Adaptive Task Selection using Threshold-based Techniques in Dynamic Sensor Networks

Author: Wesam S. Haboush *Supervisor:* David H. Shrimpton

07 June 2008

Abstract

Sensor nodes, like many social insect species, exist in harsh environments in large groups, yet possess very limited amount of resources. Lasting for as long as possible, and fulfilling the network purposes are the ultimate goals of sensor networks. However, these goals are inherently contradictory. Nature can be a great source of inspiration for mankind to find methods to achieve both extended survival, and effective operation. This work aims at applying the threshold-based action selection mechanisms inspired from insect societies to perform action selection within sensor nodes. The effect of this micro-model on the macro-behaviour of the network is studied in terms of durability and task performance quality. Generally, this is an example of using bio-inspiration to achieve adaptivity in sensor networks.

Contents

Chapter 1

Introduction and Background

1.1 Sensor Networks

Sensor networks are a category of networks that is characteristically distinct from conventional computer networks. Individual members of a sensor network are called *sensor nodes*, or *nodes* for short. Nodes in a sensor network are comparatively small in size, limited in resources, and often deployed in large numbers. Limitations in resources available to a sensor node can be in the form of small-sized memory, modest processing power, and restricted energy sources [117]. Such limitations combined with the unique methods and manners of network deployment impact the communication and control models and mechanisms employed within sensor networks [60].

Nodes within a sensor network can be heterogeneous or homogeneous [133]. This may be determined by several factors including design time decisions, failure of node components, cost issues, or any combination of these factors.

Sensor networks are often utilised to monitor the dynamics of unknown environments especially those of unpredictable and/or highly irregular nature. Usually, a sensor node monitors phenomena within very close proximity to itself. Such a mode of operation allows highly localised data collection with a great level of detail. The fuzziness and continuous change in such environments lend themselves to sensor network monitoring. Environment models would normally provide an adequately accurate view of a

phenomena without the need for continuous monitoring. For example, climate models allowed mankind to predict climate activity in different areas of the globe with adequate accuracy without high-resolution monitoring, especially if those patterns were repetitive and periodic [185]. Therefore, the availability of a model suffices to allow fair level of knowledge about a phenomenon in the past, present, and future. In sensor network applications, the situation is often substantially different. This is due to many reasons, for example, it is difficult at best to build a model of an unexplored environment. In some other sensor network applications, it is challenging to build a predictable model of the environment, due to its complexity for instance, which further complicates monitoring tasks.

Examples of sensor network applications include early alarm systems that can be used around active volcanoes, seismic hot spots, dangerous material sites, and forests to report fire breaks. Military surveillance applications can also benefit from sensor networks. For example, sensor networks can monitor and report enemy vehicle movements or enemy personnel activity. Monitoring natural phenomena and collecting information for scientific studies is also another application domain that can benefit from sensor networks. For example, habitat monitoring, weather monitoring, or ocean deep waters monitoring. Finally, and most recently, great interest arose in pollution monitoring and vehicle movement management, with the increase of global warming and fuel crises. There are many other domains and applications for sensor networks [201] [97] that we do not have the time or space to extensively list or discuss in this dissertation. However, since we are talking about applications, we take this opportunity to briefly outline the application scenario we will be discussing throughout this document. A sensor node can have a number of sensing devices to detect various measurements from the surrounding environment. The readings from a sensing device on a sensor node can be used to infer different higher-level information items, which in turn can be useful in different applications. For example, temperature sensor readings may be used to predict or monitor weather conditions, and also can be use to detect fire breaks. Another example can be wind sensors which can be used to predict rain (via cloud movement patterns), and also the direction of the spread of a fire.

Given this possibility of reusing sensor readings, it seems very plausible to imagine a sensor network that is multipurpose [172]. Therefore, instead of deploying a sensor network to perform weather monitoring, and another sensor network to perform, say, air pollution monitoring, we can deploy a single network that would do both tasks. This will require the nodes of this network to have capabilities of sensing relevant measurements for both applications. Some of these measurements may overlap, i.e.. same sensing devices may supply data to multiple applications. Based on this argument, this thesis was motivated, and hence our application scenario is a multi-tasked network. It involves a network that does three different tasks, *Traffic Monitoring*, *Pollution Monitoring*, and *Urban Environmental Monitoring*. More details about the application scenario will be given later in the dissertation as we progress through the chapters.

1.2 Sensor Network Challenges

The attention given to sensor networks by both research and industrial communities is due mainly to the development and evolution of enabling technologies. For example, the miniaturisation of many sensing components like cameras, communication components like transmitters, and processing components like CPU units enabled manufacturers to make small cheap sensor nodes [12]. In addition, different components of a node can now operate with a fraction of the power profile it conventionally required. This is especially important for CPUs and communication devices. The prospects are opening wider and wider as enabling technologies advance further with every passing day.

While sensor nodes offer advantages like fault tolerance, scalability, high availability, and low cost, they also present many challenges [29]. Here is a list of questions and challenges that sensor networks present:

Control mechanisms: control in sensor networks is often distributed which adds to the complexity. Distribution of control is dictated by the fact that inter-node communication is mostly expensive and unreliable [61].

Communication mechanisms and protocols inter-node communication in sensor networks is unreliable and ad-hoc in nature [170]. Nodes communicate with peer nodes within their vicinity to conserve energy and usually use multi-hop communication structures. Those characteristics of sensor networks require new protocols and mechanisms to be used as conventional networking protocols were not designed with these characteristics in mind. For example, TCP require reliable communication and live connections for data transmission, which is often unavailable in sensor networks.

Collective decision making, inter-node collaboration, coordination, and cooperation: nodes need to optimise their collective resources to perform network goals. Yet, nodes can not rely on a central entity to assign tasks, schedule activities, or coordinate execution of plans because this often reduces fault tolerance, which is fundamental to sensor networks. New control mechanisms, likely distributed, are needed to cope in such situation [61].

Dynamism: the environments where sensor networks are deployed are mostly inhospitable, unexplored or unpredictable [115]. The dynamic nature of the environment requires a degree of intelligence or adaptability on both the macro and micro-levels of the network in order to achieve network requirements from a human-user point of view with a reasonable degree of cost-effectiveness.

Security: given the large size of sensor networks, and the nature of the environments they are deployed within, security is a great issue in sensor networks [145]. Sensor nodes could simply be tampered with physically, or electronically which could consequently result in security breaches, unreliability, or distorted view of the target phenomenon. This is often critical in military and business applications.

Generally the term sensor network is overloaded in the scientific literature and can cover a wide variety of networks with a wide variety of characteristics and criteria [26]. As this dissertation is mainly concerned with a particular type of sensor networks, we need to clearly define what

constitutes a sensor network from our perspective - at least because we cannot cover all of them for reasons of space, time, and resources. Such a definition helps greatly and clearly identify the problem being researched in this dissertation and narrow down or specify the scope of the research applicability. Example criteria and characterisations include heterogeneous vs. homogeneous, mobile vs. static, active vs. passive, and so on [183]. In this dissertation, our target networks are generally very resource-constrained. Although in next the chapter, we will cover in more detail a wide section of the sensor network literature, the family of networks focused on in this dissertation is identified here by the constraints and characteristics in the following list:

1. Nodes typically share a *finite communication bandwidth* and an unreliable communication medium. This implies that connectivity cannot be guaranteed or relied on [40]. This is a result of the environments where many sensor networks are deployed. Examples of such environments include disaster zones, battlefields, and harsh climates. Disaster zones could expose communication equipments to excessive heat that could damage them. In battlefields, communication jamming devices could be employed by enemies to disrupt the sensor network effective operation. Finally, harsh climates such as those in the vicinity of erupted volcanoes may impede communication amongst sensor nodes. Even if communication medium is reliable, communication may be impeded by the lack of resources required for it, such as power or bandwidth. If we look at our multi-tasking application, briefly introduced above, we can see that the urban environment where our network will be deployed is potentially full of communicating devices. This may lead to restricted access to communication channels and high media unreliability, i.e. constrained bandwidth [81].

2. Nodes in sensor networks may or may not have information about peers, i.e. *autonomous*. They do not critically rely on information about other node conditions, numbers, existence(or lack of), or behaviour in making decisions [207]. Again, this is due to the nature of the areas

and manner in which sensor networks are deployed. Deployment is usually ad hoc, and communication might be very limited. Obtaining peer-related information within such circumstances may be difficult, or sketchy at best. Therefore, strong reliance on such information is a risky approach. In our multi-tasking application, a node may not know what tasks its peers are performing, and so has to make a decision based on the information available to it.

3. The physical configuration of sensor nodes may be dynamic, but not voluntarily controlled [59] by any single node or group of nodes, i.e. nodes are *physically static* or *parasitically mobile*. Nodes are typically scattered over the target geographical area randomly, and cannot move independently [119]. Autonomously mobile nodes will have to consume a large amount of energy to achieve the desired geographical displacement [108]. This goes against the main predicament of sensor nodes being limited in resources. While voluntary mobility is assumed to be available in many robotic applications, this is not the case in sensor networks. In our application scenario, nodes are distributed over an urban environment more or less randomly and uniformly, and statically.

4. *Fault tolerance* is an essential requirement of sensor networks considering the dynamic, uncertain, and probably hostile environments in which they are usually deployed [41] [28]. This is advantageous in case the communication medium, sensing devices, or whole nodes fail. In our scenario, nodes will be deployed densely to allow fault tolerance against node failures. Nodes should be able to make decisions even if communication is unavailable, which achieves fault tolerance against communication media unreliability.

5. Nodes in a sensor network can be *heterogeneous* [38]. Heterogeneity can be a system characteristic at design time, or a result of changes in some device configuration or user policies at runtime. It also could happen as node parts fail or degrade their performance due to wear and tear factors. We will experiment with homogeneous nodes in

this dissertation, but nodes behaviour may make them seem virtually heterogeneous. There is nothing in this work that is solely aimed at homogeneous [10] sensor networks. In our application scenario, nodes are deployed more or less *homogeneous* in their capabilities. However, wear and tear and other environmental or manufacturing factors is likely to result in heterogeneity developing. Our work does not depend on homogeneity of the nodes to operate successfully.

6. Task preemption is often not possible as nodes are assumed to have a minimalist build [72]. In other words, because nodes are relatively resource-constrained (limited energy, memory, CPU power, sensory capacity, etc), they can perform only one task at a time [167], i.e. they are *single-tasked* and the scheduling is *non-preemptive*. Multi-tasking could introduce complexity in the infrastructure serving the high-level applications [19] such as memory , CPU, and power management components. Also applications could work on conflicting agendas, for example if a sensor node trying to monitor different targets moving in opposite directions. Complexity in the node infrastructure may result in increased node production cost and longer design, development, testing, and therefore deployment, time lengths. In our scenario, detailed later in the dissertation, nodes perform tasks with no preemption, and only one of the three high-level tasks is performed at a time.

7. No assumptions about future task requirements or task scheduling can be made by nodes in a sensor network [191]. Therefore, *dynamic on-the-fly decision* making is an essential characteristic of any proposed control mechanism [91] [92] [141]. This comes as a result of the uncertainly surrounding the availability of communication or information from other network members in addition to the dynamic nature of the deployment environment and manner. In our application scenario, nodes make decision dynamically and on the fly depending on their context, i.e. resource availability, application requirements, and user policies.

8. Nodes in sensor networks cannot directly affect, alter, or modify the environment, i.e. they are *passive*. Their main function is to report or monitor events that occur in their locality [2]. This is a significant difference between a sensor node and a robot. In fact, this often complicates matters in sensor networks as they miss out on the benefits of the environment's feedback on the effect of the network activity. This almost reverses the role of the environment from a target of change by nodes to a driver of change of nodes [82] [188]. Clearly, in our application scenario, there is no environment alterations by the nodes. Node simply monitor, analyse, and report Traffic, Pollution, and Weather-related data.

9. The *resource-constrained* [126] nature of the nodes in a sensor network has repercussions on the nodes' ability to gather, process, and communicate data. Therefore, it is necessary to accomplish the goals of the network with minimum possible resource consumption [54] [7]. This requirement is very much linked to many of the other requirements such as unreliable communication, large number of nodes deployed, non-preemptive tasking model, heterogeneity, and static physical configuration or involuntary mobility. Nodes in our application scenario are assumed to be as minimalistic as possible.

10. Sensor networks are *large-scale* in terms of the number of nodes [58] in a network. Therefore, scalability is crucial in any proposed control mechanism [160]. This requirement support the fault tolerance requirement above. Our network scenario employs a dense sensor network to obtain fine-grained monitoring capabilities.

1.3 Why use biologically-inspired solutions ?

The differences between conventional computer networks, and sensor networks motivated researchers to look for new solutions to cope with the requirements and challenges introduced by the latter [153]. Nature has always been a source of inspiration and insight into new ideas to solve man problems [175]. Nature provides working solutions that were tried

and proved to work to solve natural problems successfully for hundreds and thousands of years [15]. Fields that have used or drawn from natural systems include, but not limited to, genetic algorithms, cellular automata, emergent systems, neural networks, artificial life, artificial immune systems, and many more [43] [66] [101].

The complexity and and sheer size of sensor networks is observed to resemble that of some societies or groups that exist in nature. These include schools of fish, flocks of bird, and ant colonies. Scientists try to understand the mechanisms used by such organisms and replicate them or adapt them to solve sensor network problems [32]. In this dissertation, we adopt this approach by taking a biologically inspired solution, and explore the employment of this solution in sensor network scenarios.

1.4 Dissertation Focus and Research Question

The dissertation examines the advantages and disadvantages of the employment of adaptive bio-inspired solutions in sensor networks. In particular, the threshold-based models are tackled as a generic adaptive solution to sensor network problems. The main issues that entail from such a research activity include:

- How to apply threshold-based algorithms to solve sensor network problems?
- What points or areas within a sensor node that can benefit from the application of threshold-based algorithms?
- How to extend the existent threshold-based models to account for factors that were not accounted for in the biology modeling literature?
- What side-effects does the application of threshold-based models, in more than one point within a sensor node, have on the dynamics of the system or node behaviour?
- What patterns of mapping between the node-level micro-rules and the macro-level behaviour of the whole network can be observed and

hence utilised to create simple micro-level solutions to a high-level network-wide requirements?

- What metrics can be used to give an estimate of the quality of a solution to a sensor network problem, particularly the networks characterised in this dissertation?

1.5 Contributions

The following are the contributions of this work:

- The application of a threshold-based model to solve the problem of action selection and control in sensor networks.
- The extension of the threshold-based model to account for network/node resources in making a variety of decisions.
- Identifying the major points where adaptability can be beneficial if introduced within a sensor node.
- Providing a general node architecture where a sensor node is divided into modules and layers that map to adaptability points and control structures.

1.6 Dissertation Organisation

In chapter 1, we gave a background to the work presented in this thesis covering general information about sensor networks, biologically inspired solutions, and how they fit in the context of this dissertation. We also introduced our application scenario in this chapter. Chapter 2 will discuss the work done in areas pertinent to the subject of this thesis, including task allocation, bio-inspired solutions, and sensor node architectures.

chapter 3 will present and discuss a general sensor node architecture that we use as a basis for introducing adaptability within a sensor node in later chapters.

Chapters 4 and 5 will focus on adaptability for task allocation and action selection. In particular, chapter 4 will present the theory and math behind the application of threshold-based models to regulate task allocation and action selection within a sensor network, while chapter 5 will contain the experiments followed by an analysis and discussion.

In chapter 6, the theory and math behind deploying a threshold-based model for task discontinuation in sensor networks is presented. This is followed, in chapter 7, by the experiments with respect to such deployment, and finally an analysis and discussion of the results.

The subject of chapter 8 is the use of threshold-based models to control sensing/sampling activities within sensor nodes. The following chapter, number 9, introduces the experiments in connection with sampling followed by results, analysis, and discussion.

Communication and threshold-based models are discussed and introduced in chapter 10, while chapter 11 contains the experiments using the ideas presented in chapter 10. Also chapter 11 discusses and analyses the results of the experiments.

Finally, chapter 12 is where we summarise and conclude the dissertation and highlight future work.

Chapter 2

Related Work

There has been a great deal of work on task allocation in systems that share common characteristics with sensor networks such as robot teams [176] and clustered processor grids [202]. The intersection between these systems is often in the number of nodes, agents, or units within a system and the need to coordinate among them. Robot teams seem to bear the highest resemblance to sensor networks and therefore will be extensively covered in this chapter.

Robot teams are similar to sensor networks in many ways. Like sensor nodes, individual robots in a team often share a communication medium that is unreliable with limited bandwidth. Also many robotic applications, like those of sensor networks, require fault tolerant mechanisms and distributed algorithms. Nonetheless, robots differ from sensor nodes in some fundamental aspects. Firstly, most robots are voluntarily mobile, whereas sensor nodes are often not. Secondly, robots for the most part can alter the environment in one way or another, while sensor nodes are typically passive monitoring devices. Thirdly, with robots, the energy requirements of sensing and communication are often marginal in comparison to those of mobility, and therefore, robots can sense and communicate with relative liberty. With sensor nodes, sensing and communication may be considered excessively power-hungry activities, and they ought to be performed sparingly. These differences between robots and sensor nodes have implications on the efficiency of the mechanisms selected for task allocation and action

selection within sensor networks. More will be said about this in later chapters of the dissertation. Meanwhile, this chapter will focus on the literature of the work undertaken in the areas of multi-tasking in robots and sensor networks, in addition to bio-inspired solutions.

2.1 Multi-Robot Task Allocation

Gage [67] and Baghaei [13] present an impressive survey of the literature on multi-task allocation in robots. Although each offer a different basis of classification for task allocation algorithms and strategies, they, combined, cover a big portion of the work done in this field and we use them as a general basis to discuss related work in this section.

In [67], strategies for the Multi-Robot Task Allocation Problem (MRTA) are classified into five categories. These are:

1. Motivation-based,
2. Mutual-inhibition,
3. Auction-based,
4. Team consensus, and
5. No allocation.

In [13], the following more specific algorithms for task allocation in multi-robot systems are explored:

1. Publish/Subscribe (MURDOCH),
2. Broadcast of Local Eligibility using Port Arbitration Behaviour,
3. Free Market Architecture for Distributed Control of a Multi-Robot System,
4. Auction Algorithms,
5. ALLIANCE,
6. Task Acquisition using Multiple Objective Behaviour Coordination,

7. Functionally-Accurate Cooperative (FA/C) Distributed Problem Solving,
8. Distributed Multi-Robot Task Allocation for Emergency Handling,
9. Team Formation-based Task Allocation, and
10. Ants Algorithms.

In the following subsections, the above strategies and algorithms are discussed and evaluated in the context of sensor networks. The aim is to gauge their suitability to the specific characteristics and requirements of sensor networks listed in section 1.2.

2.1.1 Auction-based strategy and the MURDOCH Algorithm

In auction-based strategies, task requests arrive at a node called *auctioneer*. Auctioneers could receive tasks from users or detect them from the surrounding environmental conditions. The auctioneer then broadcasts, to other nodes, a request to perform each task it receives or detects. Nodes, in turn, reply by communicating their bids back to the auctioneer. A *bid* is a measure of a node's fitness to perform the task in question. Bids are more specifically dependent on the domain and application of the network. The auctioneer, after collecting the different bids, assigns the task to the most suitable node based on some application-specific criteria incorporated in the bids, such as task performance quality, least cost, or best performance. In other words, the highest bidder is notified by the auctioneer to perform the task under bidding. Tasks can be discovered by the auctioneer, for example a target emerges, or could be given to it by other nodes or human users.

Before we discuss the suitability of auction-based strategies to sensor network task allocation, It is useful for comparison and contrast purposes to introduce an algorithm that bears high resemblance to auction-based strategies, namely MURDOCH [73]. In [67], MURDOCH is viewed as a variation of an auction-based algorithm, whereas in [13], it is viewed as a Publish/Subscribe algorithm. We tend to agree with the latter classification

for two reasons. The first is that in a Publish/Subscribe scheme, task requests are often selectively multi-cast to nodes which previously subscribed to the relevant type of task request. In auction-based algorithms, task requests are indiscriminately broadcast, and so there is no subscription mechanism employed, but rather a process of dynamic discovery of bidders. In addition, in MURDOCH, task allocation happens in a distributed fashion by team negotiation, whereas in auction-based systems, a central entity, often named "auctioneer", performs the arbitration. However, in [76], MURDOCH practically resorted back to an auctioneer-like entity to perform the final task assignment, which makes it appear as a variation of the auction-based strategy. Because of these similarities, we opt for discussing both the auction-based strategy and the Publish/Subscribe algorithm of MURDOCH together in this section.

In auction-based strategies, reliable communication is an essential requirement, while it is seen as a scarce resource in many sensor networks, given the often inhospitable deployment environment and the only-wireless communication capabilities of sensor nodes. Even though it is said in [74] that "*Robots can communicate but messages may be lost*", this loss is highly restrictive and has to be minimal for the system to operate effectively. If the communication loss is high, the bidding process will be often crippled and the task assignment process will be only marginally driven by the fitness metrics of the application. In [73], a more detailed example is given but, for reasons of time and space, we will not discuss it fully here. In addition to the reliance on communication, the requirement of an auctioneer monitoring the winner's progress in performing the task is communication-intensive, incurring an energy cost that is often too expensive to tolerate in a sensor network.

Also, highly unreliable communication media may result in unpredictable auctioneer behaviours. Thus, auction-based and Publish/Subscribe strategies do not address constraint 1 in section 1.2 (Communication Bandwidth and Reliability).

Furthermore, although there are distributed variations of the auction-based strategy, clustered auctioning for instance, they all share a degree of centrality in their operation. This is because there is always an auction-

eer which performs, as the central entity, the arbitration amongst bidders. For many sensor networks, this centrality reduces fault tolerance (requirement 4 in section 1.2), and is susceptible to communication channel failures (constraint 1 in section 1.2 regarding Communication Bandwidth and Reliability). Generally, the less coupling between peer nodes, the more robust the system is in the face of failures.

Additionally, some auction-based algorithms, MURDOCH [73] for instance, require the auctioneer to track the task performance capabilities of all nodes. This fundamentally reduces the scalability of the solution (criteria 2 in terms of communication and 10 in terms of scalability in section 1.2).

Moreover, in sensor networks, tasks are often connected to the locality of the node. Since auction-based algorithms depend on the auctioneer to issue task requests and assign tasks to bidders, the network can operate in one of three possible scenarios. The first is to leave the task detection to the auctioneers, who later assign the tasks they detect to bidders. This scenario restricts geographical coverage to auctioneers' locality, possibly leaving some locations uncovered, and/or resulting in inferior sensing resolutions. Even worse, the second option may require the auctioneer to possess significant capabilities to cover wide geographical areas, which is expensive and fault-intolerant. These options for the auctioneers' mode of operation are typically undesirable in sensor networks where high resolution monitoring and wide geographical coverage are often application requirements. The third and last scenario is to have each node sense or detect tasks, and forward findings to the auctioneer which then in turn performs the conventional auctioning process, i.e. auctioneers only arbitrate. This latter central-control choice is extremely demanding in terms of communication media reliability and bandwidth capacity. This is at odds with constraint 1 (Communication Bandwidth and Reliability). Additionally, heavy communication is often associated with high power consumption, which breaks requirement 9 (Resource Consumption) in section 1.2.

Finally, some auction-based algorithms, e.g. [73], assume that the auctioneer can monitor the performance of a task assigned to a node. This either means the node can alter the environment in a way that the auction-

eer can detect, which is against constraint 8 (Environmental Alteration and Feedback) in section 1.2, or, as discussed earlier, more communication to report task utility will be necessary.

The auction-based strategy, like other strategies discussed later, does not address the questions of how the individual nodes or the auctioneer will obtain the information based on which the task allocation decisions are made. For example, in a sensor network, is it viable to send long-haul communication messages to local nodes about required tasks? Under what conditions a node can use its sensors to obtain information about task requests? To what extent (e.g.: considering cost versus benefit) should task allocation procedures, such as task discontinuation or task engagement be conducted? in this dissertation, we try to answer these questions and experiment with models that can be used to address this kind of issues.

2.1.2 Motivation-based Strategy and ALLIANCE

Motivation-based approaches assume that nodes are working towards goals and that they can detect and assess how efficient the progress, made by the network, towards these goals is. Such assumption necessitates, at least conceptually, a node's capability to alter the environment, a feature that is unavailable in many sensor networks as highlighted in constraint 8 (Environmental Alteration and Feedback). Even if this was viable, it heightens the computational load on individual nodes. For a node monitoring itself is seen as a means of assessing and guiding a node's actions to achieve a goal, and is not the goal itself. If the goal of a network is monitoring, then there is no need for feedback, as once a node perform the monitoring, it immediately and already is able to establish it has succeeded in achieving so much towards the monitoring goal.

Motivation-based algorithms are suitable for applications with quality of task performances that can be represented by binary or boolean variables. Each of the two possible values of the variables represents one of two possible states of a node. These are either performing a task, or not. In sensor networks, task performance is often associated with a measure of quality. This measure is application-specific but typically would include the

accuracy of sampled data, data delivery timeliness, sampling resolution, or a combination of these. For example, if the task was monitoring temperature variation in a node's locality, a motivational approach would involve the node to either report temperature readings when there are sufficient resources, or otherwise not report readings at all. In sensor network applications, it might be advantageous to vary the rate of reading and reporting instead of totally halting a task. Sensor networks sometimes require a continuous spectrum of task performance quality rather than a discrete binary values space. If we look at our application scenario, it would be beneficial to allow a node to perform the *Traffic Monitoring Task* with a certain level of vigour depending on how complex, exacerbated, or heavy the traffic situation is. A node may dedicate 25% of its time to perform the *Traffic Monitoring Task* if there are indications of a possible traffic jam as opposed to an already developed one.

Motivation-based algorithms do not address the issue of restrictions on resources (constraint 9 in section 1.2). In sensor networks, these restrictions are fundamental because nodes are often expected to live for a long time, and so need to use available resources sparingly. It is typically preferred in sensor networks to only provide an acceptable task performance and last for long periods of time, rather than provide a high-quality performance and die off quickly. For example, in our application scenario, a network that can perform Weather Monitoring Task for a long period of time, say 3 years, may be preferred to a network that provides high-precision daily weather measurements for a short period of time, for example 2 months. This preference is based on reducing the financial cost and management overhead involved in frequent deployment of such large scale networks. Note that monitoring for , say, 3 years is not a goal, it is a behaviour. Providing data for as long as possible is more likely the goal.

ALLIANCE [144], as an example of a motivation-based algorithm, does not consider task prioritisation. It assumes that any behaviour can inhibit any other behaviour. This may be because the tasks addressed in ALLIANCE are fairly inter-dependent and require cooperative and/or complementary actions, therefore any one task is just as important as any other in order to finally fulfil the networks high level goals. In [144], this goal

was lifting a set of boxes from a location into a truck. The tasks were: 1) moving boxes closer to the back of the truck, and 2) lifting the close boxes onto the truck deck. The absence of either of these tasks results in overall failure in achieving the network goal. In sensor networks, some monitoring activities may intrinsically bear greater importance than others. For example, in a sensor network that is deployed to monitor traffic jams, and air pollution levels, reporting a traffic jam on a vital highway may be more important than reporting slight variations in pollution levels during the rush hour. In some situations, the latter can be done in the absence of emergency conditions.

ALLIANCE does not satisfy constraints on resource consumption in three ways. First, nodes in ALLIANCE frequently broadcast their internal status, resulting in heavy communication load. Such resource consumption should be limited in sensor networks. Second, task preemption is allowed in ALLIANCE, which incurs the extra computational load and complexity associated with multitasking. Third, tasks in ALLIANCE are organised in complex structures, called behavioural sets or high-level task-achieving functions. Each behavioural set includes a number of low-level subtasks belonging to it. Each behavioural set serves one network motivation (high-level task or goal). This tasking model contributes to the computational complexity of individual nodes. Sensor nodes, unlike ALLIANCE, typically can only communicate sparingly. Computationally, simple task representations are preferred, and therefore task preemption is unlikely to be the mechanism of choice.

Nodes in ALLIANCE communicate directly with each other, and each node can communicate directly with every other node in the network [144]. While this may be valid in small to medium-size networks, it lacks scalability to accommodate large-scale sensor networks - refer to constraint 10 (scalability) in section 1.2. Such global communication is very demanding in both energy and channel bandwidth.

Although ALLIANCE does not address the issues encountered in sensor networks, it can be adapted to deal with many of them. For example, the global network communication could be restricted to function only within a local radius. Decisions could be made based on only local samples of data

and node status, ignoring other team members if they are unavailable. The details of such adaptation are beyond the scope of this work.

2.1.3 Mutual Inhibition Strategy and Broadcast of Local Eligibility (BLE)

In mutual inhibition strategies, such as those in [194] and [130], a node has a behaviour associated with each task it can perform, e.g. the searching the terrain behaviour associated with the *Search Task*, or the behaviour of observing a target associated with the *Observe Task*. The fitness of a node to perform a task is known as its task eligibility. A node's eligibility is a scalar that represents the available amount of resources required to perform this task. Resources that have impact on this value are domain- and application-specific. For example, in [194], the relevant eligibility criteria was the distance from a target. They may include memory, battery power, communication capabilities, geographical location, or speed. In [194], the eligibility for the application used to validate the algorithm was the geographical location, as nodes which could better observe targets were generally more eligible to assume the task of monitoring that target. Each node assesses its eligibility with respect to each detected task, and then broadcasts it to other network nodes. A node then compares its eligibility for each task with those of its peers. If a node finds itself the most eligible with respect to a task, it inhibits the behaviour associated with this task on all other nodes by periodically broadcasting inhibitory messages, thus claiming the task. If the node fails at any point, i.e. no longer can perform a task, it stops broadcasting its inhibitory messages, which is interpreted by other nodes as its failure, and a new best-eligibility node is elected to perform the task and inhibit other nodes.

In such a mechanism, the communication load is high because the active inhibition process is implemented as continuous broadcasting. In addition, Werger [194] states that: "*Up to the limit of communication bandwidth, any number of BLE-enabled robots can be added to a system and properly interact*". This statement illustrates how fundamental communication is for the strategy, going against constraint 1 in section 1.2. The strategy is essentially built

on the assumption of communication channel reliability. If communication fails frequently, as can occur in sensor networks, so does the mutual inhibition mechanism as premature eligibility elections may occur, or confusion, resulting from communication reliability fluctuation, could lead to chaos in the system.

In [194] and [130], there was no consideration for the resource expenditure required to produce varying qualities of task performances. The authors only aimed at improving the quality of network performance, i.e. achieving best network coverage in their example applications, regardless of the cost this improvement may incur in terms of resource consumption. This breaks constraint 9 in section 1.2 and so is unsuited to the domain of sensor networks.

The BLE strategy does not address the tradeoff between the resource cost of obtaining the information needed to calculate the local eligibility and the associated gain to the network performance as a whole. Such cost is critical to task allocation procedures as well as network performance, and especially so in sensor networks where resources are scarce. The communication model in [194] and [130] is global, which makes it generally inappropriate for the domain of sensor networks, where communication is an expensive activity in terms of energy, and where channel reliability cannot be guaranteed. Sensor networks are better suited to unreliable localised communication schemes.

2.1.4 Team Consensus Strategy

According to *team consensus strategy*, all network nodes reach a consensus on a formation or a task allocation configuration prior to deployment. Various mechanisms can be used to realise this consensus including negotiation, communication, or sharing common models. For example, the algorithm in [174] assumes periodic communication with unlimited bandwidth and a guaranteed channel reliability to realise team consensus. This algorithm also assumes strong inter-task dependency which requires direct communication as a means of coordination amongst network nodes, without which the network goal cannot be attained. As in some other strategies, Stone in

[174] makes use of global communication capabilities, which, in many sensor networks, cannot be guaranteed. In addition, nodes can be recovered, and stay offline for a short period of time in order to synchronise their internal states, then be redeployed into the field again. These assumptions often do not apply in sensor networks, which are frequently deployed in dangerous inhospitable environments, and it is unlikely that it will be feasible to un-deploy and then re-deploy them.

Stone [174] also states that his algorithm is for time-critical team work. This is not always the case in sensor networks, and especially is not the case in our application scenario. Finally, although agents are supposed to act autonomously, there is a clear coupling between the behaviour of different agents and the success of the network. The author concludes that coordination is paramount for the goal of the network to be achieved, and severe degradation can result from the communication medium unreliability or latency. This is mentioned more than once in [174].

The algorithms in [94] and [95] employ adaptive models. They adopt behavioural adaptivity where a node's behaviour is driven by information gathered from the environment, peers, and internal state. However, they do not employ fine-grain adaptivity mechanisms, but rather a series of coarse grain pre-programmed rules only on the level of action selection. For example, a node can be foraging, or not foraging as opposed to foraging with a 20% capacity and foraging with a 95% capacity. In addition, a node's behavioural controllers are deterministic which restricts its adaptive capabilities to anticipated pre-configured situations.

Both algorithms in [94] and [95] also ignore the cost of collecting information and performing tasks, concentrating rather on improving the network performance. The algorithms do not use any form of global communication among nodes, which scales very well. However, nodes use an application-specific local communication mechanism, in the form of coloured lights fixed on node peaks to indicate their current status to other nodes in the vicinity. This form of communication requires clear line of sight to function effectively, which is mostly unavailable in forests, battlefields, disaster zones, etc. Despite this, it could be argued that applications often will allow one form of communication or another. Creativity and innovation plays a

great role in this space, and this is what the work in [94] and [95] have picked from fireflies.

Finally, the networks in [94] and [95] are constantly performing tasks without any recruitment process, which may waste resources if task performance is redundant. This is critical for many sensor networks because, for example, such mode of operation may unnecessarily deplete the batteries of a node or waste its memory space. In our example scenario, if a node keeps analysing traffic information although there is no traffic for the past 12 hours, battery, CPU cycles, and memory can be wasted.

2.1.5 The No Allocation Strategy

The no allocation strategy is given that name because all nodes perform the same task, and so there is no need to assign tasks, perform recruitment or action selection. In most monitoring applications targeted by sensor networks, only some events are interesting. For example, a sensor network to monitor seismic activity is more interested in the few occasions when earthquakes happen. In disaster recovery applications, only hazardous events may be interesting (fires, floods, smoke, etc). In structure safety applications, only changes in the safety metrics of a building or a structure may be requested by human users. If we turn our attention to our application scenario in this dissertation, nodes are only interested in recording traffic statistics, pollution variations, and weather measurements when rapid changes occur. This would help, for example, prevent or analyse traffic jams, reduce pollution, and predict weather conditions.

In the 'moving furniture application' of Rus [51], communication is used for synchronisation and coordination amongst robots, without which the network will most likely fail to achieve its goals. In addition, the algorithm is not scalable as there is always one central device that controls the system. This also applies to [102] where a central control location is required for the network to function. Central control is generally an undesirable feature in many applications of sensor networks because it lacks fault tolerance and scalability required in such distributed systems.

None of the no-allocation algorithms we could find account for the

cost of the task allocation enabling processes. These processes include communicating with the control point, sensing and collecting data from the environment, and switching states from idle to active or vice versa.

2.1.6 Free Market Architecture for Distributed Control of a Multi-Robot System

Stentz [173] proposes an architecture inspired from free market economies to manage multi-tasking in groups of robots. According to this architecture, each node has cost and revenue functions that determine the net profit it may achieve by performing a certain task. Tasks are divided into subtasks that nodes negotiate and coordinate to perform. To facilitate negotiations, the algorithm requires the availability of low-bandwidth communication at all times, which does not satisfy requirement 1 in section 1.2. Robots that cannot communicate may not be able to perform tasks as they are not part of an economy, until they reconnect to the network. Although Stentz [173] views the free economy as a production-boosting mechanism, we think that it does not lend itself to many applications of multi-robot systems and sensor networks. For example, a node in a sensor network might find it compulsory to perform a task even if it was not lucrative to itself to do so. A node that needs to issue a warning of an impending dam collapse cannot but issue a warning regardless of any benefit/loss calculations. In our example application, if sensors detect a huge rise in pollution levels, this may mean a hazardous chemical leak has occurred, which then would oblige the node to report it as soon as possible no matter what the cost involved is and away from any individual level gains or losses. Also a node may contribute to achieving network goals by, for example, performing a resource-exhausting task to relieve weak neighboring nodes, regardless of the profit it reaps. Generally, nodes in sensor networks take decisions to further the network goals rather than its own benefit, unlike decisions taken by individuals in free market economies, where selfish micro-economies drive the macro-economy.

2.1.7 Conclusion

In the past few sections, we addressed general task allocation strategies and algorithms that mainly targeted multi-robot teams. Robot teams are similar to sensor nodes in several aspects. Both are inherently distributed, and involve the cooperation, communication, and coordination amongst multiple autonomous units to achieve an overall goal. Many of the applications of both sensor network and robot teams share similar requirements, such as autonomy, self-organisation, data management, task allocation, etc.

Despite those similarities between robots and sensors, prominent differences between the two group types can be identified. First, sensor networks addressed in this paper are immobile. This has many implications on the weight placed on several other activities that are essential to sensor network operation. For example, communication cost in terms of energy consumption can be marginal compared to the cost associated with mobility. In sensor networks with no voluntary mobility, communication power requirements may be vast in comparison to power requirements of other activities such as sampling the environment. Also, sensor networks are computationally minimal, which requires simple control mechanisms to perform task allocation and coordination. A robot typically possesses a higher computational power than that of a sensor node. For these reasons, different task allocation and action selection algorithms may be needed for sensor networks from those used in multi-robot systems. In the next section, we will discuss some task allocation algorithms tailored to accommodate some characteristics that are specific to sensor networks.

2.2 Task allocation Strategies for Sensor Networks

The algorithm in [121] was designed for mobile sensor networks. It uses a combination of node interactions, stigmergy, threshold-based, and dominance hierarchy models to make task allocation decisions. A task is monitoring a certain geographical area in the application provided in [121] and [122]. While mobility is a characteristic that is often associated with robots, rather than sensor networks, there is no clear-cut line between the two fields

as some platforms fall in the grey area in between by combining characteristics from both paradigms, which we believe is the case in Low's work [121].

The algorithm in [121] focuses on collaborative sensing which depends fundamentally on inter-node communication and so effectively lends itself to problems with strong inter-task dependencies, such as target-tracking. The algorithm aims primarily at maximising the quality of network performance, in this case network coverage, but does not account for the resource cost involved. Low [121] discourages static sensor placement, however, we believe that there are cases when this is a viable, or even inevitable, solution. For example when mobility results in exhausting intolerable amounts of the network's energy. Low [121] states explicitly that he has studied the problem of task allocation in a robotic context, which may explain the algorithm's many discrepancies with our constraints in section 1.2. Low [121] also states that the type of problems addressed are those that involve high task interdependency. In contrast to our definition of a sensor network, Low [121] excludes situations where a node can autonomously and adequately perform a single task. Low [121] assumes that *task interference* may adversely affect the network performance. Task interference is the situation where too many robots decide to perform the same task, resulting in a physical congestion or control overheads that may cause the quality of task performance to rather deteriorate instead of improve. This may be alleviated by static sensor placement and low inter-node dependencies where possible. Communication in the algorithm takes place periodically every N time steps to reduce task interference. This represents a fixed resource cost even under low task demands, yet it is essential for this algorithm to function satisfactorily. Performances of nodes are compared through exchange of dominance messages, and losers of dominance interactions are less likely to leave the current region. Dominance messages are a representation of biologically-inspired phenomenon observed in many animal and insect species to determine right of control of territories or other resources. The system in [121] was tested on predefined geographical regions, i.e. there is an internal representation of the environment. Such knowledge of the environment is likely to be unavailable to nodes in a typical applica-

tion of sensor networks. Finally, the algorithm does not perform any task prioritisation within a node, a feature that may be desirable in monitoring applications like those often performed by sensor networks.

Yu [204] considers a cluster-based approach where communication is global, i.e. single-hop network. Yu [204] considers collaborative processing where inter-task dependencies are high and assume synchronisation is readily available within the cluster. A task (application) in this algorithm can be performed within an epoch using a TDMA-like slotting technique. The algorithm employs Exclusive Access Constraint, i.e. non-preemptive task performance and single-sender communication channels are in use. An application-driven task allocation is used in contrast to an environment-driven one where the environmental conditions perceived by a node determine a node's behaviour. A central control model is adopted within a cluster, i.e. a cluster head or a similar single entity is implied to oversee the clustering and task assignment procedures.

Low [122] addresses the problem of network coverage using an ant-based algorithm, namely a threshold-based algorithm, bearing resemblance to our work in this dissertation. The sensors are mobile and checkpoints or beacons are employed to guide their mobility. The authors do not address situations of independent task performance or single-node tasks, but instead they assume a strong task inter-dependence. The algorithm attempts to achieve coordination without direct communication. However, kin recognition, which can be seen as a form of communication, is employed to provide nodes within a region with information about the status of peers within their vicinity. Predetermined regions, like those in [121], were utilised which is equivalent to embedding a model of the environment into the node prior to deployment. Communication is periodic and vital for this algorithm to function, which goes against constraints 1 and 4 in section 1.2. In this dissertation, we found that our ant-based algorithm achieves a better network coverage with less energy expenditure on mobility, i.e. it is more energy-efficient.

Younis [203] studied clustered networks, which have an inherent degree of centralisation within a cluster's vicinity while a degree of distribution is achieved on a macro-level. This task allocation algorithm is designed to

work on the cluster heads level, not the node level. Cluster heads receive commands to perform tasks from a central location called the command node. This can be a base station, a user laptop, or a satellite server. The command node performs arbitration among cluster heads. This is a hierarchical system with an explicit mechanism for centralised control employed. Younis [203] assumes that the power requirements of communication between any two cluster heads is constant. The algorithm depends on communication among cluster heads, and thus fails to function in environments where communication is unreliable. The command node centrally performs task allocation using an application-specific optimisation algorithm, for example in [203], the *simulated annealing* optimisation method was found appropriate by the authors. Simulating annealing is a physically inspired solution, where good solutions are found by searching a large search space for local minima/maxima by sifting through different randomly found local ones [100].

Modi in [135] assumes distributed tasks (high task inter-dependency), which necessitates communication. He does not address fault tolerance and scalability issues. The algorithm focuses on maximising the quality of task performance, without taking the associated resource cost into account. Modi [135] follows a formal technique that although useful in understanding the problem, makes assumptions that do not hold in real world applications. For example, he assumes the reliability of agents and communication media which is very unlikely in the sensor networks domain. Although our simulations make a similar assumption regarding the reliability of sensors, we believe that it will scale well because sensors are autonomous and densely deployed. We plan in future work to test our results on unreliable, more realistic models, or even real sensor nodes/networks.

Tian [182] addresses single hop teams, with multi-task applications. Again, clustered networks are the main focus in this work, which requires both communication reliability and central control. In addition, this sense of centrality imposes additional energy requirements due to additional communication and computation overheads. In [182], tasks are predetermined and not dynamic. Tian addresses only homogeneous systems and adopts a design-time scheduling algorithm, which goes against the dynamic char-

acteristics of our target environment (see constraint 7 in section 1.2). In addition, the tasks are highly inter-dependent and the algorithm controls task-associated low-level attributes, such as CPU cycles and communication scheduling. This may be beneficial, but it is not the subject of this work as we focus on the control at a higher level of abstraction.

Oliveira [52] uses a feedback mechanism to solve the action selection and sequencing problems. He does not address any issues related to communication, distribution, or scalability. In addition, the experiments in [52] were performed on a centralised application where tasks are performed by an agent after it was assigned to it by a scheduler on a central controller. The scheduler receives task requests and performs the task allocation centrally, which is likely inapplicable in sensor networks (see constraints 1,2,4,9, and 10 in section 1.2).

2.3 Bio-inspired solutions

Nature has inspired mankind for long time. This started from fairly simple activities like cooking, or effect of fire on food, to intricate problems that mankind, unlike nature, could not yet find solutions for. Computing literature is full of such problems where nature excelled in finding a solution while humans have failed for centuries [1] [21] [36]. Bio-inspired solutions span a wide research area, however, almost every work in this space can be classified under one of the following three categories: 1) Genetics and evolutionary computing, 2) Artificial Life, and 3) Swarm Intelligence.

The first of these aspire to find solutions to complex problems via search algorithms derived from models and mechanisms from human-genetics, such as mutations and global heuristics [79] [134]. Evolutionary computing has been applied to a variety of optimisation and search-space problems. Genetic algorithms, GA for short, are the most popular form of evolutionary algorithms that has been used extensively even in optimising its own parameters, such as the rate of mutation, the generation size, and the selection model or criteria [65] [195]. We discuss in the conclusion chapter of this dissertation the idea of using GAs to tune the parameters of our model. Many studies has focused on using these algorithms in optimis-

Algorithm or Strategy	Constraint									
	(1) Communication	(2) Network Topology	(3) Mobility	(4) Fault Tolerance	(5) Heterogeneity	(6) Task Preemption	(7) Static Scheduling	(8) Environment Alteration	(9) Resource scarcity	(10) Scalability
Auction-based and MURDOCH	X	X		X				X	X	X
Motivation-based and ALLIANCE	X	X				X		X	X	X
Mutual Inhibition and BLE	X	X		X					X	X
Team Consensus	X	X							X	
No Allocation	X	X			X		X		X	X
Market-based and Economics	X			X					X	
Low 2004	X	X	X		X				X	X
Yu 2005	X	X		X			X		X	X
Low 2005	X	X	X		X				X	
Younis 2003	X	X		X						X
Modi 2002	X	X		X					X	X
Tian 2005	X	X		X	X		X			X
Oliviera 2004	X	X		X				X	X	X

Table 2.1: Comparison of different Task Allocation Approaches. Note: Low 2004[121], Low 2005[122], Yu 2005[204], Younis 2003[203], Modi 2002[135], Tian 2005[182], Oliviera 2004[52]

ing neural networks and various types of multi-agent systems [200] [118] [208]. GAs were also used in software engineering and system management work. For example, Tripathi's work [186] uses GAs to address problems of inter-module communication within a system of multiple processors.

The second research field in the list is concerned with replicating life in its original form as opposed to only borrowing the mechanisms employed by living systems [110]. Not only has Artificial Life researchers tried to create robots that possess human intelligence, but also robots that possess characteristics that are conventionally exclusive to humans, such as laughing at jokes, feeling in love or angry, suffering tiredness and pain, and even learning social skills and making friends. The AL field had achieved varied degrees of success but was, still is, and probably will always be a highly controversial topic [89].

The third and last item on the list, swarm intelligence, is the study of the behaviour of groups of living organisms in order to understand and hopefully make use of emergent complex patterns out of relatively simple individuals. In this section we will focus on swarm intelligence as it is the most relevant to our work and as sensor networks fall in this category of systems in one way or another.

Research in swarm intelligence is subdivided into various connected subjects each attempting to answer one or more of the following questions: 1) What is the effect of the micro-behaviour associated with members of the swarm on the macro-behaviour of the group as a whole, 2) What are the protocols or communication mechanisms that are sufficient and feasible to be utilised in swarming groups, and 3) What parameters control the behaviour of the swarm leading to certain steady-states or end states like convergence, disintegration, chaos, or self-organisation 4) How to map micro-behaviours to obtain a set of guaranteed outcomes observed on the macro-view of the system. In the next few paragraphs, we will present some of the work in the field of bio-inspired solutions that tried to answer one or more of these questions. We will focus naturally on sensor networks research as it is the target field of this dissertation.

In the field of sensor networks and robotics, nature-inspired ideas have been a major source of solutions. For example, in [35], a routing algorithm

drawn from ant colonies behaviour was employed. In this algorithm, called *AntNet*, routes were discovered and compared through a set of agents that constantly tour the network. Agents communicated via a concept called *stigmergy*, which is equivalent to using the environment as a medium for communication instead of direct communication. In [199], brood sorting algorithms observed in some ant species, specifically the *Leptothorax*, were used to sort objects using a group of minimalist robots. Wilson [199] used genetic algorithms to tune the parameters of his biological model, i.e. he again drew from biology. This is more of a robotics work in our opinion as it involves mobile agents. In [113], the coalition formation mechanisms drawn from primates and insects, for example [158], were used to regulate task allocation in robots and sensor networks. Li [113] applied his algorithm on a team of UAVs (Unmanned Attack Vehicles). He also used concepts from game theory to guarantee the stability of the system. The stability in a coalition formation context means the certainty that no individual member of a coalition will deviate from the team's goal. Li's algorithm aimed at maximising the lifetime of the team by minimising the resource depletion experienced by any single member of the team, while achieving the team goals through cooperation among the members.

Low [121] uses three mechanisms to regulate task allocation in a group of sensing robots. These are ant foraging mechanism through pheromone laying, threshold-based techniques, and the social dominance interactions within insect societies. A similar algorithm, but with neural network mechanisms utilised for tuning agent's behaviour is provided in [122]. Both [121] and [122] have been discussed in more detail in previous sections of this chapter.

In [159], algorithms based on game theory for coalition formation are presented and analysed extensively. Game theory is a type of multi-agent systems where single members are selfish and try to maximise their own return rather than care for the group in total. The author identifies the type of system he focuses on as not necessarily *super-additive*. This means that collaboration amongst agents does not automatically mean a gain for the group, however, it can be so. We see any system with autonomous individuals as a multi-agent system (MAS for short), while Shehory [159]

calls super-additive systems, i.e. collaborating agents for the good of the group and not the individual, a *Distributed Problem Solving system*, DPS for short. In [23], multi-robot coordination is achieved through an algorithm inspired from the nest-building behaviour of some wasp species. In [205] and [168], physics-based algorithms are used to control sensor networks and robots respectively.

In [33], a method for assigning tasks or resources, based on a model of division of labour in social insects, is introduced and applied to a dynamic flow shop scheduling problem. The problem consists of assigning trucks to paint booths in a truck facility to minimise total makespan and the number of paint flushes. Similarities between the ant-based approach and a market-based approach are highlighted and the authors found that both systems are able to adapt well to changing conditions. Note that even market-based algorithms can be seen as biologically inspired systems because humans are what constitutes markets and so human behaviour is a major player in the dynamics of the system.

In [37], brain cells learning mechanisms were replicated in a sensor network to allow decentralised perception of a phenomena. In [8], an architecture which is very much biologically inspired according to the authors, is used for control, coordination, and action selection within a robot. AuRa [8] uses the Schema Theory as the fundamental basis of his architecture, in addition to incorporating various psychological, physical, and genetic theories. In [69], Galstyan provides a stochastic analysis of a task allocation mechanism that does not need inter-agent communication at all. This no-communication mode of operation is very similar to what we adopt in this dissertation, excluding the part that investigates the communication issues (chapter 10 and chapter 11).

In [132], a wall-building application used an ant-inspired algorithm for coordination and cooperation among a group of robots. In [102], task allocation and worker recruitment for foraging were achieved through, again, an ant-inspired mechanism. In [107], the concepts of self-organisation for task allocation without the need for communication is explored. Marshall [152] used bacteria-inspired genetic algorithm to perform network management services. Sacks in [154] describes how to build a sensor network platform

using a set of biologically inspired solutions.

Britton [29] presents an approach for designing wireless sensor networks. He literally advocates treating these networks as biology-like systems in terms of their structure and the characteristics required for their survival. In [29] also the kOS is evaluated for its fitness to satisfy sensor network's OS requirements. Oliviera [52] presents a process to generate, adapt, and change multi-agent organisation dynamically at system runtime, using a swarm inspired approach, especially for task allocation without preplanning or explicit coordination. In [186], a genetic algorithm is used to regulate task allocation considering loads on the network and the individual nodes.

2.4 Sensor Node Architectures

Sensor nodes are where the adaptive algorithms will be employed as they constitute the micro-view of a network. A look at what research has been conducted in terms of a sensor node's architecture is in order. We present some of the work done in this area here, and in the following chapter, as we present our adopted architectural view of a sensor node, we will discuss some of them in more detail.

In [9], Asada presents a hardware architecture that is compact and low-power for the construction of a sensor node. The architecture is dubbed *WINS*. Nodes adopt a continuous sensing and event-based detection models. WINS applications are typically latency-tolerant and of the monitoring family. In WINS, sensors are low-power and MEMS-based (Microelectromechanical Systems). Signal processing is performed through a low-power spectrum analyser. A micro-power RF system is utilised for multi-hop internode communication.

Avancha [11] proposes a functional component-based architecture, as opposed to a hardware-based one, called *SWANS*. The main components of the architecture are Monitoring and Reporting component, Logic Component and Action Component. The Monitoring and Reporting component performs the application monitoring tasks given sensor states, network goals, and sensor ontology. The Logic component computes the state of

other components given a set of parameters associated with each. Finally, the Action component decides how and when a node will move from one state to another.

The architecture in [46] is composed of four modules: 1. interpreter 2. widgets/sensors 3. application 4. aggregator . These modules are in fact different levels of abstraction. The sensor-widget pair is composed of a low level sensor, and a higher-level widget. The latter translates the low-level readings from the former into a value that can be used by other modules. Interpreters take readings from widgets and applications and translate them into higher- and lower-level information respectively. Aggregators collect information from different widgets in one location to be retrieved by relevant applications.

Fitzpatrick in [64] demonstrates a software architecture by the name of *Sentient Object Model*. Sentient objects are made of three main components: 1. event consumer 2. inference rules 3. event producer . Each of the three components is composed of the same three subcomponents, i.e. event consumers, rules, and event producers. On the higher-level view of the architecture, a sensor node has a component that consumes events from the environment. It then uses inference rules or control logic to decide on appropriate actions. Decisions are forwarded to actuators, i.e. the event producer component, which in turn puts actions in effect. Within an event consumer, there are the same three components: an event consumer that receives events from the environment, which then is forwarded to a control subcomponent that decides which inference engine to forward the data to, and then forwards that decision to an event producer subcomponent. Similar logic applies for the high-level event producer and inference engine components.

Verissimo in [189] extends the architecture in [64], focussing on real-time applications support. Verissimo's architecture is given the name *GEAR* (short for Generic Event-based ARchitecture). GEAR uses message-driven communication paradigm, where events are filtered through *translation layers* and then sent out to event subscribers. *Event Channels* are in charge of propagating the different events across the whole architecture. There is also the *Communication Layer* that is responsible for transporting events between

different sentient objects in the system.

Farinelli in [62] presents a layered robot architecture composed of three layers. The lowest of them is called the *Operative Layer*. Within this layer, modules representing actuators and sensors exist together with world model data. Above this layer resides the *On-line Deliberative Layer*. This layer contains high-level descriptions of the environment inferred from the lower-level world model in the layer below. It also include a *Plan Execution Module* and a *Coordination Module*. Finally, at the top of the stack lies the *Off-line Deliberative Layer*. This layer contains a library of plans for task execution, a high-level knowledge base, and a module for *Plan Generation*. The knowledge base is given to the robot before deployment, then fed to the plan generation module which in turn generates a set of plans to be executed according to the situation on the ground detected by lower layers of the architecture. Robots use the plan library combined with data from the environment and peer robots to decide on which plan to execute.

Gage [68] designs a robot architecture, called SFX, based on a formal emotions model referred to as *OCC* model (short for the names of the three authors of the model, Ortony, Clore, and Collins). The SFX architecture is composed of three layers, namely deliberative, managerial, and reactive. Each of these layers contains a number of modules. The details of this work is beyond the scope of this dissertation.

In [70], again, a layered architecture is proposed, where each layer uses the data provided by lower layers to produce coarser view of the environment, which is very similar to the approach we take in our architecture in the next chapter. Handziski [84] suggests a hardware-based architecture implementing the ISO stack to a great extent. Hill [86] also proposes a hardware architecture composed mainly of a set of sensors, a wireless communication port, and a micro-controller.

Mascolo [129] provides a large survey of many software architectures of mobile computing, which include robot teams, sensor networks, and mobile devices.

2.5 Summary

This chapter gave a literature coverage in the subjects pertinent to this dissertation. It started by covering the multi-tasking problem widely, because it is the most important aspect of this thesis. Then the chapter high-lighted the bio-inspired computing paradigm and gave examples of work done in this field. Finally, sensor nodes and network architectures research examples were provided.

In the next chapter, an architectural view of sensor networks and nodes from our perspective will be presented, relating and comparing it with some of the work introduced in this chapter.

Chapter 3

Sensor Node Model and Network Architecture

In this chapter, we will discuss a general view of a sensor network from an architectural and functional viewpoints. Based on these views, we will present our sensor node model that will be adopted in the rest of the dissertation.

3.1 Introduction

The main goal of this chapter is introducing an *adaptable* sensor node and sensor network architectures. The concept of adaptivity in sensor networks is inspired by biological theories about species survival and adaptive behaviour. Biologists have found compelling ecological evidence that the capability of different species of living organisms to continue to exist over extended periods of time is attributable to their highly adaptive behaviours. The success of these species was often despite living in harsh and highly uncertain environments. In the same manner, the extinction of many species can be explained by their failure to adapt and, consequently, being overpowered by adverse environmental conditions. The success of adaptive behaviour in living organisms have led us to speculations, encouraged by previous success stories in other fields, that introducing adaptive strategies to sensor networks may yield similarly desirable outcomes in overcoming

the challenges of environmental dynamism.

Before looking into what adaptive algorithm or model we employ, we need to characterise the sensor network requirements that would fall within the space we target in this dissertation. This will be the topic of the next section.

3.2 Network Requirements

It is possible to summarise, rather simplistically, the ideal goals of a sensor network to be: 1. Nodes should remain functional for as long as possible, optimally forever 2. Nodes should work to provide highest possible quality of service to achieve the network's overall goal . These two goals are inherently contradictory, and so a middle solution needs to be found and some compromises and *prioritisation* of goals need to be done. This is exactly what adaptive algorithms are needed for. Nodes in adaptive sensor networks should be able to dynamically react to changes they can detect in the local environment and determine at which point in the wide spectrum between the two contradicting goals of the network a node should operate. Such an adaptive strategy will also lead to the automation of many activities, otherwise performed by humans. This characteristic is called *self-organisation*, and allows for a great *scalability* which is another important requirement for sensor networks. For example, instead of a human operator signalling sensor nodes to reduce temperature reading rate when he/she notices lack of interesting variations, network nodes could have adaptive rules to detect temperature variations and adjust the temperature sampling rate accordingly. In general, minimal human intervention is desirable, if not necessary, in many sensor network applications.

Nodes in a sensor network should not depend on their knowledge of the environment, or neighboring nodes to perform their tasks. *Autonomy* can be added then to the list of requirements of a sensor network. The network task performance quality should not severely deteriorate if some nodes fail, i.e. the network needs to be *fault tolerant*. This is supported, from a node-level point view, by the autonomous node model. The deployment of large number of nodes, following the ecological rule "safety in numbers",

achieves fault tolerance from a network viewpoint.

Network's support for *heterogeneity* is viewed as a main requirement for sensor networks that are addressed in this dissertation. Heterogeneity could result from design-time decisions or from operational circumstances, such as the failure of some sensors on a node. Although it might be inevitable that a node has at some point to communicate with a base station, either by single- or multi-hop communication, nodes should not fail to perform their tasks if communication channels fail or are unavailable temporarily. This means that nodes should not depend on communication with peers to make decisions, however, that does not mean that nodes will never communicate with sinks of the data or decisions they made. Based on this argument, we can say that there are two types of communication by a node [183]: 1. infrastructure communication, used for coordination, making decisions, etc, 2. application data communication, which is reporting the finding of the node, a summary of its taken readings over a period of time, etc. Section 1.2 summarises the requirements and constraints of our target sensor networks.

In the next section, we will present the functional points at which adaptivity could be introduced to satisfy the requirements provided in this section.

3.3 Critical Functional Points of a Sensor Node

There are activities that are considered critical to the operation of a sensor network. The control mechanisms of these operations are potentially good candidates for the application of adaptive models. We view task allocation on the network level, and action selection on the node's level as one of those activities. This is particularly apparent when considering the complexity of sensor networks, and the resource-constrained nature of their building blocks, i.e. sensor nodes. In this document, a task is defined as a high-level/application-layer procedure, such as *Air Pollution Monitoring Task* application, rather than low level operating system processes. The latter may include authentication, packet routing, loading an application, communicating a data item, or reading a sensor. The resource requirements of a task or an action may significantly vary from one context or application

to another. For example, real-time data retrieval might be necessary for one application, e.g. seismic activity monitoring, while it might be superfluous to another such as long-term climate monitoring. The later is the type of applications we are interested in. In this dissertation, we study and analyse an adaptive mechanism as a sensor node's action selection/task allocation tool. We investigate the relationship between action select at the node level and the emergent task allocation process in the network as whole. We also examine the impact of action selection and task allocation processes on the lifetime and effectiveness of a sensor network. The aim here is to identify if these adaptive algorithms can dynamically find the balance points between the sensor network application requirements, and the set of constraints imposed by the context of the sensor node/network itself.

From the literature in sensor networks, there are four major actions, performed by the individual nodes, which play a significant role in the general operation of a sensor network and within the task allocation and action selection processes in particular. The actions are summarised in the following list, and more detailed explanation will follow:

1. **Sampling:** this means taking readings or sensing the environment. This activity is in the essence of a sensor network/node operation, hence the naming "sensor node" and "sensor network". Nodes use sensor readings to make decisions or simply report the readings directly or indirectly to users. Applications of sensor networks are inherently about sensing the environment, for example, life monitoring (ZebraNet [96], Great Duck Island [127]), environment monitoring (Glacier Monitoring [128], ARGO Ocean Monitoring [178], oceanography [146]), and others [153].

2. **Task Allocation/Action Selection:** This refers to the process of engaging in performing a high-level task or application[53] (could be termed *task engagement*, which is analogous to loading an application). Sensor nodes/networks would be ideally performing all the jobs tasked to them incessantly and vigorously. However, due to the restrictions imposed on the networks and nodes by the environment, costs, and their hardware/software capabilities, they cannot live up to such an ideal

[106]. Instead, nodes/networks have to plan, prioritise, and sometimes make compromises in order to best serve the user/application requirements within the limits of the available resources [55]. This is why task allocation has been extensively studied and researched [3].

3. **Task discontinuation:** this is simply equivalent to quitting performing a task, and could be analogous to unloading an application. There is only few explicit studies [114] [44] [98] on the mechanisms and techniques used in making decisions about task discontinuation. However, implicitly, every action selection/task allocation scheme utilises one or more task discontinuation mechanism. The model in [25] uses a constant discontinuation rate, which is a crude non-adaptive scheme of discontinuation. We propose a different approach in chapter 6.

4. **Communication:** this could be with any other system entities (servers, neighbour nodes, base stations, etc). Communication is a very important activity in multi-agent systems, including sensor networks, as it is the means of information flow between various components of the system. Collaboration, coordination, cooperation, control, and management are all activities that most of the time depend on communication capabilities or at least are better facilitated through communication. This is why there is a massive research effort being dedicated to communication protocols [4], mechanisms [48], hardware/software designs [187], and other communication aspects.

Network nodes make decisions to start and discontinue performing tasks based on: *1*) locally available information about the phenomena they are monitoring, which is obtained by sampling the accessible environment, *2*) locally available information, obtained by communication, from peer nodes, *3*) the internal status of a node, which may include energy levels, memory, and sensor availability.

We propose that adaptivity, employed at each of the above points, can better the task allocation process by increasing its robustness, resilience, and self-management capabilities. Adaptivity can be incorporated within each of the four major actions driving the action selection and task alloca-

tion processes in many ways. In the rest of this dissertation, we study the applicability of adaptivity with respect to each of the actions, listed above, and explore its effects on the network performance and duration compared to the base case of systems based on non-adaptive approaches which are usually driven by statically preset parameters and fixed values for monitoring rates or periods. For example, a system that uses periodic sampling of the environment is non-adaptive since it uses a preset value for the period, frequency, or rate of sampling. A system that uses a sampling rate that varies according to various system, environment, and/or user-defined requirements and constrained is an adaptive system.

Before introducing adaptive algorithms to perform task allocation, identification of the layers of abstraction or modules at which these algorithms will work is in order. From previously developed architectures for sensor networks and nodes, such as those in [138] and [180], the shortly introduced two abstract models for our hypo thesized architecture were created. The next two sections will describe those models.

3.4 Abstract Generalised Network Architecture

Our network architectural view focuses on the data management aspect of a sensor network. This is because the type of networks we target in this thesis is that that mainly performs monitoring tasks. The key task of such networks is to collect and process *data* to produce *useful information*. Therefore, data management is such an important activity within those networks.

Usually, data collected by sensor networks are huge in size, however, only a fraction of the collected data items are useful or interesting to the network, based on its requirements set by users and designers. For example, *seismic activity monitoring* sensor networks may essentially aim at providing early alarms against earthquakes. Earthquakes usually happen in a sporadic manner, which means the network is most of the time reading silence so to speak. Only when earthquakes really happen, the data become interesting to humans and they need to be delivered as efficient and fast as possible. Same argument apply to forest-fire sensor networks, or structure

safety sensor networks. If we look at our application scenario, when jams occur in the vicinity of a node, then it has the responsibility to act by reporting the situation to a base station, and analyse the developments on the ground, possibly giving directions how to avoid this traffic jam spot. Same with the Air Pollution Monitoring Task, if pollution levels soar suddenly, then nodes need to react quickly and report back to a base station the type of chemical that is detected, the amount of variation, the pattern of change, etc.

Based on the previous argument, it seems unrealistic to continuously send data back to a user, or base station. Even exchanging data within a small locality seems wasteful if they are useless, uninteresting, or of little importance. Sometimes local communication of seemingly uninteresting data can help actually validate or change a node's view of the value of a piece of information. However, it is unlikely that long-distance communication will achieve the same effect because sensor nodes are inherently a reflection of the locality, and in a similar manner, a group of them within a vicinity are a reflection of this vicinity. On the contrary, far apart nodes give very low resolution information about the geographical distribution of a phenomenon, and this configuration does not lend itself to many sensor network applications. One of the main factors that also would deter us from designing networks that communication large volume of data over long distances is the energy cost involved. Sensor nodes have resource constrained systems, and energy is a very dear commodity. Communication is one of the most power-hungry activities that a node could perform in many sensor network applications. Accordingly, minimising communication is a very desirable characteristic. Also the small bandwidth makes large volume communication untenable in many sensor network scenarios.

In-network processing seems to be a commendable approach. Nodes can process the data before forwarding them. Processing data could involve filtering unimportant data and possibly discarding them, compressing data locally before forwarding them, abstracting data in a vicinity or summarising them, etc. This idea of abstracting data as you go higher up the stack of the network constitutes the basis of our data management view of a network.

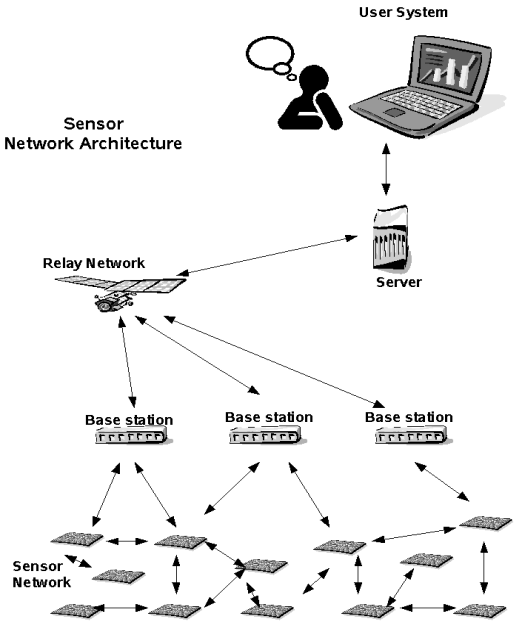

Figure 3.1: Abstract view of a sensor network architecture

To illustrate the idea, it is best to use an example while putting figure 3.2 in mind. Let's assume that we are deploying a sensor network to monitor the activity of an active volcano close to an important city. To identify a volcanic activity, nodes need to sample the air for certain chemical structures that are telltales of close eruptions. Also nodes could take readings of temperature of the surroundings, as they will grow higher because of the hot gases rushing out of the soil before an imminent eruption. Moreover, nodes may need to sample the soil for changes in chemicals that can help predict the status of this volcano. Now, we can have the sensor nodes communicate every temperature, soil, and air reading back to a base station that can analyse them and find out if an eruption is impending, or dormancy is the status quo. Another way to do this is to allow individual nodes, or small groups of nodes to examine and analyse the data they collected locally, and then send only conclusions to human users at the base stations. The latter solution will often involve much less communication and so will allow the network to have better longevity. In addition, it allows for more scalability as the distributed processing signifies.

Another example is our application scenario, where the a has three potential tasks to perform, namely Air Pollution Monitoring, Traffic Monitoring, and Weather Monitoring. All readings relevant to the three tasks can be sent to a base station to be analysed, and conclusions made in the base station. However, this will incur huge data communication, especially if compared to the data that would need to be transferred if the nodes made the processing, i.e. node would send a summary of findgins when this is necessary. That could be when a jam develops, or a hazardous chemical leak occurs, or if a storm telltales are found in the weather readings, and so on.

In figure 3.4, there is another possible application scenario depicted. Note the difference between the type of decisions taken at each of the three layers in this figure. What we want to emphasise here is that these layers do not necessarily lie on one device/node or another. For example, all layers can be implemented on one node, or on each node in a network, or on only some nodes on a network. A network could have many nodes implementing the lowermost layer, and only a base station that implements the upper two layers. Another configuration of a network may be with all nodes implementing the lower two layers, and a base station implementing the uppermost layer. Generally, any working combination is possible. In this dissertation, we will focus on networks with nodes that have the three layers implemented on them. However, we believe the adaptive algorithms presented in this dissertation can be applied to other layer configurations as well.

Our network view is generally similar to the architecture in [71], where data get coarser as it approaches the application layer, or higher up the networking stack or communication sinks. Figure 3.1 shows a typical architecture that can represent our view of a sensor network. In this section we discussed the network architecture. Next section will discuss that of a sensor node.

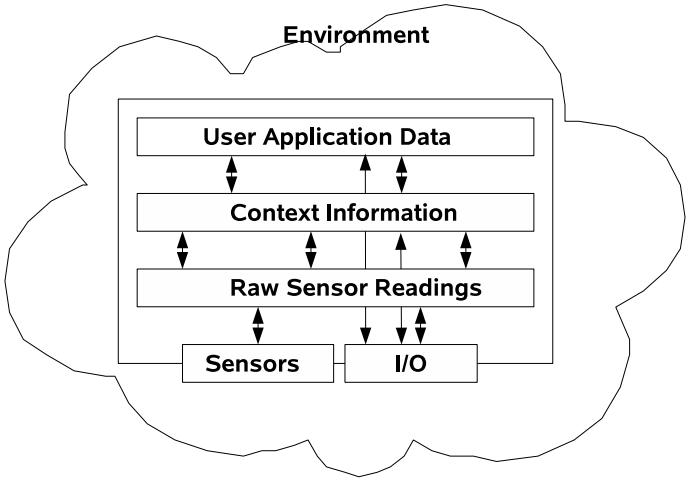

Figure 3.2: Abstract view of a sensor node architecture

3.5 Abstract Model of a Sensor Node

The architecture of a sensor node [47] can be looked at as a two dimensional entity. On the vertical dimension, a sensor node is viewed as a set of abstraction layers [180]. On the horizontal dimension, i.e. within each layer, a generally modular component-based approach is envisaged (and adopted) producing a set of modules or components [56]. Such an approach offers several advantages including: 1) flexibility, 2) reusability, 3) modularity, 4) extensibility, 5) portability, and 6) scalability [177]. It is also compatible with the way the network as a whole is viewed, which is very important in this context for several reasons. First and foremost, sensor networks vary in terms of the location of the implementation of the various network layers. There are no clear-cut boundaries within which the implementation of different layers is located. For example, the application layer might be implemented on each and every node within a sensor network, or alternatively be implemented only on a single base station. Figure 3.4 shows an example of such an application. The topmost layer may be on each and every node within a house in a certain implementation of an *Automatic House Control Sensor Network*. The same application may be implemented with one base station performing the high-level decision making, while nodes only perform the lower layer functionalities in the network stack, such as

aggregation, compression, and sampling. The same analogy applies to the Traffic Monitoring Task, in our application scenario, where the readings of all sensors on a node may be all forwarded to a base station that does the data mining for information about congestion probability or situation on the road. Alternatively, this application could be running on each node and the nodes in this case would only send high-level jam indicators individually or collectively to radio stations in order to be advertised to drivers or even to ambulance cars to allow them to avoid congested roads. A layered approach is also a used and well-studied design approach since the appearance of the ISO/OSI network model that dominates our world today. Adopting a modular approach within each layer allows us to break the problems within a layer into sub-problems. This is a well-known approach in software engineering, and complex systems design. This also helps our cause by allowing us to separate the functional points within an architecture clearly and study each separately.

The architecture we propose is event-based similar to the one in [138]. This is as opposed to a flow-based, pre-determined or pre-scheduled mechanism. Flow-based architectures have static routes for messages from one layer to another, from one node to another, or from one module to another. Pre-determined architectures refer to those that decide what type of information will be passed from one entity to another at design-time as opposed to at runtime. Finally, pre-scheduled architectures sends messages at pre-determined time points, such as periodically every five minutes. Event-based architectures make decision about the destination of messages, transmission time, and type of information to be transmitted at runtime. One approach to implement such an architecture in a sensor node is to delegate the event handling to the modules in the locality of the event. Therefore, sensors, for example, will handle sensory events, and communication modules will handle communication-related events, and so on. The sensing module, for instance, may contain a filtering functionality to detect interesting or eventful data. This will occur in a separate space of the software/hardware of a node. On figure 3.2, this could be the Raw Sensor Readings layer adaptively handling sensing events, allowing other modules to apply adaptive behaviours in different, may be higher-level,

modules of the architecture. The same argument can be applied to each layer of the architecture in figure 3.2.

Our architecture is mainly composed of four layers. The following is a description of each:

Sensor and I/O Layer: This is where the control of the lowest-level sensing devices and communication peripherals reside. This may correspond to the drivers of the physical, data link, and possibly network layers in the OSI networking model. However, It differs in that it does not only communicate to peer nodes, but also it can be seen as a provider of a dynamic view or model of the world at a very fine detail. It greatly influences the decision making process in the higher-level layers in the architecture because it literally represents the physical phenomena being monitored from the higher layers point of view. Radio transmitters, photodiodes, temperature and humidity sensors are all components that could be included in this layer.

At this layer, a degree of intelligence, adaptability or dynamic decision making process can be vastly beneficial and effectual to the operation of a sensor node/network. An example of adaptivity within this layer may be in a situation where a sensor has failed, or its readings are noticeably unreliable, then the sensing device controller might decide to report the last reliable reading, or contact neighboring nodes and report the average reading of other nodes. The decision should be taken according to the information available about neighboring nodes, the environment local to the node, and the status of the node itself. However, it is important to notice that the decision-making process does not require any information to succeed. It works in a best-effort fashion, which provides robustness and fault tolerance.

Raw Sensor Readings Layer: This is where the data from events from the previous layer are handled and stored. It filters the sensor readings to identify the interesting data items and pass them on to other, mostly higher-level, layers in the node. Interesting data items are defined by the domain knowledge of the application. Higher layers can parametrise this layer to identify which data should be considered

interesting, hence determining what constitutes an event. It is almost like one layer subscribing to a type of events that may occur in the layer below. An example of a decision making process within this layer could be the following: if a pollution-related process finds that its readings are the same for the last 10 samples, it might choose not to request any further readings from lower layer modules (pollution sensing device controllers) for some time, and instead use the value of the last reading.

Contextual Information Layer: In this layer, interesting contextual data is aggregated from different sensors and communication channels. Higher-level knowledge is then produced by analysing these data items. If events interesting to higher layers happen in this layer, notification messages would be sent to those layers. This layer could be seen as a form of middle-ware [20] that provides general distributed applications infrastructure services. Infrastructure services may include location services, routing services, security services, and so on. Decisions to run tasks based on the knowledge produced within this layer likely will occur in higher layers of the refinement architecture, just as decisions to perform tasks within this layer are likely made based on events from lower layers. This layer can be seen as mainly performing a combination of control and intelligence functionalities on a system level rather than on a node level. For example, this layer may coordinate with other nodes to preserve network energy, while the *Sensor and I/O Layer* would work on minimising energy expenditure on a node level solely.

Incorporating adaptive algorithms within this layer can be very critical to a sensor network's performance. For example, decisions related to GPS-based location services within this layer would greatly affect the longevity of a network, as communication is an energy-hungry process. This layer could try to minimise communication to provide location services by, for instance, gathering information from neighboring nodes, i.e. using local communication, rather than using the nodes long-haul communication equipment.

User Application Data Layer: This is where user programs and applications are run based on events occurring at the lower layers of the architecture. High-level data analysis could be performed within this layer. Examples of tasks that can be performed here include further processing of the findings of lower layers to produce general policies or report application-level findings to human observers. Other activities that can happen within this layer include operating/controlling actuators, planning node policies, and/or mapping received data into user-friendly representations. As with the lower layer, adaptivity at this layer can provide node- and system-wide advantages.

Notice that within each layer in this architecture we could have several modules [190]. Communication between pairs of layers is an abstraction of the communication between modules in different layers. In other words, each layer could have more than one module, which can communicate with each other. Also modules in different adjacent layers can communicate with each other. Layers cannot communicate with non-adjacent layers. This module-to-module connections view of the system adds a service-oriented aspect to the architecture, however, the architecture cannot be viewed as a purely service-oriented one. It helps to clarify our architecture if we compare it to the SOA, especially that there are many similarities between the two. Differences between our hybrid event-based architecture and the SOA can be summarised in the following points:

- Service oriented architectures *SOA* have autonomous modules throughout. Autonomy in our architecture is layer-wise only.
- The compatibility between different components in SOA is through policies and this could be true in our architecture. However, component compatibility in our architecture is restricted by the layer the modules belong to.
- In SOA, the boundaries of each component is explicit. However, in a layer, this is not a requirement in our architecture. Components are only functional concepts, rather than real software components.

Of course, some applications would adopt the SOA approach within each layer, and this is fine.

- Finally, schemas and contracts exist between components in a SOA, while they exist between layers in our architecture.

3.6 Functional/Architectural Model of a Sensor Node

In this section, we present a more detailed functional/architectural view of a sensor node [85] [87] [157] [47] [56] [84] [180] [138]. Also we identify on the architecture a more specific points of deployment for the adaptive algorithm we discuss later in this dissertation. The reader could think of the previous abstract model in figure 3.2 as a general high-level view of a node or a sensor network stack, while figure 3.3 is a modular more specific view of a sensor node.

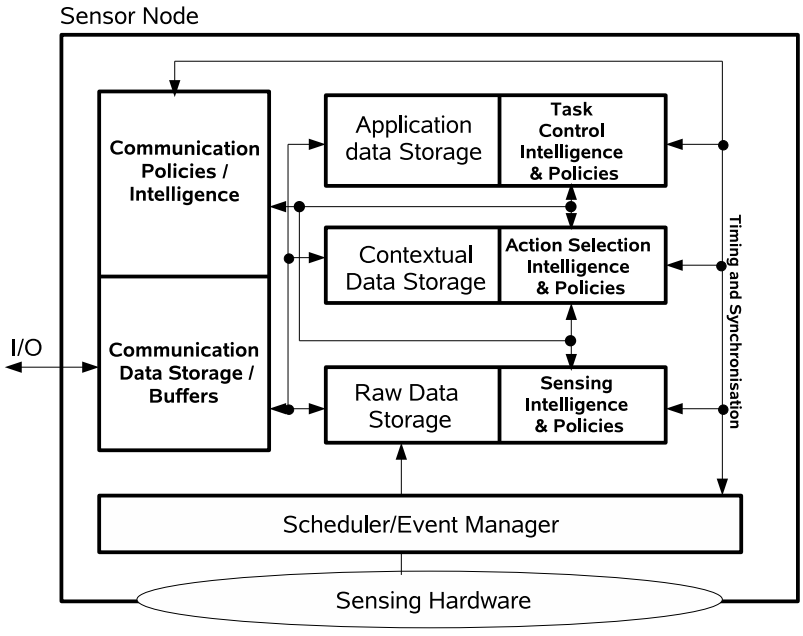

Figure 3.3: Functional/Architectural view of a sensor node

In this section, we will not discuss the hardware layer because it does not involve much data handling but rather state-related information that

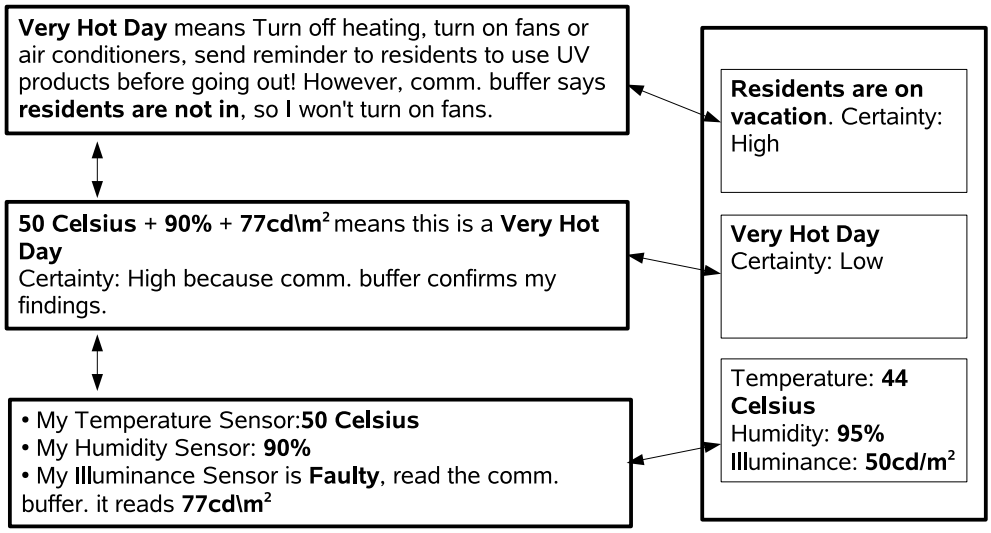

Figure 3.4: Example application for the architecture of figure 3.2

drives the operation of the hardware sensors. However, we think that the same concepts we adopt in this dissertation would apply on the hardware layer taking into consideration its specific requirements and characteristics. In addition, it could be argued that the intelligence at the Raw Sensor Data layer can incorporate rules and behaviour that controls the hardware layer, so they can be merged in some applications.

Here is a description and discussion of each module in figure 3.3:

Intelligence and Policies Modules These are: 1) Task Control Intelligence and Policies Module 2) Action Selection Intelligence and Policies Module 3) Sampling Intelligence and Policies Module 4) Communication Intelligence and Policies Module. These modules are needed in every layer of the abstraction presented in the previous section, and they represent the points where we propose that adaptive algorithms can be advantageously employed. They can be seen as the controller modules of their respective layers, but that does not necessarily mean that there could be only one actual module performing this role per layer, but they are rather the decision making mechanisms within the layer regardless of the type or number of modules involved.

Implementation-wise, the choice is domain- and application-specific.

At the Raw Sensor Readings layer, simple analysis to control the operation of the hardware sensors may be employed. For example low level power concerns may be addressed at this level by some form of intelligence, i.e. an adaptive algorithm. In the contextual information layer, the interesting raw readings from different data sources are analysed and processed into a more useful information for the application layer modules. Therefore some form of intelligence and/or user policy rules can be useful within this module too. Policies here can be some form of user or application explicit requirements. The need for intelligence at the application layer is obvious and might be included in multiple software applications within this layer. However, we emphasise that the module shown in figure 3.3 at the application layer can also be concerned with the meta data of applications rather than the details of the application purposes themselves. For example, this intelligence module might allocate more memory to one application than another because of prioritisation issues. The application-layer tasks themselves might not know about each other. Communication Intelligence Modules could make adaptive decisions on routing, communication medium frequency, etc. In figure 3.4, adaptivity in its simplest form here can be seen as a set of fuzzy-logic rules that control the behaviour of the node at each of these layers, i.e. the 'words' that drive the logic represented within each of the boxes.

Data Storage Modules Each layer of this functional architecture has its own type of data to process, hence the need for this module. These modules could be a type of DBMS system, an API for XML data, or a special storage on a mobile phone. They have the intelligence and capability to handle the low level details of data processing such as indexing, integrity, encryption, etc. These storage modules might not be physically separate, i.e. a shared physical memory or the same object structures may be used to store data from different layers or modules. However, conceptually they are treated differently according to the level of intelligence required or the granularity or purpose

of the processing of the data items. The I/O Buffer modules represent the storage module for the communication dimension of each layer. In figure 3.4, the data that appear in bold font represent the information that are saved in the storage modules.

Scheduler/Event-based manager module Intelligence modules may make conflicting decisions or need a form of inter-communication and planning. This module ensures smooth coordination between various layers, modules, and components within a node. In addition, timing and synchronisation issues need to be addressed globally within a node to increase efficiency and facilitate the operation towards a particular set of goals. In its simplest form, this module can be a timetable or scheduler for different modules operation in a TDMA slotting mechanism. So it may be sending wake-up signals, re-schedule signals, error messages, etc. The logic in this module should be minimal as modular distributed design of intelligence is fundamental to our architecture.

Before we end the chapter, we would like to clarify how the communication intelligence and data management relate to the rest of the node architecture because it is a layer/module combination that spans all other layers in figure 3.2. Inter-node communication is handled by an intelligent agent in the communication module. The communication process is assumed to be transparent to any of the intelligence and policy modules within any of the layers. For example, if the Raw Sensor Readings intelligence module suspects that one of its readings is faulty or unreliable, it may decide to communicate with a neighbour, and request another opinion, so to speak. This decision is not supposed to be explicit in our design. The suspecting module would request such an action from the communication module, and the communication module would put a value in the buffers or data repository for the suspecting module to pick up. Where this latter value comes from is not the business of the suspecting module. It only can read the target data item from the communication data storage without making assumptions about where the communication module has obtained it from exactly. Of course if the communication module decides to add indications of how it obtained the data item, it could do so, however, this is not guaranteed or

required. The suspecting module may not know for sure if communication has actually happened or has not. This abstracts the communication process into another simple data storage module, hiding the complexity of communication issues, decisions, and problems from intelligence modules at the various layers of the architecture. Of course, the intelligence modules can decide that the data coming from the Communication data storage are unreliable, or decide not to write data to communication data storage since, for instance, its processes are not functioning properly. In essence, the modules in our architecture behave as if they were sensor nodes in a sensor network, especially in terms of autonomous decision making and robustness. Communication data storage can be seen in figure 3.4 as the separate boxes on the right. Note that each box has information that are on the same granular level as that of the layer it serves or corresponds to, however, they can be stored physically on the same medium.

Most of the coming chapters will explore deploying adaptive algorithms at the communication module, the application layer, contextual information layer, and the raw sensor data layer.

3.7 Summary

In this chapter we presented the network requirements identifying the type of networks targeted in this dissertation, and clearly defined in section 1.2. Later, the major functional points within a sensor network building blocks, i.e. sensor nodes, were introduced and discussed. Then, we introduced an abstract high-level architectures of both a sensor network, and a sensor node. Finally, a functional more detailed view of a sensor node architecture was demonstrated and correlated with the high-level views of both the network and the node. The adaptivity points were highlighted within each layer briefly, as they will be further discussed in more detail in later chapters. An example application, together with our application scenario introduced in chapter 1, were used throughout the chapter to illustrate the ideas and concepts in the context of a real-life application.

In the next chapter, we will be looking at applying adaptive models at the fundamental functional points identified in this chapter.

Chapter 4

Action Selection and Task Allocation

This chapter introduces an adaptive technique for task allocation and action selection. To easily explain the technique, the process occurring at the node level, i.e. the action selection process, is presented first. Subsequently, the macro effect of these processes on the whole network, manifested in the form of a fairly complex task allocation mechanism, is discussed. The term *task* is used to refer to a high-level function/process, such as a *Air Pollution Monitoring Application*, as opposed to a low level function/process of an operating system, such as procedures for memory management. The term *action* refers to the choice made by a node as to what to do at a certain point in time. For example, a component in a multi-purpose sensor node may face a choice between performing task T_A associated with Pollution Monitoring $P1$, or task T_B associated with Traffic Monitoring $P2$. Moving towards the execution of either of the tasks is called an action in this context. Even the decision to perform neither is a type of action.

In the next section, non-adaptive techniques are briefly discussed followed by sections discussing, in more detail, the Response Threshold Model, a biologically inspired model for task allocation. This model is the key motivator of the adaptive mechanisms proposed in this thesis.

4.1 Non-Adaptive Action Selection

One approach for task allocation in sensor networks is preset scheduling, where tasks are scheduled to be performed at times determined prior to deployment of the network. For example, in a multi-purpose sensor network, a node can be configured to periodically run applications, i.e. at preset intervals, with each application representing monitoring a certain phenomenon such as Weather, Pollution, and Traffic. Another scenario is to have nodes specialised in performing only one task, e.g. constantly running a software application to manipulate data associated with a air pollution levels. Scenarios with such static schemes attempt to maintain a constant stream of readings, consequently incurring a constant rate of power consumption, independent of the conditions of the nodes, environment, and/or the network. This allows the user to have a fixed-precision representation or depiction of the phenomena, regardless of how interesting the information provided by the network is.

Other forms of non-adaptive behaviour in action selection include *fixation* or *local optima* phenomena. Consider the following scenario: a node made a decision to perform a task based on circumstances present at the decision time. After some time has elapsed, the conditions to perform the task has ceased to exist, however, the node is still stuck with the obsolete decision it previously made to perform the now unnecessary work. In such a scenario it is said that the node is tied to a local optima.

Preset scheduling schemes and local optima are examples of potentially inefficient non-adaptive techniques where data obtained by the sensor nodes may not carry much information, and consequently may waste valuable energy (or more generally *resources*). For example, in the previous scenario of the multi-purpose sensor network, if, at time t, the network reports to a base station that the traffic intensity p_i value is c_i, and then at time $t + 1$ it reports that the traffic intensity p_i value is c_i. Then the new data did not necessarily bear much information, but rather repeated the previously conveyed message. If the sensor network stayed idle, this might indicate to the user that there is no change in the traffic intensity without any associated energy expenditure. If nodes were to follow a simple adaptive rule like

reporting only relative variability in traffic intensity that exceed a certain constant value, it will still make energy savings.

Employing a sensor network that can adapt its actions to match the environmental factors may result in greater efficiency and better utilisation of resources. In this chapter, we introduce models that can allow a sensor network to achieve exactly that.

4.2 Adaptive Action Selection

As sensor nodes detect surrounding phenomena, they process their readings, and accordingly, determine their next appropriate actions. The algorithms involved in processing sensor readings are generally application-dependent. This dissertation focuses on the processing that task allocation involves, giving little attention to the processing required to perform the tasks themselves. Often, sensor networks are deployed to perform a number of tasks. Given the resource-constrained nature of the nodes, multi-tasking can be difficult and therefore sensor nodes often adopt a single-tasked model, where only one task is active at a time. As previously discussed, a task is a high-level application, such as *Monitoring of Pollution Levels* or *Monitoring of Traffic*. If a node detects the need to perform multiple tasks, it has to conduct an arbitration process, whereby only one of the required tasks is performed and the rest is deferred. This arbitration process may depend on several criteria that are application- and task-dependent.

A general model for adaptive task allocation was proposed in [25] based on observations from insect societies. We take this model as an example of adaptive algorithms that can optimise the use of network resources while maintaining the network performance at satisfactory levels. Satisfactory levels of performance are decided by human users and given as input to sensor nodes. In the next few sections, we will describe the model in general and two of its variations in addition to presenting a deployment scenario in a sensor network. In further sections, the models will be extended to support the inclusion of more criteria in its workings.

4.3 The Stimulus-Based Response Threshold Model

Within a group of insects, an individual performs a task if it observes sufficient cues indicating demand for the task to be performed. These cues might be environmental or in the form of messages from the fellow members of the society. Such cues can be categorised according to the task they stimulate the individual to perform, hence the name *task-associated stimulus* or hereafter *stimulus*.

In insect societies, many factors contribute to an individual's decision to perform a task, including genotypic, environmental, temporal, morphological, physiological, and social factors. However, certain cues can be dominant in stimulating the performance of a particular task, at least from an observational viewpoint. For example, bumping into many dead bodies in a colony can be seen as a strong stimulation for individuals to perform the task of '*Dead Body Clearance*', regardless of other potentially less important cues in this context such as weather conditions. The amount of stimulation experienced or the number of cues observed that are sufficient for an individual to start performing a task is called the *task-associated response threshold* or *task response threshold* hereafter.

Two variations of a mathematical representation of the task-stimulus correlation described above, i.e. a response threshold mathematical model, were developed in [25] (a *Fixed* Response Threshold (FRT) model) and in [181] (a *Variable* Response Threshold (VRT) model). These models were chosen to be investigated as adaptive mechanisms in this dissertation for the following reasons.

- They map algorithms that seemed to succeed spectacularly in nature in systems that bear great resemblance, as previously discussed, to sensor networks.
- They are simple enough to be employed in a sensor node, yet flexible enough to account for a large number of parameters.

Nodes in a sensor network will play the role of an individual insect in the model, while a sensor network will be analogous to an insect society. Nodes

will use cues from the environment and peer nodes, the network goals, and the node's internal status to make decisions about what actions to take. This draws from almost identical situation, according to the model, in insect societies where individual insects use peer interactions, environmental cues, insect capabilities, and society/group goals to make decisions about future actions.

4.4 Fixed Stimulus-based Response Threshold Model

In this model, each node holds N variables denoted $S_0, S_1, \ldots, S_n$ equal to the number of potential tasks as well as N corresponding constants denoted $\theta_0, \theta_1, \ldots, \theta_n$. Each pair of corresponding variable and constant (S_i) is associated with a task T_i. S_i represents the stimulation a node can detect in connection with the associated task T_i. The θ_i constant represent the critical value of S_i at which the node should start performing the associated task T_i. The latter is also known as the *task-associated threshold* in the model's nomenclature [25]. This will be referred to as *threshold* hereafter in this document.

The model proposed in [25] is probabilistic. This allows the exploration of different adaptive behaviours, provide more flexibility, and account for the noisy environments where nodes (or social insects in [25]) generally exist. In addition, probabilistic models allow the abstraction of unknown factors that contribute to the behaviour of a node/insect. Therefore, given a stimulus S_i and a threshold θ_i, a node will perform an associated task T_i with a probability $\Psi_i(S_i)$.

Theraulaz [181] used a particular equation to relate the stimulus S_i, the threshold θ_i, and the probability an insect (or node in our work) will start performing the associated task with probability, $\Psi_i(S_i, \theta_i)$. However, the authors did not exclude the validity of using other equations with similar features. The choice is likely to depend on some application-specific requirements such as curve growth rate, response sensitivity, etc. Theraulaz [181] used the following equation:

$$\Psi_i(S_i) = \frac{S_i^n}{S_i^n + \theta_i^n} \tag{4.1}$$

Another example of an equation with similar properties is:

$$\Psi_i(S_i) = 1 - e^{(-S_i/\theta_i)} \tag{4.2}$$

Notice that $\Psi_i(S_i)$, in both equations, satisfy the following two features:

1. $\Psi_i(S_i)$ approaches 0 as S_i approaches 0, and
2. $\Psi_i(S_i)$ approaches 1 as S_i approaches ∞.

The value of S_i increases if more cues for the associated task are detected, and decreases otherwise. This means the more cues are detected for a task, the higher the probability the insect (or node in a sensor network) will engage in performing that task. Thus, a task with a high threshold needs a high number of detected cues before an insect (or node in our case) starts performing it. As aforementioned, any function with similar characteristics can be used in the model. This function is analogous to the utility or fitness function in other algorithms like auction-based ones.

The values of θ_i is normalised between tasks, and can be seen to represent the task priority associated with this task. For example, in our multi-purpose sensor network, traffic monitoring may be more urgent than weather monitoring, so the network need to be more sensitive to the traffic measurement in the environment. By setting the thresholds associated with traffic to a low value, nodes will respond faster to their associated task.

4.5 Variable Stimulus-Based Response Threshold Model

The Variable Response Threshold Model is almost the same as the Fixed Response Threshold Model except that the value of the thresholds θ_i vary over time according to detected environmental conditions and insect status. This variability allows specialisation of a node to emerge if it is more frequently stimulated to perform a certain task, than to perform others. So

the probability is a function of both θ and S as follows:

$$\Psi_i(S_i, \theta_i) = \frac{S_i^n}{S_i^n + \theta_i^n} \tag{4.3}$$

The variable response threshold employs two additional parameters, a forgetting, and a learning coefficient. When a node engages in performing a task T_i, the task-associated threshold is decremented by an amount called the *learning coefficient*, β, and all other task thresholds are incremented by an amount called the *forgetting coefficient*, α. This makes easier for this insect/node to perform this task if stimulated to later, and more unlikely to perform one of the other tasks. A mathematical representation of this procedure is given in algorithm 1.

```
1  if Just engaged in T_i then
2      θ_i ← θ_i − β ;
3      if θ_i < θ_min then
4          θ_i ← θ_min
5      endif
6      for j = 0 → N do
7          if i ≠ j then
8              θ_j ← θ_j + α ;
9              if θ_j > θ_max then
10                 θ_j ← θ_max
11             endif
12         endif
13     endfor
14 endif
   // where N is the number of tasks
   // where β is the learning coefficient
   // where α is the forgetting coefficient
   // where T is a task
   // where θ is the threshold associated with a task T_i
```

Algorithm 1: Variable Response Threshold Algorithm

Given similar stimulation levels for all tasks, nodes have a greater ten-

dency to perform tasks with low threshold values, which makes them quickly specialise in performing these tasks. This also results in less task switching than if the *fixed* response threshold model is used, which is beneficial in applications where task switching involves costly resource overheads. An example of such a situation is if switching tasks require learning new communication routes, which incurs additional energy and time overheads. Also, switching from idle to active is expensive in general and so should not be done too frequently in a sensor node [5]. Notice that nodes may perform tasks with high stimulation levels even if they are not specialised to perform these tasks.

4.6 Resource-based Response Model

Both variable and fixed response threshold models perform the modulation of a node's behaviour according to environmental cues. They do not take into consideration the state of a node's resources, or that of neighboring nodes. We here extend the model to account for internal resources (e.g. battery levels, memory space, bandwidth, etc) when performing action selection processes. In our extension of the model, we will consider the battery levels as an example of a resource to incorporate into the action selection process or criteria. However, other resources could be incorporated in a similar manner without loss of generality.

Battery levels can indicate the life expectancy of a node under a known energy consumption rate, and is a very critical resource in many sensor networks. If a node has low battery, it may need to sparingly consume energy in order for its lifetime to be extended. If many nodes in one network fail due to flat batteries, reduced network coverage and increased network disconnections may result. In such situations, nodes with high battery levels may be more eligible to perform tasks, possibly relieving nodes with low energy levels.

Energy is consumed within a node mainly by three types of components. These are: 1) Sensing Devices, 2) Communication Devices, and 3) Computing Devices (CPUs, microprocessors, etc). Tables 4.1, 4.2, and 4.3 show a comparison of the power requirements of various devices. These tables are

Phenomenon	Current	Manufacturer
Photo	1.9 mA	Taos
Temperature	1.0 mA	Dallas Semiconductor
Humidity	550 uA	Sensirion
Pressure	1.0 mA	Intersema
Magnetic Fields	4.0 mA	Honeywell
Acceleration	2.0 mA	Analog Devices
Acoustic	0.5 mA	Panasonic
Smoke	5.0 uA	Motorola
Passive IR (Motion)	0.0 mA	Melixis
Photosynthetic Light	0.0 mA	Li-Cor
Soil Moisture	2.0 mA	Ech2o

Table 4.1: Energy Requirements of Various Sensing Devices

for comparison purposes only, and we are not using any particular device from these tables in our experiments since we conducted them on a more abstract level. However, generally, we assumed we used a chip like the one in [57], or the MSP, PIC, AVR families [124]. MEMS-based sensors are low power and we would recommend them for sensor networks. Finally, we recommend communication devices like the ones in [39] and [137]. Sensing devices power consumption varies significantly according to the phenomena and the device types. However, in general, they require significantly

CPU	MIPS/mA	Idle (mA)
Atmel AVR AT90LS8535	1.25 min	< 0.001
Microchip PIC16F877 (preliminary)	1.66 preliminary	< 0.001
MC68H(R)C 908JL3	0.1 typical	0.001 typical
Atmel AT91M40400 16/32 bit Strong Thumb	0.6 (1.35 static current)	< 0.001

Table 4.2: Energy Requirements of Various CPUs

Device	Current Consumption (mA)	Throughput kb/s
HX2000 Transmitter	4.5	10
RX2010 Receiver	2.5	10
Blue Tooth Compliant Philstar PH2401	20	1000

Table 4.3: Energy Requirements of Various RF Communication Devices

less power than RF communication devices, especially if we know that RF communication devices consume power even when there is no communication taking place, i.e. listening mode, while in many sensing devices, power is consumed only when a sampling event occurs. Table 4.2 shows the power consumption of some CPUs when they are active, however, when they are inactive, the power consumption is lower by orders of magnitude. Generally, power requirements of computing is much lower than those of communication in sensor networks.

Mathematically, we correlate the probability, $\Psi(B)$, by which a node will perform a task T when its current battery level or energy level is B by the following equation:

$$\Psi(B) = (\frac{B}{B_0})^m \tag{4.4}$$

Where B_0 is the node's full battery content of energy
Where m determines the response curve steepness (i.e. speed of growth) in relation to resource variation in general, or battery level variation in this case.

This equation was chosen arbitrarily from equations that fulfil the following two characteristics:

1. The probability $\Psi(B)$ approaches 0 as B approaches 0 (i.e. battery is nearly flat), and,

2. The probability $\Psi(B)$ approaches 1 as B approaches B_0 (near full-battery).

Any equation with similar characteristics could have been used. More sophisticated forms of this equation are possible in light of more realistic battery consumption models, however, this is out of the scope of our research. It is worth mentioning that there are battery models that have a linear range which we assume we work within here [147] [99], so this assumption is sufficiently realistic.

The same argument above could be used to produce an equation for memory as a resource. A node with a total physical memory of Mem_0, current free memory of Mem would respond to a task performance request with a probability calculated by the following equation:

$$\Psi(Mem) = (\frac{Mem}{Mem_0})^m \tag{4.5}$$

More complex memory equations could be used, but this one only serves as an illustration of the possibilities.

4.7 Combining Battery-Based and Stimulus-Based Response Models

The previous two sections addressed models to deal with each of the three key factors we identified in making action selection decisions, namely resource cost, task demand, and high-level user policies. The node needs to account for all three factors simultaneously when making action decisions. We propose the following equation to be used by a node to account for the first two of them, while a set of fuzzy rules to address high level user requirements:

$$\Psi(S, B) = (\frac{S^n}{S^n + \theta^n})(\frac{B}{B_0})^m \tag{4.6}$$

Where $\Psi(S, B)$ is the probability a node engages in performing a task T
Where θ is the task-associated threshold
S is the stimulus detected for task T
B is the current level of the node's resource
B_0 is the maximum amount of the node's resource

n and m are the sensitivity parameters to changes in task stimulus and battery level respectively

Notice that user policies are also represented by the θ variable, which signifies two issues. First, the sensitivity of the node to stimulus with respect to a task, and the task priority relative to other tasks. Environmental cues were represented by the S variable, and the resource status is represented by the B variable.

The equation was chosen to satisfy the following condition:

> $\Psi(S, B)$, the probability of engaging in performing a task T based on both available resource and current stimulus levels, denoted B and S respectively, approaches 0 as either B or S approach 0, and approaches 1 as the stimulus and battery levels approaches ∞.

We do not claim that the equation above is optimal for all applications and many equations that satisfy these conditions could be used. Fitness and utility functions are application- and domain-specific.

The next chapter presents simulations conducted to test the effects of employing these algorithms in a hypothetical sensor network, and observe the macro- and micro- phenomena that result.

4.8 Network Performance Metrics

It is generally difficult to produce a clear single definition of a measure of quality of the performance of a sensor network. One possible reason for this is the application-dependent nature of the problem. However, this is not the main reason the identification of a definite metric of network performance quality is complex. The main problem lies in the general contradictory requirements a sensor network needs to satisfy simultaneously [75].

Sensor nodes are very resource-limited. This means they need to do as little as possible, and use resources as wise as possible to conserve their resources. However, intrinsically, consuming more resources can lead to better performance. The forces driving a node to satisfy the previous two assumptions result in a compromise. The extreme case of consuming no

resources will most likely result in an unacceptable task performance levels. The other extreme, i.e. performing tasks without limit on resource consumption, is undesirable because the node lifetime or other capabilities will be over-strained and node failure/death will result very soon.

However, a combination of the following metrics [87] may allow us to evaluate a sensor networks performance:

Lifetime: most monitoring application depend on long operational lifetime of a network. Network density and node redundancy play a big part on defining the lifetime of a network in terms of the number of live nodes.

Coverage: ideally 100% coverage is required. This is difficult to attain in most networks. However, luckily, many applications tolerate much lower instantaneous coverage.

Response Time: some security, and other critical monitoring systems, require fast response from the network. This can range from as small as milliseconds to as long as hours.

Temporal Accuracy: time stamped data processing and correlation is paramount in many applications. For example, in our application scenario, if two nodes detected the same traffic density at the same time or around the same time, then they can fairly accurately decide that they have detected the same jam incident. Lack of timestamping here may cause confusion.

Cost and ease of deployment: self-management, self-configuration, and self-healing are all desirable features in sensor networks to reduce cost and make network deployment a viable process.

For the purpose of our experiments, we consider two measures to be relevant in assessing the quality of a sensor network operation. The first is the network lifetime. In monitoring applications, networks that last longer are considered better than networks that die out quickly. The percentage or count of dead nodes that render the network unusable is application-

and domain-specific. The second metric is the average and total task performance. Task performance in monitoring applications is usually network coverage [131]. Each node is said to give a view of the target phenomenon within a certain neighbourhood or vicinity. Often this neighbourhood is identified by the area falling within a radius r full circle from the node's centre. This document assumes a binary model of sensing [111]. At any point on the sensing surface, there is either coverage, or no coverage. There can be no 50% or 33% coverage. It is either on or off. This sometimes is called the *Boolean Sensing Model* [90]. This model is realistic enough and used in many sensor network applications and simulations. An example can be our application scenario, where for the weather monitoring task, wind speed can be measured by only one sensor node within a square kilometer, and is assumed to be uniform over this whole area. Note that the work in [90] addresses the network coverage issues, but does not deal with adaptivity issues and the regulation of multi-tasking.

Mathematically speaking, a node will provide a coverage of πr_0^2 if it is said to represent a phenomenon within a radius r_0. We will call this radius hereafter *sensing radius*. We will call the πr_0^2 the *coverage range*. In a sparse network, where nodes do not have overlapping ranges, a network composed of N nodes, would yield a total coverage, $C(N)$, according to the following equation:

$$C(N) = \sum_{i=1}^{N} C_i = \pi \times \sum_{i=1}^{N} r_i^2 \tag{4.7}$$

where r_i is the sensing radius of the *ith* node. If the nodes have equal sensing radii, then equation 4.7 can be simplified to:

$$C(N) = N \times C = N \times \pi \times \ r^2 \tag{4.8}$$

where C is an individual node's coverage range and r is a node's sensing radius. However, the coverage analysis gets more involved when overlapping ranges occur in the network.

To further investigate the coverage problem when overlapping occurs, we need to introduce a few concepts. These will be useful in the next chapter when we conduct our experiments and try to assess and analyse

the results. These critical definitions are:

Network Density, λ: It is the number of nodes present within a surface area unit. Assuming a square-shaped 2D-terrain with L side length will have a surface area A and N nodes randomly distributed over the surface, then the network density, denoted λ, will be defined as:

$$\lambda = \frac{N}{A} = \frac{N}{L^2} \tag{4.9}$$

Note that some of the equations that will follow depend on an assumption that the terrain surface area is much greater than the sensing radius of any node, i.e. $L \gg r$. This will make sure that the *edge effect* is marginal. The *edge effect* is the weight of the lost coverage by the nodes that are in close proximity to the edge of the sensing terrain. This lost coverage is marginal when $L \gg r$.

Area Coverage, f_a: It is the ratio of covered area to the total area A. Covered areas are those within r distance units from at least 1 sensor node. The value of f_a must be between 0 and 1 ($0 \leq f_a \leq 1$).

$$f_a = \frac{C(N)}{A} = \frac{C(N)}{L^2} = \frac{C(N)\lambda}{N} \tag{4.10}$$

Node Coverage, f_d: The ratio between the number of covered nodes to those which are not covered. Covered nodes are those nodes whose coverage area is totally covered by other nodes. These node's coverage area is at least doubly covered (by the node itself for one, and then by other nodes for second). This value is a measure of redundancy in a network. This document is focused on networks with high f_d. In [120] a correlation is found between this value, f_d, and the area coverage, f_a.

Mathematically, area coverage can be calculated by the following equation:

$$f_a = 1 - e^{-\lambda \pi r^2} \tag{4.11}$$

and this equation will apply regardless of the overlapping of some sensors as long as $L \gg r$ is true.

To achieve a certain coverage f_a, the density required can be calculated by the following equation:

$$\lambda = \frac{-\ln(1 - f_a)}{\pi r^2} \tag{4.12}$$

4.9 Summary

In this chapter, we briefly described non-adaptive schemes of task allocation, and compared them to simple adaptive ones. In order to demonstrate the benefits of adaptivity, we gave few simple example scenarios.

The threshold-based model of task regulation in insect societies by Theraulauz and Bonabeau [24] was introduced. This model was extended to include resource factors in making the action selection decisions. Two variations of the threshold-based models were illustrated, namely the FRT and the VRT model variations.

Finally, sensor network performance metrics were introduced. The ones that were relevant to our dissertation were further discussed and their relevance reasoned about. Some mathematical equations and definitions were identified and explained in the same context.

Next chapter will present and discuss the simulations associated with the Action Selection Model that we undertook, in addition to an analysis and conclusions from the results.

Chapter 5

Task Engagement and Action Selection Simulations

5.1 Introduction

In this chapter, we will present our experimental platform, a simulation environment called NetLogo. Experiments for action selection using 5 models will be demonstrated. General parameters of the experiment simulations will be defined and discussed. Also model-specific parameters will be identified, and discussed.

The experiments in this chapter will focus on the action selection mechanisms within a sensor node. We ran experiments with the following five models, and then compared the results:

1. *Constant Response Probability Model*, or CR for short
2. *Fixed Stimulus-based Response-Threshold Probability Model*, or FRT for short
3. *Variable Stimulus-based Response-Threshold Probability Model*, or VRT for short
4. *Fixed Stimulus- and Resource-based Response-Threshold Probability Model*, or FRT+B for short

5. *Variable Stimulus- and Resource-based Response-Threshold Probability Model*, or VRT+B for short

5.2 Simulation Environment

NetLogo [196] [162] is our choice for simulating our bio-inspired models in sensor networks. NetLogo is a general modeling environment that particularly excels in the modeling of complex systems. It has been used for sensor network modelling previously [149] [80] [123]. It allows hundreds and thousands of independent agents to be modeled and controlled simultaneously. Agents run concurrently, which is a very important premise of sensor networks. Unlimited number of variables and actions can be associated with each agent.

In addition, NetLogo allows for the modeling of the environment through *batches*. Batches represent the terrain on which agents act, move, and/or reside. They could model the environment, atmosphere, or surroundings of the nodes. We used them to model the geographical area where our sensor network is deployed. Like with agents, any number of variables or procedures can be associated with each batch. A grid of batches, with adjustable sizes, appear as the surface where the simulation happens. This area is internally represented as a torus in NetLogo, i.e. there are no edges since each edge is connected to the opposite one.

NetLogo also offers an *Observer* agent that can manipulate the collective system, i.e. all batches and agents. The observer is usually used to control global variables such as time boundaries, simulation speed, statistical data collection and analysis, etc.

Figure 5.2 shows a number of agents modeled on a set of batches, while figure 5.1 shows an example user interface to control the parameters of a model in NetLogo.

NetLogo provides facilities to monitor both micro- and macro-level effects, behaviours, and motions of a system, including graphs, counters, monitors, events, and data files. Although NetLogo is sufficiently sophisticated, it is simple to program and has a rapid learning curve. In addition, NetLogo allows connecting software agents to real sensors or other hard-

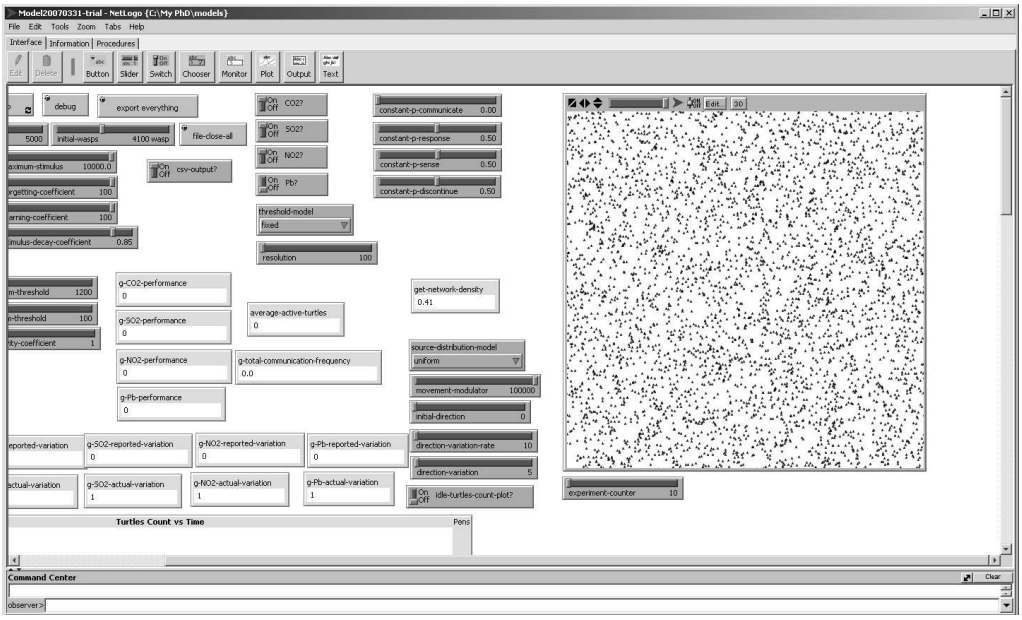

Figure 5.1: User Interface Demonstration of NetLogo

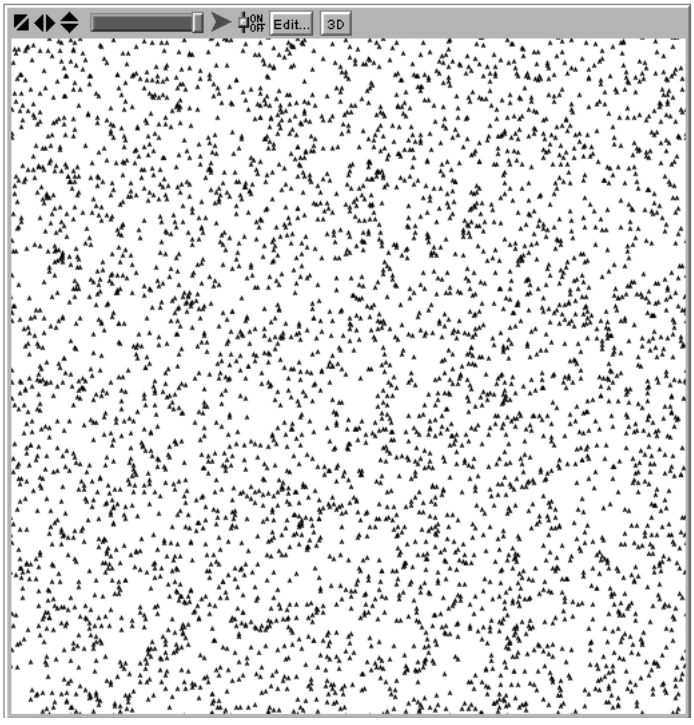

Figure 5.2: Agents on batches in NetLogo

ware. Finally, NetLogo is platform-independent because it is coded purely in Java.

NetLogo's models are hard to extend since the code is procedural, as opposed to object-oriented, and so hard to maintain and debug. As a consequence, very complicated models are not well suited to NetLogo. One of our goals was creating complex macro-effects through simple micro-rules, and so this disadvantage had limited impact on our simulations. NetLogo also slowed down considerably when the number of agents grew to a magnitude of tens of thousands, however, this was beyond our needs.

The main reason NetLogo was adopted as the simulation environment was the speed at which we could prototype and try models. This was extremely vital as the search space for our models was very large, and our Monte Carlo experiments had a lot of try-and-error cycles, dictated by the complexity of the macro-level behaviour and the number of parameters involved in the models of both the environment and the nodes. Implementing and experimenting our model with high-fidelity simulation frameworks and environments [184] [142] [78] would have required too long time which we could not afford. However, we have tried few selected experiments on a high-fidelity simulator[1] and the results were equivalent to those obtained from NotLogo. In addition, the level of abstraction our models apply at does not require such complex accurate simulations to be verified, therefore it would have been an overkill to use such high-fidelity simulators.

5.3 Network Evaluation Metrics

Before we present the results of the simulations, a glossary of terms used to represent and evaluate the results are provided. Some of these metrics or concepts were introduced in general terms in previous chapters, however, we put them here into the specific context of the experiments we conduct:

[1]We have used J-Sim [166][164][165] to run some of our simulations to confirm that the results from NetLogo match those obtained from a more complicated model of sensor networks/nodes, such as those in J-Sim. The differences were marginal, and irrelevant to our work.

- **Network lifetime:** the number of time steps elapsed from the start of the network deployment, to the time all network nodes have run out of energy. This is a very critical measure of a sensor network's efficiency. Often, sensor networks need to be able to function for periods of several years. A time step in our experiments is a generic time unit. Its value controls the speed of the experiment running. We assume here a time step is one second, and adjust the other parameters based on this assumption.

- **Network Task Performance, f_a:** a quantitative measure of how well a particular task is performed by the network. For example, if the network was intended to perform the task of monitoring the variations in air pollution, then the network performance of this task is the amount of pollution variance recorded or reported by the network, compared to the actual pollution variation. In monitoring applications, which are the main focus of this dissertation, the performance metric commonly used is network coverage. Network coverage has been discussed in detail in previous chapters and will be used as our metric for task performance in our experiments.

- **Instantaneous Network Task Performance, $f_a(t)$:** In our experiments, this is the network coverage at a particular moment in time. Any single point on a curve of the network coverage as a function of time represents an instantaneous network task performance, $f_a(t)$.

- **Total Task Performance, $\overbrace{f_a(t)}$:** this is the sum of the instantaneous task performances over the network's lifetime. For example, if an air pollution monitoring network lived for 3 time steps, and at the first time step it recorded 2 units of pollution variation, at the second it recorded 1, and finally 5 variation units were recorded at the third time step, the total task performance would be $2 + 1 + 5 = 8$ units of pollution variation. The average task performance would be 8/3 or ≈ 2.7 units of variation. Total task performance is, in continuous time mathematics, the integral of the total task performance function. In discrete time mathematics, it is represented by the sum of the

instantaneous task performances:

$$\overbrace{f_a(t)} = \sum_{t=0}^{t=n} f_a(t) \tag{5.1}$$

In continuous time mathematics, this will be:

$$\overbrace{f_a(t)} = \int_{t_0}^{t} f_a(t) \tag{5.2}$$

- **Average Network Task Performance, $\overline{f_a(t)}$:** the task performance of a sensor network may vary over time due to node failures, environmental conditions, or user/high-level policy changes. However, the instantaneous performances of a network with respect to a task can be averaged over the lifetime of the network to give a quantitative representation of the performance of the network with respect to a task. We call this quantitative representation the *Average Task Performance*. In discrete mathematical terms, this can be calculated as follows:

$$\overline{f_a(t)} = \frac{\sum_{t=0}^{t=n} f_a(t)}{n} \tag{5.3}$$

In continuous time mathematics, this will be:

$$\overline{f_a(t)} = \frac{\int_{t_0}^{t} f_a(t)}{t - t_0} \tag{5.4}$$

In general terms:

$$\overline{f_a(t)} = \frac{\overbrace{f_a(t)}}{Lifetime} \tag{5.5}$$

- **Instantaneous Task Demand Level:** represents a quantitative measure of the urgency to perform a task at a particular point in time. For example, assume that in a network, a task is to produce an air pollution analysis report aimed at mapping the pollution variance in the environment. In such a network, one pollution report will be enough as long as there is no pollution variance in the environment. This

may indicate that this task is of low priority. This can be represented internally in a node as a variable holding a normalised value of the pollution variance. Later, if a high pollution variation occurs in the neighbourhood of a node that is supposed to produce a report, then the task demand recorded by the sensors will increase, and this task will have a high task demand level, i.e. data for the pollution report should be recorded urgently. This will correspond, in our experiments, to the S variable in the FRT model discussed in more details in chapter 4.

5.4 The Experiment Scenario

We will demonstrate the experiment scenario in general terms, and then we will translate those general terms into a more concrete real-life sensor network scenario.

5.4.1 Abstract General Scenario

The experiments conducted followed a generalised scenario. They depict a network that is intended to perform three tasks. The number of tasks in the network was chosen arbitrarily to satisfy the following two conditions:

- The experiments were required to simulate a network with multiple tasks performed, so more than one task was needed. After all, the dissertation is about action selection which inherently means that there is more than one task to execute.
- However, too many tasks could have made the system highly intricate to be studied. The scalability of the algorithms we are investigating here is certainly an important issue, however, we leave investigating this for the future as it is outside the scope of this work. In this dissertation, the focus is on the capability of the thresholding models to control swarm systems with a large number of nodes, rather than tasks. Increasing the number of tasks per node will complicate the analysis as we will have to consider the performance of more tasks.

In addition, this will lengthen the time needed to perform the study beyond an acceptable limit.

We used three tasks merely to satisfy the previous two conditions. Each of the tasks, T_A, T_B, and T_C, is a monitoring task associated with the variation of a phenomenon. Monitoring a phenomenon involves running an application specific to it, and/or communicating readings pertinent to this phenomenon to a base station, or a satellite.

Each node in the simulated network can be in one of two states, active state, or idle state. An idle node is a node that is not doing any of the three tasks, i.e. not running any of the task-associated applications. Idle nodes have their CPU and communication equipment in the off or sleep mode. They consume minimal amount of energy in these states. An active node is a node that is performing one of the three aforementioned monitoring tasks. In other words, each node, if not idle, has a set of potential tasks to perform. Each task causes the node to consume resources at a predetermined fixed rate. The experiments in this dissertation will abstract resources in its calculations to account for only one resource, that is the power source, i.e. the battery. Other resource types could have been used, however, literature suggests that power consumption is the most critical resource in sensor networks [151] [109] [88], and so it is reasonable to focus our attention on it.

Monitoring tasks have different priorities, which can be determined by network goals, task criticality, and/or explicit user policies. Network goals are concerned with their general purpose. For example, a node could give self-maintenance a lower priority compared to monitoring tasks. Explicit user policies can be pushed into the network at real time, or built into the nodes at design time. They may include fuzzy rules such as which phenomenon to focus on, or be more sensitive towards. Task criticality refers to the urgency of performing a task due to reasons other than explicit user policies and general network goals. For example, in a network that harvests energy, if severe energy shortage is detected, then all tasks may be deferred until the Energy Harvesting Task has run sufficiently to allow other tasks to run regardless of any other factors.

In our experiments, task A has highest priority, followed by task B, and least priority was given to task C. Readings from a node represent the variations in a phenomenon within a radius r_0 from the node. The efficacy of a network in performing a task is evaluated in terms of two factors. The first is the accuracy of the network recording of a phenomenon, i.e. the difference between the actual variation in a phenomenon and the variation the network has recorded. The second factor is the resource consumption incurred to achieve such a level of monitoring accuracy. The network performance and network coverage will be used synonymously hereafter. Ideally, the goal of the network would be to achieve 100% coverage without consuming any resources. How close a network can be to this ideal goal represents the efficiency of this network.

5.4.2 Hypothetical Multi-purpose Sensor Network Application Scenario

Multi-purpose networks have grabbed the attention of researchers recently [93] [171] [172]. In a multi-purpose sensor network, a sensor node would have various types of sensing devices attached to it. The processor or CPU on the node would use these readings to serve various applications. We adopt such a network for our experimental scenario. Each node will have a set of sensors, and the node processor will be tasked with three different applications, each representing a *Task*. As aforementioned in chapter 1, our network application scenario will involve the following three tasks: Traffic Monitoring, Air Pollution Monitoring, and Environmental Monitoring. In the following few sub-section, more details are given about each of those tasks.

Traffic Monitoring, T_A

Traffic problems are quickly becoming an important and crucial aspect of the daily life of every citizen. Who of us did not suffer traffic jams? Road safety problems? Fuel prices? etc. These issues press for urgent solutions, and many new technologies are being harnessed to find out whether they could deliver solutions or mitigate the problems in relation to traffic. Sensor

Networks is no different in this respect [163] [179]. Such an application of sensor networks could help us better understand how traffic congestions and jams develop, subside, what are the underlying causes, and how could we avoid or resolve them.

In this dissertation, we take the traffic congestion problematic phenomenon and consider using sensor networks to mitigate its effects or detect it when it occurs. In this application, nodes take readings in connection to traffic jams, car concentrations, and bottlenecks. This data is then communicated to a server that feeds a radio station and broadcast to drivers to adjust accordingly. This is conventionally done via helicopter and other non-distributed systems.

We assume our sensor nodes are equipped with sensors to detect three traffic-jam related measurements [148]. These are *density*, *intensity*, and *speed*. Density is a measure of how many vehicles there are in a length unit, while intensity is a measure of how many vehicles there are per time unit. Finally, speed is a measure of how quick the traffic is flowing [83]. As one might expect, the higher the density, the more likelihood of a jam there is. The reverse applies to the intensity since the higher the intensity, the less likelihood there is a jam. Finally, the higher the speed, i.e. traffic flow, the less probability a jam will develop.

When a node's reading of any of the three measurements cross a certain threshold, the node will start sending a message indicating a possible jam [42]. Note that an equation, or a set of fuzzy logic rules may control the value of the stimulus associated with the Traffic Monitoring Task as a function of the intensity, density, and speed traffic measurements. How do we map those readings to a single scalar stimulus value is out of scope of this work. Once a significant change in the congestion status is detected by a node, this data will be transferred to a receiver on the closest vehicle, or vehicles, in the communication range the node, where the vehicle in turn will transfer the data to a control centre via a satellite link or any other communication links available. We will not tackle how this data is processed after this point, since this data could be for example received by peer nodes [112], base stations [209], or any other useful processing path. a possible final destination of the data could be in the form of a broadcast to drivers on

a radio channel directing them to act accordingly. We only focus on the actions of the nodes in this dissertation, which is detecting the jam, and notifying the sinks in the surrounding.

Note that this task will have the highest priority amongst tasks, because when a jam is developing, we do need to take fairly quick measure to stop it or avoid it exacerbating.

Air Pollution Monitoring, T_B

With global warming on the rise, and the increase of car emissions, authorities of big cities need to monitor the pollution levels [18] to see if its measures to curb emissions and curb carbon footprint is working. Like in [209], we assume a simple pollution monitoring application where there is a set of sensors on each node that would detect various air pollutants. We would assume our application will monitor three air pollutants, for simplicity. These are Carbon Monoxide (CO), Sulphur Dioxide (SO_2), and Nitrogen Dioxide (NO_2). The pollutant level readings will constitute the stimulus for the air pollution monitoring task [125] [77] according to some application-specific criteria that is out of the scope of this dissertation.

The same technique used in the traffic monitoring to deliver data/results to a base station will be used in the pollution monitoring task, similar to the scheme in [139]. That is results are transmitted to the closest vehicle and that then is forwarded by a satellite link, or a similar device, to a control centre that may make use of these results.

Note that this task has the second priority after the Traffic Monitoring Task, sense it has long term goals and usually does not require immediate action.

Urban Environmental Monitoring, T_C

Daily weather conditions together with various climate monitoring are vital for various scientific reasons that are connected to global warming and the contribution of the city where the network is functioning towards carbon footprint. Urban environment data is also useful for various other applications [112] [63]. Sensor networks can be used effectively study in a fine-grain

manner the developments and changes in the environment [16]. Note that even natural climate and weather change cycles can be monitored with the same network. The stream of readings can be used in many applications.

We assume this task requires the sensor node to be able to measure the following measurements: air temperature and humidity, surface temperature, incoming solar radiation, wind speed and direction, precipitation, soil moisture and pressure. When any combination of these measurements has an adequately large variation, the node should start performing this task, and send results to base stations on vehicles within its communication range. Again, this is the same technique used with both the Traffic and Air Pollution Monitoring tasks.

Note that this task has the least priority amongst the tasks, because it is usually a long term data collection task that only need to be analysed cumulatively.

In the next few sections, 10 runs were performed for each model and the results were averaged before analysis has taken place. In each experiment, the environment applies a constant demand for the three tasks T_A, T_B, and T_C. Nodes of the network are scattered over a bounded geographical area, and each node records the phenomena variations it detects in the vicinity. The collective action of the nodes gives an approximate view of the phenomena distribution over the whole geographical area the network covers. Nodes run out of energy, i.e. die out, and eventually the whole network follows. Experiments representing different models were given similar network parameters except the model used to perform the action selection decision making. This helps identify the effects of the action selection models without interference from the variation in other network parameters, such as energy consumption rates, number of nodes, or density of the network.

5.5 Simulation Assumptions

The simulation environment and the scenario we use makes a set of assumptions and abstractions about the real-life application described in the previous section. This section covers those assumptions, and explains the

rationale behind making them. These assumption hold for the rest of the simulations in this dissertation, especially since the experiments use the same scenario, and the same simulation environment. Only chapter 11 has the communication assumptions valid for coordination. Other chapters abstract communication details within the application layer, and consider them an application-dependent detail.

Note that the scenario itself is a motivation for discussing and investigating the algorithms we are experimenting with. The aim of the work was not answering a set of engineering questions, or inspecting specific sensor node hardware/software issues. The aim is basically to assess the fitness of threshold-based algorithms to solve problems that are generally present in sensor networks. The details of the implementation will depend generally on the technological advancement and choices made at the time of the solution production, manufacturing facilities, and design decisions. For example, if we make assumptions about energy cost of communication, then these are not necessarily true for the application we tackle here, because may be in the future new technologies will have new communication requirements. Also different communication schemes may have different power profiles, and so on. Moreover, running simulations in high-fidelity environments or simulators may waste massive time and energy [184] [136] [150] [161] in tackling issues that are mostly irrelevant to the focus of the research from an analytical point of view [104] [14]. We have experienced this first-hand when we tried running our experiments in a high-fidelity environment, J-Sim, and ended up having almost the same results but in much longer time and after investing much greater effort. It is generally known that the solution space is exponentially enlarged as the number of variables involved increases [27].

It is worth mentioning that many of the constant values we assume in our simulations are acceptable based on a large number of nodes and so an average over these nodes would be a good approximation for many of those values. This is for example true for transmission power, sensing power, transmission rate, packet sizes, etc.

Node Count, Type, and Distribution

The simulations have 5000 stationary nodes over the terrain of $100x100$ length units, and all of them possess the same sensing, processing, and communication capabilities, i.e. homogeneous dense network with high redundancy [139]. They do not have any voluntary mobility capability. We do not think that balanced heterogeneity will change the results we obtained here [198], but for reasons of time and space, we will not discuss heterogeneity experiments in this dissertation.

Nodes are distributed randomly but uniformly over an edgeless (torus) area [27]. We assume that this is what will happen in reality, but we need to be aware that this does not apply in all applications and situations of the real-world. For example, if sensor nodes were deployed on a highway through a desert, the readings will simply reflect the phenomenon on the road and the small area around it, as opposed to the desert in general. Our deployment is assumed to be random within a densely populated city.

Generally, the models we assume here represent *steady state* conditions of nodes, i.e. localisation, synchronisation, and other node/network initialisation activities are not included [198].

Environmental Model

We use a two-dimensional area to represent the environment, which is used in may sensor network simulation environments including high-fertility ones [169] and in many other research efforts [206]. The area is treated in the simulation environment as a torus, so the edge effect is cancelled. Another way to cancel the edge effect is to assume that the terrain is infinitely big by making the area of the terrain much greater than a single node coverage area.

Every phenomena has its own propagation, and data generation model [27]. However, we assume that our phenomena apply a uniform stimulus on the sensing devices, i.e. all sensors sense the stimulus with the same intensity at all times. This is not the case in most real-life applications, but the complexity resulting from using a more realistic sensing model could make interpretting the results more difficult, which we wanted to avoid at

this stage of the research. We wanted to establish and study the base case first, before, as a future work, delve into the behaviour of the thresholding models in real life applications.

Transmission Model

Data items are transmitted in an error-free [116], in a broadcast fashion between the sender sensor node and all its neighbouring nodes. The propagation model is a *free space model*, but without signal attenuation [150].

We use the *first order radio model* [116] as our radio transmission model. However, because nodes communicate fixed-size packets similar to the synchronisation packets in [30], we can abstract the energy model of the communication to consume a constant fixed amount of energy per communicated message.

Nodes transmit over a fixed-radius range [140] [139] [116]. Although this not very realistic [150], but it is commonly used assumption in analytical and abstract studies in order for example to establish upper bounds of performance possible to achieve using a certain protocol, technique, algorithm, etc.

Communication Model

Nodes in our scenarios have omnidirectional antennas, and have a common constant maximum transmission radius r [116]. This assumption is not particularly true in the real world, but we assume that the density of the network will be so high to offset such an defect. It remains to be seen how such a model fair in real sensor networks in future research activity.

We also assume that transmission/reception buffers are limitless [116], and work as FIFO queues. Memory nowadays is cheap, and so we can virtually have limitless queues. Certainly this is not strictly true in real life applications, although more likely true in sensor networks. In applications where queues grow huge in short time, and the communication buffers cannot cope with the amount of data, then obviously the results need to be revisited.

We assume that communicated data units have fixed size. While this

may not be strictly true in real-life applications, we envisage that the average size would be fixed, except in very rare occasions, and so the results should hold strong with this assumption. Note that also for performance reasons, networks might resort to fixed size packets which then makes this assumption very realistic. This is the case, for example, in *ATM* networks.

Communication wireless channel/medium is error-free or lossless [116], but the model does not depend on or assume so, and can easily accommodate such phenomenon in the future. This is especially because our model does not depend on communication generally. Moreover, since buffers are assumed to be of infinite capacity, data units are never lost while travelling through the network, and so there is no retransmissions.

Communication range is circular and propagation is deterministic. Communication signals are free to travel (i.e. no signal-holding obstructs), i.e. open space model. This is only applicable in communication experiments in chapter 11. Communication does not suffer any broadcast collision, nor other propagation effects. Although we do not use any acknowledgement or hand-shaking protocols, the network communication gives this effect because it is error-free.

Note: using simplified communication model with localised algorithms, and ours is, give a slightly different results, but with huge gains in terms of performance, speed of development, testing, and feedback [104].

A scheme like the one in [198] which is mainly ultra-low-power is envisioned in our work. It uses location-based addressing as opposed to IP-like routing and addressing protocols. It uses wake-up signals to perform communication, so there is no over-hearing, hyper-hearing, or idle-listening.

Processing Model

Nodes can receive, transmit, sense, and process data in one time slot because each of these processes is performed by a separate device hardware. Each application is seen as a black box which consumes energy at a constant rate, so we avoid the details of the application domain in favour of analysing the performance of threshold-based models generically.

Scheduling Model

When the sensor switches to the active mode, and the sensor schedules a time instant in the future at which it will go back to sleep. The scheduled active period, expressed in time slots, is a random variable calculated by the model. However, at the time slot at which the sensor should transition to sleep mode, the node may reassess its sleep probability to decide if it should sleep or not.

Note that while a node is performing a task, it cannot perform any other task, but it can sense, transmit or receive data.

Traffic Model

Our scenario uses an *event-based* traffic, as opposed to *time-based* or *query-based* one [140]. So, when nodes detect an event, specifically when a phenomenon probabilistically crosses a threshold, data is sent to a close by vehicle (base station) reporting the change, making recommendations, etc.

Sensing Model

Sensing radius is circular and deterministic [116]. Detection follows a Boolean, or binary, model. In many other research on sensing coverage, such as [111] [105] [34], the same model was assumed. In a binary detection model, sensor node can detect a target with a 100% probability provided that the target is within its sensing range, and cannot detect a target beyond the range. In this paper, we use a deterministic detection model. When a target is within a sensors sensing range, if the sensor node samples the environment, it will discover the phenomenon with full intensity, i.e. no propagation effects [206].

With a probabilistic detection model, we can hardly have a 100% detection probability for all the geographic points in the target area. The system will have different degrees of monitoring at different locations. However, a high node density can alleviate this problem. We assume high node redundancy and coverage which improves area coverage and alleviates the side effects of the ideal sensing model and make it virtually much closer to

a real sensing situation.

Battery Model

We assume a linear battery model [155]. In this model, the battery acts as a linear bucket of energy. The maximum capacity of the battery is provided independent of the discharge rate. Such a model enables us to investigate the efficiency of applications by offering a simple metric of energy consumption. In our fast, low delity simulations in steady-state conditions, a linear model is often preferred since the middle of the discharge curve is often almost linear [193] [192] [103]. In addition, the *relaxed battery model* acts exactly as a linear model in absence of high rate discharge. In our application, we assume that nodes will mostly act in low power mode most of the time and so the relaxation effects will have minimum impact on the simulation results.

Also although batteries are not linear, using a linear battery can represent a worst case scenario, because we do not utilise the relaxation positive effect. So the results are conservative as opposed to over-optimistic, which reflects the hardships/constraints imposed on sensor networks more or less better. Note that communication is what often results in non-linear battery consumption [143], but because most of our experiments do not include communication for coordination and control, it is not a problem to assume linear model.

Energy Consumption Model

Performing tasks consumes energy by a constant rate, i.e. constant amount of energy per time unit. Every *sensor reading* or sample consumes a fixed amount of energy, which is smaller by orders of magnitude compared to communication and processing energy profiles. *Idle* nodes consume a minute amount of energy, which is also a constant value per time unit [156]. *Transmission* of a packet consumes a fixed energy value per message (since packets are of fixed size), and the same amount is consumed for *Reception* of a message [140]. We use a symmetric energy cost for transmission and reception.

5.6 Constant Response Probability Model, Fixed Response-threshold Model, and Fixed Battery-modulated Response-threshold Model

This section will investigate the performance of the FRT and FRT+B models in a sensor network simulation scenario and compare them to that of the CR model and to each other's. Appendix B.1 lists the settings and parameter values for this set of experiments.

The FRT model addresses the action selection process based on the task demand and user policies, e.g. task prioritisation here. However, the FRT model does not take the resource consumption or availability into consideration. In chapter 4, we extended the model to include resources in its calculations. We will refer to the extended FRT model by FRT+B hereafter. Energy is typically a critical resource in sensor networks. Therefore, we use energy levels available in a node's battery as an example resource whose availability or abundance can contribute to action selection decisions within a node - and consequently within a sensor network.

5.6.1 Experiment Objectives

The experiment simulations in this section has the following objectives:

- Test the benefits of using the threshold-based model as an adaptive algorithm in sensor networks,
- Test the performance of the threshold model under various task demand levels,
- Test the threshold model capability to prioritise tasks under various task demand levels for all tasks
- Investigate the advantages and disadvantages of the FRT+B model as compared to the CR and the FRT models,
- Identify the various dynamics and side effects resultant from the adoption of the adaptive models in a sensor network.

- Identify the family of applications that can benefit from employing the FRT or FRT+B models, as opposed to the CR model.

5.6.2 Results

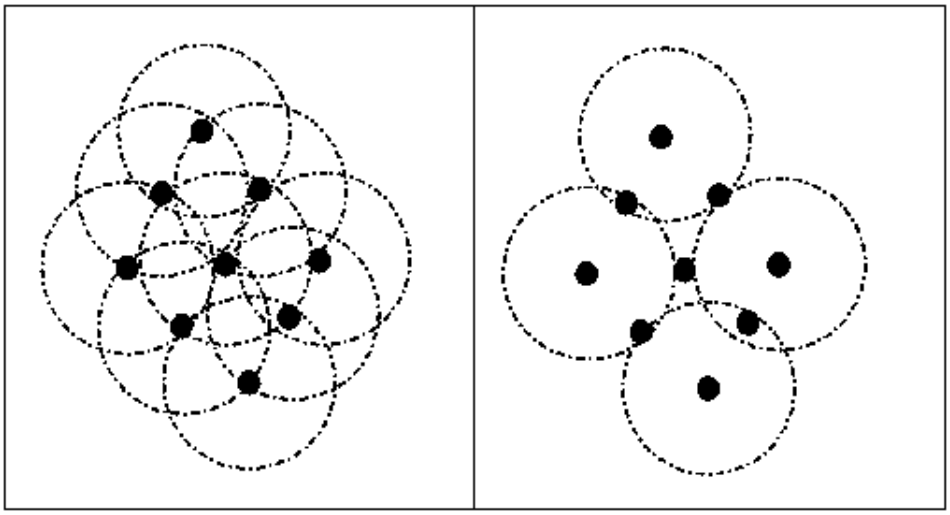

Figure 5.3: Too many active nodes does not necessarily mean radical network coverage improvement

Figures 5.4 to 5.12 depict the results of our experiments. The plots mainly show a comparison between the performance of different models with respect to metrics introduced in section 5.3 under different task demand intensities. Some of the graphs also show the task prioritisation behaviour of the various action selection models. The following sections will state observations drawn from the graphs with some possible explanations.

Network Lifetime

Constant Response networks (CR networks) maintained a constant level of resource consumption throughout the experiments. This was simply because they had constant response rates, independent of variations in task demand levels (figure 5.4). In a network where events are rare, this type of response model might result in unnecessary consumption of resources at

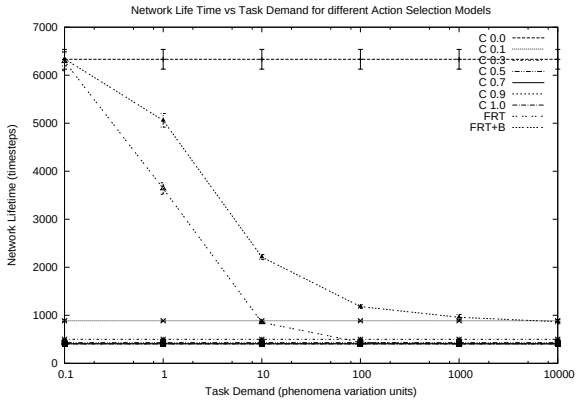

Figure 5.4: Network Life vs Task Demand

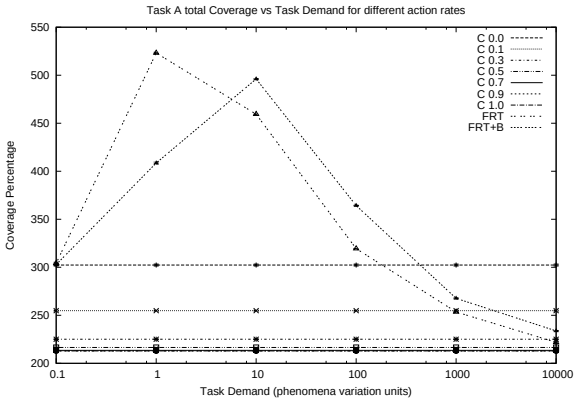

Figure 5.5: Total Task A Performance vs Task A Demand

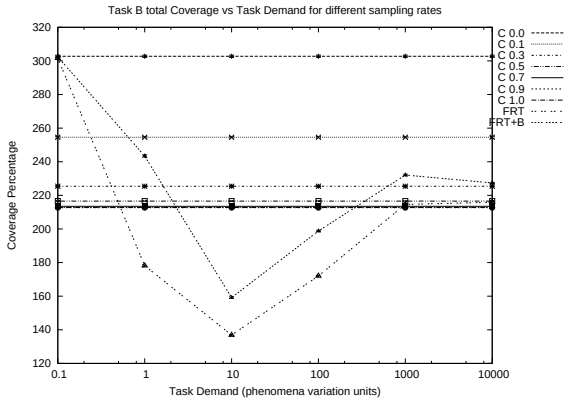

Figure 5.6: Total Task B Performance vs Task B Demand

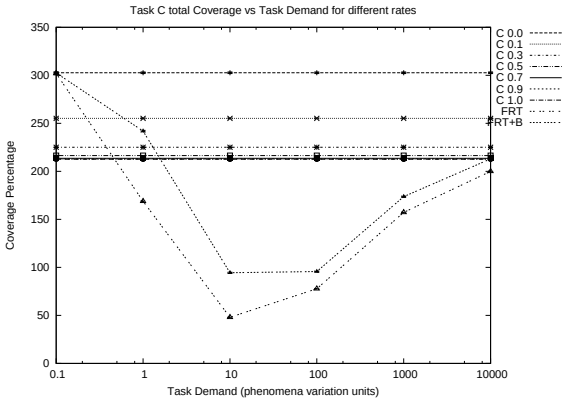

Figure 5.7: Total Task C Performance vs Task C Demand

low task demand, and a possibly sub-optimal network coverage when task demand is high (figures 5.5 to 5.7). In many sensor networks, such depletion of resources that, otherwise, could be used to elongate the network lifetime or perform extra tasks, is highly undesirable. On the contrary, the lifetime of the Fixed Response Threshold network (FRT network) varied with different task demand levels (figure 5.4). The fact that the FRT network preserved resources when the task demands were low, and dedicated more resources with the increase of the task demand led to a corresponding variation in network lifetime. Essentially, the response of the FRT network was in proportion to the task demand (figures 5.5 to 5.7).

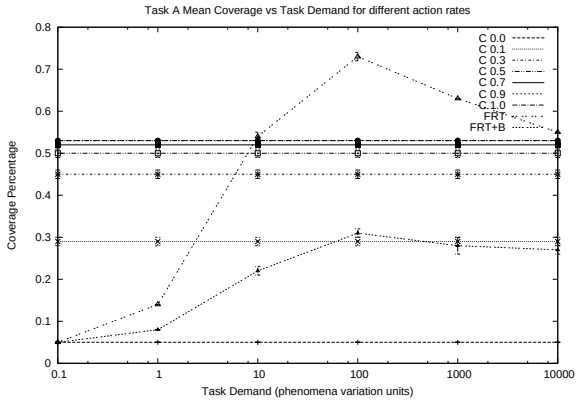

Figure 5.8: Average Task A Performance vs Task A Demand

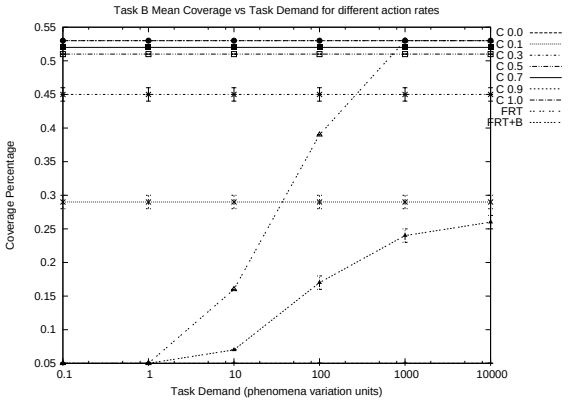

Figure 5.9: Average Task B Performance vs Task B Demand

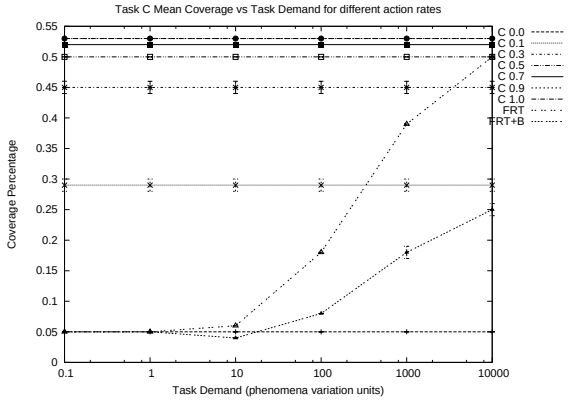

Figure 5.10: Average Task C Performance vs Task C Demand

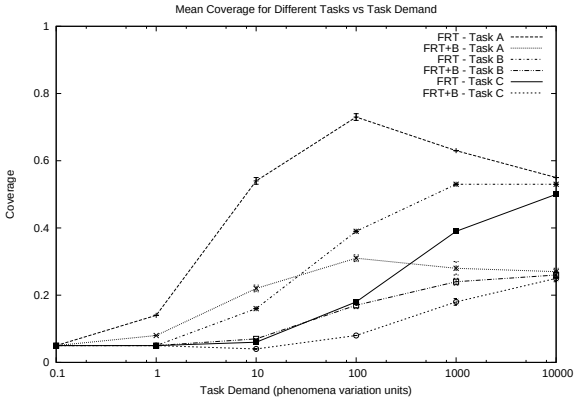

Figure 5.11: FRT Average Task Performances vs Task Demand

The battery-modulated network, FRT+B, always achieved better network lifetime (figure 5.4). The improvement ranged from 10% to just above 100%, i.e. lifetime more than doubled.

The lifetime gain varied with the task demand detected by the nodes. The best results were at relatively medium to high task demand levels (e.g. from 10 variation units per time step in figure 5.4). Marginal lifetime improvement were observed at extremely low task demand levels as the network was nearly idle most of the time. For many sensor network applications, this does not constitute a concern as at such low workload, the need to conserve energy is minimal.

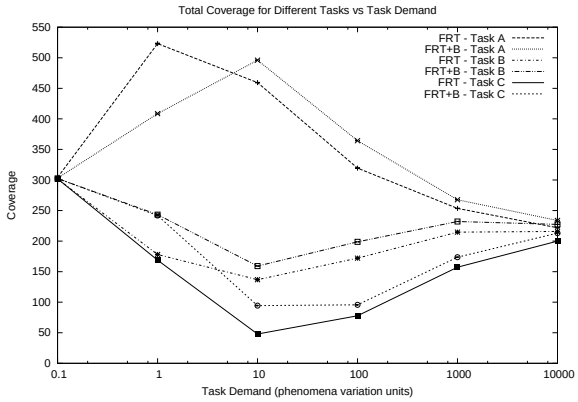

Figure 5.12: FRT Total Task Performances vs Task Demand

Mean Coverage

FRT networks were configured to give different priorities to tasks T_A, T_B, and T_C. This is represented by giving the tasks different response thresholds (Task T_A, T_B, and T_C had a threshold of 100, 300, and 1000 respectively). At low demand for task T_A, the FRT network lived longer than most CR networks (figure 5.4) and produced a higher total coverage (figure 5.5). However, the FRT network, at low demand for task T_A, had a low average performance (figure 5.8), i.e. at any instant during the experiment there were only few nodes active, and accordingly the coverage per time step was low. This might not be disadvantageous because high coverage is unlikely to be needed when the demand is low in many sensor network applications. At high task T_A demand, the FRT network gave a very high average coverage compared to many CR networks(figure 5.8). However, the total coverage was comparatively low (figure 5.5). This low total coverage resulted from the fact that the vast majority of nodes in the FRT network were active simultaneously yielding the best possible coverage to meet the high task demand imposed on the network by the environment, without considering any associated resource consumption. This is realistic and desirable in many real life applications. For example, this can be true if our Pollution Monitoring associated stimulus was extremely high, signalling a probable hazardous chemical leakage event, the sensor nodes should

certainly start all analysing fervently what is happening in the environment to give the human users as accurate a view of the environment as possible in such a critical situation. Similar observations can be made with regard to the other two tasks, namely T_B and T_C (figure 5.6, 5.7, 5.9, and 5.10). However, task T_A performance was invariably the highest at any demand level, with task T_C (the least priority task) constantly having the lowest performance amongst the three tasks(figure 5.11 and 5.12). This shows how the FRT network addressed the task prioritisation issue.

The average coverage is consistently lower for the FRT+B network, except at extremely low task demand levels. This is because this network sometimes, when energy is scarce, compromised performance in favour of saving energy. At extremely low task demand levels, both the FRT+B and FRT networks were idle most of the time and so their performance was very similar.

Total Coverage

Peaks can be seen clearly in the total coverage graphs (figure 5.11). These represent the optimal point of coverage at which the network was driven by environmental cues to take readings at a rate that optimally balances the accuracy of monitoring and the associated resource consumption (see the network on the right in the figure 5.3). At extremely high task demand, the network is required to perform at an extremely high rate, i.e. very many nodes are active simultaneously, which shortens the network's lifetime dramatically, yet does not produce noticeable gains in network coverage. From the diagram in figure 5.3, you can see that having a large number of nodes simultaneously active to monitor a phenomenon (the network on the left in figure 5.3) does not necessarily result in noticeable coverage improvement if the network is dense with nodes (high node coverage, or many nodes within each other's coverage range). This is because nodes will cover areas already covered by other neighboring nodes.

At low task demand levels for T_A, the FRT network achieved better total performance (coverage) than the FRT+B network (figure 5.5). This is because the nodes in the FRT+B network had two factors dampening their

task response. One is the low demand level and the other is the gradually decreasing battery level. These can be represented mathematically by the following equation:

$$\lim_{S\to 0,B\to 0}(\frac{S}{S+\theta})(\frac{B}{B_0}) = 0 \tag{5.6}$$

Equation 5.6 indicates that the FRT+B response decreases faster than that of the FRT (equation 5.17) because of the additional battery modulated term. At high task demand levels, the situation is reversed with FRT+B having the higher edge in terms of total performance. This is because the FRT is pushed by the high demand levels to consume its resources, i.e. energy, quickly and consequently yielding low total performance. Whereas in the FRT+B network, nodes are pushed to perform harder by the high task demand levels, but in the same time performance is dampened by the decreasing battery levels, so compromise is made on average performance to achieve better total performance. Equation 5.19 and the following equation mathematically represent the difference between the two networks at high demand levels:

$$\lim_{S\to \infty,B\to 0}(\frac{S}{S+\theta})(\frac{B}{B_0}) = 1 \cdot 0 = 0 \tag{5.7}$$

Similar arguments to those given to explain T_A's figures can be used to explain the graphs for T_B (figure 5.6).

The total performance curves of the FRT+B and FRT networks with respect to T_C were different from these of T_A and T_B (figure 5.7). The FRT network performed worse than the FRT+B network across all levels of task demand. Modulating the FRT model with a battery-based term alleviates the effect of the task demand in governing the response pattern of individual nodes as well as that of the network. The network response to T_C's task demand is low, compared to T_A and T_B because it has a much higher task-associated threshold. The battery modulation component of the FRT+B model has two effects that explain the T_C curve shapes:

- It reduces a node's *absolute* responsiveness to task demands for the three tasks. This can be represented mathematically as:

$$\begin{aligned} &\forall \quad \theta > 0,\ B > 0,\ B_0 > 0,\ B < B_0 \ : \\ &\text{If} \quad T(S,B) = T(S) \cdot (\frac{B}{B_0}) \\ &\text{then} \quad T(S,B) < T(S) \end{aligned} \tag{5.8}$$

- It reduces the *relative* difference between the responsiveness of a node to the three tasks. This can be represented mathematically as:

$$\begin{aligned} &\forall \quad \theta_A > 0,\ \theta_B > 0,\ B > 0,\ B_0 > 0,\ B < B_0,\ |T_A(S) - T_B(S)| > 0 \ : \\ &\text{If} \quad T_A(S,B) = T_A(S) \cdot (\frac{B}{B_0}) \quad , \quad T_B(S,B) = T_B(S) \cdot (\frac{B}{B_0}) \\ &\text{then} \quad |T_A(S,B) - T_B(S,B)| \ < \ |T_A(S) - T_B(S)| \end{aligned} \tag{5.9}$$

For example, assume that a node has the following settings:

$$\begin{aligned} T_A(S) &= 0.8 \\ T_B(S) &= 0.6 \\ T_C(S) &= 0.4 \\ |T_A(S) - T_C(S)| &= 0.4 \\ |T_A(S) - T_B(S)| &= 0.2 \\ |T_C(S) - T_B(S)| &= 0.2 \\ \frac{B}{B_0} &= 0.5 \end{aligned}$$

After applying the battery-based component ($\frac{B}{B_0} = 0.5$), the resultant values are:

$$\begin{aligned} T'_A(S) &= T_A(S) \cdot (\frac{B}{B_0}) = 0.8 \cdot 0.5 = 0.4 \\ T'_B(S) &= T_B(S) \cdot (\frac{B}{B_0}) = 0.6 \cdot 0.5 = 0.3 \\ T'_C(S) &= T_C(S) \cdot (\frac{B}{B_0}) = 0.4 \cdot 0.5 = 0.2 \\ |T'_A(S) - T'_C(S)| &= |0.4 - 0.2| = 0.2 \\ |T'_A(S) - T'_B(S)| &= |0.4 - 0.3| = 0.1 \\ |T'_C(S) - T'_B(S)| &= |0.2 - 0.3| = 0.1 \end{aligned}$$

Note that the new probabilities are less than the original ones, and the difference between the original probabilities is double the difference between the new ones. This leads to the difference between the number of nodes that are active for each task to be much closer when the battery term is applied. That is why at low to medium task demand levels, the total coverage for T_C in the FRT+B network is higher than that of the FRT network. However, at high task demands, the FRT+B achieved better total coverage for the same reasons the network performed better at high demand levels in terms of total performance for tasks T_A and T_B.

Task Prioritisation

Figures 5.11 and 5.12 suggest that the network follows the priority rules in performing the tasks, i.e. monitoring the phenomena. As required, under the same task demand, tasks with high priority have higher chance to be performed than tasks with low priority. However, the differences between the responses to different tasks converge at very low task demand as well as at very high task demand levels. This is because the network response threshold model gives very high response probability to high demand tasks, i.e. task-associated demand is much higher than the task-associated threshold, and very low one to low demand tasks. If all tasks experience demands much higher than their thresholds, the node's probability to respond to any of the tasks is very high and the prioritisation, represented by the thresholds, is blurred (figures 5.12 and 5.11). Mathematically, this can be represented as follows:

$$if \quad S_A \gg \theta_A \quad then \quad \Psi(S_A, \theta_A) \rightarrow 1 \tag{5.10}$$

$$if \quad S_B \gg \theta_B \quad then \quad \Psi(S_B, \theta_B) \rightarrow 1 \tag{5.11}$$

$$if \quad S_C \gg \theta_C \quad then \quad \Psi(S_C, \theta_C) \rightarrow 1 \tag{5.12}$$

A similar argument applies when the network experiences very low demand levels for all tasks.

$$if \quad S_A \ll \theta_A \quad then \quad \Psi(S_A, \theta_A) \rightarrow 0 \tag{5.13}$$

$$if \quad S_B \ll \theta_B \quad then \quad \Psi(S_B, \theta_B) \rightarrow 0 \tag{5.14}$$

$$if \quad S_C \ll \theta_C \quad then \quad \Psi(S_C, \theta_C) \rightarrow 0 \tag{5.15}$$

5.6.3 Conclusion

The observations presented in the results section suggest that a network that can adapt to the the environmental conditions can be advantageous in dynamic unpredictable environments, for example, when monitoring phenomena that happen rarely (e.g. volcanic eruptions, forest fires, chemical leakage, storms). Even in not so rare events like traffic jams (twice a day?), great longevity gains will be achieved if intensive task performance only occurs during the interesting event, otherwise, the node is in idle mode. Such networks, which utilise adaptive algorithms (e.g. Fixed Response-Threshold Model), will be able to operate for longer, by consuming its resources only when they are most needed. In the same time, compromises on performance will be minimal in the case of intensive coverage demands.

Threshold-based models seem to be effective in prioritising tasks and in the same time take the task context into consideration as well as weighing up the task demand in both relative and absolute terms.

The experiments of this section also show that FRT+B manages to improve the network longevity. However, this comes at a degree of compromise on the coverage front.

From the results, battery-modulated systems appear to perform better in applications where there is room for data extrapolation. For example, in applications where changes are not erratic or abrupt such that missing points can be predicted from available ones. It also fit applications that are tolerant to noise and irregularity of the data collected. In addition, applications where the network lifetime is of a greater importance than the amount of data collected are good candidates for using the FRT+B model.

The FRT model can be more suitable for systems that have relatively abundant resources and do not need to make performance-compromising

decisions to maintain network longevity, however, can utilise adaptivity to manage task prioritisation and network lifetime implicitly.

Next section will compare various threshold-based models (with a base CR model) in applications where task switching is an expensive operation.

5.7 CR, FRT, FRT+B, VRT, and VRT+B

In some applications, task-switching is a resource-consuming action that should be avoided whenever possible. This is especially so in minimalist resource-constrained sensor nodes. The Variable Response-Threshold (VRT) Model addresses this issue by enabling nodes to learn which tasks they have performed previously, and subsequently reinforces the tendency of nodes to perform these tasks in future. In contrast, tasks that has been rarely performed by a node are less likely to be performed in the future.

In this section, we conduct experiments to investigate, and study the differences between the VRT and FRT models if applied in sensor networks scenarios. In addition, we conducted experiments to test the effects of addressing the resource consumption profiles using the FRT+B model in comparison to using the VRT+B model. Both VRT+B and FRT+B aim at improving network longevity by considering node energy when making action selection decisions. Comparing the various models in this section furthers the understanding of their dynamics. Appendix B.2 lists the settings and parameter values for this set of experiments.

5.7.1 Experiments Objectives

This set of experiments are designed to achieve the following:

- Investigate the difference between sensor network performances when employing the FRT/FRT+B and VRT/VRT+B models under various task demand intensities.
- Test the suitability of the VRT/VRT+B models for applications where task switching is undesirable, as opposed to the FRT/FRT+B models respectively.

- Investigate the system dynamics resulting from the specialisation of nodes in the VRT/VRT+B networks as opposed to those dynamics appearing in the FRT/FRT+B networks.
- Identify, through the differences in behaviour, which type of applications would benefit from which model.

5.7.2 Results

We ran experiments for each of the CR, FRT, FRT+B, VRT and VRT+B models under different task demand levels, with the settings specific to this section, detailed in table B.2 in the appendix. Figures 5.13 to 5.21 represent the results of our simulations. The coming few section will discuss various observation from these graphs.

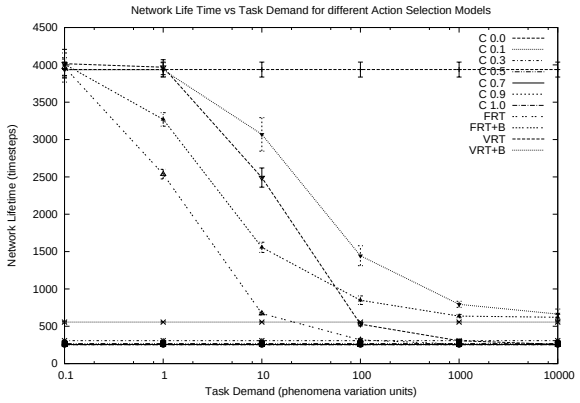

Figure 5.13: Network Lifetime of VRT vs Lifetime of FRT

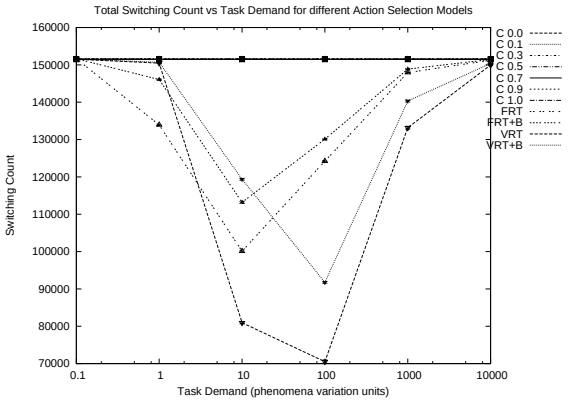

Figure 5.14: Total Switching Frequency for the VRT and FRT networks

Network Lifetime

The VRT network consistently had a longer lifetime (figure 5.13) because the reduced switching resulted in energy-savings, allowing the VRT network to last longer. However, the difference between the life time of the two networks diminished when task demands grew to extremely high levels, or dropped to extremely low levels. This is because, according to the VRT model, the decision-making process becomes dominated by the enormity, or

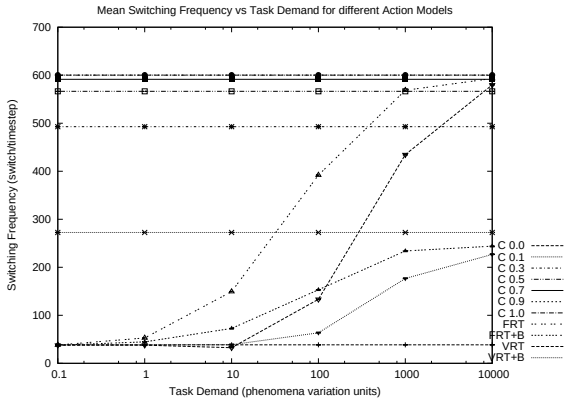

Figure 5.15: Average Switching Frequency for the VRT and FRT networks

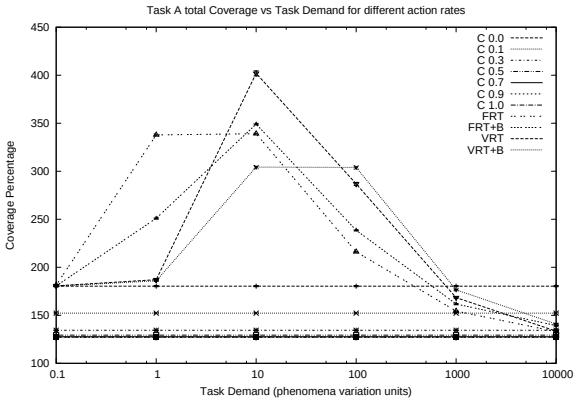

Figure 5.16: Total Task A Performance for the VRT and FRT networks

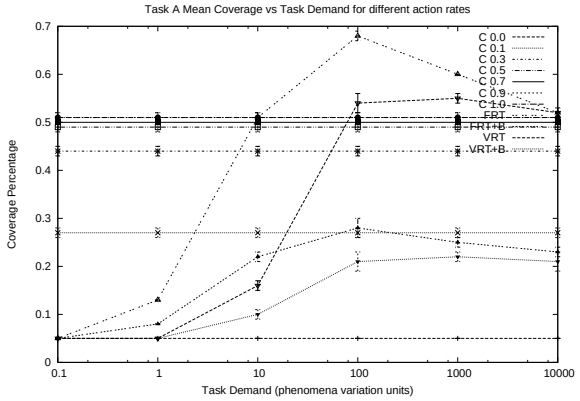

Figure 5.17: Average Task A Performance for the VRT and FRT networks

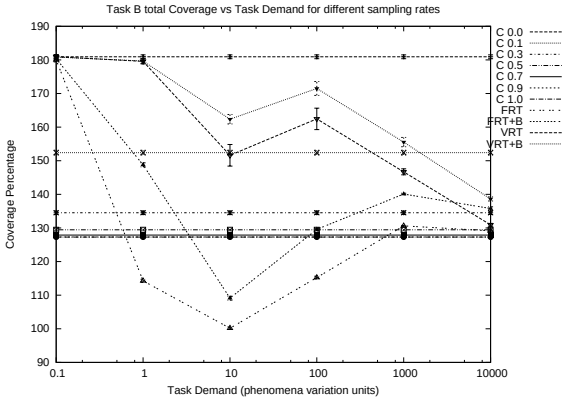

Figure 5.18: Total Task B Performance for the VRT and FRT networks

minuteness, of the task demand at extreme values. This can be represented mathematically by the following equations:

$$\lim_{S \to 0} \Psi(S, \theta) = \lim_{S \to 0} \left(\frac{S}{S + \theta}\right) = 0 \tag{5.16}$$

$$\lim_{S \to 0} \Psi(S) = \lim_{S \to 0} \left(\frac{S}{S + \theta}\right) = 0 \tag{5.17}$$

$$\lim_{S \to \infty} \Psi(S, \theta) = \lim_{S \to \infty} \left(\frac{S}{S + \theta}\right) = 1 \tag{5.18}$$

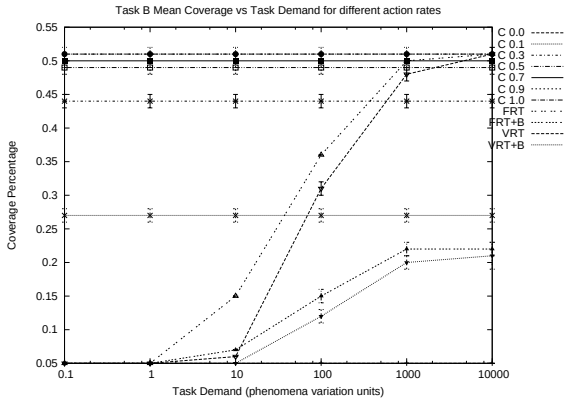

Figure 5.19: Average Task B Performance for the VRT and FRT networks

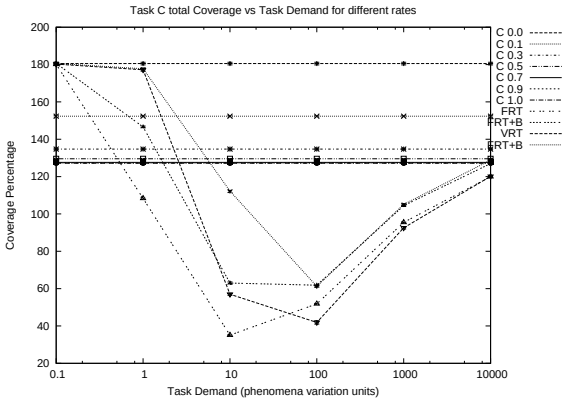

Figure 5.20: Total Task C Performance for the VRT and FRT networks

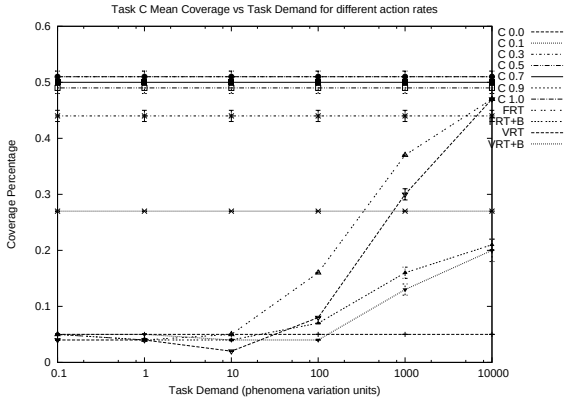

Figure 5.21: Average Task C Performance for the VRT and FRT networks

$$\lim_{S\to\infty} \Psi(S) = \lim_{S\to\infty} (\frac{S}{S+\theta}) = 1 \tag{5.19}$$

Equation 5.16 represents the value of the response probability at extremely low task demand levels for the VRT network. According to equation 5.17, the FRT network have the same response probability at such task demand levels. Figure 5.13 agrees with these mathematical equations. Similarly, equations 5.19 and 5.18 indicate that the two response probabilities converge to the value 1 at extremely high task demand values, and figure 5.13 depicts such observation. Roughly speaking, At extremely high demand levels, the benefits of performing an urgently needed task may

outweigh the cost involved in switching and, therefore, nodes do not attempt to save energy by following specialisation rules. Instead, they do their best to satisfy the currently high task demands. At extremely low demand levels, nodes do not benefit from specialisation because there is no need to perform any tasks in the first place. Idle nodes that can specialise and those that cannot use energy almost at the same rate. This explains the similarity in terms of lifetime length between FRT and VRT networks at extremely low demand levels.

In real networks, these mathematical equations and simulation results could translate to the following example situations:

- If a node in a network, which adopts FRT or VRT models, experiences extremely high task demands, it may disregard the cost involved in task-switching in favour of providing a high quality service. For example, if a node in our sensor network specialised in performing the weather monitoring task and the node experiences an extremely high pollution levels (a chemical leakage might be happening), then the node might switch to pollution-monitoring task in order to contribute to recording the possibly most accurate representation of the pollution phenomenon at this critical point.

- If a node in a network, which adopts FRT or VRT models, experiences extremely low demand levels, it may disregard the cost involved in task-switching as the computational overhead of specialisation may exceed that of task-switching. For example, if a node in our sensor network experiences very little variation of weather measurements for very long time, it might opt for shutting down itself temporarily. When it switches back on, it may be acceptable to pick any of the three tasks randomly to perform, rather than retrieve data about its task specialisation that might require some processing and management overheads with no tangible benefits.

The VRT+B network lifetime was always longer than that of the VRT network. The gain in network life time was small at low task demand levels, whereas it was a double lifetime gain at medium to high task demand levels. The following equations explain this mathematically:

$$\lim_{S\to 0} \Psi(S,\theta) = \lim_{S\to 0} \frac{S}{S+\theta} = 0 \tag{5.20}$$

$$\lim_{S\to 0} \Psi(S,\theta,B,B_0) = \lim_{S\to 0} (\frac{S}{S+\theta})(\frac{B}{B_0}) = 0 \tag{5.21}$$

From equations 5.20 and 5.21, at extremely low task demand levels, nodes in both networks tend to conserve energy predominantly because there is no task demand rather than because of any energy concerns. Therefore, the behaviour of the two network converge, with a slightly greater tendency to conserve on the side of the VRT+B network. However, at high task demands levels, equations 5.20 and 5.21 are transformed into:

$$\lim_{S\to 0} \Psi(S,\theta) = \lim_{S\to \infty} \frac{S}{S+\theta} = 1 \tag{5.22}$$

$$\lim_{S\to 0} \Psi(S,\theta,B,B_0) = \lim_{S\to \infty} (\frac{S}{S+\theta})(\frac{B}{B_0}) = \frac{B}{B_0} \tag{5.23}$$

equations 5.22 and 5.23 indicate that the VRT network lifetime depends solely on the task demand, making it work harder at high task demands, and so have a shorter lifetime. On the other hand, the VRT+B network's lifetime depends on the battery modulation term as well as the task demand, which dampens the response probability in order to conserve energy, allowing the network to live longer.

Mean Coverage

For T_A's average performance (figure 5.17), the number of specialised nodes in performing T_A in the VRT network is equal to the number of non-specialised nodes that perform T_A at any time step in the FRT network. Actually, the VRT network behaves like a FRT one in terms of the average performance for T_A. At t_0 of any run of the VRT network, no nodes are specialised yet, i.e. the VRT network is equivalent to a FRT one. at t_1, some nodes, say N_A, will be performing T_A, and are likely to specialise in performing it. At t_2, the specialisation begins to appear in stronger terms, and the nodes that performed T_A at t_1 have a higher probability of performing T_A at t_2. This process results in roughly N_A nodes performing T_A at each

time step in a VRT network run. In a run of an FRT network, at each time step, the network resembles a VRT network at t_0, i.e. N_A nodes are likely to perform T_A. This yields the almost identical average performance for T_A by both networks.

In terms of the average coverage for T_B (figure 5.19), the FRT network was superior to the VRT one throughout the task demand spectrum, except at the very extremes. This exception was because specialisation in the VRT network resulted in a large number of nodes performing T_A at any time step, and much lower number of nodes performing T_B compared to the situation in the FRT network where any node could perform any task, regardless of any switching cost involved.

In terms of average performance, the T_C performance of the VRT network was lower than that of the FRT, except at extremely high or low demand levels, where it was very similar (figure 5.21). As discussed in a previous paragraph of this section, this was a result of diminishing specialisation effects in the VRT network at extreme demand levels. Essentially, at those extremes, the behaviour of the VRT network is reduced to that of a FRT one.

Looking at the average performance in the case of T_A (figure 5.17), at low task demand levels, both networks perform similarly for the same reasons given to explain the same observation for the total performance figure (see next section). However, at high task demand levels, the VRT network boosts its performance to satisfy the network needs (network coverage), disregarding any resource consumption incurred, and hence achieving maximum possible average performance. Meanwhile, the VRT+B network increases its performance but with less intensity in an attempt to conserve energy, resulting in a lower average performance at these task demand levels.

What was said about T_A applies to T_B in terms of average performance (figure 5.19) for comparing the VRT and VRT+B models.

In terms of T_C's average performance, the VRT network maintained higher average task performance at medium to high task demand values in comparison with the VRT+B network. While both networks behaved almost identically at low task demand values (figure 5.21). These observations can be explained by the same arguments presented previously to

compare the FRT and VRT networks T_C's average performances.

Total Coverage

The network performance (or coverage in our simulation scenario) differed from one task to another (figures 5.16 to 5.21). For T_A, the VRT network performed much better than the FRT network in terms of total performance (figure 5.16). This is because T_A had the highest priority (lowest threshold) amongst tasks, so a large number of nodes in the VRT network specialised in performing it. In addition, the VRT network lived longer than the FRT network for the reasons detailed previously. Both facts resulted in this superior T_A's total performance for the VRT network compared to the FRT one.

For T_B's total performance (figure 5.18), the VRT network performed only slightly better at medium to high task demand levels, and worse otherwise. This is because at small to medium demand levels, most nodes in the VRT network specialise in performing T_A, leaving only a small number of nodes to specialise in performing T_B. At high demand levels, the specialisation effects of the VRT model become substantially diluted (see equation 5.18 and 5.19), which results in the convergence of the total coverage of both FRT and VRT networks. Similar observation can be recorded at extremely low demand levels (figure 5.18 and equations 5.17 and 5.16).

For T_C, at extremely low task demands, the two networks had identical total performances (figure 5.20). As demand increased towards small to medium values, the VRT network specialised in performing the higher priority tasks, i.e. T_A and T_B, leaving only very few nodes to perform T_C, while the FRT network responded only according to the demand levels it detected. This resulted in a lower VRT total performance for T_C in comparison to FRT. When demand reached medium to high values, the specialisation effects in the VRT network subsidised and the performance of both networks converged with a slightly higher edge, in terms of total performance, for the VRT network.

For T_A, the VRT+B network had a higher total coverage at medium to high task demand values (figure 5.16), whereas the VRT network's total

performance was higher at small to medium task demand values. This can be explained by the two response-dampening forces, the low task demand levels and the decreasing energy supplies. Applying these two forces combined to the VRT+B network, while the VRT network has only one response-dampening factor, i.e. the low task demand, causes the VRT network to work closer to the optimal coverage level than the VRT+B network producing higher total performance.

What was said about T_A applies to T_B in terms of total performance (figure 5.18) for networks VRT and VRT+B.

For T_C, the VRT network has always achieved a lower total performance (figure 5.20). This is because most nodes in the VRT network specialised to perform either T_A or T_B, leaving very few nodes to perform T_C. This amounted to a low total performance at any task demand level. On the other hand, in the VRT+B network, the battery factor dampened the tendency of some nodes to specialise, and therefore the less specialised nodes did more frequently perform T_C resulting in a better T_C total performance than that of the VRT network.

Switching

The VRT network considers switching cost when making action selection decisions, which can be seen as a context-aware decision making mechanism, or more specifically *switching-aware action selection*. In terms of the specialisation resultant from the learning and forgetting mechanisms embedded in the VRT model, the action selection process can be described as a specialisation-oriented process for action selection, and we termed this mechanism *specialisation-oriented action selection*. switching-aware/specialisation-oriented action selection can result in energy-savings on two levels:

- The frequency of performing the resource expensive switching process is largely reduced, and therefore resources that are normally consumed to perform switching are conserved to serve other purposes.
- As a node specialises in performing only a subset of the tasks, workload is reduced on this node, and hence it has less energy needs, which

contributes to the elongation of its lifetime.

As expected, the VRT network consistently switched less than the FRT network (figure 5.14 and 5.15). The VRT model caused nodes to specialise in performing a set of tasks and not others and so nodes switched tasks less frequently. In other words, nodes in the VRT network favoured performing tasks that they were specialised in over other tasks.

It is worth noting that when the environment produces extremely low task demand, the battery modulation has minimal effect on the system average coverage, total coverage, or network life time. On the contrary, the effect of the battery modulation is at its peak when the environment exhibits high task demands. This is viable because high task demands require more efficient regulation than low task demands do.

5.7.3 Conclusions

The results of the experiments suggest that the specialisation that VRT and VRT+B networks provide have resulted in longer lifetimes for sensor networks with only slight compromise in terms of performance. In contrast, FRT networks use more energy because their higher flexibility comes with the cost of the task switching overhead.

We can conclude that VRT networks are generally suitable for applications that are tolerant to slight performance compromises in favour of longevity of operation, especially when task switching is resource expensive or undesirable. Whereas FRT networks in general would perform better for applications with high performance requirements and cheap task switching.

The experiments of this section suggest also that VRT models help reduce switching, prioritise tasks, and control a network's response to match the task demand detected. It falls short of addressing the resource consumption incurred by certain action selection decisions. VRT+B networks rectify this shortcoming of the VRT model, however on the expense of average task performance.

The VRT+B model would be preferred, as shown in the experiments, to the VRT model in applications where network longevity is a paramount

requirement. Whereas the VRT model will fit applications where resources are abundant (energy in our experiments) or irrelevant in making action selection decisions. This could be for example if energy harvesting techniques were incorporated in the nodes, i.e. energy sometimes become abundant, and then the nodes in our scenario may switch to VRT or FRT model as opposed to VRT+B or FRT+B model. Also possible application like Building Security Sensor Networks, where sensors could be connected to the mains most of the time, can benefit from the VRT and FRT models.

5.8 Summary

In this chapter, we conducted several simulations to explore, investigate and study the effect of using a family of adaptive algorithms on the performance and efficiency of a sensor network in a hypothetical scenario.

The biologically-inspired model, whose variations were used in this chapter's experiments, is the response-threshold model. The main two variations of this model are the Fixed Response Threshold Model and the Variable Response Threshold Model. We extended the variation models to include a resource component in the decision making process.

The results show that each of these models, variations ,and extensions benefit networks with certain characteristics and requirements. As expected there is no generic answer model to all sensor network applications. For example, we found that the Variable Response Threshold Model can be used in networks that have resource expensive switching processes, while the FRT model is suited to networks with tasks that have different priorities and need to control the network action only according to the detected demand for different tasks.

The simple thresholding models used in this chapter managed to balance between opposing goals at various contexts to make fairly sensible decisions without resorting to complex processes.

Both the FRT and the VRT models depend on driving a node's behaviour based on information collected about the environment, user policies, and the conditions of the node itself. How much weight is put on any of these factors can be decided at node design stage according to the appli-

cation requirements. An alternative would be using some form of genetic algorithm or any of the other self-learning mechanisms to perform this weight/importance analysis dynamically and automatically.

Taking a node's resources into consideration when making decisions involves a trade-off between the network performance and its longevity. The adaptive algorithms we experimented with in this chapter try to find the optimal point of balance between these two important aspects of a sensor network's operation.

The next few chapters will expand on employing the same family of adaptive algorithms at different points of our envisaged sensor node architecture (fig 3.3), namely at communication module, application re-scheduling (or discontinuation module), and sampling modules.

Chapter 6

Task Discontinuation Model

In the previous two chapters, we studied and discussed the mechanism by which nodes within a network make action selection decisions. Nodes are assumed, in this dissertation, to be single-tasked, as opposed to multi-tasked (although multi-purpose). This requires nodes to find a mechanism to make decisions about when to quit performing a task. In the experiments of the last two chapters, we used a constant probability like the one in [22] to make discontinuation decisions. In this chapter, we will extend this to use a more capable adaptive algorithm.

6.1 Non-Adaptive Discontinuation Schemes

In the previous chapter, a preset interval was used by a node to discontinue performing a task, though probabilistically. A non-stop task performance can also be used. Both, and similar, schemes do not take into consideration the need to perform a task, or rather the variability of the need to perform a task, i.e. task demand. For example, in our network scenario, if a node is performing the pollution monitoring task, it might not need to report the static or stable air pollution level. In such case a non-stop scheme will do redundant, unnecessary, and possibly costly work. Most systems that adopt the non-stop active scheme disregard the cost of task performance, either because it is irrelevant to the application or because the resources needed to perform a task are abundant [51][121][122]. However, this is rarely the case

in many sensor network applications. We will next explore alternatives to such schemes.

6.2 The Constant Discontinuation Probability Model (CD)

The previous chapter discussed the use of threshold-based models to help individual nodes in a sensor network make decisions whether to engage in performing a task, or not. but it does not address the mechanism used by individuals to make decision to quit performing a task. In the model described in [22], an individual that is engaged in performing a task discontinues performing it with a constant probability p. The value of the probability is preset before the system is employed, i.e. at design time. The authors of the model justified using a fixed discontinuation probability in their model of task allocation in social insects by stating that it is supported by experimental data from real insect colonies. The authors of the model used the same fixed discontinuation mechanism in their experiments on both variations of the model (see section 4.4 for FRT and section 4.5 for VRT).

Setting the value of the discontinuation probability p to a relatively low value (for example it was set to 0.02 in Bonabeau [24]) mean that individuals who decide to engage in performing a task will on average spend $1/p$ time units performing it. When p is low (p approaches 0), the time spent by individuals on performing the task is relatively long ($1/p$ approaches ∞). Although this may work well in situations where it takes long for the task demand levels to subside, it will cause extra work in environments where demands change rapidly, or only appear sporadically. On the contrary, if p is high (p approaches 1), the time spent by individuals on the task will be short ($1/p$ approaches ∞). That may work well in environments with highly dynamic demand levels, but it will perform poorly in slowly changing environments, for example by introducing unnecessary high task switching rate with the associated overhead. Switching rates may incur overheads in time and quality of collected data or it might complicate the

decision making process. Medium values will give less than optimum results at either extremely high or extremely low demand variations (very slowly and very rapidly changing phenomena). In addition, such values will not be optimal for continuous spectrum demand levels. For example, wind speed variation spans a wide range of values. In many climates, it may not be possible to find a sampling rate that matches that of the wind speed variation for long periods of time. Whereas sensor networks are assumed to be deployed for extended periods of time (months if not years). The mathematical representation of the time spent by a node performing a task with discontinuation probability p is as follows:

$$T(Task) = \frac{1}{p} \tag{6.1}$$

$$\lim_{p \to 1} T(Task) = \lim_{p \to \infty} \frac{1}{p} = 1 \tag{6.2}$$

$$\lim_{p \to 0} T(Task) = \lim \frac{1}{p} = \infty \tag{6.3}$$

where $T(Task)$ is the time spent by a node in performing a task. p is the probability of discontinuing a task. These equations are disregarding the action selection decisions effects on the time spent performing a task, focusing instead on the effects of the value of the discontinuation probability. For example, if the probability of engaging in performing a task is relatively low, the total time a node spends performing a task is decreased compared to high task engagement probability. However, at similar task engagement probabilities, these equations will hold true, i.e. given a node has started performing a task, these equations apply.

Sensor networks are often employed in unpredictable environments. For example, our sensor network may experience no traffic to monitor for all night. When the rush hours arrive, the sensors in the area will have massive environment changes to report. In such a scenario, low values of discontinuation probability in normal no-traffic conditions might be inappropriate, while a high discontinuation probability may cause unnecessary task switching when traffic is heavy for an extended period of time (for example on a big event day). In the next few sections, we will extend the

threshold-based model provided by Bonabeau in [24] and [25], and Theraulaz in [181] to control task discontinuation within a sensor network node. Later, in the next chapter, simulations and findings related to this extension will be presented.

6.3 Fixed Stimulus-Based Discontinuation Threshold Model

The way many living systems survived for millions of years is said to be their ability to adapt to the environment [197]. We see the same concept applicable to sensor networks. If sensor networks can adapt to the stress the environment exerts on them, they may stand better chance to survive for longer periods of time, as well as provide better task performances. This adaptability is crucial in many behavioural aspects of the network including the task discontinuation decisions. Individual nodes may benefit from being able to make the decision of quitting performing a task dynamically according to the data it can gather, rather than sticking to one rigid hard-coded discontinuation scheme.

To make the idea clearer, let us look at our example scenario. If an individual node observes heavy rain, it may engage in high rate environmental monitoring and reporting task. While it is sending readings to a base station, a traffic congestion occurs, and the rain stops. It might then increase its discontinuation probability with respect to the environment-associated task because likely there is no need any more to pursue this task. This will allow the traffic monitoring task to be performed in place of the weather monitoring task, resulting in a better overall task performance. Or may be the traffic congestion did not happen, hence the node might decide still to quit performing the weather-associated task to stay idle instead, in which case it will conserve some energy and consequently last longer.

When demand to perform a task is high, the discontinuation probability for that task is low. On the contrary, if an individual is performing a task, while the demand for that task is low, it is likely that the individual should quit performing the task since there is no or little need to perform it. This can

be modeled mathematically, drawing from the response threshold model, as follows:

$$\Omega(\theta, S) = 1 - (\frac{S^n}{S^n + \theta^n}) \tag{6.4}$$

Where $\Omega(\theta, S)$ is the probability an active node will quit or discontinue performing a task given its threshold is θ and its task-associated stimulus is S. Note that $\Omega(\theta, S)$ approaches 0 as S approaches ∞ (there is very high demand for this task), and approaches 1 as S approaches 0 (there is little demand for this task). The following equations represent this mathematically:

$$\lim_{S \to \infty} \Omega(\theta, S) = \lim_{S \to \infty} [1 - (\frac{S^n}{S^n + \theta^n})] = 0 \tag{6.5}$$

$$\lim_{S \to 0} \Omega(\theta, S) = \lim_{S \to 0} [1 - (\frac{S^n}{S^n + \theta^n})] = 1 \tag{6.6}$$

In the next section, we incorporate to our discontinuation model a component to account for the resources available for a task within a node.

6.4 Adaptive Resource-Based Discontinuation Threshold Model

The previous section discussed the FDT extension of the FRT model in [22]. It used the task demand levels to decide either to discontinue performing a task, or continue performing it. In this section, we extend this model even further to include resource levels in task discontinuation decisions. We will in specific discuss the battery level as an example of a valuable resource that needs to be taken into consideration when making task discontinuation decisions. Sometimes, task demands are high, but resources required to perform the task are running low. In such circumstances, nodes may need to make compromises on the quality of task performance in favour of longer service lifetime. Accounting for battery levels in task discontinuation decisions is only an example of incorporating resource levels available

to nodes in making task-associated decisions. Battery level is an ideal example of a scarce resource in many sensor networks, and hence can be used without loss of generality.

The following equation is drawn from the FDT equation and uses the battery level, B to control the discontinuation probability of a task, referred to as $\Omega(B)$:

$$\Omega(B) = 1 - \frac{B^m}{B_0^m} \tag{6.7}$$

Algorithm 2 represents how a node would use battery levels and task demands to make decisions on task discontinuation:

1 $\Omega(S, \theta) = 1 - \frac{S^n}{S^n + \theta^n}$
2 $\Omega(B) = 1 - \frac{B^m}{B_0^m}$
// Now, simply take the most restrictive of the $\Omega(S)$
// and $\Omega(B)$
3 **if** $\Omega(S, \theta) < \Omega(B)$ **then**
4 $\quad \Omega(S, \theta, B) = \Omega(B)$
5 **endif**
6 **else if** $\Omega(S) >= \Omega(B)$ **then**
7 $\quad \Omega(S, \theta, B) = \Omega(S, \theta)$
8 **endif**
// where $\Omega(S, \theta)$ is the demand-based task
// discontinuation probability
// where $\Omega(B)$ is the battery-based
// discontinuation probability
// where B is the current battery level
// where θ is the task-associated threshold

Algorithm 2: Variable Response Threshold Algorithm

This algorithm results in the following mathematical equations representing the discontinuation probability trends:

$$\lim_{S\to\infty, B\to 0} \Omega(S, \theta, B) = \lim_{S\to 0, B\to B_0} \Omega(S, \theta, B) = \lim_{S\to 0, B\to 0} \Omega(S, \theta, B) = 1 \tag{6.8}$$

$$\lim_{S\to\infty, B\to B_0} \Omega(S, \theta, B) \;=\; 0 \tag{6.9}$$

These equations represent a generalised trend of node behaviour. If task demand is high and there is enough battery power, then the node is unlikely to discontinue. However, if either or both battery power and task demand are low, then the node will most likely discontinue performing the task. We will discuss in more detail the effect of such behaviour on the quality of task performance in the next chapter.

6.5 Summary

In this chapter we introduced an extension to the FRT model used originally in [22] to make action selection decisions. Our extension uses the model to make task discontinuation decisions. This extension can be seen as an additional layer of adaptivity which improves the performance and longevity of sensor networks. In the next chapter we will explain the simulations that were done on this model, together with the results. We also discuss some conclusions that can be drawn from those results.

Chapter 7

Task Discontinuation Simulations

In the previous chapter, we introduced a mathematical model to make task discontinuation decisions. In this chapter, we run simulations to investigate the effects of this model on the dynamics of a sensor network.

First we introduce the experiments scenario, then a discussion of the context of the experiments will follow. Next, the results in the form of several graphs and comments on their indications will be provided. Finally, some conclusions will be presented.

7.1 The Experiments Scenario

The goal of these experiments is to examine the effects of the use of adaptive discontinuation models in general, and the threshold-based models in particular, on the performance and longevity of sensor networks. We will use the same example we used in previous sets of simulations. Although simple, the application is realistic enough to motivate real world scenarios.

In the multi-purpose monitoring application, a network is supposed to monitor three phenomenon in its vicinity. The three phenomena have three different priorities. Traffic phenomenon, A, associated with monitoring task T_A has the highest priority to be monitored, followed by Air Pollution Phenomenon, B. Air Pollution B is in turn associated with task T_B. Finally,

Environment Phenomenon, C, has the least priority and is associated with monitoring task T_C. We do not discuss all the details of the phenomenon here in order not to distract the discussion from the task allocation mechanisms to chemical, environmental, or traffic domain-specific non-pertinent topics. Appendix C.1 lists the settings and parameter values for this set of experiments.

7.2 Experiment Objectives

The experiments of this chapter aim at:

- Testing the effect of applying the FDT Model on a sensor network's task performance dynamics.
- Testing the effects of applying the FDT+B Model on a sensor network's task performance dynamics.
- Identifying what applications could benefit from each of the three models CR, FDT, and FDT+B.

In the next few sections, a few sensor network performance criteria will be examined, and the results of the simulations will be discussed.

7.3 Results

In this section we present the results of the experiments we ran. Each subsection will address one of the metrics associated with the evaluation of a sensor network performance.

7.3.1 Network Lifetime

As longevity of sensor nodes is so important, we compare the network lifetime results from our experiments for networks using CD, FDT, and FDT+B. Figure 7.1 shows the lifetime of the networks at different task demand levels for different models. From figure 7.1, the next few paragraphs will highlight some observations:

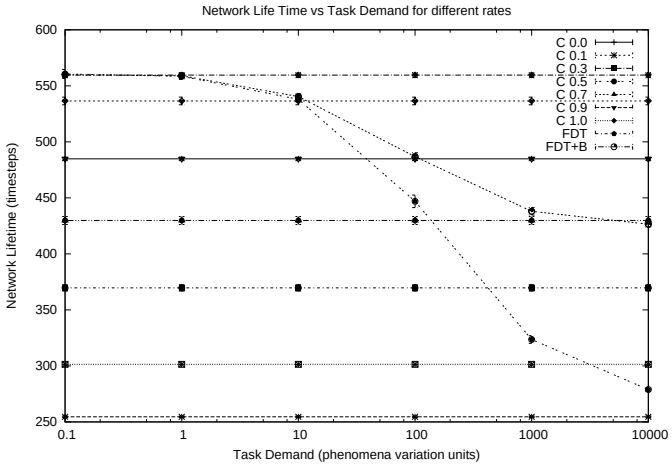

Figure 7.1: Mean Lifetime for sensor networks with different Task Discontinuation Probability

Lifetime Variability: networks that use constant discontinuation probability have constant lifetimes independent of the environmental status represented by the task demand levels in our model. This results in a stream of data or processing rate that is constant. We argue that this is undesirable in many applications. If task demand is high, and the network discontinues with high speed, switching overhead and inferior performance will result. Even if lifetime is elongated, the network fails to match its performance to the needs dictated by the environmental status. If the task demand is high, and the network rarely discontinues performing a task once engaged, lifetime will be shorted unnecessarily, as battery will be depleted quickly performing unneeded task. Alternatively, if another task exert high task demand, low discontinuation probability will result in ignoring the urgently needed task inappropriately. In comparison to constant rate discontinuation, networks that adopt FDT and FDT+B managed to live for very long when task demand was low. However, both networks had short life time under high task demands, favouring fulfilment of the application's requirements to preservation of resources or staying idle.

Imagine our sensor network deployed in a busy city, say London, where a chemical leakage has happened. Sensors that detect an exceptionally high

air pollution levels, may need quickly to take more readings, make calculations and estimations, communicate with base stations and peers to help contain identify and contain the event. In such a situation, priority is alleviating the effects of the disaster rather than preserving network/node resources. However, networks in safe zones may not need to do anything and should preserve their energy to live longer, and may contribute to the management of other future disasters.

Task Demand Effect: Task demand urges higher network performance. Therefore, nodes should increase activity when task demand is high, and use periods of low task demand to stay idle to preserve their resources or perform other tasks with high associated demands. Static discontinuation schemes would not achieve such adjustment. They would either compromise network lifetime all the way through the node's operation, or compromise the node's performance, and consequently that of the network. As we can see from figure 7.1, the lifetime of networks FDT and FDT+B varied according to the task demand. At high demand levels, both networks had considerably shorter lifetime compared to their relatively long lifetime at low task demand levels. Medium-strength task demands achieve medium-length lifetimes. This is a form of adaptivity demonstrated by the use of the simple threshold-based discontinuation models.

FDT vs FDT+B: The previous two paragraphs discussed the differences in lifetime between static discontinuation models and adaptive ones. Now, a comparison of the two adaptive discontinuation models is in order. Both FDT and FDT+B networks behave in a very similar pattern or trend as task demand grows. However, a difference in the lifetime clearly emerges at high demand levels, less so at middle-range task demand levels. Only marginal difference can be observed at low demand levels. This can be attributed to the fact that both networks stay idle, i.e. they start performing tasks with the same rate, and they are very much pushed to quickly quit performing the task they started since the associated demand is low. However, at high task demand levels, the situation is different between the two networks. From algorithm 2, we can see that FDT will decreasingly

disengage from performing tasks as task demand increases. This behaviour persists without any opposing force regardless of a node's resource levels. In contrast, FDT+B will initially increase activity to match the high task demand, but not for all its lifetime. When battery levels start to decrease and battery depletion continues, FDT+B nodes start compromising sometimes on the quality of task performance in favour of elongating their lifetime. This gives them the longer lifetimes observed at high task demand levels.

7.3.2 Mean Coverage

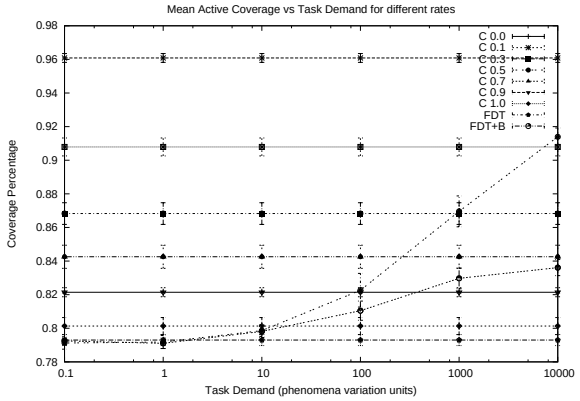

Figure 7.2: Mean Active Coverage for all tasks

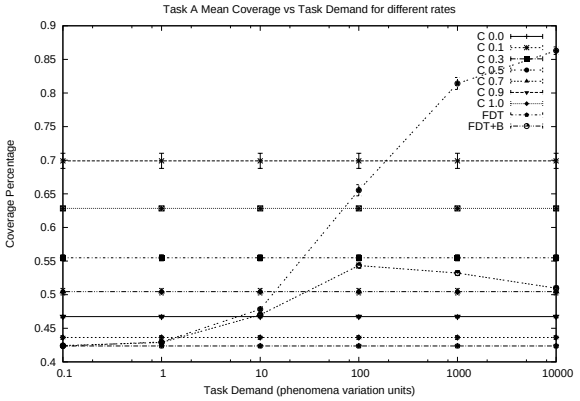

Figure 7.3: Mean Coverage for Task A

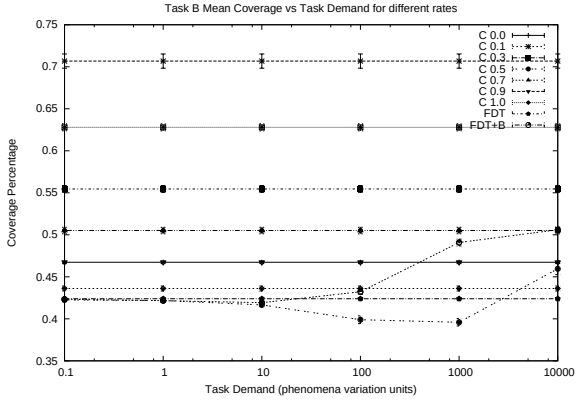

Figure 7.4: Mean Coverage for Task B

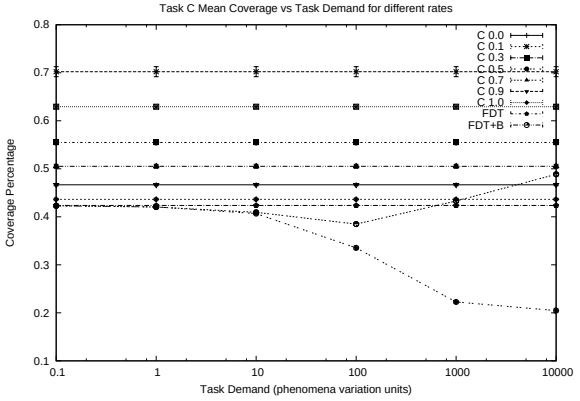

Figure 7.5: Mean Coverage for Task C

Mean Coverage represents the instantaneous coverage on average. Ideally, this should be 100%, however, in many applications, this cannot be

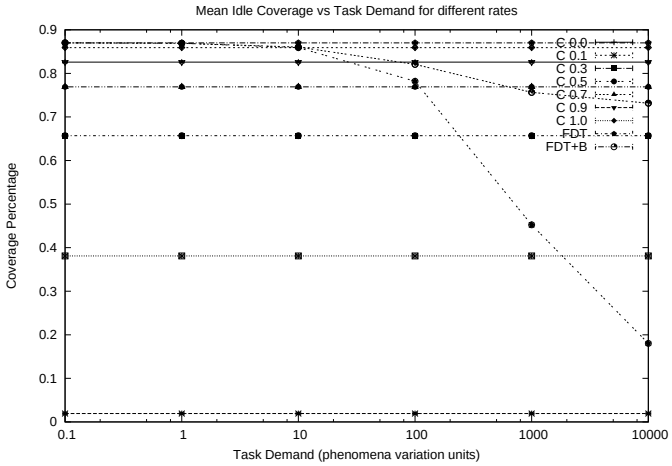

Figure 7.6: Mean Idle Nodes Coverage for all Tasks

achieved because it will be detrimental to the longevity of the network which is also important to the application's purpose and user requirements. From figures 7.2 to 7.6, the following observations can be made:

Active Coverage: for static discontinuation schemes, it is obvious from figure 7.2 that task demand has no effect on its coverage. It performs tasks monotonously at the same rate regardless of the need to perform those tasks. It can be analogous to an employee writing reports when they are not needed, or documenting systems that will never be used. Surely the employer will see this as a waste of the employees energy. In comparison, FDT and FDT+B perform better on average, i.e. yielding higher coverage, as the phenomenon intensity increases. FDT and FDT+B networks outperform most of the others at high task demand levels. Although they provide very low coverage at low demand levels. This is not a deficiency, but rather a positive behaviour that will result in resource preservation until these resources are most needed. In this context, adaptive models have outperformed static models at all task demand levels. The superior performance of adaptive models at high task demand levels indicate a rapid resource consumption, which agrees with the requirements of the family of applications of sensor networks that we are addressing in this dissertation. In the case of a hazardous chemical leakage, for instance, then you want to

contain the disaster regardless of the longevity. This can be seen elsewhere with bees that all perform suicidal attacks against enemies when their hives are infiltrated or greatly endangered.

Task A Coverage: T_A's coverage follow a trend that is very similar to that of the active nodes (figure 7.3). It is simple to see why this is the case. Networks with constant discontinuation probability give the same coverage no matter what task demand there is. This is not optimal for Task A because when its associated demand is high, the constant discontinuation probability networks may produce inferior task performance. In the same manner, when Task A's demand is low, these networks may waste energy performing unnecessary work. FDT and FDT+B networks, unlike CD networks, change their task performance time/frequency according to the task demand detected by the sensor node. This allows preserving energy when work is not needed, and producing superior coverage when it is most needed. The difference in coverage between FDT and FDT+B is a result of the extra dampening effect of the inclusion of battery levels in making the task discontinuation decisions. When battery levels are low, the discontinuation decision is probabilistically dominated by this factor, and task discontinuation is increased accordingly.

Task B Coverage: In CD networks, Task B's curves in figure 7.4 show identical trends to those of Task A, same coverage regardless of task demand intensity. However, if we look at the FDT and FDT+B networks, we see a fairly unexpected behaviour. At low task demand levels, both adaptive networks give fairly low coverage, which is normal and expected. However, when task demand is intense, a proportional growth in coverage is expected, yet little growth is observed. This is because the threshold of Task B is much higher than that of Task A, resulting in most nodes opting for performing Task A, with least threshold. This results in less effect on the number of nodes performing Task B. If Task A's demand was low, and that of B was high, a clear growth would be observed . We did not include the graphs of these results here for reasons of space and time. The difference between FDT and FDT+B is again a result of the additional battery compo-

nent effect in the FDT+B network.

Task *C* Coverage: analysis of the performance of CD networks with respect to task *C* will not be included here as it is identical to those of task *A* and *B*. However, surprisingly, task *C*'s coverage is reduced when task demand is increased, see figure 7.5. The correlation here is between the rise in both task *A*'s demand, and that of task *C*. As task *A* has higher priority, nodes ignore task *C*, even if it is highly demanded, and go on performing task *A*. Of course FDT+B nodes stand even less chance to perform task *C* because their decreasing battery levels decrease their tendencies to perform any task at all, and task *C* in particular because it has the highest threshold value. Again if task *C*'s demand intensity was high, and medium or low intensity of demand were present for task *A* and *B*, task *C* will be performed by a larger number of nodes.

Idle Coverage: Idle nodes do not perform coverage tasks, but calculating their coverage and treating them as nodes performing a costless task helps obtaining an insight into the future robustness and amount of redundancy the network provides. For example, in figure 7.6, Idle nodes made coverage of around 90% of the terrain at low demand levels, meaning that if every active node fails, there would be enough nodes to replace them all. This indicates a network that can achieve good longevity. However, this was reduced to 75% in the case of FDT+B, and to 10% with FDT networks at high task demand levels. FDT networks rush into performing the tasks continuously when the demand is high disregarding any resources considerations, while FDT+B networks continue performing the tasks incessantly as long as the resource levels allow it. Otherwise, they take a safer approach by performing the tasks less frequently than dictated by the task demand.

7.3.3 Total Coverage

Generally, at low task demand, networks are supposed to maximise total performance, $F_a(t)$, while at high task demand, networks should be maximising average performance, $f_a(t)$. This is what we see happening for FDT

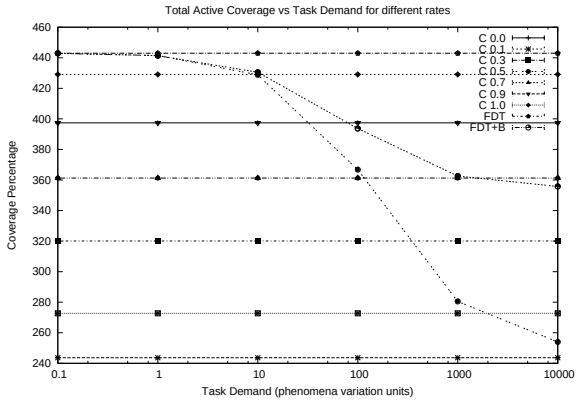

Figure 7.7: Total Active Coverage for all tasks

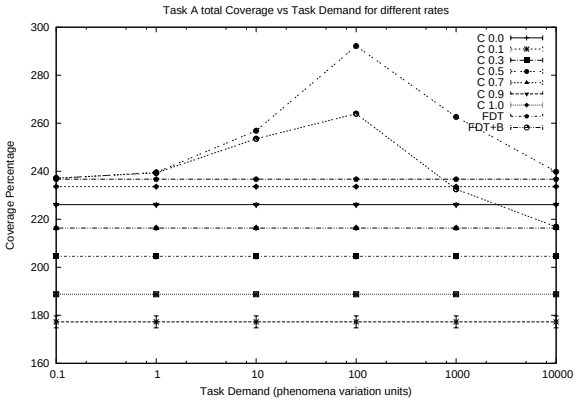

Figure 7.8: Total Coverage for Task A

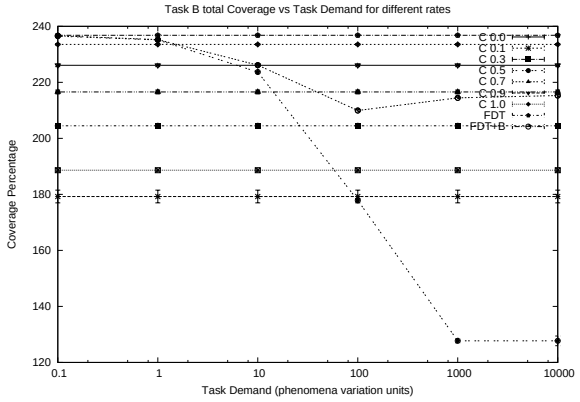

Figure 7.9: Total Coverage for Task B

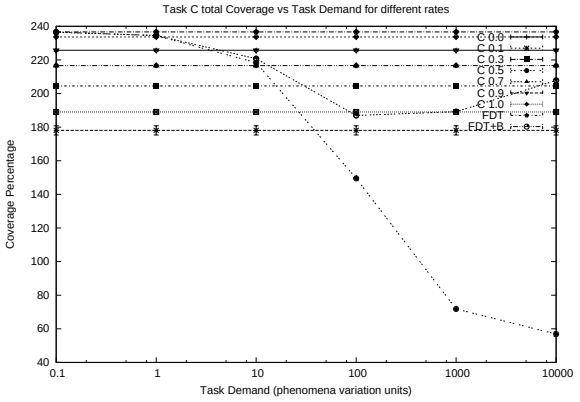

Figure 7.10: Total Coverage for Task C

and FDT+B in figure 7.7. Of course, CD networks had constant total coverage as they perform the exact amount of work regardless of task demand. We can notice the same pattern for total coverage for active, task *B*, and task *C* nodes seen in figures 7.7, 7.9, and 7.10 respectively.

However, we notice in figure 7.8 that task *A*'s total coverage rose at medium range demand levels. This is because task *A*'s threshold was the lowest, i.e. it had the highest priority to be performed by the network. At some point, the task demand caused an optimal ratio of active nodes to idle ones to occur. At very high task demand intensities, the number of active nodes increases with only little increase in network coverage, if at all. In the multi-covered regions, some of the network energy was wasted. The extra active nodes can be useful in high node-loss applications, e.g. disaster

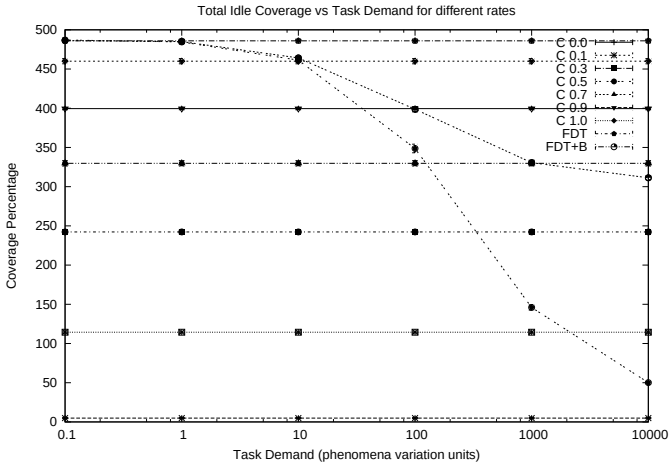

Figure 7.11: Total Idle Nodes Coverage for all Tasks

zone monitoring applications. At medium range task demand, the number of active nodes achieves high coverage with as few nodes as possible, and this is where this surge appears in figure 7.8.

7.3.4 Dead and Idle Nodes

From the dead nodes graph, figure 7.12, we can see that the two extreme cases of discontinuation (performing tasks incessantly, and discontinuing a task once performing it started) give the bounds of possible values of life times, i.e. longest and shortest longevity.

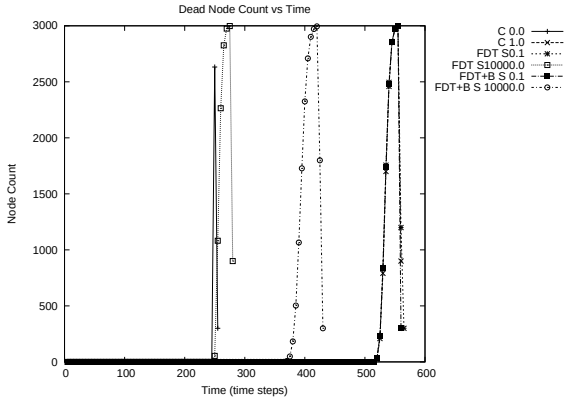

Figure 7.12: Dead Node Count vs Time

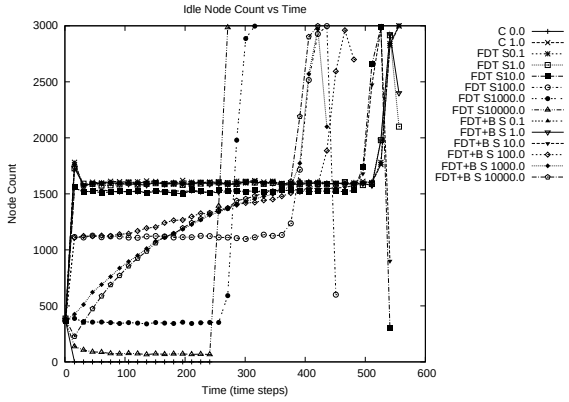

Figure 7.13: Idle Node Count vs Time

It is interesting to see that FDT and FDT+B behave as different CD

networks with respect to the network lifetime at different task demands. Which CD network they adopt or behave like is based on the status of the many parameters of the system, be it environment, internal status, or task importance. For example, in figure 7.12, FDT networks perform as if they were a CD network with (C = 0.0) at very high task demand. At very low task demands, discontinuation probability is very high, and so FDT and FDT+B both perform very similar to CD networks with (C = 1.0). At very high task demand levels, FDT+B network does not die as fast as FDT, but lives longer, and performs slightly less. It does a tradeoff between performance and resource consumption.

Figure 7.13 shows the count of idle nodes for different networks. From the graph, we can see that FDT+B have a variable number of idle nodes that rises slowly to reach the maximum with time. This is because at the beginning of the experiment, batteries were full, and so the FDT+B network did not have a reason to decrease the number of active nodes when the task demand is extremely high. Slowly, batteries are depleted and the FDT+B network starts reducing the number of active nodes, even though demand is high, to preserve energy. This scenario does not occur with the FDT network, which does not consider energy variations. This explains the level curve for the FDT network. The FDT network, however, exposes an adaptive behaviour by varying the number of idle nodes according to the task demand levels.

7.3.5 Prioritisation

Making sensible and desirable prioritisation is a very important aspect in action selection algorithms. In our scenario there are two required levels of prioritisation. The first is *task prioritisation* and the second is *cost prioritisation*. This section will discuss each level separately.

Task Prioritisation: Task prioritisation refers to the higher tendency of a node to perform a certain task in comparison with other tasks. In the FDT and FDT+B networks, a node determines which task has higher priority over another by considering task thresholds, in addition to the task-

associated demands. From figure 7.2, we can see that CD networks do not perform any prioritisation in terms of task importance. CD networks perform tasks with a constant rate independent of their importance. From figures 7.2 to 7.5, FDT and FDT+B networks coverage illustrate a correlation between coverage and the combination of thresholds and task demands. For these networks, when task demand is low, all tasks are treated equally, with slightly higher tendency to perform low threshold tasks. At high task demand levels, tasks with high importance, i.e. low thresholds, hog more resources than lower priority tasks. If high priority tasks has low demand, and low priority tasks has high demand, then tasks with low priority will be performed more frequently, without ignoring high priority tasks completely. At mid-range task demands, the network finds a rate at which a compromise is made between the importance of a task, and how much resources it needs in order to perform it.

Cost Prioritisation: The term refers to the act of balancing the importance of performing a task against the amount of resources that are going to be consumed if this task is performed. FDT+B is the only network that considers resources when making task discontinuation decisions in this chapter. In figures 7.2 and 7.3, FDT+B network did not produce a mean coverage as high as that of the FDT network. This is because after sometime, nodes' batteries were getting depleted, and so FDT+B reduced its frequency of performing the task increasing the discontinuation frequency, while FDT network ignored such a warning and so gave a higher coverage on average. However, we can see that FDT+B gave a better total performance because it used its resources more wisely.

7.3.6 Conclusions

From the results of this chapter, it seems that the FDT model is mostly suited to applications that monitor short-lived phenomena like earthquakes, structural collapses, or traffic congestions. While FDT+B could be more suited to applications with unknown, unpredictable, or relatively long-lasting phenomena, such as forest fires, floods, draughts, environmental

conditions, etc. The FDT+B model could work better in applications where longevity is more important than high resolution monitoring.

It can be concluded that using adaptive discontinuation models is beneficial in sensor networks where node/network resources are scarce, or where application requirements differ according to system status. FDT and FDT+B models both improve the longevity of sensor networks conserving energy when possible. Both FDT and FDT+B models support task prioritisation.

7.4 Summary

In this chapter, we presented the results of simulations run to examine the effects of using an adaptive behaviour for task discontinuation in sensor networks. We also discussed the different aspects, indications, and potential conclusions of these results. Threshold-based mechanisms constitute a simple, yet powerful mechanism to introduce adaptivity at task discontinuation and action selection processes. In previous chapters, we discussed how threshold-based mechanisms can be employed for action selection, and in this chapter we utilised them to control task discontinuation. In the next chapter, we will examine using the same models to control sensing processes within a sensor node, and after that we will address also the control of communication processes using the same family of models.

Chapter 8

Sampling

In previous chapters, we applied the threshold-based adaptive model of [22] to action selection and task discontinuation decision making processes of a sensor node. In this chapter, we apply the same model in order to control the sampling processes within a sensor node.

8.1 Non-Adaptive Sampling/Sensing Schemes

A preset interval can be used by a node to take readings in relation to a phenomenon. A semi-continuous polling-like sampling scheme can also be used. Both, and similar, schemes do not take into consideration the need to sample the environment, or rather the variability of the need to sample the environment. For example, a network that monitors air pollution might not need to sample pollution levels that are stable. In such case, an incessant sampling scheme will perform redundant, unnecessary, and possibly very costly sampling. Most systems that adopt static, periodic, or constant sampling schemes disregard the cost involved in these processes, either because such considerations are contradictory to the application requirements (for example in hard real-time applications), or because resources needed to perform sampling are abundant [50] and [49], [130]. For example, sensors that are deployed in buildings may be able to use the mains power.

8.2 The Constant Sampling/Sensing Probability Model (CS)

Previous chapters discussed the use of threshold-based models to help individual nodes in a sensor network make decisions whether to engage in performing a task if idle, or discontinue performing a task if active. However, we have not yet addressed the mechanism used by individual nodes to make decisions in relation to sampling the environment. In the model described in [31], sensor readings are taken every 0.5 seconds, and in the climate monitoring application in [48], the environment is sampled every 5 minutes. Nodes in such networks take readings on a regular basis, or based on a constant probability p. The value of the probability is preset before the system is employed, i.e. at design time. The authors of these models did not explicitly explain why they used these values for sensing/sampling frequency or probability. However, it is generally based on the required time resolution of samplings that is deemed satisfactory to the application in question. For example, in the case of [31], temperature monitoring application was devised to illustrate a calibration mechanism. In real applications, a more complex pattern may be needed. Applications could take the maximum required sampling rate to guarantee satisfactory overall results. However, this means wasting some of the resources as some collected data may be uninteresting. We find that an adaptive model is used in nature in many phenomena. Take for example the human brain, it needs to rest, preserve energy, recover. Therefore, they work minimally, for example, when one is asleep. Another observation is that different human sensory capabilities work at different situations. These, and similar, observations point to a conclusion that it is only logical to have an adaptive sensing. This is even more so when sensing is an expensive activity. Unnecessary sampling could result in battery depletion, degradation in other task performances, memory consumption, or waste of computational power.

In our *Weather Monitoring Task*, setting the value of the sensing probability p to a relatively low value (say 1 reading a day) may result in too coarse-grain data. In most such applications, this type of reading frequency is undesirable. However, imagine the situation in the poles, where changes

in the climate are very slow, and so very low sensing probabilities may be acceptable. On the contrary, if p is high (p approaches 1), nodes will log, process, and consume energy on dealing with these values for no, or so little, advantage. Actually this may result in faster depletion of battery power, or fast wear of sensing hardware. Frequent turning on and off of sensors may introduce overheads, yet too few readings could result in inferior performance i.e. low network coverage. Medium values for p will give less than optimum results at either extremely high or extremely low climate/weather variation speeds (very slow and very rapid). In addition, such values will not be optimal for continuous spectrum variation levels. For example, wind speed variation spans a wide range of values. In many climates, it may not be possible to find a sampling rate that matches that of the wind speed variation for long periods of time.

The mathematical representation of the arguments presented in the previous paragraph is now due. Assuming T_{max} is the maximum interval between two samples taken by a node, then the duration, $T(S)$, after which the next sampling event will occur can be calculated using the equation:

$$T(S) = T_{max} \times \Phi(S) \tag{8.1}$$

where $T(S)$ is the time after which a reading for the phenomenon in question will be taken, T_{max} is the maximum time interval between two consecutive sampling events with respect to a phenomenon, and finally $\Phi(S)$ is the probability a sampling event with respect to a phenomenon will take place. $\Phi(S)$ can be calculated using different equations according to application- and domain-specific knowledge. However, in many sensor network applications, an equation with the following characteristics will be desirable:

$$\lim_{S \to \infty} \Phi(S) = 1 \tag{8.2}$$

$$\lim_{S \to 0} \Phi(S) = 0 \tag{8.3}$$

where S is the stimulus of the phenomenon in question.

Sensor networks are often employed in unpredictable environments. For example, a sensor network to monitor traffic in an urban environment

might detect very low variations in the traffic most of the time due to normal hours conditions. However, in rush hours, the sensor nodes in the surrounding area will have massive traffic changes to report, and hence sampling rate may need to be increased significantly and temporarily until the rush hour ends. In such a scenario, high values of sampling probability in normal low-traffic conditions might be inappropriate, while a low sampling probability may cause inferior sampling resolution when traffic is heavy for an extended period of time. In the next few sections, we will extend the threshold-based model provided by [24] and [25], and [181] to control phenomenon sampling within a sensor network node. Later, in the next chapter, simulations and findings will be presented.

8.3 Fixed Stimulus-Based Sampling Threshold Model

The way many living systems survived for millions of years is said to be their ability to adapt to the environment [197]. We see the same concept applicable to sensor networks. If sensor networks can adapt to the stress the environment exerts on them, they may stand better chance to survive for longer periods of time, as well as provide better task performances. This adaptability is crucial in many behavioural aspects of the network including the environment sampling decisions. Individual nodes may benefit from being able to make on-the-fly decisions on whether to sample the environment or not with respect to a phenomenon. This dynamic decision making, driven by environmental conditions, improves network performance and longevity, compared to those of networks that adopt static sampling scheme.

To make the idea clearer, let us look at an example scenario. In the Weather Monitoring task in our multi-purpose sensor network application, if an individual node observes a great temperature increase in its vicinity, it may perform high rate sampling activity to record these variations in the environment and report them back to a base station. While it is sending readings to a base station about the environment, a traffic jam develops, and the temperature rise has, fortunately, stopped. It might then decrease its sampling rate with respect to the Environment Monitoring task because there may be no need to sample this phenomenon anymore. This will

allow other phenomena, such as Traffic, to be sampled and processed instead of the Environment or phenomena, resulting in a better overall task performance. Or may be the hypothetical traffic jam phenomenon did not happen. In this case, the node's decision to reduce sampling rate of the Environment-associated task will result in conserving energy and possibly will lead to a longer network lifetime.

When stimulus to perform a task is high, the sampling probability for that task-related phenomena is high. On the contrary, if an individual is performing a task, while the stimulus for that task is low, it is likely that the individual should sample the environment conservatively as there may be very little gain from performing unnecessary sampling. This can be modeled mathematically, drawing from the response threshold model in chapter 4, as follows:

$$\Phi(S) = \frac{S^n}{S^n + \theta^n} \tag{8.4}$$

Where $\Phi(\theta, S)$ is the probability a node will sample the environment with respect to a phenomenon associated with a task with threshold θ when the task-associated stimulus is S. Note that $\Phi(\theta, S)$ approaches 1 as S approaches ∞ (there is very high demand for this task, and so urgent need to sample the associated phenomenon), and approaches 0 as S approaches 0 (there is little demand for this task, and consequently little need to sample the associated phenomenon). The following equations represent this mathematically:

$$\lim_{S \to \infty} \Phi(S) = \lim_{S \to \infty} \frac{S^n}{S^n + \theta^n} = 1 \tag{8.5}$$

$$\lim_{S \to 0} \Phi(S) = \lim_{S \to 0} \frac{S^n}{S^n + \theta^n} = 0 \tag{8.6}$$

In next chapter, we will conduct experiments to validate the model, and investigate the effects it has on the dynamics of a sensor network behaviour. Next, we will look at incorporating resource levels availability or cost in making sampling decisions.

8.4 Adaptive Resource-Based Sampling Threshold Model

The previous section discussed the FST extension of the FRT model in [22]. It used the task demand levels to decide whether to sample a phenomenon, or keep the sensing equipment off. In this section, we extend this model even further to include battery levels in environment sampling decisions. To reiterate we mentioned in previous chapters, battery is only an example, though very important one, of a resource that can be incorporated in the decision making process, and other resources could have been used as well. Sometimes, task demands are high, and so more data is needed about the environment and the phenomenon of interest, but resources required to perform the task and sample the environment for the associated phenomena are running low. In such circumstances, nodes may need to make compromises on the quality of task performance and accuracy of data depending instead on other data sources such as historical data or environment models in favour of longer task performance. Accounting for battery levels in environment sampling decisions is only an example of incorporating resource levels available to nodes in making task-associated decisions. Battery levels are a candid representation of scarce resources in many sensor networks, and hence can be used without loss of generality.

Algorithm 3 represents how a node would use battery levels and task demands to make decisions on phenomenon sampling.

1 $\Phi(S, \Phi) = \frac{S^n}{S^n + \theta^n}$
2 $\Phi(B) = \frac{B^m}{B_0^m}$
3 $\Phi(S, \theta, B) = \Phi(B) \times \Phi(S, \theta) = \frac{S^n}{S^n + \theta^n} \times \frac{B^m}{B_0^m}$
// where $\Phi(S)$ is the probability a node will sample a
// phenomena when task demand is S disregarding the
// battery level B (essentially equation 8.4) $\Phi(B)$ is
// the probability a node will sample a phenomenon when
// the energy level on this node is B disregarding task
// demand S

Algorithm 3: FST+B Model algorithm to decide $\Phi(S, B)$

This algorithm results in the following mathematical equations representing the sampling probability trends:

$$\lim_{S\to\infty,B\to 0} \Phi(S,B) \;=\; \lim_{S\to 0,B\to B_0} \Phi(S,B) \;=\; \lim_{S\to 0,B\to 0} \Phi(S,B) \;=\; 0 \tag{8.7}$$

$$\lim_{S\to\infty,B\to B_0} \Phi(S,B) \;=\; 1 \tag{8.8}$$

These equations represent the following generalised trend of node behaviour: If task demand is high and there is enough battery power, then the node will most likely sample the environment. However, if either or both battery power and task demand are low, then most likely the node will not sample the environment. We will discuss the effect of such behaviour on task performance in the next chapter.

8.5 summary

In this chapter we introduced an extension to the FRT model used originally in [22] to make action selection decisions. Our extension uses the model to make sampling decisions. This extension can be seen as an additional layer of adaptivity which may improve the performance and longevity of sensor networks. Also, we added a resource level modulation component to the model to account for resource availability and task performance cost. In the next chapter we will present the simulations that were conducted to study and investigate this extension model, together with the results. We also discuss some conclusions that can be drawn from those results.

Chapter 9

Sampling Simulations

9.1 Introduction

In this chapter, we will use an experimental application scenario to examine the sampling decision making within a sensor network simulation. Next section will introduce the experiments scenario, and a discussion of the context of conducting these experiments. This will be followed by the results in the form of several graphs and comments on their indications. Finally, some conclusions will be presented regarding the use of the FST and FST+B models in making sampling decisions.

9.2 Experiment Scenario

The goal of these experiments is to examine the effects of the use of adaptive sampling models in general, and the threshold-based models in particular, on the performance and longevity of sensor networks. Again, we will use our multi-purpose sensor network application to run the simulations detailed in chapter 5. In this application, a network aims at monitoring three phenomena The three phenomena has three different priorities. Phenomenon A, Traffic Monitoring, associated with monitoring task T_A has the highest priority to be monitored, followed by phenomenon B, Air Pollution Monitoring. Phenomenon B is in turn associated with task T_B. Finally, phenomenon C, Environment Monitoring, has the least priority and is as-

sociated with monitoring task T_C. We did not delve into the details of the Monitoring Tasks in order not to distract the discussion from the task allocation mechanisms to chemical, environmental, or any other non-pertinent details. Also there were concerns over the space available and the scope of the research in this dissertation. Appendix D.1 lists the settings and parameter values for this set of experiments.

9.3 Experiment Objectives

The experiments of this chapter aim at:

- Testing the effect of applying the FST Model on the dynamics of task performance in sensor networks.
- Testing the effects of applying the FST+B Model on on the dynamics of task performance in sensor networks.
- Identifying what applications could benefit from each of the three models CS, FST, and FST+B.

9.4 Results

We conducted 10 runs for each of the network models (CS, FST, and FST+B) with each model settings given in appendix D.1. We collected data from the experiments, and the next few sections will present and discuss the results.

9.4.1 Network Lifetime

We can see in figure 9.1 that the FST, FST+B, and C 0.1 networks live shorter life at low task demand levels, while the situation is reversed at middle-range task demand levels. At high demand levels, differences between networks lifetimes disappear.

The pattern we see in figure 9.1 is probably surprising at first glance. However, understanding the way the nodes interpret the concept of a stimulus, or task demand, can give us an insight into why this pattern occurs.

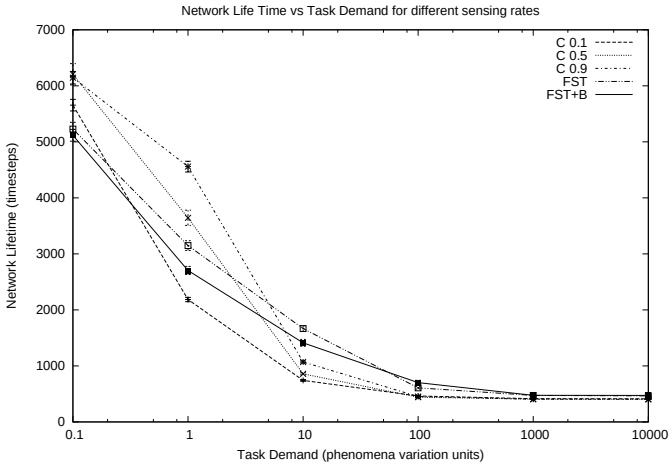

Figure 9.1: Task demand vs lifetime for different networks

At first glance, you might think that if a network does less sampling, based on its assessment of an environment that exerts minute task demand on it, then it will live longer. This is probably true for certain environment behaviours and models. For example, our network generally is stimulated by the variation in a phenomenon, which means less sampling will lead to loss of some phenomenon details. These details may have great consequences on the network dynamics. For example, if air pollution changes between different sampling events, then the amount of that variation represents the stimulus that a node detects. In a network that has a the following readings sequence associated with the pollution (notice the pollution is fluctuating):

$$\overrightarrow{S} = (5, 6, 5, 4, 5)$$

a constant variation of 1 unit is detected by the node that obtains these readings. This can be calculated by taking the absolute value of the subtraction of each reading from its predecessor or successor reading. Now, consider the following series of readings, which represents a different environment model where the pollution is steadily increasing:

$$\overrightarrow{S} = (5, 6, 7, 8, 9)$$

Again, if these readings were obtained by a node, then the average task demand will appear to be 1 unit. We can see that both set of readings were perceived similarly even though they are quite different. Now imagine that a node took only the first and the last readings of each of the readings sequences. The first series will give a variation of 0, meaning there is no task demand at all. The second series will produce a four times more stimulus magnitude at a value of 4 stimulus units. Here the difference between the two reading sets is very significant. The second series is more like what our networks in this chapter experience, hence the way the data is interpreted in the graphs.

With the previous analysis in mind, the following can be noted about figure 9.1:

At low task demand: networks that did frequent sampling, namely C 0.9, C 0.5, and C0.1 networks, lived longer. Whereas those networks that did less sampling, namely the adaptive networks FST and FST+B, had short lifetimes. This is, following our discussion above, because the environment has a constant rate of increase of each monitored phenomenon. Those networks that did scarcely sample the environment perceived higher stimulus than those who sampled it more frequently. High perceived stimulus lead to more task performance which in turn increases battery consumption leading finally to a short lifetime. This could have been avoided if the stimulus was divided (normalised) by the time elapsed between the sampling events, which we did not do in our simulations.

At high task demand: high demand is perceived the same way by all networks, hence all networks live for similar lengths of time with only marginal differences. Because the task demand is very high, and perceived as so by the networks, high task performance rate results, and the effect that we see at low task demand levels becomes too small to influence the life time of different nodes. This can be demonstrated by the following example. Assume a node that takes the following series of readings:

$$\overrightarrow{S} = (50000, 60000, 70000, 80000, 90000)$$

Here, the node's probability of performing the associated task, given its task threshold is 1000, can be calculated by:

$$\Phi(\theta, S) = S(1000, 10000) = 10000/(1000 + 10000) \simeq 0.91$$

Another node that has much lower reading rate could read the following sequence:

$$\overrightarrow{S} = (50000, 150000, 250000, 350000, 450000)$$

and from this we can calculate the probability it will engage in performing a task using the following equation, given it has the same associated task threshold:

$$\Phi(\theta, S) = S(1000, 150000) = 100000/(1000 + 100000) \simeq 0.99$$

which does not differ significantly from the other node that did 10 times more sampling. This is of course a big contrast to the situation at low task demand levels.

At mid-range task demand: slowly, with task demand growth, the low task demand pattern is reversed. There is a critical point in the mid-range task demand levels where the frequency of sampling becomes insignificant with respect to the task engagement decision making and hence the cross in the mid-range stimulus in figure 9.1.

9.4.2 Mean Coverage

In this section, we discuss the implications of the Sampling Model on the task performance, or in particular, network coverage within the different sensor networks of our simulations. From fig 9.2 to 9.6 we can make the following observations:

For C 0.1 **network:** Consistently over all graphs and at all task demand levels, this network performs high on average. Remember that it lived the least as well. This is a result of rare sampling, resulting in high perceived

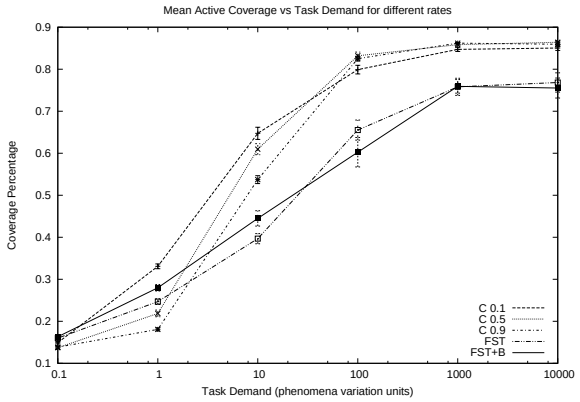

Figure 9.2: Task demand vs Mean Active Coverage for different networks

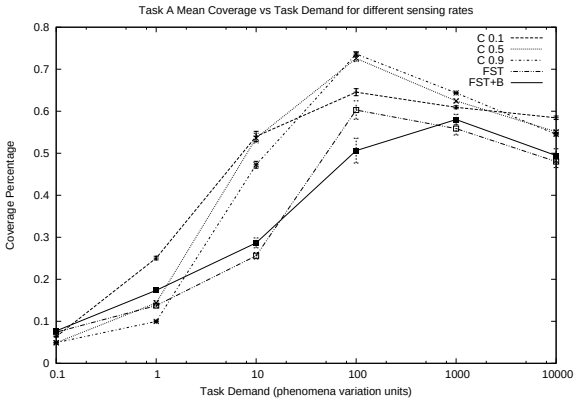

Figure 9.3: Task demand vs Mean Task A Coverage for different networks

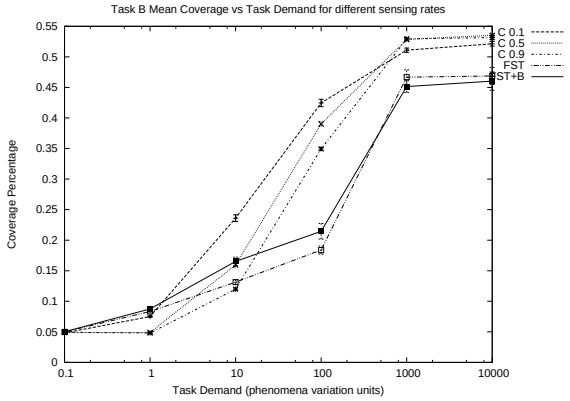

Figure 9.4: Task demand vs Mean Task B Coverage for different networks

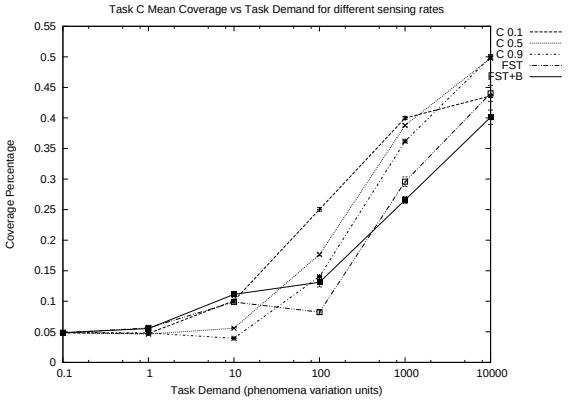

Figure 9.5: Task demand vs Mean Task C Coverage for different networks

stimulus. We can represent this using a mathematical equation. Assuming Δt is the time between two readings, $S(t)$ is the stimulus from the environment at time t and C_s is the rate at which stimulus in the environment changes, we can represent the perceived stimulus, S_{total}, just after a sampling event by:

$$S_{total}(\Delta t) = \sum_{i=0}^{i=\Delta t} S(i) = \Delta t \times C_s \tag{9.1}$$

This equation is least when Δt is least because C_s is a constant. Mathematically, this can be represented as follows:

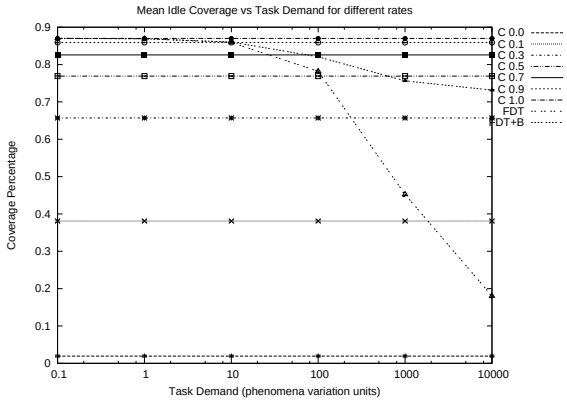

Figure 9.6: Task demand vs Mean Idle Coverage for different networks

$$\lim_{\Delta t \to 0} S_{total}(\Delta t) = 0$$

In the C 0.1 network, Δt is high and so the perceived stimulus, S_{total}, is also high. This, in turn, induces this network to perform, in average, more forcefully than others.

For FST and FST+B networks: low sampling rate at low task demand levels result in high perceived stimulus, which in turn results in high task performance in average. As task demand levels increase, more sampling happens, resulting in less perceived task demand in comparison to what other networks perceive. This in turn results in less average performance.

Indeterministic results: We discussed networks that used the C 0.1, FST, and FST+B models because they have a clear trend. Some trends can be observed from the graphs for other networks, such as those of the C 0.9 network which show clear low average performance at low demand levels, and higher performance, in comparison to other networks, at high demand levels. However, these trends are less significant, less obvious, and may incur many questions that we have no space to answer in this thesis. However, they are connected to the complexity of the model, and the unclear trends are not adverse in anyway, so we can safely say they insignificant.

In fig 9.3, Task A's performance display a peak at some point. This is

an optimal point of performance where the stimulus exerted by the environment motivates the network to activate a number of nodes that together achieve a very high coverage for Task *A* (the highest priority task), and not so high for other tasks, namely Task *B* and Task *C*. When stimulus gets very high for all tasks, the differences between their thresholds become negligible, and so Task *A* loses its high priority status, and so its average coverage descends as in the figure.

In figures 9.4 and 9.5, both tasks, *B* and *C*, exhibit a continuously ascending coverage with the increase of stimulus magnitude. This is because continuous increase in stimulus steadily increases the number of active nodes for these tasks, until they even get to be as high-priority as Task *A*. Notice the difference between the trend of the average task performance for these tasks and for task *A*.

The points made regarding the average coverage for active nodes are identically observed on the graphs of individual task performances. This is an important sign of consistency in the effects of the model on all tasks.

Mean idle nodes coverage exhibits a similar trend to that of the FRT and FRT+B networks from chapter 5. This is due to the fact that the FRT model was used throughout this chapter to make action selection decisions (note that FST and FST+B are used to make sampling decisions).

9.4.3 Total Coverage

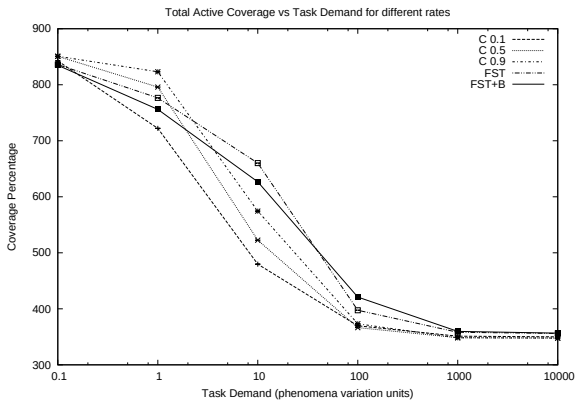

Figure 9.7: Task demand vs Total Active Coverage for different networks

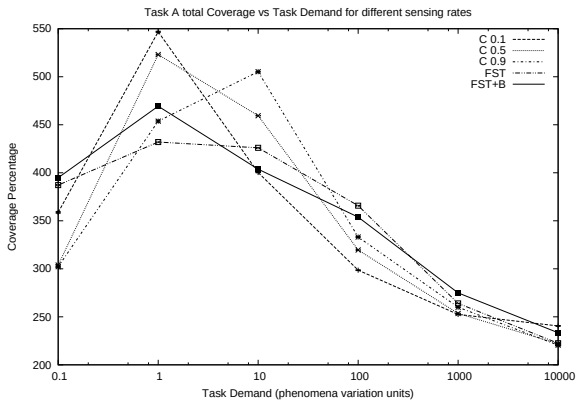

Figure 9.8: Task demand vs Total Task A Coverage for different networks

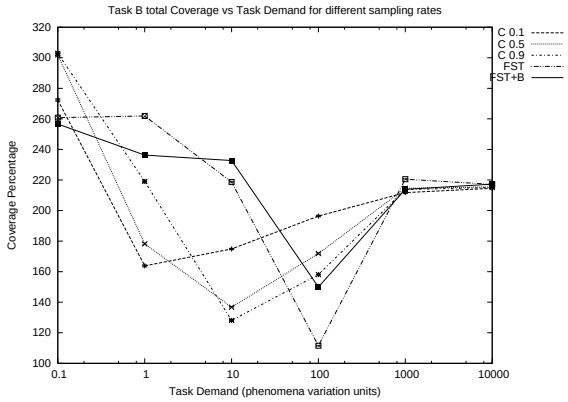

Figure 9.9: Task demand vs Total Task B Coverage for different networks

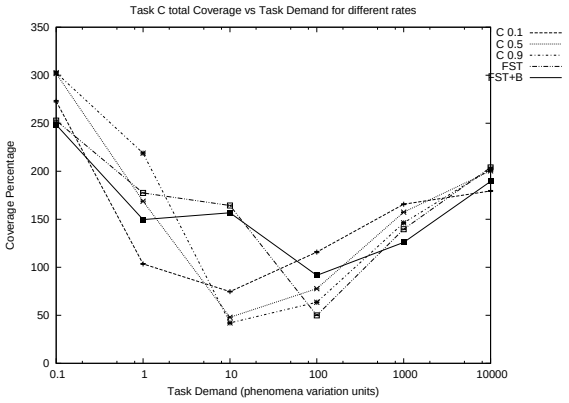

Figure 9.10: Task demand vs Total Task C Coverage for different networks

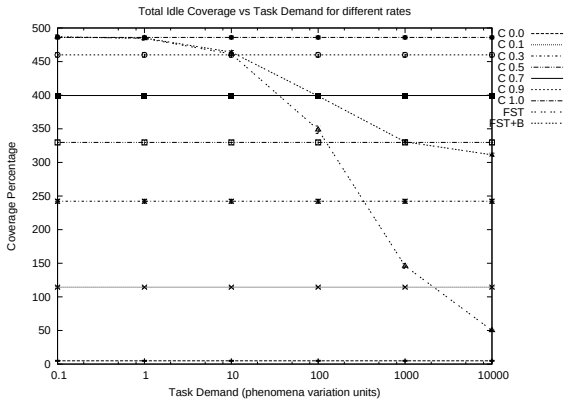

Figure 9.11: Task demand vs Total Idle Coverage for different networks

While mean coverage had fairly clear trends and differences between networks at high demand levels, total coverage statistics seem to show close resemblance between the different network curves. This is observed in all graphs of this section, figures 9.7 to 9.10.

For total coverage, we see a trend appearing in the range from low to medium magnitudes of task demand. We can see a crossover point, where networks that produced high total coverage at low task demand levels, display lower levels of coverage at medium task demand levels, and vice versa. This follows from the same argument presented in section 9.4.2. Highly sporadic sampling events within a node result in its perception of

high task demand, which in turn results in high average coverage, and a lower total coverage. The reverse is also true. High sampling frequency by nodes results in low perception of task demand, which in turn leads to low average coverage, but high total one. This crossover is most obvious for the FST and FST+B networks because they use adaptive schemes.

Network C 0.1 does not follow the trend highlighted above because it has always high perception of stimulus because it very sparingly samples the environment.

The C 0.9 network does exhibit a crossover, but for different reasons from those of the FST and FST+B networks. This network perceives low task demand at low task demand levels because it samples the environment frequently. However, it detects high demand at high stimulus magnitudes despite its frequent sampling. This is the reason of the crossover, it converges to behave very much like FST and FST+B when task demand is high.

Finally, idle node's count in figure 9.11 follows the same trend observed in average idle node coverage in figure 9.6. All constant rate sampling networks maintain a constant number of idle node's coverage. Whereas FST and FST+B, the adaptive networks, exhibit a descending idle node coverage as task demand increases.

9.4.4 Idle Nodes

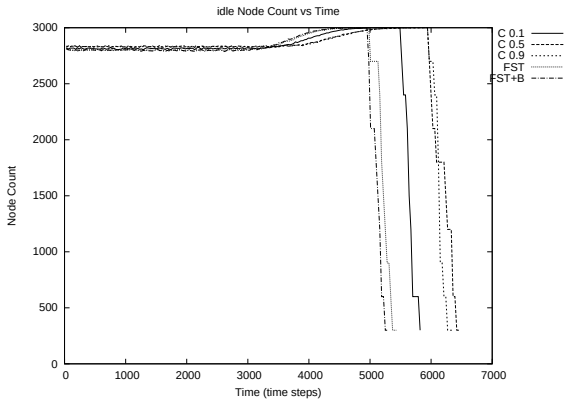

Figure 9.12: Time vs Idle Nodes Count for different networks when $S = 0.1$

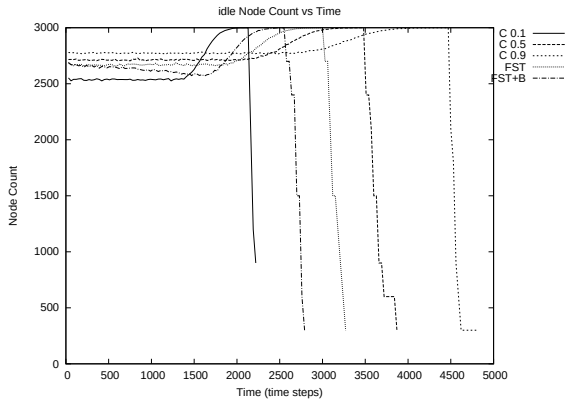

Figure 9.13: Time vs Idle Nodes Count for different networks when $S = 1$

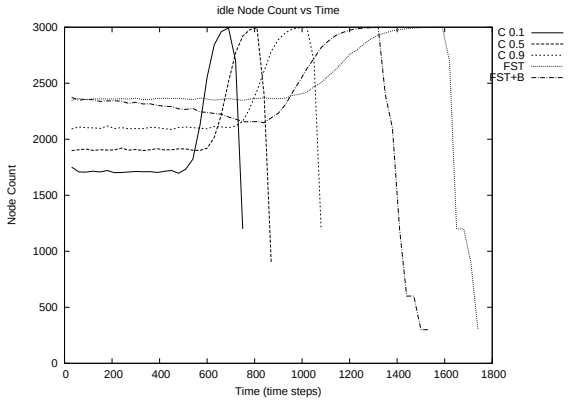

Figure 9.14: Time vs Idle Nodes Count for different networks when $S = 10$

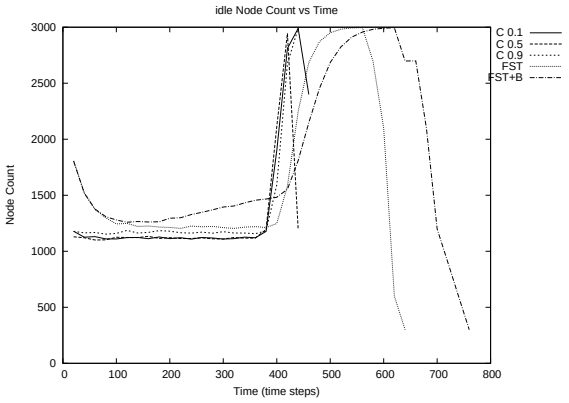

Figure 9.15: Time vs Idle Nodes Count for different networks when $S = 100$

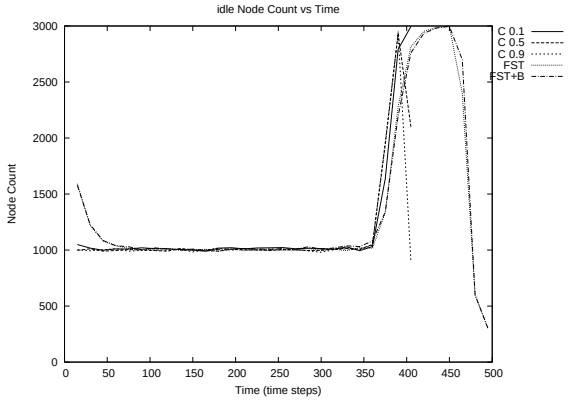

Figure 9.16: Time vs Idle Nodes Count for different networks when $S = 1000$

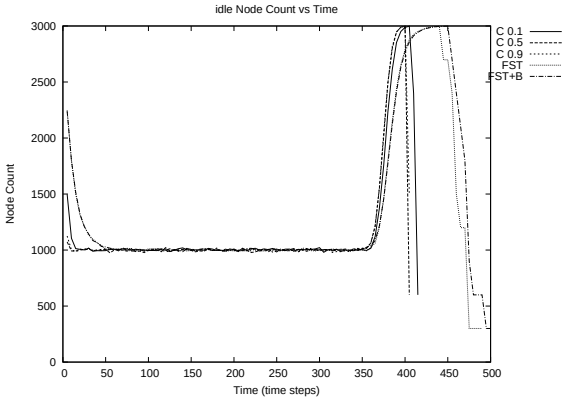

Figure 9.17: Time vs Idle Nodes Count for different networks when $S = 10000$

Figures 9.12 to 9.17 show the idle node count vs time for different networks. Notice that when stimulus is very low, as in figure 9.12, networks have similar number of idle nodes, because they all sense a relatively low stimulus. Similar phenomenon happen when task demand is extremely high, as in figure 9.17, because again, here networks perceive extremely high task demand for all tasks, regardless of the task's threshold.

Different networks have different number of idle nodes at mid to high range stimulus values. For example, in figure 9.14, network FST has the highest number of idle nodes, this is because it perceives low stimulus. Notice that in this experiment, sampling rate has a significant effect on the

perceived stimulus, therefore, this value of response rate that yielded this idle node curve seems to be a good value that balances the frequency of sampling and the task performance rate. In the same figure, we notice a sloped curve (FST+B network). This is due to the effect of the battery slowly running low on energy, and so the sampling rate was reduced, making the nodes perceive greater task demand, and hence less idle nodes result.

9.4.5 Dead Nodes

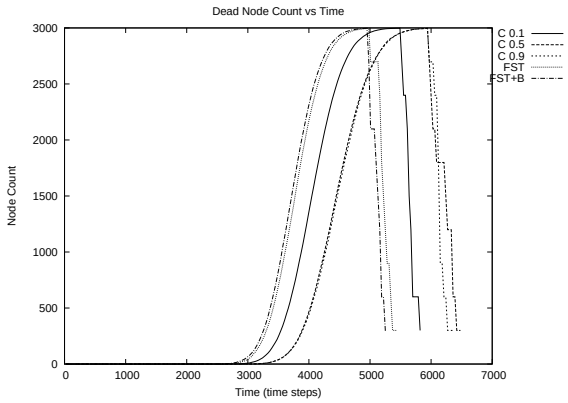

Figure 9.18: Time vs Dead Nodes Count for different networks when $S = 0.1$

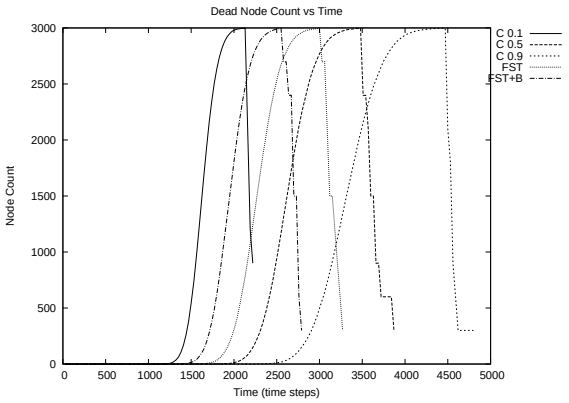

Figure 9.19: Time vs Dead Nodes Count for different networks when $S = 1$

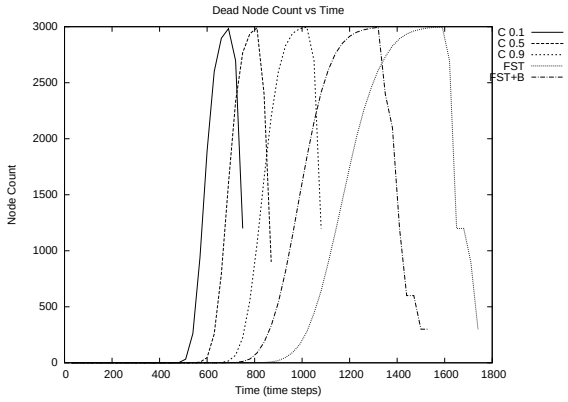

Figure 9.20: Time vs Dead Nodes Count for different networks when $S = 10$

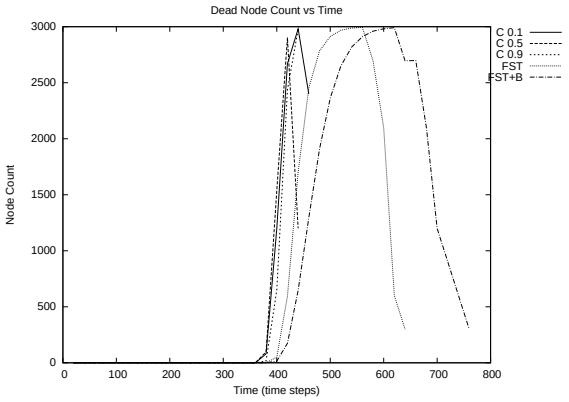

Figure 9.21: Time vs Dead Nodes Count for different networks when $S = 100$

Figures 9.18 to 9.23 show the time line of the number of dead nodes vs

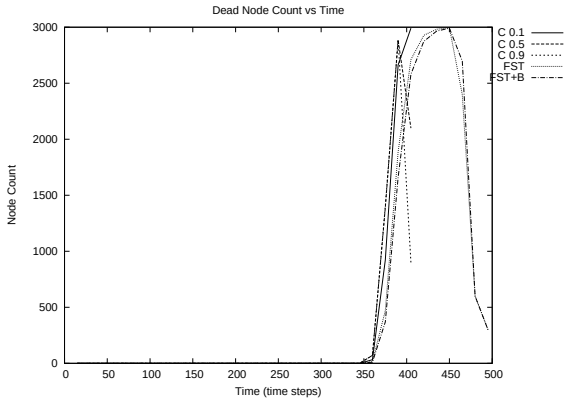

Figure 9.22: Time vs Dead Nodes Count for different networks when S = 1000

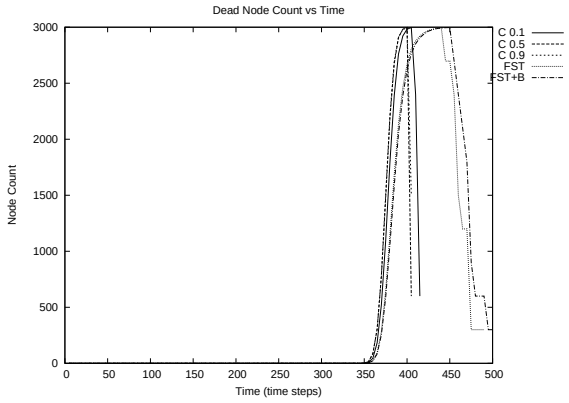

Figure 9.23: Time vs Dead Nodes Count for different networks when S = 10000

time for different stimulus levels. The following observations can be taken from these graphs:

Death Speed: at very low task demands, the network lives almost double the time it took for the very first node to die, whereas at extremely high task demand, it takes only 7^{th} of the time a network lived before its first node died for the whole network to be dead.

At very high stimulus levels: at high stimulus levels, all nodes exhibit similar death rate, or die in very close range of time. This is because all networks, independent of the sampling rate, perceive extremely high stimulus, so the active time is extremely high, and all other factors governing the energy consumption diminish in the presence of such a high task performance rate.

9.4.6 Multiple Adaptive Models

There are two adaptivity models working together in the experiments of this chapter. These are the FRT model for action selection, and FST/FST+B models for sampling decisions. The FRT is used to make action selection decisions in all experiments, while different sampling schemes were used for different networks. At extreme task demands, the influence of the

adaptivity of the sampling models was marginal. While it was much more obvious at lower task demand levels. Actually, FST and FST+B almost reverse the effect of the FRT at low task demands. This can be seen clearly by comparing figures 9.18 and 9.23.

9.4.7 Sampling Frequency

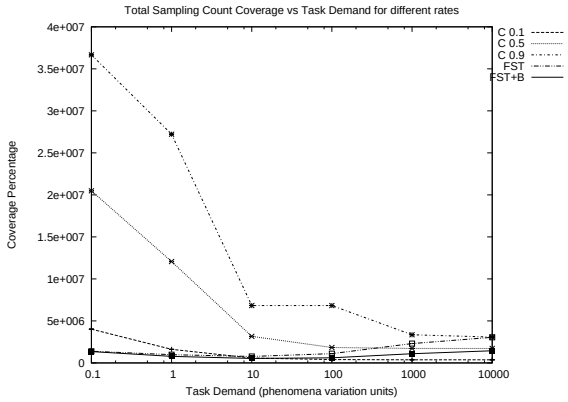

Figure 9.24: Time vs Sampling Count for different networks

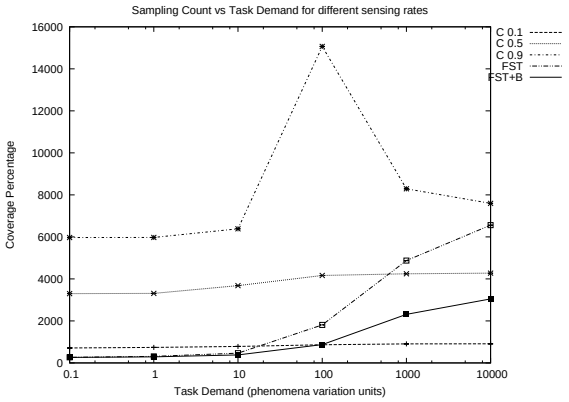

Figure 9.25: Time vs Sampling Count for different networks

Average Sampling Frequency

In figure 9.25 we can see that the average sampling rate stays constant, independent of the demand levels, for networks that use the CS models. While, for FST and FST+B, mean sampling frequency increases with the increase of demand levels. FST+B sampling growth rate is slower than that of the FST network because the battery energy content influences decisions made in the FST+B network. This growth in sampling frequency increases the resolution of a node's view of the environment, which in many sensor network applications would be an application requirement, i.e. leading to a better task performance. While when there is no interesting activity in the network, a low sampling frequency would be sufficient.

Total Sampling Frequency

In figure 9.24, at low task demand values, the *C*0.9 and *C*0.5 networks does high frequency sampling. This is because sampling is a cheap operation power-wise, and the task demand is low, hence you get nodes that sample frequently for very long time because they do not have to perform power-hungry tasks. With the FST and FST+B networks, task demand is low, so there is no power-hungry tasks being performed, but also there is little sampling to do. The data samples taken by the high-frequency sampling nodes may be redundant, or may be expensive in terms of memory or computational complexity. At high task demands, all nodes converge to similar total sampling frequency because they all consume most of their energy performing the power-hungry tasks, and so the effect of the sampling frequency differences diminish to a minute size. However, we still can see that at high demand levels, FST networks produce the highest sampling frequency, followed by the *C*0.9 network, then the FST+B and the *C*0.5 networks. Finally and as expected, the *C*0.1 network comes with the lowest sampling frequency.

9.5 Experiments Conclusions

The interpretation of the sampled data had a great effect on the way the network behaved, which actually allows us to conclude that the design of the interpretation of the environmental data gathered is just as important as modeling the sampling behaviour itself, and that both aspects are very coupled and need to be considered together.

Sampling will have minimal effects, in most cases, on a node's behaviour and resources as long as it is done with small power print. This is the case with many sensors and other sampling equipments. Therefore FST+B model may not have great positive impact on the longevity or task performance of a node, and the FST model would be sufficient for sampling control.

FST+B can be considered if sampling equipments have high power requirements.

9.6 Summary

In this chapter we investigated the effects of sampling rate on the network performance. We also saw typical effects of sampling on sensor node lifetime. We examined the inter-dependency between task performance, perceived stimulus, and the sampling models. Consequences of the sampling model on total coverage, average coverage, node coverage, and network death were all observed and discussed.

There still remain many issues that could be investigated, such as what if sensing was a power-hungry activity itself, but for reasons of time and space, we only restricted our discussion to a small window that covers the wider and more common range of sensing power requirements.

Chapter 10

Communication

In previous chapters we introduced threshold-based models to control action selection, task discontinuation, and sampling. In this chapter, we introduce a threshold-based model to control communication within a sensor network.

We will start by a general discussion of communication in sensor networks, and then sections about the models will follow.

10.1 Communication in Sensor Networks

Sensor networks need communication for a variety of reasons. These include receiving instructions from human users and sending back data to servers, and/or base stations. Communication could be used as well to coordinate among nodes, conduct collaborative tasks, or influence the operation of peer nodes in the network.

It is important to realise the communication implications on sensor nodes in terms of energy [9] [17], which create the tradeoff between the observed benefits in terms of task performance as a result of communication, and the adverse effects of communication on network and node lifetimes. If nodes are voluntarily mobile, then mobility would constitute the greatest power hungry activity that the node has to cope with, in which case communication can be allowed almost without restrictions. However, if nodes are immobile, then communication is often the activity that consumes most

of the available energy. To illustrate the difference in power consumption between communication devices and other components on a sensor node, we compare a typical sensing device to a typical communication device. Sensors are generally low energy consumption components, however, that does not necessarily preclude the possibility of the presence of a sensing component that has a high power footprint. Here, our focus is on the more common energy profiles of different device families, rather than the specific or exceptional cases. For example, operation current for the temperature sensor $MCP9700/01$ is $12\mu A$ at maximum and $6\mu A$ typically. This is compared to a very low power transmitter like $TXE - 315 - KH$ from LINX Technologies that typically requires $1.5mA$ to operate. While this transmitter is relatively low in terms of power consumption, it is still 125 times more power consuming than the temperature sensor.

Sensor nodes usually use low power protocols, such as Zigbee [6]. Common wireless communication protocols do not particularly satisfy the peculiarly energy-constrained profile of sensor nodes, and would result in quick depletion of node batteries. This is primarily due to the polling nature of many conventional protocols, in addition to the frequency they operate at, and the distances they target. Examples of such unsuitable protocols include the now infamous Wi-Fi and FDMA.

Sensor nodes often use multi-hop communication schemes as opposed to long-haul ones to conserve energy. It is established that the communication power requirements are proportional to the radius of communication, or distance, quadrupled. Although multi-hop schemes has some adverse implications on the complexity of the communication protocols, and the computation cost of networking, it is offset by a great energy profile reduction. You can refer to a more elaborate discussion of this issue in [133].

Communication seems to be inevitable in many sensor networks, however, we believe that sensor nodes should not depend on the availability of communication to perform their task allocation and action selection. We see that communication with base stations or human users is a different and separate issue from communication that affects the sensor node and/or network task performance [183]. For example, inter-node communication for the purpose of task allocation may happen depending on the environ-

mental conditions, while communication between users and nodes may happen on preset intervals. Also separate communication equipments may be used for each type of communication. For instance, a node may have a long haul equipment that is only used once per month to transfer data to a base station, while another low-power communication module could be used otherwise to coordinate with other nodes. We think that both issues should be addressed separately. In this chapter, we focus on communication activities for the purpose of improving the network and node quality of coverage, i.e. task performance, and longevity rather than data aggregation or dissemination purposes.

In the next few sections, we will discuss models we propose to use to regulate communication activities within a sensor network.

10.2 Non-Adaptive Communication Schemes

We can think of two basic non-adaptive schemes for communication: 1) A preset interval of synchronisation or communication can be used by a sensor network to regulate inter-node communication, 2) A no-communication policy . However, both policies may not be ideal under different conditions for some environments and/or application domains. Though the latter scheme results in big savings in energy consumption, it would be advantageous, and may be sometimes necessary to the application, to perform some communication. Also in some applications, the node may not depend on communication to operate, but would benefit from communication if available. Both of these schemes are also non-adaptive schemes because they are decided upon at design time, rather than runtime. They do not autonomously react to environment-based, policy-based, or system-based variables.

Let's take an example to illustrate the deficiencies that can occur from using static communication schemes. If in our *pollution monitoring task* has been observing low pollution levels, then the sensors will likely be idle. In this case, communication may be a futile activity, as there is no need to coordinate or collaborate over any monitoring activities. But imagine, if there is more than one phenomenon highly active in the vicinity of this network. In

this case, it might be advantageous if different monitoring tasks are assigned to different nodes, and this can be done through communication. If nodes do not communicate, some phenomenon may be under-monitored, while others are over-monitored. From this simple scenario, we can conclude that periodic communication schemes may waste energy when communication is unnecessary, and the no-communication scheme may adversely impact network coverage quality.

In the next section, we will discuss one of the non-adaptive schemes, namely, the constant communication probability model (CC for short).

10.3 The Constant Communication Probability Model (CC)

Previous chapters discussed the use of threshold-based models to control, regulate, or help making decision with respect to action selection, task discontinuation, and sampling frequency. However, they all assume a no-communication policy, i.e. nodes never inter-act or communicate for the purpose of making decisions. In the model described in [22], an assumption of a no-communication policy was made, even though in real insect societies it is well documented that there are several communication mechanisms employed [45]. In this chapter we will explore employing a threshold-based model to regulate communication in sensor networks. It is worth mentioning that we have not done any experiments on insects or any other real societies to suggest employing this model, but we were rather inspired by the suitability of this model to regulate other activities in insect societies, or rather imitate the regulation observed in these societies. To be able to assess the performance of the adaptive threshold-based model, we compare it to a base case model, that is the *Constant Communication Probability* model, or *CC* for short.

In the *CC* model, an individual would communicate with nodes within radius R_c from its centre with a constant probability p_c. The value of the probability is preset before the system is employed, i.e. at design time. This is a non-deterministic scheme corresponding to the periodic communication

model in deterministic terms. Periodic communication schedules were and still are employed in many computing systems.

Setting the value of the communication probability p_c to a low value will mean that individuals will communicate very rarely. Rare communication is not always beneficial for network performance. When p_c is low (p_c approaches 0), the number of communication events used by individuals out of N possible communication slots approaches 0. Although this may work well in situations where there is little work to be done, or when communication may introduce undue overhead, this is not always the case. Similarly, if p_c is high (p_c approaches 1), the number of communication events utilised by nodes will be close to the N communication time slots available. That may work well in environments with highly dynamic demand levels, or those that has high inter-task dependencies, or require a great deal of coordination. Again, this may be a waste of energy, or unnecessary, if communication does not benefit the application requirements, i.e. improve the quality of service. Medium p_c values will give less than optimum results at either extremely high or extremely low demand levels. In addition, such values will not be optimal for continuous spectrum demand levels. For example, assume that air pollution variation is the stimulus that modulates the communication rate between nodes. The variation of air pollution may span a wide range of values and therefore one p_c value may never be appropriate for all situations.

The mathematical representation of the previous discussion is as follows:

$$C(Task) = p_c \times N \tag{10.1}$$

$$\lim_{p_c \to 1} C(Task) = \lim_{p_c \to 1} p_c \times N = N \tag{10.2}$$

$$\lim_{p_c \to 0} C(Task) = \lim_{p_c \to 0} p_c \times N = 0 \tag{10.3}$$

where $C(Task)$ is the number of communication slots used by a node to communicate with nodes in its vicinity for the purpose of coordination and making task allocation decisions.

where p_c is the probability a node will communicate with peers in the vicinity.

Note that there is an assumed decoupling between the physical communication equipment, and the application view of the communication events. For example, an application program might request the number of nodes performing wind data analysis in its vicinity. The communication module would return a value, however, this does not mean that communication with peers in the vicinity has actually happened. The communication module controller makes this decision based on context information and not based on application specifically communicated requests. This was discussed in more details in chapter 3.

Sensor networks are often employed in unpredictable environments. For example, a sensor network to monitor traffic might have very low variations in the traffic due to clear roads at night and most of the day. This means there may be no particular need to communicate for the purpose of coordination as nodes may opt for staying idle most of the time, preserving their energy reserves until needed. When the rush hour starts, the sensors in the area will have massive traffic changes to report, and may be complex monitoring tasks for which coordination, collaboration, and cooperation are needed. In such a scenario, high values of communication probability in normal traffic conditions might be inappropriate, while a low communication probability may cause unnecessary energy overheads when traffic is heavy for an extended period of time. In the next few sections, we will extend the threshold-based model from [24], [25], and [181] to control inter-node communication within a sensor network. In the next chapter, the associated simulations and their findings will be presented and discussed.

10.4 Fixed Stimulus-Based Communication Threshold Model (FCT)

The way many living systems survived for millions of years is said to be their ability to adapt to the environment [197]. We see the same concept applicable to sensor networks. If sensor networks can adapt to the stress

the environment exerts on them, they may stand better chance to survive for longer periods of time, as well as provide better quality of service, i.e. better network coverage. This adaptability is crucial in many behavioural aspects of the network including the inter-node communication decisions. Individual nodes may benefit from being able to make communication-related decisions dynamically according to the information they gather, rather than adhering to one rigid periodic scheme of communication.

To make the idea clearer, let us look at an example scenario. For our Traffic Monitoring task, if an individual node observes a great traffic increase, it may engage in high rate reporting task. However, before starting to work hard to report the detected changes, it may wish to consult with its peers in the vicinity in case one or more of them are already engaged in performing this particular task. If a node or more are performing the highest priority task of reporting the traffic status to a base station, another node may use this knowledge to make the decision of turning its attention to report the pollution level increase associated with the traffic increase. As aforementioned, once the traffic subsides, communication amongst idle nodes becomes redundant, and actually can be considered an unnecessary squandering of the limited energy reserves.

When stimulus to perform a task is high, or when there are many tasks that need to be performed, the communication probability is likely to be high in order to facilitate coordination, and improve quality of service. On the contrary, if an individual is performing a task, while the stimulus for that task is low, it is likely that the individual should not need to communication with its peers as the decision making process through communication could be even more expensive than performing the task itself. This can be modeled mathematically, drawing from the response threshold model, as follows:

$$\zeta(\theta, S) = \left(\frac{S^n}{S^n + \theta^n}\right) \tag{10.4}$$

Where $\zeta(\theta, S)$ is the probability an active node will communicate with regard to a task with threshold θ when the task-associated stimulus is S. Note that $\zeta(\theta, S)$ approaches 1 as S approaches ∞ (there is very high demand for this task), and approaches 0 as S approaches 0 (there is little demand for this

task). The following equations represent this mathematically:

$$\lim_{S\to\infty}(\frac{S^n}{S^n+\theta^n}) = 1 \tag{10.5}$$

$$\lim_{S\to 0}(\frac{S^n}{S^n+\theta^n}) = 0 \tag{10.6}$$

This concludes our introduct in of the FCT model, next we will extend this model to account for the resource availability within a node.

10.5 Adaptive Resource-Based Communication Threshold Model (FCT+B)

The previous section discussed the FCT extension of the FRT model in [22]. It used the task demand levels to decide either to communicate with peers or make decisions without using information from peers. In this section, we extend this model even further to include resource levels, in particular battery levels, in making communication decisions. Sometimes, a node may be able to detect that task demands are high, however, it cannot respond because the resource required to perform the task are running low. In such circumstances, nodes may need to communicate with peers that task demands can be detected, but no response is possible on the node because of lack of resources. May be some other node could then respond to such help request messages. Accounting for battery levels in communication decisions is only an example of incorporating resource levels available to nodes in making task-associated decisions. Battery levels is a common representation of scarce resources in many sensor networks, and hence can be used without loss of generality and relevance. Note that the way adaptivity is modeled is application- and domain-dependent. For example, it could be said here that when resources are low, then nodes may favour not communicating because communication in itself is an energy-exhausting process. However, in other applications the reverse could be true. We assume the latter in our examination of the model in this and the next chapter.

The following mathematical equation represents the battery-modulated control component in the communication decision making process:

$$\zeta(B) = 1 - (\frac{B}{B_0})^m \tag{10.7}$$

The following are the trends that follow from equation 10.7:

$$\lim_{B \to 0} \zeta(B) = 1 \tag{10.8}$$

$$\lim_{B \to B_0} \zeta(B) = 0 \tag{10.9}$$

The following is an algorithm representing how a node would use battery levels and task demands to make decisions on communication events, given stimulus S, task-associated threshold θ, battery level B, and a resolution μ:

1 $\zeta(S, \theta) = \frac{S^n}{S^n + \theta^n}$
2 $\zeta(B) = 1 - (\frac{B}{B_0})^m$
// Now, simply multiply the two terms, and divide by the
// resolution
3 $\zeta(S, \theta, B) = \frac{\zeta(S,\theta) \times \zeta(B)}{\mu} = \frac{[\frac{S^n}{S^n + \theta^n}][1 - (\frac{B}{B_0})^m]}{\mu}$
// where $\zeta(S, \theta)$ is the probability a node will communicate
// with peers with regard to a task when the
// task-associated demand is S disregarding the battery
// level B (equation 10.4).
// $\zeta(B)$ is the probability a node will communicate with
// peers with regard to a task when the energy level on
// this node is B disregarding task-associated demand S

Algorithm 4: FCT+B Model algorithm to determine $\zeta(S, \theta, B)$

Algorithm 3 results in the following mathematical equations representing the communication probability trends:

$$\lim_{S \to \infty, B \to 0} \zeta(S, \theta, B) = 1 \tag{10.10}$$

$$\lim_{S\to 0, B\to B_0} \zeta(S, \theta, B) = 0 \tag{10.11}$$

$$\lim_{S\to 0, B\to B_0} \zeta(S, \theta, B) = 0 \tag{10.12}$$

$$\lim_{S\to \infty, B\to B_0} \zeta(S, \theta, B) = 0 \tag{10.13}$$

These equations represent the following generalised trend of node behaviour: 1) If task demand is high and there is enough battery power, then the node will not most likely need to communicate, as it can afford to perform the task 2) Communication is not needed as well if task demand is low, regardless of battery power status 3) If battery power is low, and task demand is high, then the node would most likely communicate with its neighbours to negotiate a solution or notify others to perform the work needed.

This concludes our introduction of the FCT+B model.

10.6 summary

In this chapter we introduced an extension to the FRT model used originally in [22] to make action selection decisions. Our extension uses the model to make task-associated communication decisions. This extension can be seen as an additional layer of adaptivity which improves the performance and longevity of sensor networks. We also introduced a component to modulate the communication decisions according to the battery levels within a node. In the next chapter we will explain the simulations that were performed to examine the effect of the models we introduced in this chapter on a sensor network, together with their results. We also discuss some conclusions that can be drawn from the results.

Chapter 11

Communication Simulations

11.1 Introduction

In this chapter, we conduct simulations of our hypothetical scenario in order to investigate the FCT and FCT+B models (see chapter 10).

Appendix E.1 lists the settings and parameter values for this set of experiments. Next section will present the objectives of our simulations followed by the results and conclusions.

11.2 Experiment Objectives

The experiments of this chapter aim at:

- Testing the effect of applying the FCT Model on the dynamics of task performance in sensor networks.
- Testing the effects of applying the FCT+B Model on the dynamics of task performance in sensor networks.
- Identifying what applications could benefit from each of the three models CC, FCT, and FCT+B.

11.3 Results

Following sections will discuss the results with respect to different metrics of the network performance.

11.3.1 Network Lifetime

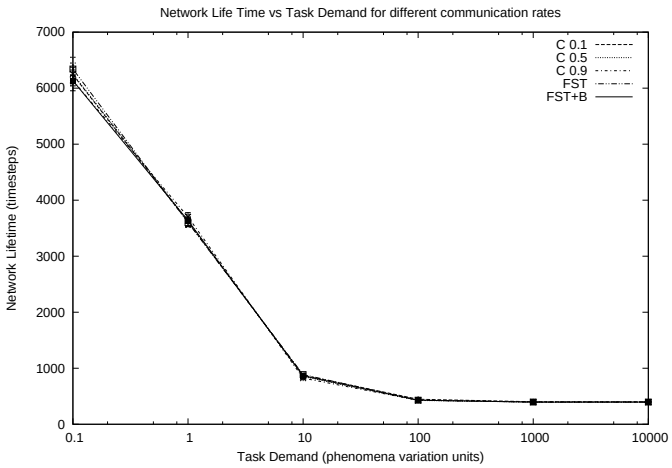

Figure 11.1: Task demand vs lifetime for different networks

From figure 11.1, it seems that no matter what communication model the network use, it does not affect the lifetime of the network. Lifetime is only affected by the intensity of the task demand. Clearly, the higher the task demand, the shorter the network lifetime is. This can be explained by two factors:

1. The increase of the task performance with the increase of the task demand, mathematically represented in equation 4.1.

2. The increase of communication frequency with the increase of the task demand, mathematically represented in equation 10.4.

Although these two factors have a strong impact on the lifetime of the network, we believe that the vast difference in the lifetime of the network observed between the two extremes of the task demand intensity is mainly a result of the communication model adopted. An example will help illustrate

this situation more clearly. In the settings of these experiments, the cost of a communication event between any pair of nodes equals to 10 energy units. The cost of performing a task, no matter which one it is, is 20 energy units per time step. Now, at low task demand intensity, say 0.1 units, the probability of performing the task T_A, with threshold θ_A is:

$$\Psi(0.1) = \frac{0.1}{0.1 + 50} \approx 0.002 \tag{11.1}$$

If we calculate the time a node will spend performing this task under these circumstances out of an interval of 100 time steps we obtain:

$$100 \times \Psi(0.1) = 100 \times 0.002 = 0.2 \tag{11.2}$$

And from this, we can calculate the energy expenditure for this level of task performance to be:

$$E_A(0.2) = 0.2 \times 20 = 4 \tag{11.3}$$

Therefore, in task performance, 4 energy units were used. Now to calculate the communication energy expenditure, we take a node with say 4 neighboring nodes. Under a 0.1 task demand intensity, the frequency of communication can be calculated to be:

$$\zeta(0.1) = \frac{0.1}{0.1 + 50} \approx 0.002 \tag{11.4}$$

Again multiply this by the time interval of 100, the result is:

$$\zeta(0.1) \times 100 = 0.002 \times 100 = 0.2 \tag{11.5}$$

Now to get the energy expenditure, multiply this by 10 to get:

$$E_{comm}(0.1) = 0.2 \times 10 = 2 \tag{11.6}$$

This seems less than the energy consumption used for task performance, until you remember that this sort of cost is repeated with the 4 neighbours of the node, leading to a total cost of $5x2 = 8$ energy units, i.e. more than double that used in task performance. Sum the task performance and the

communication energy expenditures, an amount of $10+4 = 14$ energy units is needed.

Now let's do the same calculations at high task demand, say 10000 task demand units. The probability of performing a task T_A, with threshold θ_A is:

$$\Psi(10000) = \frac{10000}{10000 + 50} \approx 0.995 \tag{11.7}$$

If we calculate the time a node will spend performing this task under these circumstances out of an interval of 100 time steps we obtain:

$$100 \times \Psi(10000) = 100 \times 0.995 = 99.5 \tag{11.8}$$

And from this, we can calculate the energy expenditure for this level of task performance to be:

$$E_A(99.5) = 99.5 \times 20 = 1990 \tag{11.9}$$

Therefore, in task performance, 1990 energy units were used. Now to calculate the communication energy expenditure, we take a node with say 4 neighboring nodes. Under a 10000 task demand intensity, the frequency of communication can be calculated to be:

$$\zeta(10000) = \frac{10000}{10000 + 50} \approx 0.995 \tag{11.10}$$

Again multiply this by the time interval of 100, the result is:

$$\zeta(10000) \times 100 = 0.995 \times 100 = 99.5 \tag{11.11}$$

Now to get the energy expenditure, multiply this by 10 to get:

$$E_{comm}(10000) = 99.5 \times 10 = 995 \tag{11.12}$$

This number needs to be multiplied by 5 to get the total communication expenditure of $5 \times 995 = 4975$. Add this to the task performance expenditure to get a whopping amount of $4975 + 1990 = 6965$ energy units.

The previous calculations give us an idea of how high the impact of communication overhead has on the system energy expenditure and consequently the network lifetime. Notice that the communication expenditure

constitutes 72% of the total energy expenditure at both high and low task demand intensities.

The similarity in lifetime among the different networks mostly will make comparison between networks easier with respect to other metrics discussed in this chapter.

11.3.2 Mean Coverage

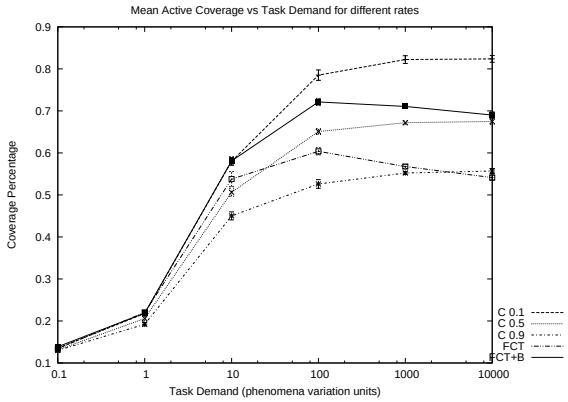

Figure 11.2: Task demand vs Mean Active Coverage for different networks

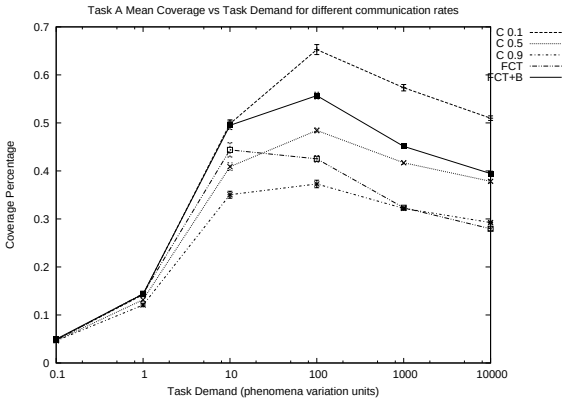

Figure 11.3: Task demand vs Mean Task A Coverage for different networks

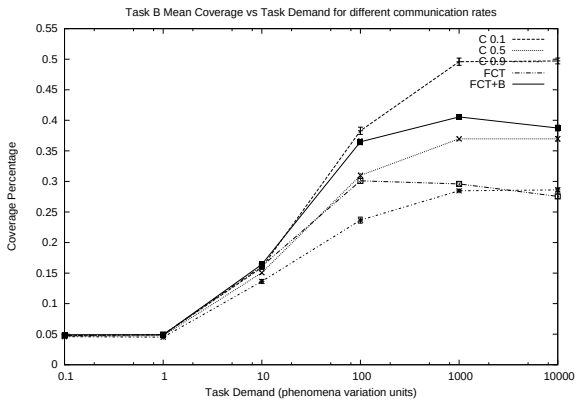

Figure 11.4: Task demand vs Mean Task B Coverage for different networks

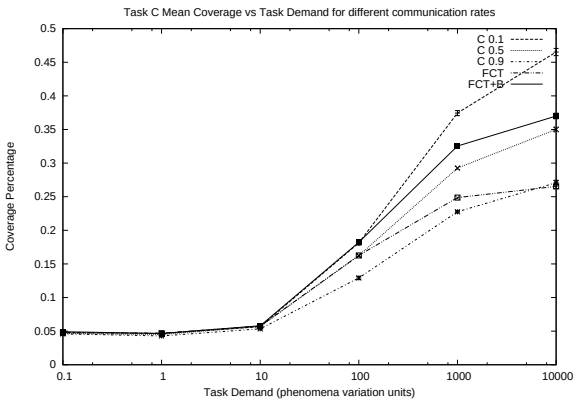

Figure 11.5: Task demand vs Mean Task C Coverage for different networks

In this section, we discuss the implications of the Communication Model on the task performance, or in particular, network coverage within the

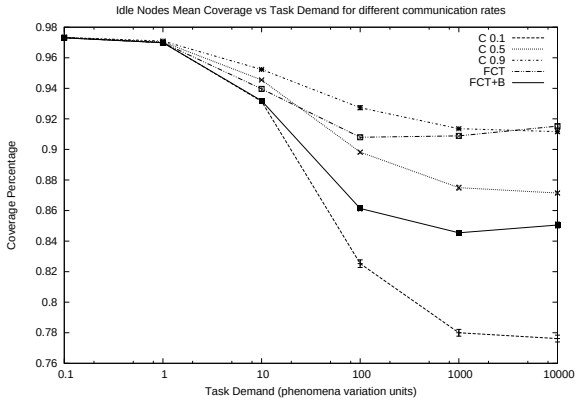

Figure 11.6: Task demand vs Mean Idle Coverage for different networks

different sensor networks of our simulations. From fig 11.2 to 11.5, we can make the following observations:

At low task demand levels

All networks converge to low average coverage of almost the same value. This is due to two factors: 1) nodes do not communicate when they are idle, and if they are idle most of the time, because there is only minute task demand detected, 2) then they also do minuscule amount of communication. This results in the observed convergence of the networks.

At high task demand levels

As task demand level increases, the difference between different networks becomes apparent. Networks that communicate rarely yield higher coverage on average. This is because more nodes per neighbourhood radius become active for a single task. Nodes are not aware of this because they rarely communicate. Networks that communicate fervently at extreme task demands perform least on average because the least number of nodes would be active with respect to a task within a neighbourhood radius. We can see an almost clear plateaus in terms of average performance curve in the region between middle to high task demand levels. This is because at certain frequency of communication, the network collectively succeeds in restrict-

ing the number of active nodes per neighbourhood to a minimum. Any further increases to the amount of communication does not result in any reduction in the number of active nodes within a neighbourhood. This frequency could be detected and used as an upper limit to the communication probability. However, this issue is out of the scope of this dissertation.

Adaptive vs non-adaptive models of communication

Adaptive models changed their skins, so to speak, with the change of task demand levels. This is beneficial in many sensor network applications. For the FCT model, the network behaved like all other networks at low demand levels. At medium demand levels, the FCT network behaved similar to a $C0.5$ CC network. Finally, at high demand levels, it behaved similar to a $C0.9$ CC network. The FCT+B network has also changed its behaviour according to demand levels. At low task demands, it behaved like a $C0.1$ CC network, and slowly with the growth of the task demand, the FCT+B network morphisized into a $C0.5$ CC network. It did not converge to a $C0.9$ network because it had the battery component that capped its communication frequency to half that of the FCT network.

In fig 11.3, Task A's performance display a peak at some point. This is an optimal point of performance where the stimulus exerted by the environment motivates the network to activate a number of nodes that together achieve a very high coverage for Task A (the highest priority task), and not as high for other tasks, namely Task B and Task C. When stimulus gets very high for all tasks, the differences between their thresholds become negligible, and so Task A loses its high priority status, and so its average descends. Also notice that this peak happens at the mid range task demand region, when the communication frequency is not yet able to cap the number of active nodes per neighbourhood to its optimal status, i.e. one per neighbourhood, which results in this high coverage. As task demand increases to reach extremes, the fervent communication makes sure that the least possible number of active nodes exist per neighbourhood, and so all tasks get the same number of active nodes at that point. While at the mid range, Task A gets highest coverage, followed by Task B, and finally comes

Task *C*.

In figures 11.4 and 11.5, both tasks *B* and *C* exhibit a continuously ascending coverage with the increase of stimulus magnitude. This is because continuous increase in stimulus steadily increase the number of active nodes for these tasks, until finally they even get to be as prioritised as Task *A*. Notice the difference between the trend of these tasks' average coverage on the graphs, and that of task *A*.

The points made regarding the average coverage for active nodes are identically observed on each task's graph separately. This is an important sign of the consistency of the network behaviour across all tasks.

Mean idle nodes coverage exhibits an interesting high average. At worst we see that 85% of the network terrain is covered by idle nodes. This emphasises the communication importance in keeping nodes that do not need to be active idle until the situation changes. Communication here helps support high redundancy and fault tolerance. It also allows for better task assignment because as nodes know that other nodes are performing a set of tasks, they can deal with other tasks that no other nodes are performing.

11.3.3 Total Coverage

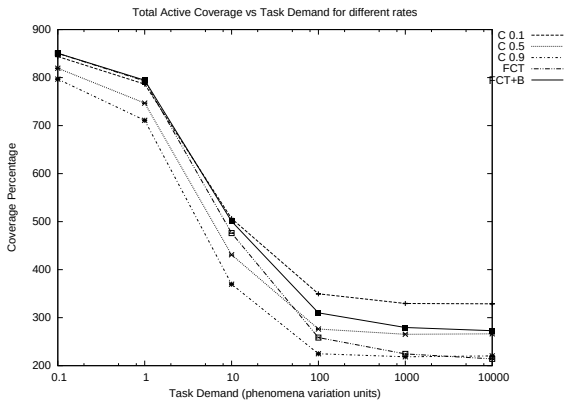

Figure 11.7: Task demand vs Total Active Coverage for different networks

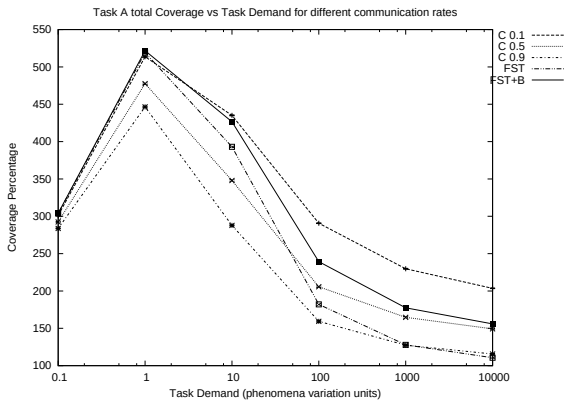

Figure 11.8: Task demand vs Total Task A Coverage for different networks

In figures 11.7 to 11.10, the total coverage graphs are presented. From these graphs we can make the following observations:

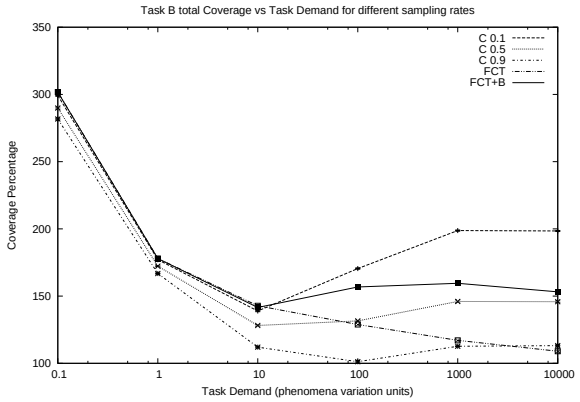

Figure 11.9: Task demand vs Total Task B Coverage for different networks

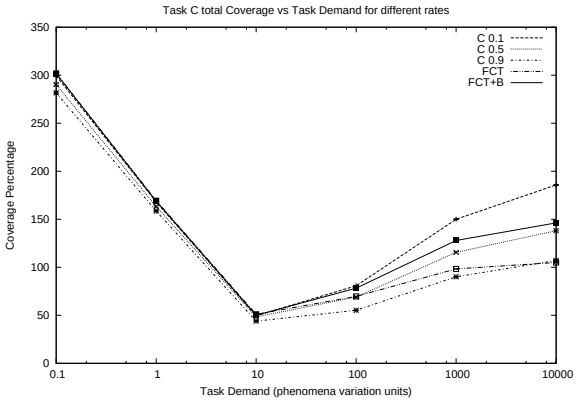

Figure 11.10: Task demand vs Total Task C Coverage for different networks

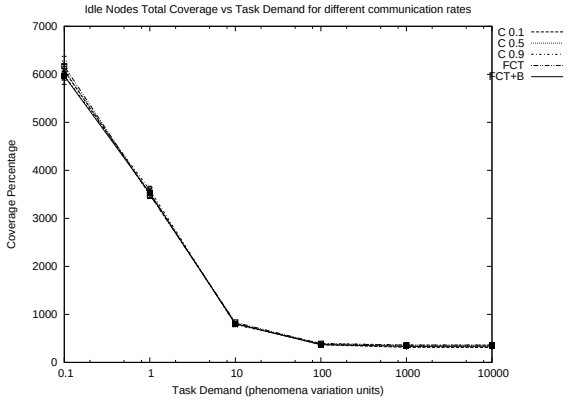

Figure 11.11: Task demand vs Total Idle Coverage for different networks

The *C*0.1 **network**

This network achieves the highest total coverage. This can be attributed to two factors:

1. This network communicates rarely irrespective of the task demand intensities, allowing it to save large amount of energy for task performance, i.e. coverage.

2. The fact that the nodes rarely communicate reduces the limitation on the number of active nodes per neighbourhood, allowing for a slightly higher coverage on average, which adds to the total coverage. Note

that extremely high coverage on average does not necessarily result in the maximisation of total coverage, and the same is true with respect to extremely low coverage on average.

The *C*0.9 network

This network always, regardless of the task demand intensity, produces the least total coverage. This is because of reasons that mirror those given for the high total coverage of the *C*0.1 CC network. These reasons are:

1. This network communicates liberally irrespective of the task demand intensities, consuming high amounts of energy for communication for coordination.

2. This network's high intensity communication minimises the number of active nodes per neighbourhood, allowing for just enough coverage on average which reduces the average coverage compared to the situation where less restrictions on the active node count per neighbourhood with the *C*0.1 CC network.

The adaptive networks, FCT and FCT+B

Both FCT and FCT+B networks behave as a *C*0.1 network at extremely low task demand levels. On the contrary, when task demands are extremely high, these two networks tend to behave more like a *C*0.9 network. However, FCT+B network converges to a *C*0.5 network because of the restriction on its average coverage imposed by the decreasing resources level throughout the experiments. Such a restriction does not exist in the FCT network, and so its average coverage keeps increasing, and hence the low total coverage figure.

Task A, Task B, and Task C's total coverages:

Generally, all individual graphs follow similar trends to these found in the total active coverage. However, there are new observations that were not apparent from the active coverage graph, which are addressed here. We notice that the graph in figure 11.8 experiences a convex at mid range task

demand, while in figures 11.9 and 11.10, a concave curve is observed. This is because as task demand increases, nodes follow the prioritisation rules dictated by the task-associated thresholds. These dictate that task A is the most important, and so nodes has much higher probability of performing this task, practically ignoring other tasks. This results in this concavity at this task demand level. However, as the task demand continue to increase, the task importance, or priority, becomes irrelevant. Similar number of nodes tend to perform any of the tasks, which makes them converge to the same total coverage. To illustrate such dynamics, we give a simple example. If we say that humans need water, and good food to survive. When the human body is dehydrated, it asks for water, and if it is hungry, it asks for food. But if both are at shortage, it asks for both, but gives higher priority to water since humans die quicker from lack of water than from lack of food. However, under severe shortage of both water and food, the human body does not particularly differentiate, knowing that the shortage on both fronts is severe and could soon lead to system collapse.

Idle Node coverage:

Due to the high node redundancy in average discussed previously, total coverage have similar levels for all networks at all task demand intensities. Remember that the increase of active nodes does not always result in increase in the coverage, and this explains to a large extent what we are observing in figure 11.11 but in terms of idle nodes rather than active ones.

11.3.4 Idle Nodes

Figures 11.12 to 11.17 show the idle node count vs time for different networks. Notice that when stimulus is very low, as in figure 11.12, different networks have almost the same number of idle nodes, because they all sense an extremely low stimulus. At extremely high demand levels, as in figure 11.17, networks vary greatly in terms of the number of idle nodes, and this is due to two factors:

1. **Communication Frequency:** communication frequency increases fervently at high task demands, causing the active node count per neigh-

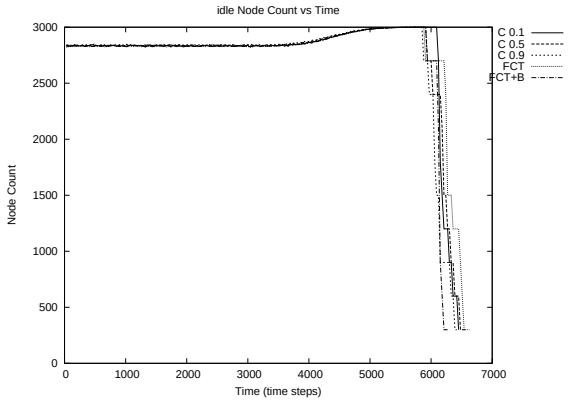

Figure 11.12: Time vs Idle Nodes Count for different networks when $S = 0.1$

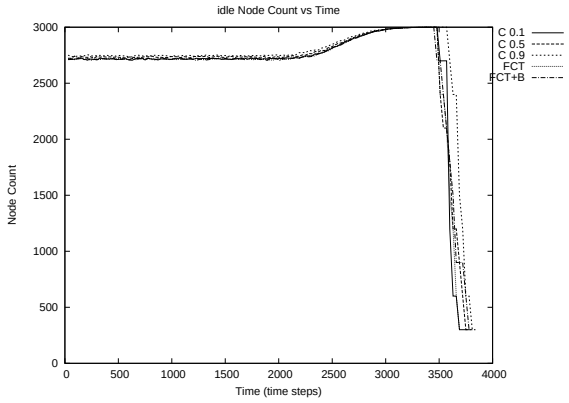

Figure 11.13: Time vs Idle Nodes Count for different networks when $S = 1$

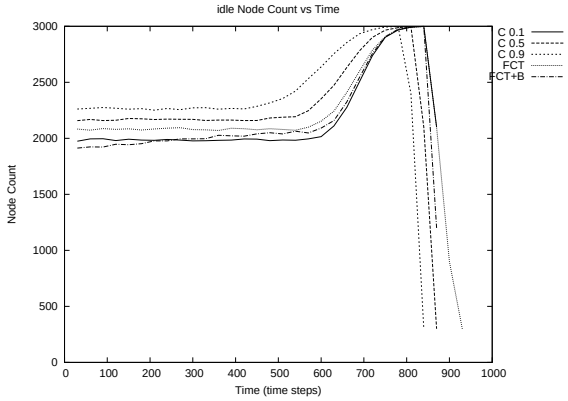

Figure 11.14: Time vs Idle Nodes Count for different networks when $S = 10$

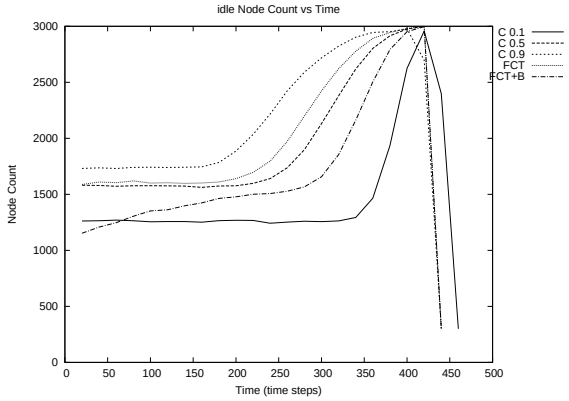

Figure 11.15: Time vs Idle Nodes Count for different networks when $S = 100$

bourhood to reach a minimum. This increases the number of idle nodes on average. This can be seen from the curves of the $C0.9$ and FCT networks in figure 11.17.

2. **Task Demand:** as this goes extremely high, nodes race to perform tasks which in the case of networks with low communication rate causes a low idle node count, as in the curve of the $C0.1$ network in figure 11.17. While in the high communication rate networks, this race does not result in any significant change in the number of active nodes per neighbourhood, hence the high idle node count in the curves of $C0.9$ and FCT networks. Networks that face medium

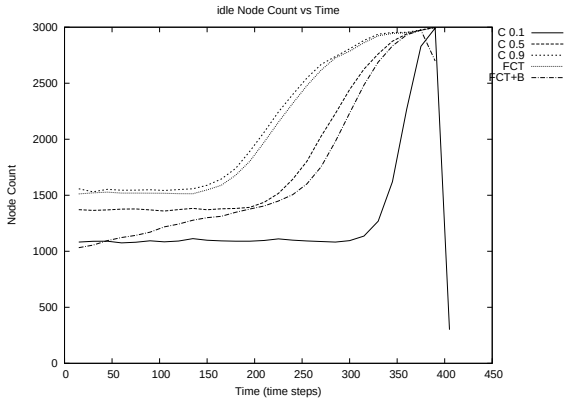

Figure 11.16: Time vs Idle Nodes Count for different networks when $S = 1000$

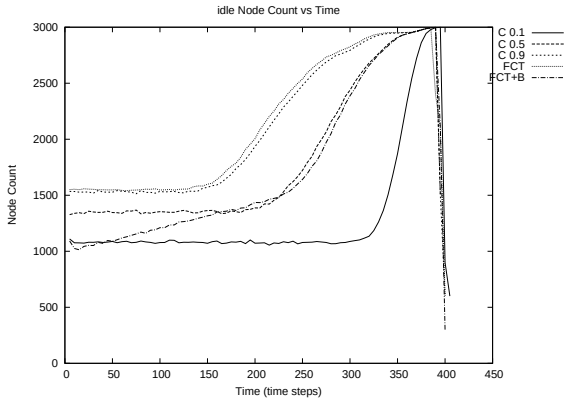

Figure 11.17: Time vs Idle Nodes Count for different networks when $S = 10000$

intensity task demand and use medium communication rate would behave somewhere between the two aforementioned extremes.

Note that the FCT+B network behaves differently at the beginning of the the experiment to at the end of its lifetime. This is governed by the battery levels within a node. As time goes by, batteries are gradually depleted, and consequently the network responsiveness to task demands in terms of communication rate decreases, and so the FCT+B network converges to become a $C0.1$ network rather than the $C0.9$-like network it was in the beginning of the experiment.

11.3.5 Dead Nodes

Figures 11.18 to 11.23 show the time line of the number of dead nodes vs time for different stimulus levels and different networks. The following observations can be drawn from these graphs:

Death Speed

At very low task demands, all networks die almost at the same time, and with the same rate. This similarity is explained by the similarity of their profiles at this task demand level, i.e. all idle, all not communicating much. At very high task demand levels, the situation is very different.

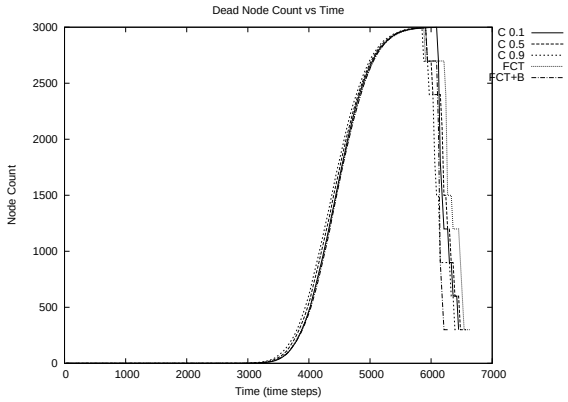

Figure 11.18: Time vs Dead Nodes Count for different networks when $S = 0.1$

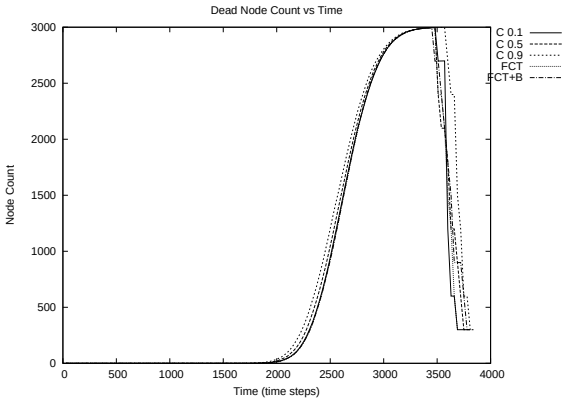

Figure 11.19: Time vs Idle Nodes Count for different networks when $S = 1$

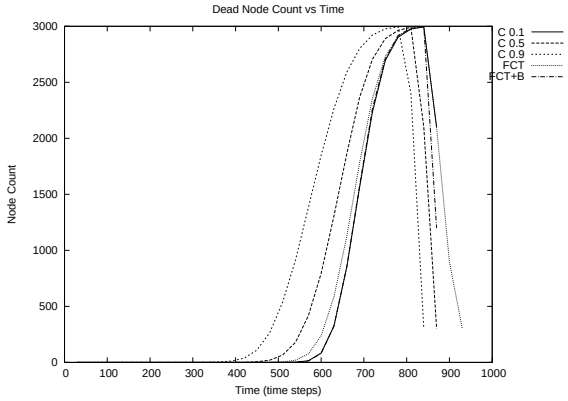

Figure 11.20: Time vs Dead Nodes Count for different networks when $S = 10$

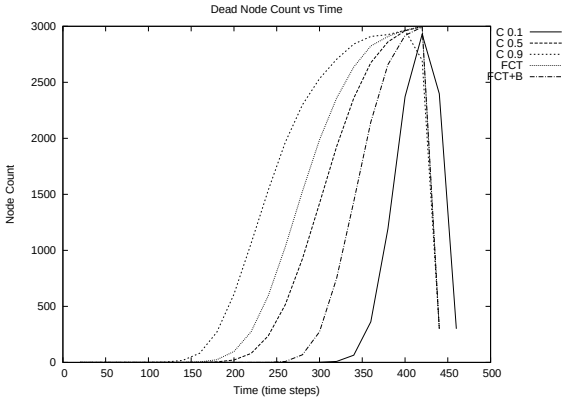

Figure 11.21: Time vs Dead Nodes Count for different networks when $S = 100$

Each node dies at a different point in time from others. Networks $C0.9$ and FCT have the slowest death rate, but the first to start losing member nodes. This is because these networks communicate and perform tasks extensively over their lifetime. Network $C0.1$ is the last to die although it performs tasks fervently because it rarely communicate, which results in great energy saving that allows it to last longer. Networks $C0.5$ and FCT+B are in the middle between the two previous death rates because they do communication but are restricted in doing so by either the 0.5 constant communication rate, or the battery component of the adaptive algorithm in the case of the FCT+B.

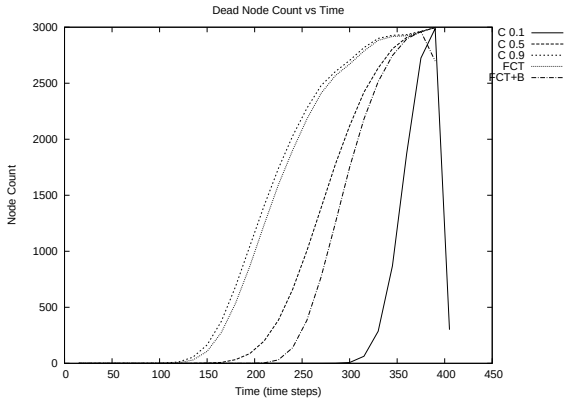

Figure 11.22: Time vs Dead Nodes Count for different networks when S = 1000

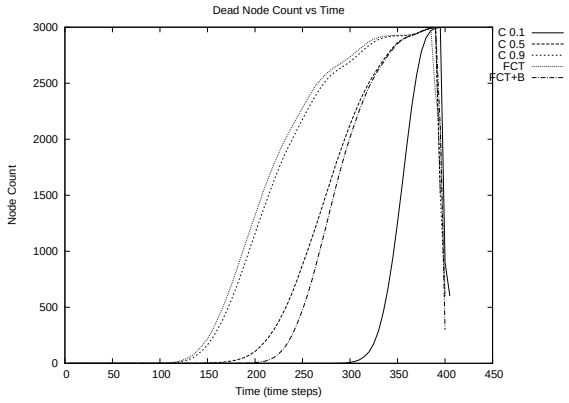

Figure 11.23: Time vs Dead Nodes Count for different networks when S = 10000

At medium range task demand levels

Each network exhibits different death rate at this task demand level. This is unlike at extremely high, or extremely low task demand levels. The fastest death occurring in network C 0.9 reflecting the lack of adaptivity of this network with respect to communication rate. The FCT network manages to live longer because it perceives the medium-range stimulus as so, and adapts its communication rate accordingly. These two are followed by the C 0.5 and the FCT+B networks respectively. The FCT+B lives longer again because it is double restricted in terms of communication by the task demand, and the battery levels. Finally, the $C0.1$ network dies last. This is because it communicates least, managing to survive longest as there is little communication overhead.

Adaptivity

There are two adaptivity models working together in this chapter's experiments. These are the FRT model for action selection, and the FCT or FCT+B models for communication decisions. FRT is used in all experiments, while different communication schemes were used for different networks. At extremely low task demands, the adaptivity of the communication models are marginally influential on the rate of death of nodes. While they are much more obvious at higher task demand levels.

11.3.6 Communication Frequency

We will address the total frequency of communication graphs first, then the mean communication frequency graphs next.

Total Communication Frequency

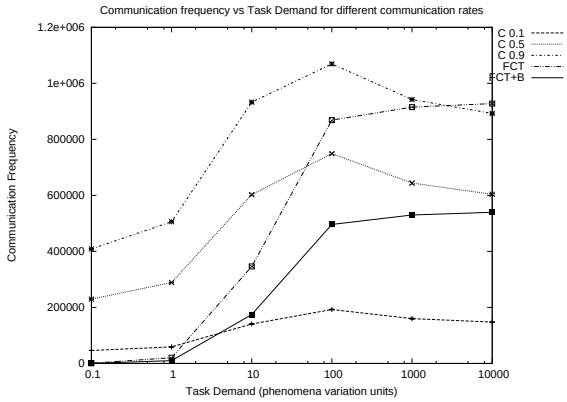

Figure 11.24: Stimulus vs Total Communication Count for different networks

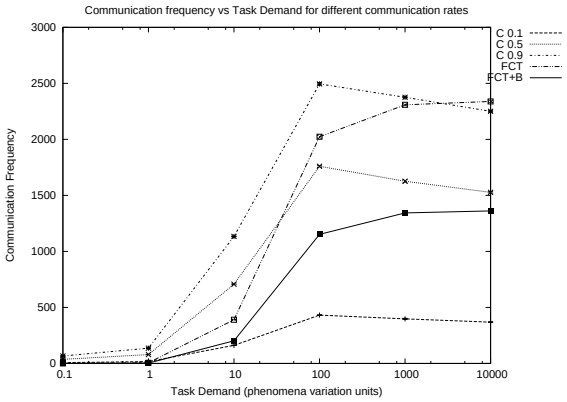

Figure 11.25: Stimulus vs Mean Communication Count for different networks

Following is explanations and discussion of some of the network's behaviours and curves appearing in figure 11.24:

C0.9 **Network:** this network maintains high total communication frequency throughout the graph, i.e no matter how much task demand it is exposed to. This is understandable and expected because it has a constant high communication probability that is independent of any task demand. This superiority in total communication changes only at extremely high task stimulus rates because then the FCT network yields a communication probability that is higher than 90%. We notice also that though the *C*0.9 network has highest communication frequency almost always, its value peaks at somewhere in the medium range stimulus. This is due to the fact that there are two opposing forces acting in this situation, and the balance is hit at some middle value of both forces. These forces are: 1) the communication frequency increase with the increase of task demand which works in the favour of increasing the total communication frequency, 2) the second is

the increase of the communication energy consumption by the task performance activity, causing less communication frequency activity in total . At some level of task demand, the communication frequency that happens on average accumulates to reach the peak as the task performance does not consume as much as it does at extremely high task demand levels.

The *C*0.1 **network:** in figure 11.24, this network exhibits low total communication frequency at all task demand levels, except at very low task demand levels. This is because communication in this network is governed solely by the constant 0.1 communication probability. At extremely low task demand levels, the FCT and FCT+B networks yield lower communication rates than 0.1, hence the lower total frequency count.

FCT and FCT+B networks: these two adaptive networks change their rate of communication with the task demand levels and energy available. This explains the way their curves appear in figure 11.24. At low task demand, the FCT network reduces its communication rate to a very low levels, similar to a *C*0.1 network. While at extremely high task demand levels, it simply fervently let communication happen to reach an optimal coverage, appearing like a *C*0.9 network. In the middle, between the two extremes, a gradually ascending communication rate occurs with the gradual increase in task demand levels. The FCT+B network behaves like an FCT or a *C*0.1 network at very low task demands. At high task demand levels, it behaves like a *C*0.5 network, unlike FCT that behaves like a *C*0.9 network. This is because it is capped by the diminishing energy levels from the battery. From these observations, we can see that adaptive networks may offer a better model of control, by exhibiting context awareness, which is beneficial in many sensor network applications.

Mean Communication Frequency

In figure 11.25, we see the mean communication frequency, i.e. the average instantaneous number of communication events per time step during an experiment. As expected, the *C*0.1 network exhibits a generally low fre-

quency of communication, while the $C0.9$ static network communication model yields a generally high average communication frequency. The FCT network behaves on average like a $C0.1$ network at low task demands, and more like a $C0.9$ at high task demand levels. Finally, the FCT+B network behaves like an FCT or a $C0.1$ network, while it behaves like a $C0.5$ network at high task demand levels.

It is worth-noting that there are no peaks in this graph, because the trend is only governed by the task demand, and not affected by the lifetime of the network, unlike total communication frequency graphs. Remember in the total communication frequency graph, the lifetime that results from the task demand levels affected the total frequency count creating the peaks apparent in figure 11.24.

11.4 Experiment Conclusions

From the result sections we can draw the following conclusions:

- The CC model is a good model for controlling communication if communication cost can be predicted and does not impact network longevity.
- The FCT model is a good model to control communication activities within a sensor node for applications that does not need periodic or frequent communication and for applications where the need for communication depends on conditions within vicinities of nodes.
- The FCT+B model is a good model to control communication when it is a costly activity and would be better avoided if resources are not available.

11.5 Summary

In this chapter we investigated the effects of communication rate on the network performance, lifetime, and general behaviour. We examined the interdependency between task performance and communication frequency. Con-

sequences of the communication model on total coverage, average coverage, node coverage, and network death rate were all discussed.

Communication energy consumption had dire consequences on the network performance. The energy overhead reduced the energy available to conduct other activities which sometimes was acceptable, or inevitable, but other times had a potentially unacceptable energy requirements.

There still remain many issues that could be investigated, such as what if the communication protocol changes, or if the reception and transmission had different energy footprints. Also the ratio of the energy consumption rate of a task to that of the communication variations may be an issue to look at. However, for reasons of time and space, we only restricted our discussion to a small window that covers somehow a profile of communication settings that resemble those appearing in typical sensor networks.

Next chapter will present general discussion on all the results of the dissertation. It will state the contributions of this thesis, and also identify future directions of research. Some general conclusions will be drawn too.

Chapter 12

Conclusions and Future Work

12.1 Conclusions

Knowledge and information has revolutionised the way we conduct our daily life today. Human hunger for information has provided great momentum towards monitoring and surveillance applications, i.e. gathering and collecting information. Wireless sensor networks is a technology that facilitate monitoring physical phenomena from a close proximity, with high resolution, and in a distributed fashion.

Through studying the literature of wireless sensor networks, we found that the main areas of concern for a sensor network can be classified into: 1) data management, 2) energy management, 3) communication management, and 4) task management . We also concluded that adaptivity can be a very advantageous characteristic of sensor networks given the often resource-constrained nature of sensor nodes. We took an approach in swarm intelligence research whereby micro-control system design leads to the macro-control effect desired by the application. Therefore, we looked at modifying the design of individual nodes to produce network-wide effects.

In order to address adaptivity within a sensor node, we produced a generalised sensor node architecture. Then, we identified four fundamental micro-control modules within that architecture. We proposed applying threshold-based adaptive algorithms as micro-control means within each of the four modules. These modules were: 1) Action Selection , 2) Sampling,

3) Task Scheduling, and 4) Communication .

We investigated the application of threshold-based control models within each of the aforementioned modules. For each module we adapted the FRT model in [22] to work according to the system requirements. This resulted in the following models explored:

- **FRT:** a model to control action selection processes given a level of task demand and a set of task-associated thresholds.
- **FRT+B:** a model to control action selection processes given a level of task demand, a set of task-associated thresholds, and a resource level, i.e. batteries, memory space, communication bandwidth, switching overhead, etc.
- **VRT:** a model to control action selection processes given a level of task demand and a set of task-associated thresholds. In addition, it does allow task specialisation if switching is a costly process.
- **VRT+B:** a model to control action selection processes given a level of task demand, a set of task-associated thresholds, and a battery level. In addition, it does allow task specialisation if switching is a costly process.
- **FDT:** a model to control task discontinuation (re-scheduling) processes given a level of task demand and a set of task-associated thresholds.
- **FDT+B:** a model to control task discontinuation processes given a level of task demand, a set of task-associated thresholds, and a resource level, i.e. batteries, memory space, communication bandwidth, switching overhead, etc.
- **FST:** a model to control environment sampling given a level of task demand and a set of task-associated thresholds.
- **FST+B:** a model to control environment sampling processes given a level of task demand, a set of task-associated thresholds, and a resource level, i.e. batteries, memory space, communication bandwidth, switching overhead, etc.

- **FCT:** a model to control communication processes given a level of task demand and a set of task-associated thresholds.
- **FCT+B:** a model to control communication processes given a level of task demand, a set of task-associated thresholds, and a resource level, i.e. batteries, memory space, communication bandwidth, switching overhead, etc.

We tested the adapted models, and their extensions, on a simulation environment within a hypothetical scenario. The results showed that the adaptive algorithms managed to produce many desirable behaviours, lacking in non-adaptive schemes, under various environmental conditions. For example, networks reduced task switching to avoid the associated overhead. Also nodes varied their activity according to the available resources within the nodes, and the task demand levels the nodes were exposed to. The proposed architecture showed great flexibility and scalability. The modular design, and the distributed nature of the architecture allowed for fault tolerance and ease of configuration.

We also identified what type of applications would be well-suited to what model and under what general conditions. Notice that these were general rules and the algorithm/models may need to be adapted and used differently in each application or domain.

12.2 Contributions

Our contributions can be summarised as follows:

- The application of a threshold-based model on a micro-level to solve the problem of action selection in sensor networks on a global-level.
- The extension of the threshold-based model to account for network resources in making various decisions.
- Identifying the major points where adaptability can be beneficial if introduced in a sensor node.

- Providing a general node architecture where a sensor node is divided into modules and layers that map to adaptability points and control structures.
- Identifying the family of applications that may be well-suited to each of the models proposed in this dissertation.

12.3 Future Work

Our work was only a step further in the state of the art in the field of sensor networks. We tried to make the work as complete as possible, however, some questions stayed open, and new problems require further investigation. The following sections will discuss a few of the areas that we think are worth pursuing in the future.

12.3.1 Model Tuning

The models we experimented with, extended, and investigated had a fairly large number of parameters. The choice of the values of these parameters was performed in this dissertation based on trial-and-error, heuristics, and common sense methods. It would be worth automating this process by may be simulating the network for sometime before deployment or at design time. It even may be possible to find a way to allow these parameters to change in realtime by base stations or sensor nodes.

12.3.2 Scalability and Extreme Conditions

Although the models we experimented with in this dissertation are designed with scalability in mind, this has not been validated in all aspects. For example, all our experiments, for simplicity sake, were conducted with only three tasks. What if the number of tasks grow to hundreds?. Scalability in terms of the number of factors contributing to the decision making process needs to be tested as well. In this dissertation these were only three factors: 1) User policies (minima and maxima), 2) task demands, 3) resources available . Other scalability issues can be also looked at in the future.

12.3.3 Time Effects

Many of the parameters of the models we studied in this dissertation could be affected by time. For example the VRT specialisation level does not change over time. This means that nodes that have specialised in performing a particular task can never de-specialise over time, which may be beneficial to some applications, but not in some others.

Also the time effect on the various coefficients like the sensitivity, learning, and forgetting coefficients. Even the control processes could have a time dimension to be studied in the future.

12.3.4 Ontology

Mapping environmental phenomenon to a set of node variables was performed in this dissertation based on a one-to-one relationship. For example, the variables that represent the task stimulus were mapped to single phenomenon, specifically a pollution level in the environment. However, in real world, the situation could be more complex, and more involved mapping rules could be needed. For example, if a task was the general monitoring of an individual's health, and the task demand was represented by a single variable that maps to the individual's health. In this situation, a sensor cannot sample a person's health directly, but need instead to take a set of readings and perform some calculations to come up with an estimate of the health variable value.

12.3.5 Equilibrium and Saturation

We discussed the phenomenon of hyper-coverage, or coverage saturation occurring as a result of two many active nodes per neighbourhood. The optimal point before this phenomenon start dominating a network behaviour is called the equilibrium point. The issue that is worth looking at is how to find this point. Also how to quantify the coverage saturation, how to detect it, and how to control it.

12.3.6 Real-world Application Validation

Our dissertation investigated the threshold-based models in hypothetical scenarios and on a simulation framework, rather than on a real-world application with a real-world sensor nodes. In the future our results should be validated through a real scenario and a real hardware. The models may have to then be optimised to run on the real platform specifics.

Applying the concepts, algorithms, and models produced in this dissertation in a real-world application will require adding domain knowledge which is a research task on its own.

Finally, we re-iterate that adaptivity advantages sensor nodes, and insect societies are a great inspiration source. Simple adaptive models are easy to change, modify and observe, and therefore are predicted to be part of future control methodologies in large-scale multi-agent systems.

Appendix A

Definition of Simulation and Experiment Parameters and Settings

A.1 Experimental Simulation Parameter Values and Settings

Table A.1 defines the parameters and settings of all experiments and simulations we perform in this dissertation. A table of the values given to each of these will be provided in the specific experiment section.

Parameter	**Definition**
Node Count, N_0	The total number of nodes in the network initially, i.e. at time t_0 of an experiment. Nodes will at some point run out of battery, and so there will be dead nodes. These can never engage in any activity, be it sensing, communication, task engagement, or even discontinuation. Node Count at any moment after t_0 will be called N.
	continue on the next page ...

Table A.1: Simulation Model Parameters

Parameter	Definition
Neighbourhood Radius, r_0	used to calculate the area of a node's coverage, and it is the same area within which a node can communicate in our experiments. These two ranges take the same value following the idealised sensor network model where nodes are separated by distance r with each node covering the area within this radius from its centre. The ideal network uses multi-hop communication schemes to transfer data because long-haul communication is too expensive in terms of power. As we need to minimise the communication power consumption, we communicate the shortest distance, and this is r. Any less communication range will isolate the node as it mostly will fail to communicate with any peers, and any longer communication range will mean consuming energy more than needed for single-hop communication. This neighbourhood radius was set to 2 length units within the simulation environments we used to achieve a high node coverage value. We adjusted the density of the nodes to achieve sufficient connectivity.
Initial Energy Level, B_0	number of Joules in a node's battery at the beginning of an experiment. This will be the same for all network nodes in our experiments.
Threshold Model, M	can be one of the two models discussed in chapter 4, namely, FRT or VRT models.
Sampling Resolution, Δ	The number of active nodes per square area units required for a network to be perfectly reporting on a phenomenon when the stimulus for this phenomena, detected by the network nodes, equals the task-threshold θ associated with this phenomenon.
continue on the next page ...	

Table A.1: Simulation Model Parameters

Parameter	Definition
Minimum Response Rate, P_{r-min}	is the minimum rate at which an application associated with a phenomenon runs.
Maximum Response Rate, P_{r-max}	is the maximum rate an application associated with any task can run. This is, for example, set in the action selection experiments to 1.0. This means that each task-associated application can be run for 100% of the time, provided the node is not performing another task, and the node does not have a flat battery.
Minimum Sampling Rate, P_{s-min}	is the minimum rate at which a node samples the environment.
Maximum Sampling Rate, P_{s-max}	is the maximum rate at which a node samples the environment.
Minimum Discontinuation Rate, P_{d-min}	The minimum rate at which a node discontinues performing a task.
Maximum Discontinuation Rate, P_{d-max}	is the maximum rate at which a node discontinues performing a task.
Minimum Communication Rate, P_{c-min}	is the minimum rate at which a node communicates with peers within the neighbourhood radius, r_0
Maximum Communication Rate, P_{c-max}	is the maximum rate at which a node communicates with peers within the neighbourhood radius, r_0
Minimum Threshold, θ_{min}	The minimum valid value of a task-associated threshold. This is useful in the simulations of the VRT model where thresholds vary with time which could lead to overspecialisation. In overspecialisation, nodes could turn to be extremely sensitive to stimulus with respect to a certain task because its threshold is almost 0. Here, this value serves as a safeguard against overspecialisation of nodes by capping the maximum value the threshold could get to.
continue on the next page ...	

Table A.1: Simulation Model Parameters

Parameter	Definition
Maximum Threshold, θ_{max}	The maximum valid value of a task-associated threshold. This is useful in the simulations of the Variable Response Threshold (VRT) model where thresholds vary with time which could lead to overspecialisation. Overspecialisation means a node would never respond to stimulus with respect to a certain task because its threshold is too high. Here, this value serves as a safeguard against overspecialisation of nodes by capping the maximum value the threshold could get to.
Task A Threshold, θ_A	the threshold associated with task A, or how much variation in phenomenon A the node needs to observe before it runs T_A. This is measured by the unit of variation. The specific unit depends on the phenomenon being measured, and the precision required. For example, if wind speed is the phenomenon being monitored, a threshold for an associated task could be 50 meters per second. Therefore, if the wind speed changes from $3m/s$ to $53m/s$, the associated task will be probably performed.
Task B Threshold, θ_B	the threshold associated with task B, or how much variation in phenomenon B the node needs to observe before it runs T_B.
Task C Threshold, θ_C	the threshold associated with task C, or how much variation in phenomenon C the node needs to observe before it runs T_C.
Task A Demand, S_A	the task demand detected by the node with respect to task A.
Task B Demand, S_B	the task demand detected by the node with respect to task B.
continue on the next page ...	

Table A.1: Simulation Model Parameters

Parameter	Definition
Task C Demand, S_C	the task demand detected by the node with respect to task C.
Task A Energy Cost E_A**:**	is the energy consumed when performing T_A for one time step.
Task B Energy Cost E_B	is the energy consumed when performing T_B for one time step.
Task C Energy Cost E_C	is the energy consumption incurred from performing T_C for one time step.
Switching Tasks Energy Cost E_{sw}	is the energy required to switch from any Task T_i to any Task T_j where $i \neq j$. This was set for all experiments except those in 5.7 to the value 0 as these experiments are not meant to investigate switching issues, and a value of 0 would neutralise this variable.
Idle Energy Cost, E_i	the energy consumption rate per time step of a totally idle node, i.e. with no task-associated applications running. This was set in the simulations to 0.001 energy units per time step. Note that the ratios between the values in our simulation models were chosen to be similar to the real world values from real components. For example, idle energy consumption is considerably lower (in micro joules) than processing or sensing (usually in Milli joules) in real components (see 4.1, 4.2, and 4.3)and we made sure this is the case in our model experiments.
Communication Energy Cost, E_{comm}	the average cost of communicating a message between two nodes within each others' neighbourhood radius.
	continue on the next page ...

Table A.1: Simulation Model Parameters

Parameter	Definition
Sampling Energy Cost, E_s	The energy cost of making a sensor reading for any phenomenon by the node. Sensing different phenomena requires different power consumption rates, however, most sensor (see table 4.1), fall in close proximity of each other's power requirements, and to simplify our model, we assume sampling the environment will cost the same regardless of the type of sensor used. We have, as mentioned before, verified our results on J-Sim high-fidelity simulation platform, and found out that this does not have any significant impact on our results and findings.
Forgetting Coefficient, α	the increase in a task's threshold when a node performs a different task in the VRT model.
Learning Coefficient, β	the decrease in a task's threshold when a node performs this task.
Stimulus Decay Coefficient, d	the rate at which stimulus decays with time. Assuming at time t_0 a node detects a stimulus S_0, then the stimulus after $\Delta t = t - t_0$, where $t \geq t_0$, can be calculated by the equation: $$S(\Delta t) = (d)^{\Delta t} \times S_0$$
continue on the next page ...	

Table A.1: Simulation Model Parameters

Parameter	Definition
Sensitivity Coefficient, ρ	controls the sensitivity of a node towards a task-associated phenomenon. After a node samples the environment, it multiplies the data be this coefficient. If the value of ρ is very high, then the node will perceive any small stimulus as major, and will react. If $\rho \ll 1$, then the node will not respond but to extremely high stimulus, according to the following equation: $$S_{perceived} = S_{read} \times \rho$$
Minimum Task Demand, S_{min}	is the minimum task demand level that can be seen as valid. This works as a safeguard against erroneous and too extreme readings. Again, too small valid values of this parameters may complicate computational complexity of a sensor node or actually fail to be handled by the node's processors.
Maximum Task Demand, S_{max}	is the maximum task demand level that can be observed by any node. This works as a safeguard against erroneous and too extreme readings. This was set in the simulations to the value 100000 as some floating point calculations were intricate for values higher than this one.
Constant Response Probability, P_R	is a constant representing the rate at which a task-associated application would run if the node uses the CR model. For example, if the minimum response rate is 0.001, i.e. at least one evaluation happens every 1000 time steps, then the node will have a chance to run by a probability of P_R every 1000 time steps.
Constant Discontinuation Probability, P_D	is the minimum rate at which a task-associated application is discontinued.
continue on the next page ...	

Table A.1: Simulation Model Parameters

Parameter	Definition
Constant Sensing Probability, P_S	represents the rate of sensing phenomenon by a node.
Constant Communication Probability, P_{comm}	represents the rate of communicating data to neighbours. This is set in all simulations to a value of 0, i.e. there is no direct inter-node communication, except in chapter 11. Eliminating communication made it possible to assess how a network could perform collectively without any explicit communication amongst its member nodes. In addition, this reduces the complexity of investigating the effects of the response models on the sensor network. It may be very difficult to have a network without communication at all, however, the communication we target here is that which is needed for coordination and action selection. For example, this dissertation does not tackle the data dissemination related communication. Nodes could have long-haul communication equipment that is only used for data dissemination. Nodes could be designed to perform data dissemination when their batteries allow, for example in the summer if the nodes can charge through the sun light. Another example is networks that send their data to touring base stations.
Simulation Environment Area, A	a torus represented as a flat surface with an area of 10000 square area units (10x10 dimensions).
Side Length, L	the simulation terrain we run our experiments within is a torus visually represented as a two-dimensional square with side length equal to L.
continue on the next page ...	

Table A.1: Simulation Model Parameters

Parameter	Definition
Environmental Task Demand Distribution, $S_i(x, y, t)$	this refers to how the task demand is distributed over the environment area and varies across time. This , for the experiments of this chapter, is set to be *uniform*. This means that a node a at any point (x_a, y_a) on the environment surface at time t_a will detect a task demand of $S(x_a, y_a, t_a)$ that equals the task demand $S(x_b, y_b, t_b)$ detected by any node b at any other point (x_b, y_b) at any time t_b.
Resource Intensity, m	this is a parameter that controls the intensity of the responsiveness of a node towards a variation in available resources as seen in equation 4.4
Stimulus Intensity, n	this is a parameter that controls the intensity of the response to stimulus as seen in 4.1.
Stimulus-based Response Equation, $\Psi(S, \theta)$	a function used to calculate the probability a node will respond to a task with stimulus S and threshold S.
Resource-based Response Equation, $\Psi(B, B_0)$	a function used to calculate the probability a node will respond to a task when its battery has currently B joules left, and had B_0 at the outset.
Stimulus-based Discontinuation Equation, $\Omega(S, \theta)$	a function used to calculate the probability a node will discontinue a task it is currently performing given stimulus S and threshold θ.
Resource-based Discontinuation Equation, $\Omega(B, B_0)$	a function used to calculate the probability a node will discontinue performing a task it is currently performing given a current battery level of B and an initial battery level of B_0.
Stimulus-based Sampling Equation, $\Phi(S, \theta)$	a function used to calculate the probability a node will perform a sampling operation of a phenomenon given a stimulus S and threshold θ.
continue on the next page ...	

Table A.1: Simulation Model Parameters

Parameter	Definition
Resource-based Sampling Equation, $\Phi(B, B_0)$	a function used to calculate the probability a node will sample the environment with respect to task given a current battery level of B, and a maximum battery level of B_0
Stimulus-based Communication Equation, $\zeta(S, \theta)$	a function used to calculate the probability a node will perform communication within its neighbourhood radius with regard to a task given stimulus S and threshold θ
Resource-based Communication Equation, $\zeta(B, B_0)$	a function used to calculate the probability a node will communicate with its peers regarding a task given its current battery level B and the initial battery level of B_0

Table A.1: Simulation Model Parameters

Appendix B

Action Selection Simulations

B.1 Experimental Simulation Parameter Values and Settings for the CR, FRT, and FRT+B Experiment Sets

The following settings are specific to the experiments in this section:

Parameter	**Value for CR**	**Value for FRT**	**Value for FRT+B**
N_0 (nodes)	3000	3000	3000
r_0 (length units)	2	2	2
B_0 (power units)	5000	5000	5000
M	CR	FRT	FRT+B
Δ (node/square area units)	1	1	1
P_{r-max}	1.0	1.0	1.0
P_{r-min}	0.01	0.01	0.01
P_{s-min}	0.03	0.03	0.03
P_{s-max}	1.0	1.0	1.0
P_{c-min}	0.0	0.0	0.0
P_{c-max}	1.0	1.0	1.0
P_{d-min}	0.0	0.0	0.0
continue on the next page ...			

Table B.1: CR vs FRT Simulation Parameters

Parameter	Value for CR	Value for FRT	Value for FRT+B
P_{d-max}	1.0	1.0	1.0
θ_{min} (phenomenon variation units)	100.0	100.0	100.0
θ_{max} (phenomenon variation units)	10000.0	10000.0	10000.0
θ_A (phenomenon variation units)	100.0	100.0	100.0
θ_B (phenomenon variation units)	300.0	300.0	300.0
θ_C (phenomenon variation units)	1000.0	1000.0	1000.0
S_A (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_B (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_C (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
E_A (power units)	20.0	20.0	20.0
E_B (power units)	20.0	20.0	20.0
E_C (power units)	20.0	20.0	20.0
E_{sw} (power units)	0.0	0.0	0.0
E_i (power units)	0.001	0.001	0.001
E_{comm} (power units)	10.0	10.0	10.0
α (phenomenon variation units)	0	0	0
β (phenomenon variation units)	0	0	0

continue on the next page ...

Table B.1: CR vs FRT Simulation Parameters

Parameter	**Value for CR**	**Value for FRT**	**Value for FRT+B**
d	0.85	0.85	0.85
ρ	1.0	1.0	1.0
S_{min} (phenomenon variation units)	0.0	0.0	0.0
S_{max} (phenomenon variation units)	10000.0	10000.0	10000.0
P_R	$\Psi(S,\theta)$	$\Psi(S,\theta)$	$\Psi(S,\theta) \times \Psi(B,B_0)$
P_D	$\Omega(S,\theta)$	$\Omega(S,\theta)$	$\Omega(S,\theta)$
P_S	$\Phi(S,\theta)$	$\Phi(S,\theta)$	$\Phi(S,\theta)$
P_{comm}	$\zeta(S,\theta)$	$\zeta(S,\theta)$	$\zeta(S,\theta)$
A (square area units)	10000 (100x100)	10000 (100x100)	10000 (100x100)
L (length units)	100	100	100
$S(x,y,t)$	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0
m	1	1	1
n	1	1	1
$\Psi(S,\theta)$	0.0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$
$\Psi(B,B_0)$	unused	unused	$(\frac{B}{B_0})^m$
$\Omega(S,\theta)$	0.5	0.5	0.5
$\Omega(B,B_0)$	unused	unused	unused
$\Phi(S,\theta)$	0.5	0.5	0.5
$\Phi(B,B_0)$	unused	unused	unused
$\zeta(S,\theta)$	0.0	0.0	0.0
$\zeta(B,B_0)$	unused	unused	unused

Table B.1: CR vs FRT Simulation Parameters

B.2 Experimental Simulation Parameter Values and Settings for the CR, FRT, and FRT+B, VRT, VRT+B Experiment Sets

Table B.2 shows the settings of the two networks simulated in this section:

Parameter	**Value for CR**	**Value for FRT**	**Value for FRT+B**	**Value for VRT**	**Value for VRT+B**
N_0 (nodes)	3000	3000	3000	3000	3000
r_0 (length units)	2	2	2	2	2
B_0 (power units)	5000	5000	5000	5000	5000
M	CR	FRT	FRT+B	VRT	VRT+B
Δ (node/square area units)	1	1	1	1	1
P_{r-max}	1.0	1.0	1.0	1.0	1.0
P_{r-min}	0.01	0.01	0.01	0.01	0.01
P_{s-min}	0.03	0.03	0.03	0.03	0.03
P_{s-max}	1.0	1.0	1.0	1.0	1.0
P_{c-min}	0.0	0.0	0.0	0.0	0.0
P_{c-max}	1.0	1.0	1.0	1.0	1.0
P_{d-min}	0.0	0.0	0.0	0.0	0.0
P_{d-max}	1.0	1.0	1.0	1.0	1.0
θ_{min} (phenomenon variation units)	100.0	100.0	100.0	100.0	100.0

continue on the next page ...

Table B.2: CR, FRT, FRT+B, VRT, and VRT+B Simulation Parameters

Parameter	**Value for CR**	**Value for FRT**	**Value for FRT+B**	**Value for VRT**	**Value for VRT+B**
θ_{max} (phenomenon variation units)	10000.0	10000.0	10000.0	10000.0	10000.0
θ_A (phenomenon variation units)	100.0	100.0	100.0	100.0	100.0
θ_B (phenomenon variation units)	300.0	300.0	300.0	300.0	300.0
θ_C (phenomenon variation units)	1000.0	1000.0	1000.0	1000.0	1000.0
S_A (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_B (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_C (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$

continue on the next page ...

Table B.2: CR, FRT, FRT+B, VRT, and VRT+B Simulation Parameters

Parameter	**Value for CR**	**Value for FRT**	**Value for FRT+B**	**Value for VRT**	**Value for VRT+B**
E_A (power units)	20.0	20.0	20.0	20.0	20.0
E_B (power units)	20.0	20.0	20.0	20.0	20.0
E_C (power units)	20.0	20.0	20.0	20.0	20.0
E_{sw} (power units)	40.0	40.0	40.0	40.0	40.0
E_i (power units)	0.001	0.001	0.001	0.001	0.001
E_{comm} (power units)	10.0	10.0	10.0	10.0	10.0
α (phenomenon variation units)	100	100	100	100	100
β (phenomenon variation units)	100	100	100	100	100
d	0.85	0.85	0.85	0.85	0.85
ρ	1.0	1.0	1.0	1.0	1.0
S_{min} (phenomenon variation units)	0.0	0.0	0.0	0.0	0.0

continue on the next page ...

Table B.2: CR, FRT, FRT+B, VRT, and VRT+B Simulation Parameters

Parameter	**Value for CR**	**Value for FRT**	**Value for FRT+B**	**Value for VRT**	**Value for VRT+B**
S_{max} (phenomenon variation units)	10000.0	10000.0	10000.0	10000.0	10000.0
P_R	$\Psi(S,\theta)$	$\Psi(S,\theta)$	$\Psi(S,\theta)$	$\Psi(S,\theta)$	$\Psi(S,\theta)$
P_D	$\Omega(S,\theta)$	$\Omega(S,\theta)$	$\Omega(S,\theta)$	$\Omega(S,\theta)$	$\Omega(S,\theta)$
P_S	$\Phi(S,\theta)$	$\Phi(S,\theta)$	$\Phi(S,\theta)$	$\Phi(S,\theta)$	$\Phi(S,\theta)$
P_{comm}	$\zeta(S,\theta)$	$\zeta(S,\theta)$	$\zeta(S,\theta)$	$\zeta(S,\theta)$	$\zeta(S,\theta)$
A (square area units)	10000 (100x100)	10000 (100x100)	10000 (100x100)	10000 (100x100)	10000 (100x100)
L (length units)	100	100	100	100	100
$S(x,y,t)$	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0
m	1	1	1	1	1
n	1	1	1	1	1
$\Psi(S,\theta)$	0.0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$
$\Psi(B,B_0)$	unused	unused	$(\frac{B}{B_0})^m$	unused	$(\frac{B}{B_0})^m$
$\Omega(S,\theta)$	0.5	0.5	0.5	0.5	0.5
$\Omega(B,B_0)$	unused	unused	unused	unused	unused
$\Phi(S,\theta)$	0.5	0.5	0.5	0.5	0.5
$\Phi(B,B_0)$	unused	unused	unused	unused	unused
$\zeta(S,\theta)$	0.0	0.0	0.0	0.0	0.0
$\zeta(B,B_0)$	unused	unused	unused	unused	unused

Table B.2: CR, FRT, FRT+B, VRT, and VRT+B Simulation Parameters

Appendix C

Discontinuation Simulations

C.1 Experimental Simulation Parameter Values and Settings

Table C.1 provides the settings of the experiments in this section.

Parameter	**Value for CD**	**Value for FDT**	**Value for FDT+B**
N_0 (nodes)	3000	3000	3000
r_0 (length units)	2	2	2
B_0 (power units)	5000	5000	5000
M	CD	FDT	FDT+B
Δ (node/square area units)	1	1	1
P_{r-max}	1.0	1.0	1.0
P_{r-min}	0.01	0.01	0.01
P_{s-min}	0.03	0.03	0.03
P_{s-max}	1.0	1.0	1.0
P_{c-min}	0.0	0.0	0.0
P_{c-max}	1.0	1.0	1.0
P_{d-min}	0.0	0.0	0.0
continue on the next page ...			

Table C.1: CD, FDT, and FDT+B Parameters

Parameter	**Value for CD**	**Value for FDT**	**Value for FDT+B**
P_{d-max}	1.0	1.0	1.0
θ_{min} (phenomenon variation units)	100.0	100.0	100.0
θ_{max} (phenomenon variation units)	10000.0	10000.0	10000.0
θ_A (phenomenon variation units)	100.0	100.0	100.0
θ_B (phenomenon variation units)	300.0	300.0	300.0
θ_C (phenomenon variation units)	1000.0	1000.0	1000.0
S_A (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_B (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_C (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
E_A (power units)	20.0	20.0	20.0
E_B (power units)	20.0	20.0	20.0
E_C (power units)	20.0	20.0	20.0
E_{sw} (power units)	0.0	0.0	0.0
E_i (power units)	0.001	0.001	0.001
E_{comm} (power units)	10.0	10.0	10.0
α (phenomenon variation units)	100	100	100
β (phenomenon variation units)	100	100	100
d	0.7	0.7	0.7
ρ	1.0	1.0	1.0

continue on the next page ...

Table C.1: CD, FDT, and FDT+B Parameters

Parameter	**Value for CD**	**Value for FDT**	**Value for FDT+B**
S_{min} (phenomenon variation units)	0.0	0.0	0.0
S_{max} (phenomenon variation units)	10000.0	10000.0	10000.0
P_R	$\Psi(S,\theta)$	$\Psi(S,\theta)$	$\Psi(S,\theta)$
P_D	$\Omega(S,\theta)$	$\Omega(S,\theta)$	$\Omega(S,\theta)$
P_S	$\Phi(S,\theta)$	$\Phi(S,\theta)$	$\Phi(S,\theta)$
P_{comm}	$\zeta(S,\theta)$	$\zeta(S,\theta)$	$\zeta(S,\theta)$
A (square area units)	10000 (100x100)	10000 (100x100)	10000 (100x100)
L (length units)	100	100	100
$S(x,y,t)$	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0
m	1	1	1
n	1	1	1
$\Psi(S,\theta)$	0.5	0.5	0.5
$\Psi(B,B_0)$	unused	unused	unused
$\Omega(S,\theta)$	0.0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0	$1-\frac{S^n}{S^n+\theta^n}$	$1-\frac{S^n}{S^n+\theta^n}$
$\Omega(B,B_0)$	unused	unused	$1-(\frac{B}{B_0})^m$
$\Phi(S,\theta)$	0.5	0.5	0.5
$\Phi(B,B_0)$	unused	unused	unused
$\zeta(S,\theta)$	0.0	0.0	0.0
$\zeta(B,B_0)$	unused	unused	unused

Table C.1: CD, FDT, and FDT+B Parameters

Appendix D

Sampling Simulations

D.1 Experimental Simulation Parameter Values and Settings

Table D.1 provides the settings of the experiments in this section.

Parameter	**Value for CS**	**Value for FST**	**Value for FST+B**
N_0 (nodes)	3000	3000	3000
r_0 (length units)	2	2	2
B_0 (power units)	5000	5000	5000
M	CS	FST	FST+B
Δ (node/square area units)	1	1	1
P_{r-max}	1.0	1.0	1.0
P_{r-min}	0.01	0.01	0.01
P_{s-min}	0.03	0.03	0.03
P_{s-max}	1.0	1.0	1.0
P_{c-min}	0.0	0.0	0.0
P_{c-max}	1.0	1.0	1.0
P_{d-min}	0.0	0.0	0.0
continue on the next page ...			

Table D.1: CS, FST, and FST+B Parameters

Parameter	**Value for CS**	**Value for FST**	**Value for FST+B**
P_{d-max}	1.0	1.0	1.0
θ_{min} (phenomenon variation units)	100.0	100.0	100.0
θ_{max} (phenomenon variation units)	10000.0	10000.0	10000.0
θ_A (phenomenon variation units)	100.0	100.0	100.0
θ_B (phenomenon variation units)	300.0	300.0	300.0
θ_C (phenomenon variation units)	1000.0	1000.0	1000.0
S_A (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_B (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_C (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
E_A (power units)	20.0	20.0	20.0
E_B (power units)	20.0	20.0	20.0
E_C (power units)	20.0	20.0	20.0
E_{sw} (power units)	0.0	0.0	0.0
E_i (power units)	0.001	0.001	0.001
E_{comm} (power units)	10.0	10.0	10.0
α (phenomenon variation units)	100	100	100
β (phenomenon variation units)	100	100	100
d	0.85	0.85	0.85
ρ	1.0	1.0	1.0

continue on the next page ...

Table D.1: CS, FST, and FST+B Parameters

Parameter	**Value for CS**	**Value for FST**	**Value for FST+B**
S_{min} (phenomenon variation units)	0.0	0.0	0.0
S_{max} (phenomenon variation units)	10000.0	10000.0	10000.0
P_R	$\Psi(S,\theta)$	$\Psi(S,\theta)$	$\Psi(S,\theta)$
P_D	$\Omega(S,\theta)$	$\Omega(S,\theta)$	$\Omega(S,\theta)$
P_S	$\Phi(S,\theta)$	$\Phi(S,\theta)$	$\Phi(S,\theta)$
P_{comm}	$\zeta(S,\theta)$	$\zeta(S,\theta)$	$\zeta(S,\theta)$
A (square area units)	10000 (100x100)	10000 (100x100)	10000 (100x100)
L (length units)	100	100	100
$S(x,y,t)$	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0
m	1	1	1
n	1	1	1
$\Psi(S,\theta)$	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$
$\Psi(B,B_0)$	unused	unused	unused
$\Omega(S,\theta)$	0.5	0.5	0.5
$\Omega(B,B_0)$	unused	unused	unused
$\Phi(S,\theta)$	0.1, 0.5, 0.9	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$
$\Phi(B,B_0)$	unused	unused	$(\frac{B}{B_0})^m$
$\zeta(S,\theta)$	0.0	0.0	0.0
$\zeta(B,B_0)$	unused	unused	unused

Table D.1: CS, FST, and FST+B Parameters

Appendix E

Communication Simulations

E.1 Experimental Simulation Parameter Values and Settings

Table E.1 provides the settings of the experiments in this section.

Parameter	**Value for CC**	**Value for FCT**	**Value for FCT+B**
N_0 (nodes)	3000	3000	3000
r_0 (length units)	2	2	2
B_0 (power units)	5000	5000	5000
M	CC	FCT	FCT+B
Δ (node/square area units)	1	1	1
P_{r-max}	1.0	1.0	1.0
P_{r-min}	0.01	0.01	0.01
P_{s-min}	0.03	0.03	0.03
P_{s-max}	1.0	1.0	1.0
P_{c-min}	0.0	0.0	0.0
P_{c-max}	1.0	1.0	1.0
P_{d-min}	0.0	0.0	0.0
continue on the next page ...			

Table E.1: CC, FCT, and FCT+B Parameters

Parameter	**Value for CC**	**Value for FCT**	**Value for FCT+B**
P_{d-max}	1.0	1.0	1.0
θ_{min} (phenomenon variation units)	100.0	100.0	100.0
θ_{max} (phenomenon variation units)	10000.0	10000.0	10000.0
θ_A (phenomenon variation units)	100.0	100.0	100.0
θ_B (phenomenon variation units)	300.0	300.0	300.0
θ_C (phenomenon variation units)	1000.0	1000.0	1000.0
S_A (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_B (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
S_C (phenomenon variation units)	$S(x, y, t)$	$S(x, y, t)$	$S(x, y, t)$
E_A (power units)	20.0	20.0	20.0
E_B (power units)	20.0	20.0	20.0
E_C (power units)	20.0	20.0	20.0
E_{sw} (power units)	0.0	0.0	0.0
E_i (power units)	0.001	0.001	0.001
E_{comm} (power units)	10.0	10.0	10.0
α (phenomenon variation units)	100	100	100
β (phenomenon variation units)	100	100	100
d	0.85	0.85	0.85
ρ	1.0	1.0	1.0

continue on the next page ...

Table E.1: CC, FCT, and FCT+B Parameters

Parameter	**Value for CC**	**Value for FCT**	**Value for FCT+B**
S_{min} (phenomenon variation units)	0.0	0.0	0.0
S_{max} (phenomenon variation units)	10000.0	10000.0	10000.0
P_R	$\Psi(S,\theta)$	$\Psi(S,\theta)$	$\Psi(S,\theta)$
P_D	$\Omega(S,\theta)$	$\Omega(S,\theta)$	$\Omega(S,\theta)$
P_S	$\Phi(S,\theta)$	$\Phi(S,\theta)$	$\Phi(S,\theta)$
P_{comm}	$\zeta(S,\theta)$	$\zeta(S,\theta)$	$\zeta(S,\theta)$
A (square area units)	10000 (100x100)	10000 (100x100)	10000 (100x100)
L (length units)	100	100	100
$S(x,y,t)$	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0	0.1, 1.0 , 10.0, 100.0, 1000.0, 10000.0
m	1	1	1
n	1	1	1
$\Psi(S,\theta)$	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$
$\Psi(B,B_0)$	unused	unused	unused
$\Omega(S,\theta)$	0.5	0.5	0.5
$\Omega(B,B_0)$	unused	unused	unused
$\Phi(S,\theta)$	0.5	0.5	0.5
$\Phi(B,B_0)$	unused	unused	unused
$\zeta(S,\theta)$	0.1, 0.5, 0.9	$\frac{S^n}{S^n+\theta^n}$	$\frac{S^n}{S^n+\theta^n}$
$\zeta(B,B_0)$	unused	unused	$1-(\frac{B}{B_0})^m$

Table E.1: CC, FCT, and FCT+B Parameters

Bibliography

[1] Scott Aaronson. Guest column: Np-complete problems and physical reality. *SIGACT News*, 36(1):30–52, 2005.

[2] Sheikh I. Ahamed, Avinash Vyas, and Mohammad Zulkernine. Towards developing sensor networks monitoring as a middleware service. In *ICPPW '04: Proceedings of the 2004 International Conference on Parallel Processing Workshops*, pages 465–471, Washington, DC, USA, 2004. IEEE Computer Society.

[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Commununication Magazine.*, 40(8):102–114, 2002.

[4] J. N. Al-Karaki and A. E. Kamal. Routing techniques in wireless sensor networks: A survey. *Wireless Communications, IEEE*, 11(6):6–8, December 2004.

[5] III Albert F. Harris, Milica Stojanovic, and Michele Zorzi. When underwater acoustic nodes should sleep with one eye open: idle-time power management in underwater sensor networks. In *WUWNet '06: Proceedings of the 1st ACM international workshop on Underwater networks*, pages 105–108, New York, NY, USA, 2006. ACM.

[6] Zigbee Alliance. Zigbee specifications. Last updated December 2006, December 2006.

[7] Anish Anthony. *A comparison of agent paradigms for resource management in distributed sensor networks*. PhD thesis, Birmingham, AL, USA, 2007. Adviser-Thomas C. Jannett.

[8] Ronald C. Arkin and Tucker R. Balch. Aura: Principles and practice in review. *Journal of Experimental and Theoretical Artificial Intelligence(JETAI)*, Volume 9(Number 2/3):175–188, April 1997.

[9] G. Asada, T. Dong, F. Lin, G. Pottie, W. Kaiser, and H. Marcy. Wireless integrated network sensors: Low power systems on a chip. European Solid State Circuits Conference, october 1998.

[10] Nauman Aslam, William Robertson, William Phillips, and Shyamala C. Sivakumar. Relay node selection in randomly deployed homogeneous clustered wireless sensor networks. In *SENSORCOMM '07: Proceedings of the 2007 International Conference on Sensor Technologies and Applications*, pages 418–423, Washington, DC, USA, 2007. IEEE Computer Society.

[11] Sasikanth Avancha. *A Holistic Approach to Secure Sensor Networks*. PhD thesis, University of Maryland, 2005.

[12] K. Baert, B. Gyselinckx, T. Torfs, V. Leonov, F. Yazicioglu, S. Brebels, S. Donnay, J. Vanfleteren, E. Beyne, and C. Van Hoof. Technologies for highly miniaturized autonomous sensor networks. *Microelectron. J.*, 37(12):1563–1568, 2006.

[13] Khashayar R. Baghaei and Arvin Agah. Task allocation methodologies for multi-robot systems. Technical Report ITTC-FY2003-TR-20272-01, Lawrence, Kansas 66045, November 2002.

[14] Amol B. Bakshi, Jingzhao Ou, and Viktor K. Prasanna. An integrated design environment to evaluate power/performance tradeoffs for sensor network applications. In *Sixth Annual Workshop on High Performance Embedded Computing (HPEC)*, September 2002.

[15] Philip Ball. *The Self-made Tapestry: Pattern Formation in Nature*. Oxford University Press, July 2001.

[16] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli. Wireless sensor networks for environmental monitoring: The sensorscope experi-

ence. In *The 20th IEEE International Zurich Seminar on Communications (IZS 2008)*, Zurich, Switzerland, March 2008. IEEE.

[17] P. Basu and J. Redi. Effect of overhearing transmissions on energy efficiency in dense sensor networks. Proceedings of Information Processing in Sensor Networks (IPSN 2004), Berkeley, California, April 2004.

[18] K. Baumgartner and S. Robert. Architecture of a scalable wireless sensor network for pollution monitoring. In *European Workshop on Wireless Sensor Networks*, february 2006.

[19] Michael Bekerman, Avi Mendelson, and Gad Sheaffer. Performance and hardware complexity tradeoffs in designing multithreaded architectures. *pact*, 00:0024, 1996.

[20] Paolo Bellavista, Antonio Corradi, Rebecca Montanari, and Cesare Stefanelli. Context-aware middleware for resource management in the wireless internet. *IEEE Transactions on Software Engineering*, 29(12):1086–1099, 2003.

[21] Bonnie Berger and Tom Leighton. Protein folding in the hydrophobic-hydrophilic (hp) is np-complete. In *RECOMB '98: Proceedings of the second annual international conference on Computational molecular biology*, pages 30–39, New York, NY, USA, 1998. ACM.

[22] E. Bonabeau, G. Theraulaz, and J.-L. Deneubourg. Phase diagram of a model of self-organizing hierarchies. *Physica A Statistical Mechanics and its Applications*, 217:373–392, feb 1995.

[23] Eric Bonabeau, S. Guerin, D. Snyers, P. Kuntz, and G. Theraulaz. Three-dimensional architectures grown by simple stigmergic agents. *Biosystems*, 56(1):13–32, March 2000.

[24] Eric Bonabeau, Andrej Sobkowski, Guy Theraulaz, and Jean-Luis Deneubourg. Adaptive task allocation inspired by a model of division of labor in social insects. In Dan Lundh, Bjorn Olsson, and Ajit

Narayanan, editors, *Biocomputing and Emergent Computation*, pages 36–45. World Scientific, 1997.

[25] Eric Bonabeau, Guy Theraulaz, and Jean-Louis Deneubourg. Fixed response thresholds and the regulation of division of labor in insect societies. *Bulletin of Mathematical Biology*, (60):753–807, March 1998. Article No. bu980041.

[26] Peter A. Boncz, Angela Bonifati, Joos-Hendrik Bse, Stefan Bttcher, Panos K. Chrysanthis, Le Gruenwald, Arantza Illarramendi, Peter Janacik, Birgitta Knig-Ries, Wolfgang May, Anirban Mondal, Sebastian Obermeier, Aris M. Ouksel, and George Samaras. 06431 working group summary: P2p, ad hoc and sensor networks - all the different or all the same?. In Stefan Bttcher, Le Gruenwald, Pedro Jos Marrn, and Evaggelia Pitoura, editors, *Scalable Data Management in Evolving Networks*, volume 06431 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2006.

[27] Yann-Ael Le Borgne, Mehdi Moussaid, and Gianluca Bontempi. Simulation architecture for data processing algorithms in wireless sensor networks. In *AINA '06: Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 2 (AINA'06)*, pages 383–387, Washington, DC, USA, 2006. IEEE Computer Society.

[28] Azzedine Boukerche, Richard Werner Nelem Pazzi, and Regina Borges Araujo. Fault-tolerant wireless sensor network routing protocols for the supervision of context-aware physical environments. *J. Parallel Distrib. Comput.*, 66(4):586–599, 2006.

[29] M. Britton, V. Shum, L. Sacks, and H. Haddadi. A biologically-inspired approach to designing wireless sensor networks. In *Proceedings of EWSN 2005 : 2nd European Workshop on Wireless Sensor Networks*, pages 256–266, Istanbul, Turkey, 31 January–2 February 2005. IEEE Computer Society.

[30] S.F. Bush. Low-energy sensor network time synchronization as an emergent property. In *Proceedings of 14th International Conference on Computer Communications and Networks, 2005. ICCCN 2005*, pages 93–98, October 2005.

[31] Vladimir Byckovskiy, Seaphan Megerian, Deborah Estrin, and Miodrag Potkonjak. A collaborative approach to in-place sensor calibration. In *Proceedings of the Second International Workshop on Information Processing in Sensor Networks (IPSN)*, volume 2634 of *Lecture Notes in Computer Science*, pages 301–316. Springer–Verlag Berlin Heidelberg, 2003.

[32] Scott Camazine, Jean-Louis Deneubourg, Nigel R. Franks, James Sneyd, Guy Theraulaz, and Eric Bonabeau. *Self-Organization in Biological Systems*. Princeton Studies in Complexity. Princeton University, 2 edition, 2003.

[33] Mike Campos, Eric Bonabeau, Guy Theraulaz, and Jean-Louis Deneubourg. Dynamic Scheduling and Division of Labor in Social Insects. *Adaptive Behavior*, 8(2):83–95, 2000.

[34] Q. Cao, T. Yan, T. Abdelzaher, and J. Stankovic. Analysis of target detection performance for wireless sensor networks. In *In Proceedings of the International Conference on Distributed Computing in Sensor Networks*, CA, June 2005.

[35] Gianni Di Caro and Marco Dorigo. Antnet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9:317–365, 1998.

[36] Leandro Nunes De Castro. *Fundamentals of Natural Computing : Basic Concepts, Algorithms, and Applications*. Chapman & Hall/CRC, 2006.

[37] E. Catterall, K. Van Laerhoven, and M. Strohbach. Selforganization in ad hoc sensor networks: An empirical study. In *Proceedings of Alife VIII: the 8th International Conference on the Simulation and Synthesis of Living Systems*, Sydney, Australia, 2002. MIT Press.

[38] Ioannis Chatzigiannakis, Athanasios Kinalis, and Sotiris Nikoletseas. Power conservation schemes for energy efficient data propagation in heterogeneous wireless sensor networks. In *ANSS '05: Proceedings of the 38th annual Symposium on Simulation*, pages 60–71, Washington, DC, USA, 2005. IEEE Computer Society.

[39] Yuen Hui Chee, Jan M. Rabaey, and Ali Niknejad. *Ultra Low Power Transmitters for Wireless Sensor Networks*. PhD thesis, EECS Department, University of California, Berkeley, May 2006.

[40] Maggie X. Cheng, Lu Ruan, and Weili Wu. Coverage breach problems in bandwidth-constrained sensor networks. *ACM Trans. Sen. Netw.*, 3(2):12, 2007.

[41] Thomas Clouqueur, Kewal K. Saluja, and Parameswaran Ramanathan. Fault tolerance in collaborative sensor networks for target detection. *IEEE Trans. Comput.*, 53(3):320–333, 2004.

[42] Sinem Coleri, Sing Yiu Cheung, and Pravin Varaiya. Sensor networks for monitoring traffic. In *Proceedings of the Forty-Second Annual Allerton Conference on Commuinication, Control, and Computing*, Univeristy of Illinois, September 2004.

[43] Michael Conrad and Klaus-Peter Zauner. Molecular computing: From conformational pattern recognition to complex processing networks. In *Selected papers from the German Conference on Bioinformatics*, pages 1–10, London, UK, 1997. Springer-Verlag.

[44] Sunwoo Jung Contact, Jaehyun Choi, Dongkyu Kim, and Kiwon Chong. *A Node Management Tool for Dynamic Reconfiguration of Application Modules in Sensor Networks*, volume 4238 of *Lecture Notes in Computer Science, Proceedings of the 9th Asia-Pacific Network Operations and Management Symposium, APNOMS 2006*, book Management of Convergence Networks and Services, pages 570–573. Springer Berlin / Heidelberg, Busan, Korea, September 2006.

[45] James T. Costa, Edward O. Wilson, and Bert Holldobler. *The Other Insect Societies*. Harvard University Press, first edition, September 2006.

[46] Michael J. Covington, Prahlad Fogla, Zhiyuan Zhan, and Mustaque Ahamad. A context-aware security architecture for emerging applications. 18th Annual Computer Security Applications Conference, Las Vegas, Nevada, December 9–13 2002.

[47] C. Crane et al. Development of an integrated sensor system for obstacle detection and terrain evaluation for application to unmanned ground vehicles. In *Proceedings of SPIE Defense and Security Symposium*, Orlando, USA, March 2005. ACM/IEEE.

[48] David Culler, Deborah Estrin, and Mani Srivastava. Overview of sensor networks. *Special Issue in Sensor Networks*, 37(8):41–49, August 2004.

[49] Torbjorn S. Dahl, Maja J Mataric, and Gaurav S. Sukhatme. Emergent robot differentiation for distributed multi-robot task allocation. In *Distributed Autonomous Robotic Systems (DARS04)*, Proceedings of the 7th International Symposium on Distributed Autonomous Robotic Systems (DARS04), pages 191–200, Toulouse, France, June 2004.

[50] Torbjrn S. Dahl, Maja J Mataric, and Gaurav S. Sukhatme. A machine learning method for improving task allocation in distributed multi-robot transportation. In Dan Braha, Ali Minai, and Yaneer Bar-Yam, editors, *Complex Engineering Systems*, Reading, Massachusetts, 2004. Perseus Books.

[51] Jim Jennings Daniela Rus, Bruce Donald. Moving furniture with teams of autonomous robots. In *International Conference on Intelligent Robots and Systems(IROS)*, volume 1, page 235, 1995.

[52] Denise de Oliveira, Paulo R. Ferreira, Jr., and Ana L. C. Bazzan. A swarm based approach for task allocation in dynamic agents organizations. In *Proceedings of the Third International Joint Conference on*

Autonomous Agents and Multiagent Systems - (AAMAS'04), volume 3, pages 1252–1253, November 2004.

[53] Mathijs M. de Weerdt, Yingqian Zhang, and Tomas Klos. Distributed task allocation in social networks. In Michael Huhns and Onn Shehory, editors, *Proceedings of the sixth International Conference on Autonomous Agents and Multiagent Systems*, pages 488–495. IFAAMAS, IFAAMAS, 2007.

[54] Budhaditya Deb. *Algorithms for resource utilization in sensor networks*. PhD thesis, New Brunswick, NJ, USA, 2005. Director-Badri Nath.

[55] Thu-Thuy Do, Daeyoung Kim, Tomas Sanchez Lopez, Hyunhak Kim, Seongki Hong, Minh-Long Pham, Kwangyong Lee, and Seongmin Park. An evolvable operating system for wireless sensor networks. *International Journal of Software Engineering and Knowledge Engineering (IJSEKE)*, 15(2):265–270, 2005.

[56] S.O. Dulman, L.F.W. van Hoesel, P.J.M. Havinga, and P.G. Jansen. Data centric architecture for wireless sensor networks. In *Proceedings of ProRisc 2003*, The Netherlands, November 2003.

[57] Virantha Ekanayake, IV Clinton Kelly, and Rajit Manohar. An ultra low-power processor for sensor networks. *SIGARCH Comput. Archit. News*, 32(5):27–36, 2004.

[58] Tamer ElBatt. On the scalability of hierarchical cooperation for dense sensor networks. In *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*, pages 387–293, New York, NY, USA, 2004. ACM.

[59] Jakob Eriksson, Lewis Girod, Bret Hull, Ryan Newton, Hari Balakrishnan, and Samuel Madden. The pothole patrol: Using a mobile sensor network for road surface monitoring. In *Proceedings of MobiSys, 2008*, 2008.

[60] D. Estrin, L. Girod, G. Pottie, and M. Srivastava. Instrumenting the world with wireless sensor networks. In *Proceedings of IEEE Inter-*

national Conference on Acoustics, Speech, and Signal Processing, 2001 (ICASSP'01), volume 4, pages 2033–2036, Salt Lake City, Utah, 2001. IEEE.

[61] Deborah Estrin, Ramesh Govindan, John Heidemann, and Satish Kumar. Next century challenges: scalable coordination in sensor networks. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270, New York, NY, USA, 1999. ACM.

[62] Alessandro Farinelli, Giorgio Grisetti, Luca Iocchi, and Daniele Nardi. Coordination in dynamic environments with constraints on resources. In *IROS Workshop on Cooperative Robotics*, 2002.

[63] Luca Filipponi, Silvia Santini, and Andrea Vitaletti. Data collection in wireless sensor networks for noise pollution monitoring. In *Proceedings of the 4th IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS'08)*, Santorini Island, Greece, jun 2008.

[64] A. Fitzpatrick, G. Biegel, S. Clarke, and V. Cahill. Towards a sentient object model. Workshop on Engineering Context-Aware Object-Oriented Systems and Environments (ECOOSE), Seattle, WA, USA, November 2002.

[65] Terence C. Fogarty. Varying the probability of mutation in the genetic algorithm. In *Proceedings of the third international conference on Genetic algorithms*, pages 104–109, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[66] Stephanie Forrest, editor. *Emergent computation*. MIT Press, Cambridge, MA, USA, 1991.

[67] Aaron Gage. *Multi-Robot Task Allocation Using Affect*. PhD thesis, University of South Florida, August 2004.

[68] Aaron Gage. *Multi-Robot Task Allocation Using Affect*. PhD thesis, Department of Computer Science and Engineering, College of Engineering, University of South Florida, August 2004.

[69] Aram Galstyan and Kristina Lerman. Analysis of a stochastic model of adaptive task allocation in robots. In *Proceedings of Autonomous Agents and Multi-agent System Conference (AAMAS-04)*, 2004.

[70] Deepak Ganesan, Alberto Cerpa, Wei Ye, Yan Yu, Jerry Zhao, and Deborah Estrin. Networking issues in wireless sensor networks. *Journal of Parallel and Distributed Computing (JPDC), Special issue on Computing and Communication in Distributed Sensor Networks, Elsevier Publishers*, 64(7):799–814, July 2004.

[71] Deepak Ganesan, Ben Greenstein, Denis Perelyubskiy, Deborah Estrin, and John Heidemann. An evaluation of multi-resolution search and storage in resource-constrained sensor networks. In *In Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[72] David Gay, Philip Levis, and David Culler. Software design patterns for tinyos. *Trans. on Embedded Computing Sys.*, 6(4):22, 2007.

[73] Brian P. Gerkey and Maja J Mataric. Murdoch: Publish/subscribe task allocation for heterogeneous agents. In *Proceedings of Autonomous Agents*, pages 203–204, Barcelona, Spain, June 2000.

[74] Brian P. Gerkey and Maja J Mataric. Sold!: Market methods for multi-robot control. Technical Report IRIS-01-399, Los Angeles, California, March 2001.

[75] Brian P. Gerkey and Maja J Mataric. A market-based formulation of sensor-actuator network coordination. In *Proceedings of the AAAI Spring Symposium on Intelligent Embedded and Distributed Systems*, pages 21–26, Palo Alto, California, March 2002.

[76] Brian P. Gerkey and Maja J. Mataric. Sold!: Auction methods for multirobot coordination. *IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION*, 18(5):758–768, October 2002.

[77] M Ghanem, Y Guo, J Hassard, M Osmond, and R Richards. Sensor grids for air pollution monitoring. In *Proceedings of UK e-Science All Hands Meeting 2004*, Nottingham UK, september 2004.

[78] 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 Trans. Sen. Netw.*, 3(3):13, 2007.

[79] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[80] A. Gonzalez, I. Marshall, and L. Sacks. A self-synchronised scheme for automated communication in wireless sensor networks. In *Proceedings of the Intelligent Sensors, Sensor Networks and Information Processing Conference, 2004*, pages 97–102, December 2004.

[81] Ben Greenstein, Christopher Mar, Alex Pesterev, Shahin Farshchi, Eddie Kohler, Jack Judy, and Deborah Estrin. Capturing high-frequency phenomena using a bandwidth-limited sensor network. In *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 279–292, New York, NY, USA, 2006. ACM.

[82] Hadeli, Paul Valckenaers, Martin Kollingbaum, and Hendrik Van Brussel. Multi-agent coordination and control using stigmergy. *Comput. Ind.*, 53(1):75–96, 2004.

[83] Robrecht Haesevoets, Bart Van Eylen, Daniel Weyns, Alexander Helleboogh, Tom Holvoet, and Wouter Joosen. Context-driven dynamic organizations applied to coordinated monitoring of traffic jams. *Engineering Environment-Mediated Multiagent Systems*, pages 126–143, 2007.

[84] V. Handziski, H. Karl, A. Köpke, and A. Wolisz. A common wireless sensor network architecture? In H. Karl, editor, *Proc. 1. GI/ITG Fachgespräch "Sensornetze" (Technical Report TKN-03-012 of the Telecommunications Networks Group, Technische Universität Berlin*, pages 10–17, Berlin, Germany, jul 2003.

[85] Jason Hill and David Culler. A wireless-embedded architecture for system level optimization. Technical report, University of California, Berkeley, January 2001.

[86] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David E. Culler, and Kristofer S. J. Pister. System architecture directions for networked sensors. In *Architectural Support for Programming Languages and Operating Systems*, pages 93–104, 2000.

[87] Jason Lester Hill. *System Architecture for Wireless Sensor Networks*. PhD thesis, UNIVERISY OF CALIFORNIA, BERKELEY, Spring 2003.

[88] Barbara Hohlt, Lance Doherty, and Eric Brewer. Flexible power scheduling for sensor networks. In *Proceedings of the IEEE and ACM Third International Symposium on Information Processing in Sensor Networks (IPSN 04)*, April 2004.

[89] J Horgan. From complexity to perplexity. *Scientific American*, 272:74–79, 1995.

[90] Chih-fan Hsin and Mingyan Liu. Network coverage using low duty-cycled sensors: Random & coordinated sleep algorithms. In *Proceedings of IPSN'04*, Berkeley, California, USA., April 2004. ACM.

[91] Chi-Fu Huang and Yu-Chee Tseng. The coverage problem in awireless sensor network. *ACM Mobile Networks and Applications (MONET), special issue on Wireless Sensor Networks*, 10(4):519–528, August 2005.

[92] Qing Huang, Yong Bai, and Lan Chen. Som-based dynamic power management (sdpm) framework for wireless sensor networks. In *IWCMC '06: Proceedings of the 2006 international conference on Wireless*

communications and mobile computing, pages 1079–1084, New York, NY, USA, 2006. ACM.

[93] T. Iwao, K. Nomura, J. Pitt, and M. Amamiya. A control model of multi-purpose sensor networks by policies. In *Proceedings of the 2005 International Conference on Active Media Technology, 2005. (AMT 2005)*, pages 429–434, May 2005.

[94] Chris Jones and Maja J Mataric. Adaptive division of labor in large-scale minimalist multi-robot systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1969–1974, Las Vegas, Nevada, October 2003.

[95] Chris Jones and Maja J Mataric. The use of internal state in multi-robot coordination. In *Proceedings of the Hawaii International Conference on Computer Sciences*, pages 27–32, Waikiki, Hawaii, January 2004.

[96] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *ASPLOS, San Jose, CA*, october 2002.

[97] I. Khemapech, I. Duncan, and A. Miller. A survey of wireless sensor networks technology. In M. Merabti and R. Pereira, editors, *PGNET*, Proceedings of the 6th Annual PostGraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting, Liverpool, UK, June 2005. EPSRC.

[98] Dong Seong Kim, Mohammed Golam Sadi, and Jong Sou Park. *A Key Revocation Scheme for Mobile Sensor Networks*, volume 4743 of *Lecture Notes in Computer Science, Frontiers of High Performance Computing and Networking ISPA 2007 Workshops*, book 5, pages 41–49. Springer Berlin / Heidelberg, Niagara Falls, Canada, August 2006.

[99] JongGyu Kim, John Paul M. Torregoza, InYeup Kong, and WonJoo Hwang. Photovoltaic cell battery model for wireless sensor networks. *IJCSNS International Journal of Computer Science and Network Security*, 6(9B):90–97, September 2006.

[100] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing,. *Science*, 220(4598):671–680, 1983.

[101] John R. Koza, Martin A. Keane, Jessen Yu, III Forrest H. Bennett, and William Mydlowec. Automatic creation of human-competitive programs and controllers by means of genetic programming. *Genetic Programming and Evolvable Machines*, 1(1-2):121–164, 2000.

[102] Michael J. B. Krieger, Jean-Bernard Billeter, and Laurent Keller. Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406:992–995, 2000.

[103] Chandra Krintz, Ye Wen, and Rich Wolski. Application-level prediction of battery dissipation. In *ISLPED '04: Proceedings of the 2004 international symposium on Low power electronics and design*, pages 224–229, New York, NY, USA, 2004. ACM.

[104] Alexander Kröller, Dennis Pfisterer, Carsten Buschmann, Sándor P. Fekete, and Stefan Fischer. Shawn: A new approach to simulating wireless sensor networks. In *Proceedings of the 3rd Symposium on Design, Analysis, and Simulation of Distributed Systems (DASD'05)*, pages 117–124, 2005.

[105] S. Kumar, T.H. Lai, and A. Arora. Barrier coverage with wireless sensors. In *Proceedings of the 11th Annual International Conference on Mobile Computing and Networking (MoBiCom 05)*, pages 284–298, 2005.

[106] Mauri Kuorilehto, Marko Hnnikinen, and Timo D. Hmlinen. A survey of application distribution in wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2005(5):774–788, 2005. doi:10.1155/WCN.2005.774.

[107] Thomas H. Labella, Marco Dorigo, and Jean-Louis Deneubourg. Self-organised task allocation in a group of robots. Proceedings of the 7th International Symposium on Distributed Autonomous Robotic Systems (DARS04), Toulouse, France, June 23–25 2004. R. Alami.

[108] M. Laibowitz and J. A. Paradiso. Parasitic mobility for pervasive sensor networks. In H. W. Gellersen, R. Want, and A. Schmidt, editors, *Proceedings of the Third International Conference, PERVASIVE 2005*, pages 255–278, Munich, Germany, May 2005. Springer-Verlag.

[109] Olaf Landsiedel, Klaus Wehrle, and Stefan Gtz. Accurate prediction of power consumption in sensor networks. In *Proceedings of The Second IEEE Workshop on Embedded Networked Sensors (EmNetS-II)*, Sydney, Australia, May 2005.

[110] C. G. Langton. Artificial life. In *Artificial Life: Proceedings of the Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems*, Santa Fe Institute Studies in the Sciences of Complexity, pages pages 1–47, Los Alamos, NM, September 1987. Addison-Wesley.

[111] Loukas Lazos, Radha Poovendran, and James A. Ritcey. On the deployment of heterogeneous sensor networks for detection of mobile targets. In *Proceedings of the 5th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007*, pages 1–10, Limassol, Cyprus, April 2007.

[112] Uichin Lee, Biao Zhou, Mario Gerla, Eugenio Magistretti, Paolo Bellavista, and Antonio Corradi. Mobeyes: smart mobs for urban monitoring with a vehicular sensor network. *Wireless Communications, IEEE (also IEEE Personal Communications)*, 13(5):52–57, October 2006.

[113] C. Li and K. Sycara. A stable and efficient scheme for task allocation via agent coalition formation. *Algrithms for Cooperative Systems, World Scientific*, 2004.

[114] Chengfa Li, Mao Ye, Guihai Chen, and Jie Wu. An energy-efficient unequal clustering mechanism for wireless sensor networks. In *Proc. of the 2nd IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*. IEEE, November 2005.

[115] Yingshu Li and My T Thai. *Wireless Sensor Networks and Applications*. Signals and Communication Technology. Springer, 2008.

[116] Wei Liang, Haibin Yu, Peng Zeng, and Chang Che. Besm: A balancing energy-aware sensor management protocol for wireless sensor network. *International Journal of Information Technology*, 12(4), 2006.

[117] Stehpanie Lindsay, C. S. Raghavendra, and Krishna M. Sivalingam. Data gathering in sensor networks using the energy delay metric. In *IPDPS '01: Proceedings of the 15th International Parallel & Distributed Processing Symposium*, page 188, Washington, DC, USA, 2001. IEEE Computer Society.

[118] B. Liu and D. Towsley. On the coverage and detectability of large-scale wireless sensor networks. In *Proc. of the Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks Conference (WiOpt). (2003)*, 2003.

[119] Donggang Liu and Peng Ning. Improving key predistribution with deployment knowledge in static sensor networks. *ACM Trans. Sen. Netw.*, 1(2):204–239, 2005.

[120] Zhengjun Liu, Aixia Liu, Changyao Wang, and Zheng Niu. Evolving neural network using real coded genetic algorithm (ga) for multispectral image classification. *Future Generation Computer Systems*, 20(7):1119–1129, October 2004.

[121] Kian Hsiang Low, Wee Kheng Leow, and Marcelo H. Ang, Jr. Task allocation via self-organizing swarm coalitions in distributed mobile sensor network. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-04)*, pages 28–33, San Jose, CA, July 2004.

[122] Kian Hsiang Low, Wee Kheng Leow, and Jr. Marcelo H. Ang. Autonomic mobile sensor network with self-coordinated task allocation and execution. *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews, (Special Issue on Engineering Autonomic Systems)*, March 2005.

[123] Chuck Lutz. Informing joint c2 system-of-systems engineering with agent-based modeling: An analysis and case study. In *Proceedings of 12th ICCRTS, Adapting C2 to the 21st Century*. Martin Loker Heed, Martin Loker Heed, 2007.

[124] Ciaran Lynch and Fergus O'Reilly. Processor choice for wireless sensor networks. In *Proceedings of the Workshop on Real-World Wireless Sensor Networks REALWSN'05*, Sockholm, Sweden, June 2005.

[125] Yajie Ma, Mark Richards, Moustafa Ghanem, Yike Guo, and John Hassard. Air pollution monitoring and mining based on sensor grid in london. *Sensors 2008*, 8:3601–3623.

[126] Geoffrey Mainland, David C. Parkes, and Matt Welsh. Decentralized, adaptive resource allocation for sensor networks. In *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*, pages 315–328, Berkeley, CA, USA, 2005. USENIX Association.

[127] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97, New York, NY, USA, 2002. ACM.

[128] K. Martinez, A. Riddoch, J. Hart, and R. Ong. *A Sensor Network for Glaciers*, chapter 9, pages 125–138. Springer, 2006.

[129] C. Mascolo, L. Capra, and W. Emmerich. *Mobile Computing Middleware (A Survey)*, volume 2497 of *Advanced Lectures on Networking. Lecture Notes in Computer Science*. Springer Verlag, Pisa, Italy, 2002.

[130] Maja J. Mataric and Gaurav S. Sukhatme. Task-allocation and coordination of multiple robots for planetary exploration. In *Proceedings of the 10th International Conference on Advanced Robotics (ICAR)*, pages 61–70, Buda, Hungary, August 2001.

[131] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M.B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *IEEE Infocom 2001*, volume 3, pages 1380–1387. IEEE, April 2001.

[132] C. Melhuish, J. Welsby, and C. Edwards. Using templates for defensive wall building with autonomous mobile ant-like robots. Towards Intelligent Mobile Robots (TIMR), Bristol, UK, 1999.

[133] V. Mhatre and C. Rosenberg. Energy and cost optimizations in wireless sensor networks: A survey. in the 25th Anniversary of GERAD, Kluwer, September 2004.

[134] Melanie Mitchell. *An Introduction to Genetic Algorithms*. The MIT Press, 1998.

[135] Pragnesh Jay Modi, Wei-Min Shen, and Milind Tambe. Distributed resource allocation: Formalization, complexity results and mappings to distributed csps. In *Proceedings of Seventh International Conference on Principles and Practice of Constraint Programming, (CP) 2001*, 2001.

[136] J.M. Molina-Garcia-Pardo, A. Martinez-Sala, M.V. Bueno-Delgado, E. Egea-Lopez, L. Juan-Llacer, and J. Garca-Haro. Channel model at 868 mhz for wireless sensor networks in outdoor scenarios. In *International Workshop on Wireless Ad Hoc Netowrks, IWWAN 2005*, London, May 2005.

[137] A. Molnar, B. Lu, S. Lanzisera, B.W. Cook, and K.S.J. Pister. An ultra-low power 900 mhz rf transceiver for wireless sensor networks. In *Proceedings of the Custom Integrated Circuits Conference, 2004*, pages 401–404. IEEE, October 2004.

[138] Henk Muller and Cliff Randell. An event-driven sensor architecture for low power wearables. In *Proceedings of ICSE 2000, Workshop on Software Engineering for Wearable and Pervasive Computing*, pages 39–41, Limerick, Ireland, June 2000. ACM/IEEE.

[139] Xiaoguang Niu, Xi Huang, Ze Zhao, Yuhe Zhang, Changcheng Huang, and Li Cui. The design and evaluation of a wireless sensor network for mine safety monitoring. In *Global Telecommunications Conference, 2007. GLOBECOM '07*, pages 1291–1295, Washington, DC, USA, November 2007. Chinese Academy of Science, Beijing, IEEE.

[140] Moslem Noori and Masoud Ardakani. A probability model for lifetime of wireless sensor networks. *CoRR*, abs/0710.0020, 2007.

[141] Nesrine Ouferhat and Abdelhamid Mellouck. Qos dynamic routing for wireless sensor networks. In *Q2SWinet '06: Proceedings of the 2nd ACM international workshop on Quality of service & security for wireless and mobile networks*, pages 45–50, New York, NY, USA, 2006. ACM.

[142] El Moustapha Ould-Ahmed-Vall, George F. Riley, and Bonnie S. Heck. Large-scale sensor networks simulation with gtsnets. *Simulation*, 83(3):273–290, 2007.

[143] Chulsung Park, K. Lahiri, and A. Raghunathan. Battery discharge characteristics of wireless sensor nodes: an experimental analysis. In *Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2005, IEEE SECON 2005*, pages 430–440, September 2005.

[144] Lynne E. Parker. Alliance: An architecture for fault tolerant multirobot cooperation. *IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION*, 14(2):220–240, April 1998.

[145] Adrian Perrig, John Stankovic, and David Wagner. Security in wireless sensor networks. *Commun. ACM*, 47(6):53–57, 2004.

[146] Dario Pompili, Tommaso Melodia, and Ian F. Akyildiz. Deployment analysis in underwater acoustic wireless sensor networks. In *WUWNet '06: Proceedings of the 1st ACM international workshop on Underwater networks*, pages 48–55, New York, NY, USA, 2006. ACM.

[147] Ravishankar Rao, Sarma Vrudhula, and Daler N. Rakhmatov. Battery modeling for energy-aware system design. *Computer*, 36(12):77–87, 2003.

[148] S. Ray, R. Ungrangsi, F.D. Pellegrini, A. Trachtenberg, and D. Starobinski. Robust location detection in emergency sensor networks. In *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2003*, volume 2, April 2003.

[149] J. Reich and E. Sklar. Toward automatic reconfiguration of robot-sensor networks for urban search and rescue. In *First International Workshop on: Agent Technology for Disaster Management*, pages 18–23, Hakodate, Japan, May 2006.

[150] Frank Reichenbacha, Matthias Kochb, and Dirk Timmermanna. Closer to reality: Simulating localization algorithms considering defective observations in wireless sensor networks. In *PROCEEDINGS OF THE 3rd WORKSHOP ON POSITIONING, NAVIGATION AND COMMUNICATION (WPNC06)*.

[151] Sokwoo Rhee, Deva Seetharam, and Sheng Liu. Techniques for minimizing power consumption in low-data rate wireless sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC04)*, March 2004.

[152] I.W. Marshall ; C. Roadknight. Adaptive management of an active service network. *BT Technology Journal, Springer*, 18(4):78–84, October 2000.

[153] K. Romer and F. Mattern. The design space of wireless sensor networks. *Wireless Communications, IEEE*, 11(6):54–61, December 2004.

[154] L Sacks, M Britton, I Wokoma, A Marbini, T Adebutu, I Marshall, C Roadknight, J Tateson, D Robinson, and A G Velazquez. The Development of a Robust, Autonomous Sensor Network Platform for Environmental Monitoring. In *IoP Sensors and their Applications (S&A XII)*, University of Limerick, Ireland, 2003.

[155] Andreas Savvides, Sung Park, and Mani B. Srivastava. On modeling networks of wireless microsensors. In *SIGMETRICS '01: Proceedings of the 2001 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 318–319, New York, NY, USA, 2001. ACM.

[156] D. Schmidt, M. Kramer, T. Kuhn, and N. Wehn. Energy modelling in sensor networks. *Advanced Radio Science*, (5):347–351, 2007.

[157] Brian Schott, Michael Bajura, Joseph Czarnaski, Jaroslav Flidr, Tam Tho, and Li Wang. A modular power-aware microsensor with >1000x dynamic power range. In *IPSN*, pages 469–474, 2005.

[158] Thomas D. Seeley and P. Kirk Visscher. Choosing a home: How the scouts in a honey bee swarm perceive the completion of their group decision making. *Behavioral Ecology and Sociobiology*, (54):511–520, 2003.

[159] Onn Shehory and Sarit Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1-2):165–200, May 1998.

[160] Jeong-Hun Shin and Daeyeon Park. A virtual infrastructure for large-scale wireless sensor networks. *Comput. Commun.*, 30(14-15):2853–2866, 2007.

[161] G. Simon, P. Volgyesi, M. Maroti, and A. Ledeczi. Simulation-based optimization of communication protocols for large-scale wireless sensor networks. In *Proceedings of the IEEE Aerospace Conference, 2003*, volume 3, pages 1339–1346, March 2003.

[162] E. Sklar. Netlogo, a multi-agent simulation environment. *Artif Life*, 13(3):303–11, 2007.

[163] Antonios Skordylis, Alexandre Guitton, and Niki Trigoni. Correlation-based data dissemination in traffic monitoring sensor networks. In *CoNEXT '06: Proceedings of the 2006 ACM CoNEXT conference*, pages 1–2, New York, NY, USA, 2006. ACM.

[164] A. Sobeih and J. Hou. A simulation framework for sensor networks in j-sim. Technical Report UIUCDCS-R-2003-2386, Department of Computer Science, University of Illinois at Urbana-Champaign, November 2003.

[165] A. Sobeih, J. C. Hou, Lu-Chuan Kung, Ning Li, Honghai Zhang, Wei-Peng Chen, Hung-Ying Tyan, and Hyuk Lim. J-sim: a simulation and emulation environment for wireless sensor networks. *Wireless Communications, IEEE*, 13(4):104–119, August 2006.

[166] Ahmed Sobeih, Wei-Peng Chen, J.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 *Simulation Symposium, 2005. Proceedings. 38th Annual*, pages 175–187, Dept. of Comput. Sci., Univ. of Illinois at Urbana-Champaign, Urbana, IL, USA, April 2005. IEEE Society, IEEE.

[167] Byungrak Son, Yong sork Her, and Jung-Gyu Kim. A design and implementation of forest-fires surveillance system based on wireless sensor networks for south korea mountains. *IJCSNS International Journal of Computer Science and Network Security*, 6(9B), September 2006.

[168] W. Spears, R. Heil, D. Spears, and D. Zarzhitsky. Physicomimetics for mobile robot formations. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS-04)*, volume 3, pages 1528–1529, 2004.

[169] Thammakit Sriporamanont and Gu Liming. Wireless sensor network simulator. Masters thesis in electrical engineering, School of Information Science, Computer and Electrical Engineering, Halmstad University, Box 823, S-301 18 Halmstad, Sweden, January 2006.

[170] J. A. Stankovic, T. E. Abdelzaher, Lu Chenyang, Sha Lui, and J. C. Hou. Real-time communication and coordination in embedded sensor networks. In *Proceedings of the IEEE*, volume 91, pages 1002–1022, July 2003.

[171] Jan Steffan, Ludger Fiege, Mariano Cilia, and Alejandro Buchmann. Scoping in wireless sensor networks: a position paper. In *MPAC '04: Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing*, pages 167–171, New York, NY, USA, 2004. ACM.

[172] Jan Steffan, Ludger Fiege, Mariano Cilia, and Alejandro Buchmann. Towards multi-purpose wireless sensor networks. In *ICW '05: Proceedings of the 2005 Systems Communications*, pages 336–341, Washington, DC, USA, 2005. IEEE Computer Society.

[173] Anthony (Tony) Stentz and M Bernardine Dias. A free market architecture for coordinating multiple robots. Technical Report CMU-RI-TR-99-42, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, December 1999.

[174] Peter Stone and Manuela Veloso. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. *Artificial Intelligence*, 110(2):241–273, 1999.

[175] Steven H. Strogatz. *Sync: The Emerging Science of Spontaneous Order*. Penguin Press Science, April 2004.

[176] P. B. Sujit, A. Sinha, and D. Ghose. Multiple uav task allocation using negotiation. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 471–478, New York, NY, USA, 2006. ACM.

[177] Hartmut Surmann and Antonio Morales. A five layer sensor architecture for autonomous robots in indoor environments. In *Proceedings of the International symposium on robotics and automation ISRA2000*, pages 533–538, Monterrey, N.L., Mxico, November 2000.

[178] Y. Tan and The Argo Collaboration. High altitude observatory ybj and argo project. In *International Cosmic Ray Conference*, volume 2 of *International Cosmic Ray Conference*, page 821, 2001.

[179] Feilong Tang1, Minglu Li1, Chuliang Weng1, Chongqing Zhang1, Wenzhe Zhang1, Hongyu Huang1, and Yi Wang1. *Combining Wireless Sensor Network with Grid for Intelligent City Traffic*, volume 4186/2006 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, August 2006.

[180] C. Tessier, C. Cariou, C. Debain, F. Chausse, R. Chapuis, and C. Rousset. A real-time, multi-sensor architecture for fusion of delayed observations: Application to vehicle localisation. In *Proceedings of the First National Workshop on Control Architectures of Robots*, Monthpellier, France, April 2006.

[181] Guy Theraulaz, Eric Bonabeau, and Jean-Louis Deneubourg. Response threshold reinforcement and division of labour in insect societies. *Proceedings of the Royal Society of London B*, (265):327–335, February 1998.

[182] Yuan Tian, Eylem Ekici, , and Fusun Ozguner. Energy-constrained task mapping and scheduling in wireless sensor networks. In *Proceedings of the First IEEE International Workshop on Resource Provisioning and Management in Sensor Networks (RPMSN05)*, Washington, DC, November 2005.

[183] Sameer Tilak, Nael B. Abu-Ghazaleh, and Wendi Heinzelman. A taxonomy of wireless micro-sensor network models. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(2):28–36, 2002.

[184] Ben L. Titzer, Daniel K. Lee, and Jens Palsberg. Avrora: scalable sensor network simulation with precise timing. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, page 67, Piscataway, NJ, USA, 2005. IEEE Press.

[185] Kevin E. Trenberth and Julie M. Caron. Estimates of meridional atmosphere and ocean heat transports. *Journal of Climate*, 14:3433–3443, august 2001.

[186] ANIL KUMAR TRIPATHI, BIPLAB KUMER SARKER, and NAVEEN KUMAR. A ga based multiple task allocation considering load. *International Journal of High Speed Computing*, 11(4):203–214, 2000.

[187] M. Tubaishat and S. Madria. Sensor networks: an overview. *Potentials, IEEE*, 22(2):20–23, April–May 2003.

[188] Paul Valckenaers, Hadeli, Bart Saint Germain, Paul Verstraete, and Hendrik Van Brussel. Mas coordination and control based on stigmergy. *Comput. Ind.*, 58(7):621–629, 2007.

[189] P. Verissimo and A. Casimiro. Event-driven support of real-time sentient objects. In *Proceedings of the Eighth IEEE International Workshop*

on Object-oriented Real-time Dependable Systems (WORDS 2003), pages 2–9, January 2003.

[190] Marcos Augusto M. Vieira, Claudionor N. Coelho. Jr., Diogenes Ceclio da Silva Jr., and Jose M. da Mata. Survey on wireless sensor network devices. In *Emerging Technologies and Factory Automation(ETFA03), IEEE Conference*, pages 16–19. IEEE, September 2003.

[191] Lan Wang and Yang Xiao. A survey of energy-efficient scheduling mechanisms in sensor networks. *Mob. Netw. Appl.*, 11(5):723–740, 2006.

[192] Ye Wen, Rich Wolski, and Gregory Moore. Disens: scalable distributed sensor network simulation. In *PPoPP '07: Proceedings of the 12th ACM SIGPLAN symposium on Principles and practice of parallel programming*, pages 24–34, New York, NY, USA, 2007. ACM.

[193] Ye Wen, Wei Zhang, Rich Wolski, and Navraj Chohan. Simulation-based augmented reality for sensor network development. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 275–288, New York, NY, USA, 2007. ACM.

[194] Barry Brian Werger and Maja J Mataric. Broadcast of local eligibility: Behavior-based control for strongly-cooperative robot teams. In *Proceedings of Autonomous Agents*, 2000.

[195] Darrell Whitley. The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, San Mateo, CA, 1989. Morgan Kaufman.

[196] U Wilensky. *NetLogo*. Center for Connected Learning and Computer-Based Modeling in the Northwestern University, Evanston, IL, 1.3 and 1.4 edition, 1999.

[197] George Christopher Williams. *Adaptation and Natural Selection*. Princeton University Press, Princeton, N.J., reprint edition, may 13 1996 edition, 1966.

[198] A. Willig, R. Shah, J. Rabaey, and A. Wolisz. Altruists in the picoradio sensor network. In *Proceedings of the 4th IEEE International Workshop on Factory Communication Systems, 2002*, pages 175–184, 2002.

[199] Matt Wilson, Chris Melhuish, Ana B. Sendova-Franks, and Samuel Scholes. Algorithms for building annular structures with minimalist robots inspired by brood sorting in ant colonies. *Autonomous Robots*, 17(2-3):115–136, 2004.

[200] J.-X. Wu, Z.-H. Zhou, and Z.-Q. Chen. Ensemble of ga based selective neural network ensembles. In *Proceedings of the 8th International Conference on Neural Information Processing*.

[201] Ning Xu. A survey of sensor network applications. Survey Paper for CS694a.

[202] Peng-Yeng Yin, Shiuh-Sheng Yu, Pei-Pei Wang, and Yi-Te Wang. Task allocation for maximizing reliability of a distributed system using hybrid particle swarm optimization. *J. Syst. Softw.*, 80(5):724–735, 2007.

[203] Mohamed Younis, Kemal Akkaya, and Anugeetha Kunjithapatham. Optimization of task allocation in a clusterbased sensor network. In *Proceedings of the 8th IEEE International Symposium on Computers and Communications(ISCC'2003)*, page 329, Antalya, Turkey, June 2003.

[204] Yang Yu and Viktor K. Prasanna. Energy-balanced task allocation for collaborative processing in wireless sensor networks. *Mobile Networks and Applications*, 10(1–2):115–131, 2005.

[205] D. Zarzhitsky, D. F. Spears, and W. M. Spears. warms for chemical plume tracing. In *Proceedings of SIS 2005*, Swarm Intelligence Symposium, pages 249–256. IEEE, IEEE, June 2005.

[206] Jingbin Zhang, Ting Yan, and S. H. Son. Deployment strategies for differentiated detection in wireless sensor networks. In *3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and*

Networks, 2006. SECON '06. 2006, volume 1, pages 316–325, Reston, VA, September 2006.

[207] Jichuan Zhao and A. T. Erdogan. A novel self-organizing hybrid network protocol for wireless sensor networks. In *First NASA/ESA Conference on Adaptive Hardware and Systems, 2006. AHS 2006.*, pages 412–419. IEEE, June 2006.

[208] Z.-H. Zhou, J.-X. Wu, W. Tang, and Z.-Q. Chen. Combining regresson estimators: Ga-based selective neural network ensemble. *International Journal of Computational Intelligence and Applications*, 1(4):341–356, 2001.

[209] Kasun De Zoysa and Chamath Keppitiyagama. Busnet a sensor network built over a public transport system. In *Proceedings of EWSN 2007, the fourth European conference on Wireless Sensor Networks*, pages 9–10, TU Delft, Netherland, January 2007.

List of Figures

List of Tables