on a non-uniform event distribution showing total distance travelled by all three robotic agents.

Figure 5.13 illustrates the distance travelled by robotic agents on *PS-Guided Move*, and *PS-Random Move* scenarios on a sparse network. As *No Move* do not involve any moves of the robotic agents, we do not add *No Move* scenario into our plot. All three robots have the same behaviour on *PS-Guided Move*, and *PS-Random Move* scenarios, where *PS-Random Move* perform slightly less moves in all three network topologies. As a result, we decided to use only one case to illustrate an example of robotic agents distance travelled. Based on distance travelled by the robotic agents and their performance, we can easily say *PS-Guided Move* performs better in non-uniformly distributed events in terms of all three performance metrics: the percentage of service availability, the number of alive nodes and the distance travelled by the robotic agents.

5.4 Summary

In this chapter, we have described an effective robot guidance technique that uses the *PS* load balancing algorithm to improve the network coverage in an attempt to address the trade-off between service availability and network lifetime in WSRNs. As the stationary sensor nodes are limited in processing capacity, we introduce mobile robotic agents in addition to the fixed sensornet topology. We propose to improve network coverage by guiding the robotic agents towards the areas of the sensor field where the sensor nodes out of battery and are unable to provide service. Thus, this not only improves the network coverage but also increases the service availability and extends the network lifetime. We showed two set of experimental results: 1) using a uniform event distribution; 2) using a non-uniform event distribution. As we showed in the previous chapters, sparse network topologies exhibit the same behaviour on the *PS* algorithm. Therefore, we decided to use sparse networks as they are more challenging as the distance between the nodes are not equal to each other. We show the effects of the *PS Robot Guidance* algorithm on event distributions separately in two different experimental sets, each containing three different sparse networks with 70, 200 or 700 nodes to show the effects of the *PS Robot Guidance* algorithm on network density.

Extensive experimental results on both 1) uniformly and 2) non-uniformly distributed events, on three different sparse network topologies demonstrate that our proposed *PS Robot Guidance* technique increases the service availability and extends the network lifetime as a result of improved network coverage, although the total distance travelled by the robots in high. Furthermore, we also show that using additional network resources increase the network performance although they do not move. From that, we believe that in the future many cyber-physical systems will benefit from employing heterogeneous entities, and it is inevitable to merge different subclasses of these systems in the same application.

In the future, we would like to consider the resource limitations of the robots, examining the trade-off between the total distance taken by a robot and the total service availability of the network.

The next chapter will be the conclusions and the future work.

Chapter 6

Conclusions and Future Work

To conclude, I recall from earlier the research question of this thesis. Specifically:

Can a distributed bio-inspired load balancing technique be used to improve service availability and network lifetime using adaptive and dynamic resource management for large scale WSNs with redundant coverage?

Classical load balancing techniques that apply centralised control mechanisms 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 look to the lightweight and distributed nature of bio-inspired mechanisms. In this thesis, we have presented a task mapping optimisation algorithm that addresses the trade-off between energy efficiency and event detection at run-time, maximising service availability while reducing energy consumption by restricting the service times of the network components.

In numerous experiments, spanning different network sizes and topologies, we have shown that our bio-inspired *PS* algorithm reduces computational redundancy by limiting the processing capabilities of the sensor nodes.

We now summarise the contributions made in this thesis, and propose future work.

6.1 Summary of Contributions

In short, the significant contributions of this thesis can be summarised as follows:

Load Balancing Using Pheromone Signalling

In Chapter 3, we satisfied research objective RO1 by developing a lightweight, bio-inspired load balancing algorithm that can satisfy the dynamic and adaptive characteristics of large scale WSNs. A number of experiments were conducted on a wide range of network sizes and topologies to illustrate the effect of *PS* technique. These all contributed to the conclusion that, given the good set of parameter selection is input to the *PS* algorithm, *PS* reduces the computational redundancy and thus, increases the service availability and extends the network lifetime.

Development of the *Fast* Simulation Infrastructure

Predominately in Chapter 3, but also in Chapter 4 and Chapter 5 *Fast* simulation infrastructure and its characteristics were presented. *Fast* was established to provide a system-level perspective of the load balancing issues on WSNs, and to investigate the characteristics of load distribution over network resources when dynamic, distributed algorithms were applied. This, therefore, satisfied our research objective RO2.

Development of *SuperFast* Simulation Infrastructure

In Chapter 4 we developed a second simulation infrastructure and its features were shown. The *SuperFast* simulation infrastructure was developed to investigate the feasibility of increasing the speed of the simulation infrastructure without sacrificing much its accuracy, and therefore address research objective RO3. The goal was to enable simulations to be performed quicker and thus evaluate different scenarios and configurations in a shorter amount of time, whilst still maintaining confidence in the accuracy of the results. The competitive analysis between *Fast* and *SuperFast* was conducted to show the accelerated speed effects in terms of time consumption and level of accuracy on the simulation infrastructures.

Parameter Tuning for the PS Load Balancing

In Chapter 4 parameter tuning was presented for the *PS* algorithm. The *Simulated Annealing* metaheuristic was use to automate the parameter selection

for the *PS* algorithm for various sizes and topologies of networks. The experimental results were illustrated to show that the applied parameter tuning technique provides a set of parameters to maximise service availability and minimise energy consumption of any given network configuration. Principle Component Analysis also was applied to the *PS* algorithm to show the effects of each parameter on the algorithm.

Pheromone Signalling-based Robot Guidance

In Chapter 5 an extension to the *PS* algorithm, used to guide robotic agents, was presented. An extensive set of experiments were performed to show that the *effective* guidance of mobile elements like robots can be used to increase network coverage both for uniformly and non-uniformly distributed events. This also increases the overall service availability and extends the network lifetime. Furthermore, all experiments presented in this chapter showed that combining different subclasses of cyber-physical systems can be used to improve network performance.

6.2 Limitations

Despite the contributions listed in the previous section, this thesis also has its limitations. Some of the important limitations can be listed as follows:

Level of Abstraction

Throughout this thesis, we underline that the simulation infrastructures we used, both *Fast* and *SuperFast*, are abstract simulators. We stated the reasons behind our choice as to provide a system-level perspective in order to extract the long-term effects of *PS* particularly for large scale WSNs. We applied *PS* on real-sensor deployment to show the short-term effects of the algorithm. Although we validated that the observed behaviour is the same on both simulated and deployed results, we could not analyse the difference in the level of accuracy due to the required time and financial cost for the real-deployments. Uncertainty in the level of accuracy on our simulated results is one of the most significant limitations behind this research.

Platform Specifications

All the experiments shown in this thesis use homogeneous sensor nodes and we have not done any research on using heterogeneous sensor nodes. We would like to underline that homogeneity or heterogeneity here only focuses on the sensor nodes and do not mean sensor node/robot combination in Chapter 5. We believe using heterogeneous sensor nodes in the platform model will lead some communication issues, time delays in both computation and communication processes and thus will effect results negatively. Furthermore, the timers of the heterogeneous sensor nodes may be different and thus may result with some challenges on applying time-triggered cycles of the *PS* algorithm such as decay and differentiation cycles. Moreover, due to the different energy consumption rates of heterogeneous nodes, some sensors may run out of energy faster than the others. Overcoming these issues may need some changes in the algorithm which may be subject to future work.

6.3 Future Work

Finally, we will address a few open problems that we think would be beneficial to explore in the future.

Level of Abstraction

In the previous section, we mention that one of the limitations of the *PS* load balancing algorithm is the uncertainty in the level of accuracy. We believe it would be very useful to further validate the level of accuracy of the *PS* using real sensor deployment where the deployment time and conditions will be same as the simulation setup. Surely, the financial cost, time and effort required for this research grows exponentially as the network size increase but we believe it will be useful to show the level of accuracy at least on a small-to-medium sized network. The results gained from this research can be used to calibrate the simulation infrastructure. An example of this is as follows: empirical results may show that the *PS* technique is 15% less accurate than the real sensor deployment and thus results in achieving higher service availability and

longer network lifetime in simulation than in reality. In order to make the simulation more accurate, noise might be added into the simulation and make the simulator (almost) as accurate as the real sensor deployment.

Verification of *PS* Technique using Formal Methods

In the *PS* load balancing algorithm, QN selection is driven by each nodes' pheromone level and the static predefined threshold (thresholdQN), as previously explained. The number of QNs, and the QN duration are controlled by the parameters set for the PS algorithm and applied computational redundancy is only based on the parameters input to the algorithm. A scenario where all the nodes are QNs at all times indicates that there is no load balancing being applied even though PS technique is being used. Similarly, in a scenario where none of the nodes are QNs means there are no nodes capable of providing service. We handle these two extreme problems with the parameter tuning method that is presented in Chapter 3. Empirical metaheuristic technique applied to automate parameter selection for the PS algorithm is validated with the experimental results in Chapter 4. We believe using simulators for WSNs are more beneficial. To recall, in Section 2.2.2 we discuss the benefits of different types of performance evaluation techniques of WSNs, showing a comparison between these techniques in Table 2.2. We claim that due to low cost, financial efficiency, high scalability, performance efficiency, and accuracy, simulators are an appropriate choice for evaluation. Simulators are typical way to evaluate the performance of WSNs because of the reasons listed in Table 2.2 together with the fact that simulators are closer matches of the behaviour of WSNs than mathematical models.

Analytical models can be created as an alternative option. Using analytical methods, however, often oversimplifies the problem domain and the results computed using these techniques are not always sufficient for real systems [28]. This is due to the complexity, size, and stochastic nature of the problem. To overcome the challenges of analysing WSNs, many analytical methods simplify

the problem dramatically (see e.g. [40, 191], [12]), and thus, result in non-realistic solutions. Analytical methods may not be capable of verifying PS as a whole, it may be possible to perform analyses of small parts of PS – with care given to ensure that the problem is not oversimplified. Given the advantages and disadvantages of formally verifying PS, we think it will be beneficial to apply some formal analysis techniques to answer specific questions such as:

- how often do nodes volunteer to become QN,
- how many QNs exist on average at any given time
- what is the probability of having at least one QN in the event occurrence range
- how quickly does a network stabilise and how does network size affect this

Care would be needed to ensure that the problem is not oversimplified to answer these questions in our future work on analytical models on PS.

Improving the PS-based Robot Guidance

In Chapter 5, when we validate our *PS-based Robot Guidance* algorithm, and showed that *PS-based Robot Guidance* algorithm covers the areas of the sensor field where there is no service providers. However, in terms of distance travelled by the robotic elements were not as short and efficient as near-optimal. The research question there was: is it better to sacrifice a little from the service availability and sensor energy usage but increase the robot lifespan by moving less? Currently, we do not know the answer of this question, neither the consequences of both. We believe it would be beneficial to extend the current *PS-based Robot Guidance* algorithm and increase the level of the complexity to increase the *effectiveness* of the network coverage.

Dependability Analysis of *PS*

In this thesis, we have analysed service availability and energy consumption, however, we have not done any research towards analysing how dependable is *PS* technique. The most important reason of not analysing dependability

is because of the application domain of the *PS*. In this thesis, we focused on developing a load balancing technique considering that our application domain will be a non-critical domain such as habitat monitoring. Given a non-critical application domain, we have shown *PS* technique successfully distributes the network load over the network components. In the future, we would like to apply *PS* technique on a safety-critical application domain such as fire alarm system used for the building management. Such systems require dependable techniques and therefore showing whether *PS* is a dependable technique or not would be important progress. In case where *PS* is currently not dependable, improving the algorithm towards achieving the dependability and showing the differences between the two approaches will be an interesting work.

Using a Different Application Domain

In the previous paragraph, we mentioned about dependability analysis of the *PS* for a safety-critical system. We also consider using a different application domain that is belongs to a non-critical system such as web servers, clusters or data centres. As the green computing is becoming more and more important every day, variety of load balancing techniques are been introduced to save energy on these domains. Most of these techniques apply central control mechanisms. Applying *PS* as distributed load balancing technique on a completely different application domain which also is a non-critical system, will help us to investigate how much *PS* can contribute towards green computing. Furthermore, we can also analyse the effects of centralised or distributed control mechanisms on these domains which also be interesting to show.

Analysing Effects of A MAC Layer Duty-Cycling Protocol on *PS*

As we state in Section 3.3.2, MAC layer duty-cycling protocols are out of scope of this research. One can easily imagine that if a duty-cycling MAC protocol has been incorporated, then network energy dissipation would have been lower and the network lifetime would have been increased, however it is difficult to know without a deep investigation. It is a known fact that duty-cycling MAC protocols are more energy efficient than traditional MAC protocols, but they

also have some limitations. Most importantly, synchronous duty-cycling MAC protocols may increase end-to-end delivery latency substantially; they rely on time synchronization between nodes to determine a pre-negotiated time and period for node communication. For example, with S-MAC [198], in each operational cycle, a packet can be forwarded over a single hop only, since an intermediate relaying node has to wait for it. In addition, the probability of network contention is high when synchronous duty-cycling MAC protocols are used because a sensor node is synchronised to be awake during the same short period as its neighbours and once a neighbouring node acknowledges the request then transmission commences [49]. On the other hand, asynchronous protocols commonly have lower latency, as transmission is on-demand. This comes at the expense of power due to the periodic listening required by neighbouring nodes. Based on the given features of synchronous and asynchronous duty-cycling MAC protocols, using a synchronous duty-cycling MAC protocol together with PS may affect network performance negatively by increasing network congestion and causing unnecessary latency. . PS signalling packages may occur at the same time together with propagation or decay cycles. And this may change the behaviour of PS entirely. Furthermore, using an asynchronous duty-cycling MAC protocol in combination with PS may result in a higher rate of power consumption. In short, without evaluating the effects of duty-cycling MAC protocols together with PS technique, it is difficult to conclude whether energy consumption of the network is likely to be lower than the presented results in this thesis, and this would be an interesting avenue of research to investigate in the future.

6.4 Closing Remarks

Load balancing in WSNs is almost certainly a necessity. By using bio-inspired algorithms, such as a bee's pheromone signalling mechanism, it is possible to improve the self-organisation and autonomous characteristic in WSNs that is used to effectively distribute the network load. Thus, improving the overall network performance in

terms of service availability and network lifetime. We hope that the *PS* technique is clear and easily implemented by any reader.

References

[1] S. Abdelhak, C.S. Gurram, J. Tessier, S. Ghosh, and M. Bayoumi. ETSSI: energy-based task scheduling simulator for wireless sensor networks. *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2821–2824, 2011.

[2] Julius Adler and Wung-Wai Tso. "Decision"-Making in bacteria: Chemotactic response of escherichia coli to conflicting stimuli. *Science*, 184(4143):1292–1294, 1974.

[3] A. Agah, K. Basu, and S.K. Das. Preventing DoS attack in sensor networks: a game theoretic approach. *IEEE International Conference on Communications (ICC)*, 5:3218–3222 Vol. 5, May 2005.

[4] Bahriye Akay and Dervis Karaboga. Parameter tuning for the artificial bee colony algorithm. In *Proceedings of the First International Conference on Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems*, pages 608–619, WrocBaw, Poland, 2009. Springer-Verlag.

[5] Bahriye Akay and Dervis Karaboga. Solving integer programming problems by using artificial bee colony algorithm. In *Proceedings of the XIth International Conference of the Italian Association for Artificial Intelligence Reggio Emilia on Emergent Perspectives in Artificial Intelligence*, pages 355–364, Reggio Emilia, Italy, 2009. Springer-Verlag.

[6] Mohmmad Arif and Tara Rani. ACO based routing for MANETS. *International Jouranal Of Wireless & Mobile Networks (IJWMN)*, pages 163–174, 2012.

[7] Baris Atakan and Ozgur B. Akan. Immune system based distributed node and rate selection in wireless sensor networks. In *Bio-Inspired Models of Network, Information and Computing Systems*, pages 1–8, Dec. 2006.

[8] Baris Atakan and Özgür B. Akan. Immune system-based energy efficient and reliable communication in wireless sensor networks. In Falko Dressler and Iacopo Carreras, editors, *Advances in Biologically Inspired Information Systems*, volume 69 of *Studies in Computational Intelligence*, pages 187–207. Springer Berlin Heidelberg, January 2007.

[9] N.A.B.A. Aziz, A.W. Mohemmed, and M.Y. Alias. A wireless sensor network coverage optimization algorithm based on particle swarm optimization and Voronoi diagram. *International Conference on Networking, Sensing and Control*, pages 602–607, March 2009.

[10] J Baras and H Mehta. A probabilistic emergent routing algorithm for mobile ad hoc networks. In *Proceedings of WiOpt '03: Modeling and Optimization in Mobile, AdHoc and Wireless Networks*, 2003.

[11] Rimon Barr, Zygmunt Haas, and Robbert van Renesse. JiST: an efficient approach to simulation using virtual machines. *Software: Practice and Experience*, 35(6):539–576, February 2005.

[12] Cinzia Bernardeschi, Paolo Masci, and Holger Pfeifer. Early prototyping of wireless sensor network algorithms in PVS. In *Computer Safety, Reliability, and Security*, pages 346–359. Springer, 2008.

[13] Amol P. Bhondekar, Renu Vig, Madan Lal Singla, C. Ghanshyam, and Pawan Kapur. Genetic algorithm based node placement methodology for wireless sensor networks. In *Proceedings of the International Multiconference of Engineers and Computer Scientists*, volume 1, pages 18–20, 2009.

[14] Zoran Bojkovic and Bojan Bakmaz. A survey on wireless sensor networks deployment. *WSEAS Transactions on Communications*, 7(12):1172–1181, 2008.

[15] T. Bokareva, N. Bulusu, and Sanjay Jha. SASHA: toward a self-healing hybrid sensor network architecture. *Embedded Networked Sensors, 2005. EmNetS-II. The Second IEEE Workshop on*, pages 71–78, May 2005.

[16] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Inc., 1999.

[17] P. Boonma and J. Suzuki. MONSOON: A coevolutionary multiobjective adaptation framework for dynamic wireless sensor networks. In *Proceedings of the 41st Annual International Conference on System Sciences*, page 497, jan. 2008.

[18] Michael Buettner, Gary V. Yee, Eric Anderson, and Richard Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pages 307–320. ACM, 2006.

[19] J. Buhl, D. J. T. Sumpter, I. D. Couzin, J. J. Hale, E. Despland, E. R. Miller, and S. J. Simpson. From disorder to order in marching locusts. *Science*, 312(5778):1402–1406, 2006.

[20] John Byers and Gabriel Nasser. Utility-based decision-making in wireless sensor networks. In *Proceedings of the First ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 143–144, Boston, Massachusetts, 2000. IEEE Press.

[21] I. Caliskanelli, J. Harbin, F Indrusiak, L. Polack, P. Mitchell, and D Chesmore. Runtime optimisation in WSNs for load balancing using pheromone signalling. In *Proceedings of the IEEE 3rd International Conference on Networked Embedded Systems for Every Application (NESEA)*, dec. 2012.

[22] I. Caliskanelli and L. Indrusiak. Search-based parameter tuning on application-level load balancing for distributed embedded systems. In *Proceedings of the 11th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC 2013)*, 2013.

[23] I. Caliskanelli and L. Indrusiak. Using mobile robotic agents to increase service availability and extend network lifetime on wireless sensor and robot networks. In *Proceedings of the 12th IEEE International Conference on Industrial Informatics (INDIN)*, 2014.

[24] Ipek Caliskanelli, James R. Harbin, Leandro Soares Indrusiak, Paul D. Mitchell, Fiona A C Polack, and David Chesmore. Bio-inspired load balancing in large-scale WSNs using pheromone signalling. *International Journal of Distributed Sensor Networks*, 2013.

[25] Tiago Camilo, Carlos Carreto, Jorge Sá Silva, and Fernando Boavida. An energy-efficient ant-based routing algorithm for wireless sensor networks. In Marco Dorigo, LucaMaria Gambardella, Mauro Birattari, Alcherio Martinoli, Riccardo Poli, and Thomas Stützle, editors, *Ant Colony Optimization and Swarm Intelligence*, volume 4150 of *Lecture Notes in Computer Science*, pages 49–59. Springer Berlin Heidelberg, January 2006.

[26] A. Caracas, T. Kramp, M. Baentsch, M. Oestreicher, T. Eirich, and I. Romanov. Mote runner: A multi-language virtual machine for small embedded devices. In *Proceedings of the 2009 Third International Conference on Sensor Technologies and Applications*, pages 117–125. IEEE Computer Society, 2009.

[27] M. Cardei, M.O. Pervaiz, and I. Cardei. Energy-efficient range assignment in heterogeneous wireless sensor networks. *International Conference on Wireless and Mobile Communications (ICWMC)*, pages 11–11, July 2006.

[28] B. Sai Chand, K. Raghava Rao, and S. Sreedhar Babu. Exploration of new simulation tools for wireless sensor networks. *International Journal of Science and Research (IJSR)*, 2(4), April 2013.

[29] Phoebus Wei-Chih Chen. *Wireless Sensor Network Metrics for Real-Time Systems*. PhD thesis, University of California at Berkeley, 2009.

[30] Elaine Cheong, Edward A. Lee, and Yang Zhao. Viptos: a graphical development and simulation environment for TinyOS-based wireless sensor networks.

In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 302–302, San Diego, California, USA, 2005. ACM.

[31] Pietro Ciciriello, Luca Mottola, and Gian Pietro Picco. Efficient routing from multiple sources to multiple sinks in wireless sensor networks. In *Wireless Sensor Networks*, pages 34–50. Springer, 2007.

[32] J. Clarke, J.J. Dolado, M. Harman, R. Hierons, B. Jones, M. Lumkin, B. Mitchell, S. Mancoridis, K. Rees, M. Roper, and M. Shepperd. Reformulating software engineering as a search problem. *Software, IEEE Proceedings*, 150(3):161 – 175, june 2003.

[33] Pau Closas and Juan A. Fernandez-Rubio. A game theoretical algorithm for joint power and topology control in distributed WSN. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2765–2768. IEEE, 2009.

[34] Riccardo Crepaldi, Albert Harris, Alberto Scarpa, Andrea Zanella, and Michele Zorzi. SignetLab: deployable sensor network testbed and management tool. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pages 375–376, Boulder, Colorado, USA, 2006. ACM.

[35] Garth V. Crosby and Niki Pissinou. Evolution of cooperation in multi-class wireless sensor networks. In *Proceedings of the 32nd IEEE Conference on Local Computer Networks (LCN)*, pages 489–495. IEEE, 2007.

[36] K. Daabaj, M.W. Dixon, and T. Koziniec. Reliable load-balancing routing algorithm for wireless sensor networks. In *Proceedings of the 19th IEEE International Conference on Computer Communication Networks (ICCCN)*. IEEE, 2010.

[37] Dipankar Dasgupta. An overview of artificial immune systems and their applications. In Dipankar Dasgupta, editor, *Artificial Immune Systems and Their Applications*, pages 3–21. Springer Berlin Heidelberg, January 1999.

[38] Gianni Di Caro and Marco Dorigo. AntNet: distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9(1):317–365, December 1998.

[39] Robert W. Dimand and Marry Ann Dimand. The early history of theory of strategic games from waldegrave to borel. In E. Roy Weintraub, editor, *Toward a History of Game Theory*. Duke University Press, 1992.

[40] Clare Dixon, Alan Winfield, and Michael Fisher. Towards temporal verification of emergent behaviours in swarm robotic systems. In Roderich Groß, Lyuba Alboul, Chris Melhuish, Mark Witkowski, Tony J. Prescott, and Jacques Penders, editors, *Towards Autonomous Robotic Systems*, volume 6856 of *Lecture Notes in Computer Science*, pages 336–347. Springer Berlin Heidelberg, 2011.

[41] Salvatore Drago, D. M. W. Leenaerts, Fabio Sebastiano, Lucien J. Breems, Kofi AA Makinwa, and Bram Nauta. A 2.4 GHz 830pJ/bit duty-cycled wake-up receiver with 82dBm sensitivity for crystal-less wireless sensor nodes. *IEEE International Solid-State Circuits Conference (ISSCC)*, 2010.

[42] F. Dressler. A study of self-organization mechanisms in ad hoc and sensor networks. *Computer Communications*, 31(13):3018–3029, 2008.

[43] Falko Dressler and Ozgur B. Akan. Bio-inspired networking: from theory to practice. *Communications Magazine, IEEE*, 48(11):176–183, 2010.

[44] Wan Du, David Navarro, Fabien Mieyeville, and Frédéric Gaffiot. Towards a taxonomy of simulation tools for wireless sensor networks. In *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, page 52, 2010.

[45] Harald Dyckhoff. A typology of cutting and packing problems. *European Journal of Operational Research*, 44(2):145–159, 1990.

[46] E. Egea-Lopez, J. Vales-Alonso, A. Martinez-Sala, P. Pavon-Mario, and J. Garcia-Haro. Simulation scalability issues in wireless sensor networks. *Communications Magazine, IEEE*, 44(7):64 – 73, july 2006.

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

[48] Orhan Engin and Alper Döyen. A new approach to solve hybrid flow shop scheduling problems by artificial immune system. *Computational science of lattice Boltzmann modelling*, 20(6):1083–1095, August 2004.

[49] Mark Louis Fairbairn and Iain Bate. Using feedback control within WSNs to meet application requirements. In *Proceedings of the 5th Workshop on Performance Control in Wireless Sensor Networks (PWSN)*, pages 415–422. IEEE, 2013.

[50] R. Falcon, Xu Li, A. Nayak, and I. Stojmenovic. A harmony-seeking firefly swarm to the periodic replacement of damaged sensors by a team of mobile robots. *IEEE International Conference on Communications (ICC)*, pages 4914–4918, June 2012.

[51] J. Faruque and A. Helmy. RUGGED: Routing on fingerprint gradients in sensor networks. In *Proceedings of the IEEE/ACS International Conference on Pervasive Services*, pages 179 – 188, july 2004.

[52] Andrew Faulkner. Altruistic and egoistic behaviour in wireless sensor networks. Master's thesis, The University of York, 2014.

[53] M. Felegyhazi, J-P Hubaux, and L. Buttyan. Cooperative packet forwarding in multi-domain sensor networks. *Pervasive Computing and Communications Workshops, 2005. PerCom 2005 Workshops. Third IEEE International Conference on*, pages 345–349, March 2005.

[54] Konstantinos P. Ferentinos and Theodore A. Tsiligiridis. Adaptive design optimization of wireless sensor networks using genetic algorithms. *Computer Networks*, 51(4):1031–1051, 2007.

[55] D. W. Gage. *Command control for many-robot systems.* Naval Command Control and Ocean Surveillance Center, San Diego, 1992.

[56] Meijuan Gao and Jingwen Tian. Path planning for mobile robot based on improved simulated annealing artificial neural network. In *Proceedings of the 3rd International Conference on Natural Computation*, volume 3, pages 8 –12, aug. 2007.

[57] Jacques Gautrais, Pablo Michelena, Angela Sibbald, Richard Bon, and Jean-Louis Deneubourg. Allelomimetic synchronization in merino sheep. *Animal Behaviour*, 74(5):1443 – 1454, 2007.

[58] Lewis Girod, Nithya Ramanathan, Jeremy Elson, Thanos Stathopoulos, Martin Lukac, and Deborah Estrin. Emstar: A software environment for developing and deploying heterogeneous sensor-actuator networks. *ACM Transactions Sensor Networks*, 3(3):13, 2007.

[59] M.M. Goswami, R.V. Dharaskar, and V.M. Thakare. Fuzzy ant colony based routing protocol for mobile ad hoc network. *International Conference on Computer Engineering and Technology (ICCET)*, 2:438–444, January 2009.

[60] M. G. Gouda and T. M. McGuire. Accelerated heartbeat protocols. In *Proceedings of the 18th International Conference on Distributed Computing Systems*, pages 202–209, 1998.

[61] Daniel Grünbaum, Steven Viscido, and JuliaK. Parrish. Extracting interactive control algorithms from group dynamics of schooling fish. In Vijay Kumar, Naomi Leonard, and A. Stephen Morse, editors, *Cooperative Control*, volume 309 of *Lecture Notes in Control and Information Science*, pages 103–117. Springer Berlin Heidelberg, 2005.

[62] M. Gunes, U. Sorges, and I. Bouazizi. ARA-the ant-colony based routing algorithm for MANETs. In *Proceedings of the International Conference on Parallel Processing Workshops*, pages 79–85, 2002.

[63] Zygmunt J. Haas, Joseph Y. Halpern, and Li Li. Gossip-based ad hoc routing. *IEEE Transactions on Networking*, 14(3):479–491, 2006.

[64] N. Haddadou, A. Rachedi, and Y. Ghamri-Doudane. Advanced diffusion of classified data in vehicular sensor networks. *7th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 777–782, July 2011.

[65] 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)*, pages 914 –919, Oct. 2009.

[66] Khaled A. Harras, Kevin C. Almeroth, and Elizabeth M. Belding-Royer. Delay tolerant mobile networks (DTMNs): controlled flooding in sparse mobile networks. In *Proceedings of the 4th IFIP-TC6 International Conference on Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communication Systems*, pages 1180–1192, Waterloo, Canada, 2005. Springer-Verlag.

[67] 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 Sciences*, volume 2, page 10, Jan. 2000.

[68] John H. Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* University of Michigan Press, 1975.

[69] Chi-Fu Huang and Yu-Chee Tseng. The coverage problem in a wireless sensor network. In *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications*, pages 115–121, San Diego, CA, USA, 2003. ACM.

[70] Kou-Yuan Huang and Yueh-Hsun Hsieh. Very fast simulated annealing for pattern detection and seismic applications. In *Proceedings of the IEEE International Conference on Geoscience and Remote Sensing Symposium*, pages 499 –502, july 2011.

[71] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: Social-based forwarding in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 10(11):1576–1589, 2011.

[72] M. Imran, A.M. Said, and H. Hasbullah. A survey of simulators, emulators and testbeds for wireless sensor networks. *2010 International Symposium on Information Technology (ITSim)*, 2:897–902, June 2010.

[73] Francois Ingelrest, Guillermo Barrenetxea, Gunnar Schaefer, Martin Vetterli, Olivier Couach, and Marc Parlange. SensorScope: application-specific sensor network for environmental monitoring. *ACM Transactions Sensor Networks*, 6(2):1–32, 2010.

[74] Yichao Jin, Dali Wei, Alexander Gluhak, and Klaus Moessner. Latency and energy-consumption optimized task allocation in wireless sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, Apr. 2010.

[75] Jingfeng Xue, Jiansheng Li, Yuanda Cao, and Ji Fang. Advanced PROPHET routing in delay tolerant network. In *Proceedings of the International Conference on Communication Software and Networks (ICCSN)*, pages 411–413, February 2009.

[76] Mikael Johansson, Lin Xiao, and Stephen Boyd. Simultaneous routing and power allocation in CDMA wireless data networks. In *Proceedings of the IEEE International Conference on Communications (ICC)*, volume 1, pages 51–55. IEEE, 2003.

[77] David S. Johnson, Alan Demers, Jeffrey D. Ullman, Michael R. Garey, and Ronald L. Graham. Worst-case performance bounds for simple one-

dimensional packing algorithms. *SIAM Journal on Computing*, 3(4):299–325, 1974.

[78] Joon Yoo, Sunwoong Choi, and Chong-Kwon Kim. The capacity of epidemic routing in vehicular networks. *Communications Letters, IEEE*, 13(6):459–461, June 2009.

[79] D. Jourdan and Olivier L. de Weck. Layout optimization for a wireless sensor network using a multi-objective genetic algorithm. In *Proceedings of the IEEE 59th Vehicular Technology Conference (VTC)*, volume 5, pages 2466–2470. IEEE, 2004.

[80] Konstantinos Kalpakis, Koustuv Dasgupta, and Parag Namjoshi. Maximum lifetime data gathering and aggregation in wireless sensor networks. *Proceedings of IEEE Networks*, 2:685–696, 2002.

[81] A.A. Kannan, Guoqiang Mao, and B. Vucetic. Simulated annealing based localization in wireless sensor network. In *Proceedings of the IEEE Conference on Local Computer Networks*, pages 2 pp. –514, nov. 2005.

[82] Anushiya A. Kannan, Guoqiang Mao, and Vucetic Branka. Simulated Annealing based Wireless Sensor Network Localization. *Journal of Computers*, 1(2):15–22, May 2006.

[83] R. Kannan and S. S. Iyengar. Game-theoretic models for reliable path-length and energy-constrained routing with data aggregation in wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 22(6):1141–1150, 2006.

[84] D. Karaboga and B. Basturk. On the performance of artificial bee colony (ABC) algorithm. *Applications Soft Computing*, 8(1):687–697, 2008.

[85] Dervis Karaboga. Artificial bee colony algorithm. *scholarpedia*, 5(3):6915, 2010.

[86] Dervis Karaboga and Bahriye Akay. A modified artificial bee colony (ABC) algorithm for constrained optimization problems. *Applications Soft Comput.*, 11(3):3021–3031, 2011.

[87] Dervis Karaboga and Bahriye Basturk. Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. In *Proceedings of the 12th International Fuzzy Systems Association World Congress on Foundations of Fuzzy Logic and Soft Computing*, pages 789–798, Cancun, Mexico, 2007. Springer-Verlag.

[88] Dervis Karaboga and Bahriye Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, 39(3):459–471, 2007.

[89] Dervis Karaboga and Celal Ozturk. A novel clustering approach: Artificial bee colony (ABC) algorithm. *Applications Soft Computing*, 11(1):652–657, 2011.

[90] Wolfgang Kiess and Martin Mauve. A survey on real-world implementations of mobile ad-hoc networks. *Ad Hoc Networks*, 5(3):324–339, 2007.

[91] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[92] T. Kontos, E. Zaimidis, C. Anagnostopoulos, S. Hadjiefthymiades, and E. Zervas. An adaptive epidemic information dissemination scheme with cross-layer enhancements. *IEEE Symposium on Computers and Communications (ISCC)*, pages 230–235, June 2011.

[93] Marko Korkalainen, Mikko Sallinen, Niilo Karkkainen, and Pirkka Tukeva. Survey of wireless sensor networks simulation tools for demanding applications. In *Proceedings of the Fifth International Conference on Networking and Services (ICNS)*, pages 102–106. IEEE, 2009.

[94] David Kotz, Calvin Newport, Robert S. Gray, Jason Liu, Yougu Yuan, and Chip Elliott. Experimental evaluation of wireless simulation assumptions. In *Proceedings of the 7th ACM International Symposium on Modeling, Analysis*

and Simulation of Wireless and Mobile Systems, MSWiM '04, pages 78–82, New York, NY, USA, 2004. ACM.

[95] Bhaskar Krishnamachari and Fernando Ordónez. Analysis of energy-efficient, fair routing in wireless sensor networks through non-linear optimization. In *Proceedings of the 58TH IEEE Vehicular Technology Conference*, volume 5, pages 2844–2848. IEEE, 2003.

[96] R.V. Kulkarni and G.K. Venayagamoorthy. Particle swarm optimization in wireless-sensor networks: A brief survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 41(2):262–267, March 2011.

[97] J. Suresh Kumar and E. Babu Raj. Genetic algorithm based multicast routing in wireless sensor networks-a research framework. *IEEE Sensors Journal*, 7(5):627–636, May 2007.

[98] C. C. Lee and Der-Tsai Lee. A simple on-line bin-packing algorithm. *Journal of the ACM (JACM)*, 32(3):562–572, 1985.

[99] Uichin Lee, Eugenio Magistretti, Mario Gerla, Paolo Bellavista, Pietro Lió, and Kang-Won Lee. Bio-inspired multi-agent data harvesting in a proactive urban monitoring environment. *Ad Hoc Networks*, 7(4):725–741, 2009.

[100] Philip Levis and David Gay. *TinyOS Programming*. Cambridge University Press, first edition, April 2009.

[101] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. TOSSIM: accurate and scalable simulation of entire TinyOS applications. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, pages 126–137, Los Angeles, California, USA, 2003. ACM.

[102] Xu Li, R. Falcon, A. Nayak, and I. Stojmenovic. Servicing wireless sensor networks by mobile robots. *Communications Magazine, IEEE*, 50(7):147–154, July 2012.

[103] Tiong Hoo Lim. Detecting anomalies in wireless sensor networks. *Qualifying dissertation, Department of Computer Science, University of York*, 2010.

[104] Tiong Hoo Lim, HuiKeng Lau, Jon Timmis, and Iain Bate. Immune-inspired self healing in wireless sensor networks. In *Proceedings of the 11th international conference on Artificial Immune Systems*, pages 42–56, Taormina, Italy, 2012. Springer-Verlag.

[105] Lin Cui and HongPeng Wang. Reachback firefly synchronicity with late sensitivity window in wireless sensor networks. In *Proceedings of the 9th International Conference on Hybrid Intelligent Systems (HIS)*, volume 1, pages 451–456, August 2009.

[106] Aristid Lindenmayer. Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of Theoretical Biology*, 18(3):280–299, 1968.

[107] S. Lindsey and C.S. Raghavendra. Pegasis: Power-efficient gathering in sensor information systems. In *Proceedings of the IEEE Aerospace Conference*, volume 3, pages 1125–1130, 2002.

[108] Matthew Liotine. *Mission-critical Network Planning*. Artech House, Jan 2003.

[109] M. C. Little. JavaSim, 2009.

[110] Zhonghai Lu, Lei Xia, and A. Jantsch. Cluster-based simulated annealing for mapping cores onto 2d mesh networks on chip. In *Proceedings of the 11th IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems*, pages 1 –6, april 2008.

[111] Sean Luke. *Essentials of metaheuristics*. Lulu, second edition, 2013.

[112] RT Marler and JS Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, April 2004.

[113] S Mccanne, S Floyd, and K Fall. ns2 Network Simulator 2. http://www.isi.edu/nsnam/ns/.

[114] Jonathan M. McCune, Elaine Shi, Adrian Perrig, and Michael K. Reiter. Detection of denial-of-message attacks on sensor network broadcasts. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 64–78. IEEE, 2005.

[115] Michael Meisel, Vasileios Pappas, and Lixia Zhang. A taxonomy of biologically inspired research in computer networking. *Computer Networks*, 54(6):901–916, 2010.

[116] Tommaso Melodia, Dario Pompili, Vehbi C. Gungor, and Ian F. Akyildiz. A distributed coordination framework for wireless sensor and actor networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 99–110. ACM, 2005.

[117] Inc Memsic. Iris Wireless Measurement System. http://www.memsic.com/support/documentation/wireless-sensor-networks/category/7-datasheets.html?download=135%3Airis.

[118] S. Merkel, C.W. Becker, and H. Schmeck. Firefly-inspired synchronization for energy-efficient distance estimation in mobile ad-hoc networks. In *Proceedings of the 31st IEEE International Performance Computing and Communications Conference (IPCCC)*, pages 205–214, December 2012.

[119] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[120] David A. Miller, Sameer Tilak, and Tony Fountain. "token" equilibria in sensor networks with multiple sponsors. In *International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 2005.

[121] Julian F. Miller, Simon L. Harding, and Gunnar Tufte. Evolution-in-materio: evolving computation in materials. *Evolutionary Intelligence*, 7(1):49–67, April 2014.

[122] Daniel Miorandi, Lidia Yamamoto, Amir Mujkanovic, and David Lowe. Fault tolerance of embryonic algorithms in mobile networks. In *Evolvable Systems: From Biology to Hardware*, volume 6274 of *Lecture Notes in Computer Science*, pages 49–60. Springer Berlin Heidelberg, 2010.

[123] Daniele Miorandi, Lidia Yamamoto, and P. Dini. Service evolution in a bio-inspired communication system. *International Transactions Systems Science and Applications*, 2(1):51–60, 2006.

[124] Renato E. Mirollo and Steven H. Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, 50(6):1645–1662, 1990.

[125] Iñigo Monedero, Carlos León, Robert Denda, and Joaquín Luque. Datacab: a geographical-information-system-based expert system for the design of cable networks. *Expert Systems*, 25(4):335–348, 2008.

[126] Roberto Montemanni, Luca Maria, Gambardella Arindam, and K. Das. The minimum power broadcast problem in wireless networks: a simulated annealing approach. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, 2005.

[127] Raymond Mulligan and Habib M. Ammari. Coverage in wireless sensor networks: A survey. *Network Protocols & Algorithms*, 2(2), 2010.

[128] S. Nandy, M. Karmakar, P.P. Sarkar, A. Das, A. Abraham, and D. Paul. Agent based adaptive firefly back-propagation neural network training method for dynamic systems. In *Proceedings of the 12th International Conference on Hybrid Intelligent Systems (HIS)*, pages 449–454, December 2012.

[129] Arash Nikdel, Seyed Mahdi Jameii, and Hagar Noori. A novel scheduling mechanism based on artificial immune system for communication between cluster

head and cluster members in WSNs. *International Journal of Information and Electronics Engineering*, 2(3):333–337, May 2012.

[130] Nojeong Heo and P.K. Varshney. A distributed self spreading algorithm for mobile wireless sensor networks. In *Proceedings of the IEEE Conference on Wireless Communcations and Networking (WCNC)*, volume 3, pages 1597–1602, March 2003.

[131] Selcuk Okdem and Dervis Karaboga. Routing in wireless sensor networks using an ant colony optimization (aco) router chip. *Sensors (Basel)*, 2:909–921, 2009;9.

[132] Fernando Ordónez and Bhaskar Krishnamachari. Optimal information extraction in energy-limited wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 22(6):1121–1129, 2004.

[133] Luciano Ost, Marcelo Mandelli, Gabriel Marchesan Almeida, Leandro Moller, Leandro Soares Indrusiak, Gilles Sassatelli, Pascal Benoit, Manfred Glesner, Michel Robert, and Fernando Moraes. Power-aware dynamic mapping heuristics for noc-based mpsocs using a unified model-based approach. *ACM Transactions Embedded Computing Systems*, 12(3):1–22, April 2013.

[134] Vasileios Pappas, Dinesh Verma, Bong-Jun Ko, and Ananthram Swami. A circulatory system approach for wireless sensor networks. *Ad Hoc Networks*, 7(4):706–724, 2009.

[135] N. Pari, A. Kailas, and M. Nogueira. Bio-inspired time synchronization for cognitive radio ad hoc networks. *IEEE Globecom Workshops*, pages 980–985, December 2012.

[136] Heemin Park and Mani B. Srivastava. Energy-efficient task assignment framework for wireless sensor networks. Technical report, UC Los Angeles, 2003.

[137] Dimosthenis Pediaditakis, Yuri Tselishchev, and Athanassios Boulis. Performance and scalability evaluation of the castalia wireless sensor network simu-

lator. In *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, pages 1–6, Torremolinos, Malaga, Spain, 2010.

[138] Jovan Pehcevski and Benjamin Piwowarski. Evaluation Metrics. In Ling Liu and M. Tamer Özsu, editors, *Encyclopedia of Database Systems*. Springer-Verlag, 2007.

[139] Andres Perez-Garcia, Christos Siaterlis, and Marcelo Masera. Studying characteristics of an emulab testbed for scientifically rigorous experiments. In *Proceedings of the 2010 IEEE/ACM 14th International Symposium on Distributed Simulation and Real Time Applications*, pages 215–218. IEEE Computer Society, 2010.

[140] 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*, pages 90 –100, feb 1999.

[141] Wint Yi Poe and Jens B. Schmitt. Node deployment in large wireless sensor networks: Coverage, energy consumption, and worst-case delay. In *Proceedings of the Asian Internet Engineering Conference*, AINTEC '09, pages 77–84, New York, NY, USA, 2009. ACM.

[142] J Polley, D Blazakis, J McGee, D Rusk, and JS Baras. ATEMU: a fine-grained sensor network simulator. pages 145–152. IEEE Press, October 2004.

[143] Vasaki Ponnusamy, Anang Hudaya, and Alan G. Downe. A biologically inspired energy efficient intrusion detection system. In *Proceedings of the International Conference on Computer & Information Science (ICCIS)*, volume 2, pages 729–735. IEEE, 2012.

[144] V. Rafe, H. Momeni, and M. Sharifi. Energy-aware task allocation in wireless sensor actor networks. In *Proceedings of the 2nd International Conference on Computer and Electrical Engineering (ICCEE)*, volume 1, pages 145 –148, dec. 2009.

[145] Outi Räihä, Erkki Mäkinen, and Timo Poranen. Using simulated annealing for producing software architectures. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, pages 2131–2136. ACM, 2009.

[146] Ram Ramanathan, Richard Hansen, Prithwish Basu, Regina Rosales-Hain, and Rajesh Krishnan. Prioritized epidemic routing for opportunistic networks. In *Proceedings of the 1st International MobiSys Workshop on Mobile Opportunistic Networking*, pages 62–66, San Juan, Puerto Rico, 2007. ACM.

[147] 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)*, pages 1109 –1117, sept. 2007.

[148] C. Radhakrishna Rao. The use and interpretation of principal component analysis in applied research. *The Indian Journal of Statistics*, 26(4):pp. 329–358, 1964.

[149] MIT Technology Review. 10 emerging technologies that will change the world. http://www2.technologyreview.com/emtech/.

[150] Carl A. Richmond. Fireflies flashing in unison. *Science*, 71(1847):537–538, 1930.

[151] M. B. V. Roberts. *Biology: a functional approach.* Nelson, 1986.

[152] A. Rowe, D. Goel, and R. Rajkumar. FireFly mosaic: A vision-enabled wireless sensor networking system. In *Proceedings of the 28th IEEE International Real-Time Systems Symposium (RTSS)*, pages 459–468, December 2007.

[153] Narayanan Sadagopan, Mitali Singh, and Bhaskar Krishnamachari. Decentralized utility-based sensor network design. *Mobile Networks and Applications*, 11(3):341–350, 2006.

[154] Kashif Saleem and Norsheila Fisal. Bio-inspired self-organized secure autonomous routing protocol for WSN. In *Proceedings of the IEEE International RF and Microwave Conference, 2008. (RFM)*, pages 417–421. IEEE, 2008.

[155] Kashif Saleem, Norsheila Fisal, Muhammad Sharil Abdullah, A. B. Zulkarmwan, Sharifah Hafizah, and Sharifah Kamilah. Proposed nature inspired self-organized secure autonomous mechanism for WSNs. In *Proceedings of the First Asian Conference onIntelligent Information and Database Systems, 2009. (ACIIDS)*, pages 277–282. IEEE, 2009.

[156] Muhammad Saleem and Muddassar Farooq. Beesensor: A bee-inspired power aware routing protocol for wireless sensor networks. In Mario Giacobini, editor, *Applications of Evolutionary Computing*, volume 4448 of *Lecture Notes in Computer Science*, pages 81–90. Springer Berlin Heidelberg, 2007.

[157] Paolo Santi. Topology control in wireless ad hoc and sensor networks. *ACM Computing Surveys*, 37(2):164–194, Jun. 2005.

[158] S. Sarafijanovic and J.-Y. Le Boudec. An artificial immune system approach with secondary response for misbehavior detection in mobile ad hoc networks. *IEEE Transactions on Neural Networks*, 16(5):1076–1087, September 2005.

[159] Slaviša Sarafijanović and Jean-Yves Boudec. An artificial immune system for misbehavior detection in mobile ad-hoc networks with virtual thymus, clustering, danger signal, and memory detectors. In Giuseppe Nicosia, Vincenzo Cutello, PeterJ. Bentley, and Jon Timmis, editors, *Artificial Immune Systems*, volume 3239 of *Lecture Notes in Computer Science*, pages 342–356. Springer Berlin Heidelberg, January 2004.

[160] J. Senthilkumar and M. Chandrasekaran. Improving the performance of wireless sensor network using bee's mating intelligence. 55, 2011.

[161] Srinivas Sethi and Siba K. Udgata. The efficient ant routing protocol for MANET. *International Journal on Computer Science and Engineering*, 2(07):2414–2420, 2010.

[162] S.D. Shirkande and R.A. Vatti. ACO based routing algorithms for ad-hoc network (WSN, MANETs): a survey. In *Proceedings of the 2013 International Conference on Communication Systems and Network Technologies (CSNT)*, pages 230–235, April 2013.

[163] Lei Shu, Chun Wu, Yan Zhang, Jiming Chen, Lei 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)*, volume 1, pages 17 –20, dec. 2008.

[164] S. Slijepcevic and M. Potkonjak. Power efficient organization of wireless sensor networks. In *Proceedings of the IEEE International Conference on Communications*, volume 2, pages 472 –476 vol.2, 2001.

[165] Ahmed Sobeih, Wei-Peng Chen, Jennifer C. Hou, Lu-Chuan Kung, Ning Li, Hyuk Lim, Hung-Ying Tyan, and Honghai Zhang. J-sim: A simulation environment for wireless sensor networks. In *Proceedings of the 38th Annual Symposium on Simulation*, pages 175–187. IEEE Computer Society, 2005.

[166] L. P. Steyn and G. P. Hancke. A survey of wireless sensor network testbeds. In *Proceedings of the 2011 AFRICON*, pages 1–6. IEEE, 2011.

[167] Yanjun Sun, Omer Gurewitz, and David B. Johnson. RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. In *Proceedings of the 6th ACM Conference on Embedded Network Ssensor Systems*, pages 1–14. ACM, 2008.

[168] Sameer Sundresh, Wooyoung Kim, and Gul Agha. SENS: a sensor, environment and network simulator. In *Proceedings of the 37th Annual Symposium on Simulation*, page 221. IEEE Computer Society, 2004.

[169] P. Szczytowski, A. Khelil, A. Ali, and N. Suri. TOM: topology oriented maintenance in sparse wireless sensor networks. In *Proceedings of the 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pages 548–556, June 2011.

[170] George Theodorakopoulos, J.-Y. Le Boudec, and John S. Baras. Selfish response to epidemic propagation. In *Proceedings of the American Control Conference (ACC)*, pages 4069–4074. IEEE, 2011.

[171] J. Timmis, M. Neal, and J. Hunt. An artificial immune system for data analysis. *BioSystems*, 55(1-3):143–150, 2000.

[172] BL Titzer, DK Lee, and J Palsberg. Avrora: scalable sensor network simulation with precise timing. pages 477–482. IEEE Press, April 2005.

[173] Bin Tong, Zi Li, Guiling Wang, and Wensheng Zhang. On-demand node reclamation and replacement for guaranteed area coverage in long-lived sensor networks. In Novella Bartolini, Sotiris Nikoletseas, Prasun Sinha, Valeria Cardellini, and Anirban Mahanti, editors, *Quality of Service in Heterogeneous Networks*, volume 22 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 148–166. Springer Berlin Heidelberg, January 2009.

[174] Nigel Tracey, John Clark, and Keith Mander. Automated program flaw finding using simulated annealing. *SIGSOFT Software Engineering Notes*, 23(2):73–81, March 1998.

[175] Wolfgang Trumler, Tobias Thiemann, and Theo Ungerer. An artificial hormone system for self-organization of networked nodes. In *Biologically Inspired Cooperative Computing*, volume 216, pages 85–94. Springer, US, 2006.

[176] Yuh-Ren Tsai. Coverage-preserving routing protocols for randomly distributed wireless sensor networks. *IEEE Transactions on Wireless Communications*, 6(4):1240–1245, April 2007.

[177] D. Turgut, B. Turgut, R. Elmasri, and T.V. Le. Optimizing clustering algorithm in mobile ad hoc networks using simulated annealing. In *Proceedings of the IEEE Conference on Wireless Communications and Networking*, volume 3, pages 1492 –1497, march 2003.

[178] A. Tyrrell and G. Auer. Imposing a reference timing onto firefly synchronization in wireless networks. In *Proceedings of the 65th IEEE Vehicular Technology Conference (VTC)*, pages 222–226, April 2007.

[179] A. Tyrrell, G. Auer, and C. Bettstetter. Fireflies as role models for synchronization in ad hoc networks. *Bio-Inspired Models of Network, Information and Computing Systems, 2006. 1st*, pages 1–7, December 2006.

[180] Amin Vahdat and David Becker. Epidemic routing for partially connected ad hoc networks. Technical report, Technical Report CS-200006, Duke University, 2000.

[181] Andras Varga. OMNeT++ modeling and tools for network simulation. In Klaus Wehrle, Mesut Güneş, and James Gross, editors, *Modeling and Tools for Network Simulation*, pages 35–59. Springer Berlin Heidelberg, 2010.

[182] J.E. Veenstra and R.J. Fowler. MINT: a front end for efficient simulation of shared-memory multiprocessors. In *Proceedings of the 2nd International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 201–207, February 1994.

[183] Feng Wang and Jiangchuan Liu. Duty-cycle-aware broadcast in wireless sensor networks. In *IEEE INFOCOM*, pages 468–476. IEEE, 2009.

[184] M. Wang and T. Suda. The bio-networking architecture: a biologically inspired approach to the design of scalable, adaptive, and survivable/available network applications. In *Proceedings of the 2001 Symposium on Applications and the Internet*, pages 43–53, 2001.

[185] Tsang-Yi Wang, Y.S. Han, P.K. Varshney, and Po-Ning Chen. Distributed fault-tolerant classification in wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 23(4):724 – 734, april 2005.

[186] Horst F. Wedde, Muddassar Farooq, and Yue Zhang. Beehive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior. In Marco Dorigo, Mauro Birattari, Christian Blum, LucaMaria Gambardella, Francesco

Mondada, and Thomas Stützle, editors, *Ant Colony Optimization and Swarm Intelligence*, volume 3172 of *Lecture Notes in Computer Science*, pages 83–94. Springer Berlin Heidelberg, 2004.

[187] Elias Weingartner, Hendrik Vom Lehn, and Klaus Wehrle. A performance comparison of recent network simulators. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 1–5. IEEE, 2009.

[188] G. Werner-Allen, P. Swieskowski, and M. Welsh. MoteLab: a wireless sensor network testbed. In *Fourth International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 483–488, April 2005.

[189] Geoffrey Werner-Allen, Geetika Tewari, Ankit Patel, Matt Welsh, and Radhika Nagpal. Firefly-inspired sensor network synchronicity with realistic radio effects. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 142–153. ACM, 2005.

[190] A Wieder and B.B. Brandenburg. Efficient partitioning of sporadic real-time tasks with shared resources and spin locks. In *Proceedings of the 8th IEEE International Symposium on Industrial Embedded Systems (SIES)*, pages 49–58, 2013.

[191] Alan F.T. Winfield, Jin Sa, Mari-Carmen Fernández-Gago, Clare Dixon, and Michael Fisher. On formal specification of emergent behaviours in swarm robotic systems. *International Journal of Advanced Robotic Systems*, 2(4):363–370, 2005.

[192] Jie Wu and Fei Dai. Efficient broadcasting with guaranteed coverage in mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 4(3):259–270, 2005.

[193] Xiaodong Xian, Weiren Shi, and He Huang. Comparison of OMNET++ and other simulator for WSN simulation. In *Proceedings of the 3rd IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 1439–1443. IEEE, 2008.

[194] Lin Xiao, Mikael Johansson, and Stephen P. Boyd. Simultaneous routing and resource allocation via dual decomposition. *IEEE Transactions on Communications*, 52(7):1136–1144, 2004.

[195] Xiao-Min Hu and Jun Zhang. Ant routing optimization algorithm for extending the lifetime of wireless sensor networks. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 738–744, October 2010.

[196] Xiaolu Liu and Shumin Zhou. Evaluation of several time synchronization protocols in WSN. *Information Science and Management Engineering (ISME), 2010 International Conference of*, 1:488–491, August 2010.

[197] J. Yackovich, D. Mosse, A. Rowe, and R. Rajkumar. Making WSN TDMA practical: Stealing slots up and down the tree. In *Proceedings of the 17th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, volume 1, pages 41–50, August 2011.

[198] Wei Ye, John Heidemann, and Deborah Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of the 21st IEEE Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1567–1576. IEEE, 2002.

[199] Sangho Yi, Hong Min, Junyoung Heo, Boncheol Gu, Yookun Cho, Jiman Hong, Jinwon Kim, Kwangyong Lee, and Seungmin Park. Performance analysis of task schedulers in operating systems for wireless sensor networks. In MarinaL. Gavrilova, Osvaldo Gervasi, Vipin Kumar, C.J.Kenneth Tan, David Taniar, Antonio Laganá, Youngsong Mun, and Hyunseung Choo, editors, *Computational Science and Its Applications - ICCSA 2006*, volume 3983 of *Lecture Notes in Computer Science*, pages 499–508. Springer Berlin Heidelberg, January 2006.

[200] Yi Sun, Qing Jiang, and Kai Zhang. A clustering scheme for reachback firefly synchronicity in wireless sensor networks. In *Proceedings of the 3rd IEEE*

International Conference on Network Infrastructure and Digital Content (IC-NIDC), pages 27–31, September 2012.

[201] Yingzhuang Liu, Hong Zhang, Qiang Ni, Zongyi Zhou, and Guangxi Zhu. An effective ant-colony based routing algorithm for mobile ad-hoc network. In *Proceedings of the 4th IEEE International Conference on Circuits and Systems for Communications (ICCSC)*, pages 100–103, May 2008.

[202] O. Younis and S. Fahmy. HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing*, 3(4):366–379, Oct.-Dec. 2004.

[203] Jun Yuan and Wei Yu. WSN11-1: distributed cross-layer optimization of wireless sensor networks: A game theoretic approach. In *Proceedings of the IEEE Global Telecommunications Conference, 2006. (GLOBECOM)*, pages 1–5. IEEE, 2006.

[204] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: a library for parallel simulation of large-scale wireless networks. In *Proceedings of the 12th Workshop on Parallel and Distributed Simulation (PADS)*, pages 154–161.

[205] Zhiwen Zeng, Anfeng Liu, Deng Li, and Jun Long. A highly efficient dag task scheduling algorithm for wireless sensor networks. In *Proceedings of the 9th International Conference for Young Computer Scientists (ICYCS)*, pages 570–575, Nov. 2008.

[206] Jiliang Zhou, Qiying Cao, Caixia Li, and Runcai Huang. A genetic algorithm based on extended sequence and topology encoding for the multicast protocol in two-tiered WSN. *Expert Systems with Applications*, 37(2):1684–1695, 2010.

[207] Hubert Zimmermann. OSI reference model–the ISO model of architecture for open systems interconnection. *IEEE Transactions on Communications*, 28(4):425–432, 1980.