COMBINATORICS-BASED ENERGY CONSERVATION METHODS IN WIRELESS SENSOR NETWORKS

WONG YEW FAI
M.Eng., B.Eng,(Hons,), NUS

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY OF ENGINEERING

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

FACULTY OF ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

COMBINATORICS-BASED ENERGY CONSERVATION METHODS IN WIRELESS SENSOR NETWORKS

WONG YEW FAI
M.Eng., B.Eng,(Hons,), NUS

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY OF ENGINEERING

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

FACULTY OF ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

Acknowledgement

I would like to express my gratitude to Professor Lawrence Wong Wai Choong for his kind supervision and guidance in my research work. I am grateful for his ideas, thoughts, experience, suggestions and all the discussion time that has always made our work an improvement from its original form. In particular, Professor Lawrence Wong has been generously understanding and patience, for those times when my research progress is slow due to my company work commitments.

I would also like to thank Dr Ngoh Lek Heng for his co-supervisory role in my research work amidst his busy schedules at the Institute for Infocomm Research. His detailed analysis and in-depth discussion over very specific parts of my research has no doubt made progress faster and smoother. He has also suggested clear organization of ideas and concepts that has made our work in every way, a significant improvement.

I am appreciative of Dr Wu Jian Kang, Institute for Infocomm Research, for his experience and our discussions on the application aspect of sensor networks. He has made insightful suggestions and comments to our work related to signal processing, data fusion and particle filters. Dr Tele Tan, Curtin University of Technology, has also contributed many ideas on the application aspects in the early days of this research work. Similarly, I would also like to express my thanks to Dr Winston Seah Khoon Guan, Institute for Infocomm Research, for sharing his experience in underwater sensor networks and our discussions on wakeup schemes for underwater sensor networks. My sincere appreciation also goes to Professor Tham Chen Khong and Professor Vikram Srinivasan, National University of Singapore, for their kind review of our work related to query-based sensor networks. They have shared a wealth of useful comments and ideas in our discussions so that our formulations and concepts are elaborated clearly and unambiguously. My special thanks extend to Dr Jaya Shankar, Institute for Infocomm Research, for his keen interest in all my research efforts and his understanding whenever I need to manage between academic research work and company research work.

I would like to thank Professor Chua Kee Chaing, Professor Soh Wee Seng and once again Professor Tham Chen Khong for their invaluable comments and feedback as panel members during my qualifying examinations.

I must also mention Mr Jean-Christopher Renaud for his contributions in our implementation experiments using Crossbow motes and Miss Trina Kok for her help on the classification of energy conservation schemes in the literature. Credit also goes to Mr Dale Green, Teledyne-Benthos Inc., for his invaluable experiences and thoughts on underwater sensor networks and the acoustic signaling challenges in real world deployments. I am also thankful for the help I received from Mr J. Vedvyas in some of the related demonstrations that we undertook.

Last but not least, credit must also go to Flossie and Oh Chew Ling for their friendly assistance in organizing our biweekly meetings. Finally, I would like to thank all that have directly or indirectly contributed to the fulfillment of our research work.

Wong Yew Fai

31 October 2007

Table of Contents

Summary

Wireless sensor networks have emerged as one of the new fields of research where their potential applications may range widely from elderly healthcare, military defense, wildlife monitoring, disaster recovery, construction safety monitoring, tsunami warning systems, target tracking, intrusion detection and others. Owing to their relatively small form factor and cheap manufacturing costs, sensors may be deployed in high density to monitor an area of interest. One main challenge in deploying such wireless networks is the energy scarcity problem since sensors are often powered only by regular batteries. This energy conservation issue in sensor networks is paramount and complicated by application requirements such as network connectivity, sensing coverage, information delay, and implementation cost constraints, which are not all taken into account in the existing literature. While energy expenditure in the network must be controlled, the sensor network must still serve the purpose of the sensor network application.

We propose a class of deterministic wakeup schemes, the cyclic symmetric block designs (CSBD), related to the field of Combinatorics. We consider important requirements of sensor network applications and propose appropriate CSBD wakeup schedules to conserve energy for each purpose. We describe the application of CSBD-based schemes to three main categories of sensor networks – Agent-based sensor networks, Query-based sensor networks, and Ad-hoc and sparse sensor networks. Each category of sensor networks operates with different requirements/assumptions and we provide detailed analysis and discussion on the benefits of CSBD in our work. We further support and justify our claims with comprehensive simulation studies and selected implementation results.

Keywords:

Combinatorics, Energy Conservation, Energy Saving, Wakeup Scheme, Wireless Sensor Networks, Adaptation

List of Tables

List of Figures

Chapter 1: Introduction

1.1. Wireless Sensor Networks & Their Key Challenges

The widespread interest in wireless sensor networks research in recent years may be attributed to the possibility of such networks emerging as a disruptive force in shaping the way many activities are carried out. With the ability to sense, store and communicate a host of different kinds of information about the environment from seismic data to air quality records to electromagnetic fluctuations, the potential impact of sensor networks on many different disciplinary fields can be considerably diverse and huge. With further advancements in reducing the form factors of such sensors, the realistic deployment size of sensor networks can also be predictably large.

While sensor networks can be applied to solve many different problems across various different platforms, several challenges arise from this relatively new domain of research. Being small in size and wireless, most sensors are powered only by batteries, and energy becomes a scarce resource in such networks. The issue of managing or controlling the use of energy for the sensors' operations is principally important. The physical deployment of sensors to sense the environment presents itself a sensing coverage and deployment density problem. This coverage problem is further complicated by sensors optionally switching themselves off during certain periods to conserve energy. Depending on the application, a sufficiently good coverage of the intended environment under monitoring may be required. With sensors deployed in large numbers, each collecting vast amounts of data individually, organization of sensor information and data flow within the network becomes another huge challenge. Since sensors are often scattered randomly during deployment, efficient and cost-effective localization techniques for individual sensors to discover either their absolute or relative positions in the sensing field may also be necessary. Other main challenges may include security issues, routing issues and collision avoidance issues related to deploying sensor networks in large numbers or in high density scenarios.

1.2. Real world Implementations of Sensor Networks

The most commonly known and probably the de-facto sensor network platform, is the Berkeley family of motes from Crossbow Technology Inc. [1]. Other competing platforms include the Cricket [2], the WINS Sensoria nodes [3] and the Specknet [5, 4] systems. The influence of such sensor networks on applications is wide-ranging, and this section can only highlight a fraction of the many real-world applications of sensor networks.

Collaborations between Fujitsu laboratories, Venturi Wireless and San Jose State University [6] report on a sensor networks prototype developed for the purpose of elderly healthcare. The aim is to monitor the medication conditions of elderly patients by integrating Radio Frequency Identification (RFID) technology with sensor network technology. Similarly, the collaboration between Motion Analysis Laboratory at the Spaulding Rehabilitation Hospital and Harvard University [7, 8] are developing a sensor board for monitoring limb movements and muscle activity of stroke patients during rehabilitation exercise.

The Sensor Networks Research Group at the University of Wisconsin [9] successfully deployed a sensor network at 29 Palms, California to detect and classify signals from moving military targets. In their experiments, acoustic, seismic and passive infrared signals were collected from two different types of military vehicles – the Assault Amphibian Vehicle (AAV) and the Dragon Wagon (DW). The University of California at Berkeley also designed and deployed 100 sensors in a 400m^2 outdoor sensing field for the purpose of vehicle tracking and intruder interception.

Sensor networks have also been applied to habitat and wildlife monitoring. Noticeably, the deployment at Great Duck Island [10] with about 200 sensors measure basic environmental parameters such as light, pressure, temperature and humidity information to serve as long-term baseline data for further work. At James Reserve, sensors are also deployed to monitor the ecosystem for understanding the response of vegetation to climate changes. The collaboration between Intel Corporation and the University of California at Berkeley also developed a habitat monitoring kit for biologists and researchers to reliably collect data from previously inaccessible locations.

Recently, the building and construction sector has also developed interests in employing wireless sensing technologies to monitor the health of structural beams during construction and excavations. As it is a legal requirement in many countries for construction companies to sufficiently monitor supporting beams in any construction activity, the current cabled-sensing solutions that are in use are costly. In-lay cables are expensive and are commonly severed accidentally in work sites thereby incurring frustrating schedule delays and costly repair overheads. Moreover, in muddy terrains and deep troughs, cabling becomes impossible and manual monitoring by a worker becomes necessary. Manual monitoring involves manual recording of long strings of data and identification numbers that are often erred by human mistakes.

Researchers from the University of Pittsburgh [11] are also hoping to develop a network of ocean floor mobile sensors to complement existing deep water tsunami detection buoys in the Pacific and Indian Oceans. By offering greater coverage of the ocean floor, detections that are previously missed by the more expensive deep water buoys that are spaced far apart, may be picked up by the cheaper sensor network that is in place.

1.3. Related Work

As real world sensor network deployment is becoming a trend and reality, the key problem of conserving energy in sensors has encouraged many researchers to devise various solutions based upon different assumptions. In this section, we provide a review of some of these related works in energy conservation techniques.

1.3.1 Existing Energy-Conservation Wakeup Schemes

The Random Wakeup Solution

Paruchuri et. al. proposed the Random Asynchronous Wakeup (RAW) scheme [12] where each node randomly wakes up once in every time frame, be awake for a predetermined fixed time and then sleeps again. Data is sent from a node N to a forwarding set of neighbouring nodes so that delay can be minimized. The forwarding set includes all nodes that lie in the area intersection of the circular transmission range of node N and the circular range of a certain radius centered about the destination node. It is reported that for 10 nodes in the forwarding set, a per-hop packet loss rate of 18% is expected. In their work, a node deployment density of at least $10/R_C^2$ is used where R_C is the communication radius of nodes. This also represents the frequency at which nodes wake up but find no other nodes in communication range to transmit or forward data. However, wakeup schedules are time-asynchronous owing to the randomness in the solution. Delays incurred are small because of numerous choices in forwarding nodes in the forwarding sets.

Kumar et. al. proposed the Random Independent Scheduling (RIS) [13] where time is divided into cycles using some time synchronization method. At the start of each cycle, every node independently takes on an “Awake” mode with probability p and “Sleep” mode with probability $(1 - p)$. Therefore, RIS uses this parameter p to control network lifetime. RIS also determines how nodes should be initially deployed to ensure asymptotic m-coverage. In asymptotic m-coverage, the network is m-covered only when the number of sensors deployed approaches infinity. However, although RIS has no communication overheads and requires no location information, it does not address connectivity issues and the problem of nodes waking up to find no communicable neighbours is obvious. The scheme is also not robust against node failures and requires expensive time-synchronization techniques that inhibit scalability.

The Connected Dominating Set Wakeup Solution

A connected dominating set (CDS) of a network graph $G(V, E)$ with nodes V and links E, is a set of nodes $V' \subset V$ such that every node not in V' is connected to at least one node in V' by some link in E; and the subgraph induced by V' is also connected. CDS sensor nodes are switched to the “Awake” or “On” state while non-CDS sensor nodes are put to the “Sleep” or “Off” state. The CDS in a network therefore acts as a “backbone” of nodes where information may be sent from one node to another across the network in relatively short time. To reduce energy consumption as much as possible, many algorithms aim to elect a minimum connected dominating set (MCDS), i.e. a CDS with minimum cardinality. Election of nodes to form the MCDS is an NP-complete problem, but in practice, heuristics may be used to form a CDS that approximates the MCDS. Centralized CDS election algorithms such as that by Guha and Khuller (GH) [13] can theoretically be implemented in a distributed manner, albeit with higher control overhead in exchanging neighbour information. Topology Management by Priority Ordering, TMPO [14] is a distributed algorithm that elects CDS nodes in an energy-aware network by addressing the load-balancing aspect of the network, but without considering sensing coverage. Yet, another algorithm SPAN [15] is a distributed randomized algorithm that maintains the original connectivity of the network via the “backbone” of nodes, based on a “willingness” factor that is dependent on remaining node energy and neighbour count. Wu and Li (WL) [16] further proposed an algorithm similar to SPAN that incorporates additional pruning rules to reduce the cardinality of the elected CDS of sensor nodes.

CDS election schemes require periodic broadcasts which limit true energy savings.

The Two-Channel Paging Wakeup Solution

The Sparse Topology and Energy Management (STEM) protocol [17] for sensor networks proposes the use of two channels, one for data transmission and the other as a control or paging channel to wake up neighbouring sensor nodes. When a sensor node has data to send, it uses a wakeup tone or beacon message to wake up the necessary neighbouring nodes using the paging channel and transmits actual data on the data channel. In this manner, sensors are reactively being turned on as and when required. The drawback of such a solution is that it requires the cost of two channels and energy savings are insignificant because the paging channel is required to be always at the "monitoring" state or in "Idle" mode (in contrast to "Sleep" mode) to receive possible wakeup beacons. In the "Idle" mode, the sensor node continues to monitor the channel for possible control packets to facilitate the transition into other modes of operation. It is widely known that energy savings are not significant [18, 71, 89] when nodes are merely set to the "Idle" mode instead of the "Sleep" mode, where the latter switches off its communication module completely. Moreover, for nodes to operate in the "Idle" mode, the required dual channel communication increases implementation costs. However, connectivity of the network is equivalent to one that is fully awake and delays incurred in data transmission are minimized, less the time to wakeup neighbouring sensors.

While STEM uses a separate channel to page neighbouring nodes into the "Awake" mode, the Power Aware Multi-Access Protocol with Signaling (PAMAS) [19] proposes the use of a separate signaling channel that conserves energy by turning off the sensor node if it has no data to send and a neighbour node is transmitting at the same time to another node. Again, the added cost is the extra channel and its maintenance.

The Information-Configured Wakeup Solution

There are schemes that configure their sensor wakeup schedules based on information received from neighbouring sensor nodes. The Probing Environment and Adaptive Sleeping (PEAS) algorithm [20] for sensor networks is one where nodes configure their wakeup times based on counting the number of neighbouring nodes that they discover after deployment. It is assumed that nodes wake up asynchronously after they are first deployed, after which sensor nodes that operates in “Awake” mode send PROBE messages to neighbours. If no replies were received, the node stays in the “Awake” mode until it is completely depleted of its energy. If at least one reply is received, the node operates in the “Sleep” mode. Nodes in the “Sleep” mode regularly wake up to send PROBE messages. The probing range may also be chosen to meet certain sensing coverage criterion. PEAS is time-asynchronous and assumes a very dense network deployment scenario. Since nodes in PEAS permanently operate in the “Awake” mode and subsequently deplete of all their energies once they discover no PROBE replies, energy consumption in the network is unbalanced and may cause network partitioning. Gui et. al. improved PEAS by proposing the Probing Environment and Collaborative Adaptive Sleeping (PECAS) scheme [21] with additional features that allow a sensor node that is already in the “Awake” mode to go back into “Idle” or “Sleep” mode beyond some energy threshold limit. Thus, PECAS can also be classified under the “Paging Solution” described earlier when equipped with this dual channel capability.

While PEAS and PECAS are all configured by neighbour count, the Coverage Configuration Protocol (CCP) [22] configures the wakeup times of a sensor node by the degree of sensing coverage of its neighbour nodes. The scheme establishes a relationship between sensing coverage and network connectivity where a *m*-covered network implies a *m*-connectivity network, for as long as the communication range is twice its sensing coverage radius (double range property). With this, CCP strives to maximize the number of sleeping nodes, while maintaining both *m*-coverage and *m*-connectivity in the network at the same time. Each node first evaluates if its coverage area is *m*-covered and this computational complexity is O(N^3) [23] where N is the number of sensors within a distance of two times its sensing coverage radius. If it is *m*-covered, it is also *m*-connected and the node goes into “Idle” mode, and subsequently into the “Sleep” mode after expiration of a random timer. Nodes in the “Sleep” mode periodically enter the “Idle” mode to monitor the channel to check if the area is still *m*-covered. If not, it enters the “Awake” mode; otherwise, it goes back to “Sleep” mode. CCP operates together with SPAN [15] for the case when the double range property fails. SPAN is used as a connectivity preserving scheme and some nodes working under CCP+SPAN remains in “Awake” mode even if they are redundant in sensing coverage so that desired connectivity is maintained.

Wakeup schemes may also be configured by information other than neighbour count or neighbour sensing coverage. The Adaptive Self-Configuring Sensor Networks Topologies (ASCENT) [24] protocol measures neighbour connectivity as well as data loss rate to configure wakeup times. Each node keeps track of monotonically increasing sequence numbers in packets and infers the data loss rate. Nodes also infer the number of active neighbours by keeping track of packets received from each neighbour. Therefore, there is no periodic probing required to discover neighbours. ASCENT aims to achieve optimal and maximum connectivity that minimizes collision rate. The drawback of ASCENT is its assumption of a very dense network scenario and that network partitioning is not a key issue.

<u>The Deterministic Wakeup Solution</u>

A class of deterministic wakeup solutions based on the field of Combinatorics [25, 26] has been proposed. Combinatorics is a branch of mathematics concerned with the selection, arrangement and operation of elements in a set. In sensor wakeup schemes context, they represent the arrangement of a number of wakeup time slots in a set of all available time slots within one time cycle. Each sensor is assigned one time schedule based on this arrangement. Zheng et. al. [27] proposed a cyclic symmetric block design (CSBD) where every sensor schedule has exactly one active wakeup slot overlap with any other sensor schedule in the set. All wakeup schedules in this design are also cyclic shifts of each other. Each sensor is assigned one schedule based on the design set. The existence of such a design is not trivial and implies that any sensor node using any schedule from this set is always guaranteed to be able to communicate multi-hop to any other node in the set within bounded time. Moreover, many other properties such as network connectivity and node sensing coverage can also be shown to be preserved within bounded time. Unlike most other straightforward deterministic schemes, this design is time-asynchronous despite wakeup times being arranged in slots and cycles, thereby requiring no expensive synchronization of clocks amongst the sensor nodes. This is achieved by having beacons announcing the beginning of every active time slot in each schedule. This scheme also consider only nodes in the "Awake" mode and "Sleep" mode and do not put nodes in the "Idle" mode, thereby without requiring separate communication module for channel monitoring and this save on implementation cost. Being deterministic, it is also easy to see that they are easy to implement and deploy requiring less operational overheads. At the moment, the work in such wakeup techniques is currently only limited to the Mobile Ad-hoc NETwork (MANET) context where nodes are mobile by default with no sensing capabilities.

1.3.2 Other Energy-Conservation Methods in Sensor Networks

Energy Conservation in Routing

Techniques in energy conservation are not limited to wakeup schemes for sensors. Intelligent routing methods that are energy-aware may be deployed in conjunction with an underlying wakeup scheme to jointly conserve power in sensor networks. In [86, 85], both propose energy-efficient routing algorithms for sensor network applications. [86] ensures that delay constraints of applications are met while performing energy-efficient routing, and [85], aggregates packet streams during routing, and demonstrates that energy reductions can be achieved by a factor of 2 to 3. We shall, however, defer the discussion on data aggregation later.

In [87], the authors identified the drawbacks of single-path routing and multi-path routing in terms of guaranteed delivery and energy consumption. While single-path routing saves more energy, it often suffers from poor packet delivery ratios because of the unpredictable nature of the network nodes and its environment. Although multi-path solutions ensure better packet delivery probabilities, energy consumption scales with the number of paths used. [87] proposes to forward data along a single path and repairs the path ‘on the fly’ only when a link breakage is detected. [87] demonstrates that both delivery guarantees and energy usage can be controlled with their proposed protocol.

[36] investigates an agent-based approach to routing to conserve energy. Before a next-hop node is considered in routing, data agents take into consideration both routing cost and remaining node energies. The probability of choosing a next-hop node is therefore proportional to its remaining energy and inversely proportional to its routing cost. Data aggregation is also considered in their routing protocol.

Yet, one of the most popular and influential data dissemination paradigms in sensor networks is Directed Diffusion (DD) [88]. It proposes a novel data-centric approach to disseminate or ‘route’ data in a sensor network, which can result in significant energy savings. In DD, data is named using attribute-value pairs so that a sensing task can be disseminated throughout the sensor network as an interest for that named data. The dissemination process itself sets up ‘gradients’ in the network that ‘attracts’ events so that the ‘data’ can be matched to ‘interest’. Events flow towards originators of interests along multiple paths, where only one, or a small number of paths, are ‘reinforced’ for data propagation. Since routing paths, or more appropriately data dissemination paths, are decided based on data and interests, such an approach also facilitates data aggregation along paths in the network, thereby saving energy.

In-Network Data Aggregation Energy Conservation

Data aggregation techniques in sensor networks promise to conserve energy by attempting to aggregate, suppress or summarize information before every transmission. This acknowledges the fact that communication energy forms the bulk of energy usage in sensors, and seeks to minimize packet transmissions or reduce the size of every transmission. The Temporal coherency-aware In-Network Aggregation (TiNA) [80] scheme is the first of such schemes to exploit temporal correlation in a sequence of sensor readings to support energy-efficient quality of data in the context of in-network aggregation. It is possible to increase the quality of data during an aggregation process when the time given to perform readings is too short for all data to be propagated up through the network. Depending on where in the network the sensor is, the information kept is different. In TiNA, every leaf node keeps only the last reading successfully sent or reported to its parent, while each internal node keeps both last reported reading, and the last view it received from each child node. The basic idea behind temporal coherency is to send a reading from the sensor only if the reading differs from the last recorded reading by more than some stated tolerance. This tolerance can be user-dependent or network-dictated if the network cannot support the specified tolerance level. [80] shows that power consumption may be reduced by up to 60% without any loss of data quality and the network lifetime may be extended by up to three times. These results, however, ignore the possibility of an underlying wakeup scheme that can potentially further extend network lifetime, and can be employed in conjunction with data aggregation methods.

A predecessor of TiNA is the Tiny AGgregation (TAG) [60] service for ad-hoc sensor networks. Here, temporal correlations between sensor readings are not taken into account. Instead, it provides a declarative interface for data collection and aggregation, inspired by selection and aggregation facilities in database query language. TAG also distributes and executes aggregation queries in the sensor network in a power-efficient manner. By making use of the original SQL specification options of COUNT, MIN, MAX, SUM and AVERAGE, information may be quickly summarized and the amount of transmissions required largely reduced, thereby achieving power consumption efficiency. TAG further provides a general classification of aggregate functions so that their proposed service is not limited to just five of the original aggregator specifications. In their results, COUNT, MIN, MAX and AVERAGE aggregators using in-network TAG service significantly reduces the number of bytes transmitted in the network, while other functions such as MEDIAN and COUNT DISTINCT show very little or no improvements compared to centralized processing.

The performance of data aggregation, however, depends very much on network density. For dense networks, the proportion of redundant information is usually higher than sparse networks. The effect and impact of data aggregation techniques on the performance of applications can therefore vary. [81] compares a greedy aggregation approach with an opportunistic aggregation method over different network densities. The greedy approach appears to have better energy efficiency, particularly for denser networks. The key explanation for this is that denser networks offer more shortest paths from a source to a sink that greedy algorithms depend on. For sparse networks, [82] offers an aggregation technique that allow two nodes that wish to communicate at roughly the same time to discover each other at a cost that is proportional to their network distance. The authors in [82] further evaluate the quality of a sparse aggregation tree that is formed as a result. Other related work on in-network aggregation includes [83], which investigates single-level aggregation and hierarchical aggregation to conserve energy in the network, and [84] which proposes a model-driven data acquisition method in sensor networks, by enriching interactive sensor querying with statistical modeling techniques. Queries are therefore answered by introducing approximations (based on some predefined model) with probabilistic confidences. Again, one of the objectives is to conserve energy by approximating answers to a query.

1.4. Motivation & Contributions

Limitations of Existing Wakeup Schemes

The random wakeup solutions rely on dense network deployment scenarios and do not provide any deterministic guarantees in terms of data delays and network connectivity. For the CDS-based wakeup methods, election of MCDS nodes is an NP-complete problem and they hardly consider sensing coverage issues. While two-channel paging wakeup solutions are costly to implement on a large scale and energy savings in the "Idle" mode are not known to be significant, deterministic wakeup schemes that are cost-effective to implement have not been analyzed and studied in detailed in the sensor network context. In the case of the various information-configured wakeup solutions, they may incur high operation overheads in terms of periodic control messages, have high computational complexities in translating measured information into wakeup schedules for sensors, or can sometimes be over-simplistic. Moreover, none of these wakeup techniques address database issues where sensors may be queried for information by application users. The idea of treating a sensor network as a distributed database of stored information forms an important part of the sensor networks research literature. However, little is known of the performance of such sensor database systems when applied with wakeup schemes for sensors. Indeed, existing energy conservation techniques each have their limitations and do not address a majority of the specific issues that are important to sensor networks deployment.

While energy conservation in sensor networks is vital in extending the useful lifetime of a network, it is also important to consider several performance aspects of sensor networks that directly affect its applicability in the real world. In our opinion, the following factors are crucial:

- Network connectivity issues,
- Sensing coverage issues,

- Query waiting delays from sensors, and
- Implementation costs.

Existing schemes do not consider all these aspects and their applications in the real world can only be limited and specific. This motivates our work to propose a unified energy-efficient wakeup architecture for sensor networks that considers all these issues at the same time.

Our Proposed Solution

We have selected to base our work on a class of mathematically-inspired deterministic wakeup methods – the Cyclic Symmetric Block Design (CSBD) that researchers have often overlooked, and thus is lacking in detailed research analysis. We shall show later in this thesis, that this class of deterministic wakeup schemes, and its variants, are simple to implement, and are capable of addressing all the issues (connectivity, coverage, query delays and implementation issues) that are crucial for real world application. CSBD promises to address all these issues where other existing wakeup schemes do not consider, or only consider them in part. Our proposed CSBD design is therefore superior to existing schemes that are unable to address all these issues simultaneously. Since CSBD is deterministic, the amount of computational overheads is minimal. We show further that communication overhead can also be low with our proposed On-Demand Neighbour Discovery (ODND) scheme.

In our work, we highlight that although time is discretized and slotted in CSBD wakeup schemes, time-asynchronous neighbour discovery can be guaranteed within some finite time, thereby requiring no costly time synchronization techniques to be implemented in the nodes. Similarly, we show further that sensing coverage and network connectivity can both be guaranteed to be preserved within some known bounded time. Since this time bound may be configured by setting certain parameters in the design set, our designs are generally applicable to a wide-ranging set of applications from delayinsensitive monitoring applications to real-time target tracking and intrusion detection applications. We shall also show that our proposed wakeup design possesses interesting properties when the sensor network is treated and queried as a distributed database by different classes of users. One such property is that our wakeup design guarantees a theoretical zero waiting time for query replies to reach the users that are approximately one-hop away from the event of interest, provided that certain criteria are fulfilled.

Both CSBD, in its original form, and its variants are proposed to suit different application needs. In cases where CSBD works well in its original form, we show how it may be configured to take into account several key design considerations. In cases where additional constraints are to be fulfilled, we propose variants for CSBD to meet these additional requirements. In particular, we proposed the Tracking Wakeup Schedule Function (TWSF) for target tracking applications and the Adaptive Wakeup Schedule Function (AWSF) for sparse networks in certain environments. In such cases, variant-solutions of CSBD continue to inherit a subset of the desirable properties from its parent design.

The main contributions of this thesis can be summarised as follows:

- We examine and analyze in detail, a class of deterministic wakeup methods – CSBD, based on the mathematical field of Combinatorics. Analysis and study have been focused on the four factors of Sensing Coverage, Network Connectivity, Query Waiting Delays and Implementation issues related to sensor networks.
- We propose the use of CSBD, and its variants, to different classes of sensor systems, namely the Agent-Based Sensor Network Systems, the Query-Based Sensor Network Systems, the Ad Hoc Sparse Sensor Network Systems and even a combination of these systems.

- A list of related publications based on our work has been included in Appendix A

1.5. Organization of thesis

This thesis is organised as follows: We justify the choice of our selected approach to conserving energy using Cyclic Symmetric Block Designs (CSBD) based on the mathematical field of Combinatorics in Chapter 2. We apply our analysis and study to Agent-Based Sensor Networks in Chapter 3, Query-Based Sensor Networks in Chapter 4, and Ad Hoc and Sparse Sensor Networks in Chapter 5. We conclude our work and highlight possible future work in Chapter 6.

Chapter 2: Combinatorics-Based Wakeup Scheme and Its Properties

We base our solution on a class of deterministic wakeup schemes related to the field of Combinatorics. In particular, we are interested in the Cyclic Symmetric Block Design (CSBD) first proposed by Zheng [27] in the context of MANETs for its time-asynchronous neighbour discovery property, which we will provide further discussion. In this chapter, we first provide an overview of this design and its characteristics in sections 2.1 and 2.2. We describe neighbour discovery and data transmission issues in CSBD in section 2.3, and introduce our "on-demand" neighbour discovery technique. Subsequently, we analyze and discuss this design with respect to two of the four important sensor network factors: Network Connectivity in section 2.4, and Implementation Costs in section 2.5. The other two consideration factors of Sensing Coverage and Query Waiting Delays will be discussed in later chapters as their analysis is more specific in nature. We summarize this chapter in section 2.6.

2.1. The Cyclic Symmetric Block Design (CSBD)

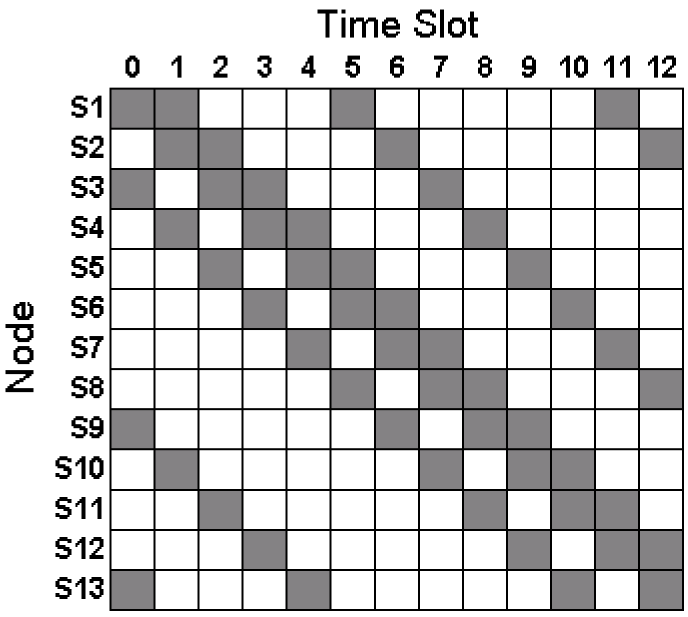
In recent years, cyclic symmetric block designs, related to the field of Combinatorics [25, 26], have slowly found their way into applications that solve real world problems. Apart from its known mathematical elegance, they have also showed promise in solving problems related to resource scheduling [28], data security [29], networking [27] and other applications [30].

A node is defined to be in the "Sleep" mode (or "Switched Off") when there are no data transmissions, reception of data and channel monitoring activities. Otherwise, it is in the "Awake" or "Active" mode (or "Switched On"). We do not discuss an intermediate state – the "Idle" mode where nodes are not completely switched off but continue to monitor the channel for packets while suppressing transmissions. It is known [18, 71, 89] from hardware behaviour that putting nodes to “Idle” mode is almost as energy costly as packet reception. Due to small transmission distances, power consumed while receiving data can at times be even higher than power consumed while transmitting packets [89]. In the “Idle” mode, both the computing subsystem consisting of a microprocessor or microcontroller and the communication subsystem consisting of a short range wireless communication component in a sensor node cannot be switched off if the channel is to be monitored. This explains why it is almost as energy consuming to operate in the “Idle” mode as it is in packet reception, except that control packet sizes that are processed in “Idle” mode are smaller than data packets. [89] therefore concludes that operating the radio in “Idle” mode does not provide any advantage in power and schemes that ignore this fact leads to fallacious savings in power consumption. The radio should be completely shut off (“sleep” mode) whenever possible, to obtain energy savings.

In block designs, we define a wakeup mechanism that associates each node with a slot of length L, termed as the wakeup schedule function (WSF). The WSF of a node v can be represented as a polynomial of order $L-1$ as

$$f_v(x) = \sum_{i=0}^{L-1} a_i x^i \tag{1}$$

where L is the length of the schedule, $a_i = \{0, 1\}$, $\forall i \in [0, L-1]$, and x is a placeholder. When $a_i = 1$, the node wakes up in slot i and sleeps otherwise. By definition, $M_\zeta = f_\zeta(1)$ is the total number of slots in which a node ζ is scheduled to be awake every L slots. The (v, k', λ)-design is defined as v schedules of length v slots each; with k' active slots in each schedule, and any two schedules have exactly λ overlapping active slots. A special class of cyclic designs exists, called the cyclic symmetric $(k^2+k+1,k+1,1)$ design, where $L = v = k^2+k+1$, $k' = k+1$, $\lambda = 1$ Existence of such a wakeup design is only guaranteed for values of k that are powers of a prime number (we discuss the possibility of overcoming this constraint in section 3.3). In the design, any schedule can be compactly represented using any single schedule with an offset because all slots (active or sleep) in a schedule are cyclic translations of a single schedule. There are (k+1) active slots in every schedule and there is exactly 1 overlap between any two schedules in the design. Figure 1 illustrates a cyclic symmetric (13,4,1) design. The choice of this class of design with $L = v = k^2+k+1$, $k' = k+1$, $\lambda = 1$ ensures that there is exactly one overlap between any two arbitrarily chosen schedules. Other polynomial choices for v are not known to provide such guarantees.


Time Slot

Node

	0	1	2	3	4	5	6	7	8	9	10	11	12
S1	▨	▨				▨						▨	
S2		▨	▨				▨						▨
S3	▨		▨	▨				▨					
S4		▨		▨	▨				▨				
S5			▨		▨	▨				▨			
S6				▨		▨	▨				▨		
S7					▨		▨	▨				▨	
S8						▨		▨	▨				▨
S9	▨						▨		▨	▨			
S10		▨						▨		▨	▨		
S11			▨						▨		▨	▨	
S12				▨						▨		▨	▨
S13	▨				▨						▨		▨

Figure 1: The Cyclic Symmetric (13,4,1) Block Design with k = 3.

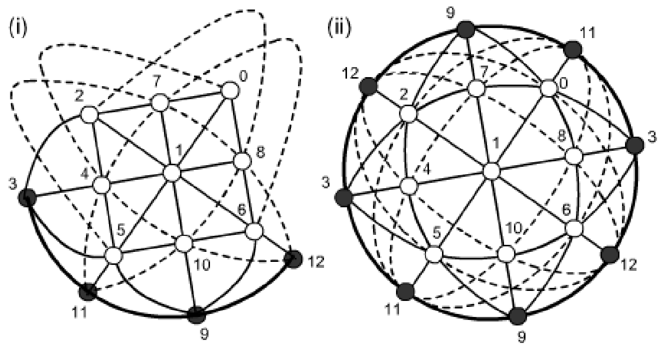


Figure 2: Illustrating geometric symmetry in the Cyclic Symmetric (13,4,1) design. Lines/Curves represent schedules and dots represent time slots. Numbers correspond to time slot numbers in Figure 1.

2.1.1 Symmetries of CSBD

The Cyclic Symmetric design can be interpreted and better understood in terms of symmetry. Symmetry often offers elegance and simplicity in implementation to solve complex real-world problems. We are therefore motivated to explicitly quantify symmetry for the cyclic symmetric (k^2+k+1,k+1,1) design in this section.

Consider any cyclic symmetric (k^2+k+1,k+1,1) design (see Figure 1). The symmetric property of such designs can be stated as follows:

Symmetry 1. The number of active schedules at any time slot is equal to the number of active time slots in any schedule.

Symmetry 1 can also be restated in terms of energy, where the total amount of awake energy consumed by all unique schedules in the design at any time slot is equal to the total amount of awake energy consumed by any schedule in one cycle. The duality of the terms “time slot” and “schedule” used in *Symmetry 1* is also revealed, for they can be interchanged. This duality is, in fact, a known principle in projective planes finite geometry [26, 31], which our designs are also related to. To visualize the symmetry, consider the following mapping:

- Every unique time slot is mapped to a unique point.
- Every unique schedule is mapped to a unique line.

It is therefore required that every line should contain k+1 distinct points and every point must lie on k+1 distinct lines. Note that the axioms of Finite Geometry are very different from those of Euclidean Geometry. It is beyond the scope of this thesis to discuss these axioms, but briefly, there is no measure of distance and there are only a finite number of points in Finite Geometry. A point can only be defined when two lines intersect. By the term “line”, a line need not be a straight line or of finite length. Therefore, circles and curves are to be defined as lines. In particular, the projective plane has no parallel lines and therefore, any two lines in the plane must intersect (related to *Symmetry 3,* which we shall describe later).

Figure 2 shows geometric illustrations of the cyclic symmetric (13,4,1) design. White and black dots in the figure represent a total of 13 points and the 13 lines are also printed dotted, solid, or solid-bold for clarity. The numbers are labeled to correspond to time slots in Figure 1. It may first appear that there are a lot of line intersections in Figure 2, but only the dots (white and black) are to be interpreted as real intersection points. Figure 2(i) shows that every line passes through exactly 4 points and every point lies on exactly 4 lines (because k = 3). This figure however, still does not exhibit sufficient visual symmetry. Suppose we define the solid-bold line to be at infinity, in the form of an outer circle, as shown in Figure 2(ii). In addition, each pair of antipodal points on this outer circle corresponds to just one point (these points are shown as black dots). Figure 2(i) is therefore equivalent to Figure 2(ii), but with the latter exhibiting much more visual symmetry. Figure 3 further illustrates the visual symmetry of other designs that have low orders, namely k=2 and k=4 respectively.

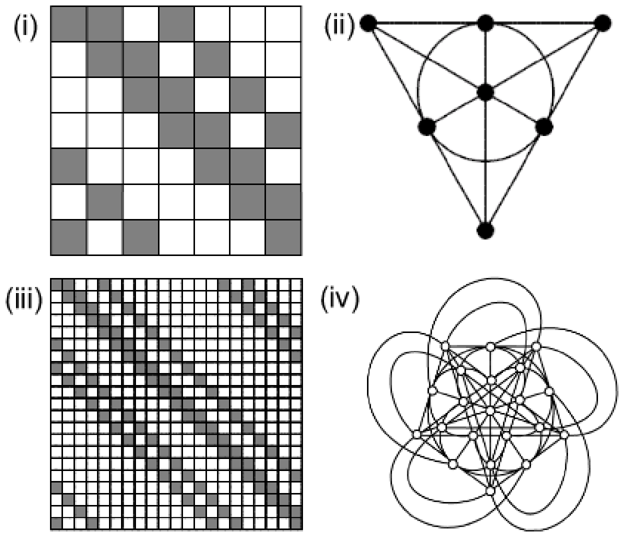


Figure 3: Illustrating geometric symmetry. (i) The Cyclic Symmetric (7,3,1). (ii) Symmetry of (7,3,1). (iii) The Cyclic Symmetric (21,5,1). (iv) Symmetry of (21,5,1).

In this thesis, we choose to use the term "Symmetry" in a broader sense to refer to resulting invariance (or self-similarity) of certain properties in the schedules after a defined set of mathematical operations has been applied.

Consider any arbitrary schedule in the design and denote a wakeup slot in the schedule as "1" and a sleep slot as "0". Let the r^{th} schedule be generated by r shifts of the original schedule and put the r^{th} schedule in the r^{th} row of a (k^2+k+1) by (k^2+k+1) square matrix R, where $r < (k^2+k+1)$. R is then an incidence circulant matrix. We further define a row vector R_r as the elements of the r^{th} row of R and a column vector R_q as the elements of q^{th} row of R, with $r < (k^2+k+1)$. Denote U to be the set of all schedules (rows of matrix R) in the cyclic symmetric ($k^2+k+1,k+1,1$) design space. We state:

Symmetry 2a. The cyclic shift of any U_1 belonging to the set U is always another schedule U_2 also belonging to the set U.

Symmetry 2b. The transpose of the circulant matrix R is itself another circulant matrix.

Symmetry 3. The matrix product of R_r with R_q is always the same and equal to unity for all $r \neq q$.

Because of *Symmetry 3*, all lines in the projective plane must therefore intersect exactly at only one distinct point. These symmetries provide the "hidden forces" for solutions that employ them to solve different aspects of the energy conservation problem in wireless networks.

2.2. Characteristics of CSBD

In the description of these properties, the term "schedule" can also be interpreted as "node" because each node operates one schedule from the design. Let T_{slot} be the slot time and T_{cycle} be one cycle time of the design.

Lemma 1. Let T_{awake} be the longest duration of continuous active slots in a cyclic symmetric (k^2+k+1, $k+1$, 1) schedule. Then, $T_{awake} = 2T_{slot}$.

Proof: Suppose there were no continuous (adjacent) awake slots in a schedule for at least 2 time slots, a cyclic shift of this schedule would not generate 1 overlap in any slots between these two schedules and would contradict the definition of the design. Similarly, if there were more than 2 continuous time slots that the schedule dictates the sensor to be stay awake, a cyclic shift would generate more than 1 overlap, contradictory to the definition again. Hence, $T_{awake} = 2T_{slot}$. ■

Lemma 2. There exists only one T_{awake} in any cyclic symmetric (k^2+k+1, k+1, 1) design.

Proof: Suppose there is more than one longest continuous duration of two active slots (because $T_{awake} = 2T_{slot}$ by Lemma 1), in a schedule. A cyclic shift of this schedule would generate more than one overlap of awake slots between these two schedules. Therefore, by proof of contradiction, there is only one such longest continuous duration of $2T_{slot}$ in a cyclic symmetric schedule. ■

Lemma 3. There are exactly one duration of continuous active slots of length $2T_{slot}$ and exactly (k-1) active slots of length T_{slot} in any cyclic symmetric (k^2+k+1, k+1, 1) design.

Proof: This follows immediately from Lemmas 1 and 2. ■

Lemma 4. The length of any durations of continuous sleep slots from a selected schedule in a cyclic symmetric (k^2+k+1, k+1, 1) design is unique within that schedule.

Proof: We need to prove that in any schedule, there exists no two continuous durations of sleep slots that are of equal length. Suppose we assume that there exists two continuous durations of sleep slots that are of equal length, and we label them as sleep duration *Dur1* and *Dur2*. Since all schedules in the design are cyclic shifts of each other, there exist a finite number of shifts such that *Dur1* and *Dur2* will coincide. This implies that the two schedules that *Dur1* and *Dur2* coincide will have two overlapping wakeup slots, and this contradicts the definition of the design. Hence, the result. ■

Lemma 5. Let T_{sleep} be the longest duration of continuous sleep slots in any cyclic symmetric (k^2+k+1, k+1, 1) design. Then, T_{sleep} is upper bounded by $T_{sleep,\max} = \frac{1}{2}k(k+1)T_{slot}$.

Proof: To find the upper bound for T_{sleep}, it is necessary to arrange all (k+1) awake slots as close to each other as possible in a total of (k^2+k+1) empty slots. By Lemmas 3 and 4, this is only possible with the arrangement of an increasing number of sleep slots between every duration of continuous awake slots. Since there exists only one longest duration of awake slots of length 2T_{slot} with all other active slots lasting only T_{slot}, and the integer function that generates an increasing number of sleep slots between them is increasing only at the smallest rate when it is starting from 1 sleep slot with an incremental step of also 1 sleep slot. We get:

$$T_{sleep,\max} = \left\{\left(k^2 + k + 1\right) - \left(2 + \sum_{i=1}^{k-1} 1 + \sum_{i=1}^{k-1} i\right)\right\} T_{slot} \quad (2)$$

$$T_{sleep,\max} = \left\{\left(k^2 + k + 1\right) - \left(2 + \sum_{i=1}^{k-1} 1 + \sum_{i=1}^{k-1} i\right)\right\} T_{slot} \quad (3)$$

$$T_{sleep,\max} = \frac{1}{2}k(k+1)T_{slot} \quad (4)$$

■

Corollary 5.1. $T_{sleep,\max} < \frac{1}{2}T_{cycle}$.

This is because $\frac{1}{2}k(k+1)T_{slot} < \frac{1}{2}\left(k^2 + k + 1\right)T_{slot}$ for all k > 0. ■

Lemma 6. Consider any T_{sleep} duration in any schedule from a cyclic symmetric (k^2+k+1, k+1, 1) design. All other schedules in the design (other than the schedule under consideration) have at least one wakeup active slot during T_{sleep}.

Proof: By definition, T_{sleep} is the longest duration of continuous sleep slots in any schedule. Suppose that there exists a schedule S_i (other than the schedule under consideration) with no wakeup slot during T_{sleep}, then two cases can occur:

- Case 1: S_i has a continuous duration of sleep slots exactly equal to T_{sleep}.
- Case 2: S_i has a continuous duration of sleep slots longer than T_{sleep}.

For case 1, S_i would then have at least two overlaps with the original schedule. This contradicts the definition of the (k^2+k+1,k+1,1) design of exactly one overlap between schedules.

Since S_i must be some cyclic shift of the original schedule, case 2 contradicts the definition that T_{sleep} is the longest duration of continuous sleep slots in the original schedule under consideration. Therefore, there exists no schedule in the design that would not wake up at least once in the duration of T_{sleep}. ■

Lemma 7. All schedules from the cyclic symmetric (k^2+k+1, k+1, 1) design have at least one awake slot within a time duration of $\frac{1}{2}(k^2+k+2)T_{slot} \approx \frac{1}{2}T_{cycle}$.

Proof: By Lemma 6, all schedules (except the schedule under consideration) must have been at least one awake slot within T_{sleep}. Since T_{sleep} is upper-bounded by $\frac{1}{2}k(k+1)T_{slot}$ in Lemma 5, all schedules must have at least one awake slot within $\frac{1}{2}k(k+1)T_{slot}$. To include the schedule under consideration, an additional T_{slot} is required. Hence, all schedules have at least one awake slot within a time duration of

$$\frac{1}{2}k(k+1)T_{slot} + T_{slot} = \frac{1}{2}(k^2+k+2)T_{slot} \approx \frac{1}{2}T_{cycle}. \blacksquare$$

These fundamental properties of CSBD serve to provide further insights into network connectivity (section 2.4), sensing coverage (section Chapter 1: 3.1.1) and query waiting delays (section 4.1.1) in sensor networks which we shall investigate in turn.

2.3. Asynchronous Neighbour Discovery and Data Transmissions

Sensor nodes, each using one schedule from the chosen CSBD set, are required to discover their immediate one-hop neighbourhood for the purpose of bookkeeping and inference (e.g. node failures). Since any two unique schedules have exactly one overlap “Awake” slot within one time cycle, the opportunities for neighbour discovery is guaranteed with a cycle. (Note that for nodes using the same schedules in the CSBD set, there are k+1 slot opportunities to discover each other within one time cycle). We further adopt the notion of using BEACON messages [27] to advertise the presence of a node to its immediate neighbours as illustrated in Figure 4. BEACON messages are advertised at the beginning of each “Awake” slot in the schedule at slot times t=0, t=1, t=5 and t=11. We illustrate later (section 2.4) that although time slots appear slotted, neighbour discovery is still guaranteed when these slot times are misaligned with its neighbour’s.

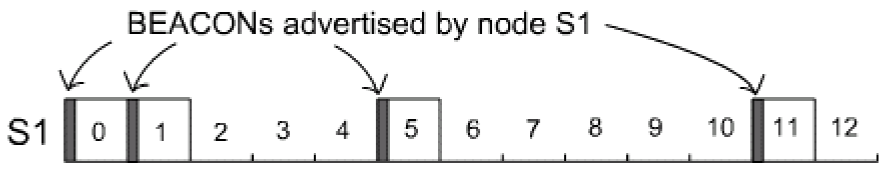


Figure 4: Illustrating BEACON messages as discussed in Zheng’s work [27] for neighbour discovery.

We highlight that the work in [27] is designed for mobile nodes where the set of neighbour nodes with respect to an arbitrary node change very often. Periodic BEACON messages are therefore very important to update the set of new neighbours every cycle. In our context for static sensor nodes or nodes with limited mobility, these BEACON messages are usually only important during the initial neighbour discovery phase when sensors are first deployed. Since the network topology does not usually change rapidly with time, the use of such BEACONs can be largely reduced after deployment.

In fact, we propose an “On-Demand” Neighbour Discovery (ODND) mechanism where a sensor node only transmits BEACONs under certain predefined condition(s). One such predefined condition can be the availability of loss-sensitive data to transmit to its neighbours. Another predefined condition can be based on time elapsed since the last neighbour discovery event. A combination of these conditions can also be implemented. For the time duration between successive neighbour discovery events, sensor nodes may assume that their previous sets of discovered neighbours remain valid. Note that ODND will take two time cycles to complete between any two neighbouring sensor nodes.

After neighbour discovery, an arbitrary sensor node may now transmit data to the set of neighbour nodes that they “hear” BEACON messages from. It is also possible to adopt the approach as in [27], where nodes may optionally send an “Awake Request (AREQ)” signal to its neighbours to request them to stay “Awake” for the next subsequent time slot (if they are scheduled into “Sleep” mode in the next slot) if data transmissions cannot be completed within the current time slot. This can happen when the assigned T_{slot} value is small or when traffic load is high. However, the receiver node may still reject such an AREQ request if its battery energy is low or for some other reasons. In [27], these AREQ requests are made on a per-slot basis for power control and management purposes. In the rest of this thesis, although we have implemented AREQ packets for our simulations, we have largely ignored the effect of clock synchronization mismatches to simplify our analytical work.

2.4. Network Connectivity

When a sensor network is deployed, its maximum or full network connectivity is to be determined by its physical arrangement of sensor nodes in the field when all nodes are in the “Awake” mode. Our main concern in network connectivity is therefore restricted to its preservation as nodes are switched off and on based on CSBD schedules. In [27], the authors introduced the concept of a network being connected within some finite time, instead of being connected at all times. We formally define:

Definition 2.4.1 A network of nodes is (n,T)-connected if there exists at least one path that connects any two nodes in the network within a time duration of T when (n-1) nodes (and their incident links) are removed.

Definition 2.4.2. Full connectivity is defined to be the maximum connectivity achievable when all nodes are awake.

Now, let N_G be a network of sensor nodes. Assume that the original network graph, G, where all nodes are awake at the same time, is α-connected. G is said to be α-connected if any two nodes in the network remain connected when any (α - 1) nodes and their incident links are removed (no time constraint). Nodes in N_G employ any arbitrarily (randomly) chosen schedule from the cyclic symmetric ($k^2+k+1,k+1,1$) design with a slot time of T_{slot}. We assume that sensor network nodes are static, and we have:

Theorem 2.4.1. The network N_G is (α ,$N_{hop}T_{cycle}$)-connected where T_{cycle} = (k^2+k+1)T_{slot} and N_{hop} is the maximum number of hops between any two nodes in the network dictated by the routing algorithm.

Proof: Let T_{cycle} be the cycle time for the cyclic symmetric ($k^2+k+1,k+1,1$) design. With a total of k^2+k+1 slots in each cycle, $T_{cycle} = (k^2+k+1)T_{slot}$. Since there is exactly one overlap between any two arbitrarily chosen schedules in the design, the longest wait duration to travel from one node to a neighbouring node is T_{cycle}. Assume there are a maximum of N_{hop} hops between any two nodes in the network, it takes a maximum time duration of $N_{hop}T_{cycle}$ to move between any two nodes in the network because network topology does not change within this time duration. Since G is α-connected, and the union of all network graphs generated by N_G within one T_{cycle} ($\le N_{hop}T_{cycle}$) is the graph G itself, N_G must also be α-connected within the maximum time duration of $N_{hop}T_{cycle}$. Hence, N_G is (α,$N_{hop}T_{cycle}$)-connected. ■

Therefore, the network connectivity of a system using CSBD wakeup will be preserved within T_{cycle} per hop. If information is to traverse N_{hop} hops, then network connectivity is preserved within $N_{hop}T_{cycle}$. Indeed, depending on the application requirement, the value of T_{cycle} may be tuned accordingly so that the desired network connectivity can be achieved.

The network connectivity of CSBD remains preserved even if time clocks are not synchronized amongst the individual sensor nodes. This is a consequence of the *symmetries* we have described in Section 2.1. Zheng [27] showed that neighbour nodes are always able to discover each other within bounded time even if time slots in the schedules are misaligned. Therefore, the network remains connected in bounded time despite non-synchronized clocks.

Theorem 2.4.2. Consider any two neighbour nodes X and Y in the network operating schedules S_X and S_Y from the same cyclic symmetric ($k^2+k+1,k+1,1$) design. Nodes X and Y can always discover each other within bounded time for any arbitrary time offset of the schedule S_Y from S_X, or vice versa.

Proof: Refer to [27]. ■

We illustrate this pictorially in Figure 5. In Zheng's work for MANETs, every node transmits a BEACON message at the beginning of every "Awake" slot for neighbour discovery. This frequent BEACON messages are necessary for a continuously mobile node network because neighbour nodes change very often. We have argued that in the sensor network context, where sensor nodes are either always static or have limited mobility, such BEACON messages can be largely reduced using ODND (section 2.3). The idea behind time-asynchronous neighbour discovery involves two neighbour nodes, such as S1 and S2 in Figure 5, with a time offset in one of the schedules with respect to the other due to non-synchronized clocks. Because schedules are cyclic in nature, both nodes S1 and S2 can discover each other within T_{cycle}. With respect to S1's clock, S2 “hears” the BEACON from S1 at time slot 0 and S1 “hears” the BEACON from S2 at time slot 1, and both nodes discover each other. If clock asynchrony or clock drifts are severe, this will affect the amount of remaining active time for data transmissions. In such cases, we have discussed in section 2.3 that nodes send “Awake Request” (AREQ) packets to signal to the recipient node to stay awake for one or more subsequent time slots to complete the necessary data transmissions.

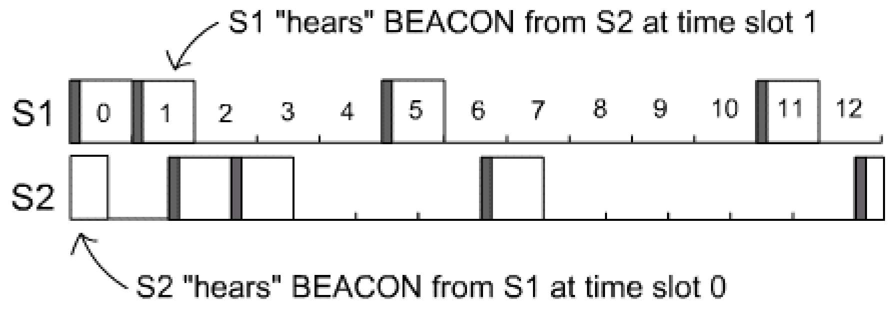

S1 "hears" BEACON from S2 at time slot 1
S1
0
1
2
3
4
5
6
7
8
9
10
11
12
S2
S2 "hears" BEACON from S1 at time slot 0

Figure 5: Illustrating asynchronous neighbour discovery with misaligned time slots.

AREQ packets, however, may not be an ideal solution if data flows across the node are large, especially if they behave like gateway nodes in the network where network traffic often traverses. AREQ packets may cause such gateway nodes to always expend more energy by having to always stay awake in their usual sleep slots. A simple way to solve this is for such gateway nodes to broadcast a local timestamp piggy-backed in the BEACONs within one time cycle on a regular basis so that its immediate 1-hop neighbor nodes can be approximately time-synchronized with such gateways. Time-synchronization of other nodes is less critical between other senders and receivers. It is important to realize that such coarse and approximate time-synchronization is an attempt to reduce unnecessary energy consumption caused by AREQ packets and is not a necessity or requirement for the operation of our wakeup schemes.

2.5. Implementation Costs

The implementation costs of CSBD-based wakeup schemes are low compared to many other existing schemes. The main advantage of CSBD lies in its deterministic wakeup, which incurs few computational and communication overheads. Yet, symmetries in its design (section 2.1) offer additional features such as bounded-time connectivity discussed in section 2.4. We shall also show in later chapters that CSBD offers other features including bounded-time coverage (section 3.1.1) and bounded-time query waiting delays (section 4.1.1). The selection and distribution of schedules from the CSBD to sensor nodes is different for different applications requirements, and we shall discuss them later in their appropriate chapters. Compared to RIS [13], our proposed method does not require distributed time synchronization techniques that are expected to be costly over a large scale. Compared to CDS-based methods and information-configured schemes, CSBD-based schemes are expected to have lower operational computation and communication overheads. Unlike MANETs, sensor network nodes are often static and asynchronous neighbour discovery costs in CSBD is largely reduced by ODND (discussed in Section 2.3). Compared to paging wakeup methods such as STEM [17], PAMAS [19], and even PECAS [21], our proposed solution do not require any dual-communication channel to function, thereby simplifying the hardware architecture and reducing implementation costs for deployment. Moreover, CSBD-based systems put nodes completely to the "Sleep" mode instead of merely the "Idle" mode. This simplifies implementation code complexity and saves more energy.

One other alternative approach to solve this energy conservation problem is by computing optimal schedules at a fixed one-time cost without using CSBD or other designs as a starting point and subsequently distributes these optimal schedules to the sensors. The drawback of such an approach is often the inability for optimal schedules to adapt to dynamical changes, such as node failures. Recomputing new optimal schedules for every node failure can be prohibitively costly at runtime. While our CSBD-based schemes are designed for static sensor networks, they are also capable of handling nodes with limited mobility, which would add an additional dimension of complexity when solving for optimal schemes. Moreover, we shall show later that one particular CSBD-based scheme offer certain characteristics such as zero query waiting delays (section 4.1.1) which can be considered as a form of optimal schedule under given conditions.

We provide more detailed overheads analysis with our simulation results in subsequent chapters.

2.6. Summary

In this chapter, we discussed the symmetries of the Cyclic Symmetric Block Design (CSBD) and investigated its fundamental characteristics. In particular, these characteristics are particularly important for understanding network connectivity, sensing coverage and query waiting delays in the context of sensor networks. We have discussed neighbour discovery and data transmission issues in CSBD-based wakeup schemes. We introduce an "On-Demand" Neighbour Discovery (ODND) scheme for sensor networks. We have provided an analysis of network connectivity based on CSBD in this chapter as it is generally applicable to our works in chapters 3 and 4. Similarly, we provided a discussion on implementation costs of CSBD systems in a more general context. We defer the discussion of sensing coverage and query waiting delays to later chapters because their analysis is more application specific.

Chapter 3: Agent-Based Sensor Networks

Several sensor network systems have been proposed with the use of “agents” to monitor and track events in the network. As data are being continuously generated from sensors, these data measurements may be correlated to each other as they may be related by the same event of interest. Mobile agent-based systems [33, 34, 35] gained their reputation in applications where the event is persistent in the network and propagates from one part of the network to another. A mobile event is usually characterized by a target, and can include a moving vehicle, a human target, a hurricane, an ant raid event by the Ceraphachys Ant species, etc. Applications may therefore include target tracking, intrusion detection, target interception, disaster management, wildlife monitoring etc. As the event is mobile and moves in the network, the mobile agent also moves with the event real-time in an attempt to perform the necessary data collection from sensors, event association, event classification, event identification, and other event management functions. We provide a formal definition of an “Agent”:

Definition 3.1. An Agent is a piece of information, data or software code that uniquely identifies a target in the sensor network and moves within some distance of the target as it traverses the network.

In [36], L. Gan et. al. proposed the use of autonomous mobile data agents in their sensor networks to transport data independently to a sink. In [34], a mobile agent-based signal and information processing computing model is proposed for sensor networks. A multi-agent framework is also proposed for real-time tracking in [37], where the influence of agent behaviour on tracking accuracies have been analysed. For distributed multisensor data fusion, the use of mobile agents also seems to be an effective way to save bandwidth and reduce latencies [38]. In fact, D. B. Lange et. al. [39] spelled out seven good reasons why mobile agents should be used. However, research topics involving the interactions between an underlying energy conserving wakeup scheme and agent-based sensor network systems have been generally lacking.

Since a majority of the agent-based literature has been focused on target tracking, we therefore choose to limit the rest of our discussion in this chapter to this application although our work can equally apply well to any agent-based application.

In this chapter, we assume that tracking is to be performed real-time by agents and sensor deployment is required to be dense (e.g. at least 3-covered for tracking applications to resolve target location in 2D space). As such, information delays and collision probability becomes important factors to consider in this application scenario. Section 3.1.1 discusses CSBD-based sensing coverage issues in sensor networks. Section 3.1.2 discusses delay issues in CSBD while section 3.1.3 discusses schedule diversity issues related to collision probability in CSBD. Section 3.1.4 further analyzes node lifetimes in CSBD networks and section 3.3 briefly outlines a typical data fusion algorithm based on the particle filtering approach. Both sections 3.2 and 3.4 discuss implementation issues with CSBD systems for tracking under the assumption that maximum target speed v_{max} is known before deployment. Section 3.5 then improves CSBD wakeup by introducing a scheme that can be deployed without prior knowledge of v_{max}. We summarise our work in this chapter in section 3.6.

3.1. Key Design Considerations

3.1.1 Sensing Coverage

Apart from network connectedness (section 2.4), another key function of sensor networks is to sense. During deployment of sensors, the sensing coverage of the entire sensor network as a whole is an important consideration for the practical purposes of applications. Numerous wakeup schemes [13, 40, 41, 42] have been proposed to optimize sensing opportunities while conserving network energy. We have previously described the Randomized Independent Scheduling (RIS) scheme [13] in Section 1.3.1 that ensures asymptotic *m*-coverage. The Sponsored Sector Scheduling (SSS) scheme [40] is also capable of preserving existing sensing coverage in the network by allowing a sensor node to turn itself off only if its sensing area is completely covered by other neighbour sensors. In [41], the Maximization of Sensor Network Life (MSNL) scheme was formulated as a maximization problem that maximizes network lifetime while promising to preserve *m*-coverage. The Lightweight Deployment-Aware Scheduling (LDAS) [42] challenged the need of GPS and location information in the network and proposed to provide statistical guarantees to sensing coverage when such location information is not available in the network.

All these existing schemes, however, define sensing coverage as a requirement at all time instants. While this may be useful for some applications, it tends to be excessive for most. For instance, a temperature monitoring or event recording application may not require a region to be always *m*-covered. It is often sufficient when such guarantees are provided within some known bounded time. Even for real-time applications such as target tracking, an *m*-covered network by *m* different sensors ($m \geq 3$ to resolve target location in 2D space) within a tolerable time duration smaller than the time resolution of data fusion algorithms [33, 35, 43] or user requirement at the application level is sufficient. The experience with many data fusion algorithms [43, 44] is that they only require sensing data at discrete time steps based on an iterative measurement-prediction approach. Such schemes include [45] Kalman Filtering, Extended Kalman Filtering, Particle Filtering and other Bayesian-based state estimation approaches. This suggests that it is sufficient that sensing coverage of the network is preserved within some finite time duration, and not for all time instants. We shall provide more discussion on data fusion algorithms in a later section (section 3.2).

Definition 3.1.1.1. An area A is (m,T)-covered if every point in A is always covered by the sensing coverage radii of at least m different sensor nodes within a time duration of T.

Let N_G be a network of sensor nodes deployed in some 2D geographical space. Let the region covered by all nodes in N_G when awake be denoted as A, and assume that A is β-covered (ignoring edge effects of A). Nodes in N_G employ any arbitrarily (randomly) chosen schedule from the cyclic symmetric ($k^2+k+1,k+1,1$) design with a slot time of T_{slot}.

Theorem 3.1.1.1. Region A is $\left(\beta, \frac{1}{2}(k^2+k+2)T_{slot}\right)$*-covered.*

Proof: By Lemma 7 (see section 2.2), all sensor nodes are awake at least once within a time duration of $T_\beta = \frac{1}{2}(k^2+k+2)T_{slot}$. Since region A is β-covered by nodes in N_G when all are awake at the same time, any point P in A must be covered by β different sensors at any one time. Since these β different sensors must employ cyclic symmetric ($k^2+k+1,k+1,1$) design, they must have woken up at least once within T_β. Region A is therefore $\left(\beta, \frac{1}{2}(k^2+k+2)T_{slot}\right)$-covered. ■

What *Theorem 3.1.1.1* implies, is that if an area A is β-covered by a network of sensor nodes, N_G, that are always awake, then A is also necessarily $\left(\beta, \frac{1}{2}(k^2+k+2)T_{slot}\right)$– covered by that same network of nodes operating the cyclic symmetric ($k^2+k+1,k+1,1$) design wakeup schedules.

Definition 3.1.1.2. Full coverage is defined to be the maximum coverage achievable when all sensor nodes are awake.

Corollary 3.1.1.1. For a cyclic symmetric ($k^2+k+1,k+1,1$) design wakeup network, it takes approximately $2N_{hop}$ times longer to guarantee full connectivity than to guarantee full coverage.

Proof: This follows from the result of Theorems *3.1.1.*1 and 2.4.1 since

$$\frac{N_{hop}(k^2+k+1)T_{slot}}{[(k^2+k+2)T_{slot}]/2} \approx 2N_{hop}$$.■

Corollary 3.1.1.2. For a cyclic symmetric (k^2+k+1,k+1,1) design wakeup network, it takes approximately twice as long to guarantee information propagation across one network hop than to guarantee full coverage.

Proof: The proof follows from Corollary *3.1.1.1* and set $N_{hop} = 1$. ■

The convenience of our results is now obvious. For any application that requires preservation of the original β-coverage in sensing and resides at most N_{hop} hops away from the event, the delay, D_u, experienced is approximately upper-bounded by:

$$D_u \approx \left(N_{hop} + \frac{1}{2} \right) T_{cycle} \tag{5}$$

Suppose an application can only tolerate a time resolution of T_{res}, then the following condition must be satisfied:

$$T_{res} \geq \left(N_{hop} + \frac{1}{2} \right) \left(k^2 + k + 1 \right) T_{slot} \tag{6}$$

Theorem 3.1.1.2. A β-covered network implies a β-connected network, if $R_C \geq 2R_S$, where R_C is the communication range of sensors and R_S is their sensing coverage radius.

Proof: The proof can be found in [46]. ■

Theorem 3.1.1.3. For a cyclic symmetric (k^2+k+1,k+1,1) design wakeup network that is β-covered when all nodes are awake, it is also:

$\left(\beta, [(k^2 + k + 2) T_{slot}]/2 \right)$ *-covered, and* $\left(\beta, N_{hop} (k^2 + k + 1) T_{slot} \right)$ *-connected if* $R_C \geq 2R_S$.

Proof: The proof follows directly from Theorems 2.4.1, *3.1.1*.1 and *3.1.1*.2. ■

For the purpose of clarity, we term sensor nodes that employ CSBD for the objective of providing sensing coverage and connectivity within bounded time as Bounded-Time Covered/Connected (BTC) Nodes.

3.1.2 Delay

Apart from bounded-time coverage preservation (section *3.1.1*) and bounded-time connectivity preservation (section 2.4) requirements, another important consideration factor in target tracking is delay. The speed at which information (or agent) propagates from node to node may be slower than the speed of the mobile target as it traverses boundaries of nodes, because nodes are not always "awake". This therefore renders the target "untrackable", given that the agent speed is slower. This is actually a more specific form of the requirement where information travel per hop is upper-bounded by less than T_{cycle}. To be more specific, we state:

Theorem 3.1.2.1. Assume that sensor nodes operate wakeup schedules from the same cyclic symmetric ($k^2+k+1,k+1,1$) design. Data packets from one node to the next hop wait at most $T_{cyclic,sleep} = k(k+1)T_{slot}$ where T_{slot} is the slot time.

Proof: Since there is exactly one overlap "awake" slot between any two arbitrarily chosen schedules in the design, say at slot $S_{overlap}$, the longest time duration a data packet needs to wait is when it arrives at a time $S_{overlap}$ which has just elapsed. Hence, the longest wait duration for the next $S_{overlap}$ to arrive is $T_{cyclic,sleep} = (k^2+k+1)T_{slot} - T_{slot} = k(k+1)T_{slot}$ ■

We assume that sensors are randomly distributed in a geographical area of interest and that they are deployed such that the area of interest is sensing-covered to the required degree. Suppose that the communication range of sensors, R_C, is some factor ω of its sensing coverage radius R_S, where $\omega \in \Re$ and $\omega > 0$. Hence,

$$R_C = \omega R_S. \qquad (7)$$

A communication link is maintained between any two nodes whenever the condition $d \le \omega R_S$ is satisfied, where d denotes the distance between the two nodes.

Neighbour nodes need to wakeup sufficiently often to exchange tracking information and this depends on the time durations between overlapping active slots in the schedules of the neighbouring nodes.

Definition 3.1.2.1: The longest non-common wakeup time (LNWT) between any two schedules in a wakeup design is defined to be the longest time duration between any two nodes using schedules in a wakeup design such that both nodes are not awake at the same time.

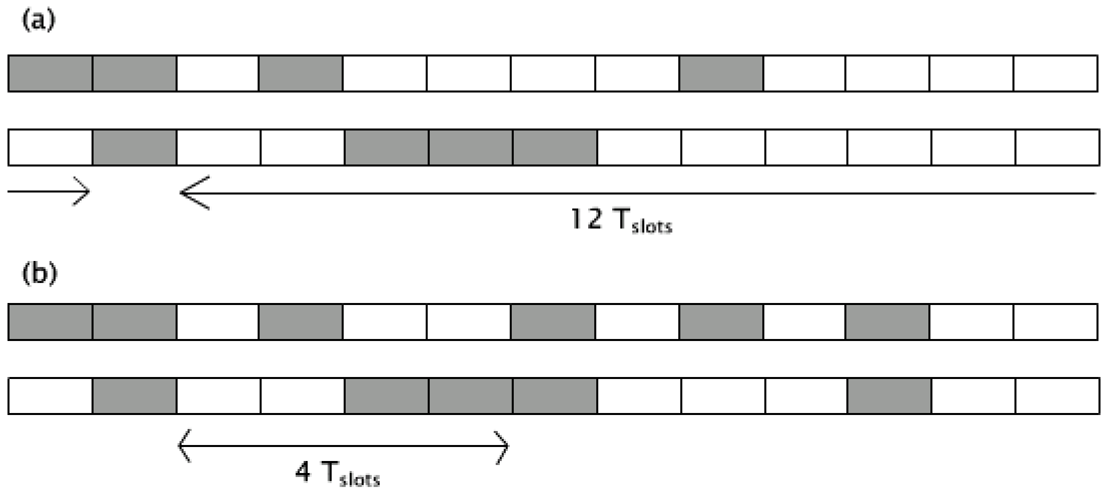


Figure 6: (a) LNWT = 12 T_{slot}. (b) LNWT = 4 T_{slot}.

As an illustration, Figure 6(a) shows that the LNWT between the two schedules is $12T_{slot}$ and Figure 6(b) shows the LNWT is $4T_{slot}$.

In the remaining of this section, we find the desired $LNWT_{track}$, the longest non-common wakeup time between any two schedules (nodes) given that targets can move no faster than a known maximum speed of v_{max}. Suppose the maximum distance between the Agent and the target is to be no more than H communication hops. We consider several cases for H.

<u>Case 1: H=1</u>

We assume that $R_C = \alpha R_S$, where α is a natural number ($\alpha \in \aleph$). For ease of explanation, we first consider the case $\alpha = 3$. It will be easy to see later that the result can also be extended to α > 3, but there will be no solution for α < 3. In Figure 7, the agent resides at an arbitrary node *i*. The objective is to ensure that the agent can be transferred to any one-hop neighbor node *j*, before the target leaves the sensing coverage radius of node *i*, within the longest possible time duration where both nodes are not awake at the same time. If this condition is met, all one-hop neighbours can then potentially receive the Agent before the target actually enters into their sensing regions. Note that it is not the focus of this thesis to discuss target velocity prediction or schemes that propose sending the same agent to multiple second hops to minimize tracking and target association errors. We assume that this is done by existing algorithms [47, 48].

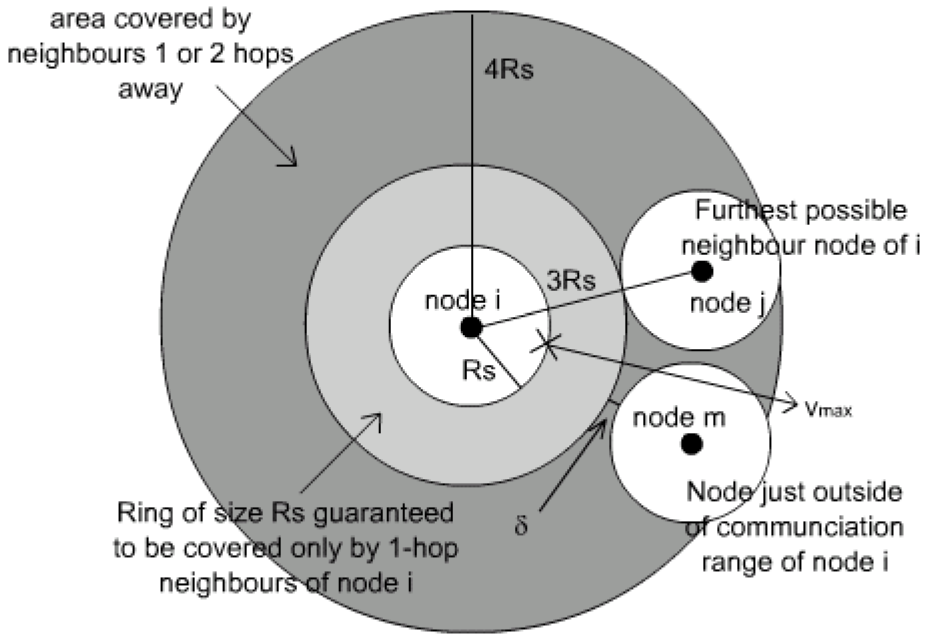

area covered by neighbours 1 or 2 hops away
4Rs
Furthest possible neighbour node of i
3Rs
node i
node j
Rs
Vmax
node m
Node just outside of communciation range of node i
Ring of size Rs guaranteed to be covered only by 1-hop neighbours of node i
δ

Figure 7: Illustrating the proof of a tracking delay bound.

Since $R_C = 3R_S$ and we assume the region is completely sensing-covered, there must exist a geographical ring of radius R_S that is covered only by one-hop neighbours of node *i* (Figure 7). We consider the worst case that a target's movement is such that it will result in the fastest possible departure from node *i*. For any moving target *X*, the worst case (i.e. the shortest time to leave the one-hop neighborhood of node *i*) happens when it is at the edge of sensing coverage of node *i* (i.e. at a distance R_S from the node), and the target moves at a maximum speed v_{max} in an outward radial direction away from node i. If the distance travelled by the target during the time when the two nodes are not both awake at the same time is more than R_S, the target will no longer be in the one-hop neighbourhood of node i:

$$v_{max} LNWT_{track} > R_S \tag{8}$$

Therefore, the target only remains "trackable" when the reverse condition is true:

$$v_{\max} \le \frac{R_S}{LNWT_{track}} \tag{9}$$

It is easy to see that the condition may be generalized for any α ≥ 3 such that $R_C = \alpha R_S$. Therefore,

$$LNWT_{track,\max} = \left\lfloor \frac{(\alpha - 2)R_S}{v_{\max} T_{slot}} \right\rfloor T_{slots} \text{ seconds} \tag{10}$$

or

$$LNWS_{track,\max} = \left\lfloor \frac{(\alpha - 2)R_S}{v_{\max} T_{slot}} \right\rfloor \text{slots} \tag{11}$$

where $\lfloor x \rfloor$ is the largest integer smaller or equal to x, and $LNWS_{track}$ is the largest number of slots in a tracking schedule such that both node i and any of its neighbour nodes do not have overlapping "awake" slots. For $\alpha \le 2$, no such bounds can be derived. We call this "The $\alpha \le 2$ Constraint" and we shall relax it later for cases when this condition cannot be fulfilled (section 3.3).

Case 2: H > 1

The extension of results to a general H-hop agent case is not straightforward. This is because of the infinite possibilities in deployment topologies. For instance, from Figure 7, it is clearly possible that when $H = 2$, the second hop node (node m) can be at a distance of just over $3R_S$ from node i, but still within communication range of node j. As $\delta \to 0$, the delay bound per hop can be written as:

$$LNWS_{track,\max} = \left\lfloor \frac{(\alpha - 2)R_S}{2v_{\max}T_{slot}} \right\rfloor \text{slots} \tag{12}$$

which is always smaller than (11). For H=3 and α=3, we prove that any third hop node from a reference node i cannot be within a distance of $4R_S$.

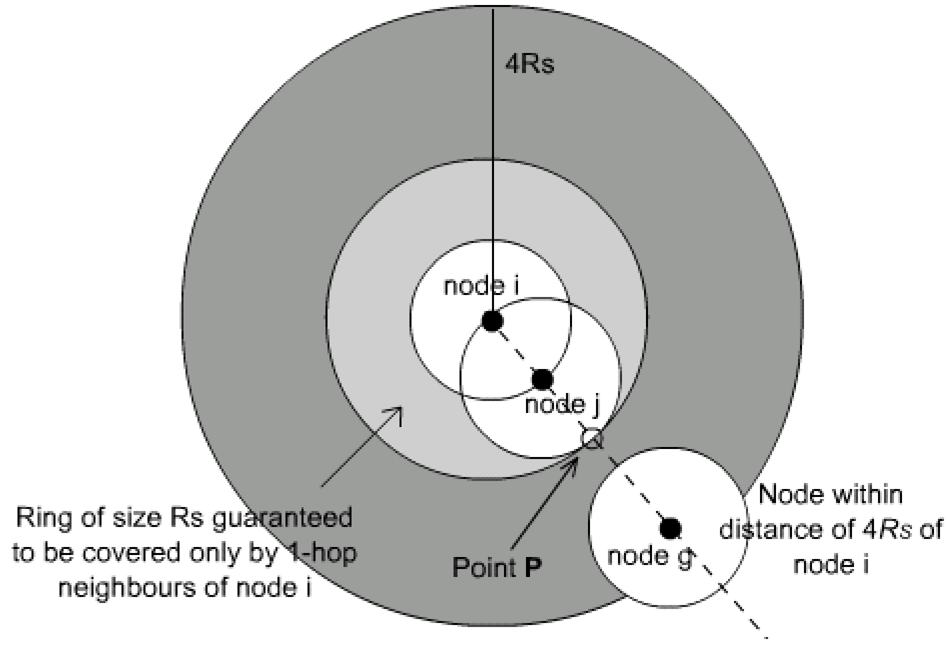

4Rs
node i
node j
Ring of size Rs guaranteed to be covered only by 1-hop neighbours of node i
Point P
node g
Node within distance of 4Rs of node i

Figure 8: Illustrating the proof of a third hop node outside 4Rs for α = 3.

Proof: Consider Figure 8. We shall prove by contradiction and first assume that there exists a third hop node g within a distance of $4R_S$ of any arbitrary node i. Let P be the point where the straight line joining nodes i and g intersect with the outer circumference of the ring of one-hop neighbours of node i. Since the ring must be covered only by one-hop neighbour nodes of i and that point P is part of this ring, there must exist some node j that will cover point P and one-hop away from node i. In order for node j to be both furthest from node g and cover point P at the same time, node j must be centered along the straight line joining both nodes g and i, and at a distance of less than or equal to R_S from point P. Since the communication radius of node j is also $3R_S$, node g would be in communication range of node j. Node g would therefore have been a second hop node, instead of a third hop node, and we arrive at a contradiction.

Therefore, any third hop node of node *i* must be at least 4*Rs* away. ■

By a similar proof, if $R_C = \alpha R_S$, where $\alpha > 2$, the 3rd hop node must be at least $2(\alpha - 1)R_S$ from node *i* because the one-hop covered ring size (linear radial distance from the inner circumference to the outer circumference of the ring) would have been $(\alpha - 2)R_S$. With this, the delay bound can also be derived appropriately.

For $H > 3$, it is usually beyond the practical scope of tracking agents to be too far away from the target and we shall not attempt to derive bounds for those cases. For simplicity, we focus only on the H=1 case from now on.

3.1.3 Schedule Diversity

Most agent-based systems, such as the target tracking application, usually involve a dense sensor deployment. It is therefore important to ensure that the choice of schedules by sensor nodes is diverse, and collision probability may be reduced. In this section, we consider schedule diversity issues in CSBD-based wakeup schemes for dense sensor networks and predict packet collision probabilities. In this section, we assume that Multiple Access Control (MAC) layer collision avoidance techniques are not available.

Consider any sensor network deployment of N_{deploy} nodes with a uniform deployment density of ρ_{deploy} (where ρ_{deploy} is the ratio of N_{deploy} to area of deployment), with each node randomly (arbitrarily) selecting a wakeup schedule from CSBD (cyclic symmetric (k^2+k+1,k+1,1) design). For small values of k, more sensor nodes will be using the same schedules per unit area. We term this as a lack of schedule diversity within that neighbourhood area. As more nodes operating the same schedules wakeup at the same times, more transmission collisions are expected, leading to either more packet retransmissions, thereby depleting more node energies, or leading to more frequent packet losses. Therefore, this illustrates the importance of diversity of schedules and the motivation for choosing larger values of k.

We assume a uniform distribution of nodes on a 2D geographical plane, and all nodes have a communication range R_C. The number of sensor nodes N_I, within a radius of R_C of any arbitrary reference node that have the potential of having packet collisions with that reference node is:

$$N_I \approx \pi\rho_{\text{deploy}} R_C^2. \tag{13}$$

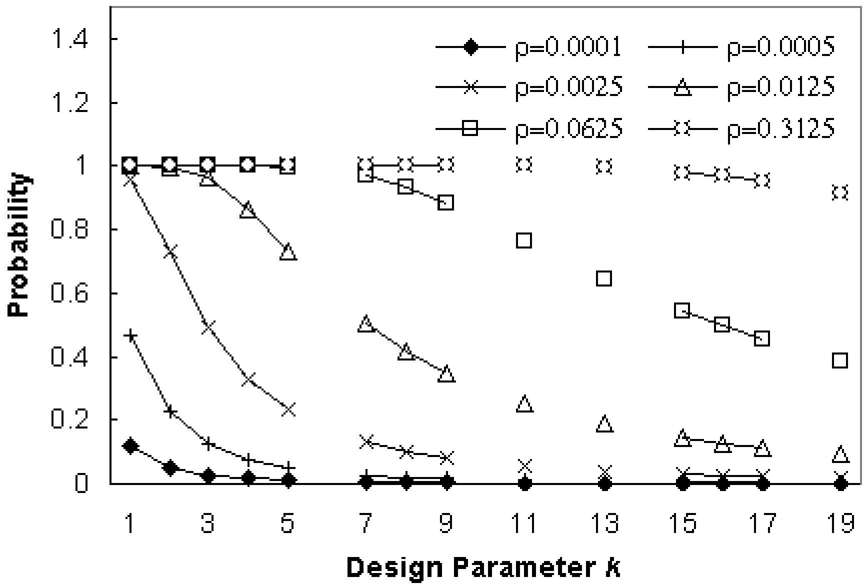


Figure 9: Collision Probability for different values of *k* with R_C=150m, p_{tx}=20%.

Let p_{tx} be the probability that a node will transmit a packet when it is in the awake mode. To simplify the analysis, we assume that whenever two nodes transmit in the same time slot (even if it is at different times within the same slot), a collision will occur. The probability of a collision is therefore:

$$P_{no.collision,N_I} = \left(p_{tx} \bullet \frac{k+1}{k^2+k+1}\right)\sum_{i=0}^{N_I}\left[\frac{N_I!}{i!(N_I-i)!}\right]^2\left[\frac{k^2}{k^2+k+1}\right]^{N_I-i}\left[\frac{k+1}{k^2+k+1}(1-p_{tx})\right]^i \tag{14}$$

We use an approximation instead by assuming that the diversity of schedules in a locality is sufficiently good so that the probability of more than two schedules transmitting at the same time within an overlapping active time slot is small and that p_{tx} is not large. Hence,

$$P_{collision,N_I} \approx 1 - \left[1 - \left(p_{tx} \bullet \frac{k+1}{k^2+k+1}\right)^2\right]^{\pi\rho_{deploy}R_C^2} \qquad (15)$$

Figure 9 shows that $P_{collision,N_I}$ decreases as *k* increases for different values of node deployment densities. The nulls in the graphs are a result of the prime power constraint which we have previously mentioned in section 2.1 (see section 3.3).

3.1.4 Node Lifetime

In this section, we show that the node lifetimes can be lower bounded because of the deterministic nature of the wakeup slots in the schedule. Assume that each awake time slot consumes E_{awake} units of energy (for simplicity, this also includes transmission energy), and the total energy in each node is E_{node}, then the maximum energy consumed per T_{cycle} per node is $(k+1)E_{awake}$ and the network lifetime, ζ_{life}, of each node is lower bounded by:

$$\varsigma_{life} \geq \varsigma_{life,LB} = \frac{(k^2+k+1)T_{slot}E_{node}}{(k+1)E_{awake}} \qquad (16)$$

Setting equality in (6) and substituting into (16), we get:

$$\varsigma_{life} \geq \varsigma_{life,LB} = \frac{T_{res}E_{node}}{(k+1)(N_{hop}+0.5)E_{awake}} \qquad (17)$$

In (17), $\varsigma_{life,LB}$ increases as *k* decreases for any fixed T_{res}.

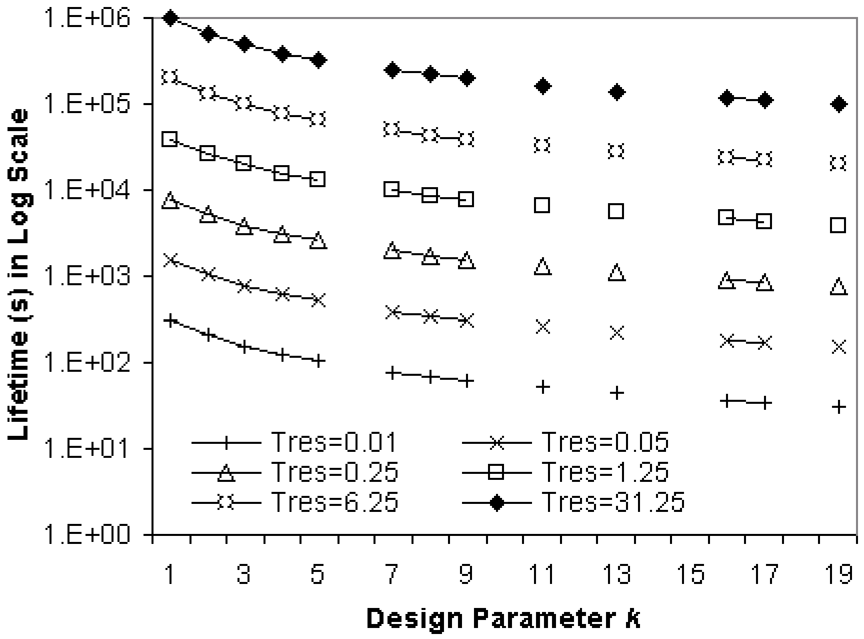


Figure 10: Theoretical node lifetime bound for different application time resolution requirements with N_{hop}=1.

Figure 10 illustrates this trend further for different application requirements of T_{res}. Again, the null values for certain *k* values are because of the prime power constraint (see section 2.1, 3.3).

3.2. Agent-Based Data Fusion for Target Tracking

Agent-based sensor network systems have been popularly employed for the data fusion problem in target tracking. The use of agents in data fusion offers the availability of a mobile fusion center, where data collected from different sensors and across different sensing modes can be fused all at once in a consistent manner for target identification, association, classification and localization. This allows researchers to focus on the signal processing data fusion aspect of the problem and make other simplifying assumptions such as small data latencies to the agent (fusion center). This is possible because the agent "follows" the event or the target where sensing data is generated and is expected to be only one or two hops away from the event (section *3.1.2*).

Tracking of targets is usually modeled as a dynamic system, by using Bayesian network representation where the state of the targets can be estimated using Maximum A Posterior (MAP). However, in most cases, the measurements and state distributions are non-Gaussian, and the dynamic system is not linear. The dynamic equation of Bayesian network representation therefore becomes not trackable. Recently, importance sampling, especially particle filtering [45], has been widely recognized to solve problems of this nature effectively. In [49, 50] Hu employed a signal propagation model to localize the target source, and then obtain the target location. Indeed, a similar problem and its accompanying solution exists in the field of heat transfer where the Markov chain Monte Carlo (MCMC) algorithm [45] is used to obtain estimates of the statistics of the unknown heat flux [51]. By using the MCMC sampling strategy it is possible to extend the Bayesian inference approach to inverse problems having high-dimensional, non-standard distribution, and/or complex distribution functions. These previous works therefore provide good methods for solving the target localization problem.

The problem of fusing data from multiple sensors and across multiple modalities for tracking in sensor networks remains largely unsolved. We formulated the target tracking problem using inverse source localization in the context of sensor networks where particle filtering is used to implement the tracking [43]. As the primary focus of this thesis is sensor energy conservation rather than data fusion algorithms, this section only briefly outlines this application layer algorithm for the following purposes:

- To appreciate the fusion approach of a typical distributed algorithm for target tracking
- To demonstrate that such algorithms provide a convenient way to define the time-resolution accuracy of the tracking, T_{res}, that can be used in (6)

- To provide details of the application layer tracking algorithm that is implemented in our simulations where results are obtained

Particle filters can be thought of as a mathematical tool for the probabilistic estimation of a physical state in the presence of noise (that can be non-Gaussian), in our case, the state of interest is target location. However, the physical state cannot be measured directly, but can only be inferred through measurements, such as acoustic and/or seismic waves detections, etc. at the sensing detector (In our case, we infer distance from measured acoustic and seismic wave energies in each sensor). The relationship between measurements and the real physical state are related or represented by likelihood probability functions in particle filters. A measurement in any one sensing mode is therefore able to influence/predict the state according to a weighting function, which is given by the product of the measurement likelihood functions. The physical state is then, also represented by a probability function, which changes and gets refined with each new measurement by updating the weighting function in the filtering algorithm. To estimate the continuous probability function representing the state, particles are used to discretize the probability function over the state space. Each particle can therefore be thought of as the state having a certain value, with a certain probability. The expectation of the value of the state can then be computed by averaging over all particles.

We summarize the particle filter implementation of Bayesian filter for target tracking in sensor networks using acoustic and seismic sensors as the Cross-Sensor Cross-Modality (CSCM) data fusion algorithm:

CSCM Algorithm

1. Initialize x^i, i=1,2,…,L_T where x^i is the target state of the ith particle, and L_T is the total number of particles.

2. Prediction

FOR each particle i = 1: L_T,

Draw $x_j^i = x_{j-1}^i + disp_{j-1}$, where j is a discrete time index and $disp_{j-1}$ is the displacement of the target at time j-1

3. Determine the active neighborhood by the threshold of received sensor signals to generate vector z^{aco} and z^{sei}, where z^{aco} is the measured acoustic signal strength and z^{sei} is the measured seismic signal strength.

4. Each particle is associated with a particle weight to be computed by the weight update equation:

$w_j^i = p(z^{aco} \mid x_j^i)p(z^{sei} \mid x_j^i)$, where $p(\bullet|\bullet)$ is the likelihood function.

5. Normalization: weights w_j^i need to be normalized over all i so that the sum of all weights adds up to unity.

6. Compute expectation of the target state using:

$$\overline{E(x_j)} = \frac{1}{L_T}\sum_{i=1}^{L_T} w_j^i x_j^i$$

7. Resample – drop those samples with too small weights, and split those samples with largest weights so that the number of samples remain to be L_T.

The CSCM algorithm is therefore a series of prediction-measurement steps at each discrete time j. The time interval between successive time steps can be determined by the user and can therefore be used to set T_{res} in (6). The elegance of such a data fusion approach requires only predicted information from the previous time step and measurements from the current time step. Moreover, data from different sensors and across different modalities can be fused using the same consistent mathematical tool –

particle filtering. The choice of the likelihood functions in step 4 can be obtained from [43].

3.3. Overcoming the Prime Power Constraint

In section 2.1, we stated that the existence of (k^2+k+1,k+1,1)-designs are only guaranteed when k is an integer power of a prime number. However, because node schedules are not required to be unique, this constraint can be removed easily. One approach is as follows: Let $\aleph$ be the set of all natural numbers. For any general value of $M = L \in \aleph$, $L > 3$, we start off with a design with $L^- = (k')^2 + k' + 1$ number of schedules where k' is the largest integer power of a prime such that $L^- \leq M$. These schedules can first be assigned to the L^- sensing nodes and the remaining (L–L^-) sensing nodes can be randomly assigned repeated schedules found in the previous L^- set of sensing nodes.

Therefore, supposing there are 100 sensor nodes, one could choose k = 8 so that k^2+k+1=73 of these sensor nodes have unique schedules while the remaining 27 can randomly be assigned repeated schedules from the same (73,9,1) CSBD design. This ensures a good degree of schedule diversity (section 3.1.3) in the nodes.

Since analysis in our previous sections does not require schedule assignments to be unique, their results continue to apply.

3.4. Applying CSBD for Target Tracking

In this section, we use the results from sections 2.4, 3.1.1, 3.1.2 and 3.1.3 to solve for the design parameters k and T_{slot}, and the required deployment density ρ_{deploy} for a target tracking application. We consider three different requirements of a tracking system.

- Requirement A: Coverage/Connectivity Preservation
- Requirement B: Collision Control
- Requirement C: Information/Agent Travel Speed

Requirement A is satisfied by considering the more stringent of the two requirements between coverage and connectivity within some bounded time, T_{res}. This has been discussed at length in Sections 2.4 and 3.1.1. Suppose the coverage requirement is more stringent, we assume $N_{hop} = 1$ in (6):

$$2T_{res}/3T_{slot} = (k^2 + k + 1) \tag{18}$$

If the connectivity requirement is a more stringent constraint, an equivalent expression for connectivity that is similar to (6) can be written. (18) can then be rewritten in terms of connectivity.

Requirement B has been discussed in section 3.1.3 and the relevant equation to consider is (15).

Requirement C has been discussed in Sections 3.1.2. Using *Theorem 3.1.2.1* and (10), and assuming H =1, we obtain:

$$\frac{(\alpha - 2)R_S}{v_{\max} T_{slot}} \geq k(k+1) \tag{19}$$

With equations (18), (19) and (15), the variables k, T_{slot} and ρ_{deploy} can then be solved.

Suppose we consider further that the deployment density cannot be controlled and ρ_{deploy} is therefore not a variable, but a constant. The value of k can then be determined immediately by using (15). Given k, T_{slot} can be computed using (18), and (19) then becomes a condition to check that information propagation in the network is indeed faster than the maximum achievable speed of the target v_{max}. We shall adopt this latter approach in our simulations studies later.

With the computed T_{slot} and k values, the lower bound for node lifetime can be computed using (16) or (17). In some practical cases, the condition that $R_C > 2R_S$ (which forms the foundation of our derivations in section 3.1.2) may not be achievable. This problem can be solved by increasing the node density (assuming that this is possible) by first assuming that the sensing radius of nodes is smaller than its actual achievable radius and deploying more sensors to fill the “sensing gaps”. The derived results will therefore still apply. We provide our simulation results related to CSBD-based target tracking in the next section.

3.4.1 Simulation Results

While analytical reasoning and derivations have been provided in previous sections, this section attempts to compare our proposed CSBD with other wakeup schemes in terms of several performance metrics. We choose an agent-based target tracking application as the scenario for our simulations.

It is not a straightforward task to find the ideal wakeup schemes to compare with CSBD, for the reason that no other scheme to the best of our knowledge, simultaneously considers coverage/connectivity in bounded time, information travel time and collision control as target tracking requirements in the network, in an energy-constraint sensor network. We have nonetheless, selected RAW [12] as a representative algorithm for random wakeup schemes and PECAS [21] as a representative algorithm for on-demand wakeup schemes to compare with CSBD. Similar to CSBD, both RAW and PECAS are time-asynchronous schemes.

Simulation Setup

Simulations are performed using Network Simulator 2 with a random (uniformly distributed) deployment of 1,000 nodes with R_S, = 50m and α = 3 in a geographical square area of 1km by 1km with each node arbitrarily using one of the schedules from an original (k^2+k+1,k+1,1) design. We assume that a tracking application requires a collision probability of less than 10% and p_{tx} = 0.2. Then, using (15), k = 5. Assuming that the application requirement is such that T_{res} = 1s, we set T_{slot} = 21.5ms (unless otherwise stated in some simulations) and assume that E_{node} = 16.56 kJoules and E_{awake} = 0.17625 Joules. In our simulation, we assume the average speed of the target to be 15m/s (about 54km/h) with v_{max} assumed to be 30m/s (about 108km/h), unless otherwise stated. These simulation parameters are summarized in Table 1.

Table 1: Simulation parameters for target tracking

Total Nodes	1,000	Deployment Field	1km x 1km
R_S	50m	T_{res}	1s
α	3	T_{slot}	21.5ms
k	5	E_{node}	16.56 kJoules
v_{max}	30m/s	E_{awake}	0.17625 Joules

With these parameters, we test that the condition in (19) is satisfied, implying that agent's speed is faster than the maximum speed of the target. The scenario is repeated for 100 times, each time with a different random sensor deployment in the 1km by 1km sensor field and with different targets (unless otherwise stated, the number of targets in the sensor field is five, with each target randomly generated at any location inside the sensor field) moving in the field with Gauss-Markov mobility [78] for a total simulation time of $10\varsigma_{life}$, where we compute $\varsigma_{life,LB} \approx 3$ hrs. Note that this is just a lower bound assuming full duration transmissions in all wakeup slots in all nodes. For information routing, Geographic Routing (GR) [71] is used. Agents (some software code implementation) propagate from one node to another following the target for the purpose of feature extraction, target classification, target association, intrusion monitoring or other purposes, such that $N_{hop} = H = 1$. We assume that target velocity prediction has zero errors and target association with measured signals is 100% accurate (so as to single out certain tracking errors to be described later). We assume a mobile command center where this information fusion takes place and for simplicity, this is assumed to be where the agent is.

Delay and Delay Variances

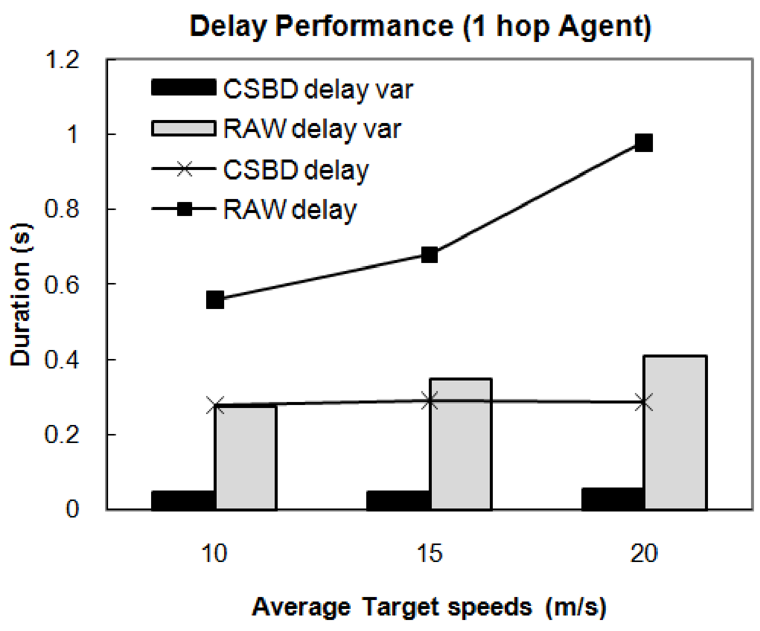


Figure 11: Delay Performance

We investigate the delay and delay variance performance of our CSBD design, and compare with a random wakeup scheme RAW with a similar duty cycle. Figure 11 shows that the CSBD design outperforms RAW by a good margin in terms of delay and delay variance. For instance at the average target speed of 15m/s, CSBD more than halves the average delay compared to RAW and operates with very low delay variance not exceeding 0.05s. Delay variance of CSBD is low because of the deterministic nature of its wakeup schedule. We have not shown the delay performance of PECAS in the graph because the scheme essentially employs a different technique using a dual-radio architecture to track targets. Both delay and delay variance for PECAS are approximately zero for an agent that is one-hop away from the target (ignoring propagation and processing delays). Although this appears to have significantly outperformed CSBD, it is important to note that the tradeoff for PECAS will then be in terms of a more costly implementation and a shorter node lifetime for multiple targets, which we shall show later. The simulations also show that the delay performance of RAW is poorer than CSBD at higher speeds because the average number of hops packets have to traverse to reach the mobile agent is more than RAW than it is for CSBD. In fact, the delay and delay variance performances for CSBD hardly changes because the number of hops packets have to traverse is almost certainly just one hop.

Energy Consumption and Network Lifetime

We assume that the network lifespan is the time duration until any sensor node in the network is completely depleted of energy [73, 74]. For network lifetime comparison with PECAS, we assume that the idle mode power parameter for PECAS is 35mW and simulation results in Figure 12 indicate that CSBD outlasts PECAS by about 1.67 times in terms of average network lifespan for five targets. The main reason is that PECAS puts nodes to the idle mode when not involved in tracking targets so that the channel can still be monitored for possible packets. This is known to conserve very little energy compared to CSBD nodes being set to sleep mode when not in operation.

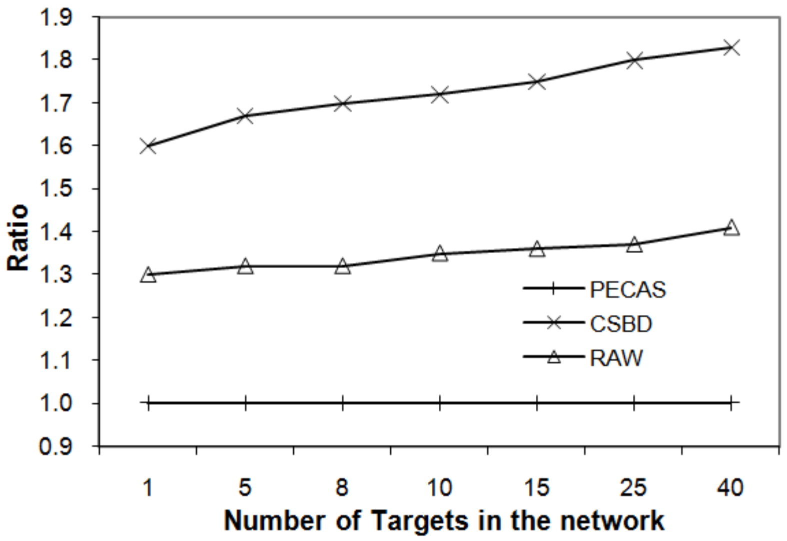


Figure 12: Network Lifetime Performance for R_C = 150m.

Furthermore, energy consumption in PECAS nodes becomes approximately 25% higher than CSBD nodes when more targets are in the network. This is because in PECAS, more nodes are required to be woken up to track more targets. Our solution, CSBD, however, is rather independent (ignoring transmission energies which must be consumed for both PECAS and CSBD) of the number of targets in the network because of its fixed schedules. For the case of RAW, it achieves a poorer lifetime performance compared to CSBD because of the more transmissions required to forward data to the agent and wasted energy in transmissions due to more collisions.

In section 3.1.4, we note that the CSBD lifetime in (17) is a lower bound and a function of the application time resolution requirement T_{res}. Figure 13 illustrates the CSBD lifetime with different values of T_{res} for 5 and 25 targets. All these values of T_{res} can be tested to satisfy the condition in (19). As the number of targets increase, the simulated lifetime approaches the approximate theoretical bound.

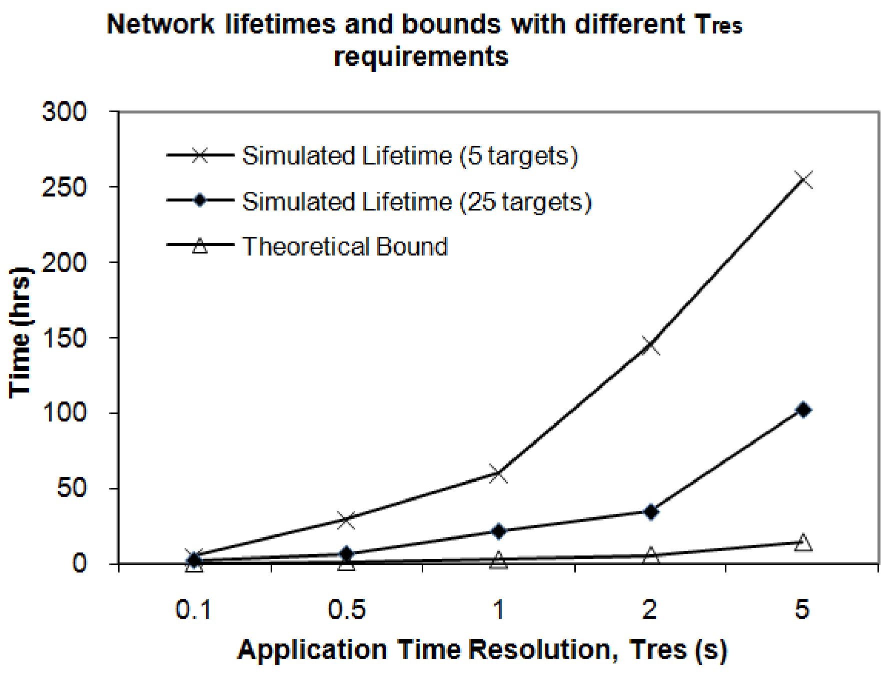


Figure 13: Network Lifetimes and approximate lower bound for different T_{res} values.

In a practical deployment scenario for target tracking with T_{res} = 5s, we can expect to achieve of an average network lifetime of more than 10 days. Again, this is based on the assumption that two standard AA alkaline batteries are used for each node. If lithium-based AA batteries are used, the lifetime performance can extend approximately by another seven times. In some sensor network deployments, it is also possible that extra battery packs are connected to each node so that every node can connect up to six AAsized batteries. The average node lifetime can therefore potentially be extended to more than 210 days.

Tracking Continuity and Target Speeds

We define the Loss of Continuity in Tracking (LCT) as the loss of the ability to perform an association between the target and the agent that was first created to track that target. When this happens, nodes are unable to distinguish between the detection of a new target and the continual tracking of an old target. In our simulations, we assume that LCT is equivalent to the event that the target is found to be out of the sensing coverage radius of the node that the agent resides.

For LCT performance comparison across schemes with different average target speeds, we discover that the agent in RAW has much higher LCT compared to CSBD (as promised by the algorithm because information travels faster than the target). In our simulation, PECAS also has low LCT because of its double-radio architecture, assuming no velocity prediction and data association errors in the application layer. As the average speeds of targets are increased, it becomes progressively easier for RAW to “lose” targets and LCT increases. Figure 14 investigates this trend in LCT for different average target speeds with the maximum speeds v_{max} set to be always 30m/s. In one simulation, we assume the communication channel to be lossless and no packet collisions could occur to first identify the underlying trend in LCT. RAW LCT increases at a faster-than-linear rate to more than 90% at an average speed of 25m/s. CSBD and PECAS LCT remains at 0% for all target speeds.

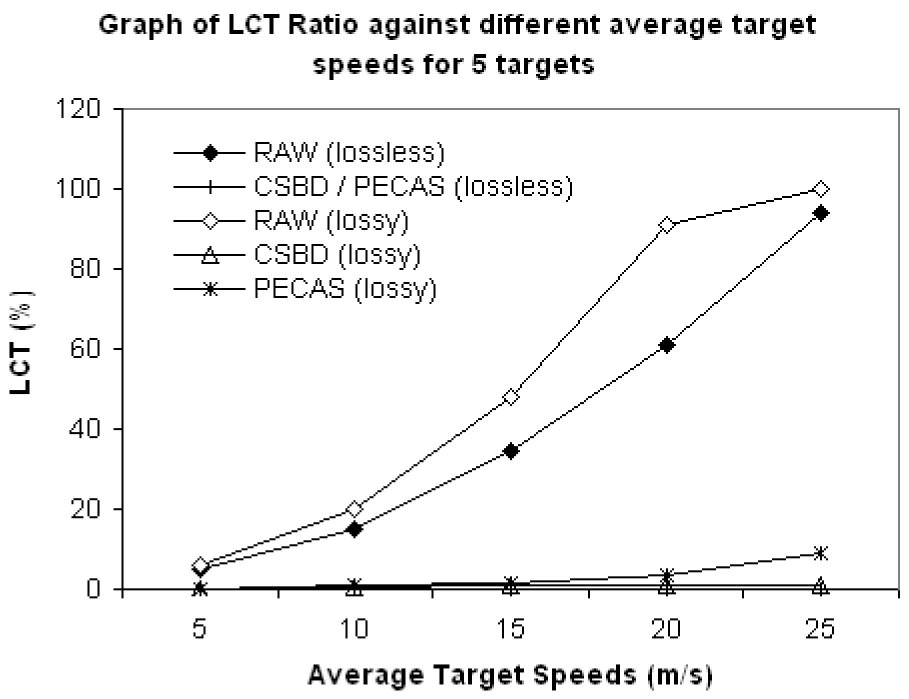


Figure 14: Loss of Continuity in Tracking (LCT) for different average target speeds

When the channel is lossy, packets are lost even within a single hop and both CSBD and PECAS would have encountered packet losses. These packet losses are due to channel impairments, represented by log-normal shadowing with a variance of 10dBm, and packet collisions when more than one node transmit packets at the same time. Collectively, packet loss rate is about 5%. It is thus possible for targets to be “lost” when they move fast enough and packet retransmissions do not happen quickly enough. In this set of simulations, packets are retransmitted after some random backoff time if no acknowledgement packet is received from the agent (located at some node one-hop away from the target) within twice the propagation delay.

Our simulations in Figure 14 show that the LCT of CSBD is no more than 1%, even at high average speeds of 25m/s. One reason is that the agent is always within one hop of the target. The LCT of PECAS is slightly higher primarily because of more collision losses. CSBD handles losses due to collision better because of the arrangement of active slots in its schedules. On the other hand, a lossy channel affects RAW drastically with its LCT reaching 91% and 100% for average target speeds of 20m/s and 25m/s respectively.

In these simulations, the node density is fixed such that (15) is satisifed. If, however, the deployment density is increased, the probability of packet collisions would be higher resulting in lower throughput and more packet losses. More energy is also expended if packet retransmissions are required. If the deployment density is lower, as long as k and T_{slot} remains the same and the network remains connected, performance in terms of network lifetime and LCT should remain the same, if not better. Of course, if the deployment density is so sparse such that the network is eventually disconnected, packet losses and LCT will also increase.

3.5. Tracking Wakeup Schedule Function (TWSF)

The results in sections 3.4 are sufficient to track any target of a known maximum speed v_{max} by solving simultaneous equations for the values of k, T_{slot} and ρ_{deploy}. However, these values must be computed before deployment. Suppose v_{max} is not known before deployment and the network is to be configured later to track a mobile target of any maximum speed that is to be specified after deployment. This section addresses this problem and introduces the Tracking Wakeup Schedule Function (TWSF) for the purpose. TWSF is also based on an initial CSBD design, but is coupled with an adaptation technique.

Suppose we start off with an arbitrary CSBD from which schedules are selected for sensor nodes with a certain k value. Assume that we choose a reasonably large value for k for the moment with the motivation of having a low duty cycle. We would then attempt to add in a sufficient number of new "Awake" slots into the CSBD so that a target of an arbitrary maximum speed v_{max} can be tracked. The challenge now is to determine the slot times at which New "Awake" Slots (NAS) should be added so that ideally, the least amount of energy is expended (because every new "Awake" slot added increases energy consumption for the nodes). Our strategy is to add sufficient NAS between any two schedules at regular intervals so that the *LNWT* (see *definition 3.1.2.1*) between them is eventually small enough for the tracking to be possible. This strategy is repeated for all schedule pairs that are neighbours of each other in the deployment.

Consider (11) and *Theorem 3.1.2.1* for the H=1 hop case, and the original CSBD schedules of any two arbitrary sensor nodes. For a target to remain “trackable”, the maximum number of NAS required between these two nodes, $S_{new,max}$, is the maximum number of adjacent sleep slots in any one of the schedule, $T_{Cyclic,sleep}$, divided by the longest allowable duration that both schedules are not awake at the same time is given as:

$$S_{new,\max} = \left\lfloor \frac{T_{Cyclic,sleep}}{LNWS_{track,\max} + 1} \right\rfloor = \left\lfloor \frac{k(k+1)}{\left\lfloor \frac{(\alpha-2)R_S}{v_{\max}T_{slot}} \right\rfloor + 1} \right\rfloor \tag{20}$$

The $\lfloor x \rfloor$ operation and the unity term in the denominator is the result of integer division. For analysis purpose, we may remove the rounding down operation and the unity term in the denominator by an inequality:

$$S_{new,\max} \le \frac{k(k+1)v_{\max}T_{slot}}{(\alpha-2)R_S} \text{ slots} \tag{21}$$

This is the maximum number of additional slots required for CSBD between any two nodes to track a target with maximum speed $v_{\max}$ within its sensing field such that H=1. Suppose that each node has a maximum of $N_{\deg,\max}$ neighbours, the maximum number of new slots required for any node in the network (taking into account of all its neighbours) is simply bounded by:

$$\Psi_{new,\max} = N_{\deg,\max} \left\lfloor \frac{k(k+1)v_{\max}T_{slot}}{(\alpha-2)R_S} \right\rfloor \tag{22}$$

For the extension of results to a H=2 hop agent case, it can similarly be shown that:

$$S_{new,\max} \le \frac{2k(k+1)v_{\max}T_{slot}}{(\alpha-2)R_S} \text{ slots} \tag{23}$$

which is always larger than (21).

Based on the above derivations, it is then possible to present an algorithm that adapts CSBD to a tracking-enabled design that tracks targets that is predicted to move at some maximum speed $v_{\max}$ after deployment. This can be achieved by strategically adding additional NAS based on the result in (11).

Assuming H=1, we present our 1-hop Agent TWSF algorithm below for any arbitrary node i:

1-HOP AGENT TWSF ALGORITHM PSEUDO CODE FOR NODE *I*

```
Let $LNWS_{track,\max} = floor\left[\frac{(\alpha-2)R_s}{v_{\max}T_{slot}}\right]$
trackingEnabled = False
while trackingEnabled = False
    maxDeg = True
    for $j \in J$ do
        if $N_{\deg}(i) < N_{\deg}(j)$ then maxDeg = False
    end for
    resultID = False
    if  maxDeg = True
        counter = 0  // initialize to zero
        For timeslot = 1 to $(k^2+k+1)$,
            If i and j are not both awake,
               counter = counter + 1
               If counter > $LNWS_{\text{track,max}}$ then
               //Add a new active slot to both schedules
                   call addNewSlot()
                   counter = 0
               End If
            Else
                counter = 0
            End If
        End For
        // sends ID and new schedule to 1-hop neighbour nodes
        call notifyOneHop( )
        trackingEnabled = True
    else if maxDeg = False
        while resultID = False
            resultID = listenNotify( )
            // waits for the return of ID (& new schedule)
            // of the tracking-enabled neighbour
        end while
    end if
end while
```

Figure 15: Distributed 1-hop Agent TWSF Algorithm

where

- I is the set of all nodes in the network,
- J is the set of all 1-hop neighbour nodes of a reference node $i \in I$ that has not yet adapted for target tracking (i.e. not tracking-enabled yet),
- $LNWS$ is the longest time duration (in slots) between common wakeup times between any node i and any of its one-hop neighbour node $j \in J$ in the original CSBD,
- T_{slot} is the duration of one slot time, and

$N_{\mathrm{Deg}}(X)$ refers to the degree of a node X.

Each node first computes $LNWS_{track,\max}$ for a given $v_{\max}$, known after deployment. The *floor* operation rounds down the expression to an integer. At first, all nodes are not tracking enabled (*trackingEnabled* = False). If the node is not one with the highest degree amongst its neighbours, it listens for a message through the function *listenNotify*() from other neighbours until it is the largest degree neighbour that is not tracking-enabled (i.e. *trackingEnabled* = False). The function *listenNotify*() returns a True value to the variable *resultID* when an ID message together with the new schedule from a neighbouring node has been received. The node can then proceed to retest the condition if it is the highest degree node amongst the rest of the other neighbours (those that are not target-tracking enabled yet). If this is not true, the loop continues. Otherwise, the node can now perform a search through every time slot of every of its neighbour nodes, and when necessary, assigns a new slot common to itself and the respective neighbour node using the *addNewSlot*() function. The *addNewSlot*() function is only necessarily called when both a neighbour node and itself are not awake at the same time for more than $LNWS_{track,\max}$ slots. After all slot assignments, the algorithm sends a message through the function call *notifyOneHop*() to all its one hop neighbours that it has completed its slot assignments and subsequently terminates the algorithm by setting *trackingEnabled* = True.

Let C be the number of actual NAS assigned between two neighbour nodes. Now, every sensor node will have at most ($CN_{\deg,\max}+k+1$) "Awake" slots after the adaptation. Assuming that each awake time slot consumes E_{awake} units of energy (for simplicity, this also includes transmission energies), the maximum energy consumed per T_{cycle} per node is $(CN_{\deg,\max}+k+1)E_{awake}$. Let the total energy in each node be denoted by E_{node}, then the network lifetime, ς_{life}, of each node can be given by:

$$\varsigma_{life} \approx \frac{\left(k^2+k+1\right)T_{slot}E_{node}}{\left(CN_{\deg,\max}+k+1\right)E_{awake}} \geq \frac{\left(k^2+k+1\right)T_{slot}E_{node}}{\left(S_{new,\max}N_{\deg,\max}+k+1\right)E_{awake}} \tag{24}$$

Note that the bound is a worst-case minimum bound and not the typical average node lifetime. If the network lifetime is defined as the time duration until the first network node is completely depleted of energy, then the network lifetime is also exactly bounded by the same expression.

The presented TWSF algorithm therefore guarantees the per-hop delay bound derived in (11), total new slots bounds derived in (21) and (22), and node lifetime bound derived in (24). Having formulated these worst-case performance quality-of-service (QoS) guarantees, we provide simulation results to investigate the typical average performance of the TWSF algorithm.

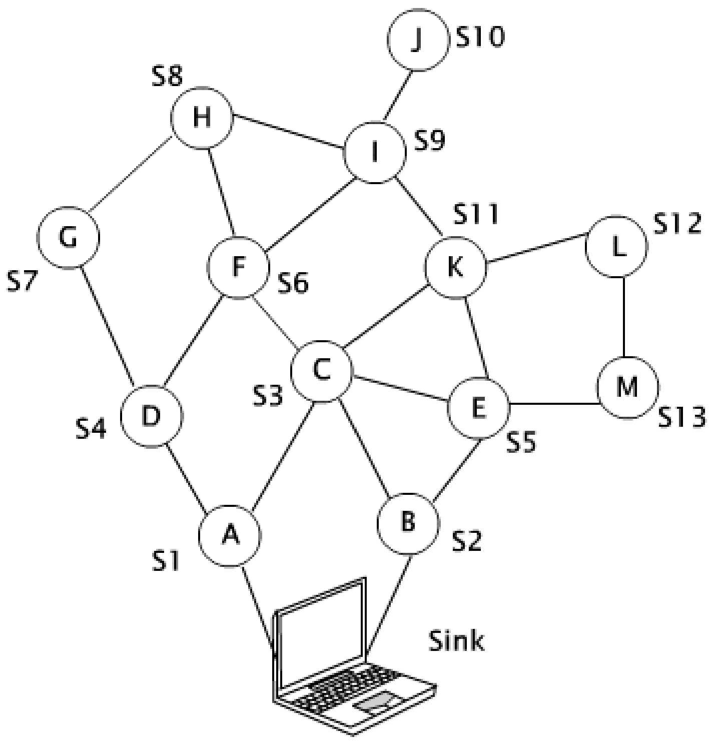


Figure 16: An example (13,4,1) network.

As an example, we consider the network of nodes in Figure 16 employing the (13,4,1)-design as shown in Figure 1. Suppose we have R_S = 50m, v_{max} = 10m/s, α = 3 and T_{slot} = 1s. The TWSF algorithm will therefore ensure that LNWTs between any two nodes in the deployment is no more than 50/(10 × 1) = 5 time slots. In any locality (nodes within one hop), only the highest degree neighbour will initiate the algorithm and there will be no contention in assigning NAS. In cases where there are two or more nodes with the highest degree, the node with the largest schedule number (or serial number) will initiate assignment of slots. In our example, both nodes C and I will initiate the slot assignment with all their neighbours. We note that node F is a neighbour node to both C and I. In this case, node F answers the slot assignment request from the node with the highest neighbour count, that is node C, before it attends to node I. Again, if both nodes C and F have the same neighbour counts, node F answers to the node with the larger schedule number (it would have been node I in this case).

Subsequently, after slot assignments have been completed for both nodes C and I, both nodes will set the condition *trackingEnabled* = True. Note that by now, the following slot assignments between the following node pairs have been completed:

- C and A, C and B, C and E, C and F, C and K
- I and F, I and H, I and J, I and K

The next generation of nodes with the next highest degrees in their neighbourhoods includes nodes J, F and K. By the same algorithm, the following slot assignments between the following node pairs will be performed:

- F and D, F and H
- K and E, K and L

Note that there are no more slot assignments necessary for node J because it only has one neighbour (node I). Further, it is also worthy to note that although slot assignment is "considered" for the K-E node pair, no additional slots are added. This is because previous slot assignments between node pairs K and C, node pairs E and C, and node pairs K and I are already sufficient.

The third generation of nodes that will initiate slot assignments includes nodes D, E, H and M. Here, the following node pairs were considered:

- D and A, D and G
- E and B, E and M
- H and G
- M and L

Similarly, not all node pairs considered will have new slot assignments. Node pairs E and B, and node pairs H and G do not require new "Awake" slots.

Finally, all neighbour nodes of the remaining nodes A, B and G will then have *trackingEnabled* set to *True*. The resulting wakeup schedule after TWSF is shown in Figure 17.

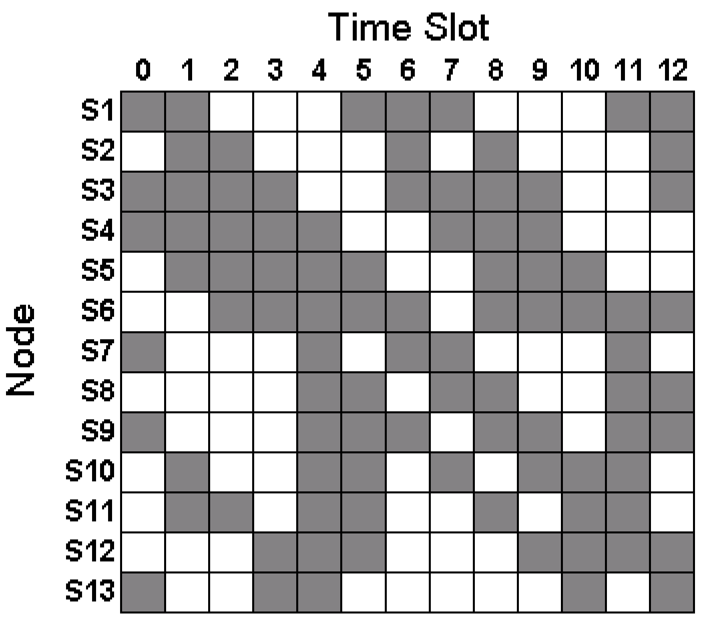


Figure 17: TWSF-Enabled Schedules using example in Figure 16.

Note that the properties discussed in sections 3.1.1 and 2.4 remains valid for TWSF-enabled schedules because only new slots are added with none shuffled and none removed. In the final design of Figure 17, a total of 40 NAS were added, representing about 77% more "Awake" slots needed to track the target. Still, the total duty cycle of all nodes considered is about 54% with T_{slot} = 1s. From a trace of the number of NAS assigned between any two neighbour nodes in the deployment, no more than two NAS per node-pair are required. This certainly verifies our bound in (21) where we have $S_{new,\max} \leq \frac{3(4)10(1)}{(3-2)50} = 2.4$ slots.

Another useful attribute of TWSF is its flexibility to revert back to its original CSBD design by removing all previously assigned NAS when tracking of such a target is no longer required. This saves energy and makes TWSF re-configurable to other user requests of a different v_{max}. Yet, its underlying CSBD design continues to guarantee minimum delay bounds, sensing coverage time bounds, connectivity time bounds and other features.

As new slots are added, the probability of collision will also increase, especially if the number of targets to be tracked increases. However, it is important to note that TWSF is a wakeup scheme, and it does not prevent the implementation of a concurrent MAC layer scheme to improve throughput. Further, we can also formulate an expression similar to (14) to solve for an approximate ρ_{deploy} value if there is a certain degree of control over the density of deployment.

3.5.1 Simulation Results

TWSF simulations are performed using Network Simulator 2 with a random (uniformly distributed) deployment of 100 nodes in a geographical square area of 1km by 1km with each node arbitrarily using one of the schedules from the original (13,4,1) CSBD design (before TWSF is implemented). We assume that the sensing coverage radius of sensors is $R_S = 50$m with $\alpha = 3$, and $T_{slot} = 1$s. Assume further that a single target traverse into the sensor field using the Gauss-Markov Mobility Model [78]. In our simulation, we investigated the average speeds of the target to be either 10m/s (about 36km/h) or 15m/s (about 54km/h), with v_{max} assumed to be 20m/s (about 72km/h). The scenario is repeated for 100 times, each time with a different random sensor deployment in the 1km by 1km sensor field and with a different target moving into the field with Gauss-Markov mobility for a total simulation time of 80,000 simulation seconds. We further use E_{node} = 16.56 kJoules and E_{awake} = 0.17625 Joules based on two AA batteries per node and 100% transmission time during all active slots with 100mW transmission power, 45mW active mode power, 90mW sensing power and negligible sleep mode power. These simulation parameters are summarized in Table 2.

Table 2: Simulation parameters for TWSF Simulations

Total Nodes	100	Deployment Field	1km x 1km
R_S	50m	T_{slot}	1s
α	3	E_{node}	16.56 kJoules
v_{max}	20m/s	E_{awake}	0.17625 Joules
Transmission Power	100mW	Active Mode Power	45mW
Sensing Power	90mW	Sleep Mode Power	0mW

Geographic Routing (GR) [71] is used. We simulate agents (some software code implementation, section 3.2) that propagate from one node to another following the target for the intended purpose of feature extraction, target classification, target association, target tracking, intrusion monitoring or other purposes. However, no actual data fusion algorithms are implemented in this part of the simulations studies. We assume that target velocity prediction has zero errors, so as to single out the Loss of Continuity in Tracking (LCT) errors described later. We assume a mobile command center where this information fusion takes place (for simplicity, assumed to be where the agent is). We investigate the delay, delay variance and LCT performance of our TWSF algorithm and compare with regular (13,4,1)-CSBD and RAW [12].

The LCT is defined as the loss of the ability to perform an association between the target and the agent that was first created to track that target. When this happens, nodes are unable to distinguish between the detection of a new target and the continuation of the tracking of an old target.

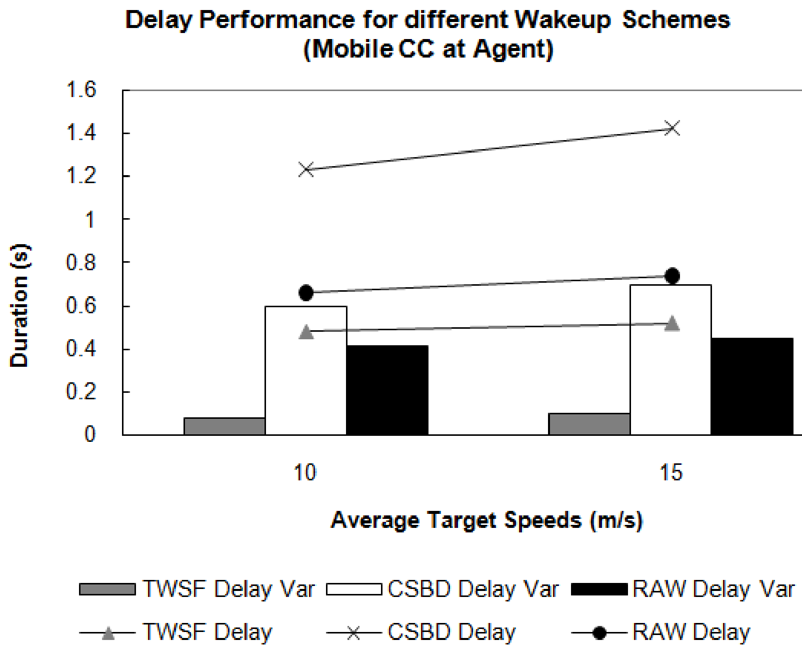


Figure 18: Delay Performance Comparison

Figure 18 shows that TWSF outperforms regular (13,4,1)-CSBD by a good margin in terms of delay and delay variances for both average target speeds. This is primarily because of the larger duty cycle of TWSF after new active slot assignments. We have also performed other simulations comparing between TWSF and CSBD, and discovered that a doubling of duty cycle reduces the average delay by about 2.8 times. In RAW, the duty cycle is made equivalent to that of TWSF and the performance is also not as good. The energy consumption of TWSF is the same as RAW and about twice that of CSBD. For LCT performance comparison across schemes, we discover interestingly that the agent in CSBD has an LCT of about 40%, while TWSF experiences no such problem with an LCT of 0% (as promised by the algorithm). The reason why CSBD LCT is high is because LNWT in CSBD are not sufficiently small enough to track targets of the maximum speed v_{max} = 20m/s. On the contrary, TWSF has been adapted just for this purpose and LCT is therefore 0%.

3.6. Summary

We proposed CSBD-based wakeup for agent-based sensor networks and analyzed sensing coverage, delay, schedule diversity and node lifetime aspects of the problem. We further introduced a typical agent-based data fusion algorithm using the particle filters approach (section 3.2), applied to target tracking. Before we apply CSBD for target tracking, we discussed how the prime power constraint in CSBD might be overcome. While the CSBD-based target tracking relies on prior knowledge of the maximum target speed, we introduced another algorithm TWSF that do not require this prior knowledge for deployment. TWSF is also based on CSBD and inherently possesses the desirable properties of a CSBD because no “Awake” slots are shuffled or removed; only added. We have provided simulation results for both CSBD and TWSF tracking.

For CSBD-based tracking, results indicate its superiority in terms of delay, delay variance and tracking continuity of the target (LCT). Even in the presence of a lossy communication channel (with packet collisions), LCT of CSBD is low at high average speeds. In terms of energy consumption, CSBD also outlast PECAS for multiple targets in the network.

TWSF results reveal that when a CSBD is used without a proper choice of the parameter *k*, both delays and LCT can be very high. This happens when the maximum target speed v_{max} is not known before deployment and an insufficiently large duty cycle results in the poor performance. After adaptation with the TWSF algorithm, delays can be dramatically reduced and the continuity of tracking can be satisfied.

Chapter 4: Query-Based Sensor Networks

While agent-based systems are widely used for collecting real-time data from sensors as they are generated, query-based systems serve as a cache and storage solution for users to query information about the network or environment data of the past. These two systems can also be complimentary to each other, and can be implemented together in a single application. This chapter discusses in detail, the practical issues of implementing a Query-based sensor network based on CSBD.

Section 4.1 is concerned with a type of delay called the Query Waiting Delay that is only specific to query-based database sensor networks operating some underlying wakeup schedule. Results in section 4.1 are established on three major constraints. Section 4.2 attempts to relax each of these constraints. In section 4.3, we encourage the use of a query-based sensor system to compliment that of an agent-based one. We further show in section 4.4 that such a system can work well for static nodes, as well as for sensor nodes that have limited mobility for the purpose of energy balancing in the network. In section 4.5, we provide simulation and implementation results of our solution. We summarise this chapter in section 4.6.

4.1. Key Design Considerations

4.1.1 Query Waiting Delays

The popularity of many query-based sensor networks may be attributed to its diverse applicability to solve numerous problems in the real-world environment. These query-based solutions are made possible by advancements in query in-network processing and data aggregation technologies [52, 53, 54, 55, 56] that have been developed over the years for large-scale distributed sensor systems that are most often limited in terms of resources and energy. Query-based sensor networks are not only conceptualized in theory, but with real world implementations [1, 57, 58], thereby opening up the possibilities of a wide variety of sensor monitoring, detection and tracking applications.

The COUGAR project at Cornell University [57] is one of the first attempts to view a sensor network as a distributed database. The main idea is to use declarative queries to abstract the user from the physical details of query processing (such as node selection for data collection). A Query Optimizer, located on the sensor gateway, is responsible for generating a query plan that specifies both the data flow and exact in-network computation plans for the incoming query before sending it to all relevant nodes.

The TinyDB [58] implementation effort of the Acquisitional Query Processing technology [52] focuses on opportunities that arise in sensor networks when specifications of time and sampling constraints are added in data streams produced by sensors in queries. Time constraints determine how long the query should run and sampling constraints determine how often sensors should collect data. The system therefore hides from the user how actual queries are processed and how results are returned, allowing the user to focus on analyzing the query results.

Data Aggregation techniques in sensor networks also offer opportunities to conserve communication energy by summarizing measured data. In [59], the authors construct an analytical model that describes energy, timeliness of data and the degree of data aggregation performed. The control of lossy aggregation is by means of a feedback loop that responds to the amount of data generated against system capacity. Such aggregation service in sensors can, in practice, be provided by Tiny AGgregation (TAG) [60] designed for TinyOS [61] by the use of aggregate functions that may include COUNT, MIN, MAX, SUM and AVERAGE operations, classified according to state requirements.

It may not be immediately obvious that symmetry in the cyclic symmetric $(k^2+k+1,k+1,1)$ design has yet, another benefit, other than ensuring bounded-time sensing coverage and connectivity in sensor networks. In this section, unless otherwise stated, we refer to the term “coverage” or “coverage requirement” as just *m*-coverage. While results in Section 3.1.1 apply to applications that require sensing coverage within known bounded time, this section assumes the requirement of network sensing at all times while analyzing the data querying aspect of the network.

These two results (in both the previous and current sections) may then be applied either differently in separate applications, or as a two-tier solution for a single application, which we shall illustrate further.

With sensors deployed in a sensor field, individual sensors record environmental or sensed data in a distributed way. A distributed database of information is therefore naturally constructed. Any user can then query this sensor network for information. However, since sensors are put on some sleep-wake schedule, the required information may not always be available for querying. We term this waiting time due to the temporary unavailability of sensors (in “Sleep” mode) where the query response, whether wholly or in part, is to be generated as the Query Waiting Delay (QWD). In this section, our interest in delay is only primarily restricted to QWD. Processing and propagation delays are ignored because they are hardware and medium dependent, but wakeup schedule independent. Routing delays are affected by processing delays, propagation delays and the underlying wakeup schedules of the nodes. We provide simulation results of routing delays in a later section while our analysis in this section remains focused on QWD.

Sensors are often deployed in excess due to their small form-factor and relatively cheap cost, more than one sensor often measure the same event at the same time. We focus on one such region where all sensors in this region are in sufficient proximity to measure the same event. Let the set of all sensors in this region of sufficient proximity be the set U_S. This, in fact, is the m-coverage requirement for some m. For the moment, we further impose the following assumptions:

- The 1-Hop User Constraint: The user (or application/agent) is 1-hop away from all sensors in the set U_S.
- The Unique Schedule Constraint: All sensors in the set U_S have unique wakeup

schedules selected from a $(k^2+k+1,k+1,1)$ cyclic symmetric design.

- The Design Space Spanning Constraint: The set of all sensors in U_S has schedules that span the $(k^2+k+1,k+1,1)$ cyclic symmetric design. This means we assume that there are exactly k^2+k+1 sensors and each using a unique schedule from the design set.

We shall relax all of these assumptions in a later discussion. As applications may have different requirements, the concept of different classes of query users arises. From the last constraint, it is clear that the region under consideration is always $m = (k+1)$-covered (by *Symmetry 1*). This leads to a relatively dense network of nodes but this ensures at least m sensors measures the same event at any one time and we later show that QWD can be minimized for a class of query users known as coarse data (CD) users.

Definition 4.1.1. Coarse Data (CD) users require coarse information about the environment of a region of interest and are satisfied with any one of the many possible responses for every non-overlapping time interval T_I that spans some desired time duration T_D.

Definition 4.1.2. All Data (AD) users require detailed information about the environment of a region of interest and can only be satisfied with all possible responses for every non-overlapping time interval T_I that spans some desired time duration T_D.

For instance, in a temperature acquiring application, users that are interested in the temperature of a particular region within some time duration may be classified as CD users. Temperature information usually varies slowly across both space and time and it is not necessary to acquire all the sensor reports of temperature in that region. One response (assuming accurate enough) from one of the many sensors in the region may be sufficient. On the other hand, users that are interested in tracking a target of interest in a region may be classified as AD users. In order to evaluate the exact target coordinate through triangulation (using ultrasonic sensors as an example), positioning data from at least three sensors are required; and with more sensor readings available, the variance of the tracking result can be minimized. Other applications may also fall somewhere in between the two classes of users. This is true because even for the temperature-monitoring example, more than one sensor reading is usually required for error reduction; and for the target-tracking application, we may not require the retrieval of all the sensor readings to calculate the "center of mass" of the target location due to energy concerns. In many cases, we wish to have some kind of tradeoff between accuracy and energy. Nevertheless, we have defined these two classes of users in sensor networks to highlight the possible extremes and simplify our analysis later.

Definition 4.1.3. A query of length T_Q is defined to be the duration of a set of past measurements over which is of interest to the user.

Definition 4.1.3 may be interpreted by considering the following commonly used SQL query statements:

```
SELECT temp
FROM sensors as s
WHERE s.time <=TQ1 AND s.time > TQ2
```

Here, (TQ1 – TQ2) = T_Q. We pose the problem in the following manner:

Problem D1. For a cyclic (k^2+k+1,k+1,1) design, determine the delay bounds it takes to reply a query of any length T_Q in the set U_S.

Problem D2. For a cyclic (k^2+k+1,k+1,1) design, determine the energy bounds it takes to reply a query of length T_Q in the set U_S.

We further classify our solution according to the class of user that issues the query. *Theorems 4.1.1*, *4.1.2* and *4.1.3* apply to CD users while *Theorems 4.1.4* and *4.1.5* apply to AD users.

Theorem 4.1.1. A cyclic symmetric (k^2+k+1,k+1,1) design guarantees the existence of a *zero Query Waiting Time (QWD) for a CD user at any arbitrary time slot.*

Proof. Let N_R be the number of nodes in the set U_S. Let A_S be the set of all sensors (k+1 sensors in total) in the set U_S that are awake at any arbitrary time slot S. Since these sensors already have one overlapping active slot in slot S, there are zero remaining overlaps in the remaining (N_R –1) time slots between them. The intersection of all the remaining active time slots (other than the active time slot of S) in A_S is the null set. This is equivalent to arranging $(k+1)^2 - (k+1)$ remaining active slots into $N_R - 1$ remaining empty time slots with no two active slots occupying the same empty slot. In order for the condition to be satisfied and for all active slots from sensors in A_S to cover the entire duration of T_{cycle}, there must be exactly the same number of remaining empty slots as there are remaining active slots. Hence,

$$(k+1)^2 - (k+1) = N_R - 1 \tag{25}$$

Solving gives:

$$N_R = k^2 + k + 1. \tag{26}$$

Since all active slots in A_S that span T_{cycle} will also span all time durations because of the schedule repetition after every T_{cycle}, this is sufficient proof that as long as our design is of the $(k^2+k+1,k+1,1)$ design, it is possible to guarantee zero QWD for CD users. Since the proofs are obtained without imposing any restrictions at the time instant of querying, there will always exist a zero QWD response for CD users at any arbitrary time slot. ■

Theorem 4.1.2. For a cyclic symmetric $(k^2+k+1,k+1,1)$ design, an upper bound delay on a query of length $T_Q \le 2T_{slot}$ for a CD user is given by $\frac{1}{2}k(k+1)\,T_{slot}$ for minimized energy consumption.

Proof: We assume that in order to minimize energy consumption to respond to a query, a response solution from a minimum number of sensor nodes is required, even at the expense of a larger delay. We would therefore like to find an upper bound for this delay that minimizes energy. Suppose a query of length T_Q is sent to the sensor nodes and assume that $T_Q \leq 2T_{slot}$. In this case, there exists a single sensor that can reply to this query for a CD user because of the cyclic property and the fact that $T_{awake} = 2T_{slot}$ (from Lemma 1). The worst possible delay arises when the sensor, whose required data are stored in the longest continuous duration of active slots, has the next wakeup time furthest away from the time instant of querying. This is equivalent to the longest continuous duration of sleeping slots, T_{sleep}, in the schedule. By Lemma 5, this delay D_{ME}, is upper-bounded by $\frac{1}{2}k(k+1)\,T_{slot}$. ■

In the proof of *Theorem 4.1.2*, we have largely ignored that energy consumed by a sensor node to reply to a query is proportional to the length of each data packet. In fact, our assumption is that the energy required for a sensor to reply to a query is a constant given by some *E*. We make this assumption based on overheads incurred in each transmission and extra retransmission energies consumed due to possible packet collisions when more than one sensor node needs to reply to a single query from the same user that is only 1-hop away. Having a simplified model for energy consumption for replying queries also simplifies analysis and reveals underlying theoretical trends.

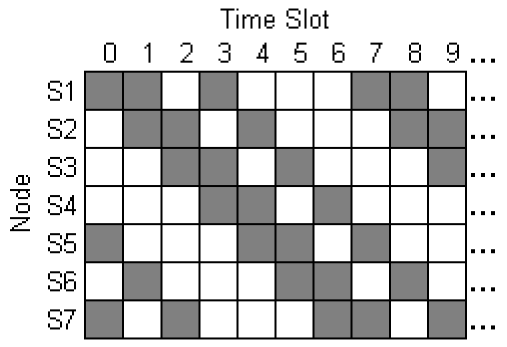
Theorem 4.1.1 is therefore a minimum QWD theorem while *Theorem 4.1.2* is a QWD bound theorem using minimum energy for CD users.

Theorem 4.1.3. For queries of length $T_Q \leq 2T_{slot}$, using a cyclic $(k^2+k+1,k+1,1)$ design, the energy required to reply to a CD user query is bounded by [E, 2E].

Proof: Since $T_Q \leq 2T_{slot}$, it is easy to see that the lower energy bound is simply *E*, because there exists a single sensor schedule that can provide all the query results. On the other hand, the upper bound is 2*E* because database information in a maximum time span of $2T_{slot}$ can always be contained in at most 2 different sensors. Therefore, only at most 2 sensors need to reply to the query. ■

Theorem 4.1.3 is thus an energy bound theorem to reply to a query from CD users. For clarity, we illustrate the physical interpretations of *Theorems 4.1.1*, *4.1.*2 and *4.1.*3

using an example. Consider a cyclic symmetric (7,3,1) design with k = 2 as illustrated in Figure 19.



Possible Combinations	QWD	Cost
{S1 ∪ S2}	T_{slot}	2E
{S1 ∪ S7}	0	2E
{S4 ∪ S2}	3T_{slot}	2E
{S4 ∪ S7}	3T_{slot}	2E
{S3}	2T_{slot}	E

Figure 19: Illustrating different possible solutions to the same CD query request in a (7,3,1) design.

Assume a CD user query of duration 2T_{slot} with interests in the data contained at times t = 2s and t = 3s time slots and that the current time is t = 7s. The sensor nodes {S1, S2, S3, S4 and S7} either contain partial or all relevant information of interest. In order to satisfy this criterion of the CD user, we need to reply with data from both time slots 2 and 3. Hence, possible combinations, their maximum delays and communication costs are given in Figure 19. As an example, consider the possible combination {S1 ∪ S2}. S1 contains data from time slot t = 3s and S2 contains data from time slot t = 2s. However, only S1 is awake at time t = 7s. In order to wait for the data from S2, time duration of one T_{slot} is required (when t = 8s). Hence, QWD is T_{slot}. Since this combination of {S1 ∪ S2} requires two sensors to respond to the query, cost is 2E. Using *Theorems 4.1.1*, *4.1.2* and *4.1.3,* there exists a zero QWD solution, the QWD bound for minimum energy is 3T_{slot} and the energy bound is [E, 2E]. The zero QWD solution of Figure 19 has energy of 2E and the minimum energy solution has delay of 2T_{slot}, within our formulated theoretical bounds.

We have so far described two solutions based on our CSBD design, one of QWD (*Theorem 4.1.1*) and one based on minimum energy (*Theorem 4.1.2*). In Figure 20, both are illustrated. For the zero QWD solution, the energy consumption increases with increasing user query lengths from a minimum of *E* to a maximum of 3*E*. As the query length increases, the required data have more chances of being stored in more sensors. Therefore, more sensors have to reply to the request to minimize the delay. For the minimum energy solution, the upper delay bound fluctuates with increasing query lengths T_Q, giving no clear trend at first. Intuitively, the length of the query should be one of the influencing factors because the longer the query length, the more information is required from the network and delay is expected to increase. This trend is not evident in Figure 20 because there is another competing factor. If we explore the total number of possible combinations, *NC*, of sensors available for satisfying the query request for different query lengths, we can discover that the delay has an approximate inverse relationship with *NC*. When *NC* is large, the likelihood of finding a lower delay solution is greater. When *NC* is small, this possibility become limited and often, this delay is not one that is small. These two factors contribute to the observed fluctuating delay behavior.

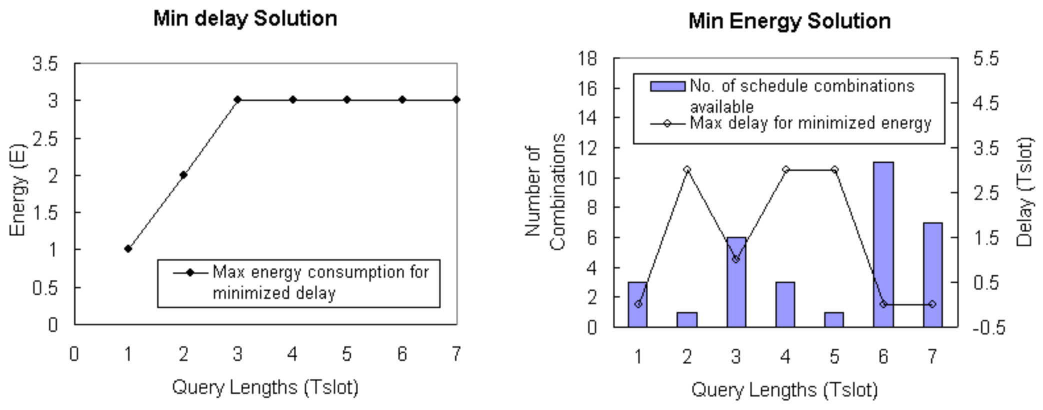


Figure 20: (a) Delay and (b) Energy Behaviours for varying CD user query lengths with the cyclic symmetric (7,3,1) design.

Theorem 4.1.4. For a cyclic (k^2+k+1,k+1,1) design and assuming AD users, the delay of a query on all collected sensor readings in the region is upper bounded by D_{AD} $=\frac{1}{2}(k^2+k+2)T_{slot} \approx \frac{1}{2}T_{cycle}$.

Proof: By *definition 4.1.2*, this is a worst-case scenario where the AD user demands every single piece of available data over the entire schedule cycle. It is sufficient to find the longest wait duration for all sensors in U_S to wake up at least once. By Lemma 7, all sensors must have at least one wake slot up within any duration of $\frac{1}{2}(k^2 + k + 2)T_{slot} \approx \frac{1}{2}T_{cycle}$ to answer the query. Hence, the result. ■

Theorem 4.1.5. For a cyclic (k^2+k+1,k+1,1) design, the energy expenditure per query request is (k^2+k+1)E.

Proof: AD users require all possible responses that exist in the region of interest. All sensors in the set U_S must respond to the query. Hence, energy is $(k^2+k+1)E$. ■

We note that the proofs of *Theorem 4.1.4* and *4.1.1* make use of the same Lemma 7. The same property in the symmetry of the cyclic symmetric design can be used to either guarantee bounded-time coverage, or if continuous time coverage is required, that property can be used to bound the worst-case QWD for AD users. Of course, the tradeoffs lie in the restrictions that are spelled out earlier as constraints: *The 1-Hop User Constraint*, *The Unique Schedule Constraint* and *The Design Space Spanning Constraint*.

Fortunately, all these constraints can be slightly relaxed for the practical purpose of sensor deployment and we shall discuss them later (section 4.2).

For the purpose of clarity, we term sensor nodes that employ CSBD for the objective of providing minimized query times in a sensor database as Database Wakeup Schedule Function (DWSF) Nodes.

4.2. Relaxation of Constraints

In section *4.1,* we recall that CSBD-based wakeup can guarantee bounded QWD times and offers a minimum query response energy solution for different classes of users (namely CD and AD users). However, all the results are only applicable if the three constraints “The 1-Hop User Constraint”, “The Unique Schedule Constraint”, and “The Design Space Spanning Constraint” are satisfied. In this section, we attempt to relax these constraints for practical purposes.

The 1-Hop User Constraint

The derivations presented for *Theorems 4.1.1 to 4.1.5* are based on the assumption that the user can immediately query the set of all sensors A_S with awake slots in some arbitrary time slot S. If this were not true, and that in order to query a sensor in the set A_S requires a relaying of information in one or more hops through another sensor not in the set A_S, additional QWD through these hops may be incurred and not been accounted for. The derived results will therefore not apply. However, this can be overcome by ensuring that the selection of slot time, T_{slot}, is sufficiently large so that no additional QWD will be incurred even though additional processing delays, $\tau_{process}$, and transmission delays, τ_{tx}, are required. This is reasonable only if $\tau_{process}$ and τ_{tx} are small compared to QWD. Alternative, another solution is to combine the benefits of this query-based solution with an agent-based approach (section 3.1.2) by ensuring that the agent is at most 1-hop away from the sensors of interest.

The Unique Schedule Constraint

This constraint can be removed only when there are sufficient nodes in the set U_S to span the cyclic symmetric ($k^2+k+1,k+1,1$) design. If there are more than k^2+k+1 nodes within sufficient proximity to measure the same event, those extra nodes can always re-use schedules from the same ($k^2+k+1,k+1,1$) design and all our derived results would still hold true.

As a consequence of this constraint, query-based solutions are best engaged in selected strategic locations in a sensor network that is primarily agent-based. We will illustrate this in a later section and this is also our recommended choice for implementation. Alternatively, the deployment density of sensors can be set very dense with sufficient sensor redundancies to ensure the set U_S spans the CSBD design set.

The Design Space Spanning Constraint

Suppose there are less than k^2+k+1 nodes of sufficient proximity to measure the same event. It is then necessary to have at least some of nodes in U_S to operate the union of multiple schedules. Such nodes can be treated as multiple "virtual nodes" located at exactly the same geographical location running multiple concurrent schedules using the same hardware. In this case, the duty cycles of sensor nodes in U_S are no longer all the same, but at least $\frac{k+1}{k^2+k+1}$. The set of "virtual nodes" and real nodes continue to maintain *symmetries 1, 2* and *3* in the wakeup design.

Again, we recommend that query-based solutions be employed relatively sparingly on top of a mostly agent-based system where additional storage and caching facilities in the network (from the query-based solutions) compliment an existing agent-based real-time monitoring or tracking application.

4.3. Complementing the Agent-Based System

As a consequence of the constraints discussed in section 4.2, a query-based solution where nodes may be queried for past information can only guarantee *Theorems 4.1.1 to 4.1.5* when the sensor density is relatively dense compared to agent-based systems. If sensor density is not a limitation, query-based solution can certainly be implemented as a standalone system in sensor networks. However, since sensor density directly affects implementation costs, as we have suggested, a better approach is to allow the query-based solution to be applied as a complimentary solution to an existing agent-based system. We propose a 2-tier sensor architecture for this purpose.

For clarity, we term sensor nodes that employ the agent-based solution for providing bounded-time coverage/connectivity, information delay and collision control as BTC nodes. We term sensor nodes that employ the query-based solution for providing bounded-time query delays and query response energies as DWSF nodes. It is important to note that both BTC and DWSF nodes essentially employ exactly the same CSBD-based wakeup, but only for two very different purposes. BTC nodes are involved in real-time monitoring or tracking of events, while DWSF nodes function as storage and caching facilities in the network to store past or important information about the network and its environment.

Consider a target tracking or intrusion detection application. As discussed in section 3.2, real-time tracking can be achieved by fusing measurements from one or more of these sensing modalities such as acoustic, magnetic or seismic sensors for reliable accuracy. Even within this tracking domain, seismic waves may travel at a different speed underground than acoustic waves in air, and underground seismic waves can sometimes travel a further distance than acoustic waves in a noisy environment and vice versa. The tracking problem is therefore not as straightforward as previously believed and sensing information from different modalities of the same target may not be available all at the same time. The importance of the role of DWSF nodes as a distributed storage or caching facility with bounded querying times to meet the real-time requirements of the data fusion application is therefore clear.

It is often not sufficient to merely track targets, but also to identify the nature of the target in intrusion detection applications. Chemical and temperature sensing information are often exploited to judge if a target is friendly or hostile. The diffusion speeds of chemicals and temperature from the target source to the sensors is always much slower than modalities used for tracking (such as acoustic or seismic). Moreover, the processing times for chemical testing in sensors are often slower than sound detection or vibration detection. The need for a sensor network to assume the role of a distributed storage database is obvious for caching measurements from slow sensing modes. Future in-network querying of these measured data by possibly decision fusion algorithms to identify threats is then possible. We propose a two-tier network for applications such as target tracking and intrusion detection, illustrated in Figure 21.

Suppose all BTC nodes are deployed randomly using seismic and acoustic sensors for tracking. With some computed value of k based on T_{res} requirement in (6), our network guarantees (m,T_1)-coverage, (m,T_2)-connectivity and per-hop delay of D_H, for some bounded T_1, T_2 and D_H derived in section 3.1.1. The deployment of DWSF nodes that sense chemicals and temperature require strategic placement on the field. A set of DWSF nodes is put under the charge of a leader node (which can be a BTC node). Each leader node is approximately 1 hop away from all DWSF nodes in each Query-Storage facility (see The 1-Hop User Constraint, section 4.2). In Figure 21, as the target leaves Query-Storage Facility 2, the leader node makes a query on its DWSF nodes to collect possible detections of chemical deposits within time bounds provided in section 4.1, and updates a tracking agent (section 3.2) as it propagates in the sensing field with the target.

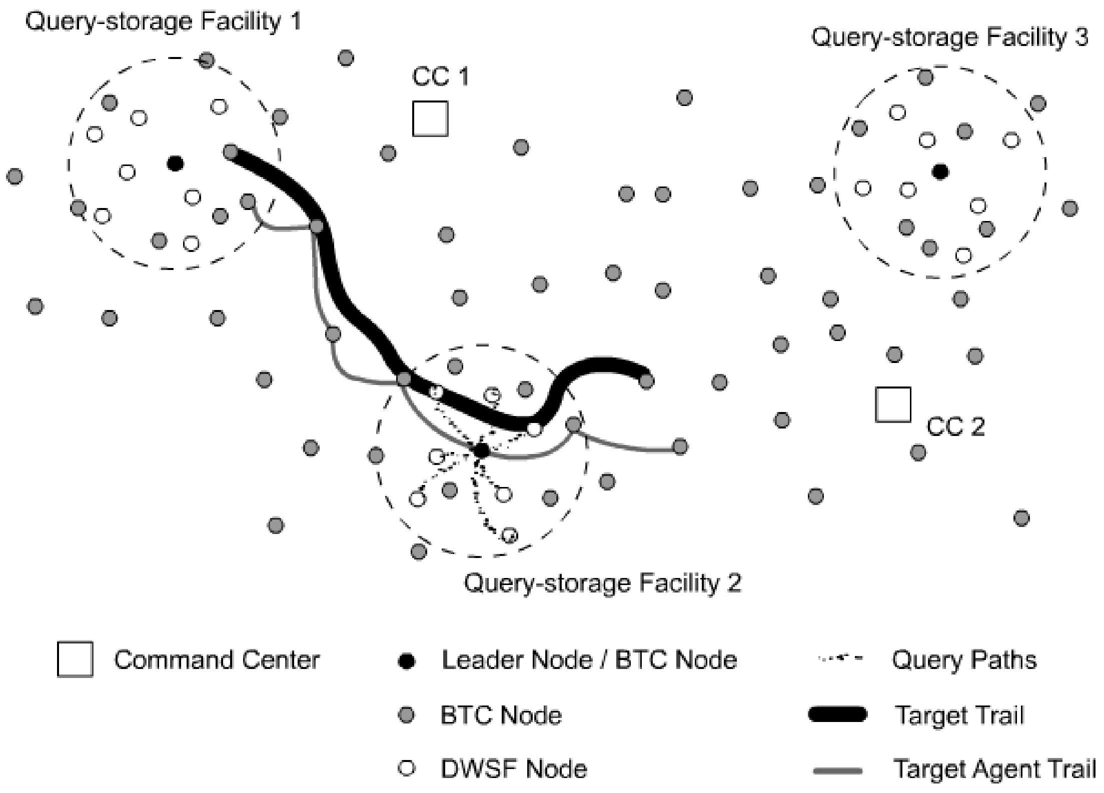


Figure 21: Illustrating BTC and DWSF nodes in a two-tier solution for a target tracking application.

4.4. Mobile Sensor Nodes

In our previous discussions, we have so far restricted sensor nodes to purely static nodes in the network which are not mobile after first deployment. Yet, in some situations, as the sensor network operates, uneven traffic load distribution in the network may cause early network partitioning and subsequently, reduce the usability and even network lifetime of the system. Inspired by the grouping behaviour of Emperor Penguins in the Antarctic region, the authors [62] proposed a distributed mobility algorithm – Energy-Aware Swap Protocol (ESAP) for wireless nodes. The algorithm draws parallel between the heat loss of a penguin and the routing energy burden of wireless nodes, where neighbour nodes swap positions with each other depending on their energy levels. This is indeed a valid reason why sensor nodes need to be mobile. The requirement of having a sensor network that provides guarantees in terms of delay, coverage and connectivity, together with the requirement of having sensor nodes to be mobile may become a real challenge.

Fortunately, for our proposed wakeup schedule, our solution is simple. We note that our results for BTC nodes in section 3.1.1 and 2.4 are based on sensor nodes randomly selecting a schedule from a cyclic symmetric ($k^2+k+1,k+1,1$) design. There is no requirement that nodes must operate unique schedules (unlike DWSF nodes). There is no difference if we were to interchange the schedules of any two arbitrary nodes in the network. There is therefore, no difference if we were to interchange any two nodes in the network. Coverage, connectivity and per-hop delays are still guaranteed (or preserved) by the governing equations in sections 3.1.1, 2.4 and 3.1.2. For results in section 4.1 applicable to DWSF nodes, all the derived results apply as long as nodes or schedules are interchanged only between DWSF nodes within the proximity that measures the same event. This is the case for our proposed two-tier architecture in Figure 21 where such DWSF nodes are deployed within the same region.

ESAP is therefore an algorithm that promises to work with both BTC and DWSF nodes that employ the CSBD wakeup design for two different purposes. Our proposed system works equally well for static sensor nodes, as well as for sensor nodes with limited mobility for load/energy balancing.

4.5. Database Wakeup Schedule Function (DWSF)

We have therefore proposed the Database Wakeup Schedule Function (DWSF) as a sensor wakeup scheme for sensors to sense and store information about the environment for future querying operations. DWSF is based on CSBD, and provides bounded QWD and other properties to different user classes. In particular, for the CD user class, this QWD bound is zero, ignoring processing and propagation delays (*Theorem 4.1.1*). We further provide simulation results and implementation results and experiments with real motes [1]. As we have previously argued our preference for DWSF nodes to be deployed to compliment an agent-based system, simulation scenarios presented in this section will be focused on the two-tier implementation presented in Figure 21.

4.5.1 Simulation Results

Simulation results in this section focus on the two-tier architecture as described in section 4.3 using the target tracking and intrusion detection application example that combines BTC and DWSF nodes, both employing the CSBD wakeup design, but with different objectives.

<u>Simulation Setup</u>

5,000 BTC nodes are deployed uniformly distributed in a flat geographical area of about 1km by 1km. BTC nodes provide the necessary routing of information and data in a multi-hop manner from one part of the network to another. BTC nodes are equipped with multiple mode sensing capabilities, consisting of measuring acoustic signals and seismic signals from targets. Data from both sensing modes are later fused with the use of a data fusion algorithm to estimate the location of the target. BTC nodes therefore also guarantee that connectivity and sensing coverage is bounded within some known finite time (*Theorem 3.1.1.1* applies). All BTC nodes randomly choose a schedule from the cyclic symmetric (21, 5, 1) wakeup design, with a slot time of T_{slot}=0.01s (unless otherwise specified).

Ninety-one DWSF nodes are arranged in thirteen groups of seven nodes each. Since DWSF nodes serve as Query-Storage facility in the network, each group of DWSF nodes may be arranged evenly on the circumference of a circle of radius R_S, where R_S is the sensing coverage radii of the DWSF sensor nodes. For the purpose of simulation, the center of the circle for each group of DWSF nodes is termed as a landmark and is determined randomly in the simulation area of 1km by 1km. In reality, the positioning of each group of DWSF nodes should be strategically located near important landmarks on a real geographical map, such as where major road crossings and sensitive areas requiring security validations are found. DWSF nodes are equipped with sensing chemical deposits to identify if targets are friendly or adversary. All DWSF nodes in the same group span the cyclic symmetric (7, 3, 1) wakeup design, with a slot time of T_{slot}=0.5s (unless otherwise specified).

For simplicity, we set the sensing radii of all nodes to be the same at R_S = 50m and their communication radii set to R_C = 120m. We assume that all nodes have acquired knowledge of their own position in the sensor field for any target tracking to be meaningful.

For the acoustic signal model, we adopt, as in [43] the acoustic energy decay function in free and homogenous space given by:

$$z_n^{aco} = g_n \frac{S}{\left\| y - r_n \right\|^2} + nz_n \tag{27}$$

where z_n^{aco} is the acoustic signal received by the nth sensor, nz_n is the noise term that summarizes the net effects of background additive noise and modeling errors. g_n and r_n

are the gain factor and location of the nth sensor, respectively. S and y are respectively, the energy emitted by the source and its location.

For the seismic signal model, we similarly adopt the signal attenuation model as in [43] given by:

$$z_n^{sei} = \frac{S^{sei}}{e^{\kappa d}} \tag{28}$$

where d is the distance, and κ is attenuation factor, being 0.0013 – 0.003 in [75]. In our simulations, we choose κ = 0.00215. All other parameters are set to values as provided in [43]. The fusion algorithm, based on a particle-filtering approach, is also given as the Cross-Sensor Cross-Modality (CSCM) Data Fusion Algorithm in section 3.2 and [43]. The elegance of such a data fusion approach requires only information from the previous time step and measurements from the current time step to compute the estimation of the required target location. Such an approach has also been proven to provide good results even in the presence of non-Gaussian noise and non-linearity in the target motion. Moreover, signals from different modalities can be fused in the same consistent mathematical framework to ensure superiority in terms of tracking accuracies in the estimations. Using (6), it can be shown that the data fusion iteration time step can be set to 0.5s.

For chemical sensing in DWSF nodes, we assume the concentration of chemicals is diffused in free space in the absence of wind. This relation is given by [76]:

$$z_n^{chem} = \left(\frac{C}{t^{3/2}}\right)\exp\left(\frac{-\left\|y - r_n\right\|^2}{4Dt}\right) \tag{29}$$

where z_n^{chem} is the number of chemical trace particles received by the n^{th} sensor, C is the number of chemical trace particles at the source and D is a diffusion constant. For simulation purposes, D=1 and C=10^6. This sensing mode is not fused with the other signals because its role is solely the identification of the target type as friendly or adversary, rather than for resolving target location. A DWSF sensor node will decide independently the presence of an adversary target if the detection concentration is such that $z_n^{chem} > 1000$. Depending on the target-sensor distance $\|y - r_n\|$, the detection time is of the order of seconds to minutes. As the target traverses into a group of DWSF nodes (landmark), the agent issues a CD query (unless otherwise indicated) on the DWSF nodes requiring chemical trace reports. DWSF nodes in the group subsequently monitor the region for such chemical traces only when they are in the "Awake" mode for a total duration of 10s, managed by a leader node (which is a BTC node within one-hop from all DWSF nodes in the group). Any reports of harmful chemical traces reported to the leader node from the group within the 10s window will be forwarded to the agent for target identification. For simplicity, we assume that a single report of harmful chemical trace from a single DWSF node shall trigger the identification of the target as an adversary. As such, this justifies the use of CD queries (unless otherwise indicated). *Theorem 4.1.1* therefore applies.

Three landmarks (where each landmark is represented as a group of seven DWSF nodes) in the sensor field are randomly selected to determine the target path. The origin of the target is randomly selected in the sensor field when it approaches each of the three landmarks in turn. The target motion model from the target origin to the first landmark, and from one landmark to the next, is determined by the Random Waypoint motion model [77]. The average target speed is 12m/s (or 43.2km/h) and the maximum target speed, v_{max}, is capped at 20m/s (or 72km/h). We only focus on a single target at any one time in our simulations. Once a target has stopped (at the last landmark), it vanishes and a new target is generated in the sensor field after a pause of 1s. Simulations are performed until the first BTC node in the network is completely depleted of energy.

In our simulations, the Agent-based tracking approach [33, 38] is used where the agent is responsible for the data fusion using the CSCM algorithm, target identification, and is implemented as a software code that propagates in the network with the target as it moves through the network (see Figure 21). Hence, all information is routed towards the mobile agent for computation of the target location and for its identification. As the agent passes through one BTC node to the next, it leaves a "passer-by marker" (PBM) in that node for some stipulated time τ_{PBM}. The PBM serves as a trail for delayed queried information from DWSF nodes to find its way to the mobile agent. Data from DWSF nodes are forwarded to its associated agent in the form of a limited broadcast where only BTC nodes with an associated PBM may re-broadcast the data to its neighbors. Further, in our selection of parameters, we have ensured that $R_C > 2R_S$ and $v_{max} \leq R_s/T_{cycle}$, (see (9)) so that it is assured that target location information can reach the agent from BTC nodes within one communication network hop.

Performance metrics includes information delay, network lifetime, accuracy of target tracking and accuracy of target identification. Note that some of these metrics rely on good sensing coverage and connectivity of the network, which we have analyzed rigorously for our proposed scheme. These metrics thus serve as performance benchmarks for comparisons between our proposed two-tier BTC and DWSF wakeup architecture and two other wakeup schemes, RAW [12] and PECAS algorithm [21]. RAW is selected as a representative algorithm from the random wakeup schemes category and PECAS is selected as a representative algorithm for the on-demand wakeup schemes category. Similar to our two-tier BTC and DWSF architecture, both RAW and PECAS are time-asynchronous schemes.

The RAW scheme requires nodes to randomly wakeup to sense and route information in the network. No synchronization of clocks between individual sensor nodes is required. The PECAS scheme is also time-asynchronous where nodes start working only if there is no other working neighbour node within a minimum specified distance. For PECAS, a node can be defined to be working if it is a designated network node for forwarding data in the network, or equivalently, in the “Awake” mode. PECAS nodes that are not in the “Awake” mode may then wake up periodically to check that they can remain in the same operational mode. To avoid unbalanced energy usage in the network, PECAS allows sensor nodes that are already in working mode to go back to “Idle” mode after some time so that other neighbouring nodes may wakeup to perform the duty of sensing and routing. With the use of a dual radio architecture (see section 1.3.1), it is therefore important to acknowledge that PECAS is very close to the ideal scenario of having perfect sensing coverage subjected to the initial sensor deployment arrangement, and having perfect connectivity, subjected only to extra delays and overheads used to switch nodes from the “Idle” to the “Awake” mode. The performances of this scheme serve as benchmarks for our proposed scheme to achieve, while in an attempt to extend network lifetime substantially and without the additional cost of a second radio channel.

To ensure a fair comparison, the same random network deployment scenario is used for the simulations for all three different wakeup architectures. The routing protocol is the geographic routing (GR) scheme [71] employing a store-and-forward approach to data forwarding. Data is stored and only forwarded to the next node in the routing table at the next available opportunity when the neighbour node is awake. Loss packets are not retransmitted due to the real-time requirement of the application. We further provide simulation results that include the implementation of the energy-balancing algorithm ESAP (see section 4.4) for BTC nodes in our proposed solution.

Delay and Network Lifetime

In the use of the term “information delay” in this section, we refer to the time duration from which data packets are transmitted upon detection of a target, to the arrival of those packets at the agent. We ignore all packet losses and we do not consider the accuracy of the information in the packets for the moment.

For the performance metric – network lifetime, we assume that the network lifetime is the time duration from the start time of the simulation to the time at which the first BTC node is completely depleted of energy.

It is important to acknowledge that information delay in the network is closely related to the network lifetime by tuning several parameters of each wakeup scheme. For instance, each RAW node may be tuned to randomly wakeup less often to conserve energy, but the tradeoff lies in the speed at which information is able to propagate in the network. Although PECAS is not as tunable as the other schemes, working nodes may be made to switch to the "Idle" mode more often to better balance the energy, and hence, improve network lifetime.

In an attempt to ensure fair comparison of "information delay", we perform several simulations with each wakeup scheme until the network lifetimes of each scheme are approximately equal (within 5% error from each other). Figure 26 shows the delay performance for two types of data, namely target tracking packets (TTP) and target identification packets (TIP). TTP originates from BTC nodes while TIP originates from DWSF nodes.

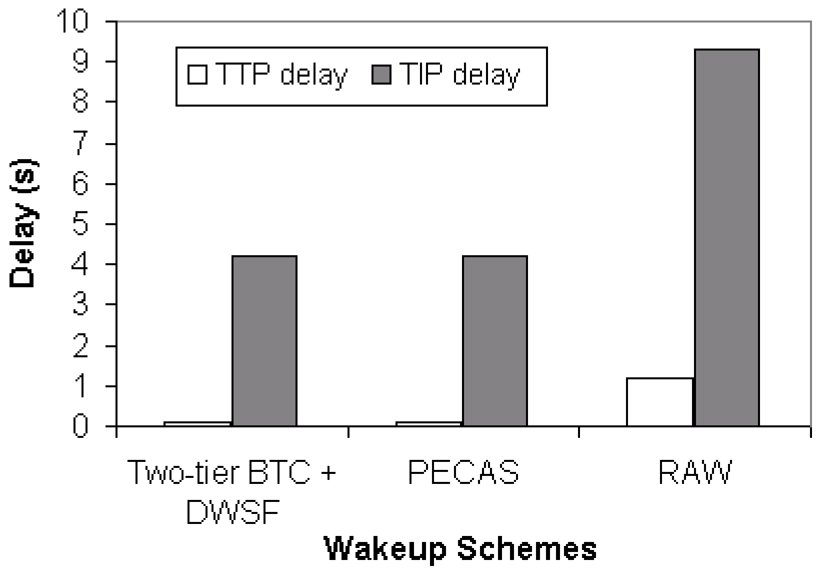


Figure 22: Delay performance for different wakeup schemes for different packet types. In the simulations, the network lifetimes of all three schemes are within a 5% difference from each other.

From Figure 22, our two-tier solution shows a negligible difference in terms of delay when compared with PECAS. Information about the target location (TTP packets) reaches the mobile agent (which moves as the target moves) in approximately 0.11s for both schemes. We further discovered that it is possible to substantially improve network lifetime for our two-tier solution and TTP delay remains approximately the same. Moreover, unlike PECAS, our two-tier solution comes at no extra overheads in terms of a second monitoring channel for the "Idle" mode.

RAW TTP delays are about 1.2s, that is in excess of 4.5 times that of our proposed solution. The randomness of the RAW solution provides no per-hop delay guarantees and average longer wait periods between hops are experienced.

In all three schemes, TIP packets arrive substantially later than TTP packets because chemical diffusion times are much slower than sound waves and ground waves propagation.

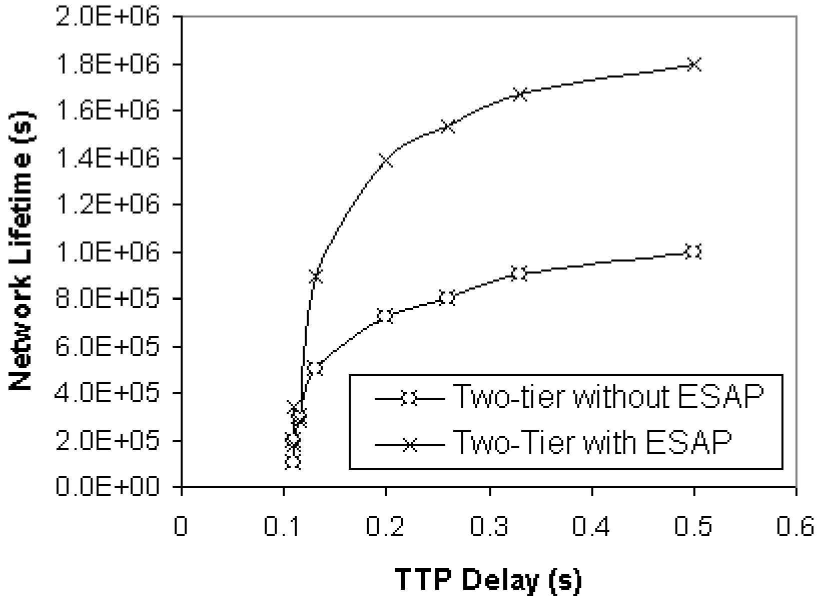


Figure 23: Tradeoff between network lifetime and packet delay for the proposed two-tier BTC + DWSF scheme, with and without energy balance algorithm ESAP (section 4.4).

To further understand the tradeoff between TTP delay and network lifetime, Figure 23 shows that a TTP delay of approximately 0.11s can be maintained even when the network lifetime is extended by almost four times. The curve without ESAP is used in reaching this conclusion. The ESAP curve will be discussed in a later subsection. A lifetime extension of four times cannot be repeated with the PECAS scheme. In fact, PECAS cannot be as readily fine-tuned as the other schemes to achieve slower delays so that network lifetime may be improved. In Figure 23, as the TTP delay increases, the network lifetime improvement increases at a decreasing rate.

Target Tracking and Target Identification Accuracy

While the delays in which information reach the mobile agent is important, the accuracy of that information reaching the agent is equally crucial. Here, the slot time has been set to T_{slot} = 0.01s for our two-tier BTC + DWSF solution so that the TTP delay is approximately 0.25s with an average network lifetime that is expected to last 8 times longer than PECAS. Our objective is to show that even with such a configuration, target tracking accuracy and target identification accuracy is comparable to PECAS and substantially outperforms that of RAW.

We first define the Target Tracking Accuracy (TTA) as the percentage of time (over the entire simulation time per simulation) in which the predicted target location is within a distance error of δ from the actual location of the target at any one time. For our simulations, we set δ = 5m.

We further define the percentage of False Negatives (PFN) as the number of times an adversary target has been mistakenly identified as a friendly target taken as a percentage of the total number of targets in one set of simulations. PFN arises when information about the identification of an adversary target does not reach the agent. We further assume that an adversary target will be mistaken as a friendly target if its associated TIP packets do not reach the target within a stipulated time of ω_{TIP} = 15s.

Table 3: Comparing target tracking accuracies (TTA) and identification errors (PFN) across different wakeup schemes. Both the proposed Two-tier scheme and RAW has a network lifetime that is about 8 times longer than PECAS when the comparisons are made.

Schemes	TTA	PFN
Two-tier BTC + DWSF	95.3%	1.86%
PECAS	96.2%	1.24%
RAW	71.0%	9.98%

In Table 3, we show that despite our proposed two-tier scheme having a low duty cycle such that network lifetime lasts about 8 times longer than PECAS, their TTA and PFN are very similar. TTA is affected by the availability of sensed target data and network connectivity. We have earlier proven (section 3.1.1) that coverage for two-tier BTC + DWSF is guaranteed given equation in (6) is satisfied. TTA for two-tier is therefore similar to that of PECAS. For RAW, the tracking inaccuracies of about 29% arises mainly from information arriving to the agent at a later time, thereby causing larger errors between the perceived target location and the ground truth.

We have also shown earlier (section 4.1) that since QWD for CD queries is zero (*Theorem 4.1.1*), the arrival of packets to the mobile agent in both the two-tier and PECAS scheme are very similar in the absence of such query waiting times. Such PFN errors are largely due to packet losses that are not re-transmitted. Such errors can thus be effectively reduced with the implementation of mechanisms to retransmit TIP packets. Target identification errors for RAW as depicted in PFN performs poorly, in comparison, due to lack of such specific guarantees when queries are made. Moreover, since TIP data arrives much later than TTP data, the randomness in which RAW nodes wakeup add to the delays in which such queried packets are delivered to the mobile agent.

We have also performed two other sets of simulations with appropriate parameters such that two-tier BTC + DWSF have network lifetimes that are approximately (within 5% error) 4 times and 12 times that of PECAS.

Table 4: Target tracking accuracies (TTA) and identification errors (PFN) for the proposed Two-tier scheme when its network lifetime is 4 times, 8 times and 12 times that of PECAS.

Schemes	**TTA**	**PFN**	**Network Lifetime**
Two-tier BTC + DWSF	73.24%	8.74%	$\approx 12 \times$ PECAS
Two-tier BTC + DWSF	95.3%	1.86%	$\approx 8 \times$ PECAS
Two-tier BTC + DWSF	95.31%	1.3%	$\approx 4 \times$ PECAS

Table 4 shows the results. When the network lifetime is 12 × PECAS, equation (6) could not be satisfied using the same application value of T_{res} = 0.5s, and the TTA falls significantly because sensing coverage is no longer guaranteed within the time resolution requirement of the application. The longer cycle time in the design further affects the timeliness in which TIP packets reach the agent, thereby causing poorer performance in PFN.

When the network lifetime is 4 × PECAS, equation (6) is satisfied but resulting in no further improvement in TTA compared to the 8 × PECAS case because the latter case is already one in which nodes have slot times that are sufficiently small not to be noticeable by the data fusion application.

Energy Balance with ESAP

We further implement the energy balance algorithm ESAP on BTC nodes in our two-tier architecture to study the effects on network lifetime. As expected, Figure 23 shows that with ESAP, network lifetime can be extended by another 1.787 times on average because nodes positions are exchanged to balance energy usage in the network so that the time at which the first BTC node is completely depleted of energy happens much later.

A Study of CD and AD Queries

We have previously assumed that a single report from a single DWSF node detecting sufficient harmful chemical traces will trigger the belief that the target is adversary. In this set of simulations, we assume that all reports from the same group of DWSF nodes near the same landmark must be collected before forwarding to the mobile agent to reach a conclusion on the identification of the target. As such, AD queries must be issued and the QWD is then bounded by the result given in *Theorem 4.1.4*. For simplicity, we assume that for as long as more than half the nodes belonging to the same landmark detect the harmful chemical traces, the target is marked as adversary.

Table 5: Comparing CD queries and AD queries for the Two-tier BTC + DWSF scheme across different performance metrics.

Metrics	CD queries	AD queries
TIP Delay	4.2s	6.02s
TTA	95.3%	91.77%
PFN	1.86%	2.47%

Since the QWD bound on AD queries is approximately half the cycle time on the DWSF schedules (larger than the QWD bound for CD queries, which is theoretically zero), the TIP delay increases by about 43.3% (from 4.2s to 6.02s). This analytically computed QWD bound for AD queries using *Theorem 4.1.4* is 2s. Note that this bound should not be compared directly with 6.02s because the latter includes processing delays and multi-hop propagation times from the storage facility to the agent. This increase in delay subsequently affects tracking accuracies, TTA, as well as target identification errors, PFN. TTA is therefore reduced from 95.3% to 91.77% and PFN errors are increased from 1.86% to 2.47%. Still, these tracking accuracy and identification errors are reasonable and remain acceptable.

4.5.2 Implementation Experimental Results

We implemented a small network of eight DWSF nodes using real Crossbow MICA2 motes [18] with one leader node and seven member nodes. The leader node is assumed to be always "Awake" with the remaining seven motes running the standard CSBD (7,3,1) wakeup scheme (see Figure 1(i)). Both query delay and energy measurements were made.

Delay Results

The leader node issues time-stamped queries from both CD and AD users to its member nodes. The times (as measured by the leader node) at which member nodes reply the queries are measured from this experiment. Figure 24 shows the delays encountered for member nodes to reply to queries with different slot time values. These delays therefore correspond to the time between the instant the leader node issues a query and the instant it receives a reply from each member node that is in the "Awake" mode in the region.

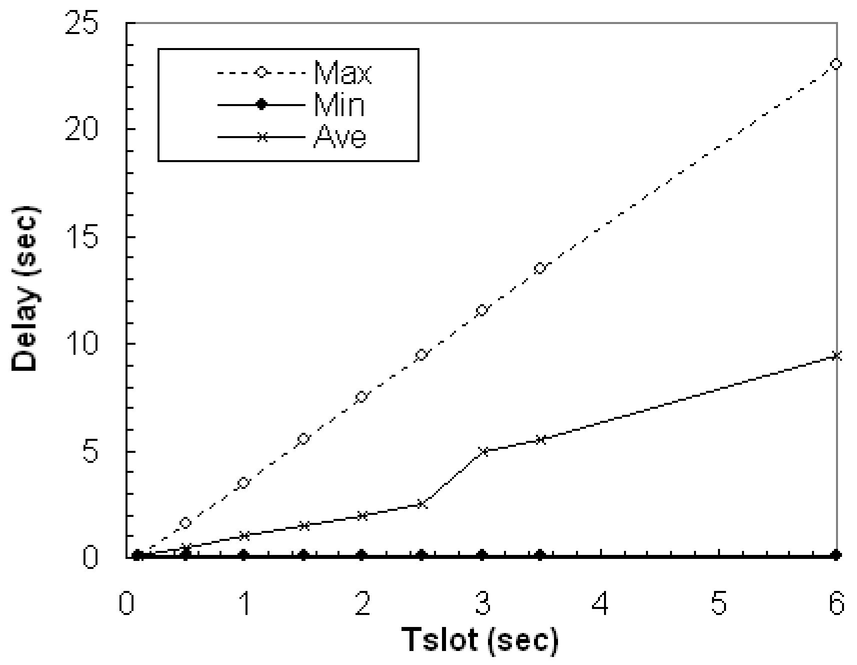


Figure 24: Query delays for slot times. T_{slot} = 2 seconds

We observe that the maximum value of this delay is bounded by *Lemma 7* or $4T_{slot}$. That is, the delay is proportional to the duration of the longest "sleeping" period of the

CSBD (7,3,1). We notice that the minimum delays observed are not exactly equal to zero but are slightly greater than zero (several milliseconds). This is due to the fact that some delay is incurred by the processing of the received queries before the motes can send their replies as well as by the communication between the leader node and its member nodes. However, these results prove that our solution implies no additional delay beyond these unavoidable delays, verifying *Theorem 4.1.1*.

Figure 25 shows the delay for AD queries for different varying query lengths (see *Definition 4.1.3*). These queries are requests for data from measurements at *TslotAgo* time slots in the past for several periods of interest (see definition of T_Q in section *4.1*). The average value corresponds to the average over 300 queries run with random values of *TslotAgo* and random inter-arrival time between two queries. The value of T_{slot} was fixed at 2 seconds for this simulation. These results show that for queries of lengths between 2 and 4 T_{slot}, the minimum value varies almost linearly with the query length before becoming nearly constant. The maximum and average delays are also constant for queries greater than or equal to 4 T_{slot}. This can be explained based on the chosen schedule. In particular, we observe that the maximum value of the delay is nearly constant and slightly greater than 6 = 3 x T_{slot} (except for the query of length greater than 6 T_{slot}). This result is coherent with our bound in *Theorem 4.1.4*. The maximum delay corresponds to the delay of the case where the leader node always has to wait for the data from a member node that has just entered its longest "sleeping" period of the schedule. The increase of the maximum delay for queries longer than 6 T_{slot} is due to the delay introduced by packet loss due to congestion. In fact, the longer the query, the more data has to be transmitted. Still, it is within the bound we claim in *Theorem 4.1.4*.

We do not show the case of CD queries, as their delays are all slightly greater zero.

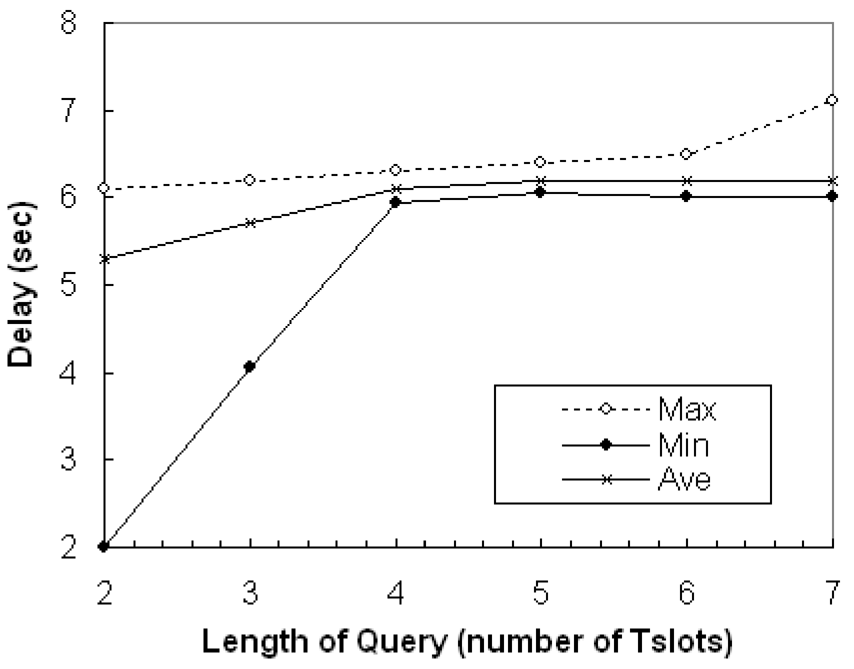


Figure 25: Query Delay for Queries of Different Query Lengths. T_{slot} = 2 seconds.

Energy Results

For energy measurements, the leader node sends requests to member nodes requesting for data in the current time slot. For each result, 400 queries (both CD and AD user queries) were sent at random times by the leader node and we averaged the energy spent by the motes to reply.

Table 6: Ratio of energy spent by a mote with CSBD over the energy spent by a mote that is always "Awake".

T_{slot}	0.1 sec	1 sec	2 sec	4 sec	6 sec	10 sec
Ratio (%)	62.57	61.15	60.15	59.46	59.7	59.61

Table 6 reveals that the energy results for six different T_{slot} values. As expected, these values are all greater than the theoretical value of 58.36% based on the MICA2 mote power benchmark number quoted by the PowerTOSSIM [79] group. This is due to extra energies consumed to turn on and off the radio. Logically, we observe that this added energy is less significant if we consider higher values of T_{slot}.

4.6. Summary

Treating sensor networks as a distributed database of information has opened up a wealth of research opportunities in the subject. While existing work focus on query language, query routing, data aggregation and in-network processing techniques, we discovered yet another problem to solve when such a sensor network is coupled with an underlying energy conservation wakeup scheme. We analyzed query waiting delays and query response energies for different classes of users/applications using CSBD, and propose to use this solution to compliment an agent-based sensor network. Although both agent-based and query-based sensor networks are based on the same CSBD design, they have been used for a very different purpose. CSBD wakeup in BTC nodes primarily ensures bounded-time sensing coverage and bounded multi-hop tracking delays in agent-based sensor networks, and CSBD wakeup in DWSF nodes primarily ensures bounded-time query delays. Both solutions can be applied in a single two-tier architecture to complement each other. We have also shown that our wakeup scheme work equally well with static nodes, as well as with mobile nodes. We tested our proposed solution with extensive simulation results and an actual implementation of the CSBD design to real sensor motes to make measurements.

In our two-tier architecture simulations, target tracking delays from the target to an agent that is one-hop away is as low as 0.11s. Target identification delay is also almost as low as the idealized PECAS algorithm. Yet, our solution promises to ensure a network lifetime that can be several times longer than PECAS. Our solution achieves 95.3% target tracking accuracies and 1.86% target identification errors. We show that with energy balancing in the network, lifetime can be further extended by about 1.787 times on average.

In our implementation results, we performed delay and energy measurements with real Crossbow motes. We verified the zero query delay theorem (*Theorem 4.1.1*) and tested other delay bounds on different classes of user queries and different slot times. We also showed that the ratio of energy used by a mote with CSBD to the energy used by a mote that is always “Awake” is higher for smaller slot times. This arises because more energy is required to switch motes between the on and off modes when slot times are shorter.

Chapter 5: Ad-Hoc and Sparse Sensor Networks

Agent-based and query-based systems have been designed for relatively denser networks. In some scenarios, a dense deployment of sensors is not practical, usually because of cost concerns. This chapter is about sparse networks that are deployed in an ad-hoc fashion (e.g. in harsh environments), and how wakeup schemes should be designed for this purpose. By the term ad-hoc, we refer to nodes being deployed in a more or less random manner, without prior order or organization. The term Ad-hoc is *not* meant to refer to a network with mobile nodes in this chapter. In section 5.1, we illustrate scenarios where ad-hoc and sparse sensor networks are often deployed. We explore the limitations of existing wakeup schemes in the context of an ad hoc and sparse network in section 5.2. We subsequently introduce our proposed wakeup scheme in section 5.3, and discuss asynchronous neighbour discovery issues in section 5.4, data transmissions and algorithm maintenance issues in section 5.5, network connectivity and sensing coverage issues in section 5.6 and other algorithm properties in section 5.7. We further justify our algorithm with a specific application scenario in section 5.8. We summarise this chapter in section 5.9.

While solutions in previous chapters rely on the cyclic-symmetric property of CSBD, this chapter proposes another solution that deviates from this property. However, our solution remains related to CSBD and remains a deterministic wakeup scheme.

5.1. Scenarios for Deployment of Ad hoc and Sparse Networks

Ad hoc and sparse sensor networks may be deployed for a variety of reasons. For instance, in a sensor search and rescue mission, such networks may be deployed in hostile environments with unreliable communication channels. Examples of these hostile environments may include fire-fighting environments, floods, underwater communication, underground communication and even communication on other terrestrial planets.

In a fire-fighting environment, communication may be temporary or permanently disrupted due to physical changes in the environment and even physical damages to sensors. Collapsing structures in a fire may also cause frequent changes in the network topology and change network size as sensors are destroyed. An initially dense network in such hostile environments may become sparse with time and algorithms or protocols designed for such purposes must be able to adapt. Similarly, a high altitude land rescue mission using sensor networks affected by adverse weather conditions can also pose challenging communication problems. In underwater communication networks, the communication channel is primarily acoustic and it may be subjected to severe fading and multipath effects, and easily susceptible to environmental noise. Noise sources may originate from snapping shrimps [63], blue whales and other aquatic animals, and man-made sources including motors of large commercial vessels and underwater drilling activities. Another limitation with acoustic communication is its low data rate compared to electromagnetic wave propagation. For space exploration projects, communication can also be a big issue on planets where atmospheric pressure and air composition is largely different from Earth’s. Replacement of batteries on distant planets is almost prohibitively expensive and sensor networks that are proposed for this purpose [64, 65] must take into consideration such energy constraints and communication challenges.

However, in many situations, it is cost concerns that explain why sensor networks are deployed sparse. Very often, sensors that are to be deployed in harsh environments are not expected to be recoverable, recyclable or reusable. Deploying such sensors in large numbers can be commercially unviable. Where sensors may be recoverable and reusable, they may still require special protection casing and impact-proof designs that adds to the cost of deployment. In this case, dense deployment of these specialized sensor networks is similarly not possible.

5.2. Limitations of Existing Wakeup Schemes

We take the underwater environment scenario as one example to explain limitations of existing wakeup schemes in more detail.

One major difference between land and underwater systems is the medium of propagation. Land systems communicate wirelessly over electromagnetic radio channels where bit rates can reach as much as 250kbps [66] for MICAz IEEE 802.15.4 radio systems. In contrast, underwater communications can only be achieved using acoustics propagation. The WHOI modem [67] for instance has a data rate of only 80bps or 300-5000bps, some three orders slower. This single difference has important implications. First, achievable data transmission speed becomes slower and this makes any time synchronization efforts very challenging, if not impossible. Second, underwater acoustics communication is largely affected by many external factors including water temperature, water pressure, and more prone to diffraction and fading effects, thereby inducing unreliability in the communication channel. Third, since data rate is significantly slower underwater, more energy may be consumed to transmit the same volume of data compared to on land transmissions. Energy conservation using dual radio systems where hardware is put on standby using the “Idle” mode may not be practical.

Owing to slower achievable data transmission rates in the underwater context, schemes that require time synchronization to operate such as RIS [13] will not be possible. With nodes spaced far apart, synchronization will become even more challenging with acoustics-based communication. Electromagnetic-based communication underwater is known only to travel over very short distances [69] and is only suitable for short-range underwater communications. Unless the network is dense, such techniques remain not realizable.

The unreliability of the underwater channel presents a very difficult environment for wakeup schemes that require substantial amounts of control overheads to operate. With frequent and unpredictable losses in control packets, schemes that rely on recurrent communication with its neighbouring nodes to acquire information to adjust its own wakeup schedule will not prove too useful. Such schemes may include the “Paging Wakeup” class of schemes such as STEM [17], PAMAS [19] and PECAS [21]. Moreover, these schemes require a dual-communication channel to put nodes to the “Idle” mode to monitor the channel and the cost of such technologies for underwater can be exceedingly high as it is not default or common technology.

Another major difference between land and underwater sensor networks is the node form factor. Underwater nodes still continue to be of a larger form factor size compared to the increasingly popular small mote-sized form factor of land sensors. Although inter-sensor node communication may be more challenging underwater, storage capacities, computation and processing capabilities of such nodes are often superior than those of mote-sized land sensors, subjected to energy constraints. Due to their larger form factor and special water-sealing requirements, they are more costly and realistically, result in sparser deployment densities. A single Crossbow MICAz mote cost $300 [68] each while that of an underwater acoustic modem alone can costs in excess of $3000 [70], an order of 10 times more.

Due to its large form factor and high deployment cost, underwater network schemes that rely on a dense deployment are not practical. These schemes, including ASCENT [24], PEAS [20], PECAS [21], RAW [12] etc., will not work as expected in a sparse deployment scenario compared to a dense deployment. Even for our proposed deterministic Agent-based and Query-based schemes in chapters 3 and 4 will not be ideal, for they are not designed to operate under such conditions.

Cost, coupled with the fact that more energy is usually expended per data bit transmitted underwater than on land, explains the true importance of energy conservation underwater despite having more energy storage due to their larger form factors. Moreover, changing of batteries for wireless nodes in UWSN is far more laborious and costly compared to land systems. Energy conservation techniques proposed for land systems are unsuitable for underwater deployment due to insufficiently realistic assumptions or different ambient operating conditions. This motivates us to propose a separate scheme for sparse networks.

5.3. Adaptive Wakeup Schedule Function (AWSF)

In dense networks, the neighbour count per node is large. The probability of a sensor node waking up to discover that are no other nodes in its communication range is small. It is appropriate at this point to state:

Definition 5.3.1. The "Lonely Node Problem" is the phenomenon when nodes wakeup to find no other neighbour nodes within its communication range.

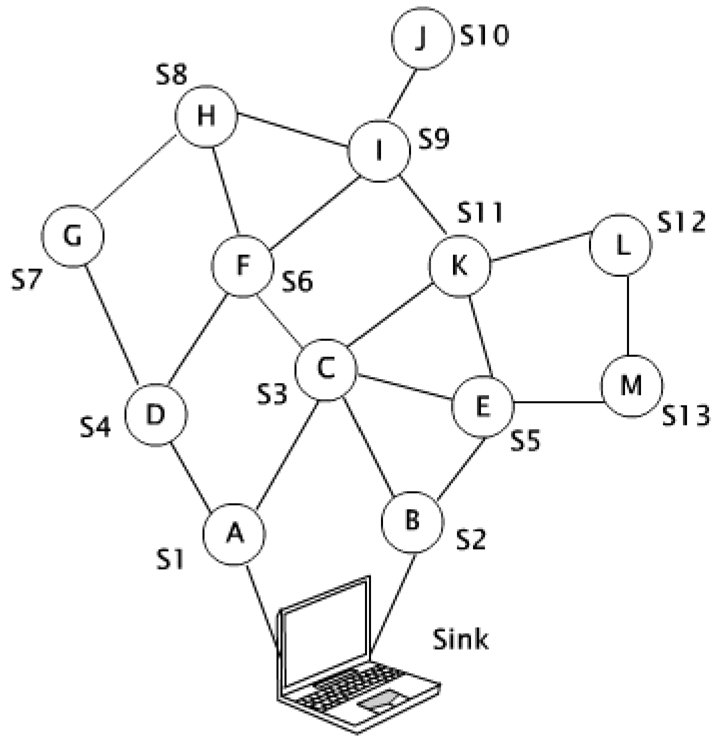


Figure 26: Illustrating The "Lonely Node Problem" in Sparse Networks.

In sparse networks, the neighbour count per node is small and the probability of the "Lonely Node Problem" occurrence becomes large. The "Lonely Node Problem" is, in fact, a direct measure of energy wastage in the network. If there exists no opportunity for a node to communicate at a certain time slot, there is little reason why that node should be awake at that time slot. Intuition reveals that the node should wake up at time slots that maximizes its communication opportunities with its neighbours within one time cycle, given any duty cycle. We illustrate further with an example.

Consider a sparse network of 13 nodes (labelled *A* to *M*) randomly deployed as in Figure 26 employing the (13,4,1) cyclic design of Figure 1, with each sensor having its own arbitrary wakeup schedule assignment (*S1* to *S13*). Lines between two nodes indicate bi-directional communication links. A mobile sink is connected to both nodes A and B. Notice that some of these sensors may have very few neighbour nodes, particularly those near the perimeter edges of the network, and those near communication obstacles. As an example, consider node *J* assigned with the schedule *S10*. The only neighbour of node *J* is node *I*. Since the time slot overlap between schedules *S10* and *S9* happens only at $t = 9$, the wakeup times of *S10* at $t = 1$, 7 and 10 offer no additional connectivity (or routing) benefit to the network. In fact, these represent energy inefficiencies in the CSBD and this is the main problem that we shall solve. Note that this problem is not unique to cyclic block designs. Even in random wakeup schemes having the same duty cycle, it is possible for sensors to wake up only to find that there are no other active neighbors to communicate with.

We propose our Adaptive Wakeup Schedule Function (AWSF) in 2 stages – WSF Pruning and WSF Reconstruction. The term “Adaptive” is used because the wakeup scheme “adapts” to sensor deployment topology.

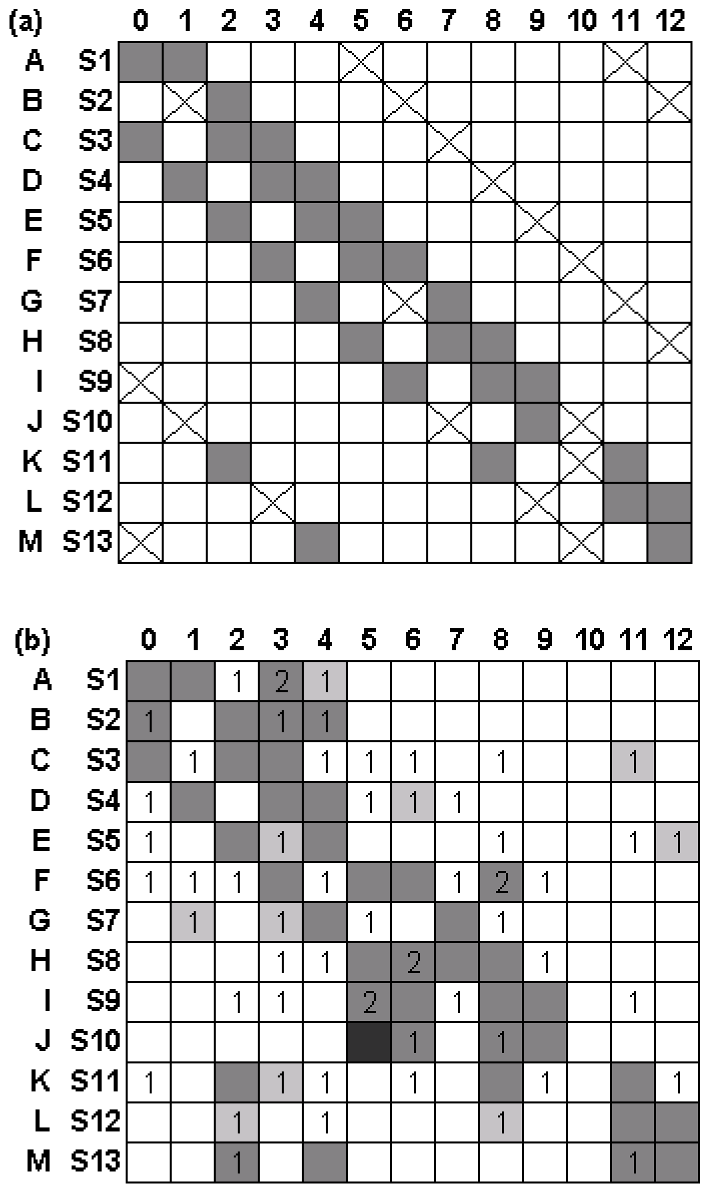


Figure 27: (a) AWSF pruning where crosses indicate active time slots that are pruned off; and (b) BRS scheme for a cyclic symmetric (13,4,1)-design. Integers indicate the number of active slot overlaps with neighbour nodes, or equivalently reassignment priorities. Light-gray boxes refer to randomly reassigned slots amongst slots with the same non-zero priority number in the schedule, dark-gray boxes refer to either original active slots or reassigned slots with unique priority numbers in the schedule, and black boxes refer to slots reassigned based on a special case.

<u>Stage 1: WSF Pruning</u>

After deployment and the initial neighbour discovery phase, every node will have knowledge about their neighbours. [27] also suggests that nodes discover their neighbours' WSF offset relative to its own. The problem discussed in the previous section can then be solved by putting node *J* to sleep at those known times when there are no active neighbours. The WSF of node *J* can therefore be pruned from $f(x)= x+x^7+x^9+x^{10}$ to $f(x)=x^9$, thereby potentially reducing energy consumption by 75% for node *J*. Therefore, Figure 27a. shows active slots in time slots t= 1, 7 and 10 for node *J* are removed (or pruned). Similarly for other nodes, active slots are pruned whenever that active slot does not coincide with another active slot in any of its immediate neighbours' schedules. Figure 27a. illustrates WSF pruning of the original 52 active slots to 30 active slots for all the nodes in the network. This solution is unique and depends only upon the network connectivity matrix. However, we would like to point out that the 75% energy savings of node *J* may not be valuable, for if node *I* exhausts its batteries much faster than node *J* due to the higher duty cycle of the former, network isolation does not come any later. Moreover, if the sink is mobile and connects itself to only node *J* instead, then WSF pruning has a negative effect on routing delays. With node *J* being the only gateway node to the rest of the sensor network, WSF pruning will also reduce the responsiveness of the network by approximately the same factor of four fold! Alternatively, instead of operating with the new WSF after the pruning stage, the "energy savings" can be put to better use to enhance the immediate connectivity of the network by reconstructing a new WSF. The mobile sink problem can also be solved at the same time.

Stage 2: WSF Reconstruction

Here, the spatial location of the sensor affects the WSF design. In the example of Figure 27a, the WSF pruning stage reduces the duty cycles of sensors without reducing the original network connectivity of the network. Let ps_i be the number of pruned off time slots for node i. In the reconstruction stage, each sensor attempts to introduce a maximum of ps_i new active time slots into its own WSF so that connectivity with its neighbours can be improved. In this way, duty cycles of every node do not exceed that of the original before pruning. Choosing time slots to be active can be done by examining the neighbours' WSF schedules after the pruning. The strategy is to choose a time slot that will maximize connectivity with all neighbour nodes. While many reconstruction schemes may be devised, we introduce a Basic Reconstruction Scheme (BRS).

Basic Reconstruction Scheme (BRS)

To illustrate this scheme, consider node *A* with schedule *S1*. During the pruning stage, node *A* frees up two active time slots for reconstruction. To avoid confusion later, we denote a free active slot as "Released Slot" (RS). From the connectivity diagram, node *A* is only connected to nodes *C* and *D*. If there exists a time slot in which both *C* and *D* are awake at the same time, that time slot would be the first to be made active by node *A*. The declaration of an initial sleep slot into an active slot is called "Free Slot Assignment" (FSA). Indeed, there are three possible combinations for node A, namely {*C*}, {*C, D*} and {*D*}. The second combination indicates maximum overlap with the neighbour set at time slots *t*=3. The algorithm therefore makes a FSA at *t* = 3 for node *A*. Since the initial number of RS is equal to 2, there remains one other possible assignment but with two possible time slots. The algorithm then randomly selects any one of the possible time slots (*t*=2 or *t*=4) with equal probability and makes a FSA. Figure 27a reveals all the possible time slots for all nodes. The integers represent the number of overlaps with neighbouring nodes in that time slot. In a sense, the integers also represent the priority for FSA for each node. Time slots with larger integers are always selected first for FSA provided there are enough RSs for the assignment. When there are insufficient RSs to cover all the possible time slots with equal priorities, the algorithm does a random selection, uniform distributed over the set of possible time slots.

A special case arises for node *J* since it has more RSs than possible slot assignment positions based solely on the pruned schedule of its only neighbour. In this case, the last FSA cannot be made until node *J* informs all its neighbours (in this case only to node *I*) of this case in an "Assignment Request" (AR) message, where the latter will choose a time slot that both nodes can stay awake at the same time. In this example, node *I* has already assigned all of its RSs before the AR message arrives and is able to inform node *J* to stay awake at *t*=5. In the event that node *I* also has more RSs than possible slot assignment positions, it defers the reply to node *J* until it receives a solution from one of its other neighbour nodes. This seldom happens as it indicates that the connectivity graph of nodes approximates a straight line deployment. Figure 27b illustrates one possible result after the reconstruction stage using BRS. Light gray boxes are those chosen at random by BRS. Black boxes are those selected on the basis of the special case of BRS. From Figure 27b, the duty cycles of all nodes remain the same as the original cyclic design of Figure 1. Yet, the "Lonely Node Problem" of the cyclic wakeup has been eliminated. The mobile sink problem is also solved in this new AWSF design.

5.4. Asynchronous Neighbour Discovery

As mentioned in previous chapters, it is not realistic to assume time synchronization in large-scale distributed sensor networks. Unfortunately, nodes in a distributed network require the discovery of neighbour nodes and knowledge about their neighbours' active time slots for bookkeeping and node failure inference. In both the Zheng's original work [27] (section 2.3) and RAW [12] BEACON messages are set up to achieve this without synchronizing clocks (section 2.4). We use a similar strategy for AWSF. During initial deployment, the schedules of our AWSF nodes use the original CSBD design. We set up our BEACONs for neighbour discovery and data transmissions in the same way as previously discussed for the CSBD system in section 2.3.

Our AWSF scheme undergoes WSF-Pruning and WSF-Reconstruction next. The schedules of all neighbours after these stages will remain connected by at least one time slot as long as the original communication graph is connected. However, for the worst case of two neighbours being connected by only one time slot, due to the pruning of the schedules, neighbour discovery become uni-directional in the presence of slot mis-alignments. Figure 28a illustrates this.

Schedules C and F only have one active slot overlap. The dark gray slots represent BEACONs, where nodes transmit at the beginning of each active slot (see section 2.4)

[27]. The horizontal axis is the time axis. Due to slot mis-alignment, when C sends the BEACON, F is unable to hear. F mistakenly concludes that C is no longer a neighbour. However, C is able to hear F's BEACON when sent. The C-F link becomes unidirectional. To solve this, we allow nodes to send out another BEACON just before it goes back into sleep mode. Figure 28(b) shows the implementation. Light gray slots represent additional BEACONs required. In this way, both C and F detect the presence of each other and the bidirectional property of the C-F link is restored. Once the reconstruction stage has completed, it is mandatory that neighbours exchange their schedules so that F is able to make inference about the time discrepancies of node C. Hence, F can send data to C as long as C is detected to be "alive" in the last cycle. AWSF is therefore an asynchronous wakeup scheme where no time synchronization is required for neighbour discovery and subsequently, for data transmissions.

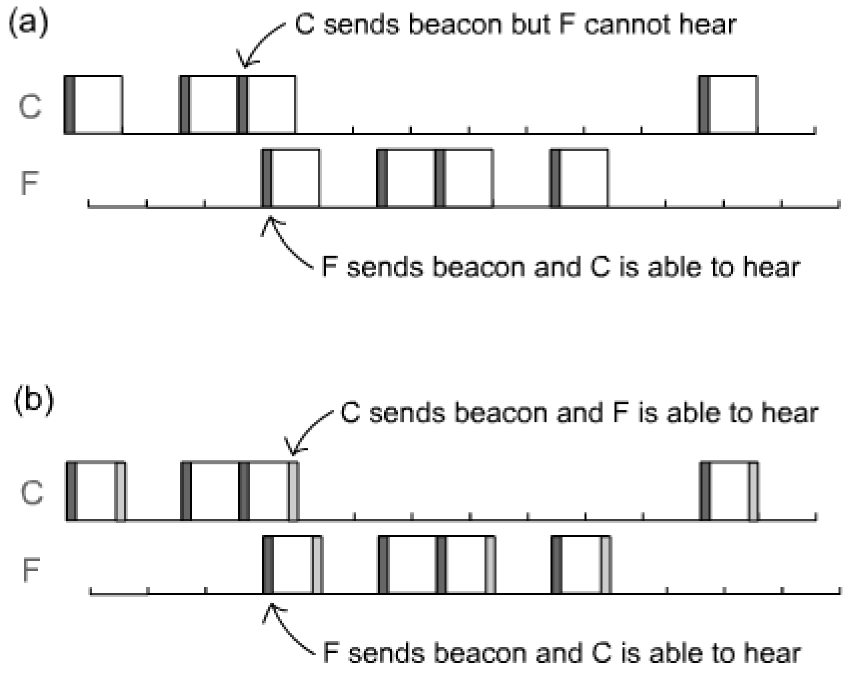

(a)
C sends beacon but F cannot hear
C
F
F sends beacon and C is able to hear
(b)
C sends beacon and F is able to hear
C
F
F sends beacon and C is able to hear

Figure 28: (a) Illustrating Slot Mis-alignment and loss of bidirectional C-F link for the worst case of one active slot overlap between schedules. (b) Illustrating Slot Mis-alignment and restoration of bidirectional C-F link for the worst case of one active slot overlap between schedules.

We emphasis that as our sensor network is assumed to be static with no mobile nodes. We could again employ On-Demand Neighbour Discovery (ODND) described in section

2.3. Finally, we would like to point out that although time slot misalignment will not affect neighbor discovery, it will have an effect on data throughput capacity if slot time, T_{slot}, is too small for data packets to complete their transmission within that slot. However, as we have discussed in section 2.3, we can use AREQ packets on a per-slot basis to allow for the completion of data transmissions across a slot boundary.

5.5. Data Transmission and AWSF Maintenance

Nodes transmit data to their neighbours as long as they hear BEACONs from them in the last T_{lost} time units. Due to communication imperfections such as fading, nodes may temporarily be unable to hear from their neighbours. However, if no BEACONs are received from them for more than T_{lost} cycles, they are believed to be "lost" (e.g. run out of battery life or malfunctioned). AWSF Maintenance is then activated (see below). We also adopt, as in [27], the option for sender nodes to request their neighbours to stay awake for the next time slot (assuming they are scheduled to go into sleep mode next) to adapt to high traffic loads. Essentially, the sender node signals to the neighbour receiver node using packets to stay awake in the next slot to prepare for more data. The receiver node may reject the request if its own battery life is limited. Further, such requests have to be made on a per-slot basis for power control and management purposes.

In real world deployment, nodes may malfunction or die with time. Consider the AWSF-BRS solution of Figure 27. When node *G* has exhausted its batteries, node *H* finds itself waking up to find no communicable neighbours in time slot t=7. It is important to note that this problem is not unique to our AWSF scheme. The original CSBD design has this exact same problem by examining Figure 1 and Figure 26. Random wakeup schemes such as RAW [12] is already prone to this problem from the very statistical nature of its solution. By the use of BEACONs previously described, nodes can detect neighbours that are no longer reachable. AWSF maintenance is then activated per node and the usual

WSF pruning and reconstruction stages are applied again. After maintenance, nodes exchange their WSF schedules with their immediate neighbours.

5.6. Network Connectivity and Sensing Coverage

It is not difficult to see that with our AWSF solving the “Lonely Node Problem”, network connectivity is improved over the cycle schedule. If we define the *Real Degree* of a node *X*, *Deg(X)*, as the ratio of the sum of neighbours, N_S, detected in all active time slots of *X* over one cycle to the total number of active time slots in one cycle, the improvements in *Deg(X)*, $\forall X \in \{\cup Sx\}$ for AWSF over the original CSBD can be very substantial. For our illustrative example of Figure 27b, improvements range from 220% to 400% for different nodes. If we define the *Real Network Degree*, *DegNet(N)*, of a network, *N*, as the average of all Real Node Degrees of all nodes in the network, then it can be computed that *DegNet(AWSF)* = 1.827 and *DegNet(CSBD)* = 0.731, an improvement of 250%. Indeed, AWSF achieves about 62.5% of the maximum possible network connectivity, but with a duty cycle of about only 30.77%. On the other hand, CSBD achieves only 25% of the maximum connectivity with the same duty cycle. Figure 29 illustrates this improvement graphically. Note the peculiar case of the graphs for *t*=10. For the CSBD, the *node degree* for the active nodes is zero at time slot *t*=10. No real network connectivity is achieved by switching on the nodes. In the AWSF case, these nodes are put to sleep instead, and with their released active slots transferred to other time slots.

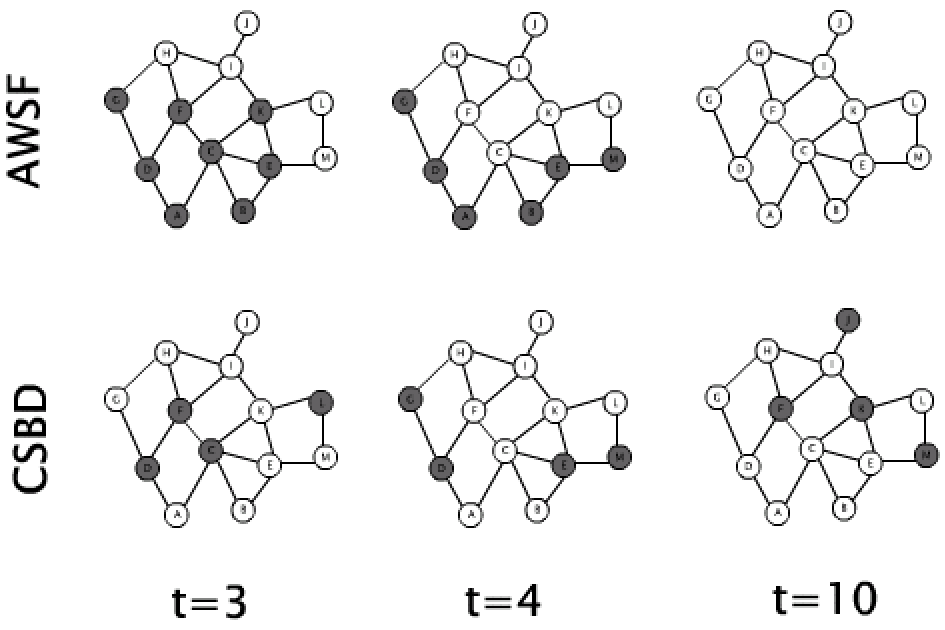


Figure 29: Comparing network connectivity for AWSF-BRS and CSBD at time snapshots *t*=3, 4 and 10. Awake nodes are coloured grey and Sleep nodes are coloured white. Both schemes have the same duty cycle over one cycle. All nodes in AWSF always find a neighbour to communicate when in wakeup mode.

To further illustrate the impact of improved network connectivity of AWSF on routing delays, we consider a simple example. Suppose at time slot *t*=6, node *I* has data for the sink (Figure 26). Suppose the route has been determined (by the routing algorithm, say shortest path) to be $I \to F \to C \to A$. Using AWSF, this data arrives at the sink after $8T_{slot}$. However, using CSBD, this data can only reach the sink after $20T_{slot}$! The bulk of the cyclic WSF delays are the sleep delays, which our AWSF solution minimized. The superiority of our AWSF solution compared to CSBD is very encouraging, as we have not yet considered routing optimizations. When more intelligent routing schemes are incorporated, such as routing based on forwarding sets [12], further improvements can be anticipated.

In section 3.1.1, we provided an analysis on sensing coverage for CSBD systems. Here, AWSF is no longer cyclic-symmetric (although its adaptation was initially based on a CSBD). However, sensing coverage remains at its worse bounded in time by $(k^2+1)T_{slot}$. using the same definitions as in *Theorem 3.1.1.1,* we state:

Theorem 5.6.1. Region A is $\left(\beta,(k^2+1)T_{slot}\right)$*-covered for AWSF networks.*

Proof: In AWSF, schedules are no longer cyclic-symmetric. However, each schedule still contains (k+1) awake slots and individual schedules still repeat themselves after one cycle time. All sensor nodes are therefore awake at least once within a time duration of $T_\beta = (k^2+k+1-k-1+1)T_{slot} = (k^2+1)T_{slot}$. By a similar line of argument as in *Theorem 3.1.1.1*, region A is therefore $\left(\beta,(k^2+1)T_{slot}\right)$-covered. ■

5.7. Other Properties of AWSF

Property 5.7.1. Nodes operating AWSF always wake up to find at least one neighbour to communicate.

Proof: Consider the original communication graph, *G(V,E)*, of nodes in set V and links in set E. Let $V_i \subseteq V$ be the subset of all nodes that are in active mode in time slot i and $E_i \subseteq E$ be the set of all links that connect two nodes of V_i in time slot i in a cyclic symmetric (k^2+k+1,k+1,1) design. By definition, V_i always has (k+1) members (that is, the cardinality of V_i is k+1) and note that E_i can be an empty set. $G_i(V_i, E_i)$ is therefore the graph of active nodes at time slot i. Pruning considers the partition of the set V_i as follows. Let $V_{pi} \subseteq V_i$ be the subset of all nodes in V_i that are connected by at least one of the links in E_i. V_{pi}', the complement set of V_{pi}, is therefore the subset of all nodes in V_i that are not connected by any links found in E_i. Pruning removes the set V_{pi}' from the graph G_i. Hence, the pruned graph of active nodes at time slot i can be denoted by $G_{pi}(V_{pi},E_i)$. If $E_i = \{\emptyset\}$, then $V_{pi} = \{\emptyset\}$ and G_{pi} is an empty graph. If $E_i \neq \{\emptyset\}$, by definition of the partitioned set V_{pi}, all nodes in G_{pi} wakes up to find at least one neighbour to communicate.

Let the set of nodes $V_{ri} \subset V$ and the set of links $E_{ri} \subset E$ be added to G_{pi} during reconstruction. We denote this new graph after reconstruction as G_{ri}. By definition of reconstruction, $V_{ri} \cap V_{pi} = \{\varnothing\}$, $E_{ri} \cap E_i = \{\varnothing\}$, cardinality of set V_{ri} is less than or equal to the cardinality of set V_i, and every link in E_{ri} always connects one node from the set V_{pi} and one node in V_{ri}. If $V_{pi} = \{\varnothing\}$, then $V_{ri} = \{\varnothing\}$ and G_{ri} is an empty graph. This property is not concerned with this case. If $V_{pi} \neq \{\varnothing\}$, every node in the set V_{ri} has a neighbour in the set V_{pi}. This implies that every node in the graph $G_{ri}(V_{ri} \cup V_{pi}, E_{ri} \cup E_{pi})$ also has a neighbour at every time slot i. Nodes in the AWSF always wake up to find at least one neighbour to communicate. ■

Property 5.7.2. AWSF is delay upper-bounded.

Assuming the original network is a connected network and in the worst case, any two neighbouring schedules will have at least one overlapping active slot within the cycle time after the pruning stage. During reconstruction stage, even if no new active slots are added with respect to any neighbour, the delay is bounded within T_{cycle}. We assume that data packets can be delivered with T_{slot} and that there is clock synchronization mismatches are negligible.

5.8. An Application of AWSF

It turns out that our algorithm AWSF fits well to an application for offshore underwater oil exploration. Oil exploration and trading companies may have potential interests in such technologies that will save costs in the long run. Currently, in order to discover new underwater oil wells, an array of sensors are towed from behind a ship to collect data about the seafloor and analyzed at a later stage. This array of sensors can potentially span a geographical area larger than the ship itself and every such operation is costly. We propose a Underwater Wireless Sensor Network (UWSN) architecture with static sensor nodes for underwater event monitoring and optional patrolling Autonomous Underwater Vehicles (AUVs) as mobile users accessing information from the UWSN anywhere, anytime. Static nodes are capable of communicating with each other wirelessly over an acoustic channel and also with the AUVs. The UWSN may also be wirelessly linked up to an offshore platform for data analysis without AUVs. Figure 30 shows such an architecture for an example underwater seismic monitoring application for undersea oil field detection.

The nature of oil discovery is that whenever an area has been surveyed to be negative for the possibility of oil deposits, it no longer requires any surveillance for perhaps, a very long time, in the order of months to years. This means that sensors can be switched off to a “hibernation” mode for a very long period of time (hibernation period) and switched back to the “operational” mode after that. Time synchronization amongst the sensors on their next wakeup cycle becomes a very fatal problem after years of node isolation rendering many existing schemes not practical. AWSF becomes a natural solution for it is not only a time asynchronous scheme during its operational cycle, it is also locally optimally connected. It works very well in a sparsely connected network scenario such as in this underwater application. Since AWSF is derived from a CSBD, AWSF wakeup schedules can be changed back to a CSBD before any hibernation periods so that even if sensors have slightly changed positions due to water currents or when clock drifts have become extremely severe in their next scheduled operational cycle, they are still able to discover each other and re-initiate the AWSF algorithm again.

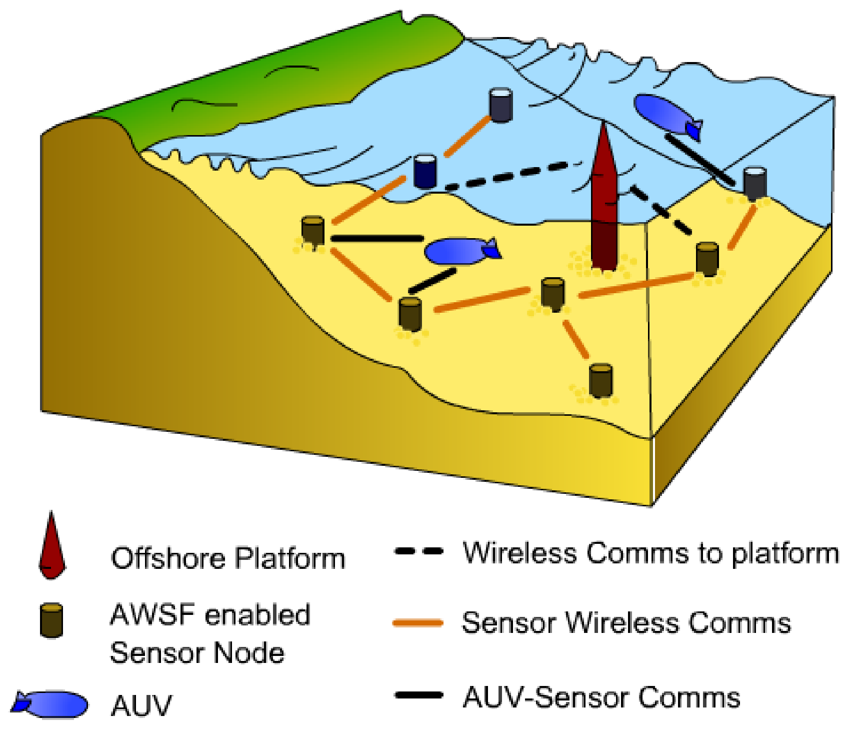


Figure 30: Potential UWSN for seismic imaging of underwater oil fields

The CSBD, originally proposed for mobile nodes is disadvantaged by the "Lonely Node Problem". In UWSNs, sensor nodes are usually deployed stationary. The RAW scheme [12] cannot guarantee delay bounds and is similarly prone to the "Lonely Node Problem". Moreover, such schemes rely upon a dense sensor deployment strategy, often a luxury in any UWSN. In contrast, our AWSF proposal adapts to deployment topography in a completely distributed fashion to support fast routing in UWSNs. The network architecture designer first plans for a desired duty cycle for the underwater nodes based on the desired lifetime of the UWSN. Each node then adaptively performs the AWSF algorithm to compute their respective "Sleep" times based on immediate neighbour information without the need for location information. We have shown (section 5.4) that our AWSF solution is time-asynchronous and bounded in delays. We show later that AWSF has better average delays and smaller delay variances compared to competitor schemes. AWSF eliminates the "Lonely Node Problem" where nodes wakeup to find no communicable neighbours, thereby substantially reducing energy wastage. Further, the sparser the deployment density, the better AWSF becomes in terms of improving network connectivity compared to CSBD. Since AWSF assumes complete shutdown of its communication module in the “Sleep” mode, it is largely different from wakeup schemes that merely put nodes to the “Idle” mode for “data snooping”. Moreover, AWSF does not assume any dual communication channels and is simple to implement. All these features of AWSF are ideal for this proposed underwater oil field discovery application.

5.8.1 Simulation Results

Simulation Setup

For our simulations, we consider the scenario as in Figure 30 with an underwater grid of 100 nodes (10 x 10). We choose the Geographic Routing (GR) scheme [71] for illustration. The first 73 primary nodes employ a standard (73,9,1) schedule; with the remaining 27 secondary nodes redeploying any schedules from the primary nodes. The choice of $k = 8$ is used so that $k^2 + k + 1 = 73$ is the largest integer smaller than 100. This allows the network to have maximum schedule diversity and at the same time, the entire design set of schedules are used at least once in the network. The sensor network is assumed to be deployed in a flat topography and covering an area of 100 x 100 units^2 with an approximate 10 units separation between nodes. Node density is therefore about 0.14/units^2. Communication range of nodes is taken to be about 15 units, and transmission rate is 300bps. Arbitrary mobile users (such as an Underwater Autonomous Vehicle or AUV) may query for information about any location from any arbitrary location in the sensor network. Query arrivals are assumed to be independent and Poisson distributed. The simulation ceases when the count of total queries reaches 10,000.

Other than our proposed AWSF-BRS scheme (section 5.3), we make comparisons with other schemes including RAW [12] and that a regular CSBD design.

Delay Simulation Studies

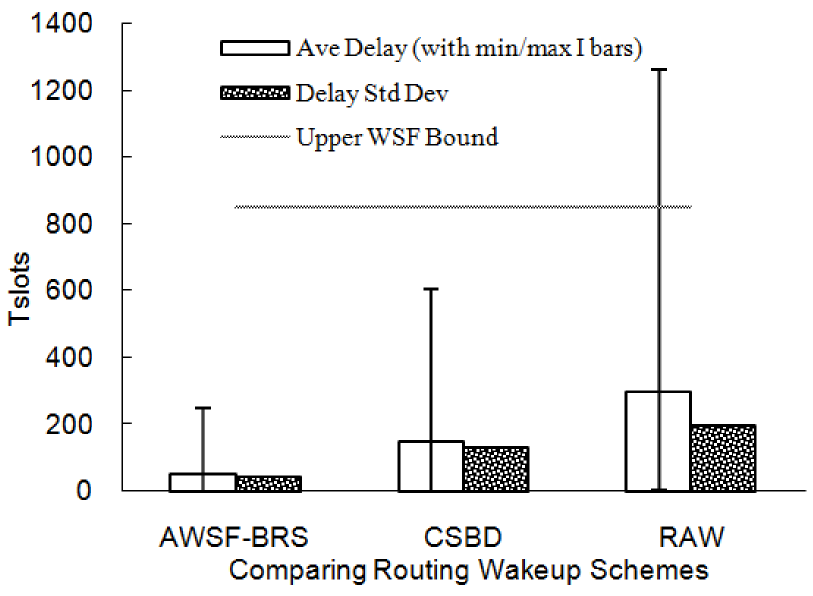


Figure 31: Comparing the delay results across different routing wakeup schemes. The delay bound is a loose bound assuming the worst-case possible number of hops in the network. All schemes use the same node duty cycle of about 12.4%.

Figure 31 compares AWSF-BRS, CSBD and the RAW. AWSF-BRS has the lowest average delays, delay deviations and lowest maximum delays. CSBD is not as efficient as AWSF because of the "Lonely Node" problem. RAW fails in comparison to even CSBD because the latter has at least the guarantee that any two neighbours will wake up in the same time slot within one cycle. There is no such guarantee in RAW. With a 12.4% duty cycle, packets have to wait on average of $65T_{slot}$ before a one-hop progress is made. With an average hop count from an arbitrary source to an arbitrary sink to be about 5, the average delay is indeed about $5 \times 65 = 325T_{slot}$. Clever routing techniques [12, 72] (other than simple GR) may also be applied with AWSF or CSBD so that delays can be further reduced or other tradeoffs could be made. In our simulations, the maximum delay of RAW is also found to be exceedingly high owing to its statistical nature with no bounded delay guarantees. In fact, both the AWSF and CSBD solutions are within the worst-case delay bound, while the RAW solution can exceed this bound. The worst-case delay bound is a loose bound computed based on the maximum possible number of hops possible in the network using the GR protocol.

Energy and Overhead Simulation Studies

AWSF wakeup scheme incurs additional energy to send BEACONs in order to perform proper neighbour discovery. However, we have also previously discussed in section 2.3 that BEACON energy can be largely saved by ODND for static sensor networks. With ODND, we use the condition that neighbour discovery is only necessary after every $100T_{cycle}$ (approximately every 12 minutes). In this set of simulations, we assume that the transmission power is 10W, the sensing power is 9W, the power in the active mode of 3W and finally the power consumed in the sleep mode is negligible. T_{slot} is set to 5s and BEACONs are transmitted over 2% of T_{slot}. AWSF without ODND consumes about 33.9kJ of beaconing power (Figure 32a). In comparison, CSBD (also without the ODND option) consumes only 26.2kJ in beaconing. However, with ODND, AWSF can perform significantly better and consumes only 10.2kJ. For RAW, we have assumed that no beaconing is required. While CSBD expends less beaconing power than AWSF in general, it utilizes its wakeup time slots much less efficiently. The power wastage for CSBD amounts to 32.9kJ and that of RAW is 50.1kJ. Our AWSF proposal does not incur any wastage. AWSF therefore remains the most energy efficient when both the beaconing and "lonely node" wastage factors are taken into consideration. However, AWSF incurs more one-time setup costs than the other wake-up schemes (Figure 32b). AWSF initialization time is about four times that of CSBD; and consumes three times more energy in initial schedule exchanges. The long initialization time is a result of the additional pruning and reconstruction phases of AWSF. After each phase, schedules need to be exchanged between neighbours for bookkeeping, thereby incurring additional energy. RAW is assumed to have zero initialization time once deployed and assumes no requirement of schedule exchanges.

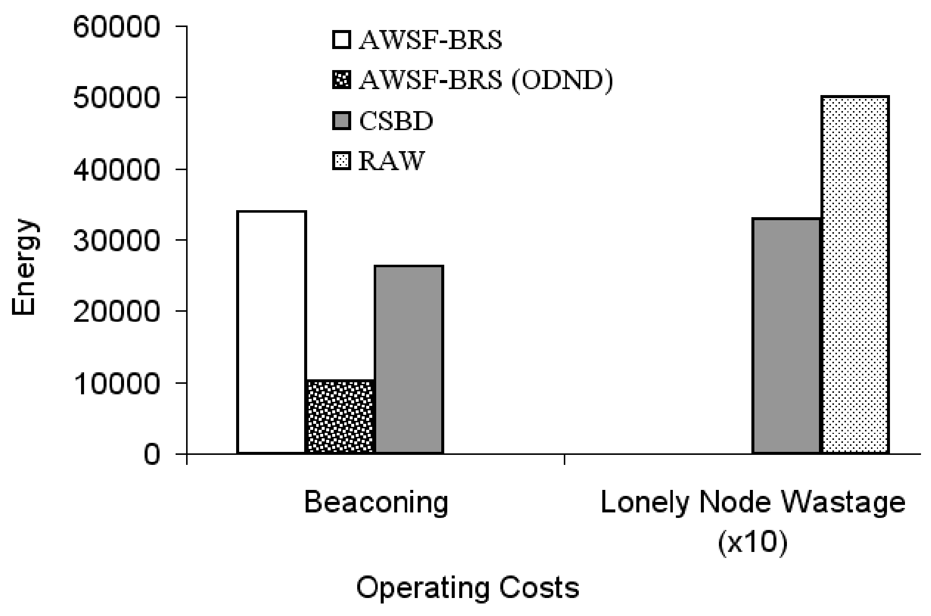


Figure 32a: Comparing energy overheads for different routing wakeup schemes.

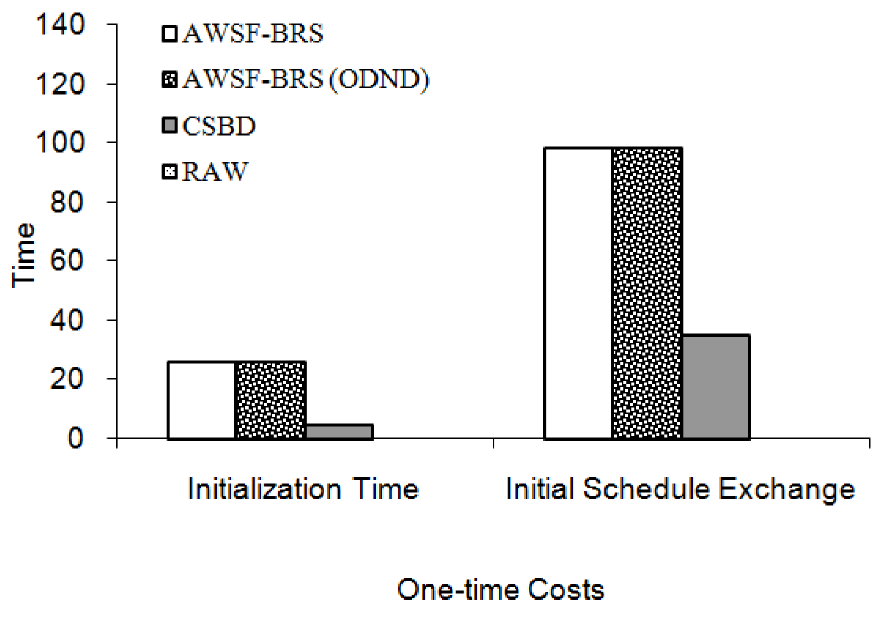


Figure 33b: Comparing delay for different routing wakeup schemes.

Network Lifetime Simulation Studies

We further compare network lifetimes for our proposed AWSF with CSBD, PEAS [20] and PECAS [21]. Since nodes in PEAS will continue to stay awake until they are completely depleted of energy, there is no balanced distribution of energy consumption among sensors in a region. Therefore, if we define [73, 74] the lifetime of a sensor network as the duration of time until the first sensor node fails due to energy depletion, the lifetime of a PEAS network is theoretically far less than that of our AWSF proposal, by assuming that the traffic and event generation pattern is also approximately randomly distributed. This idea is confirmed by Table 7 that summarizes the lifetime of CSBD (without ODND), AWSF (without ODND), PEAS and PECAS.

Table 7: Comparing Network Lifetimes (in default units)

	CSBD	AWSF	PEAS	PECAS
Lifetime	1668.03	1572.69	715.99	1220.89

It reveals that the average lifetime of an AWSF network is 2.33 times longer than that of a PEAS network. Active PECAS nodes go back to sleep after a fixed time interval. Depending on its value, the lifetime of the network may differ. In Table 7, we report the best average lifetime that is obtained for different parameter values. This table shows that the average lifetime of a network of nodes running AWSF is about 37% longer than a network of nodes running PECAS with a sleeping period of 10 time units (equivalent to about 122 periodic checks to its average lifetime). Moreover, if the periodic check in PECAS is reduced to one check every 500 time units, the improvement of AWSF over PECAS is significantly more obvious (68%). The lifetime variance of PECAS is also very large with occasional very long lifetimes, depending on the actual topology. Note that energy savings in PECAS is actually smaller than PEAS because of the periodic message exchanges in PECAS: it extends the overall network lifetime by balancing energy expenditure in the network. CSBD has a longer lifetime than AWSF because of the additional beaconing energies and overheads incurred by the latter.

An additional point to note is that these lifetime values for CSBD and AWSF are reported without ODND. With ODND, less beaconing energies are expensed and their lifetimes are expected to be even longer.

5.9. Summary

Where sensors can be deployed relatively dense, we have previously proposed our CSBD solution for agent-based sensor networks (BTC sensor nodes), and proposed the same CSBD solution for query-based sensor networks (DWSF sensor nodes). However, we have described why, in some scenarios, a dense deployment of sensor nodes is not possible. We described limitations of existing wakeup methods in such scenarios and proposed our solution – the Adaptive Wakeup Schedule Function (AWSF). Unlike all our previous solutions, AWSF is not cyclic-symmetric, but remains time-asynchronous in neighbour discovery and is a deterministic wakeup scheme adapted from the original CSBD design. We have analysed network connectivity and sensing coverage issues in AWSF-based networks, and provided simulation results to support our proposal.

We compared our scheme with PEAS, PECAS, RAW and regular CSBD. AWSF incurs smaller average delays and delay variances. Although it incurs more one-time setup overhead costs in terms of longer initialization times and more initial schedule exchange energies, these are comparatively negligible compared to the "Lonely Node" energy savings and improved beaconing technique using ODND.

Chapter 6: Conclusion and Future Work

6.1. Conclusion

We have provided a comprehensive and detailed analysis of one class of deterministic wakeup schemes based on Combinatorics, for the purpose of energy conservation in battery-powered wireless sensor networks. Although these cyclic symmetric block design (CSBD) wakeup schedules are time slotted, it has been shown that they can work even without clock synchronization amongst sensors, by employing BEACON messages. For static sensor networks, we proposed “On-Demand Neighbour Discovery” (ODND) that promises to reduce communication overheads from BEACON control messages significantly. Moreover, CSBD ensures that data propagation times are always bounded at worst by almost one cycle time for every network hop.

We further show that CSBD schedules preserves sensing coverage in finite time and preserves network connectivity within bounded time. We justified that these requirements are sufficient from the perspective of applications. These wakeup designs can then be applied to agent-based sensor networks where an agent actively follows the event or target in realtime for reasons that are meaningful to the application. We illustrated with a target-tracking example and developed an adapted algorithm (Tracking Wakeup Schedule Function – TWSF) for tracking in the absence of prior knowledge of the target maximum speed. TWSF also allows a network to be reconfigurable (in terms of its wakeup schedules) for different application requirements.

We also show that CSBD schedules can also be used to synchronize wakeup times for data collection. The distributed database that is created in this way can guarantee bounded query delay times for different classes of users. In a special case, this delay bound is theoretically zero. We encouraged the use of query-based CSBD for specialized data collection to complement agent-based sensor networks. We demonstrated this possibility with a two-tier architecture using a target tracking and target identification application as an example.

Both agent-based and query-based sensor networks require a relatively dense deployment of sensors. For ad-hoc and sparse networks, we introduced an Adaptive Wakeup Schedule Function (AWSF) that works to optimize local network connectivity and eliminates the “Lonely Node Problem”, where nodes wakeup to find no other neighbour nodes in its communication range. Sparse networks are often deployed because of cost concerns and physical limitations in the deployment scenario.

Simulations are used to support our claims in agent-based sensor networks, query-based sensor networks and ad-hoc & sparse sensor networks. In some cases, we implemented CSBD-based schemes to real Crossbow sensor motes where real experimental measurements are reported.

Finally, CSBD-based schemes are simple and cost-effective to implement with little overheads and implementation costs, owing to their deterministic nature. CSBD-based schemes do not require dual-radio channels, do not require costly time-synchronization across the network and have low computational complexities and low communication overheads. CSBD schemes are distributed and they are easily adaptable to different application scenarios.

6.2. Future Work

The scope for the application of CSBD to sensor networks is huge with many possible future research directions and areas of focus. In this section, we briefly discuss some possibilities.

For query-based sensor networks, we have focused primarily on the minimal delay solution in replying queries. However, we have also showed the existence of another solution – the minimal energy solution. The impact of this latter solution on application requirements is not investigated. Since the minimum delay solution is, in general, different from the minimum energy solution, it would be particularly interesting to derive the set of all possible query response combinations where both maximum delay and maximum energy are minimized simultaneously, subject to certain application constraints. This set of responses may not yield a delay that is as small as the minimum delay solution or energy as small as the minimum energy solution, but is expected to provide solution(s) that satisfies the constraints, if it exists. Further, we have focused on query lengths that are within two slot times, which is why the minimum energy solution can always be obtained from a single schedule, and thus unique. The uniqueness and the number of such minimum energy solutions in existence for longer query lengths for a given CSBD are however, not known. There has not been any analytical or comprehensive study of query lengths that exceed two slot times. This is not an easy problem to solve given the growing number of possibilities in the way queries are replied for longer query lengths. We have only provided an exhaustive search result of the query delay and energy graphs with respect to different query lengths for a small (7,3,1) system in Figure 20. This problem is clearly more difficult if the system is much larger with a larger *k* value.

In our work, we have shown how CSBD may be adapted for target tracking using Tracking Wakeup Schedule Function (TWSF) and Adapted Wakeup Schedule Function (AWSF). In TWSF, new active slot assignments are made to the original CSBD schedules so that real-time applications, such as target tracking, are possible with the use of an agent. In AWSF, existing active slots in the CSBD are rearranged according to neighborhood information to locally minimize energy consumption in the network. There are indeed many other different ways in which CSBD may be adapted for other purposes. Like TWSF and AWSF, some of these adaptations may continue to inherit desirable properties of the original CSBD. One such possibility is in the area of data aggregation, by assuming that the network "learns" sources and sinks by observing traffic flows. As data flows become predictable, its underlying wakeup scheme may also be adapted in such a way to minimize delay while data is aggregated along the path to its destination sink with nodes using a known duty cycle schedule. Such an approach to network data aggregation, where not all network nodes are awake at the same time, is expected to continue to have bounded-time delays, sensing coverage and network connectivity while controlling energy expenditure on a per-node basis through its CSBD-based schedule.

While research in CSBD for sensor networks continues to be important, it is equally crucial to deploy this system to the real world for testing and validation. We have had preliminary implementation experience with Crossbow motes. It is our immediate future work to realize the deployment of CSBD-based wakeup schemes for sensor networks in real application scenarios. One such scenario is in the monitoring of structural beams for construction work where costly cabled-monitoring systems and manual monitoring are still in use today. With CSBD, this potentially saves construction companies high recurring operating costs and reduces errors and time wastage when cables are cut or when readings are erred by human monitoring.

References

[1] Crossbow Technology Inc. [Online]. Available: http://www.xbow.com/Products/wproductsoverview.aspx, 9 May 2009.

[2] N. B. Priyantha, A. Chakraborty and H. Balakrishnan, "The Cricket Location-Support system", *In Proc. of the 6th ACM MOBICOM*, August 2000, Boston, MA, USA.

[3] Sensoria Corporation [Online]. Available: http://www.sensoria.com/, 9 May 2009.

[4] M. Leach and D. Benyon, "Speckled Computing: A new challenge for Human-Computer interaction", *In Proc. of HCI 2005*, Vol. 2, September 2005, Edinburgh, UK.

[5] K. J. Wong and D. K. Arvind, "Specknets: New Challenges for Wireless Communication Protocols", *In Proc. of The IEEE Intl. Conf. on Information Technology and Applications*, Vol 2, pp 728-733, July 2005, Australia.

[6] L. Ho, M. Moh, Z. Walker, T. Hamada and C.-F. Su, "A Prototype on RFID and Sensor Networks for Elderly Healthcare: Progress Report", *In Proc. of the ACM Special Interest Group on Data Communication (SIGCOMM) Workshops*, pp. 70 - 75, August 2005.

[7] M. John, R. F. Thaddeus, Fulford-Jones, P. Bonato and M. Welsh, "A Wireless, Low-Power Motion Analysis Sensor for Stroke Patient Rehabilitation", Abstract 143281, *Biomedical Engineering Society (BMES) 2005 Annual Fall Meeting*, September 28-October 1, 2005, Baltimore, MD, USA.

[8] S. Baird, S. Dawson-Haggerty, D. Myung, M. Gaynor, M. Welsh, and S. Moulton, "Communicating Data from Wireless Sensor Networks using the HL7v3 Standard", *In Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks (BSN)*, April 2006.

[9] M. Duarte and Y.-H. Hu, “Vehicle Classification in Distributed Sensor Networks”, *Journal of Parallel and Distributed Computing*, Vol. 64 No. 7, pp. 826-838, 2004.

[10] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler and J. Anderson, “Wireless Sensor Networks for Habitat Monitoring”, *In Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, September 2002, Atlanta, Georgia, USA.

[11] Exploratory Workshop on Sensor Based Infrastructure for Early Tsunami Detection. [Online]. Available: http://www.cs.pitt.edu/s-citi/tsunami/workshop1/index.htm, 9 May 2009.

[12] V. Paruchuri, S. Basavaraju, A. Durresi, R. Kannan and S. S. Iyengar, “Random Asynchronous Wakeup Protocol for Sensor Networks”, *In Proc. of the 1st International Conference on Broadband Networks (BROADNETS)*, October 2004, California, USA.

[13] S. Kumar, T. H. Lai, and J. Balogh, “On k-coverage in a mostly sleeping sensor network”, *In Proc. of the 10th Annual International Conference on Mobile Computing and Networking (Mobicom)*, pp 144-158, September 26 – October 1, 2004, Philadelphia, USA.

[13] S. Guha and S. Khuller, “Approximation algorithms for connected dominating sets”, *Algorithmica*, Vol. 20, No. 4, 374-387, 1998.

[14] L. Bao and J. J. Garcia-Luna-Aceves J. J., “Topology Management in Ad Hoc Networks*”, In Proc. of the Fourth ACM International Symposium on Mobile ad hoc networking & computing*, pp.129 – 140, 2003, Annapolis, Maryland, USA.

[15] B. Chen, K. Jamieson, H. Balakrishnan and R. Morris, “Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks”, *ACM Wireless Networks Journal*, Vol. 8, No. 5, September 2002.

[16] J. Wu and H. Li, “On Calculating Connected Dominating Sets for Efficient Routing in Ad Hoc Wireless Networks”, *In Proc. of the 3rd ACM International Workshop Discrete Algorithms and Methods for Mobile Computing and Communications*, pp.7-14, August 1999, Seattle, Washington, USA.

[17] C. Schurgers, V. Tsiatsis and M. Srivastava, “STEM: Topology management for energy efficient sensor networks”, *IEEE Aerospace Conference*, Vol. 3, pp.1099-1108, 2002, Los Angeles, California, USA.

[18] CrossBow Mica2 data sheet. [Online]. Available: http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf, 9 May 2009.

[19] C. S. Raghavendra and S. Singh, “PAMAS – Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks,” *ACM Computer Communications Review*, 1999.

[20] F. Ye, G. Zhong, J. Cheng, S. Lu, and L. Zhang, “Peas: A robust energy conserving protocol for long-lived sensor networks”, *In Proc. of the 23rd International Conference on Distributed Computing Systems (ICDCS)*, pp 28–37, May 2003, Rhode Island, USA.

[21] C.Gui and P.Mohapatra, “Power conservation and quality of surveillance in target tracking sensor networks*”, In Proc. of the 10th Annual International Conference on Mobile Computing and Networking (Mobicom)*, pp129–143, September 26 – October 1, 2004, Philadelphia, USA.

[22] G. L. Xing, X. R. Wang, Y. F. Zhang, C. Y. Lu, et al., “Integrated Coverage and Connectivity Configuration for Energy Conservation in Sensor Networks”, *ACM Transactions on Sensor Networks*, Vol. 1, No. 1, pp.36-72, 2005.

[23] L. Wang and Y. Xiao, “A Survey of Energy Saving Mechanisms in Sensor Networks”, *In Proc. of the 2nd Intl. Conf. on Broadband Networks (BROADNETS)*, pp.724-732, October 2005, Boston, Massachusetts, USA.

[24] A. Cerpa and D. Estrin, “Ascent: Adaptive Self-Configuring sEnsor Networks Topologies”, *IEEE Trans. on Mobile Computing*, Vol. 3, No. 3, pp.272-285, July 2004.

[25] Anderson, “Combinatorial designs and tournaments”, Oxford University Press, 1998.

[26] E. Kramer and S. Magliveras, Finite Geometries and Combinatorial Designs, *American Mathematical Society*, 1992.

[27] R. Zheng, J. C. Hou and L. Sha, “Asynchronous Wakeup for Ad Hoc Networks”, *In Proc. of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, June 1-3, 2003, Annapolis, MD, USA.

[28] E. Kutanoglu, S.D. Wu, “On Combinatorial Auction and Lagrangean Relaxation for Distributed Resource Scheduling”, *IIE Transactions*, Vol. 31, No. 9, pp813-826, 1999.

[29] L. Brankovic and M. Miller, “An application of combinatorics to the security of statistical databases”, *Austral. Math. Soc. Gazette 22,* pp173-177, 1995.

[30] D.J.A. Welsh (Ed.), “Combinatorial Mathematics and Its Applications”, New York: Academic Press, 1971.

[31] M. Bennett, “Affine and Projective Geometry”, Wiley-Interscience Publishing, 1995.

[33] Y. F. Wong, J. K. Wu, L. H. Ngoh, W. C. Wong. “Collaborative Data Fusion Tracking in Sensor Networks using Monte Carlo Methods,” *In proc. of the 29th Annual IEEE International Conference on Local Computer Networks– 1st IEEE*

Workshop on Embedded Networked Sensors (LCN 2004 – EMNETS-I), pp. 563-564, November 16-18, 2004, Tampa, FL, USA.

[34] H. Qi, W. Xu and X. Wang, "Mobile-Agent-Based Collaborative Signal and Information Processing in Sensor Networks", *Proceedings of the IEEE*, Vol. 91, No. 8, pp. 1172-1183, August 2003.

[35] H. Qi, S. S. Iyengar, and K. Chakrabarty, "Multi-resolution Data Integration using Mobile Agents in Distributed Sensor Networks", *IEEE Trans. Systems, Man, Cybernetics*, Vol. 31, No. 3, pp. 383-391, August 2001.

[36] L. Gan, J. Liu and X. Jin, "Agent-Based, Energy Efficient Routing in Sensor Networks", *In Proc of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Vol. 1, pp.472 – 479, 2004, New York, USA.

[37] L.-K. Soh and C. Tsatsoulis, "Reflective Negotiating Agents for Real-Time Multisensor Target Tracking", *Int. J. Conference On Artificial Intelligence (IJCAI)*, 2001, Seattle, Washington, USA.

[38] H. Qi, X. Wang, S. S. Iyengar, et al, "Multisensor Data Fusion in Distributed Sensor Networks Using Mobile Agents", *In Proc. of the Intl. Conference on Information Fusion (FUSION)*, pp. 11 – 16, August 2001, Montreal, QC, Canada.

[39] D. B. Lange and M. Oshima, "Seven good reasons for mobile agents," *Commun. ACM*, Vol. 42, No. 3, pp. 88–89, 1999.

[40] D. Tian and N.D. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks", *In Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, pp. 32–41, September 2002, Atlanta, Georgia, USA.

[41] P. Berman, G. Calinescu, C.Shah, and A. Zelikovsly, “Efficient energy management in sensor networks”, *Ad Hoc and Sensor Networks*, Nova Science Publishers, 2005.

[42] K. Wu, Y. Gao, F. Li, and Y. Xiao, “Lightweight deployment-aware scheduling for wireless sensor networks”, *ACM/Kluwer Mobile Networks and Applications (MONET) – Special Issue on Energy Constraints and Lifetime Performance in Wireless Sensor Networks*, Vol. 10, No. 6, pp. 837-852, December 2005.

[43] J. K. Wu and Y. F. Wong, “Bayesian Approach for Data Fusion in Sensor Networks”, *In Proc. of the 9th International Conference on Information Fusion (FUSION)*, July 2006, Florence, Italy.

[44] Z. Khan, T. Balch, F. Dellaert, “MCMC-Based Particle Filtering for Tracking a Variable Number of Interacting Targets”, *IEEE Trans. PAMI*, Vol. 27, No.11, pp. 1805-1819, 2005.

[45] M.S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, “A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking”, *IEEE Trans. on Signal Processing*, Vol. 50, No. 2, pp 174-188, February 2002.

[46] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, “Integrated coverage and connectivity configuration in wireless sensor networks”, *In Proc. of the 1st International Conference on Embedded Networked Sensor Systems (SENSYS)*, pp 28-39, November 5-7, 2003, Los Angeles, CA, USA

[47] J. Yick, B. Mukherjee and D. Ghosal, “Analysis of a Prediction-based Mobility Adaptive Tracking Algorithm”, *In Proc. of the 2nd Intl. Conf. on Broadband Networks (BROADNETS)*, Vol. 1, pp.753 – 760, October 2005, Boston, Massachusetts, USA.

[48] Q. Wu, Nageswara S. V. Rao, J. Barhen, S. S. Iyengar, V. K. Vaishnavi, H. Qi, K. Chakrabarty, “On Computing Mobile Agent Routes for Data Fusion in Distributed

Sensor Networks”, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 6, pp.740-753, June, 2004.

[49] X. Sheng, Y. H. Hu, “Energy based source localization”, *In Proc. of the Intl. Conf. on Information Processing in Sensor Networks (IPSN)*, pp.285 – 300, 2003.

[50] X. Sheng, Y. H. Hu, “Maximum Likelihood Multiple Source Localization Using Acoustic Energy Measurements with Wireless Sensor Networks”, *IEEE Trans. on Signal Processing*, Vol. 53, No. 1, January 2005.

[51] J. Wang and N. Zabaras, “A Bayesian inference approach to the inverse heat conduction problem”, *International Journal of Heat and Mass Transfer*, Vol. 47, pp3927–3941, 2004.

[52] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, “The design of an acquisitional query processor for sensor networks”, *In Proc. of the ACM Intl. Conf. on Special Interest Group on Management of Data (ACM SIGMOD)*, June 9-12, 2003, San Diego, CA, USA.

[53] S. R. Madden, “The Design and Evaluation of a Query Processing Architecture for Sensor Networks”. Doctoral Thesis. UMI Order Number: AAI3183836., University of California at Berkeley, 2003.

[54] D. Tulone, S. R. Madden, “PAQ: Time Series Forecasting for Approximate Query Answering in Sensor Networks”, *Wireless Sensor Networks*, Springer, pp. 21 -37, January 2006.

[55] X. Zhang, X. Yu and X. Chen, “Inter-query data aggregation in wireless sensor networks”, *In Proc. of the Wireless Communications, Networking and Mobile Computing*, Vol.2 , pp. 930-933, September 2005.

[56] B. Krishnamachari, D. Estrin, and S. Wicker, “The Impact of Data Aggregation in Wireless Sensor Networks”, *In Proc. of 22nd International Conference on Distributed Computer Systems*, pp. 575-578, 2002.

[57] COUGAR: The network is the database. [Online]. Available: http://www.cs.cornell.edu/database/cougar/, 9 May 2009.

[58] TinyDB: A declarative database for sensor networks. [Online]. Available: http://telegraph.cs.berkeley.edu/tinydb, 9 May 2009.

[59] T. Abdelzaher, H. Tian and J. Stankovic, "Feedback Control of Data Aggregation in Sensor Networks", *In Proc. of the 43rd IEEE Conf. on Decision and Control (CDC)*, Vol.2, Iss. 14-17, pp.1490- 1495, December 2004, Atlantis, Paradise Island, Bahamas.

[60] S. R. Madden, M. Franklin, J. Hellerstein, W. Hong, "TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks", *In Proc. of the 5th Symposium on Operating System, Design and Implementation (OSDI)*, December 2002, Boston, Massachusetts, USA.

[61] TinyOS: Open Source Operating System for Wireless Embedded Sensor Networks: [Online]. Available: http://www.tinyos.net/, 9 May 2009.

[62] R. Jayanthi and S. Biswas, "Biologically Inspired Mobility Models for Energy Conservation in Sensor Networks", *The 2nd IEEE Workshop on Embedded Networked Sensors (EMNETS-II)*, pp. 133-140, May 30-31, 2005, Sydney, NSW, Australia.

[63] L. Hong, T.B. Koay, J.R. Potter and S.H. Ong, "Estimating Snapping Shrimp Noise in Warm Shallow Water", *In Oceanology International '99*, Singapore: Spearhead Exhibitions, 1999.

[64] X. Hong, M. Gerla, R. Bagrodia, J. K. Taek, P. Estabrook, G. Pei, "The Mars sensor network: efficient, energy aware communications", *Military Communications Conference (MILCOM). Communications for Network-Centric Operations: Creating the Information Force. IEEE*, Vol.1, pp.418- 422, 2001.

[65] X. Hong, M. Gerla, W. Hanbiao, and L. Clare, “Load Balanced, Energy-Aware Communications for Mars Sensor Networks”, *IEEE Aerospace Conference*, vol.3, pp.1109-1115, 2002.

[66] Crossbow Technology Inc. MICAz ZigBee (MPR2400) [Online]. Available: http://www.xbow.com/Products/productdetails.aspx?sid=101, 9 May 2009.

[67] L. Freitag, M. Grund, S. Singh, J. Partan, et al., “The WHOI Micro-modem: An Acoustic Communications and Navigation Systems for Multiple Platforms”, *MTS/IEEE Oceans Conference*, September 2005, Washington DC, USA.

[68] C. Park, J. Liu and Pai. H. Chou, “Eco: an Ultra-Compact Low-Power Wireless Sensor Node for Real-Time Motion Monitoring”, *In Proc. of the Intl. Conf. on Information Processing in Sensor Networks (IPSN)*, April 2005, Los Angeles, California, USA.

[69] N. K. Yagnamurthy and H. J. Jelinek, “A DSP based underwater communication solution”, *MTS/IEEE Oceans Conference*, Vol.1, pp.120-123, September 2003, San Diego, California, USA.

[70] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, et al., “Data Collection, Storage, and Retrieval with an Underwater Sensor Network”, *In Proc. of the 3rd ACM Conf. on Embedded Networked Sensor Systems (SenSys)* , 2005, San Diego, California, 2005.

[71] Y. Xu, J. Heidemann, and D. Estrin, “Geography-informed Energy Conservation for Ad Hoc Routing,” *In Proc. of the 7th Annual Intl. Conf. on Mobile Computing and Networking (MOBICOM)*, July 16-21, 2001, Rome, Italy.

[72] Y.F. Wong and W.C. Wong, “A Fuzzy-decision-based routing protocol for Mobile Ad hoc Networks”. *In Proc. of IEEE Intl. Conf. on Networks (ICON)*, pp. 317-322, 27-30 August 2002. Singapore.

[73] J. H. Chang and L. Tassiulas, “Maximum lifetime routing in wireless sensor networks”, *IEEE/ACM Transactions on Networking*, Vol. 12, No. 4, pp609-619, 2004.

[74] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “An Application-Specific Protocol Architecture for Wireless Microsensor Networks”, *IEEE Transactions on Wireless Communications*, Vol. 1, No. 4, pp. 660-670, October 2002.

[75] J. F. Scholl, L. P. Clare and J. R. Agre, “Seismic attenuation characterization using tracked vehicles”, Report, Rockwell Science Center Thousand Oaks, CA 91360, Aug 1999.

[76] Encyclopedia Britannica, 15th Edition, Vol. 25, pp818, 1991.

[77] T. Chu and I. Nikolaidis, “On the artifacts of random waypoint simulations*”, In Proc. of the 1st Intl. Workshop on Wired/Wireless Internet Communications (WWIC), in conjunctions with the Intl. Conf. on Internet Computing (IC)*, 2002.

[78] T. Camp, J. Boleng, and V. Davies, “A Survey of Mobility Models for Ad Hoc Network Research,” *Wireless Communications and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, Vol. 2, No. 5, pp. 483–502, 2002.

[79] V. Shnayder, M. Hempstead, B. Rong Chen, G. W. Allen and M. Welsh, “Simulating the power consumption of large-scale sensor network applications”, *In Proc. Of the 2nd Intl. Conf. on Embedded Networked Sensor Systems (SenSys)*, 2004.

[80] M. A. Sharaf, J. Beaver, A. Labrinidis and P. K. Chrysanthis, “TiNA: A Scheme for Temporal Coherency-Aware in-Network Aggregation”, *In Proc. Of the 3rd Intl. Workshop on Data Engineering for Wireless and Mobile Access*, pp. 69–76, 2003.

[81] C. Intanagonwiwat, D. Estrin, R. Govindan and J. Heidemann, “Impact of Network Density on Data Aggregation in Wireless Sensor Networks”, *In Proc. Of the 22nd Intl. Conf. on Distributed Computing Systems (ICDCS)*, pp. 457–458, Jul 2002.

[82] J. Gao, L. J. Guibas, J. Hershberger and N. Milosavljevic, “Sparse Data Aggregation in Sensor Networks”, *In Proc. Of the 6th Intl. Conf. on Information Processing in Sensor Networks (IPSN)*, 2007.

[83] Y. P. Chen, A. L. Liestman and J. Liu, “Energy-Efficient Data Aggregation Hierarchy for Wireless Sensor Networks”, *In Proc. Of the 2nd Intl. Conf. on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine)*, 2005.

[84] A. Deshpande, C. Guestrin and S. R. Madden, “Model-Driven Data Acquisition in Sensor Networks”, *In Proc. Of the 30th Intl. Conf. on Very Large Data Bases (VLDB)*, Aug 2004.

[85] C. Schurgers and M. B. Srivastava, “Energy efficient routing in wireless sensor networks”, *In MILCOM*, pp. 357–361, 2001.

[86] S. Coleri Ergen and P. Varaiya, “Energy Efficient Routing with Delay Guarantee for Sensor Networks”, *ACM Wireless Networks (WINET) Journal*, Jan 2006.

[87] D. Tian and N. D. Georganas, “Energy Efficient Routing with Guaranteed Delivery in Wireless Sensor Networks”, *In Proc. Of the Intl. Conf. on Wireless Communications and Networking (WCNC)*, Vol. 3, pp. 1923–1929, 2003.

[88] C. Intanagonwiwat, R. Govindan and D. Estrin, “Directed diffusion: A scalable and robust communication paradigm for sensor networks”, *In Proc. of the 6th Annual Intl. Conf. on Mobile Computing and Networking (MobiCOM '00)*, August 2000.

[89] V. Raghunathan, C. Schurgers, S. Park and S.B. Srivastava, “Energy Aware Wireless Sensor Networks”, IEEE Signal Processing Magazine, Vol. 19, No. 2, pp. 40 – 50, 2002.

Appendix A – Related Publications

Published Journal Papers

Y. F. Wong, T. Kok, L. H. Ngoh, W.C. Wong and W. Seah, "Waking up Sensor Networks", *Encyclopedia for Internet Technologies and Applications*, pp. 670 – 677, 2008.

Y. F. Wong, L. H. Ngoh, Winston K. G. Seah and W.C. Wong "Dual Wakeup Design for Wireless Sensor Networks", *Elsevier Computer Communications*, Vol. 32, No. 1, pp. 1 – 13, 23 Jan 2009.

Submitted Journal Papers

Y. F. Wong, L. H. Ngoh and W.C. Wong "Query-Enabled Sensor Networks using the Cyclic Symmetric Wakeup Design", *Springer Wireless Networks (WINET)*, 2009.

Published Conference Papers

Y. F. Wong, J. K. Wu, L. H. Ngoh, W. C. Wong. "Collaborative Data Fusion Tracking in Sensor Networks using Monte Carlo Methods," lcn, pp. 563-564, *29th Annual IEEE International Conference on Local Computer Networks – 1st IEEE Workshop on Embedded Networked Sensors (LCN'04 – EMNETS-I,),* November 2004, Tampa, Florida, USA.

Y. F. Wong, L. H. Ngoh and W. C. Wong, "An Adaptive Wakeup Scheme to Support Fast Routing in Sensor Networks", *2nd International ACM Performance Evaluation in Wireless Sensor and Ubiquitous Networks (PE-WASUN*), pp18-24, October 2005, Montreal, Canada.

Y. F. Wong, L. H. Ngoh, W.C. Wong and W. Seah, "Wakeup Scheme for Ocean Monitoring Underwater Sensor Networks (UWSN)", *MTS/IEEE OCEANS*, May 2006, Singapore.

J. K. Wu and Y. F. Wong, "Bayesian Approach for Data Fusion in Sensor Networks", *9th International Conference on Information Fusion (FUSION),* July 2006, Florence, Italy.

Y. F. Wong, L. H. Ngoh, W. C. Wong and W. K. G. Seah, "Intelligent Sensor Monitoring For Industrial Underwater Applications", *4th International IEEE Conference on Industrial Informatics (INDIN)*, August 2006, Singapore.

Y. F. Wong, Winston K. G. Seah, L. H. Ngoh, and W.C. Wong, "Sensor Traffic Patterns in Target Tracking Networks", *IEEE Wireless Communications & Networking Conference (WCNC)*, Hong Kong, March 2007.

Y. F. Wong, L. H. Ngoh, W.C. Wong and Winston K. G. Seah, "A Combinatorics-Based Wakeup Scheme For Target Tracking in Wireless Sensor Networks", *IEEE Wireless Communications & Networking Conference (WCNC)*, Hong Kong, March 2007

Y. F. Wong, L. H. Ngoh and W.C. Wong, "Energy-Efficient Time-Bounded Wireless Sensing Networks", *IEEE International Conference on Information, Communications and Signal Processing (ICICS),* 2007.

Appendix B – List of Definitions, Lemmas, Theorems, Corollaries and Properties

Definitions

Definition 2.4.1 A network of nodes is (n,T)-connected if there exists at least one path that connects any two nodes in the network within a time duration of T when (n-1) nodes (and their incident links) are removed.

Definition 2.4.2. Full connectivity is defined to be the maximum connectivity achievable when all nodes are awake.

Definition 3.1. An Agent is a piece of information, data or software code that uniquely identifies a target in the sensor network and moves within some distance of the target as it traverses the network.

Definition 3.1.1.1. An area A is (m,T)-covered if every point in A is always covered by the sensing coverage radii of at least m different sensor nodes within a time duration of T.

Definition 3.1.1.2. Full coverage is defined to be the maximum coverage achievable when all sensor nodes are awake.

Definition 3.1.2.1: The longest non-common wakeup time (LNWT) between any two schedules in a wakeup design is defined to be the longest time duration between any two nodes using schedules in a wakeup design such that both nodes are not awake at the same time.

Definition 4.1.1. Coarse Data (CD) users require coarse information about the environment of a region of interest and are satisfied with any one of the many possible responses for every non-overlapping time interval T_I that spans some desired time duration T_D.

Definition 4.1.2. All Data (AD) users require detailed information about the environment of a region of interest and can only be satisfied with all possible responses for every non-overlapping time interval T_I that spans some desired time duration T_D.

Definition 4.1.3. A query of length T_Q is defined to be the duration of a set of past measurements over which is of interest to the user.

Definition 5.3.1. The "Lonely Node Problem" is the phenomenon when nodes wakeup to find no other neighbour nodes within its communication range.

Lemmas

Lemma 1. Let T_{awake} be the longest duration of continuous active slots in a cyclic symmetric (k^2+k+1, k+1, 1) schedule. Then, $T_{awake} = 2T_{slot}$.

Lemma 2. There exists only one T_{awake} in any cyclic symmetric (k^2+k+1, k+1, 1) design.

Lemma 3. There are exactly one duration of continuous active slots of length $2T_{slot}$ and exactly (k-1) active slots of length T_{slot} in any cyclic symmetric (k^2+k+1, k+1, 1) design.

Lemma 4. The length of any durations of continuous sleep slots from a selected schedule in a cyclic symmetric (k^2+k+1, k+1, 1) design is unique within that schedule.

Lemma 5. Let T_{sleep} be the longest duration of continuous sleep slots in any cyclic symmetric (k^2+k+1, k+1, 1) design. Then, T_{sleep} is upper bounded by

$$T_{sleep,\max} = \frac{1}{2}k(k+1)T_{slot}.$$

Lemma 6. Consider any T_{sleep} duration in any schedule from a cyclic symmetric (k^2+k+1, k+1, 1) design. All other schedules in the design (other than the schedule under consideration) have at least one wakeup active slot during T_{sleep}.

Lemma 7. All schedules from the cyclic symmetric (k^2+k+1, k+1, 1) design have at least one awake slot within a time duration of $\frac{1}{2}(k^2+k+2)T_{slot} \approx \frac{1}{2}T_{cycle}$.

Theorems

Theorem 2.4.1. The network N_G is (α,$N_{hop}T_{cycle}$)-connected where T_{cycle} = (k^2+k+1)T_{slot} and N_{hop} is the maximum number of hops between any two nodes in the network dictated by the routing algorithm.

Theorem 2.4.2. Consider any two neighbour nodes X and Y in the network operating schedules S_X and S_Y from the same cyclic symmetric (k^2+k+1,k+1,1) design. Nodes X and Y can always discover each other within bounded time for any arbitrary time offset of the schedule S_Y from S_X, or vice versa.

Theorem 3.1.1.1. Region A is $\left(\beta, \frac{1}{2}(k^2 + k + 2)T_{slot}\right)$*-covered.*

Theorem 3.1.1.2. A β-covered network implies a β-connected network, if $R_C \geq 2R_S$

Theorem 3.1.1.3. For a cyclic symmetric (k2+k+1,k+1,1) design wakeup network that is β-covered when all nodes are awake, it is also:
$\left(\beta, [(k^2 + k + 2)T_{slot}]/2\right)$ *-covered, and* $\left(\beta, N_{hop}(k^2 + k + 1)T_{slot}\right)$ *-connected if* $R_C \geq 2R_S$.

Theorem 3.1.2.1. Assume that sensor nodes operate wakeup schedules from the same cyclic symmetric (k^2+k+1,k+1,1) design. Data packets from one node to the next hop wait at most $T_{cyclic,sleep} = k(k+1)T_{slot}$ *where* T_{slot} *is the slot time.*

Theorem 4.1.1. A cyclic symmetric (k^2+k+1,k+1,1) design guarantees the existence of a zero Query Waiting Time (QWD) for a CD user at any arbitrary time slot.

Theorem 4.1.2. For a cyclic symmetric (k^2+k+1,k+1,1) design, an upper bound delay on a query of length $T_Q \leq 2T_{slot}$ *for a CD user is given by* $\frac{1}{2}k(k+1)\,T_{slot}$ *for minimized energy consumption.*

Theorem 4.1.3. For queries of length $T_Q \leq 2T_{slot}$*, using a cyclic (k^2+k+1,k+1,1) design, the energy required to reply to a CD user query is bounded by [E, 2E].*

Theorem 4.1.4. For a cyclic (k^2+k+1,k+1,1) design and assuming AD users, the delay of a query on all collected sensor readings in the region is upper bounded by $D_{AD} = \frac{1}{2}(k^2 + k + 2)T_{slot} \approx \frac{1}{2}T_{cycle}$.

Theorem 4.1.5. For a cyclic (k^2+k+1,k+1,1) design, the energy expenditure per query request is (k^2+k+1)E.

Theorem 5.6.1. Region A is $\left(\beta, (k^2 + 1)T_{slot}\right)$*-covered for AWSF networks.*

Corollaries

Corollary 5.1. $T_{sleep,\max} < \frac{1}{2} T_{cycle}$.

Corollary 3.1.1.1. For a cyclic symmetric (k^2+k+1,k+1,1) design wakeup network, it takes approximately 2N_{hop} times longer to guarantee full connectivity than to guarantee full coverage.

Corollary 3.1.1.2. For a cyclic symmetric (k^2+k+1,k+1,1) design wakeup network, it takes approximately twice as long to guarantee information propagation across one network hop than to guarantee full coverage.

Properties

Property 5.7.1. Nodes operating AWSF always wake up to find at least one neighbour to communicate.

Property 5.7.2. AWSF is delay upper-bounded.

Appendix C – List of Common Acronyms

AD	All Data (user class)
AREQ	“Awake REQuest” Packets
ASCENT	Adaptive Self-Configuring sEnsor Networks Topologies
AUV	Autonomous Underwater Vehicles
AWSF	Adaptive Wakeup Schedule Function
BRS	Basic Reconstruction Scheme
BTC	Bounded-Time Connectivity/Coverage
CC	Command Center
CD	Coarse Data (user class)
DD	Directed Diffusion (paradigm)
CSBD	Cyclic Symmetric Block Design
CSCM	Cross-Sensor Cross-Modality (data fusion algorithm)
DWSF	Database Wakeup Schedule Function
ESAP	Energy-Aware Swap Protocol
LCT	Loss of Continuity in Tracking
LNWS	Longest Non-common Wake Slots
LNWT	Longest Non-common Wake Time
MANET	Mobile Ad-hoc NETwork
NAS	New “Awake” Slots
ODND	On-Demand Neighbour Discovery
PAMAS	Power Aware Multi-Access protocol with Signalling
PEAS	Probing Environment and Adaptive Sleeping
PECAS	Probing Environment and Collaborative Adaptive Sleeping
PFN	Percentage of False Negatives
QWD	Query Waiting Delay
RAW	Random Asynchronous Wakeup
RIS	Random Independent Scheduling
STEM	Sparse Topology and Energy Management
TAG	Tiny Aggregation (service)
TiNA	Temporal coherency-aware In-Network Aggregation
TIP	Target Identification Packets
TTA	Target Tracking Accuracy
TTP	Target Tracking Packets

TWSF	Tracking Wakeup Schedule Function
UWSN	UnderWater Sensor Network
WSF	Wakeup Schedule Function